Continuous Data Home

Continuous Data

Exported on 08/25/2023

Table of Contents

Welcome to the Delphix Continuous Data documentation!

Here you will find all the information you will need to use the Delphix Continuous Data Engine for data virtualization. Learn how to deploy our application, use our features, or tune it for optimal performance. We have organized this content into several categories that you can browse via the navigation bar.

The following is a list of the PDF copy of all the Continuous Data documentation versions:

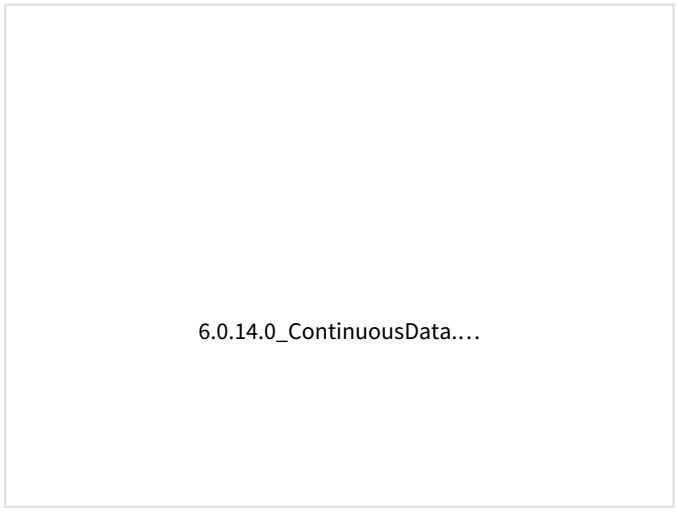
Versions in primary support:

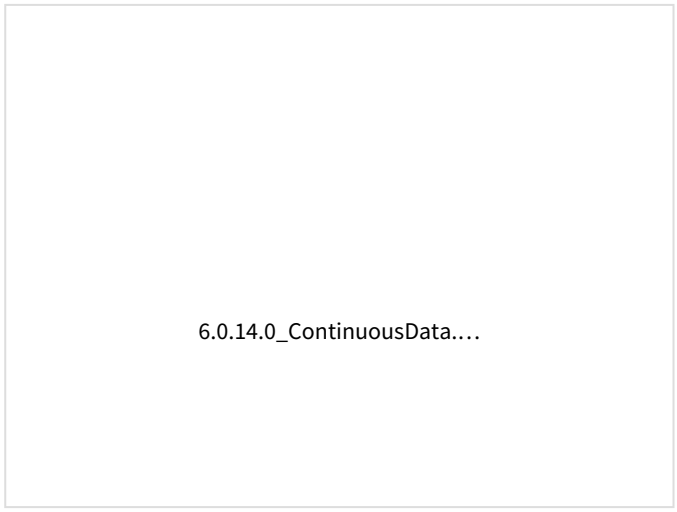
- 13.0.0.0_ContinuousData.…
- 12.0.0.0_ContinuousData.…

- 11.0.0.0_ContinuousData.…
- 10.0.0.0_ContinuousData.…
- 9.0.0.0_ContinuousData.p…

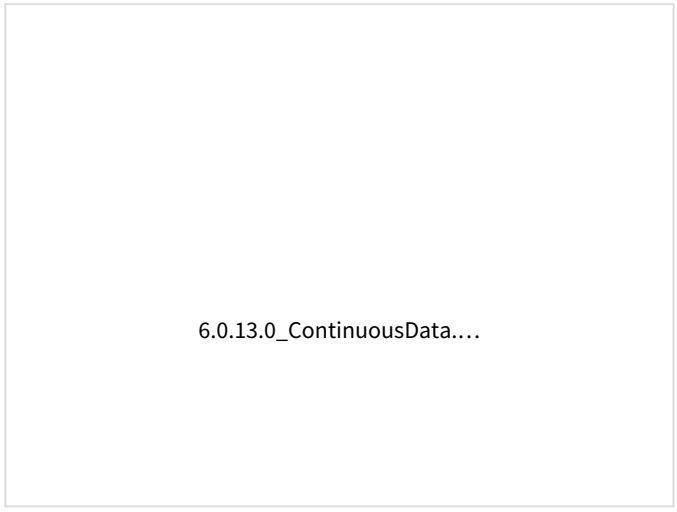
- 8.0.0.0_ContinuousData.p…
- 7.0.0.0_ContinuousData.p…
- 6.0.17.2_ContinuousData.…

- 6.0.17.0_ContinuousData.…
- 6.0.16.0_ContinuousData.…
- 6.0.15.0_ContinuousData.…

- 

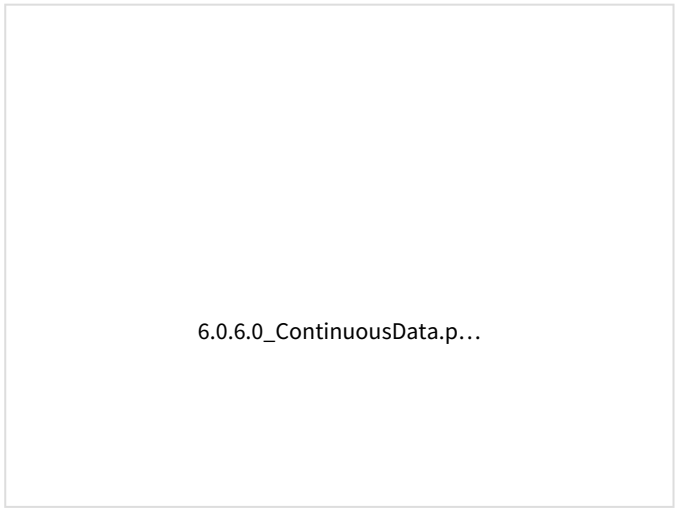
- 

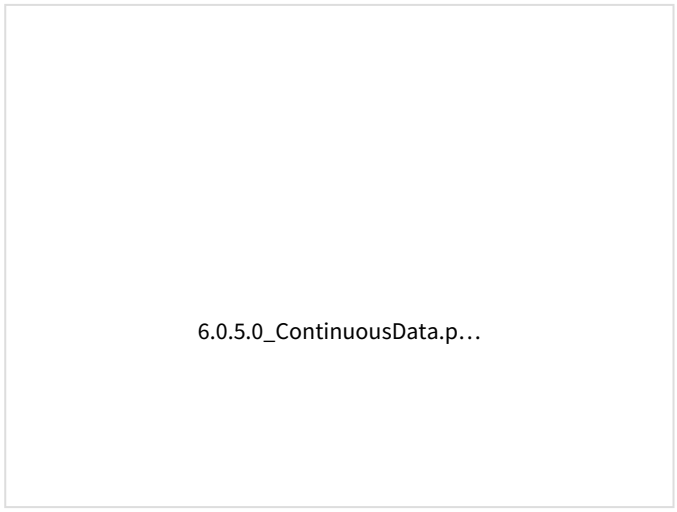
Versions in extended support:

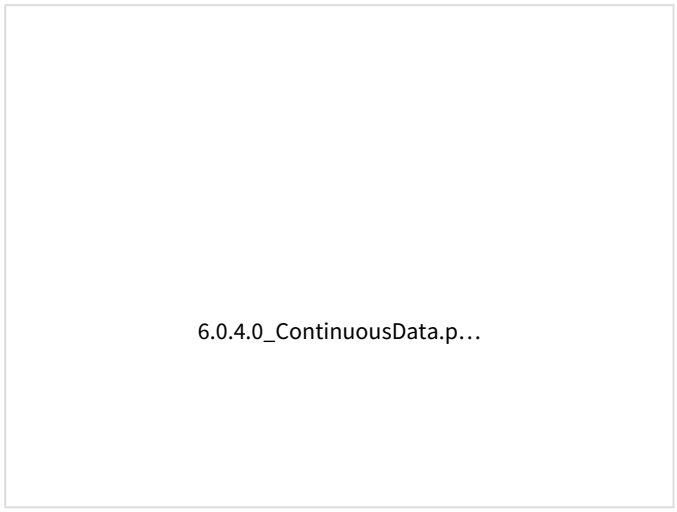
- 

- 6.0.12.0_ContinuousData.…
- 6.0.11.0_ContinuousData.…
- 6.0.10.0_ContinuousData.…

- 6.0.9.0_ContinuousData.p…
- 6.0.8.0_ContinuousData.p…
- 6.0.7.0_ContinuousData.p…

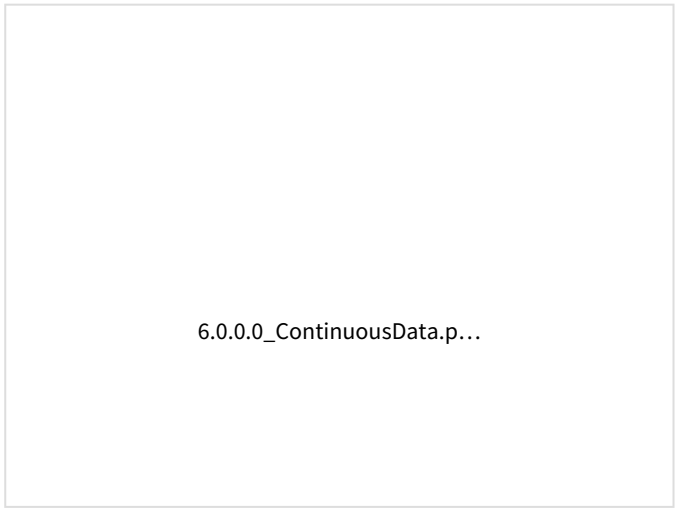
- 

- 

- 

Versions in legacy support:

- 6.0.3.0_ContinuousData.p…
- 6.0.2.0_ContinuousData.p…
- 6.0.1.0_ContinuousData.p…

- 
6.0.0.0_ContinuousData.p…

Release notes

This section covers the following topics:

New features

Release 14.0.0.0

- **Elastic Data**
  Previously referred to as “Cloud Engines”, Continuous Data with Elastic Data allows you to leverage lower-cost object storage in addition to traditional block storage. This has the effect of dramatically decreasing the operational overhead of Continuous Data, while enabling new use cases like long-term archival and retention.
    - **Private data center Elastic Data**
      Previously, Elastic Data was only available for Continuous Data Engines deployed in AWS (using S3) or Azure (using BLOB storage). We now support deploying Elastic Data with on-premises, S3-compatible object storage arrays.
    - **Elastic Data on Oracle Cloud Infrastructure**
      Elastic Data may now be used in OCI, providing decreased operational overhead and enabling new use cases, as mentioned above.
- **Replication Failback**
  In the case of a failure on a primary Continuous Data Engine, failover may be used to swap operations to a secondary Engine. Previously, this was a one-time, terminal action. With failback, you can restore operations back to the primary engine if the failure has been resolved, or if you simply want to test the failover process.
- **PurgeLogs operation**
  The PurgeLogs operation now includes support for Oracle Multitenant dSources.
- **Oracle Staging Push**
  The Staging Push method of ingestion now supports point-in-time provisioning of data.
- **Couchbase**
  Couchbase 7.1.3 is now supported and several bugs have been resolved, such as increasing buffer size in the bucket-list command, backup restore failures, and replication status checks.
- **IBM Db2**
  SSL/TLS connections are now supported with HADR dSources.
- **MySQL/Linux**
  There is now guidance for upgrading Source, Staging, and Target environments from MySQL 5.7 to v8.0. In addition, new guardrails have been introduced to prevent incompatible refresh and rollback operations.
- **PostgreSQL**
  All RHEL v8.x operating systems are now supported. In addition, new protections have been introduced to prevent accidental modification of parameters via VDB Config Template configuration.
- **SAP HANA**
  VDB provisioning has been improved in scenarios when SAP HANA services `scriptserver` and others are missing volume information.
- **Support for upgrading OpenJDK Java**
  Enables the ability to provide your own Adoptium OpenJDK upgrades without upgrading the product. Extends “Bring Your Own Java” to include Adoptium, not just Oracle.
- **ESXi 8.0 U1**
  Continuous Data may now be run on VMware ESXi 8.0 U1.

Release 13.0.0.0

- **Recovery from failed VPDB in a Linked CDB**
  A variety of factors may cause an Oracle virtual pluggable database to fail in a linked container database. It is now possible to recover from a failed Pluggable Virtual Database with a forced refresh or rewind operation.

- **IBM Db2**
  Support for RHEL 8.8 has now been certified for Delphix. This release was originally published in June.
- **MongoDB**
  Additional parameters are now supported in the `mongodump` and `mongorestore` commands. Backups and restores can be customized more flexibly to improve data extraction and ingestion speeds, such as parallel collections processing.
- **PostgreSQL**
  Support has been added for PostgreSQL 15.0 on RHEL 9.0 and Azure Flexible Server for PostgreSQL 14.0. The source sizing calculation has also been updated to more accurately represent the correct database size.
- **SQL Server**
  The following are now supported:
  - SQL Server 2016 on Windows Server Core 2016
  - SQL Server 2019 on Windows Server Core 2019
  - SQL Server 2022 on Windows Server Core 2022
  - SQL Server Developer Edition for all supported SQL Server versions

Release 12.0.0.0

- **User audit for Delphix support**
  In a case where engine access is granted to Delphix Support to debug or fix issues, or any actions take any actions, this is now tracked in its own audit log.
- **SMTP TLS**
  You can now add a certification for SMTP TLS authentication via the API or command line.
- **Refresh/Rewind of Oracle VDBs and VPDBs that have been exported to Oracle ASM Databases**
  Users may now refresh or rewind Oracle VDBs or VPDBs that have been exported to physical ASM databases. The performance of the export to ASM can also be improved with the ability to specify specific RMAN file section sizes.
- **IBM Db2**
  The v4.6.0 release supports Staging Push with the Database Partitioning feature (DPF) and custom mount points.
- **MongoDB**
  The v1.3.0 release supports passing additional parameters to mongodump and mongorestore. This adds parallel processing and flexibility to improve data extraction and ingestion speeds.

Release 11.0.0.0

- **Staging push read-only Oracle databases**
  Supports staging-push ingestion from read-only databases.
- **Partial ingestion via Oracle staging push**
  Partial ingestion enables the ability to ingest only selected Oracle tablespaces with Staging Push. This feature allows you to subset the Oracle database by choosing only specific tablespaces.
- **MongoDB**
  Supports MongoDB Enterprise 6.0 and Mongo Atlas, including cluster-to-cluster sync to enable ingestion of large, sharded databases.
- **MySQL/Linux**
  Supports MySQL v8 across all certified editions.
- **PostgreSQL**
  Supports point-in-time (PIT) provisioning using external logs and ingestion using the “pg_dumpall” utility.

Release 10.0.0.0

- **Azure key vault integration**
  This release adds Azure Key Vault support for storing and accessing secrets, keys, and certificates necessary for Continuous Data operations.

Release 9.0.0.0

- **READ_COMMITTED_SNAPSHOT parameter can now be set during VDB provisioning**
  This feature allows the SQL Server *READ_COMMITTED_SNAPSHOT* database parameter to be defined during VDB Provision operations, either by specifying it as a VDB Configuration Parameter or associating the VDB with a VDB Configuration Template. If defined, the parameter will be automatically set during Provision, Refresh, and Rewind operations for the VDB.
- **Export an Oracle virtual DB or virtual PDB to a physical Oracle ASM or Exadata database**
  This feature enables you to export data from an Oracle Virtual database (VDB) or a Multi-Tenant (MT) VPDB to a physical Oracle database that uses Automatic Storage Management (ASM). This feature is especially useful for Oracle target environments running on Exadata or ExaCC systems.

Release 8.0.0.0

- **Oracle: Multiple virtual PDBs in a virtual CDB**
  This feature enables you to provision and manage multiple Oracle virtual PDBs to a virtual CDB (vCDB).
  - A vCDB is a Delphix-managed, fully compatible Oracle Container Database created and maintained by Delphix as part of the VDB provisioning workflow. This workflow allows organizations to automate their Oracle data lifecycle end-to-end, which results in developer productivity enhancements. Additionally, because container databases allow it, Oracle users benefit from the Oracle licensing agreement (which allows hosting up to three PDBs inside a vCDB).

This feature is only supported for Oracle versions 12.1.0.2 (with a patch for Oracle bug 18967466) and later.

- **Source sizing for HANA**
  This feature enables you to calculate the source dataset size based on the total allocated space on the mount point provided by the Delphix Engine for the dataset.
- **Support for multiple backup paths in DB2**
  This feature supports multiple backup paths instead of one for dSource ingestion. This allows you to provide a single backup path for the dSource ingestion and add more backup paths with a dynamic UI to accommodate for the extra backup directories where the backup was split into.

Release 7.0.0.0

- **Oracle: Multi-Tenant TDE for VCDBs**
  Use transparent data encryption (TDE) with virtual container databases to secure data-at-rest. Existing TDE keys can be used or data can be rekeyed upon deployment. Automated KeyStore sanitization ensures production TDE keys are not shared to non-production environments.
- **Oracle: staging push**
  Pushes new data to the Delphix DevOps Data Platform instead of requesting data on a polling interval. Isolate ingests to Delphix from production instances. Gives the ability to leverage third-party backup applications. Full-tested integration with Oracle Data Guard and Active Data Guard allows for staging DB to be the standby DB, enabling tight point-in-time snapshots.
- **Cassandra: new Continuous Data connector**
  Continuously sync to Cassandra databases and deploy data to non-production environments. Deploy data

from multiple Cassandra nodes as a single virtual database. Track source changes along a continuous time flow and redeploys data from any time. Quickly deploy data to new ephemeral instances. Branch production data, apply data masking, and deploy to new developer sandbox environments.

⚠ Multiple Device Removal for Delphix Engines version 6.0.12.0 to 6.0.17.X contains a breaking kernel module change that requires a reboot in order for the new module to load. With that, a deferred reboot engine upgrade operation will be unable to remove devices until a reboot is performed.

Release 6.0.17.0

- **Source sizing for Postgres and IBM Db2**
  This feature brings improved visibility for data source ingest management and metrics with data source size information logs. The source volume sizes can now be viewed through engine APIs and easily integrate with automation scripts.
- **Environment variables in hooks**
  This feature allows users to customize data deployments with hook scripts using environment variables, defined by plugin connectors in DLPX_DATA_DIRECTORY. Automated hook scripts can be executed during pre and post-deployment.

Release 6.0.16.0

- **Elastic Data engines on Microsoft Azure**
  You can now use Azure Blob object storage in place of block storage to reduce operational costs. Cache only necessary data to engines in order to maintain consistent VDB performance. Allows for elastic expansion of your storage footprint as-you-go in the cloud. Quickly deploy data to engines across regions, combining these tools to achieve potential cost savings of 25%-80% depending on use and workloads.
- **Elastic Data engines cache resize**
  This feature allows your Elastic Data engine cache to meet changing cost-performance demands on-the-fly by fine-tuning cache ratios for data ingestion and distribution use cases. Adapt to different workloads, saving costs during idle situations and delivering high performance during heavy workloads.
- **Continuous Vault: additional security controls**
  New security controls are introduced for Ransomware Protection when using Delphix Continuous Vault. Alert Profiles, dSources, and LogSync profiles can now be locked on Continuous Data Source Engines. Delphix Continuous Ransomware Protection Solution provides Continuous Data Protection, Recovery, Detection, and Compliance.
- **Non-admin user password reset**
  Non-admin users will be able to request a password reset if an email is associated with the account. The password reset will continue to be controlled through LDAP/SSO for users on IdP platforms. Self-service password reset is available both within the UI and CLI.
- **Zero trust update for SAP HANA connector**
  A “least-privileged” OS user can now be used to run virtualization operations on a Delphix target host. Aligning with Zero Trust initiatives, Delphix no longer requires a high-level OS user for virtualization.
- **Increased mount security**
  This change brings improved security controls that limit connections to Delphix Engines only from a defined host list. NFS mount checks will now run on ephemeral mount points matched to specific host UIDs. Mount checks are used to verify that the Continuous Data Engine is mountable before running virtualization operations.
- **iSCSI parameter warnings**
  New warning to ensure iSCSI parameters on Windows Targets are optimized for Delphix Engines. The check runs for Target environments during Add, Refresh, and Enable operations.
- **Database port customization for EBS plugin**
  This feature supports the provisioning of VDBs in EBS dbTechStack & AppsTier plugins to run on a

customized port. You can now provision database binaries(ORACLE_HOME) cloning and listener process to run on a custom DB port and Application tier to connect to that port.

Release 6.0.15.0

- **Oracle MT TDE for system tablespaces**
  The next release enables system tablespaces on Oracle Virtual Databases to be encrypted using transparent data encryption (TDE). Oracle system tables store database table information, index, sequences, and other objects. Encrypting system tables in non-prod environments enforce security controls and prevents database index and metadata tampering.
- **Staging push for Postgres**
  Staging push enables organizations to push data into Delphix without having to directly query production instances. With staging push, teams can use 3rd party solutions to recover data to staging instances that automatically sync to Continuous Data. This also enables logical replication workflows (common for PaaS sources) and the use of tools like pgbackrest.
- **Staging Push for IBM Db2 HADR**
  Staging push will be added in the next release for IBM Db2 in a High Availability Disaster Recovery (HADR) configuration. This gives teams an alternative way to sync data into the Continous Data.
- **Cache analysis for Elastic Data engines**
  We plan to add a UI for teams to better understand the effectiveness of allotted Delphix Elastic Data engine cache size. This enables users to optimize DCE cache sizes to meet required cost-performance levels for VDBs.
- **Storage device removal UI**
  The team will be able to remove disks on engines backed by block storage through the UI in addition to the CLI in 6.0.15.
- **Zero trust for Postgres**
  6.0.15 supports custom “least privilege” OS user roles to be assigned to the “Delphix_OS” user to perform virtualization operations from Postgres sources.
- **TDE source database support for EBS plugin**
  6.0.15 supports TDE source databases for the EBS plugin.

Certifications

- Oracle 19c on RHEL 8.6 (6.0.15.0+)
- Oracle 21c on RHEL 8.6 (6.0.15.0+)
- AppData on RHEL 8.6 (6.0.15.0+)

Release 6.0.14.0

- The **Delphix DevOps data platform** will feature some product name changes to better align with our offered solutions:
  - **Continuous Data** (Virtualization)
  - **Continuous Compliance** (Masking)
  - **Continuous Vault** (Data vault)
- **Single engine Continuous Vault**
  The Single Engine Continuous Vault provides effective protection against ransomware attacks in a standalone Delphix Engine. This option may be preferable for deployments where maintaining two separate engines is not architecturally necessary. See Delphix Continuous Vault for more information.
- **AWS Elastic Data engines**
  This offering will support the use of AWS S3 object storage in place of traditional block storage, making it easier and more efficient to store more data for longer with Delphix. For this release, only engines deployed on AWS are supported, with support for other infrastructures to come.
- **Microsoft SQL Server Virtual-to-Physical (V2P) improvements**
  With these new improvements, users can specify a separate drive for SQL Server transaction log files, since

these often need to split across multiple locations due to size. This ensures that DB files rehydrate in the manner that is most efficient during initial V2P, which helps avoid unnecessary downtime; this works in tandem with our ransomware solution. An additional enhancement to improve transfer performance involves replacing XCopy with Robocopy multithreading capabilities.

- **Oracle MT TDE support for encrypted system tablespaces**
  Every Oracle database has a system tablespace that is always used to store system data, including information about database tables, indexes, sequences, and other objects. Oracle TDE can encrypt, in addition to the data itself, the system tablespace for maximum security. This feature allows Delphix to provision VDBs with TDE-enabled encrypted system tablespaces to maintain the overall database security in non-production environments.
- **Staging push for SAP HANA via CommVault**
  CommVault is the most popular backup application for SAP HANA. We have certified that the backup application can be used as expected by our customers with our new Staging Push capability.
- **Salesforce data protect and version performance improvements**
  The March release will allow the user to configure up to 30 parallel upload threads (a 10x increase), improving our restore performance to ensure a fast recovery. We've also added several improvements towards faster and leaner retries for any records that fail during restore, plus increased the efficiency and reuse of API calls while downloading data (leading to faster backups and better resource utilization).
- **Ingest and restore for PostgreSQL**
  This feature allows you to ingest logical backups (pg_dump/pg_restore) from PostgreSQL, which enables workflows for direct ingestion from PaaS data sources (AWS Aurora & RDS), deployment to IaaS-hosted targets, and selective ingestion of individual databases from multiple database instances of PostgreSQL.

Certifications

- Oracle 19c on RHEL 8.5 (6.0.14.0+)
- Oracle 21c on RHEL 8.5 (6.0.14.0+)
- AppData on RHEL 8.5 (6.0.14.0+)

Release 6.0.13.0

New in this release

- **Support for AWS object storage**
  This release adds support for the use of AWS S3 object storage in place of traditional block storage. This release currently supports only those engines that are deployed on AWS.
- **Staging push for SQL Server**
  This release supports staging push for SQL Server. Staging push allows you to push data to Delphix if you have tools, configurations, or security concerns that do not work with the existing pull model. Staging push enables, for example, compatibility with data stored in backup systems and gives you control over the staging database. For more information, see Staging Push Implementation for SQL Server.
- **Support for TLS 1.3**
  This release provides support for TLS 1.3 for connections from the virtualization engine to the host and engine-to-engine communication.
- **Auto vPDB restart for single tenant linked CDBs**
  This release supports the auto-restart feature for single-tenant vPDBs in a linked CDB. The auto-restart feature allows the virtualization engine to detect and restart VDBs if the remote host has been restarted.
- **Transparent Data Encryption (TDE) for Oracle Multitenant RAC (Oracle 18c and 19c)**
  This release adds support for provisioning a TDE-encrypted vPDB to a linked (Physical CDB) target RAC environment on Oracle 12.2.
- **FluentD monitoring**
  FluentD is a popular open-source data logging layer for DevOps and Site Reliability Engineering (SRE) teams to stream telemetry to their chosen monitoring solution to improve systems observability. Delphix

already provides the ability to forward data events and performance metrics to Splunk. Delphix 6.0.13 enables additional Fluentd configurations to be developed and added to the engine. extending telemetry output to new tools like ELK, New Relic, and DataDog.
- **Db2 plugin version 4.1.3**
  This plugin release resolves some issues that are listed in the Db2 Release Notes.
- **EBS plugin version 3.0.3**
  This plugin release resolves some issues that are listed in the EBS Release Notes.

Certifications

- **Virtualization**
  - Microsoft Windows Server 2022
  - Oracle E-Business Suite (EBS) 12.2.11 in OCI
- **Hypervisors/Clouds**
  - ESXi 7.0 U3c

Release 6.0.12.0

New in this release

- **HANA staging push**
  The HANA 2.0 plugin introduces the Staging Push feature. This feature provides a new data ingestion mechanism that helps users to push data into the Delphix-provided mount point on their own. For more information, see Delphix Architecture for HANA.
- **Multiple device removal**
  You can now remove multiple storage devices at a time when engines are over-provisioned or when moving to new storage. This is done only after ensuring sufficient resources to support the removal.
- **Transparent Data Encryption (TDE) for Oracle Multitenant RAC (Oracle 12.2 and 21c)**
  The Oracle TDE feature encrypts the sensitive data (database tables and tablespaces) stored on the disk. This prevents misuse of the data if the disks or storage mediums are lost or stolen. The data is transparently decrypted for authorized users when they access the data. Our large enterprise customers leverage Oracle RAC configurations for their business-critical applications. We have added support for provisioning a TDE-encrypted vPDB to a linked (Physical CDB) target RAC environment on Oracle 18c and 19c in this release. For more information, see Provisioning a TDE-enabled vPDB to a Cluster Target.

Certifications

- Oracle 21c on SLES 15 SP3 (6.0.11.0+)
- Oracle 19c on SLES 15 SP3 (6.0.11.0+)
- Oracle 21c on SLES 15 SP2 (6.0.11.0+)
- Oracle 19c on SLES 15 SP2 (6.0.11.0+)
- AppData on RHEL 8.4 (6.0.11.0+)

Release 6.0.11.0

New in this release

- **OAuth2 API support**
  The Virtualization and Masking engine APIs are now accessible via OAuth2 tokens that improve Delphix's security offerings. For more information, see Configuring OAuth2 Authentication for API Access.

- **Oracle support for Exadata, Exadata Cloud, or Exadata Cloud-at-Customer (ExaCC) cluster**
  This release adds support for Exadata, Exadata Cloud, or Exadata Cloud-at-Customer (ExaCC) Cluster for Oracle databases. For more information, see Oracle Support Matrix.
- **Apply Hotfixes using the self-service UI**
  You can now apply hotfixes using the self-service upgrade UIs.
- **TDE for Oracle Multitenant - support for rekey**
  Oracle Advanced Security Transparent Data Encryption (TDE) provides the ability to create virtual pluggable databases with a new key (independent from the source database). This is to facilitate another layer of security - ensuring that different keys are used in the production and non-production systems. For more information, see Provisioning a TDE-enabled vPDB.
- **Modify DBID for non-multitenant Oracle VDBs after provisioning**
  You can now generate a new DBID after a VDB is provisioned and refreshed. For more information, see Generate a New DBID for Oracle VDBs. Support for multitenant databases is currently not available.
- **New wizard for creating replication profile**
  A new multi-step wizard is now available that replaces the previous in-place editing option to manage Replication Profiles. For more information, see Replication User Interface.

Certifications

- Oracle 21c on RHEL 8.4, RHEL 8.3, SLES 15 SP1 (6.0.11.0+)
- Oracle 19c on RHEL 8.4 (6.0.10.0+)
- Exadata Cloud-at-Customer(ExaCC)/Exadata support for Oracle 12.2 on OEL 7.8 and OEL 7.9

Release 6.0.10.0

Virtualization

- **SQL Server-support for Azure storage backups**
  SQL native backups can now be read directly from Azure Cloud storage. For more information, see Restoring SQL Backups Stored in Azure Cloud Storage.
- **Support dSource upgrades from Non-MT to MT in Oracle**
  Once a dSource is converted to a multitenant PDB, you will be able to share its storage blocks with its non-multitenant predecessor. Delphix will only store the incremental changes to the database. For more information, see Prepare and Upgrade a Non-MT Oracle dSource to MT.
- **Oracle-provision to the latest Point-In-Time on RAC**
  Provision to the latest point-in-time is now supported.
- **Flexible ORACLE_HOME permissions configuration**
  Removed the need to set permissions of the "$ORACLE_HOME/dbs" subdirectory using STARTUP SPFILE syntax to simplify Oracle operations. For more information, see Requirements for Oracle Hosts and Databases.
- **Support for manually starting an Oracle VDB**
  An Oracle VDB can now be manually started. For more information, see Manually Starting a VDB.
- **Attach and detach Oracle CDB containers**
  Detaching, attaching, and linking of the Oracle CDB containers is now supported via CLI. For more information, see CLI Cookbook: Attaching, Detaching, or Linking a CDB.
- **Password vaults and remote hooks for UI**
  In the 6.0.9.0 release, we introduced the ability to use password vaults with hooks. This allows our customers to ensure a high level of security with all operations with external systems. This can now be configured via the user interface. For more information, see Passing Credentials Securely to Hook Operations.
- **NFSv4**
  NFSv4 is now enabled by default. For more information, see NFSv4 Configuration.

- **12-month support for upgrades and Forward Compatible Replication (FCR)**
  To better align with the Delphix support program, Engine upgrades, and FCR operations must be within versions no more than 12 months apart. For example, upgrading to version 6.0.10.0 will require the previous version to be at least 6.0.4.0. For more information, see Upgrade Matrix and Replication Overview.
- **HANA plugin port control**
  You can now keep the port numbers consistent throughout the HANA VDB life cycle so that the connections made to the VDBs are not disrupted during their life cycle. For more information, see Provisioning HANA VDBs: An Overview.
- **User-specified mount path For Db2 dSource**
  You can now specify a mount path of your choice to host the dSource dataset on the target host. For more information, see Linking a Db2 dSource.
- **Staging push automation for Db2 dSource**
  You can now use a set of scripts that can be used to automate the restore and roll forward operations on the dSource.

Certifications

Virtualization

- **OCI VM.Standard.E4**
- **AWS r5n.24xlarge**
- **Oracle 21C RH 8.3**

Release 6.0.9.0

Virtualization

- **Delphix data vault - additions**
  In continuation to the **Delphix data vault for ransomware protection** released in 6.0.8.0, this release enables you to manage this feature from the Delphix Engine user interface and also lets you monitor the regular valid replication received. For more information, see Delphix Data Vault.
- **TLS 1.3**
  The TLS 1.3 support is added as an available secure connection option at the Engine Admin Console to be used between engines.
- **Phonehome data collection frequency**
  The default collection period for the Phonehome users is now changed to daily. For more information, see System Configurations - Enable Phone Home.
- **Password vaults for remote hooks**
  With this release, the hooks running on environments can now obtain credentials from the engine and its configured password vaults. These credentials can be used to perform custom authentication tasks in a secure manner. Currently, this feature is supported via the command line interface only. For more information, see Passing Credentials Securely to Hook Operations.
- **Delphix integrations (dxi) docker image**
  A new dockerized version of the dxi library is now available. For more information, see Docker Image.

Certifications

- **Virtualization**
  - EBS 12.2 on Oracle 19c
  - Db2 11.1.4.6 Fix pack
- **Hypervisors/Clouds**
  - ESXi 7.0 U2

Release 6.0.8.0

Virtualization

- **Delphix data vault**
  The Delphix Data Vault for ransomware protection (accessible via CLI) enables organizations to recover access to their application data much faster than traditional backup solutions after malicious attacks. It relies on the new **data vault** Replication feature, which replicates critical business DB data stored on Delphix engines to a new target engine called Data Vault. Once securely stored on the Data Vault, the replicated DB data can be used to recover business applications upon a ransomware attack with very low RTO and RPO.
- **Dxi executable and support for encrypted credentials**
  We will be distributing the dxi CLI, a Delphix solution built to facilitate simpler and seamless integration of Delphix Platform Operations into existing workflows, such as Windows, macOS, and RHEL binaries. This will simplify adoption and remove the requirement on Python. Delphix has also added encryption for the login credentials.
- **Expansion of retention period on replicated objects**
  At present, when snapshots on the replication source engine are deleted (either due to retention policies or user action), the next replication job will delete those snapshots on the replication target engine. This improvement will allow you to extend the retention period of replicated objects on target engines while keeping the original retention at the source. Once the object in the target engine reaches its retention period, it will be flagged and deleted by the policy agent based on a daily schedule.
- **Db2 staging push**
  Delphix now supports Staging Push for non-DPF Db2 databases. The Staging Push architecture will allow organizations to bring their data to Delphix, with their own tools and standard processes. This facilitates the use of any backup tool, a major ongoing source of requests. This should dramatically increase the volume of data that can be easily managed by Delphix.
- **HANA plugin staged architecture**
  Delphix introduces a staging architecture for HANA virtualization. This will make it consistent with other virtualized data sources. This new architecture will build a foundation for future staging push capabilities, as Delphix has begun to introduce other platforms. These changes, together, will allow us to support more prospective organizations with various SAP-certified, 3rd-party backup applications for HANA. Delphix will continue to support the pull ingestion method with HANA native backups and logs.
- **ASE native encryption support**
  SAP ASE version 16.0 introduces the ability to fully encrypt databases and provides protection for all the data, indexes, and transaction logs in a database. This offers full database protection while allowing the user to query and manage the data as usual, as the encryption is transparent to existing functions. In response to customer demand, we have added a security enhancement to support encrypted ASE databases. This allows customers to maintain the ASE encryption that is active on their sources and propagate that through to their VDBs.
- **System tunable interface**
  A new web service API is introduced which allows you to set or receive the values of a system tunable via Delphix CLI. A Support engineer can now provide context via a Support case to modify these values.
- **Oracle customized full backups**
  There is a rarely-seen bug in Oracle that results in some blocks not being written to the datafiles during an Oracle SnapSync operation. When this happens, the datafiles can become incomplete and provisioning/ refreshing from that snapshot might fail. We are providing a SnapSync option via CLI that you can customize to accept all datafiles during an Oracle SnapSync operation to prevent this error.

Release 6.0.7.0

Virtualization

- **Simplified connection management for Oracle databases**
  This feature streamlines the way that Delphix communicates and interacts with Oracle databases by simplifying the connection management infrastructure. Prior to this release, connections were established to Oracle databases using two different methods (remote connections from the Delphix Engine and local connections from the Delphix toolkit) and communication was performed with Oracle databases using two different users (a Delphix OS user and a Delphix DB user). Starting with this release, all communication with Oracle databases will be performed locally on the Oracle host and all connections to Oracle databases will be established using OS authentication. Existing Oracle dSources and VDBs will continue to function with no user intervention required. This feature results in several key benefits for Oracle DB customers such as the elimination of the requirement for a Delphix DB user when linking, automated PDB discovery, elimination of Delphix interaction with any network listener, and many more.
- **Virtualization SDK support for password vaults**
  Building off of the existing CyberArk and Hashicorp support for Oracle, SQL Server, and SAP ASE database user credentials, Continuous Data will extend password vault coverage to the virtualization SDK (vSDK). This will enable data sources that are connected via a vSDK plugin to incorporate this more secure method of authentication.
- **SAP ASE device mapping improvements**
  The Continuous Data experience with SAP ASE heavily relies on and mirrors a database's device allocation from the initial load (creating the dSource) to provision (creating VDBs). As these source device allocations shift over time, Delphix maps these changes and propagates them to their associated Delphix objects. However, dramatic device layout changes can negatively impact performance. This enhancement provides a quality-of-life (QoL) improvement to the overall SAP ASE experience by providing better error handling and escape valves should a dSource get into a bad state due to a major device layout shift.
- **Improved storage utilization for large pools**
  Up through the 6.0.6 release, Continuous Data has enforced a storage usage limit of 85%. Once met, this limit will cause certain API operations to be disabled to ensure engine data integrity. In the 6.0.7 release, this threshold is relaxed significantly. The new thresholds are as follows:
    - "Warning": when 85% of the total storage quota is reached or 1536GB of free storage is remaining (whichever is less), which can be resolved/ignored, with no impact on system behavior.
    - "Critical": when 90% of the total storage quota is reached or 1024GB of free storage is remaining(whichever is less), which cannot be resolved/ignored, with some impact on system behavior.
    - "Minimum": when 95% of the total storage quota is reached or 512GB of free storage is remaining (whichever is less). In this case, a critical fault is raised and cannot be resolved/ignored, with a substantial impact on system behavior (stop policies, VDB operations, etc).
- **PVSCSI support**
  In addition to LSI Logic, with the 6.0.7 release, Delphix has added support for the VMware Paravirtual vSCSI controller (aka PVSCSI). While VMware designed PVSCSI to support very high throughput with minimal processing cost, the performance improvements on Delphix engines can vary from case to case. In 6.0.7, we also support manual changes from LSI Logic to PVSCSI for currently deployed engines.

Release 6.0.6.0

Virtualization

- **Solaris x86 to Linux x86 Oracle DB provisioning**
  This feature allows the provisioning of Oracle Virtual Databases from Solaris x86 dSources to Linux x86 target environments.

- **TDE for Oracle Multitenant**
  Oracle Advanced Security Transparent Data Encryption (TDE) provides the ability to encrypt sensitive application data on storage. Delphix will now support TDE for Oracle 12cR2, 18c, and 19c multitenant. This release introduces support for single-instance linked container databases (CDBs) using software keystores. Virtual Container Database (vCDBs), RAC, and rekeying of the TDE encryption keys are not supported in this release.
  Note:
  Please note the following important restrictions for **TDE for the Oracle Multitenant** feature:
    - TDE-enabled vPDBs must be provisioned to a linked CDB, not a vCDB.
    - RAC dSources and target CDBs are not supported.
    - The Oracle version must be 12.2 or higher (12.1 is not supported).
    - System tables or tablespaces either in the PDB or CDB must not be encrypted.
    - Oracle Key Vault is not supported.
    - Hardware keystores are not supported.
    - Keystores must not be on ASM storage.
    - The dSource from which the initial provision is done must be encrypted when it is linked. Existing dSources cannot be encrypted without unlinking and creating a new dSource.
    - Encrypting an already-provisioned unencrypted vPDB (with clear data) which is managed by Delphix is not supported
- **Single to Multitenant VDBs**
  Oracle announced the end of support for non-multitenant databases in their 20c release, and as such, Oracle DB customers are planning their upgrade and migration programs. Delphix will now support provisioning a virtual pluggable database from a non-multitenant virtual database.
- Added support for **HashiCorp namespaces**.

Certifications

- **Virtualization**
    - ASE 15.7/16 on RHEL 7.9
    - Oracle 12.1 on RHEL 7.9 and SLES 12 SP5
    - Oracle 12.2 on RHEL 7.9 and SLES 12 SP5
    - Oracle 19c on RHEL 7.9 and SLES 12 SP5
- **Hypervisors/Clouds**
    - ESXi 7.0 U1

Release 6.0.5.0

Virtualization

- **NFSv4 support:**
  Support has been added for Oracle and ASE on AIX.
- **Expanded Replication: Replication of non-data objects**
  Our customers are increasingly using replication to facilitate moves of data across network boundaries, to the cloud, and for DR purposes. We've had long-standing requests to replicate more than just the data, and in this release, we will support the replication of users, roles, permissions, policies, and configuration templates.
- **Upgraded Windows Connector:**
  The Windows Connector will now support newer versions of Microsoft's .NET framework (4.x), which encompasses myriad higher security standards, new functionality, etc. Previously, the connector relied on .NET 3.5 due to two dependencies: SQL Server and Powershell, both have since been removed with SQL Server 2016+ and Powershell update in 6.0.3.0.

- **Db2 Extensible Ingestion:**
  We will now support an extensible model for ingesting Db2 data. In this new, additive, model, we will support customers manually performing a restore & roll forward of their staging database to Delphix from native backups or arbitrary third-party backup tools which integrate directly with Db2. This will allow customers to bring data from whatever system or backup they have and restore it to an exact point in time, as needed.

Certifications

- **Virtualization**
  - ASE 16.0 on RHEL8.1 and RHEL8.2 on 6.0.4+
  - ASE 16.0 on SLES12.4 on 6.0.4+
  - Oracle 19.7 on RH7.8 and RH8.0 on 5.3.9+ and 6.0.3+
  - ESX 7.0
  - NFS v4 support on AIX
  - IBM Cloud Catalog. Delphix is now available in the IBM Cloud Catalog, a private marketplace for trusted IBM Technology partners that is offered to large IBM enterprise customers. In 6.0.5 we will start with a few certified instances for virtualization and masking and will grow our presence as more the business justifies the cost and efforts. Specifically, we support the following instances:
    - mx2-8x64
    - mx2-16x128
    - mx2-32x256
    - mx2-48x38
  - Oracle Cloud: The following are newly supported instance types:
    - VM.Standard2.8
    - VM.Standard2.16
    - VM.Standard2.24

Release 6.0.4.0

Virtualization

- **HashiCorp and expanded CyberArk support:**
  Delphix has extended both CyberArk and HashiCorp Vault support to Oracle Database Users in addition to previously supported ASE and MSSQL domain users. GUI support for HashiCorp Vault has been added during setup to authenticate host users and database users.
  Note:
  The HashiCorp namespace Enterprise feature is supported starting 6.0.6.0.
- **NFSv4 support:**
  Support has been added for SuSE and Db2 on AIX.

Certifications

- **Virtualization**
  - OCI Support
  - NFSv4 support for Db2 on AIX and SuSE
  - SQL Server Instances with a Managed Service Account

Release 6.0.3.0

Virtualization

- **CyberArk and Hashicorp vault support for virtualization**:
  Delphix is introducing password vault support to authenticate environment and database linking and will support both CyberArk with Oracle and Hashicorp with CLI only.
- **Capacity management**:
  Understanding where and how storage is used on Virtualization Engines can be a challenge, in particular, understanding how and where space is held and how to recover it. In this release, Delphix provides better details of held space, particularly around locked objects, and provides clear instructions about what steps are required to free up space.
- **Diagnosability**:
  Additional performance health-check analytics in phone-home have been added to better troubleshoot and understand customer problems.
- **Powershell upgrade**:
  Delphix is reducing our requirements for Windows hosts running PowerShell by allowing you to use any PowerShell version from 2.0 to 5.1. Delphix will now use the default available PowerShell version on each host. When specifying hooks (such as “configureClone”), users may specify whether to use 2.0 or whatever PowerShell version is installed on the host.
- **Support for Oracle read-only homes:**
  Delphix is introducing support for Oracle read-only homes, which is a new Oracle feature starting with Oracle 18c. In a read-only Oracle home, all the configuration data and log files reside outside of the read-only Oracle home. This feature allows you to use the read-only Oracle home as a software image that can be distributed across multiple servers.
- **Replication performance:**
  Delphix will continue to improve replication performance for replication specifications that include multiple objects and single-object replication throughput.
- **SAP ASE support for VDB upgrade:**
  SAP ASE Customers will now be able to validate DBMS Upgrades with this feature that enables provisioning VDBs to a higher version than the source DB (e.g. ASE 15.7 > ASE 16).
- **Shared NFS for toolkits**
  With this release, Delphix introduces shared NFS for clustered environments. Customers wish to use a common NFS mount point, in which the Delphix toolkit for each cluster node can be deployed. The product today only creates a directory with appliance UUID and OS user in the folder name and uses this for detection to determine if a host is already managed by that Engine. As such, this prevents the customer from utilizing common NFS storage due to name conflict.
  When a new environment is created, upgraded or if the toolkit path is changed, a new toolkit is created with naming convention Delphix_COMMON_ for common directory and Delphix_ for user directory.
  With this change, the customer can use mounts on shared file systems (like NFS) as a toolkit path for clustered environments without any naming conflict.This change is not intended for windows environments.

Certifications

- **Virtualization**
  - EBS 12.2 with RHEL 7.6
  - PostgreSQL 12.1 & 12.2 with RHEL 7.8
  - Oracle 11g R2 and Oracle 19c with RHEL 7.8 on 5.3.9.0 and 6.0.2+

Release 6.0.2.0

Virtualization

- **Support for Db2 Database Partition Feature (DPF)**:
  Delphix has long supported distributed Db2 (running on Unix/Linux Systems). However, Db2 supports partitioned databases as a means of scaling to larger, more complex systems. With this release, Delphix will now support Db2 DPF allowing you to scale to an increasing number of your Db2 databases.
- **Windows authentication for SQL Server**:
  You will now be able to use Windows Authentication to link SQL Server databases. Rather than providing both a database user and a Windows user to ingest data, you can leverage one set of credentials (a Windows OS user) to perform all source operations. This capability will simplify SQL Server deployments and reduce Delphix's security requirements on source databases.
- **Smart failover**:
  Smart Failover allows the Delphix Administrator to simplify failover processes by automating object conflict resolution. By selecting a new option "Automate Object Conflict Resolution" before the failover process starts, the failover process will rename all conflicting objects and show a report of all object changes at the end.
- **NFSv4 support:** In 6.0.2 Delphix will start providing NFSv4 for data sources running on RedHat 7.0 or later. NFSv4 can be enabled using the CLI. Support for additional host OS versions will be added in subsequent releases. Delphix will consider enabling NFSv4 by default for those supported configurations in a future release.
- **Support bundles not required for upgrade**: When upgrading from 6.0.0 or greater to a release 6.0.2 or greater, we no longer require support bundles to be sent to Delphix. This allows you to execute more self-service upgrades.

Certifications

- ASE 16.0 on AIX 7.1
- AWS r5n Instance Support: r5n.2xlarge, r5n.8xlarge, r5n.16xlarge
- Azure E Series Instance Support: E8s_v3, E16s_v3, E32s_v3
- Masking support of Oracle 19c.

Release 6.0.1.0

- **Masking extended connectors**: A very common request for masking has been to support additional data sources, outside of the currently supported list. Thus, the next step in the strategy is the release of Masking Extended Connectors, which will allow our customers to add JDBC drivers to the masking engine to facilitate the masking of additional data sources. This will allow masking to be used for other common databases that can be accessed via JDBC, like SAP HANA, Informix, etc.
- **SQL server CDC support**: We have expanded our support for SQL Server databases using Change Data Capture (CDC), a SQL Server feature that captures all the change information that is applied to the databases and stores it in change tables. Now, users will have the ability to preserve CDC data and enable CDC for SQL Server VDBs.

Certifications

- **Virtualization**
  - ASE 16 and 15.7 with Solaris SPARC 11U3 and SPARC 11U4
  - ASE 16 and 15.7 with RHEL 7.7
  - SQL Server 2019 Support with Windows 2016 and Windows 2019
  - Oracle 19c with SUSE SLES 15 SP1

    - Oracle 19c with Solaris 11 U4 and U3 x86
- **Hypervisors**: The following hypervisors have been certified in 6.0.1.
    - VMware ESX 6.5 U1, U2, U3
    - VMware ESX 6.7 U3

Release 6.0.0.0

Google cloud support: Delphix now supports running in Google Cloud Platform for existing supported databases.

Enhanced Networking Adapter (ENA) support: Delphix supports networking on AWS instances with the Elastic Network Adapter (ENA). This offers our customers enhanced networking capabilities and more economical options. Notably, this includes the AWS R4 instance types.

- **Masking NFS/CIFS mount**: Our customers increasingly are masking files alongside their databases. The masking engine has classically supported this via FTP/SFTP but now to make things easier Delphix has introduced the ability to directly mount and mask a file system - over NFS and CIFS. This should dramatically simplify the process of file masking.
- **Oracle quality**: Continued focus on Oracle quality and have introduced several quality improvements with our 6.0 release.
- **Masking API updates**: 6.0 introduces a significant number of new endpoints, including mainframe control, as well as updates for existing endpoints. This release also introduces versioning for the masking API, allowing our customers to upgrade without risk of breaking their integrations.
- **AdoptOpenJDK 8 for the Delphix toolkit**: Delphix has changed the Java Development Kit (JDK) that is included with the toolkit, and is sent to all Delphix connected environments. Customers who require using Oracle Java may continue to do so with the feature to provide their own Java, which shipped in 5.3.5.
- **Removed instance check**: When running in AWS or Azure, the product will no longer raise a fault when it detects that it is running on an unsupported instance. This enables Delphix to certify previously released software on new instances without having to modify the software.
  The product will still detect what instance it is running on and include this information in the user interface and phone home bundles. We will also continue to publish a matrix of supported instances for Azure, AWS, and GCP in the product documentation. Delphix provides no guarantee of performance or support for unsupported instance types.
- **Upgrade process**: The upgrade to 6.0 will be an in-place upgrade like other Delphix releases, there are a few changes that will improve the process overall for 6.0:
    - We will require an upgrade to an interim release first (either 5.3.6). This can be done at the same time as the customer upgrades to 6.0 or in the months prior.
    - We will be introducing new upgrade checks to ensure that customers are not using features that have been removed. For a list of removed features see Deprecated and End-of-Life Features.
        - We will provide an upgrade image specific for each platform we support with Virtualization (VMWare, AWS, Azure, GCP). This will allow us to be more precise in customizing the images for each.

Fixed issues

Multiple Device Removal in the Delphix Engine 6.0.12.0 and higher version introduces a breaking kernel module change that requires a reboot to load the new module. Therefore, a deferred reboot engine upgrade operation will be unable to remove devices until a reboot is performed.

Release 14.0.0.0 Changes

Fixes that take effect after an optional reboot

Bug Number	Description
DLPX-86764, DLPX-87360	Added a mechanism to automatically restart internal service if I/O to object storage is not making progress.
DLPX-86962	Eliminated deadlock scenario introduced by upstream changes to ZFS.

Fixes that take effect immediately after upgrade

Bug Number	Description
DLPX-67911	Fixed issues related to having stale mounts on staging and target host.
DLPX-71914	Capacity table UI for received replica now shows aggregated size for virtual and source datasets.
DLPX-79098	Seconds is also shown as part of the snapshot point in time on the Summary of Refresh Wizard.
DLPX-83250	Support for SSL for Fluentd/Insight data consumers.
DLPX-84003	Fixed an issue where provisioning an Oracle vPDB into a new vCDB when the source PDB$SEED has a TEMP tablespace, where the initial allocation plus file headers is greater than 100M (for 8k blocksize) fails with “ORA-03214: File Size specified is smaller than minimum required”.
DLPX-85770	Fixed a failure reported by a start operation on a self-service container or an enable operation for a Oracle virtual source that is already enabled.

Bug Number	Description
DLPX-86195	Fix ensures the `is_encrypted` flag is inherited during provisioning via snapshot. Activate with the tunable `CREATE.ENCRYPTED.SEED.DB` set to true (the default is false).
DLPX-86620	Instead of raising faults for `java.util.concurrent.RejectedExecutionException`, changes have been made to log the issue in debug logs and fail silently, to reduce noise.
DLPX-86644	Fixed an issue where a TDE-enabled vPDB provision or refresh operation fails with a misleading error if the parent CDB or parent PDB's primary key is missing in the parent TDE keystore on the target host.
DLPX-86685	Fixed an issue causing VDB operations to fail after upgrade verification is run.
DLPX-86805	Updated Spring Framework dependency version to 5.3.28.
DLPX-86854	Fixed an issue that caused post-upgrade cleanup to fail.
DLPX-86906	Fixed an issue to prevent linking an Oracle PDB that is added manually in a virtual CDB.
DLPX-87006	Fixed an issue in refreshing self-service containers with ordered sources, post upgrade to 12.0.0.0.
DLPX-87038	Fixed a size discrepancy in the OVA to fix certain deployment issues.
DLPX-87219	Post-refresh Snapsync of an Oracle VDB/vPDB fails with `exception.oracle.vdb.uncustomizable.parameters.changed` after the PDB is detached/attached following a standby switchover.

Release 13.0.0.0 Changes

Fixes that take effect immediately after upgrade

Bug Number	Description
DLPX-61403	If an Oracle CDB/vCDB has a quota policy configured, Delphix will ensure that PDB/vPDBs are disabled/enabled properly when quota policy disables/enables the CDB/vCDB.

Bug Number	Description
DLPX-71066	Fixed an issue to provide better action when a plugin upload fails due to the existence of a badly provisioned VDB.
DLPX-74553	Fixed an issue where the `Target Database Exists` error may be thrown during VDB provisioning, if a VDB was present during the last Environment Refresh.
DLPX-78987	A RAC node can only be disabled if no virtual sources are in 'running' status on the node.
DLPX-81268	Fixed an issue where executing `expect` commands logged sensitive information.
DLPX-84624	Removed build date from the Upgrade Images page UI.
DLPX-85298	Created new Oracle faults for quota policy violation with more details for error and action text.
DLPX-85747	AWS Cloud engines now support IMDSv2.
DLPX-86109	Fixed an issue where Oracle VDB unquiesce/enable may fail after a failed quiesce/disable on 10.0.0.X or 11.0.0.X.
DLPX-86163	Fixed an issue where if TNS_ADMIN for an Oracle vCDB was wrongly set during provision or refresh, it would cause older clients such as SQL*Plus 9.2.0.8 and Oracle Application Server 10.1.0.3 to fail with `ORA-28040: No matching authentication protocol`.
DLPX-86272	Fixed an issue for secondary nodes with different MSSQL instance owners than the primary node, where it now works with Delphix Managed backups.
DLPX-86344	Fixed an issue related to failed Delphix engine upgrades due to plugin operations that were failing.
DLPX-86496	Fixed an issue in the interaction between replication and replica retention policy.
DLPX-86497	Fixed the functionality to relink/attach SQL Server Availability Source Group database using the Link dSource dialog.
DLPX-86563	Fixed an issue where if any detached dSources were present then the performance history page would hang.

Bug Number	Description
DLPX-86787	Fixed the GUI un-responsiveness of the Datasets page, after viewing the replication profiles.
DLPX-86842	Fixed a failure in the `verify upgrade` job while upgrading to 12.0.0.0.

Release 12.0.0.0 Changes

Security Fixes

Bug Number	Introduced	Description	Security Bulletin
DLPX-86329	6.0.13.0	Sysadmin can execute shell commands on the underlying Operating System.	TB109

Fixes that take effect after an optional reboot

Bug Number	Description
DLPX-86177	Fixed a bug where Accelerated Networking was broken due to missing drivers.

Fixes that take effect immediately after upgrade

Bug Number	Description
DLPX-44117	Fixed an issue where a null pointer exception is encountered during parsing of an Oracle archive log not present in the Delphix file system.
DLPX-59966	Fixed an issue where the Snapsync of an Oracle vCDB may become unresponsive if the Delphix archive log destination is removed or changed, pointing to an invalid location.
DLPX-72422	Reduced VDB downtime on upgrade.
DLPX-82702	Improved error behavior when attempting to expand storage devices with underlying problems.
DLPX-83430	Fixed an issue where the initial configuration of Syslog breaks most of the pre-existing appenders.

Bug Number	Description
DLPX-84611	Fixed an issue allowing environment users that are NOT in the primary group of the primary OS user to monitor the builtin files VDBs.
DLPX-85578	Replaced the Win32_Volume class output with mountvol output to fetch volumeId for Delphix ISCSI mount points.
DLPX-85647	Added a filter to fetch only the IPv4 address for AG cluster nodes.
DLPX-85748	Now displaying the state of enabled services in the Delphix Startup Screen.
DLPX-85857	Fixed an issue where after a Delphix Continuous Data engine upgrade to 10.0.0.0, faults related to listener registration are thrown during source enable.
DLPX-85883	Fixed an issue where a vPDB provision fails when invoked as a non-instance environment user, with standby source and datafile added in the recovery stream.
DLPX-85925	Added a new condition in subquery to uniquely identify filegroup for commvault backup files.
DLPX-86050	Fixed an issue where TDE diagnostics for directory permission fails if the TDE artifact directory has a space in the path or name.
DLPX-86102	Fixed an issue where ASE instance discovery is failing during environment add/refresh when ASE instance is running in multi-process mode with multiple ASE engine.
DLPX-86201	Fixed an issue where users are sometimes unable to generate a complete support bundle.
DLPX-86240	Internal services will now restart during a deferred upgrade, enabling the delivery of fixes without requiring a reboot.

Release 11.0.0.0 changes

Security fixes

Bug number	Introduced	Description	Security bulletin
DLPX-85604, DLPX-85606, DLPX-85608	Product Inception	Several input fields in the Self-Service feature are vulnerable to cross site scripting (XSS).	TB104

Fixes that take effect after an optional reboot

Bug number	Description
DLPX-82702	Improved error behavior when attempting to expand storage devices with underlying problems.
DLPX-85526	Eliminate the possibility that a spurious udev event during drive expansion will cause the pool to suspend.

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-86068	Fixed an issue from release 10.0.0.0 where users could not replace certificate chains when using root certificate authorities from Java's default list.
DLPX-85915	Fixed an issue where users were unable to add an environment when part of the host name contained all numeric elements.
DLPX-85752	The icons for the export functionality have been updated to secondary buttons (along with the icon) for clarity.
DLPX-85732	Fixed an issue where provisioning a TDE-enabled vPDB into a new vCDB with "Mask this vPDB" check fails with "ORA-46636: cannot add second keystore to the target keystore".
DLPX-85718	Starting from 7.0.0.0, the URL for "Help Documentation" has beed updated.
DLPX-85621	Fixed a UI message saying, "management stack needs restart after replacing certificate', when it did not.
DLPX-85601	Removed weak CBC ciphers from default list.
DLPX-85493	MSSQL: Linking and AttachSource operations will not require unnecessary permissions of source user on the staging host and staging database.
DLPX-85445	Fixed the software version main window issue so that it now updates if a previous version is selected.
DLPX-85205	Improved the error message for environment authentication failure by including username.
DLPX-84877	Fixed an issue where adding HPUX environment failed with an Unsatisfied link error.

Bug number	Description
DLPX-84792	Added new tunable, MSSQL.ROLLBACK_HANDLING_AFTER_FAILED_RESTORE. If this tunable is set to true, the Delphix engine will perform a zfs rollback on DATA LUN right after a restore db failure.
DLPX-84738	Fixed an issue where the environment user public key is not displayed in the CLI/API.
DLPX-84709	Fixed an issue to disallow CLI or API transactions for an Environment user with a null publickey or privatekey.
DLPX-84610	Improved error message to show correct host name and user name in the case of builtin files virtual source status failure.
DLPX-83927	Fixed an issue so that Fluentd/Splunk integration will accept protocol parameter in any case.
DLPX-83897	Fixed an issue with changing port number while configuring Kerberos.
DLPX-83360	Fixed an issue where new authorization would not be granted unless the old one gets removed successfully.
DLPX-81587	DeletionDependency object in API documentation has been updated to reflect that size is Valid for TimeFlowSnapshot and HeldSpace objects.
DLPX-82895	Fixed an upgrade verify issue when fs.inotify.max_user_watches exceeds the limit of 16384.
DLPX-80325	Fixed an internal error that appeared in the Oracle dSource upgrade operation while resuming logsync for the dSource.
DLPX-80221	Resolved issues related to incorrect mount path by normalizing the path.
DLPX-79492	Improved the selection of faults and the toggle behavior of "Hide Resolved/Ignored" from the "Verification Result's Faults List" on the Version Upgrade page.
DLPX-78839	Fixed an issue where the NTP server would not validate when configured.
DLPX-77826	Added timeout functionality to JDBC queries made through DSP. Leaked connections due to network issues will be auto-closed after the timeout period. Timeout can be set using the tunable dsp.jdbc.queryTimeout.
DLPX-77386	Fixed a misleading prompted message about host being unavailable.

Bug number	Description
DLPX-76762	Performance improvement: Enabled unbuffered copy and multithreading options with the robocpy command for vFile operations.
DLPX-75448	Fixed an internal error that appeared when the purgeLogs API was used for timeflows with no snapshots.
DLPX-74905	Fixed an issue where environment refresh and VDB provision/refresh operation will fail with an "internal error" if the target host's filesystem becomes 100% full.
DLPX-73058	Fixed an issue where if an environment is disabled, VDB operations like start, stop, rollback, refresh, undo, disable would not be allowed.
DLPX-63076	Fixed upgrade check result messages to make them consistent with the failure's severity level.
DLPX-63425	Delphix will now throw a job warning and raise a critical fault in the Delphix Admin UI if a failure is encountered while backing up the archived logs and LogSync is disabled. In this case, Delphix will invoke an RMAN script to delete the backups.
DLPX-67347	Fixed an exception encounter (paired with an indefinite loading sign in the UI) when an environment addition task was cancelled in between. The environment got deleted in the next attempt but the toolkit folder did not. Added DUE to show the warning for toolkit deletion failure.
DLPX-84981	Improved the error message for null or whitespace timezone registry.
DLPX-85674	Fixed the issue where snapshots were created with the wrong (older) timeflow that led to a failure to enable the linked sources with internal error.
DLPX-85102	Reduced the size of the temporary directory unique name to be backward compatible with the name `_delphix`.

Release 10.0.0.0 changes

Fixes that take effect after an optional reboot

Bug number	Description
DLPX-85019 (only applicable to initial deployments)	Fixed a bug that causes the management service to fail startup upon initial VM deployment in the cloud, if the DHCP domain name of the VM ends with a period (.).

Bug number	Description
DLPX-85109 (only applicable to initial deployments)	Fixed a bug that causes the management service to fail startup upon initial VM deployment in the cloud, if the DHCP domain name of the VM contains bad characters (does not belong to [a-zA-Z0-9.-]).
DLPX-84985	Fixed a deadlock which caused iSCSI connections to fail on Windows hosts.
DLPX-84995	Fixed an issue where NFS could cause excessive CPU usage when open files from NFSv4 mounts exceeded 16,384.
DLPX-62215	Fixed a nuance in the CAPACITY_RECLAMATION job.

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-51276	Configuration discovery script fixed to handle Windows host timezone more smoothly.
DLPX-53420	Improved error message for failure in deleting CDB sources with no associated PDBs, but with timeflow dependencies from migrated vPDBs.
DLPX-59162	Fixed failure in connecting to a Compliance engine while running a compliance job as a post-provision hook operation for RAC VDBs.
DLPX-61781	Fixed authentication failures past the first node while adding Kerberized RAC environments.
DLPX-66944	Fixed logsync failures caused by commands exceeding shell limit of 1,024 characters.
DLPX-68327	Certificates in the Truststore are considered for making connections to SMTP servers.
DLPX-74134	Fixed an issue where Oracle provisioning/refresh/rewind operations may fail with error "ORA-01169: DATAFILE number 1 not found. Must be present."
DLPX-76200	To handle unusually demanding workloads, the maximum heap size of the management application can now be increased by system administrators via the maxHeapSizeGb property of the CLI at /system.

Bug number	Description
DLPX-78673	Fixed an issue related to a message being ignored during the mount check when provisioning virtual databases. The message is not thrown because the provision completes successfully, but it could still be seen in the logs.
DLPX-80925	Fixed an issue of a raised fault due to inconsistent snapshot creation time, due to manual change of Delphix engine time.
DLPX-81874	Fixed an issue that occurred during the discovery of a cluster when adding or refreshing a Windows FCI environment with multiple NICs. The issue involved the use of the IP address of hosts that were already added.
DLPX-81982	The Delphix Fluentd logs do not automatically rotate when exceeding their expected maximum size of 100MB.
DLPX-82169	Fixed an issue where provisioning an Oracle TDE-enabled vPDB into a linked CDB with a different patch level than the source CDB will fail leaving the vPDB in a broken state.
DLPX-84108	The validation of RSS on the target host will now only occur when the customer uses an IP address for an environment. This update removes the detection of the RSS property on interfaces that are not relevant.
DLPX-84422	You can now use a single script to copy all transaction log backup files, regardless of whether they were created using native backup or Litespeed backup.
DLPX-84647	The syslog pattern is now fully configurable and can be changed to conform to RFC 5424.
DLPX-84655	A false warning message that listener registration was not successful is posted when enabling a VDB or a vCDB, users can ignore this message.
DLPX-84679	Unable to add Oracle Staging push PDB if the Staging Environment has more than one repository.
DLPX-84686	To address the issue of multiple IP addresses, the validation of RSS on the target host will now only occur when the user uses an IP address for an environment. This update removes the need to find the IP address in the code.
DLPX-84898	Fixed clean-up job failure for V2ASM of a RAC VDB when one of the nodes's DB instance is shutdown.
DLPX-84929	Improved initial load time of datasets page.

Bug number	Description
DLPX-84944	The Delphix Continuous Data Engine now considers truststore certificates for more connections, including secure connections to proxy and SMTP hosts.
DLPX-85053	Powershell scripts code enhancement.
DLPX-85095	Fixed an issue which caused the vPDB Refresh operation to fail due to vPDB unplug operation timing out in 30 mins.

Release 9.0.0.1 changes

Bug number	Description
DLPX-85176	Fixed a bug that can bring down VDBs using NFSv3.

Release 9.0.0.0 haanges

Fixes that take effect after an optional reboot

Bug number	Description
DLPX-85019	Fixed handling of domain names with a terminal period (.) in Terminal.
DLPX-68852	Fixed a bug in out of memory situations that could result in service interruptions.

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-41505	Posted a job warning message when listeners are not registered successfully as part of an Oracle VDB/vPDB provision job.
DLPX-46210	Fixes `fault.policy.log.retention.old.snapshot`, which is not raised against a PDB snapshot instead of CDB snapshots.
DLPX-57047	For MSSQL, 'Delphix Copy Only Full Backup' is now excluded when synchronizing to most recent backup files.
DLPX-58762	Provides a better action message in Oracle nologging-related faults.

Bug number	Description
DLPX-65557	Retention no longer removes the previous timeflow until the current timeflow is successful.
DLPX-68852	Fixed an issue that could cause upgrade failures.
DLPX-69790	Fixed an issue where API login with an ambiguous user name returns an incorrect action.
DLPX-70090	The correct exception is now showing when hostname is not resolvable during Environment Validation.
DLPX-79792	If there are any DB snapshots associated with the VDB during creation of a Self-Service bookmark, there will be a message to drop all associated snapshots with the VDB and try again.
DLPX-81260	The CLI error message that appears due to 'invalid primaryAddress value', has been updated to provide better insight on a resolution.
DLPX-83702	Fixed an issue where the environment discovery path was not checked to ensure it is a valid unix path.
DLPX-83905	Fixed an issue causing `HOST_REFRESH` failures when toolkit is on a shared filesystem, with an Oracle RAC configuration.
DLPX-84068	For MSSQL, warning fault raised in case there is a failure while querying instance port.
DLPX-84284	After upgrades, NFSv3 services are automatically disabled if they are no longer required.
DLPX-84339	Fault raised due to inconsistent snapshots creation time, due to manual change of Delphix Engine time.
DLPX-84351	SMTP Test now has a check in the UI to let the user know if the password was unaltered. Save now has a check to not send the password if it was unaltered.
DLPX-84528	Datasets Search bar will now work for the staging push dSource as well.
DLPX-84589	Increased size of ssh key fields to allow for larger key, for environment users.
DLPX-85081	Fixed an NFS issue that could cause NFSv3 services to be disabled even though there were active v3 mounts after an upgrade.

Release 8.0.0.0 changes

Fixes that take effect after an optional reboot (Activated after option Reboot)

Bug number	Description
DLPX-83859	Fixed a rare deadlock in the kernel that can cause a Delphix Engine to become unresponsive to all management operations.

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-43174	Fixed an issue where Oracle VDB provision job will fail if the "Open database after provision" flag is set to false.
DLPX-48083	Removed the CLI ability to set default user as it was not needed and prevents deletion.
DLPX-52904	After initial server setup, Default Snapshot/Snapsync policies are always in "US/Pacific", regardless of the timezone selected during initial Engine Setup. An admin has to manually change the time zone.
DLPX-53209	Improved error messaging during toolkit preparation (discovery) when there are host problems.
DLPX-59248	Added a fix to throw a critical alert on the last failed attempt to collect the archive log.
DLPX-59308	Added a timeout in the drop database PowerShell script.
DLPX-59689	Fixed an issue where the environment discovery could hang indefinitely when UserLAnd commands hang (i.e. lsnrctl status, ps, etc.).
DLPX-61734	Fixed an issue where the user was unable to provision an Oracle VDB when there is a dollar sign in the Tablespace name.
DLPX-62857	Permits listing users with only the "domainUserType" parameter.
DLPX-64413	Fixed an issue where provisions/refreshes of Oracle VDBs could fail when an obsolete parameter is specified in the VDB config template.
DLPX-72691	Fixed an issue where multiple snapshots were reporting the same time in the Delphix Continuous Data GUI for standby databases.

Bug number	Description
DLPX-72740	Increased users' pubkey support from 10 to 1000.
DLPX-75521	The correct error message will now be displayed in case of a provisioning failure due to waiting for the 'DB STARTUP' process timeout.
DLPX-75605	Fixed an issue where the Security Banner was not displayed after SSH login.
DLPX-77438	Added a fix to disable UNDO operation if any children for the timeflow are present.
DLPX-77792	Fixed a misleading "toolkit inaccessible" error if password expired.
DLPX-78702	Fixed an issue where users with the SYSTEM permission are able to disable any source.
DLPX-78741	For MSSQL, raised a critical fault in case the Validated Sync interval increases to 16 minutes or more.
DLPX-79136	Fixed an issue where canceling an Oracle VDB preprovision job or a vPDB (into linked CDB) provision job leaves the auxiliary database mounted on the target host.
DLPX-79919	Fixed an issue where the Delphix Continuous Data Engine allowed an environment with a duplicate "crs_database_name" to be added.
DLPX-81182	Improved the error message for when the system is out of space.
DLPX-81425	The UI will now show the running jobs up to 60 days old.
DLPX-81497	Added a fix to improve "keystore.merge.required" fault for an Oracle TDE-enabled vPDB.
DLPX-82152	Fixed an issue where navigating resolved faults was slow.
DLPX-83575	The port in the connection string for a vPDB in a Linked CDB may be shown incorrectly when it is registered to a non-default listener.
DLPX-83635	Fixed an issue where exporting the TDE encryption keys failed when the keystores root is on ACFS and referenced via a symlink.
DLPX-83788	Suggested action in "exception.ccc.authenticate.failed" no longer references Delphix Connector.

Bug number	Description
DLPX-83823	The issue where a Oracle PDB dSource snapshot is marked as not provisionable after detaching and re-attaching the PDB to a different CDB using force flag is now resolved
DLPX-83904	Fixed an issue where a provision of an Oracle vPDB may fail during recovery if there are many datafiles that are renamed under a single ORA-01244 error.
DLPX-83954	Package details are no longer revealed during HTTP redirection.
DLPX-83977	Fixed an issue causing inability to add a hook, due to "Duplicate key" error.
DLPX-84103	Persisting total "database_transaction_log_bytes_used" while taking a snapshot, for debugging purposes.
DLPX-84151	The Replication Page performance has been optimized by making network calls efficiently.
DLPX-84255	In a Single Engine Continuous Vault product, adding a new Sybase dSource to a locked group may result in the background environment monitoring process to stop working.
DLPX-84324	Fixed an issue where NFS mounts on a Solaris target could fail after a deferred upgrade.
DLPX-84495	Fixed an issue that causes upgrades from versions < 6.0.17.0 to any version between 6.0.17.0 and 7.0.0.0 on a replication target engine which may fail due to the management services being down, requiring a support call.

Release 7.0.0.0 changes

Fixes that take effect after an optional reboot

Bug number	Description
DLPX-83579	Fixed handling of domain names with a terminal period (.) in Terminal.
DLPX-83611	Fixed a race condition between storage device link creation and the storage pool import process.
DLPX-83701	Added additional diagnosability tracepoints to the kernel unmount code.
DLPX-83697	Fixed a hang in the iSCSI initiator.

Bug number	Description
DLPX-83675	Fixed an issue that was causing stale entries to be created in a system file.
DLPX-83684	Fixed a crash in the zcache_probe command during upgrades.
DLPX-83916	Fixed cases when zcachedb was opening devices for writing when it should be read-only access.

Fixes that take effect after an optional reboot (Activated after option Reboot)

Bug number	Description
DLPX-80130	Fixed an issue which may cause the Delphix Engine to hang when doing storage migration.
DLPX-83395	Fixed an issue when storage device removal consumes an unexpected high amount of memory upon its completion.
DLPX-83697	Fixed a hang in the iSCSI initiator.

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-66792	Fixed a misleading "toolkit inaccessible" error on VDB Stop/Refresh if password expired.
DLPX-68053	Fixed an issue causing phone home redaction fails when JSON contains management stack errors.
DLPX-71907	Fixed an issue where C drive label gets over-written during mount script execution.
DLPX-74604	For MSSQL, provided a fix to raise warnings in case CommVault restore fails.
DLPX-75278	Fixed display data in Datasets Performance page table.
DLPX-75488	Listeners and Instances for SQL Server AG (Availability Group) will now be shown on the User Interface correctly.
DLPX-77214	Fixed an invalid version status transition when upgrade is cancelled.

Bug number	Description
DLPX-78014	Automatically disable NFSv3 services when they are no longer required.
DLPX-78506	Fixed dSource Database Authentication issue. Now the user is able to edit the credentials of the dSource DB, where authentication is configured using “Domain User with HashiCorp Vault Credentials”.
DLPX-80300	Added more error information for Environment discovery failure when ISCSI target port is blocked on network.
DLPX-80473	Fixed an issue where the vPDB unplug operation timed out during disable operation.
DLPX-80512	Fixed pagination on Snapshots tab of Storage Capacity page. Now the user will be able to see all the snapshots using the pagination control at the bottom of the page.
DLPX-81238	DFE caused by policy schedule with a non-recurring quartz cron string.
DLPX-81750	Added a more descriptive message while handling the condition where VDB is renamed outside.
DLPX-81794	For MSSQL, raised a warning fault during environment monitoring in case dlpxrunas is removed. Also, provided fix to propagate the cause of host unavailability while environment refresh.
DLPX-82288	It is now permissible to remove devices after a deferred upgrade from before 6.0.12.0 (multi-device removal support added) to 7.0.0.0 or after.
DLPX-82316	The issue with the failure of the Oracle vPDB provisioning from a standby source and datafile added in the recovery stream is now resolved.
DLPX-82882	Fixed Script: Replacing Self-signed Certificates, on the Delphix Connector.
DLPX-83002	Cron expression handling has been fixed as per quartz cron expressions.
DLPX-83153	Fixed the issue where an invalid fault "fault.oracle.db.connection.failed" was raised during VDB refresh.
DLPX-83372	Profiling script code enhanced.
DLPX-83422	Fixed an issue where Oracle move-to-asm script fails due to missing initialization parameter file init.ora in `$ORACLE_BASE_CONFIG/dbs`.

Bug number	Description
DLPX-83434	Upgrade Spring framework to 5.3.20.
DLPX-83564	Users can now perform V2P of dSource and VDB snapshot to root of a Windows drive.
DLPX-83608	Users can now V2P an MSSQL dSource or VDB snapshot to a Mounted volume on a Windows path.
DLPX-83622	Upgrade verify will fail from coming from 6.0.15.0 if a Fluentd plugin other than splunkHec is configured. Support help will be needed to upgrade.
DLPX-83706	Fixed network connectivity issues in cases where the MAC address changes.
DLPX-83783	Implemented a fix to prevent invalid transition after a deferred upgrade.
DLPX-83787	Fixed an edge case issue that could cause an engine to be rebooted after a device has been removed from the storage pool.
DLPX-83789	MountLunData.ps1 script code enhanced.
DLPX-83819	Fixed an issue causing script failure output duplication in debug logs.
DLPX-83824	Fixed an issue where datasets would intermittently going inactive with critical faults.
DLPX-83828	While editing an Environment User, the Password field will be empty by default in order to get the correct password from the end user.
DLPX-84073	Fixed an issue where existing TDE-configured vPDBs would fail to enable after upgrading the Delphix Engine from 6.0.14.0 to a later Delphix Engine version.

Release 6.0.17.0 changes

Security Fixes

Bug number	Introduced	Description	Security Bulletin
DLPX-83043	5.2.0.0	Weak DH 1024 bit exchange key detected by security scanner for the Delphix connector.	TB099

Fixes that take effect after an optional reboot

Bug number	Description
DLPX-75209	Fixed an issue that could cause an AWS EC2 Delphix Engine to be left with no network configuration following a change of instance type.
DLPX-80122	Fixed bug that caused disks with write errors in their history to have degraded performance (activated after optional reboot).
DLPX-81081	Fixed bug that would sometimes cause VMs with a lot of MSSQL VDBs to report issues on reboots (activated after optional reboot).
DLPX-81701	Fixed bug that would sometimes cause OS panics when issuing a reboot (activated after optional reboot).
DLPX-82405	Fixed an issue where the NFS server could fail when restarted (activated after optional reboot).

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-46621	The issue with adding concurrent environment that resulted in `exception.executor.object.exists` exception is now resolved.
DLPX-56976	The Delphix engine now returns a descriptive error message if Oracle SnapSync or environment monitor fails to connect to the database in the scenario where the value of `ORACLE_HOME` or `ORACLE_BASE` is set incorrectly in `orabasetab` file.
DLPX-57078	The issue that occurred during Oracle provisioning where if a job is canceled during recovery steps, provisioning will continue until the end of recovery before the job is actually cancelled is now resolved.
DLPX-57934	The issue where large text objects inserted into MDS caused errors and potentially OutOfMemory is now resolved.
DLPX-59179	Warning fault will no longer be raised when linking is done to a source host running Enterprise Edition SQL Server and a staging host is running Standard Edition SQL Server.
DLPX-60981	Fixed issue that prevented customers from reusing disks from old engines to new ones.

Bug number	Description
DLPX-63889	For SAP ASE, pre and post validated sync scripts have been removed. Any existing hooks are converted to pre-hooks list and post-hooks list, respectively.
DLPX-64169	For MSSQL, provided a fix to handle registry exceptions for source environment.
DLPX-70817	Fixed issue that prevented customers from using a disk after running the I/O report card on it.
DLPX-71064	Updated the error action for Lua and Platform plugin upgrade validations.
DLPX-73533	Added fix to prevent multiple connector operations' testing for the presence of the same SCRIPT directory.
DLPX-78787	For MSSQL, added a retryer to get the FQDN of the host.
DLPX-81043	Added additional guidance while doing Replace Certificate on the Upload Certificate step.
DLPX-81532	Standby files for MSSQL V2P operations are now created in the DATA/db folder.
DLPX-81842	Fixed an issue of Arithmetic overflow while fetching database size.
DLPX-82050	Fixed an issue in the intersection of extended replica retention, chained replication, and an entire timeflow getting deleted on the source engine.
DLPX-82145	Fixed an issue where TDE-enabled RAC vPDB provisions may fail with “ORA-28374: typed master key not found in wallet” when activating the key in the target CDB.
DLPX-82262	Disabled Apollo client DevTools to avoid vulnerabilities.
DLPX-82371	Delphix Engine improvements.
DLPX-82433	Fixed an issue where provisioning or refresh of TDE-enabled vPDBs could fail with an internal error if the `tdeKeystoresRootPath` contains any special characters.
DLPX-82514	Password field is now initially empty so that users can enter password.
DLPX-82527	Delphix Engine improvements.
DLPX-82686	Rollback during upgrade now works on Cloud Engines.

Bug number	Description
DLPX-82859	Fixed an issue where API clients that close their connections early while getting a list of snapshots could cause an internal resource leak that could make the application unresponsive if a replication job is initiated afterward.
DLPX-82908	Managed Source Data UI page now correctly displays the Type of Data for AppData sources.
DLPX-82972	Provided the capability to override default root squash behavior for mounted filesystem of unstructured files via Tunable: `LUA_VFILE_TOOLKIT_ROOT_SQUASH_DISABLE` .
DLPX-83103	Fixed an issue of environment refresh failure for Windows host after upgrade to 6.0.16.0 occurred due to iSCSI related registry parameters' monitoring.
DLPX-83105	Fixed an issue following upgrades where SSO entityID changes without user intervention.
DLPX-83148	Fixes environment refresh of an Oracle Live Source environment failure reporting an internal error.
DLPX-83149	From Windows Connector side, while handshaking, stop using the cipher suites which uses Diffie-Hellman key exchange with keys less than 2,048 bits in size.
DLPX-83206	Improved the error message that is displayed when RMAN/sqlplus connection to an Oracle virtual database fails.
DLPX-83359	Delphix Engine improvements.
DLPX-83504	Delphix Engine improvements.
DLPX-83595	The default behavior for VDBs is altered, to disable DBCC CHECKTABLE commands for datafile accessibility.

Release 6.0.16.0 Changes

Bug number	Description
DLPX-67753	Fixed an issue causing redirect responses to reveal server type and version when HTTP redirection is enabled.
DLPX-72068	Improved the way volumes are fetched while working on mounts.

Bug number	Description
DLPX-74396	Fixed an issue that occurred when manually adding a database to an environment which has the same unique name as a database in another environment managed by the same Delphix engine. Previously, Delphix reported an incorrect environment containing the same unique name.
DLPX-80172	Updated the Self-Service refresh warning message.
DLPX-80271	changeArchivelogMode now has an associated job event.
DLPX-80387	Fixed an issue where Oracle move-to-asm script would fail while dropping temp due to tempfiles being in use and unable to be dropped after the database was started.
DLPX-81184	For S3 object store, the "Base URL" input is renamed to "endpoint". The "region" input is now a dropdown for the user to select from a list of standard regions. The endpoint corresponding to that region will now be auto-populated.
DLPX-81692	Fixed an issue where Direct NFS was not being detected for Oracle 21.
DLPX-81996	Fixed an issue where an environment refresh after upgrade did not remove outdated/ obsoleted toolkit components in 6.0.014.0 and 6.0.15.0.
DLPX-82075	Fixed an issue that prevented the creation of a network route whose gateway is reachable through multiple interfaces.
DLPX-82112	Added checks to prevent using NFSv4 with Direct NFS for some Oracle 19 versions that don't support v4 due to an Oracle bug.
DLPX-82236	Fixed an issue where Speculative Logging was being called out of context and could lead to unbounded consumption of rpool.
DLPX-82308	Fixed an issue where the provision/refresh of an Oracle Key Vault vPDB fails with, "ORA-28365: wallet is not open".
DLPX-82329	Action-based alerts now include the success or failure state of the action, including the reason in case of failure.
DLPX-82334	Enabled the creation of on-link network routes; routes whose destination are directly reachable without a gateway.
DLPX-82381	Improved Replication performance on engines with a lot of objects.

Bug number	Description
DLPX-82392	Fixed an issue where after editing credential environment variables, the create/provision VDB wizard would fail because the environment variables were missing the password field in the payload. The required password field has now been added.

Release 6.0.15.0 changes

Fixes that take effect after an optional reboot

Bug number	Description
DLPX-80760	Increased the inotify limit to address a defect during upgrade.

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-68240	Fixed an issue where LogSync for an Oracle RAC standby does not set max number of backup tasks correctly.
DLPX-73375	Added granularity in restore job events to mark various phases of Restore Backup process. Added events in Restore Backup job to provide information at the start of the Redo phase of a restore command.
DLPX-75677	Fixed an issue causing auto population of the encryption key when no input is given.
DLPX-78913	Fixed a minor typing issue when deleting a MSSQL database.
DLPX-79228	Fixed a VDB Start Fault after a VDB start job is successful.
DLPX-79528	Made improvements in environment monitoring to return the NO_DELPHIX_DATABASE status for Staging Push dSources when they are not managed by Delphix, and avoids monitoring other attributes for such databases.
DLPX-79596	Made a change in the enable flow for Staging Push dSources to always make an attempt at unmounting the DATA storage before dropping the staging database.
DLPX-79780	Mount with local_lock=all on Linux, when mounting VDBs with NFSv3.

Bug number	Description
DLPX-79999	Use the offset in the time_zone column of msdb.dbo.backupset to convert the timestamp correctly for a backup from a source, and then continue using the staging host timezone to display the time of a snapshot on the UI.
DLPX-80206	Updated MD5 checksums for Oracle 12.1.0.2.0 OJDBC jars.
DLPX-80254	Fixed an issue where Oracle JDBC jar checksum checks are being reported as false positives despite underlying database connection issues.
DLPX-80406	The Spring framework has been updated due to the Spring4Shell vulnerability.
DLPX-80487	Fixed an issue where the Delphix UI would sometimes not render, showing waiting for response.
DLPX-80494	Added an action item to check for support Host and Server Type combination while adding MSSQL environments.
DLPX-80619	This change will introduce the ability to add sporadic failures via tunable: ADDITIONAL_SPORADIC_FAILURES.
DLPX-80909	Cross-Site Scripting (Reflected) in /resources/json/delphix/session.
DLPX-81048	Removed the requirement for Linux kernel recover-lost-locks setting when using NFSv4 with Oracle dNFS.
DLPX-81090	Can now enable/disable SNMPv3 vs. v1/v2.
DLPX-81100	Updated the command for checking database files accessibility while fetching the VDBs status with a lighter and less intrusive command, to get relief from VDBs being stopped randomly.
DLPX-81242	Fixed an issue preventing the upgrade of a replication Continuous Vault source with automatic replication.
DLPX-81308	Fixed an issue causing Appdata SnapSync to crash with NullPointerException when a virtual database is not successfully refreshed or rollbacked.
DLPX-81358	Fixed the unnecessary alerts of timezone discovery failure that users were facing randomly for the cluster environments.
DLPX-81502	Improved performance of the Replication page.

Bug number	Description
DLPX-81696	Users should now be able to enable the feature flag AZURE_DATA_BANK.
DLPX-81710	Fixed an issue where an engine could not be setup when objectStorage is enabled.

Release 6.0.14.0 changes

Security Fixes

Bug number	Bug Introduced in	Description	Security bulletin
DLPX-81059	5.2.2.0	Arbitrary Code Execution may be performed when configuring masking environments	TB098

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-38908	Oracle LogSync should automatically resolve faults for transient issues
DLPX-39193	Null Pointer Exception in Oracle LogSync backup stream handler
DLPX-57078	Job cancel requests during Oracle provisioning are not processed until the end of recovery
DLPX-57405	move-to-asm.sh does not support TDE-enabled databases
DLPX-62343	disable the OPTIONS method for all HTTP(S) requests
DLPX-64386	Oracle LogSync thread may hang when trying to remove temporary RMAN command file from source host toolkit directory
DLPX-65413	Ensure "source-archive" directory is unmounted at start of Oracle Provision
DLPX-66879	Oracle LogSync can create orphaned logs in certain scenarios
DLPX-69453	Provide tunable for Oracle LogSync client timeout

Bug number	Description
DLPX-69802	Common Toolkit directory is not removed from a mounted shared NFS location when the environment is deleted
DLPX-76382	Force disable should succeed despite environmental problems
DLPX-77840	Fixed an issue on the Setup pane to allow successful completion of an smtp test against a specific email address
DLPX-78412	SCM/Talaria failure reason should be communicated in the Delphix fault warnings
DLPX-78726	Removing windows environment performs cleanup of iSCSI persistent login target
DLPX-78754	Disable operation is prohibited on replicated sources
DLPX-78986	Prevent DSP connections for disabled Oracle RAC cluster nodes
DLPX-79077	Resolved an issue of an infinite spinner when validating BEQ credentials for Oracle dSource with duplicate unique_name. While linking a dSource, credentials are now required when discovering an unknown CDB.
DLPX-79242	Splunk HEC token logged to debug logs during splunkHec test
DLPX-79396	Allow users to unset Oracle database user name and credentials through CLI if Simplified Connection Management is enabled.
DLPX-79502	SnapSync fails if more than 1000 tempfiles exist in the whole CDB
DLPX-79591	Changed the NFSv4 minimum supported target Redhat version to 6.4 (was previously 6.3).
DLPX-79742	Unable to provision PostgreSQL VDB to Linux host with processor type of ppc64le
DLPX-79823	Improved the action item for failure to enable/attach the staging push dSource.
DLPX-79942	Improved the action item for failure to enable/attach the staging push dSource.
DLPX-80137	Limit SNMP configuration access to sysadmin
DLPX-80144	Oracle SnapSync crashes with NullPointerException when a PDB dSource is renamed and replaced with a new PDB of the same name

Bug number	Description
DLPX-80217	Fixed issue with filename conflicts during source backup restore
DLPX-80302	It was necessary to restart the auxiliary CDB during a TDE provision after recreating the autologin keystore
DLPX-80369	Oracle environment monitor triggers fault.oracle.db.connection.failed fault immediately on dataset stop or disable.
DLPX-80415	Fixes long delay in operations such as VDB start/stop when JDBC Thin connection to database fails due to unable to establish network connection.
DLPX-80439	Provide mount location to upgrade scripts during Lua to Python upgrade process if mount specification has not been provided.
DLPX-80440	Fixed spinner issue while provisioning a VDB from a SQL Server staging dSource from Provision VDB option in datasets menu
DLPX-80482	TDE-enabled provisions to a RAC target fail with "ORA-28365: wallet is not open" while attempting to reopen the database in start_database.sh in the auxiliary
DLPX-80483	Fix failing TDE-enabled vPDB provisions to a linked RAC container database due to "ORA-28365: wallet is not open" errors
DLPX-80487	Improved performance across dataset and replication related pages.

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-80078	The issue with removing files with complex file permissions on EBS is now fixed

Release 6.0.13.1 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-80818	libc upgrade necessitates PostgreSQL re-index.

Release 6.0.13.0 changes

Security Fixes

Bug number	Bug Introduced in	Description	Security Bulletin
DLPX-79789	5.3.0.0	Arbitrary Code Execution May Be Performed by Engine System Administrators.	TB096

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-7868	The issue with the confusing vPDB error on a snapshot after resetlogs of a linked CDB is now fixed.
DLPX-41671	You can now update the Oracle cluster home through the user interface.
DLPX-43467	When dSource is an Oracle standby in RTA mode, LogSync was raising the fault.oracle.linkedsource.log.conflict error and getting disabled on its own. This issue is now fixed.
DLPX-50309	Users can now change logSyncInterval for Oracle dsources.
DLPX-59757	The count for masking jobs fetched from the Masking Engine is now configurable. By default, it is set to 500.
DLPX-67604	The manual recovery of a database after V2P from a snapshot of dSource was failing with an error. This issue is now fixed.
DLPX-68684	The self-signed certificate is now compliant with the requirements for trusted certificates in MacOS 10.15.
DLPX-74613	Oracle VDB migration check needs to be done against the Oracle target host instead of the source. Furthermore, the Error and Action plan provided should include the target hostname.

Bug number	Description
DLPX-75467	The CLI now returns correct and descriptive error messages when executing unauthorized requests on uninitialized engines.
DLPX-75646	To diagnose BEQ connection failure, this release adds MD5 checksums for ojdbc*.jar for Oracle release versions up to currently supported release version.
DLPX-75878	The issue with the JDBC connection string for an Oracle vPDB not getting updated after an IP address change is now fixed.
DLPX-75989	The issue with the failure of environment discovery of an Oracle Cluster with a NullPointerException error is now fixed.
DLPX-76956	Previously, the Oracle JDBC test connection with the wrong password was increasing the `LCOUNT` value by more than 1. This issue is now fixed.
DLPX-77140	This release now speeds up metadata that is sent during replication when Extended retention is involved.
DLPX-77231	Previously, when source discontinuity on the dSource was followed by resync on the livesource, one or more livesource workers were failing to start. This prevented livesource status from getting updated and the first snapshot from being taken after resync. This issue is now fixed.
DLPX-77600	This release fixes NPE when Linking dSource with missing backup and unreachable nodes.
DLPX-77880	This release improves scalability for engines with an extremely large number of snapshots that were causing them to run out of memory.
DLPX-78015	Previously, V2P export with absolute data files was failing with an internal error. This issue is now fixed.
DLPX-78174	Insecure DES is no longer supported for SNMPv3.

Bug number	Description
DLPX-78420	This release adds V2P support for Windows server 2022 host machines.
DLPX-78473	This release improves load times for Datasets and Dataset Performance pages for engines with a large number of datasets and containers.
DLPX-78488	Previously, when switching from the backup server to ASE, dump history was not working for dSources configured to use the remote backup server. This issue is now fixed.
DLPX-78594	This release fixes an issue with disabled VDBs not being able to undo a refresh.
DLPX-78688	TLS 1.0 and TLS 1.1 ciphers are no longer available. Any system that is only configured with TLS 1.0 or TLS 1.1 ciphers is switched to use the default cipher set.
DLPX-78693	Invalid sync parameters will not cause DE server unavailability.
DLPX-78696	Switching sync strategy from source sync strategy type to sourceless strategy type is not allowed.
DLPX-79126	VDBs can now be automatically started with the tunables if stopped intermittently because Windows fail to write on the mount (Msg 9001).
DLPX-79292	This release eases restrictions on taking a snapshot of PDBs with encrypted UNDO tablespaces.
DLPX-79344	Previously, Snapsync of a standby PDB in mount mode was failing with the `ORA-01109: database not open` error message. This issue is now fixed.
DLPX-79422	Previously, clicking on a Replication Profile was resulting in the following error message `An error happened while communicating with the server`. This issue is now fixed.

Bug number	Description
DLPX-79789	Under certain conditions, arbitrary code execution may be performed by sysadmins.
DLPX-79808	This release fixes failures if the VDB name is more than 68 characters.

Release 6.0.12.1 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-79151	The issue with remote syslog configurations preventing engine upgrades or virtualization service restarts is now fixed.

Release 6.0.12.0 changes

Log4j Updates

Based on detailed testing and analysis, all the currently supported products are not susceptible to known log4j vulnerabilities. Please refer to TB095 Technical Bulletin for more information. All instances of log4j in currently supported Delphix products are updated to **log4j 2.17.1** as of this release.

Delphix keeps you updated on the latest developments and keeps releasing hotfixes, procedures, and workarounds for such critical vulnerabilities. For more information on how Delphix supports our product and customers in such cases, see Delphix Product Security

For more information, refer to the following pages:

- TB095 log4j vulnerabilities
- Uninstalling the delphix connector service from the target database servers
- Delphix product lifecycle policies
- Product security

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-23068	Validation for target database parameter 'DB_FILES' for single-tenant databases for the following operations: provisioning, refresh, rewind, and converting a dSource to LiveSource is now added. Furthermore, specific error messages for handling ORA-00059 errors are added.
DLPX-44544	The issue with the SnapSync of an Oracle standby dSource in Real-Time Apply mode calculating the snapshot's timestamp incorrectly is now fixed. This issue was resulting in ORA-01194 or ORA-01152 errors when provisioning to a timestamp after the snapshot.
DLPX-56691	The issue with the data files of a VDB getting unmounted when provisioning, refresh, or rollback job is canceled manually is now fixed.
DLPX-57971	The issue with the latest snapshot of a LiveSource taking a long time to show the SCN/timestamp range on its card in the GUI is now fixed.
DLPX-60320	UI now allows the selection of older dataset repositories (downgrade) in addition to selecting newer ones (upgrade).
DLPX-67069	The issue with the stopped Oracle VDB monitoring by the environment monitor that resulted in connection errors flooding the debug log is now fixed.
DLPX-68132	The issue with the “Copy query to clipboard” SQL copy functionality in the “Managed Source Data” is now fixed to have correct apostrophe characters.
DLPX-72123	The issue with the failure of detaching or deleting an Oracle dSource operation on RAC environments (This issue was occurring due to failure of deletion of RMAN backups on RAC and the operation needs to be retried with a force option) is now fixed.

Bug number	Description
DLPX-72779	The UI showing enabled or disabled for cluster nodes now uses a grid table. The Enabled column contains a checkmark to show whether the cluster is enabled or disabled. Users are able to select or unselect the checkmark to enable or disable a cluster.
DLPX-73975	Critical storage faults should not be ignorable nor manually resolvable
DLPX-74862	This release fixes an issue where the RESUME operation failed without any error thrown to the user on dSources where ENFORCE was still in progress. The fix will make sure that even if RESUME fails, it throws a DUE to the user with suggested actions to resolve the issue.
DLPX-75763	The issue with the failure of refreshing a VDB provisioned as an empty vfiles since there is no parent container to refresh from is now fixed.
DLPX-76266	The issue with the VDB Disable operation that resulted in an error message while connectivity with the host machine can't be established is now fixed.
DLPX-77123	You can now run Upgrade Verify when another upgrade is in progress.
DLPX-77347	This release fixes the difference in time shown for MSSQL snapshot based on different database authentication methods.
DLPX-77638	The issue with the failure of the End Entity Certificate expiration fault is now fixed.
DLPX-77664	The issue with the failure of the Oracle SnapSync with an error message, "RMAN-06183: datafile or datafile copy (file number) larger than MAXSETSIZE" if a datafile resized in the middle of SnapSync is now fixed.
DLPX-77913	The issue with the faults table missing data if there was more than one page of faults is now fixed.

Bug number	Description
DLPX-77925	The issue with the unsupported Windows release error message is now fixed.
DLPX-78113	MSSQL VDB database size will now be refreshed periodically based on environment_monitor.dynamic_attributes_check_period tunable.
DLPX-78183	The issue with the MSSQL validated sync schedule not getting updated without successful backup restoration is now fixed.
DLPX-78244	The issue with the failure of a few operations on self-service containers due to incorrect entries corresponding to the Oracle log metadata on the Delphix engine is now fixed.
DLPX-78258	The issue with the input bug that retained cleared out DB credentials in the dSource linking wizard is now fixed.
DLPX-78263	The issue with the failure of a SnapSync of an Oracle standby dSource in Real-Time Apply mode with an error message, "exception.oracle.snl.linkedsource.current_scn.invalid" if the rate of change in the database is low is now fixed.
DLPX-78265	Offline Oracle bystander PDBs data files can now be optimized leading to improved provision performance.
DLPX-78309	The issue with the CLI being unable to log in to system users when the main virtualization service is down is now fixed.
DLPX-78334	The issue with a large number of missing Oracle archive logs causing an error while viewing dataset is now fixed.
DLPX-78392	Hosts running Windows Server 2022 can now be added as Source and Target environments to the Delphix Engine.

Bug number	Description
DLPX-78522	SSLv3, TLS 1.0, and TLS 1.1 are no longer configurations options for HTTPS. Any system configured only with these removed options will be automatically set to use TLS 1.2.
DLPX-78791	This release upgrades log4j from 1.2.17 to the latest 2.x in Windows Connector.
DLPX-78938	This release upgrades log4j in virtualization to 2.17.1.

Release 6.0.11.0 changes

Security Fixes

Bug number	Bug Introduced in	Description	Security Bulletin
DLPX-77921	6.0.8.0	Arbitrary Code Execution by Delphix System Administrators may be Performed on Virtualization and Masking Engines	TB094

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-53019	The issue with the missing redo alert raised during the environment monitor check has now been resolved.
DLPX-59299	Discovery and monitoring rely on "Connected" in sqlplus output, which may not be the case if NLS_LANG is set to another language (e.g. Japanese). Downstream operations, like linking or provisioning, may then fail due to missing user privileges.
DLPX-59662	The issue with copy-Only Backups failure with Virtual Service Accounts has now been resolved.

Bug number	Description
DLPX-62706	The issue with the Hostchecker not properly checking /home/delphix permissions has now been resolved.
DLPX-64082	The issue with Oracle Provisioning scripts having hard-coded timeout issues has now been resolved.
DLPX-65729	Added retry functionality to the 'read backup files' operation during a validated sync to an account for an unstable environment.
DLPX-69778	SAML response is not logged on successful SSO login.
DLPX-72043	The issue where LiteSpeed `xp_restore_headeronly` stored procedure failure message are displayed when validated sync is active for dSources with LiteSpeed backup has now been resolved.
DLPX-72220	A UI issue that occurred while updating the vault when only the private key is changed has now been resolved.
DLPX-72225	Admin user created from management UI is no longer showing as 'non-admin' type.
DLPX-72237	The 'Verify Credentials' button from the DSP Throughput test page is now removed.
DLPX-72369	The dependency on a parent snapshot relying on the latest snapshot is now removed if a parent snapshot does not exist during the VDB enable operation.
DLPX-72778	Oracle dSource attach operation with changed DB ID using 'Force' option is now allowed.
DLPX-74555	Updated the "no Delphix connector" message while provisioning a Windows source environment.
DLPX-74676	Oracle LiveSource LogSync should only catalog valid archive log files.

Bug number	Description
DLPX-74851	In the Add Environment GUI, the mouseover information for "Set Delphix Session Protocol Options (DSP)" has been currented.
DLPX-74896	The race condition issue when running Oracle VDB refresh and dSource snap sync resulting in incorrect MDS entry for the parent snapshot of a VDB in the `dlpx_timeflow` table has now been resolved.
DLPX-75335	Added a product name and product version for the Delphix Connector executable so this information can be available before installation.
DLPX-75500	For ag cluster nodes, if the refresh fails due to timezone discovery failure, don't delete the nodes from MDS as it doesn't mean we had an issue with the nodes.
DLPX-75952	Database configs will be replicated only if the associated VDB is replicated.
DLPX-75995	The issue causing environment 'Add' or 'Refresh' to fail when PowerShell Transcription is enabled has now been resolved.
DLPX-76244	The issue where TCP fallback connection to database stops responding if the Oracle database instance is down has now been resolved.
DLPX-76290	Databases of UNKNOWN cdb type are now included in the attachment of a non-PDB container.
DLPX-76731	Added Delphix support for `WALLET_ROOT` and `TDE_CONFIGURATION` parameters to manage wallets in 19c instead of sqlnet.ora.
DLPX-76759	Added "Response" to faults along with other details when logged in the Admin App.
DLPX-76777	Remove orphaned Oracle logs resulting from archive log fetch timeouts.

Bug number	Description
DLPX-76793	Added execution timeout for execution of `UpdateFileACL.ps1`.
DLPX-76974	The issue where a user was unable to change 'from address' of SMTP server to noreply@delphix.com in the GUI has now been resolved.
DLPX-77112	The issue where an Oracle VDB cannot be provisioned between different minor versions if the Source is on higher RU has now been resolved.
DLPX-77284	The issue where after a hotfix was removed due to a successful upgrade, the system would still indicate the hotfix was installed post-upgrade has now been resolved.
DLPX-77345	The issue where provisioning a vVDB fails with `java.lang.OutOfMemoryError` when sqlplus is used to rename the datafiles has now been resolved.
DLPX-77405	Replicated password vaults will no longer be visible in the UI.
DLPX-77676	The issue where provisioning a vPDB from a PDB dSource fails with "ORA-65114: space usage in container is too high" if PDB `max_size/ max_pdb_storage` is configured has now been resolved.
DLPX-77708	The issue where a refresh/disable/destroy of a VDB using NFSv3 could cause loss of access to other VDBs that were using NFSv3 has now been resolved.
DLPX-77844	The issue where V2P operations from a VDB snapshot would result in the deletion of any production datafiles that exist on specified V2P target directory has now been resolved.
DLPX-77904	Removed 'Factory Reset' for Delphix Engines that are Data Vaults, as the operation is disabled for those engines.

Bug number	Description
DLPX-77912	The issue that can cause a VDB stop, refresh, or rollback to fail with an internal error has been resolved.
ORB-2465	Removed the requirement that SAML SSO email addresses must match case-sensitively for SSO logins.

Fixes that take effect after an optional reboot

Bug number	Description
DLPX-77577	Increased nvme I/O timeout to prevent storage issues in EC2 (Activated after optional Reboot).

Release 6.0.10.1 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-77467	Loading the setup app dashboard (as sysadmin) was rendering a server error popup with instruction to contact Delphix Support. This 6.0.10.0 error has been known to impair the ability to configure web proxy, PhoneHome, SMTP servers, and other connectivity settings via the GUI. It has now been resolved.

Release 6.0.10.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-18438	The issue with provisioning to the latest available time that resulted in generating the `exception.oracle.target.point.not_provisionable` exception has now been resolved.

Bug number	Description
DLPX-35480	Previously, static routes were being added using the add command. Now, the same can be added using the create command.
DLPX-57516	The issue with the failure of management service to start after configuring some abbreviated timezones from a picklist in server setup or sysadmin CLI has now been resolved.
DLPX-58133	Previously, the Oracle SnapSync operation was resulting in a warning for BCT usage on editions that do not provide it. This issue has now been resolved.
DLPX-58675	The issue with the deletion of the last snapshot on timeflow by Retention during a failed Oracle DB_SYNC operation has now been resolved.
DLPX-63003	The issue with memory being exhausted while reading too many snapshots from MDS has now been resolved.
DLPX-63347	If the staging source has the "Use as Staging" flag set as off, the user was seeing a specific exception while trying to enable a linked dSource to point in the direction of what needs to be done. Any compatibility failure will now have a specific exception.
DLPX-63601	Previously, querying the following operation "backupset table" and "whether a database is part of AG or not" was resulting in deadlocks and lock timeouts errors. We have now added retries to resolve the issue.
DLPX-64369	The issue with throwing `fault.oracle.linkedsource.incomplete.tempfile` for physical standby has now been resolved.
DLPX-65949	The issue with misleading status in the progress bar while taking a copy-only backup has now been resolved.

Bug number	Description
DLPX-69831	Previously, the Oracle dSource SnapSync operation was not displaying a clear failure message if a dSource `db_unique_name` is changed. This issue has now been resolved.
DLPX-70317	The issue with the restarting of the NFS-server by the reaper thread while deleting a vPDB from a linked CDB with Talaria turned on has now been resolved.
DLPX-71018	The issue with UI displaying only the suffix of the device name used by Hyper-V has now been resolved. UI now displays a unique device name for storage in Hyper-V.
DLPX-71292	The issue with the allowance of incremental SnapSync after LogSync throws `fault.oracle.linkedsource.log.conflict` has now been resolved.
DLPX-71639	The NFSv4 is now set as the default option when mounting datasets from OS platforms that support it.
DLPX-71769	The need to set permissions of `$ORACLE_HOME/dbs` subdirectories using STARTUP SPFILE syntax is now removed.
DLPX-72011	The issue with the CLI network setup not configuring the first network interface when multiple interfaces exist has now been resolved.
DLPX-72186	The issue with CDB log file retention working incorrectly if a PDB has multiple time flows pointing to the same CDB timeflow has now been resolved.
DLPX-72432	The format of `zpool_iostat_60.log` has been enhanced in this release. A timestamp is recorded for each sample in the log, making it easier to determine the time for each sample.
DLPX-72956	The issue with disabling the Oracle LogSync after running the validated sync job has now been resolved.

Bug number	Description
DLPX-73575	The timezone monitoring is now added for the Windows hosts.
DLPX-73590	You can now refresh a VDB whose parent dataset is in a different group without needing authorization on the parent or its group.
DLPX-73800	The issue with the failure of olsnodes when run as a non-Oracle user has now been resolved.
DLPX-74504	The issue with throwing a new DUE and NotFoundException when ojdbc libs cannot be read has now been resolved.
DLPX-74945	UI now displays a detailed error message for transaction log-chain break fault.
DLPX-74992	The issue with the failure of SnapSync operation when Database incarnation reset-logs end time is changed from "2021-03-13 22:03:07.0" to "2021-03-13 21:03:07.0" for virtual pluggable database “ " has="" now="" been="">">
DLPX-75389	The issue with the recording of the insufficient details by Logsync when dbid change was detected has now been resolved.
DLPX-75517	The issue with the failure of the Oracle vPDB provisioning with the "ORA-00959: tablespace 'TEMP' does not exist" error has now been resolved.
DLPX-75721	The issue with the failure of an Environment discovery with the "DelphixFatalException: Unknown Oracle Database status: REFRESHING" error has now been resolved.
DLPX-75737	The issue with saving unnecessary logs by Retention if bookmark falls exactly on a snapshot end SCN or snapshot end timestamp has now been resolved.
DLPX-75897	Previously, failure to start I/O services after the upgrade operation was resulting in a stack restart loop. This issue has now been resolved.

Bug number	Description
DLPX-75951	The internal error being signaled during VDB SnapSync by `removeUnneededZFSFiles` when a data file is physically removed during processing by an external cause has now been resolved.
DLPX-76288	The NFS latency for workloads involving a lot of parallel I/O (e.g. Oracle VDBs with concurrent accesses to many data files) is now improved.
DLPX-76388	Previously, entering key pairs directly into hook environment variables, as opposed to via a vault or as passwords were resulting in an internal error. This issue has now been resolved.
DLPX-76406	The issue with NFS-based VDBs becoming unresponsive has now been resolved.
DLPX-76447	The issue with the V2P Functionality to customize target directory structure for exporting database files to separate file systems not working as documented has now been resolved.
DLPX-76613	The issue with the unnecessary accumulation of heap when validated sync is active for dSources using an environment user that eventually can cause out of memory issues has now been resolved.
DLPX-76690	The issue with the removal of extraneous Oracle data files while creating snapshots has now been resolved. The extraneous Oracle data files are now removed during the provisioning operation.
DLPX-76692	For V2P operation, we now use the unbuffered copy method for better performance.
DLPX-76718	The issue with the creation of the extraneous Self-Service branch segments during the replication operation has now been resolved.
DLPX-76760	Environment clusters will now show faults from their child nodes.

Bug number	Description
DLPX-76802	Previously, the engines that are in the DEFERRED upgrade state were resulting in the "Large Receive Offload" option turned off which was leading to performance degradation in network transfers. This issue has now been resolved. Upgrading the engines that are in the DEFERRED state will also resolve the issue. Screen reader support is now enabled.
DLPX-76891	CRON expression labels now ask for Quartz format on the user interface.

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-76619	The NFS reliance on DNS to prevent VDB unresponsiveness related to DNS unreliability is now reduced.
DLPX-76203	The NFS latency for engines with many Oracle dNFS clients is now improved.
DLPX-76119	The issue with the Delphix Engine crashing or becoming unresponsive when canceling a replication job has now been resolved.
DLPX-76991	Optimized in-memory cache eviction by making minor improvements to I/O performance.

Release 6.0.9.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-28435	MS SQL instances PatchLevel will be displayed in preference to the version on the UI.

Bug number	Description
DLPX-40005	Improved the error message that is displayed when a JDBC connection cannot be established or the Oracle database Instance is unavailable.
DLPX-48080	Oracle Home Check may generate spurious faults if Oracle Home entry does not exist in /etc/oratab.
DLPX-55951	Attempting to provision a plugin-based VDB onto an incompatible OS (Windows to Linux or vice versa) is possible in the UI, and would fail with a crash requiring a restart. Now an informative error message is shown after the attempt is made.
DLPX-59613	Fixed creation of retention policies workers on replica objects after failover.
DLPX-64307	Environment refresh should ignore cluster discovery for Oracle VDBs.
DLPX-66191	Fixed the side-effects of the native Windows "Recent Files" behavior, when large numbers of PowerShell operations are being run concurrently.
DLPX-67537	Domain administrators can now create, view, and edit the alert profiles of other domain users.
DLPX-69605	Poor error message when selecting Timeflow range.
DLPX-70502	On detaching a dSource, delete backup server entry from MDS and related ones if the backup server is unused.
DLPX-71002	In case we have null values coming for recovery_model from msdb.dbo.backupset table, the user will see a generic exception for manual sync and a fault for validated sync.
DLPX-71908	Users will now see a warning when they remove any object from the replication specification list.
DLPX-72012	Prevents the same IP address from being configured on more than one network interface.

Bug number	Description
DLPX-72411	Environment names for Windows and Oracle Clusters are once again editable by users.
DLPX-72609	When we do MSSQL standalone environment discovery, the user will see a warning for databases attached to AG that are present and will not be discovered unless cluster environment discovery is selected.
DLPX-72695	Improved Oracle SnapSync performance by eliminating unnecessary calls to getTotalHoleBlocks.
DLPX-73409	Duplicate listener entry gets generated in MDS if Oracle listener is manually started with a non-uppercase name.
DLPX-73586	Fixed a display error of some snapshot names in the command line interface which showed references instead of actual names.
DLPX-73720	Provisioning an Oracle vPDB fails with "ORA-65149: PDB name conflicts with existing service name in the CDB or the PDB" if the PDB and CDB names are the same.
DLPX-74050	Environment names for Windows and Oracle Clusters are once again editable by users.
DLPX-74078	The Target Directory path is combined with other directories such as Data Directory, Archive Directory, Temp Directory, etc to build the full path for data files, archive logs, temp files, etc. As long as the combined paths are valid the V2P job proceeds.
DLPX-74197	Fixes a misleading warning about insufficient space on a replication target.
DLPX-74201	Snapshots created as a result of refresh or rewind operations will now be labeled as just "Snapshot" to avoid confusion. Users are advised to look at the Timeflow markers to know when the Timeflow operation was performed.

Bug number	Description
DLPX-74367	Delphix Engine repeatedly reports "Failed to parse logfile".
DLPX-74377	Improved diagnostics information for the case when Delphix Engine fails to connect to the Windows host.
DLPX-74387	fix an issue that causes the management service to crash under heavy CPU load.
DLPX-74398	Added handling of dangling nodes during Windows cluster environment add and refresh operations.
DLPX-74486	Delphix OS users cannot provision 12.2 TDE vPDB due to directory permissions in the default wallet location.
DLPX-74495	Enabled more logging in Delphix connector logs for timeouts.
DLPX-74681	During RAC vPDB provision, Oracle 19.9 target CDB crashes with ORA-00600 [krccfl_chunk] when BCT is enabled.
DLPX-74806	Improved error message displayed when the storage device initialization fails unexpectedly.
DLPX-74860	Provisioning the 2nd generation VDB fails if the dSource has imported read-only transportable tablespaces fails.
DLPX-74975	Allow adding invalid or unreachable paths as a shared backup location for dSources.
DLPX-75026	Invalid JDBC connections are not purged from the connection pool when the home is changed.
DLPX-75134	Improved performance for the Environment Databases pages when there are a lot of databases.
DLPX-75363	Update exception description is seen when the SnapSync fails for ASE encrypted database.

Bug number	Description
DLPX-75401	local listener set to null if oracle.lsnr.protocol_registration_order is quoted.
DLPX-75506	Fix a bug that can cause Oracle RAC VDBs to fail with stale NFSv3 mounts if NFSv4 is also enabled.
DLPX-75532	Insufficient heap memory settings on AIX cause connector and SnapSync to hang or crash.
DLPX-75663	FIPS compliant algorithms will be used while merging the old and new toolkit directories during environment refresh.
DLPX-75716	Delphix may remove Oracle VDB temp tablespaces during Snapsync.
DLPX-75735	Fixed creation of retention policies workers on replica objects after failover.
DLPX-75834	Rearranged Syslog configuration dialog inputs to avoid confusion and have a more consistent user experience.
DLPX-75858	BEQ processes can hold on to file descriptors leading to hook scripts hanging after upgrading to 6.0.7.0.
DLPX-76018	Remove hardcoded 5-minute timeout for doDropPDBKeepDatafiles.sh.
DLPX-76140	TDE SnapSync should ignore WARNING plugin violations.

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-71980	Fix a bug that was causing the Delphix Engine storage pool to fail to import on boot under certain circumstances.

Release 6.0.8.1 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-75804	Some Delphix operations may fail if mount and umount commands, on staging or target hosts, are setup to run as sudo and if sudo rules prohibit these commands from running with unrecognized options. The issue is fixed now after removing "-v" added in 6.0.8.0.

Release 6.0.8.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-49694	Skip parsing of DBCC when code 0 is not present in the DBCC page output. In case the DBCC page has code 0 but not dbid, use bdbid (present in the buffer section).
DLPX-68764	VDB SnapShot does not progress if ASE database devices are not on Delphix storage, environment monitoring raises a fault. Subsequent VDB operations like enable, disable, start, stop, delete, snapshot, or refresh will fail.
DLPX-70793	Delphix Engine should not allow linking Oracle DB with null `db_unique_name`.
DLPX-71300	For newer ASE versions (>= 15.7 SP138 and 16.0 SP02 PL05 and ASE 16.0 SP03), do not run DBCC PAGE anymore, as it was an identifier for DBCC CHECKALLOC that is already not run.
DLPX-71471	Error message asking user to manually perform disable/enable operation or correctly rename the target database back will be displayed during Start VDB, if VDB does not exist.
DLPX-71687	Provide a mechanism to enable VDBs up to filesystem mount point.

Bug number	Description
DLPX-71875	Fixed a bug that results in a memory reservation not being represented correctly in the Delphix API.
DLPX-72046	Deletion of vPDB in a vCDB shows this warning, "Encountered an error while shutting down and cleaning up Oracle files."
DLPX-72209	Downloading a support bundle is not supported at the same time that an upload of an upgrade image has been initiated by the same Delphix user.
DLPX-72319	Fixed an issue where some error dialogs would freeze in Internet Explorer 11.
DLPX-72705	Connection timeout when deleting remote shipper script can cause a timeout in LogSync client.
DLPX-72757	ASE sync using Dump History fails for large dump history files.
DLPX-72780	Timezone is set incorrectly for snapshots of Solaris 10 dSources and VDBs.
DLPX-72904	Storage capacity now includes usage from all file system objects, not just snapshots.
DLPX-73143	Fixed an issue where the support bundle dialog showed a loading spinner intermittently while jobs were running.
DLPX-73354	Traverse all shared backup locations while syncing, even if some of the paths are invalid or not reachable.
DLPX-73489	Fixed bug where adding or editing a parameter using the UI VDB Config Template "Text" tab was truncating the parameter's value.
DLPX-73602	Incorrect mount options used when a single instance RAC is linked as a standalone single instance.
DLPX-73607	Added paging for days with large numbers of snapshots to prevent slowdown.

Bug number	Description
DLPX-73623	Fixed an out-of-memory condition that occurs in SSH tunneling for encrypted log-syncs when storage latencies are high.
DLPX-73627	The help text on upgrade replication warnings have been updated to avoid confusion between Ignored and Resolved.
DLPX-73668	Fixed Missing security headers.
DLPX-73669	Cross-site request forgery (CSRF) issue in management UI.
DLPX-73727	Fixed an issue where the faults table was unable to navigate to other pages.
DLPX-73797	Fixed VDB refresh failures due to SQL Server Error 924 after setting VDB to single user mode.
DLPX-74025	Implemented logic to retry offline database along with a drop database to overcome deadlocks while off-lining or dropping the database.
DLPX-74029	VMware Hot-Add memory is not immediately reflected in the system API.
DLPX-74057	Fixed a typo in "Download Support Bundle" UI component where the word "suport" was missing a "p".
DLPX-74254	Ownership of files inside VDB now matches new owner when VDB owner is changed.
DLPX-74298	Fixed an issue where the user could not upload a keystore with a blank keystore passcode.
DLPX-74362	Fixes an issue with namespace deletion when the replication receive jobs have been cleaned up.
DLPX-74442	VDB Enable with attemptStart=false will now mount the datasets so that VDB can be started.

Bug number	Description
DLPX-74457	Cluster discovery for Oracle RAC clusters are partially failing on Solaris 10.
DLPX-74529	Fixed a bug so that an upgrade completes even when jobs fail.
DLPX-74542	Fixed a bug so that upgrade completion is properly handled after kernel upgrades.
DLPX-74645	Delphix Engine uses the uptime command to keep track of a target host reboot and auto start VDBs on the host. In some cases, the output of this command is not what is expected and causes unintended restart of a stopped VDB. This issue is now fixed.
DLPX-74656	Oracle errors during doCreateSPFile.sh are not captured.
DLPX-74704	Fixed a bug where the Dataset scroll does not extend to the bottom of a dataset list, thus truncating the status of the last dataset in the expanded group.
DLPX-74883	Prevent support bundle collection from cancelling replication.
DLPX-74911	Talaria TCP fallback fault may be misconstrued if an Oracle RAC node is down.
DLPX-74997	Prevent granting replicated roles to users.
DLPX-75083	Post upgrade cleanup task may become unresponsive while attempting a migration from 5.3.x to 6.0.x due to several threads stuck in WAITING state.
DLPX-75095	Provisioning an Oracle VDB fails if change-archivelog-mode.sh takes longer than 5 minutes.
DLPX-75134	Improved performance of the Environment Databases page under certain conditions.
DLPX-75188	Fixed "out of memory" issue when processing a large number of objects on the Target engine.

Bug number	Description
DLPX-75204	Addressed a performance issue on the Target engine when receiving large number of replicated objects.
DLPX-75208	Snapshot names are incorrectly redacted in the MDS dlpx_action table in support bundles.
DLPX-75416	Fixed a replication issue when there are sources with TLS enabled.

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-72065	Fixed a bug that can cause a Windows iSCSI initiator to fail connect to the Delphix Engine.
DLPX-72681	Console Delphix status screen shows a Python stack trace if the system is configured with a static IP address.
DLPX-73423	Console Delphix status screen shows a Python stack trace if the system has no default route.
DLPX-74216	Fixed an issue that causes management service failures in low memory situations.
DLPX-74622	Fixed a bug that can cause a replication job to fail with an internal error.
DLPX-75089	Fixed a bug that can cause NFSv3 clients to lose locks during upgrade verification.
DLPX-75524	Fixed a bug that can lead to Oracle data corruption when running VDBs on Oracle 19c with dNFS.

Release 6.0.7.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-39006	LogSync failed with "Cannot read archived log due to failure of log shipping script".
DLPX-39245	Fixed a bug that caused the management service to become inaccessible if the storage pool ran out of space.
DLPX-59155	Provisioning a VDB or vPDB failed with unclear error message 'A database with the name "xxx" already exists'.
DLPX-60317	Fixed Out of Memory issue when replicating a large number of objects.
DLPX-60947	Replica VDBs will be updated when performing a point-in-time restore.
DLPX-62805	vPDB provision did not raise an error when a non-provisionable target point-in-time was provided.
DLPX-62969	Fixed Out of Memory issue when receiving large number of replicated objects.
DLPX-64600	Skipped connecting to ASE dSources during SnapSync policy runs as it is not applicable for them, hence prevent recurrent faults that the policy throws for connectivity issues.
DLPX-67363	Maximum identify provider authentication time age can be customized for single sign-on.
DLPX-67607	Fix to make Snapsync throw exception if manifest file is missing or of 0 bytes instead of internal error with null pointer exception.
DLPX-67767	Fixed a bug that caused the upgrade to hang, while waiting for running jobs to finish.

Bug number	Description
DLPX-70821	Allow the entity id for SAML single sign-on to be a URL for compatibility with Azure AD.
DLPX-71783	doRenameDatafiles cleanup of extra files fails due to file permissions mismatch.
DLPX-72010	Fixed an issue that prevents changing the default gateway using the network setup CLI.
DLPX-72075	Maximum SAML response time skew can be customized for single sign-on.
DLPX-72191	Oracle privilege discovery not performed for all homes if an invalid home exists.
DLPX-72351	When a user tries to change credentials for a dSource, validating the credentials before updating them. In case of invalid credentials, showing user an error message about it.
DLPX-72545	Initial ORA-65294 error not reported to user when vPDB provision fails due to compatible parameter mismatch.
DLPX-72652	Fix and issue that prevents use of the NFSv4 on some versions of SUSE Linux targets.
DLPX-72698	Patching Oracle 19C vCDB leads to ORA-25153 as described in 2285159.1.
DLPX-72807	Fixed issue with SQL Server 2014 dSources with filestreams where sync failed in merging filestream directories due to long path names.
DLPX-72882	Datasets hooks script editor properly displays multiline scripts instead of as one long line on non-Chrome browsers.
DLPX-72916	Empty string in SNMPv3 USM username creation no longer throws fatal error.

Bug number	Description
DLPX-73048	Non-sys user credentials for Oracle sources cannot use password vault.
DLPX-73108	Fix a bug that prevents the API from displaying the correct number of CPUs or amount of memory assigned to a Delphix Engine after a hot-add operation.
DLPX-73201	Fix an issue that prevents the configuration of additional NICs on Azure Delphix Engines.
DLPX-73202	Fix a bug that can cause a VDB to fail to mount while other VDBs are being stopped.
DLPX-73424	Fix a bug that prevents the sysadmin from deleting a default route.
DLPX-73527	SnapSync job fails with 'internal error during execution' due to ONS/FanManager errors.
DLPX-73528	Fixed a bug that prevented accessing SDD specs from CLI.
DLPX-73611	Kerberos ticket expiration date parsing is incorrect after migration from Illumos to Linux.
DLPX-73742	Provisioning an Oracle TDE-enabled vPDB fails with the error "ORA-28367: wallet does not exist" if the TDE wallet for the target linked CDB is stored on ASM storage.
DLPX-73765	Fix a file descriptor leak that causes the management service to crash over time.
DLPX-73789	Auxiliary CDB instance uses dSource keystore location if WALLET_ROOT is configured in dSource.
DLPX-74030	CDB database password may be leaked as part of environment monitor checks that launch sqlplus command on the source or target host.

Bug number	Description
DLPX-74043	Delphix OS user cannot provision TDE-enabled vPDB due to directory permissions in the default wallet location.
DLPX-74044	Delphix OS user cannot provision TDE-enabled vPDB in Delphix-writable keystore location due to directory permissions.
DLPX-74119	Drop database fails if default database is set to any other than master.
DLPX-74164	Sync fails with db.aseldb.source.dump_history.incomplete_stripes after dump history file is purged and Use dump history is enabled for the dSource.
DLPX-74233	During failover of a namespace, if there is a collision between an environment in the namespace with one on the target engine, the namespace environment will get renamed if its host does not match that of the environment on the target.

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-73390	Improve replication receive throughput.
DLPX-73393	Improve write performance under extreme disk fragmentation.
DLPX-73280	Improve write performance under extreme disk fragmentation.

Release 6.0.6.1 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-73848	Fixed an issue that can cause the management service to fail to start after upgrade on systems that have had SNMP enabled.
DLPX-73859	Fixed a file descriptor leak triggered by faults and alerts that can cause the management service to fail.

Release 6.0.6.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-47065	VDB recovery failed when files other than archive logs were detected by Oracle.
DLPX-47493	Fixed the bug where VDB directory under the DelphixConnector directory was not being removed from the target host on MSSQL VDB deletion.
DLPX-48046	Added sorting parameter to network test APIs.
DLPX-61405	Replication may send more data than expected if masking involves dropping large DBF files.
DLPX-61525	The height of the storage configuration list was limited to show 3 disks at a time. It will now dynamically grow with the number of disks.
DLPX-63603	Increased connector timeout from 10 minutes to 30 minutes to avoid unnecessary faults due to timeout during Validated Sync operation.
DLPX-67368	Delphix Engine hostname change is now immediately reflected in Splunk events.

Bug number	Description
DLPX-67593	Fixed an issue that caused the management service to remain offline following an out-of-space condition.
DLPX-68531	Introduced better handling of UniversalConnectionPoolException errors during SnapSync.
DLPX-69759	Oracle environment discovery failed due to an unhandled exception occurring at insert into dlpx_faults.
DLPX-69852	Fixed a bug that caused network configuration problems when removing and adding additional NICs.
DLPX-70426	Redaction of usernames took forever on tables with millions of entries.
DLPX-70583	move-to-asm.sh fails if timing is set in glogin.sql.
DLPX-70638	Removed Failed Actions section of Actions sidebar, in favor of manually dismissing from Running Actions and falling to Finished Actions section.
DLPX-70653	Removal of all instances in a RAC VDB should not be allowed.
DLPX-70808	Fixed issue related to the creation of empty DisableBroker.sql on the Windows machine in case DisableBroker.sql execution fails in the first attempt.
DLPX-70896	Added more detailed error message for when the Delphix Engine fails to push a script to Windows host.
DLPX-70919	Fixed an issue that causes job progress to not update in Self-Service.
DLPX-70928	Fixed a bug that results in a Delphix Engine remaining powered on following a shutdown from the user interface.

Bug number	Description
DLPX-71093	For AG databases, a full backup is not required even recovery fork guid changed but the LSN chain didn't break because of transactional log backups.
DLPX-71097	Unable to ignore snl.bct.needed warnings if Block Change Tracking is legitimately disabled on an Oracle dSource.
DLPX-71153	Recovery of PDB should fail if the database is down after offlining datafiles.
DLPX-71370	While deleting initiator in Windows environment deletion operation, delete all the views as well for that initiator.
DLPX-71685	VDB is auto disabled if the hook fails.
DLPX-71865	Reduced the size of support bundles.
DLPX-71961	When a PDB is selected for replication, its CDB and all other PDBs in the parent CDB get automatically selected for replication. Going forward, in the above scenario, while the CDB will get selected, its other PDBs will no longer get selected.
DLPX-72031	Fixed VDB refresh operations failures due to 'DB STARTUP' background process spid greater than 50.
DLPX-72066	Migrate VDB verifies against the old configuration, rather than new.
DLPX-72083	Fix an issue that causes a fully-qualified hostname to be changed on upgrade from 5.3 to 6.0.
DLPX-72131	Added namespace support for HashiCorp password vaults.
DLPX-72265	doCreateTempfiles.sh.template exits with code 0 on failure.
DLPX-72340	Incomplete recovery not detected during provisioning.

Bug number	Description
DLPX-72386	Unlock Solaris x86 Solaris -> Linux x86 provisioning.
DLPX-72452	For clusters with long hostnames, vPDB sync fails with exception.oracle.accessor.instances.missing.
DLPX-72495	Fixed a bug that prevents the application from coming up after an upgrade
DLPX-72686	Delphix no longer logs environment variables in logs on connected hosts since this could leak sensitive information such as passwords that are sometimes stored as environment variables on database hosts such as for the ASE database.
DLPX-72730	Fixed a Snapsync performance issue.
DLPX-72790	SnapSync job fails with 'internal error during execution' due to ORA-01652.
DLPX-72862	The scenario which was causing the null pointer has been fixed now.
DLPX-73300	Validation of connection to a container for PDBs should allow connecting to CDB$ROOT.
DLPX-73311	Added platform detection for ESX 7.0u1.
DLPX-73449	Replication of policies between two engines, in a loop, could lead to OOM exceptions.

Fixes that take effect after an optional reboot

Bug number	Description
DLPX-72990	Addressed a minor CVSS 5.9 security issue with no known attack vectors.
DLPX-73067	Fix for CVE-2020-10753.
DLPX-73069	Fix for CVE-2020-12059.

Bug number	Description
DLPX-73070	Fix for CVE-2020-1760.

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-71924	Fixed a bug that causes support bundle collection to fail with an internal error.
DLPX-72918	Fixed a system crash that can happen when replicating a masked VDB using SDD.
DLPX-73147	Fixed a bug that can cause a replication source to crash if it had run replication while running on 5.0.

Release 6.0.5.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-23360	The unistallation of the Delphix Connector installer should succeed even if one of the component connector services doesn't exist.
DLPX-69155	Reduced the time taken to generate support bundles in some cases.
DLPX-67316	Recreating a controlfile against an Oracle source may yield misleading error during snapshot.
DLPX-70766	JDBC driver updated to resolve intermittent JDBC connection failures due to JDBC SSL bug.
DLPX-70785	Options passed to VDB mounts on target AIX hosts did not include read and write size values. This fix adds the rsize and wsize parameters to mount command depending on the maximum values host is configured to support.

Bug number	Description
DLPX-70741	Enabling Validated Sync while SAP ASE SnapSync job is running leaves staging database unrecoverable.
DLPX-69865	Fixed a bug that causes a network interface to become unconfigured if its MAC address changes.
DLPX-71233	If LogSync is suspended when performing SnapSync of a standby database in real-time apply, SnapSync attempts to backup the archived logs which can cause SnapSync to become unresponsive.
DLPX-69800	UEM/Hostchecker directory ownership checks fail on HPUX environment with long usernames.
DLPX-69807	Provided mechanism for the user to bypass corrupted/ incomplete jdbc libraries.
DLPX-66585	Bundle ID "fault.environment.configuration.file.owner" reports insufficient host address.
DLPX-65739	createDelphixDBUser.sh fails when "@" used in the password.
DLPX-70973	SAP ASE database provisioning fails if the source database has holes in log fragments.
DLPX-71532	Improved error handling for Oracle memory configuration errors.
DLPX-62987	Allowed assigning privileges over replicated objects through the UI.
DLPX-71593	TIMEFLOW_REPAIR incorrectly skips a log because of "wrong database".
DLPX-71751	Added NFSv4 support on AIX for Oracle and SAP ASE.
DLPX-71736	Dynamically disable RPC services if NFSv3 is no longer in use.

Bug number	Description
DLPX-71772	Network DSP Test between versions 5.3 and (6.0.3, 6.0.4) is fixed.
DLPX-71513	Replicate non-data objects like delphix engine users, authorizations, roles, permissions, policies and DB config templates.
DLPX-71305	Unable to load dummy recovery database dump due to SAP ASE error 15728.
DLPX-71172	Enabling SAP ASE dump history causes IllegalStateException in getDumpListFromLastRestoreDateAndFiles due to timestamp mismatch because of TZ.
DLPX-71178	SAP ASE internal error raised when dump history file is purged using sp_dump_history.
DLPX-65101	Fixed a race condition between a DB_DELETE job and the Oracle retention policy worker for the same container that can lead to a deadlock between the job and the worker.
DLPX-71918	Fixed an issue that causes the Delphix Engine UUID to change upon rebooting in IBM Cloud.
DLPX-71141	Fixed an issue where upgrading an Oracle dSource or changing the environment user in a linked Oracle dSource fails with the error "SOURCE_UPGRADE job for xxx failed due to an internal error during execution."
DLPX-71611	Updated UI time zone library to IANA 2020a.
DLPX-70349	Fixed a memory leak during incremental replication.
DLPX-72038	Fixed an issue that prevents 5.3.x - 6.0.x upgrade if a static route exists that goes over a DHCP interface.
DLPX-72148	Fixed a bug of always order hooks alphabetically rather than the running order set by users.

Bug number	Description
DLPX-71971	Allowed Enable/Disable of VDB if its current Timeflow has at least one snapshot.
DLPX-72115	Changed the Time Point field on a VDB back to reflecting the point on the parent the VDB was created from, but displayed in the timezone of the parent.
DLPX-71995	6.0.4.0 can no longer interact with 5.3.x remote Masking Engines.

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-70675	Fixed a bug that causes the system to become unresponsive after expanding multiple storage devices.

Release 6.0.4.2 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-72155	Fixed an issue that can render a Delphix Engine unbootable if a reboot occurs after upgrade verification but before the upgrade is applied.
DLPX-71141	Fixed an issue where upgrading an Oracle dSource or changing the environment user in a linked Oracle dSource fails with the error "SOURCE_UPGRADE job for xxx failed due to an internal error during execution."

Release 6.0.4.1 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-71930	Fix a bug that causes feature flags to be disabled when upgrading to 6.0.4.0.

Release 6.0.4.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-68173	Resolved an issue where temporary database backup/ device files created for cleaning up the target database were not being deleted.
DLPX-68773	The management stack runs out of memory as the environment monitor does not purge stale objects.
DLPX-69573	Allow linking an Oracle PDB with a lowercase name.
DLPX-69634	Allow provisioning an Oracle PDB with a lowercase name.
DLPX-69962	After detaching a PDB, perform unplug/plug, and attach again, if disabled is performed before SnapSync, the PDB can no longer be enabled.
DLPX-66045	Prevent Self-Service Container branches getting into an unusable state by blocking deleting the last segment of branches.
DLPX-7037	Snapsync performs an unnecessary checkpoint.
DLPX-68277	Users will see the detailed error message upon connection failure to Delphix connector during OS user validation and there will also be a "More" button with an error message which will open an error popup with all error details.

Bug number	Description
DLPX-70288	On the "Add Environment" screen when OS user validation will get fail, they will see the "More" button along with the error message. When the user clicks the button, an error popup opens with all details of the error and suggested action.
DLPX-70832	NFSv4 support for appdata sources running on AIX.
DLPX-68495	Fixed GUI reporting conflict information when creating a Retention Policy.
DLPX-70788	Added Environment User field for MSSQL sources in Datasets -> Configuration -> Source tab -> Staging Environment section.
DLPX-58047	Fixed bug where the sort sequence was incorrect. Fixed in Hook Operation Templates.
DLPX-67931	Provision against VPDB after create/drop a new tablespace failed with exception.oracle.targetscripts.rename.datafiles.
DLPX-59910	Comps.xml associated with Oracle Homes are marked as unparseable if they are longer than 65535 characters.
DLPX-55476	CLI provisioning fails when the mount point provided includes quotes around the path.
DLPX-71168	Changed type to text and spaced "Secret Key" and "Username Key".
ORB-3285	Support using `api.delphix.com` as a proxy for verifying the Cloud Agent binary's code signature certificate.
DLPX-71006	Allow provisioning across patch versions for Oracle versions on or after 18.X.
DLPX-71334	Migrate NTP configuration when upgrading between 5.3 and 6.0.

Bug number	Description
ORB-3286	Communication with Central management servers is now routed through the web proxy when one is configured for the Engine.
ORB-3117	Summary: Increase an action's failure message size to 256 characters so users can view large failure messages.

Release 6.0.3.1 changes

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-71339	Fixed an issue that can cause the Virtualization Management service to become inaccessible when the system memory became highly fragmented.

Release 6.0.3.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-63192	More details will be displayed in the error message in case DB_SYNC fail due to missing SELECT permission on database 'msdb'.
DLPX-67708	Removed unnecessary Source Continuity source-archive file system.
DLPX-66878	A meaningful error message will be displayed in case the user is missing 'VIEW ANY DEFINITION' and 'VIEW SERVER STATE' permissions on AG Instance and linking is performed.
DLPX-68668	Fixed the issue when Environment discovery fails with Internal Error when Oracle DB instance name is > 15 characters.

DLPX-68539	Added support for Read-Only Oracle homes.
DLPX-62027	Fixed a bug that causes factory reset to fail when there are provisioned VDBs.
DLPX-67830	Eliminated a virtualization management service crash caused by egregious use of memory by environment monitoring.
DLPX-69067	Enabled NFSv4 support for older RedHat NFS clients.
DLPX-68491	Fixed an issue when SCAN Listener is not discovered for Oracle 19c Cluster Environment.
DLPX-68931	Improved replication throughput by parallelizing data streams.
DLPX-69350	Fixed an issue where in some cases a VDB's Time Point would not appear.
DLPX-62602	Prevented a full snapsync after detecting an incarnation change of and reverting to a previous incarnation of an Oracle database.
DLPX-69579	Resolved the issue of intermittent failure of DB_SYNC for source database full backups containing in-memory tables which were caused due to improper merging of filestream folders.
DLPX-69104	Fixed an issue when Environment Monitor task monitors replicated entities could lead to Out of Memory.
DLPX-58561	Increased online Redo Log size when using VDB Provisioning defaults from 50mb to 1024mb.
DLPX-68831	Storage is removed even when the drop database fails, causing ASE error 823.
DLPX-68323	Linking will not fail in case a slash is used as the path delimiter on the source database.
DLPX-69561	Allowed NoLogging Diagnosis to be shown and edited for Oracle CDBs.

DLPX-69625	Fixed an issue that causes the CLI to hang when deleting an object.
DLPX-65357	Source Environment selection in Attach dSource dialog is now alphabetically sorted.
DLPX-65215	Fixed an issue where Hotfixes aren't listed until after management service restarts.
DLPX-57988	The increased timeout of doShutdownOracleInstance.sh script from 20 seconds to 10 minutes.
DLPX-70018	Resolved the issue where during validated sync, fault "fault.mssql.source.next.backup.missing" was caused due to backupsets with similar first and last lsn.
DLPX-66671	dSource selection in dSource Linking Wizard is now alphabetically sorted.
DLPX-65723	MSSQL server cluster address is now editable through the Environments GUI.
DLPX-69514	Gracefully handle accelerated networking on Azure.
DLPX-68942	Implemented retries with some time delay in case of a failure while switching database user mode.
DLPX-56626	Some orcl_log_info entries have a very large and incorrect end_scn (281474976710655).
DLPX-67579	Deleted users' actions should be included in the action/audit log API results.
DLPX-69863	Enhanced instruction text relating to SSH when editing environment users.
DLPX-70081	Removed excessive debug logging for DSP connections which results in fast rollover of debug logs.
DLPX-66203	CLI / API calls to refresh/rewind vCDB directly should be disallowed.

DLPX-67194	RHEL 7.6 connector log shows Unidentifiable version string: RedHatEnterpriseServer 7.6.
DLPX-66754	When VDB is disabled, environment configuration can now be edited in the UI.
DLPX-62095	A wrong certificate is identified as an issuer of a self-signed certificate in rare cases.
DLPX-69243	Do not require an issuer to be present or keep the full chain intact on Truststore operations.
DLPX-59331	Permit non-CA certs in user Truststore.
DLPX-60779	Changed error message when there are no compatible installations on provisioning.
DLPX-69518	Provisioning failures due to BitLocker encryption will be identified and a proper error message will be displayed.
DLPX-64797	Fixing memory leak in hk2 library.
DLPX-70039	Password vault migration nullifies ASE linked source dump credentials.
DLPX-70089	Protection against a variant of billion laughs attack (XML entity expansion).
DLPX-64207	Added API support to revert from static to DHCP DNS settings.
DLPX-68857	Faults reported for Oracle Home missing where the Central Inventory does not show this Oracle Home present.
DLPX-64435	Exclusively specifying 'required' parameters to discover Oracle cluster via CLI results in an exception.
DLPX-69604	Alerts & Faults are reported for hosts in a namespace that can cause Out of Memory issue.
DLPX-39882	Prevented cloning of Tiimeflow storage for Oracle source continuity.

DLPX-67425	Resolved an issue when validated sync (with full/diff) restored multiple backupsets and a restore failed with a SQL server transient issue after a source continuity reset event resulting in a state where no operations could be performed on the dSource.
DLPX-70433	"DLPX_EXECUTE_SQL_CLEANUP_RETRY" will also print nested SQL error messages in case of command failure.
DLPX-68582	Customers now have access to an API to display the Engine License information.
DLPX-61335	Displayed in confirmation dialog the name of the user being deleted.
DLPX-70639	Resolved output buffer issue while identifying BitLocker encryption during provisioning on Win19.
DLPX-66259	Updated messages on the upgrade page when the operation fails.
DLPX-70782	Bumped up connector version for NET 4x installer as shipped OpenJDK version had been upgraded.

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	**Description**
DLPX-68995	Improved performance of dataset deletion.
DLPX-68997 DLPX-68999	Improved single connection replication throughput.
DLPX-70697 DLPX-70703	Addressed an issue that causes long periods of I/O unresponsiveness.
DLPX-69953	Fixed a bug that can cause a Windows iSCSI initiator to fail to connect to the Delphix Engine.
DLPX-70512	Fix a hang in the I/O subsystem that can cause the Delphix Engine to become unresponsive.

Release 6.0.2.1 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-70065	Provisioning a VDB from a dSource or another VDB will fail if the following conditions are met: • Delphix Engine has at least one dSource and a VDB created using a Python plugin prior to the upgrade • Delphix Engine was upgraded to 6.0.2 • Provisioning was attempted from the UI after the completion of the upgrade
DLPX-69350	Fixed an issue that the time point attribute of a VDB is not shown.

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-69864	Fixed an issue that causes MSSQL operations to hang after the reception of an iSCSI LUN reset.

Release 6.0.2.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-62806	Fixed an issue where provision against PDB after unplug/replug against the same linked PDB fails with exception.oracle.targetscripts.controlfile.create.
DLPX-67567	Oracle Source Continuity creates an unnecessary source-archive file system on zfs.
DLPX-27807	LogSync may fall behind when connected to an Oracle physical standby database in Real-Time Apply mode.
DLPX-68385	Customer provided Oracle Java missing in the search path for Java on hosts.

Bug number	Description
DLPX-62782	Reducing the number of nodes for RAC VDB and VDB in NOARCHIVELOG mode may result in ORA-00258 errors during VDB enable operation.
DLPX-62738	Better error message when plugins are uploaded out of sequence.
DLPX-68722	The product now recognizes VMware with BIOS date of 12/12/2018 as VMware 6.7.0u2.
DLPX-68579	SnapSync of Oracle 19c DB with encrypted tablespace fails with fatal exception "Block header 91 is not empty".
DLPX-68689	Fixed the issue where a huge number of error messages from ASE caused OutOfMemory Error.
DLPX-68957	Always On AG discovery will not fail in a multi-subnet environment.
DLPX-63088	Can now recover multiple Self-Service containers at the same time.
DLPX-47977	Improved handling of snapshot standby.
DLPX-64125	SnapSync failed with exception.oracle.dsource.sync.no_hosts.rac on RAC clusters with very long hostnames.
DLPX-62584	PDB enable failed after migration if mountBase has a trailing slash.
DLPX-68657	Virtualization can now fetch jobs from Masking engines configured with HTTP redirection.
DLPX-69121	It is no longer mandatory to have at least one enabled system administrator with local credentials.
DLPX-68167	Fixed an issue where too many requests were being sent for Faults from the Datasets pages.

Bug number	Description
DLPX-69082	Large stderr produced by failed rsync jobs are truncated to prevent Java OutOfMemory errors.
DLPX-58600	Datasets filter updated so that all items within a group that matches the filter string are displayed, even if the items contained in the group do not match the filter string.
DLPX-65896	VDB deletion failed due to the inability to delete LogSync worker.
DLPX-57903	Improved diagnosability of PDB discovery issues.
DLPX-68878	Fixed issue where start/stop buttons were not being displayed in the RAC instances configuration table.
DLPX-69271	Enabled replication smart failover by default.
DLPX-66715	The user-visible name for Oracle cluster objects is being replaced with the Oracle cluster name. For Windows clusters, the user-visible name is being replaced with the cluster address.
DLPX-68929	Changed default replication settings for better out of the box performance.
DLPX-68930	Improved replication throughput when sending multiple timeflows.
DLPX-69245	Fixed a memory leak that occurs when experiencing connectivity errors.
DLPX-69377	At least one non LDAP system user should be enabled when the LDAP server is being disabled.
DLPX-68575	LDAP principal fields were not being redacted in phone-home bundles.
DLPX-68528	Self Service Recover operation failed due to missing Timeflow.

Fixes that take effect after an optional reboot

Bug number	Description
DLPX-66808	Re-introduced console splash screen with IP address and service states.

Release 6.0.1.1 changes

Fixes that take effect after an optional reboot (Optional on Reboot)

Bug number	Description
DLPX-69203	Improved synchronous write performance over iSCSI.
DLPX-69167	Improved SQL Server data ingestion performance by leveraging asynchronous writes on underlying storage.
DLPX-69298	Eliminated possible data corruption on SQL server and vFiles over iSCSI that can occur when a Delphix Engine reboots.

Release 6.0.1.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-60689	For SAP ASE, instead of using the DBCC CHECKALLOC command to fix DBID mismatch issue, the MOUNT command with FIXDBID and ALLOW_DBID_MISMATCH clauses will be used, to improve performance.
DLPX-65831	VDB snapshots need to clean unneeded ZFS datafiles.
DLPX-63949	Improved boot time after 5.3 to 6.0 migration by optimizing metadata indexing.
DLPX-66261	Upgrades to 6.0.0.0 will only be supported from a release greater than or equal to 5.3.6.0.

Bug number	Description
DLPX-66486	Snapshot of a linked database can end up with extra datafiles that do not belong to the database which might cause VDB on VDB provision to fail during rename of datafiles.
DLPX-66558	Cluster environment discovery was incomplete if the host locale was not English.
DLPX-66804	DB_LINK using incorrect user when RAC node also configured as a standalone environment.
DLPX-66768	vPDB save state lead to rollback or child provisioning failures.
DLPX-66823	Unable to link database with CL8MSWIN1251 charset.
DLPX-64538	Fixed a bug causing the timezone selector to only be visible when manually setting the time.
DLPX-66809	Removed the Windows Diagnostics Files and Directories on successful Diagnostics upload.
DLPX-67279	Provision failed when the source was from a RAC Oracle Standard Edition database and the target was Oracle Standard Edition.
DLPX-67451	Fixed an issue that sporadically caused replication to fail with an internal error.
DLPX-67454	Delphix Engine should select the highest version ojdbc driver available at ORACLE_HOME/ojdbc/lib.
DLPX-66077	Ensures child worker threads are gracefully exited when parent linked source sync job has completed/ terminated.
DLPX-45983	MSSQL Validated sync will resume when storage usage falls below the threshold if storage threshold enforcement failed in the past.

Bug number	Description
DLPX-67560	Fixed an issue where MT provision may result in ORA-02058 due to un-purged or inflight 2PC transactions on dSource.
DLPX-67594	Old timeflows and snapshots are not getting removed by snapshot retention.
LX-2020	Report the correct amount of memory allocated to EC2 Nitro instances.
DLPX-67413	Fixed an issue where VDB point in time provisioning might fail if Oracle database environment is configured in a non-English locale.
DLPX-67684	PDB provisioning failed if the source had shutdown triggers.
DLPX-67575	Fixed failure during point in time 'Virtual to Physical' provisioning.
DLPX-67668	After setting the database online give it some extra time to startup completely, before doing any further operation on it.
DLPX-67759	Redact sensitive information from phone-home data.
DLPX-64638	Validated sync stops working if Delphix cannot connect to the backup server.
DLPX-65559	Even when the staging instance is down, attempt counter to detect backup files keeps on increasing and eventually, it stops detecting backups.
DLPX-56537	When a target host is used by a large number of dSources for staging or has a large number of objects, the performance of Delphix operations like validated sync, refresh, rewind, etc can be slow due to Powershell processes being serialized.

Bug number	Description
DLPX-67894	Removing cluster resource without removing its dependency can result in cluster failure. So, added retryer logic while fetching the resource dependencies (Get-ClusterResourceDependency) and ultimately fail the operation after all the retries.
DLPX-67813	Unsupported SQL server backup type gets picked while validated sync and the operation fails while looking for the backup. So, introduced a tunable filter to automatically skip SQL backups taken by backup software not supported by yet Delphix.
DLPX-67925	Added env host connectivity toolkit support for SLES on Power9.
DLPX-67934	Retries to fetch image identifiers during Netbackup restore if there is a mismatch between MSDB and Netbackup Master.
DLPX-67655	Fixed an issue where retention enforcement can generate user-visible errors while attempting to delete snapshots with dependencies after PDB migration to new CDB.
LX-1944	EBS NVMe devices can now be used in Delphix Engines.
DLPX-68022	Fixed an issue where hostchecker 'Check Oracle DB Instance' fails on HPUX and AIX.
DLPX-68124	PDBs with lower/mixed case names will not enable after an upgrade.
DLPX-68126	Fixed a bug that limits the number of disks that can be added in GCP.
DLPX-67421	Update the primary db file names in a transaction with the Timeflow creation to make sure whenever a Timeflow is created successfully we have its primary file information.

Bug number	Description
DLPX-67440	Skip VDBs having its current Timeflow as null from 'PrimaryDbFileAvailabilityCheck' as these VDBs doesn't undergo queisceing and are recoverable by refreshing them.
DLPX-61818	Linking wizard - Target Environment step - Privileged Credentials authenticates on the selected target now.
DLPX-68117	Some non-Admin users, lack all permissions, are unable to login to upgraded engine.
DLPX-67290	A wrong version input by user while manually adding a SQL Server instance, created issues in provisioning VDBs. SQL Server version will now be auto-discovered for manually added instances on adding or refreshing the environment.
DLPX-66238	Updated error message to let know user that non discovered CDBs are filtered out from the list when linking a detached source.
DLPX-68457	When a target host is used by a large number of dSources for staging or has a large number of objects, the performance of Delphix operations like validated sync, refresh, rewind, etc can be slow due to Powershell processes being serialized.
DLPX-68484	Fixed the issue where 'lstart' column value of sysusages table was beyond the range of Integer data type by taking the Long data type to store the lstart value.
DLPX-68500	Fixed an issue where the NTP service is not started following a reboot.
DLPX-68290	Support bundle generation can be time-consuming if the engine has a large number of snapshots to process.
DLPX-67792	Fixed issue in grids in which the selection checkbox was unclickable.

Bug number	Description
DLPX-67555	Provision vPDB/vCDB fails with ORA-45900 if the parameter enable_pluggable_database is omitted when specifying database parameters for new vCDB.

Fixes that take effect after an optional reboot

Bug number	Description
DLPX-67782	Engines running 5.3 on EC2 i3 can now be migrated to 6.0.
DLPX-67961	Fixed an issue that prevents ssh access after switching to a static IP address.
DLPX-65948	Fixed a bug that could cause replication jobs to fail with internal errors
DLPX-68025	Improved boot time after 5.3 to 6.0 upgrade by reducing the overhead of setting ZFS properties.
DLPX-67868	Fixed a bug that can cause the management service to run out of memory when disabling the Splunk integration.

Release 6.0.0.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-27433	The analytics GUI network graph shows newly added NIC information without requiring a management service restart.
DLPX-33998	If you add a hook script via the CLI, newlines are removed erroneously.
DLPX-40094	Correctly set the default type for the parameters to all operations in the CLI according to the container type.

Bug number	Description
DLPX-43215	Exclude sybsecurity from the list of auto-discovered databases.
DLPX-48712	Java 6 packages are no longer included in the product image.
DLPX-48280	When a user is set with the Provisioner role the 'provision' button does not appear, meaning anyone set with this role only is unable to provision VDBs.
DLPX-53996	The Delphix Engine does not provide instructions to browsers to avoid caching HTTP responses (pages).
DLPX-54740	Ensure Windows mount points are always unmounted as part VDB refreshes to prevent future VDB refreshes from failing due to "ERROR_ASSIGN_MOUNTPATH: failed to assign mount path for disk at="">>, error="">,>
DLPX-55282	In environments where the vPDB has been provisioned using a Delphix provisioned virtual CDB, shutting down the virtual PDB causes it to get into an incorrect "Cannot monitor" state, this has now been fixed to show the correct "Stopped" state.
DLPX-55598	Fixed an issue where vPDB refresh/rollback triggers spurious vCDB restart jobs, after vPDB+vCDB auto-restart.
DLPX-55829	Validated Sync can fail when monitoring ASE backup servers started by using the $DSLISTEN environment variable instead of the "-S" argument. This can be worked around by accessing $DSLISTEN in the RUN_xxxxx script and pass it down as -S.
DLPX-55958	VDBs with no snapshots failed to re-enable after a Delphix Engine upgrade, this has now been fixed.
DLPX-57454	Display underlying ssh error when environment host connections fail.

Bug number	Description
DLPX-58519	Enable Oracle LiveSource when LiveSource is in RESYNC_NEEDED state currently re-start Oracle Redo Apply. Oracle Redo Apply should not be restarted in this state.
DLPX-58760	Fixed a TCP port leak in the network throughput test feature.
DLPX-58845	Provisioning vFiles to the same host using different OS Environment Users no longer fails.
DLPX-59772	The API to list all snapshots consumes a significant amount of memory when there are more than 100,000 snapshots on the engine.
DLPX-60356	Fixed an issue where Oracle remote listener registration fails if set to empty string.
DLPX-60603	Network settings dialog now displays actual MTU value rather than a checkbox.
DLPX-60907	Fixed an issue where the Environment Monitor on Redhat 6.9 and 6.10 might throw unidentified version errors.
DLPX-60979	When user configures connection strings manually, these connect strings can end up connecting to incorrect PDBs/CDBs causing invalid snapshots. Verify that each connection to a PDB/CDB connects to the expected PDB/CDB.
DLPX-60993	Delphix backups create controlfile records; in rare circumstances, these records can cause invalid snapshots. To avoid this problem, remove Delphix backups control file records when using SCN-based SnapSyncs once a SnapSync completes successfully.
DLPX-62094	Allow certificates to expire after issuer certificate expiration.
DLPX-62241	Reduce SSH connections by temporarily preserving and reusing existing Delphix<->host connections where possible.

Bug number	Description
DLPX-62781	Spurious job event "DISCOVERED_TO_MANUAL_ORACLE_CLUSTER_NODES" no longer shows up for non-Oracle RAC environment refreshes.
DLPX-62892	In Oracle versions 18c and 19c, an Oracle bug can prevent the datafile headers from being updated for a standby database when managed recovery is running, resulting in failed SnapSync operations. Alert the user that an Oracle patch might be needed.
DLPX-62962	Removed unneeded EMPTY_RENEGOTIATION cipher
DLPX-62998	Fixed an issue where stale file mounts may be leftover when vPDB provision fails.
DLPX-63469	Initial setup now fails if the system was not provisioned with enough storage.
DLPX-63600	Network settings dialog now displays actual MTU value rather than a checkbox.
DLPX-64641	Fixed an issue where the last snapshot of a vPDB Timeflow can be deleted after the vPDB has been disabled, thus leaving the vPDB in a state with no provisionable snapshots.
DLPX-64711	Allow provisioning to complete when source CDB includes PDBs in a broken state.
DLPX-66020	Provision should remove files present in datafile filesystem that are not part of the database when provisioning a VDB from a VDB.
DLPX-67299	ASE environment discovery will not fail if there is a mismatch of "dataserver name argument" and value of "@@servername".
Bug number	**Description**
DLPX-67753	Fixed an issue causing redirect responses to reveal server type and version when HTTP redirection is enabled.

Bug number	Description
DLPX-72068	Improved the way volumes are fetched while working on mounts.
DLPX-74396	Fixed an issue that occurred when manually adding a database to an environment which has the same unique name as a database in another environment managed by the same Delphix engine. Previously, Delphix reported an incorrect environment containing the same unique name.
DLPX-80172	Updated the Self-Service refresh warning message.
DLPX-80271	changeArchivelogMode now has an associated job event.
DLPX-80387	Fixed an issue where Oracle move-to-asm script would fail while dropping temp due to tempfiles being in use and unable to be dropped after the database was started.
DLPX-81184	For S3 object store, the "Base URL" input is renamed to "endpoint". The "region" input is now a dropdown for the user to select from a list of standard regions. The endpoint corresponding to that region will now be auto-populated.
DLPX-81692	Fixed an issue where Direct NFS was not being detected for Oracle 21.
DLPX-81996	Fixed an issue where an environment refresh after upgrade did not remove outdated/obsoleted toolkit components in 6.0.014.0 and 6.0.15.0.
DLPX-82075	Fixed an issue that prevented the creation of a network route whose gateway is reachable through multiple interfaces.
DLPX-82112	Added checks to prevent using NFSv4 with Direct NFS for some Oracle 19 versions that don't support v4 due to an Oracle bug.

Bug number	Description
DLPX-82236	Fixed an issue where Speculative Logging was being called out of context and could lead to unbounded consumption of rpool.
DLPX-82308	Fixed an issue where the provision/refresh of an Oracle Key Vault vPDB fails with, "ORA-28365: wallet is not open".
DLPX-82329	Action-based alerts now include the success or failure state of the action, including the reason in case of failure.
DLPX-82334	Enabled the creation of on-link network routes; routes whose destination are directly reachable without a gateway.
DLPX-82381	Improved Replication performance on engines with a lot of objects.
DLPX-82392	Fixed an issue where after editing credential environment variables, the create/provision VDB wizard would fail because the environment variables were missing the password field in the payload. The required password field has now been added.

Release 6.0.15.0 changes

Fixes that take effect after an optional reboot

Bug number	Description
DLPX-80760	Increased the inotify limit to address a defect during upgrade.

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-68240	Fixed an issue where LogSync for an Oracle RAC standby does not set max number of backup tasks correctly.

Bug number	Description
DLPX-73375	Added granularity in restore job events to mark various phases of Restore Backup process. Added events in Restore Backup job to provide information at the start of the Redo phase of a restore command.
DLPX-75677	Fixed an issue causing auto population of the encryption key when no input is given.
DLPX-78913	Fixed a minor typing issue when deleting a MSSQL database.
DLPX-79228	Fixed a VDB Start Fault after a VDB start job is successful.
DLPX-79528	Made improvements in environment monitoring to return the NO_DELPHIX_DATABASE status for Staging Push dSources when they are not managed by Delphix, and avoids monitoring other attributes for such databases.
DLPX-79596	Made a change in the enable flow for Staging Push dSources to always make an attempt at unmounting the DATA storage before dropping the staging database.
DLPX-79780	Mount with local_lock=all on Linux, when mounting VDBs with NFSv3.
DLPX-79999	Use the offset in the time_zone column of msdb.dbo.backupset to convert the timestamp correctly for a backup from a source, and then continue using the staging host timezone to display the time of a snapshot on the UI.
DLPX-80206	Updated MD5 checksums for Oracle 12.1.0.2.0 OJDBC jars.
DLPX-80254	Fixed an issue where Oracle JDBC jar checksum checks are being reported as false positives despite underlying database connection issues.

Bug number	Description
DLPX-80406	The Spring framework has been updated due to the Spring4Shell vulnerability.
DLPX-80487	Fixed an issue where the Delphix UI would sometimes not render, showing waiting for response.
DLPX-80494	Added an action item to check for support Host and Server Type combination while adding MSSQL environments.
DLPX-80619	This change will introduce the ability to add sporadic failures via tunable: ADDITIONAL_SPORADIC_FAILURES.
DLPX-80909	Cross-Site Scripting (Reflected) in /resources/json/delphix/session.
DLPX-81048	Removed the requirement for Linux kernel recover-lost-locks setting when using NFSv4 with Oracle dNFS.
DLPX-81090	Can now enable/disable SNMPv3 vs. v1/v2.
DLPX-81100	Updated the command for checking database files accessibility while fetching the VDBs status with a lighter and less intrusive command, to get relief from VDBs being stopped randomly.
DLPX-81242	Fixed an issue preventing the upgrade of a replication Continuous Vault source with automatic replication.
DLPX-81308	Fixed an issue causing Appdata SnapSync to crash with NullPointerException when a virtual database is not successfully refreshed or rollbacked.
DLPX-81358	Fixed the unnecessary alerts of timezone discovery failure that users were facing randomly for the cluster environments.
DLPX-81502	Improved performance of the Replication page.
DLPX-81696	Users should now be able to enable the feature flag AZURE_DATA_BANK.

Bug number	Description
DLPX-81710	Fixed an issue where an engine could not be setup when objectStorage is enabled.

Release 6.0.14.0 changes

Security fixes

Bug number	Bug Introduced in	Description	Security Bulletin
DLPX-81059	5.2.2.0	Arbitrary Code Execution may be performed when configuring masking environments	TB098

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-38908	Oracle LogSync should automatically resolve faults for transient issues
DLPX-39193	Null Pointer Exception in Oracle LogSync backup stream handler
DLPX-57078	Job cancel requests during Oracle provisioning are not processed until the end of recovery
DLPX-57405	move-to-asm.sh does not support TDE-enabled databases
DLPX-62343	disable the OPTIONS method for all HTTP(S) requests
DLPX-64386	Oracle LogSync thread may hang when trying to remove temporary RMAN command file from source host toolkit directory
DLPX-65413	Ensure "source-archive" directory is unmounted at start of Oracle Provision

Bug number	Description
DLPX-66879	Oracle LogSync can create orphaned logs in certain scenarios
DLPX-69453	Provide tunable for Oracle LogSync client timeout
DLPX-69802	Common Toolkit directory is not removed from a mounted shared NFS location when the environment is deleted
DLPX-76382	Force disable should succeed despite environmental problems
DLPX-77840	Fixed an issue on the Setup pane to allow successful completion of an smtp test against a specific email address
DLPX-78412	SCM/Talaria failure reason should be communicated in the Delphix fault warnings
DLPX-78726	Removing windows environment performs cleanup of iSCSI persistent login target
DLPX-78754	Disable operation is prohibited on replicated sources
DLPX-78986	Prevent DSP connections for disabled Oracle RAC cluster nodes
DLPX-79077	Resolved an issue of an infinite spinner when validating BEQ credentials for Oracle dSource with duplicate unique_name. While linking a dSource, credentials are now required when discovering an unknown CDB.
DLPX-79242	Splunk HEC token logged to debug logs during splunkHec test
DLPX-79396	Allow users to unset Oracle database user name and credentials through CLI if Simplified Connection Management is enabled.
DLPX-79502	SnapSync fails if more than 1000 tempfiles exist in the whole CDB

Bug number	Description
DLPX-79591	Changed the NFSv4 minimum supported target Redhat version to 6.4 (was previously 6.3).
DLPX-79742	Unable to provision PostgreSQL VDB to Linux host with processor type of ppc64le
DLPX-79823	Improved the action item for failure to enable/attach the staging push dSource.
DLPX-79942	Improved the action item for failure to enable/attach the staging push dSource.
DLPX-80137	Limit SNMP configuration access to sysadmin
DLPX-80144	Oracle SnapSync crashes with NullPointerException when a PDB dSource is renamed and replaced with a new PDB of the same name
DLPX-80217	Fixed issue with filename conflicts during source backup restore
DLPX-80302	It was necessary to restart the auxiliary CDB during a TDE provision after recreating the autologin keystore
DLPX-80369	Oracle environment monitor triggers fault.oracle.db.connection.failed fault immediately on dataset stop or disable.
DLPX-80415	Fixes long delay in operations such as VDB start/stop when JDBC Thin connection to database fails due to unable to establish network connection.
DLPX-80439	Provide mount location to upgrade scripts during Lua to Python upgrade process if mount specification has not been provided.
DLPX-80440	Fixed spinner issue while provisioning a VDB from a SQL Server staging dSource from Provision VDB option in datasets menu

Bug number	Description
DLPX-80482	TDE-enabled provisions to a RAC target fail with "ORA-28365: wallet is not open" while attempting to reopen the database in start_database.sh in the auxiliary
DLPX-80483	Fix failing TDE-enabled vPDB provisions to a linked RAC container database due to "ORA-28365: wallet is not open" errors
DLPX-80487	Improved performance across dataset and replication related pages.

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-80078	The issue with removing files with complex file permissions on EBS is now fixed

Release 6.0.13.1 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-80818	libc upgrade necessitates PostgreSQL re-index.

Release 6.0.13.0 changes

Security Fixes

Bug number	Bug Introduced in	Description	Security Bulletin
DLPX-79789	5.3.0.0	Arbitrary Code Execution May Be Performed by Engine System Administrators.	TB096

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-7868	The issue with the confusing vPDB error on a snapshot after resetlogs of a linked CDB is now fixed.
DLPX-41671	You can now update the Oracle cluster home through the user interface.
DLPX-43467	When dSource is an Oracle standby in RTA mode, LogSync was raising the fault.oracle.linkedsource.log.conflict error and getting disabled on its own. This issue is now fixed.
DLPX-50309	Users can now change logSyncInterval for Oracle dsources.
DLPX-59757	The count for masking jobs fetched from the Masking Engine is now configurable. By default, it is set to 500.
DLPX-67604	The manual recovery of a database after V2P from a snapshot of dSource was failing with an error. This issue is now fixed.
DLPX-68684	The self-signed certificate is now compliant with the requirements for trusted certificates in MacOS 10.15.
DLPX-74613	Oracle VDB migration check needs to be done against the Oracle target host instead of the source. Furthermore, the Error and Action plan provided should include the target hostname.
DLPX-75467	The CLI now returns correct and descriptive error messages when executing unauthorized requests on uninitialized engines.
DLPX-75646	To diagnose BEQ connection failure, this release adds MD5 checksums for ojdbc*.jar for Oracle release versions up to currently supported release version.
DLPX-75878	The issue with the JDBC connection string for an Oracle vPDB not getting updated after an IP address change is now fixed.

Bug number	Description
DLPX-75989	The issue with the failure of environment discovery of an Oracle Cluster with a NullPointerException error is now fixed.
DLPX-76956	Previously, the Oracle JDBC test connection with the wrong password was increasing the `LCOUNT` value by more than 1. This issue is now fixed.
DLPX-77140	This release now speeds up metadata that is sent during replication when Extended retention is involved.
DLPX-77231	Previously, when source discontinuity on the dSource was followed by resync on the livesource, one or more livesource workers were failing to start. This prevented livesource status from getting updated and the first snapshot from being taken after resync. This issue is now fixed.
DLPX-77600	This release fixes NPE when Linking dSource with missing backup and unreachable nodes.
DLPX-77880	This release improves scalability for engines with an extremely large number of snapshots that were causing them to run out of memory.
DLPX-78015	Previously, V2P export with absolute data files was failing with an internal error. This issue is now fixed.
DLPX-78174	Insecure DES is no longer supported for SNMPv3.
DLPX-78420	This release adds V2P support for Windows server 2022 host machines.
DLPX-78473	This release improves load times for Datasets and Dataset Performance pages for engines with a large number of datasets and containers.
DLPX-78488	Previously, when switching from the backup server to ASE, dump history was not working for dSources configured to use the remote backup server. This issue is now fixed.

Bug number	Description
DLPX-78594	This release fixes an issue with disabled VDBs not being able to undo a refresh.
DLPX-78688	TLS 1.0 and TLS 1.1 ciphers are no longer available. Any system that is only configured with TLS 1.0 or TLS 1.1 ciphers is switched to use the default cipher set.
DLPX-78693	Invalid sync parameters will not cause DE server unavailability.
DLPX-78696	Switching sync strategy from source sync strategy type to sourceless strategy type is not allowed.
DLPX-79126	VDBs can now be automatically started with the tunables if stopped intermittently because Windows fail to write on the mount (Msg 9001).
DLPX-79292	This release eases restrictions on taking a snapshot of PDBs with encrypted UNDO tablespaces.
DLPX-79344	Previously, Snapsync of a standby PDB in mount mode was failing with the `ORA-01109: database not open` error message. This issue is now fixed.
DLPX-79422	Previously, clicking on a Replication Profile was resulting in the following error message `An error happened while communicating with the server`. This issue is now fixed.
DLPX-79789	Under certain conditions, arbitrary code execution may be performed by sysadmins.
DLPX-79808	This release fixes failures if the VDB name is more than 68 characters.

Release 6.0.12.1 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-79151	The issue with remote syslog configurations preventing engine upgrades or virtualization service restarts is now fixed.

Release 6.0.12.0 changes

Log4j Updates

Based on detailed testing and analysis, all the currently supported products are not susceptible to known log4j vulnerabilities. Please refer to TB095 Technical Bulletin for more information. All instances of log4j in currently supported Delphix products are updated to **log4j 2.17.1** as of this release.

Delphix keeps you updated on the latest developments and keeps releasing hotfixes, procedures, and workarounds for such critical vulnerabilities. For more information on how Delphix supports our product and customers in such cases, see Delphix Product Security

For more information, refer to the following pages:

- TB095 log4j vulnerabilities
- Uninstalling the delphix connector service from the target database servers
- Delphix product lifecycle policies
- Product security

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-23068	Validation for target database parameter 'DB_FILES' for single-tenant databases for the following operations: provisioning, refresh, rewind, and converting a dSource to LiveSource is now added. Furthermore, specific error messages for handling ORA-00059 errors are added.
DLPX-44544	The issue with the SnapSync of an Oracle standby dSource in Real-Time Apply mode calculating the snapshot's timestamp incorrectly is now fixed. This issue was resulting in ORA-01194 or ORA-01152 errors when provisioning to a timestamp after the snapshot.

Bug number	Description
DLPX-56691	The issue with the data files of a VDB getting unmounted when provisioning, refresh, or rollback job is canceled manually is now fixed.
DLPX-57971	The issue with the latest snapshot of a LiveSource taking a long time to show the SCN/timestamp range on its card in the GUI is now fixed.
DLPX-60320	UI now allows the selection of older dataset repositories (downgrade) in addition to selecting newer ones (upgrade).
DLPX-67069	The issue with the stopped Oracle VDB monitoring by the environment monitor that resulted in connection errors flooding the debug log is now fixed.
DLPX-68132	The issue with the “Copy query to clipboard” SQL copy functionality in the “Managed Source Data” is now fixed to have correct apostrophe characters.
DLPX-72123	The issue with the failure of detaching or deleting an Oracle dSource operation on RAC environments (This issue was occurring due to failure of deletion of RMAN backups on RAC and the operation needs to be retried with a force option) is now fixed.
DLPX-72779	The UI showing enabled or disabled for cluster nodes now uses a grid table. The Enabled column contains a checkmark to show whether the cluster is enabled or disabled. Users are able to select or unselect the checkmark to enable or disable a cluster.
DLPX-73975	Critical storage faults should not be ignorable nor manually resolvable
DLPX-74862	This release fixes an issue where the RESUME operation failed without any error thrown to the user on dSources where ENFORCE was still in progress. The fix will make sure that even if RESUME fails, it throws a DUE to the user with suggested actions to resolve the issue.

Bug number	Description
DLPX-75763	The issue with the failure of refreshing a VDB provisioned as an empty vfiles since there is no parent container to refresh from is now fixed.
DLPX-76266	The issue with the VDB Disable operation that resulted in an error message while connectivity with the host machine can't be established is now fixed.
DLPX-77123	You can now run Upgrade Verify when another upgrade is in progress.
DLPX-77347	This release fixes the difference in time shown for MSSQL snapshot based on different database authentication methods.
DLPX-77638	The issue with the failure of the End Entity Certificate expiration fault is now fixed.
DLPX-77664	The issue with the failure of the Oracle SnapSync with an error message, "RMAN-06183: datafile or datafile copy (file number) larger than MAXSETSIZE" if a datafile resized in the middle of SnapSync is now fixed.
DLPX-77913	The issue with the faults table missing data if there was more than one page of faults is now fixed.
DLPX-77925	The issue with the unsupported Windows release error message is now fixed.
DLPX-78113	MSSQL VDB database size will now be refreshed periodically based on environment_monitor.dynamic_attributes_check_period tunable.
DLPX-78183	The issue with the MSSQL validated sync schedule not getting updated without successful backup restoration is now fixed.
DLPX-78244	The issue with the failure of a few operations on self-service containers due to incorrect entries corresponding to the Oracle log metadata on the Delphix engine is now fixed.

Bug number	Description
DLPX-78258	The issue with the input bug that retained cleared out DB credentials in the dSource linking wizard is now fixed.
DLPX-78263	The issue with the failure of a SnapSync of an Oracle standby dSource in Real-Time Apply mode with an error message, "exception.oracle.snl.linkedsource.current_scn.invalid" if the rate of change in the database is low is now fixed.
DLPX-78265	Offline Oracle bystander PDBs data files can now be optimized leading to improved provision performance.
DLPX-78309	The issue with the CLI being unable to log in to system users when the main virtualization service is down is now fixed.
DLPX-78334	The issue with a large number of missing Oracle archive logs causing an error while viewing dataset is now fixed.
DLPX-78392	Hosts running Windows Server 2022 can now be added as Source and Target environments to the Delphix Engine.
DLPX-78522	SSLv3, TLS 1.0, and TLS 1.1 are no longer configurations options for HTTPS. Any system configured only with these removed options will be automatically set to use TLS 1.2.
DLPX-78791	This release upgrades log4j from 1.2.17 to the latest 2.x in Windows Connector.
DLPX-78938	This release upgrades log4j in virtualization to 2.17.1.

Release 6.0.11.0 changes

Security Fixes

Bug number	Bug Introduced in	Description	Security Bulletin
DLPX-77921	6.0.8.0	Arbitrary Code Execution by Delphix System Administrators may be Performed on Virtualization and Masking Engines	TB094

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-53019	The issue with the missing redo alert raised during the environment monitor check has now been resolved.
DLPX-59299	Discovery and monitoring rely on "Connected" in sqlplus output, which may not be the case if NLS_LANG is set to another language (e.g. Japanese). Downstream operations, like linking or provisioning, may then fail due to missing user privileges.
DLPX-59662	The issue with copy-Only Backups failure with Virtual Service Accounts has now been resolved.
DLPX-62706	The issue with the Hostchecker not properly checking /home/delphix permissions has now been resolved.
DLPX-64082	The issue with Oracle Provisioning scripts having hard-coded timeout issues has now been resolved.
DLPX-65729	Added retry functionality to the 'read backup files' operation during a validated sync to an account for an unstable environment.
DLPX-69778	SAML response is not logged on successful SSO login.

Bug number	Description
DLPX-72043	The issue where LiteSpeed `xp_restore_headeronly` stored procedure failure message are displayed when validated sync is active for dSources with LiteSpeed backup has now been resolved.
DLPX-72220	A UI issue that occurred while updating the vault when only the private key is changed has now been resolved.
DLPX-72225	Admin user created from management UI is no longer showing as 'non-admin' type.
DLPX-72237	The 'Verify Credentials' button from the DSP Throughput test page is now removed.
DLPX-72369	The dependency on a parent snapshot relying on the latest snapshot is now removed if a parent snapshot does not exist during the VDB enable operation.
DLPX-72778	Oracle dSource attach operation with changed DB ID using 'Force' option is now allowed.
DLPX-74555	Updated the "no Delphix connector" message while provisioning a Windows source environment.
DLPX-74676	Oracle LiveSource LogSync should only catalog valid archive log files.
DLPX-74851	In the Add Environment GUI, the mouseover information for "Set Delphix Session Protocol Options (DSP)" has been currented.
DLPX-74896	The race condition issue when running Oracle VDB refresh and dSource snap sync resulting in incorrect MDS entry for the parent snapshot of a VDB in the `dlpx_timeflow` table has now been resolved.
DLPX-75335	Added a product name and product version for the Delphix Connector executable so this information can be available before installation.

Bug number	Description
DLPX-75500	For ag cluster nodes, if the refresh fails due to timezone discovery failure, don't delete the nodes from MDS as it doesn't mean we had an issue with the nodes.
DLPX-75952	Database configs will be replicated only if the associated VDB is replicated.
DLPX-75995	The issue causing environment 'Add' or 'Refresh' to fail when PowerShell Transcription is enabled has now been resolved.
DLPX-76244	The issue where TCP fallback connection to database stops responding if the Oracle database instance is down has now been resolved.
DLPX-76290	Databases of UNKNOWN cdb type are now included in the attachment of a non-PDB container.
DLPX-76731	Added Delphix support for `WALLET_ROOT` and `TDE_CONFIGURATION` parameters to manage wallets in 19c instead of sqlnet.ora.
DLPX-76759	Added "Response" to faults along with other details when logged in the Admin App.
DLPX-76777	Remove orphaned Oracle logs resulting from archive log fetch timeouts.
DLPX-76793	Added execution timeout for execution of `UpdateFileACL.ps1`.
DLPX-76974	The issue where a user was unable to change 'from address' of SMTP server to noreply@delphix.com in the GUI has now been resolved.
DLPX-77112	The issue where an Oracle VDB cannot be provisioned between different minor versions if the Source is on higher RU has now been resolved.

Bug number	Description
DLPX-77284	The issue where after a hotfix was removed due to a successful upgrade, the system would still indicate the hotfix was installed post-upgrade has now been resolved.
DLPX-77345	The issue where provisioning a vVDB fails with `java.lang.OutOfMemoryError` when sqlplus is used to rename the datafiles has now been resolved.
DLPX-77405	Replicated password vaults will no longer be visible in the UI.
DLPX-77676	The issue where provisioning a vPDB from a PDB dSource fails with "ORA-65114: space usage in container is too high" if PDB `max_size/ max_pdb_storage` is configured has now been resolved.
DLPX-77708	The issue where a refresh/disable/destroy of a VDB using NFSv3 could cause loss of access to other VDBs that were using NFSv3 has now been resolved.
DLPX-77844	The issue where V2P operations from a VDB snapshot would result in the deletion of any production datafiles that exist on specified V2P target directory has now been resolved.
DLPX-77904	Removed 'Factory Reset' for Delphix Engines that are Data Vaults, as the operation is disabled for those engines.
DLPX-77912	The issue that can cause a VDB stop, refresh, or rollback to fail with an internal error has been resolved.
ORB-2465	Removed the requirement that SAML SSO email addresses must match case-sensitively for SSO logins.

Fixes that take effect after an optional reboot

Bug number	Description
DLPX-77577	Increased nvme I/O timeout to prevent storage issues in EC2 (Activated after optional Reboot).

Release 6.0.10.1 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-77467	Loading the setup app dashboard (as sysadmin) was rendering a server error popup with instruction to contact Delphix Support. This 6.0.10.0 error has been known to impair the ability to configure web proxy, PhoneHome, SMTP servers, and other connectivity settings via the GUI. It has now been resolved.

Release 6.0.10.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-18438	The issue with provisioning to the latest available time that resulted in generating the `exception.oracle.target.point.not_provisionable` exception has now been resolved.
DLPX-35480	Previously, static routes were being added using the add command. Now, the same can be added using the create command.
DLPX-57516	The issue with the failure of management service to start after configuring some abbreviated timezones from a picklist in server setup or sysadmin CLI has now been resolved.

Bug number	Description
DLPX-58133	Previously, the Oracle SnapSync operation was resulting in a warning for BCT usage on editions that do not provide it. This issue has now been resolved.
DLPX-58675	The issue with the deletion of the last snapshot on timeflow by Retention during a failed Oracle DB_SYNC operation has now been resolved.
DLPX-63003	The issue with memory being exhausted while reading too many snapshots from MDS has now been resolved.
DLPX-63347	If the staging source has the "Use as Staging" flag set as off, the user was seeing a specific exception while trying to enable a linked dSource to point in the direction of what needs to be done. Any compatibility failure will now have a specific exception.
DLPX-63601	Previously, querying the following operation "backupset table" and "whether a database is part of AG or not" was resulting in deadlocks and lock timeouts errors. We have now added retries to resolve the issue.
DLPX-64369	The issue with throwing `fault.oracle.linkedsource.incomplete.tempfile` for physical standby has now been resolved.
DLPX-65949	The issue with misleading status in the progress bar while taking a copy-only backup has now been resolved.
DLPX-69831	Previously, the Oracle dSource SnapSync operation was not displaying a clear failure message if a dSource `db_unique_name` is changed. This issue has now been resolved.
DLPX-70317	The issue with the restarting of the NFS-server by the reaper thread while deleting a vPDB from a linked CDB with Talaria turned on has now been resolved.

Bug number	Description
DLPX-71018	The issue with UI displaying only the suffix of the device name used by Hyper-V has now been resolved. UI now displays a unique device name for storage in Hyper-V.
DLPX-71292	The issue with the allowance of incremental SnapSync after LogSync throws `fault.oracle.linkedsource.log.conflict` has now been resolved.
DLPX-71639	The NFSv4 is now set as the default option when mounting datasets from OS platforms that support it.
DLPX-71769	The need to set permissions of `$ORACLE_HOME/dbs` subdirectories using STARTUP SPFILE syntax is now removed.
DLPX-72011	The issue with the CLI network setup not configuring the first network interface when multiple interfaces exist has now been resolved.
DLPX-72186	The issue with CDB log file retention working incorrectly if a PDB has multiple time flows pointing to the same CDB timeflow has now been resolved.
DLPX-72432	The format of `zpool_iostat_60.log` has been enhanced in this release. A timestamp is recorded for each sample in the log, making it easier to determine the time for each sample.
DLPX-72956	The issue with disabling the Oracle LogSync after running the validated sync job has now been resolved.
DLPX-73575	The timezone monitoring is now added for the Windows hosts.
DLPX-73590	You can now refresh a VDB whose parent dataset is in a different group without needing authorization on the parent or its group.

Bug number	Description
DLPX-73800	The issue with the failure of olsnodes when run as a non-Oracle user has now been resolved.
DLPX-74504	The issue with throwing a new DUE and NotFoundException when ojdbc libs cannot be read has now been resolved.
DLPX-74945	UI now displays a detailed error message for transaction log-chain break fault.
DLPX-74992	The issue with the failure of SnapSync operation when Database incarnation reset-logs end time is changed from "2021-03-13 22:03:07.0" to "2021-03-13 21:03:07.0" for virtual pluggable database “ " has="" now="" been="">">
DLPX-75389	The issue with the recording of the insufficient details by Logsync when dbid change was detected has now been resolved.
DLPX-75517	The issue with the failure of the Oracle vPDB provisioning with the "ORA-00959: tablespace 'TEMP' does not exist" error has now been resolved.
DLPX-75721	The issue with the failure of an Environment discovery with the "DelphixFatalException: Unknown Oracle Database status: REFRESHING" error has now been resolved.
DLPX-75737	The issue with saving unnecessary logs by Retention if bookmark falls exactly on a snapshot end SCN or snapshot end timestamp has now been resolved.
DLPX-75897	Previously, failure to start I/O services after the upgrade operation was resulting in a stack restart loop. This issue has now been resolved.
DLPX-75951	The internal error being signaled during VDB SnapSync by `removeUnneededZFSFiles` when a data file is physically removed during processing by an external cause has now been resolved.

Bug number	Description
DLPX-76288	The NFS latency for workloads involving a lot of parallel I/O (e.g. Oracle VDBs with concurrent accesses to many data files) is now improved.
DLPX-76388	Previously, entering key pairs directly into hook environment variables, as opposed to via a vault or as passwords were resulting in an internal error. This issue has now been resolved.
DLPX-76406	The issue with NFS-based VDBs becoming unresponsive has now been resolved.
DLPX-76447	The issue with the V2P Functionality to customize target directory structure for exporting database files to separate file systems not working as documented has now been resolved.
DLPX-76613	The issue with the unnecessary accumulation of heap when validated sync is active for dSources using an environment user that eventually can cause out of memory issues has now been resolved.
DLPX-76690	The issue with the removal of extraneous Oracle data files while creating snapshots has now been resolved. The extraneous Oracle data files are now removed during the provisioning operation.
DLPX-76692	For V2P operation, we now use the unbuffered copy method for better performance.
DLPX-76718	The issue with the creation of the extraneous Self-Service branch segments during the replication operation has now been resolved.
DLPX-76760	Environment clusters will now show faults from their child nodes.

Bug number	Description
DLPX-76802	Previously, the engines that are in the DEFERRED upgrade state were resulting in the "Large Receive Offload" option turned off which was leading to performance degradation in network transfers. This issue has now been resolved. Upgrading the engines that are in the DEFERRED state will also resolve the issue. Screen reader support is now enabled.
DLPX-76891	CRON expression labels now ask for Quartz format on the user interface.

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-76619	The NFS reliance on DNS to prevent VDB unresponsiveness related to DNS unreliability is now reduced.
DLPX-76203	The NFS latency for engines with many Oracle dNFS clients is now improved.
DLPX-76119	The issue with the Delphix Engine crashing or becoming unresponsive when canceling a replication job has now been resolved.
DLPX-76991	Optimized in-memory cache eviction by making minor improvements to I/O performance.

Release 6.0.9.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-28435	MS SQL instances PatchLevel will be displayed in preference to the version on the UI.

Bug number	Description
DLPX-40005	Improved the error message that is displayed when a JDBC connection cannot be established or the Oracle database Instance is unavailable.
DLPX-48080	Oracle Home Check may generate spurious faults if Oracle Home entry does not exist in /etc/oratab.
DLPX-55951	Attempting to provision a plugin-based VDB onto an incompatible OS (Windows to Linux or vice versa) is possible in the UI, and would fail with a crash requiring a restart. Now an informative error message is shown after the attempt is made.
DLPX-59613	Fixed creation of retention policies workers on replica objects after failover.
DLPX-64307	Environment refresh should ignore cluster discovery for Oracle VDBs.
DLPX-66191	Fixed the side-effects of the native Windows "Recent Files" behavior, when large numbers of PowerShell operations are being run concurrently.
DLPX-67537	Domain administrators can now create, view, and edit the alert profiles of other domain users.
DLPX-69605	Poor error message when selecting Timeflow range.
DLPX-70502	On detaching a dSource, delete backup server entry from MDS and related ones if the backup server is unused.
DLPX-71002	In case we have null values coming for recovery_model from msdb.dbo.backupset table, the user will see a generic exception for manual sync and a fault for validated sync.
DLPX-71908	Users will now see a warning when they remove any object from the replication specification list.
DLPX-72012	Prevents the same IP address from being configured on more than one network interface.

Bug number	Description
DLPX-72411	Environment names for Windows and Oracle Clusters are once again editable by users.
DLPX-72609	When we do MSSQL standalone environment discovery, the user will see a warning for databases attached to AG that are present and will not be discovered unless cluster environment discovery is selected.
DLPX-72695	Improved Oracle SnapSync performance by eliminating unnecessary calls to getTotalHoleBlocks.
DLPX-73409	Duplicate listener entry gets generated in MDS if Oracle listener is manually started with a non-uppercase name.
DLPX-73586	Fixed a display error of some snapshot names in the command line interface which showed references instead of actual names.
DLPX-73720	Provisioning an Oracle vPDB fails with "ORA-65149: PDB name conflicts with existing service name in the CDB or the PDB" if the PDB and CDB names are the same.
DLPX-74050	Environment names for Windows and Oracle Clusters are once again editable by users.
DLPX-74078	The Target Directory path is combined with other directories such as Data Directory, Archive Directory, Temp Directory, etc to build the full path for data files, archive logs, temp files, etc. As long as the combined paths are valid the V2P job proceeds.
DLPX-74197	Fixes a misleading warning about insufficient space on a replication target.
DLPX-74201	Snapshots created as a result of refresh or rewind operations will now be labeled as just "Snapshot" to avoid confusion. Users are advised to look at the Timeflow markers to know when the Timeflow operation was performed.

Bug number	Description
DLPX-74367	Delphix Engine repeatedly reports "Failed to parse logfile".
DLPX-74377	Improved diagnostics information for the case when Delphix Engine fails to connect to the Windows host.
DLPX-74387	fix an issue that causes the management service to crash under heavy CPU load.
DLPX-74398	Added handling of dangling nodes during Windows cluster environment add and refresh operations.
DLPX-74486	Delphix OS users cannot provision 12.2 TDE vPDB due to directory permissions in the default wallet location.
DLPX-74495	Enabled more logging in Delphix connector logs for timeouts.
DLPX-74681	During RAC vPDB provision, Oracle 19.9 target CDB crashes with ORA-00600 [krccfl_chunk] when BCT is enabled.
DLPX-74806	Improved error message displayed when the storage device initialization fails unexpectedly.
DLPX-74860	Provisioning the 2nd generation VDB fails if the dSource has imported read-only transportable tablespaces fails.
DLPX-74975	Allow adding invalid or unreachable paths as a shared backup location for dSources.
DLPX-75026	Invalid JDBC connections are not purged from the connection pool when the home is changed.
DLPX-75134	Improved performance for the Environment Databases pages when there are a lot of databases.
DLPX-75363	Update exception description is seen when the SnapSync fails for ASE encrypted database.

Bug number	Description
DLPX-75401	local listener set to null if oracle.lsnr.protocol_registration_order is quoted.
DLPX-75506	Fix a bug that can cause Oracle RAC VDBs to fail with stale NFSv3 mounts if NFSv4 is also enabled.
DLPX-75532	Insufficient heap memory settings on AIX cause connector and SnapSync to hang or crash.
DLPX-75663	FIPS compliant algorithms will be used while merging the old and new toolkit directories during environment refresh.
DLPX-75716	Delphix may remove Oracle VDB temp tablespaces during Snapsync.
DLPX-75735	Fixed creation of retention policies workers on replica objects after failover.
DLPX-75834	Rearranged Syslog configuration dialog inputs to avoid confusion and have a more consistent user experience.
DLPX-75858	BEQ processes can hold on to file descriptors leading to hook scripts hanging after upgrading to 6.0.7.0.
DLPX-76018	Remove hardcoded 5-minute timeout for doDropPDBKeepDatafiles.sh.
DLPX-76140	TDE SnapSync should ignore WARNING plugin violations.

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-71980	Fix a bug that was causing the Delphix Engine storage pool to fail to import on boot under certain circumstances.

Release 6.0.8.1 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-75804	Some Delphix operations may fail if mount and umount commands, on staging or target hosts, are setup to run as sudo and if sudo rules prohibit these commands from running with unrecognized options. The issue is fixed now after removing "-v" added in 6.0.8.0.

Release 6.0.8.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-49694	Skip parsing of DBCC when code 0 is not present in the DBCC page output. In case the DBCC page has code 0 but not dbid, use bdbid (present in the buffer section).
DLPX-68764	VDB SnapShot does not progress if ASE database devices are not on Delphix storage, environment monitoring raises a fault. Subsequent VDB operations like enable, disable, start, stop, delete, snapshot, or refresh will fail.
DLPX-70793	Delphix Engine should not allow linking Oracle DB with null `db_unique_name`.
DLPX-71300	For newer ASE versions (>= 15.7 SP138 and 16.0 SP02 PL05 and ASE 16.0 SP03), do not run DBCC PAGE anymore, as it was an identifier for DBCC CHECKALLOC that is already not run.
DLPX-71471	Error message asking user to manually perform disable/enable operation or correctly rename the target database back will be displayed during Start VDB, if VDB does not exist.
DLPX-71687	Provide a mechanism to enable VDBs up to filesystem mount point.

Bug number	Description
DLPX-71875	Fixed a bug that results in a memory reservation not being represented correctly in the Delphix API.
DLPX-72046	Deletion of vPDB in a vCDB shows this warning, "Encountered an error while shutting down and cleaning up Oracle files."
DLPX-72209	Downloading a support bundle is not supported at the same time that an upload of an upgrade image has been initiated by the same Delphix user.
DLPX-72319	Fixed an issue where some error dialogs would freeze in Internet Explorer 11.
DLPX-72705	Connection timeout when deleting remote shipper script can cause a timeout in LogSync client.
DLPX-72757	ASE sync using Dump History fails for large dump history files.
DLPX-72780	Timezone is set incorrectly for snapshots of Solaris 10 dSources and VDBs.
DLPX-72904	Storage capacity now includes usage from all file system objects, not just snapshots.
DLPX-73143	Fixed an issue where the support bundle dialog showed a loading spinner intermittently while jobs were running.
DLPX-73354	Traverse all shared backup locations while syncing, even if some of the paths are invalid or not reachable.
DLPX-73489	Fixed bug where adding or editing a parameter using the UI VDB Config Template "Text" tab was truncating the parameter's value.
DLPX-73602	Incorrect mount options used when a single instance RAC is linked as a standalone single instance.
DLPX-73607	Added paging for days with large numbers of snapshots to prevent slowdown.

Bug number	Description
DLPX-73623	Fixed an out-of-memory condition that occurs in SSH tunneling for encrypted log-syncs when storage latencies are high.
DLPX-73627	The help text on upgrade replication warnings have been updated to avoid confusion between Ignored and Resolved.
DLPX-73668	Fixed Missing security headers.
DLPX-73669	Cross-site request forgery (CSRF) issue in management UI.
DLPX-73727	Fixed an issue where the faults table was unable to navigate to other pages.
DLPX-73797	Fixed VDB refresh failures due to SQL Server Error 924 after setting VDB to single user mode.
DLPX-74025	Implemented logic to retry offline database along with a drop database to overcome deadlocks while off-lining or dropping the database.
DLPX-74029	VMware Hot-Add memory is not immediately reflected in the system API.
DLPX-74057	Fixed a typo in "Download Support Bundle" UI component where the word "suport" was missing a "p".
DLPX-74254	Ownership of files inside VDB now matches new owner when VDB owner is changed.
DLPX-74298	Fixed an issue where the user could not upload a keystore with a blank keystore passcode.
DLPX-74362	Fixes an issue with namespace deletion when the replication receive jobs have been cleaned up.
DLPX-74442	VDB Enable with attemptStart=false will now mount the datasets so that VDB can be started.

Bug number	Description
DLPX-74457	Cluster discovery for Oracle RAC clusters are partially failing on Solaris 10.
DLPX-74529	Fixed a bug so that an upgrade completes even when jobs fail.
DLPX-74542	Fixed a bug so that upgrade completion is properly handled after kernel upgrades.
DLPX-74645	Delphix Engine uses the uptime command to keep track of a target host reboot and auto start VDBs on the host. In some cases, the output of this command is not what is expected and causes unintended restart of a stopped VDB. This issue is now fixed.
DLPX-74656	Oracle errors during doCreateSPFile.sh are not captured.
DLPX-74704	Fixed a bug where the Dataset scroll does not extend to the bottom of a dataset list, thus truncating the status of the last dataset in the expanded group.
DLPX-74883	Prevent support bundle collection from cancelling replication.
DLPX-74911	Talaria TCP fallback fault may be misconstrued if an Oracle RAC node is down.
DLPX-74997	Prevent granting replicated roles to users.
DLPX-75083	Post upgrade cleanup task may become unresponsive while attempting a migration from 5.3.x to 6.0.x due to several threads stuck in WAITING state.
DLPX-75095	Provisioning an Oracle VDB fails if change-archivelog-mode.sh takes longer than 5 minutes.
DLPX-75134	Improved performance of the Environment Databases page under certain conditions.
DLPX-75188	Fixed "out of memory" issue when processing a large number of objects on the Target engine.

Bug number	Description
DLPX-75204	Addressed a performance issue on the Target engine when receiving large number of replicated objects.
DLPX-75208	Snapshot names are incorrectly redacted in the MDS dlpx_action table in support bundles.
DLPX-75416	Fixed a replication issue when there are sources with TLS enabled.

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-72065	Fixed a bug that can cause a Windows iSCSI initiator to fail connect to the Delphix Engine.
DLPX-72681	Console Delphix status screen shows a Python stack trace if the system is configured with a static IP address.
DLPX-73423	Console Delphix status screen shows a Python stack trace if the system has no default route.
DLPX-74216	Fixed an issue that causes management service failures in low memory situations.
DLPX-74622	Fixed a bug that can cause a replication job to fail with an internal error.
DLPX-75089	Fixed a bug that can cause NFSv3 clients to lose locks during upgrade verification.
DLPX-75524	Fixed a bug that can lead to Oracle data corruption when running VDBs on Oracle 19c with dNFS.

Release 6.0.7.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-39006	LogSync failed with "Cannot read archived log due to failure of log shipping script".
DLPX-39245	Fixed a bug that caused the management service to become inaccessible if the storage pool ran out of space.
DLPX-59155	Provisioning a VDB or vPDB failed with unclear error message 'A database with the name "xxx" already exists'.
DLPX-60317	Fixed Out of Memory issue when replicating a large number of objects.
DLPX-60947	Replica VDBs will be updated when performing a point-in-time restore.
DLPX-62805	vPDB provision did not raise an error when a non-provisionable target point-in-time was provided.
DLPX-62969	Fixed Out of Memory issue when receiving large number of replicated objects.
DLPX-64600	Skipped connecting to ASE dSources during SnapSync policy runs as it is not applicable for them, hence prevent recurrent faults that the policy throws for connectivity issues.
DLPX-67363	Maximum identify provider authentication time age can be customized for single sign-on.
DLPX-67607	Fix to make Snapsync throw exception if manifest file is missing or of 0 bytes instead of internal error with null pointer exception.
DLPX-67767	Fixed a bug that caused the upgrade to hang, while waiting for running jobs to finish.

Bug number	Description
DLPX-70821	Allow the entity id for SAML single sign-on to be a URL for compatibility with Azure AD.
DLPX-71783	doRenameDatafiles cleanup of extra files fails due to file permissions mismatch.
DLPX-72010	Fixed an issue that prevents changing the default gateway using the network setup CLI.
DLPX-72075	Maximum SAML response time skew can be customized for single sign-on.
DLPX-72191	Oracle privilege discovery not performed for all homes if an invalid home exists.
DLPX-72351	When a user tries to change credentials for a dSource, validating the credentials before updating them. In case of invalid credentials, showing user an error message about it.
DLPX-72545	Initial ORA-65294 error not reported to user when vPDB provision fails due to compatible parameter mismatch.
DLPX-72652	Fix and issue that prevents use of the NFSv4 on some versions of SUSE Linux targets.
DLPX-72698	Patching Oracle 19C vCDB leads to ORA-25153 as described in 2285159.1.
DLPX-72807	Fixed issue with SQL Server 2014 dSources with filestreams where sync failed in merging filestream directories due to long path names.
DLPX-72882	Datasets hooks script editor properly displays multiline scripts instead of as one long line on non-Chrome browsers.
DLPX-72916	Empty string in SNMPv3 USM username creation no longer throws fatal error.

Bug number	Description
DLPX-73048	Non-sys user credentials for Oracle sources cannot use password vault.
DLPX-73108	Fix a bug that prevents the API from displaying the correct number of CPUs or amount of memory assigned to a Delphix Engine after a hot-add operation.
DLPX-73201	Fix an issue that prevents the configuration of additional NICs on Azure Delphix Engines.
DLPX-73202	Fix a bug that can cause a VDB to fail to mount while other VDBs are being stopped.
DLPX-73424	Fix a bug that prevents the sysadmin from deleting a default route.
DLPX-73527	SnapSync job fails with 'internal error during execution' due to ONS/FanManager errors.
DLPX-73528	Fixed a bug that prevented accessing SDD specs from CLI.
DLPX-73611	Kerberos ticket expiration date parsing is incorrect after migration from Illumos to Linux.
DLPX-73742	Provisioning an Oracle TDE-enabled vPDB fails with the error "ORA-28367: wallet does not exist" if the TDE wallet for the target linked CDB is stored on ASM storage.
DLPX-73765	Fix a file descriptor leak that causes the management service to crash over time.
DLPX-73789	Auxiliary CDB instance uses dSource keystore location if WALLET_ROOT is configured in dSource.
DLPX-74030	CDB database password may be leaked as part of environment monitor checks that launch sqlplus command on the source or target host.

Bug number	Description
DLPX-74043	Delphix OS user cannot provision TDE-enabled vPDB due to directory permissions in the default wallet location.
DLPX-74044	Delphix OS user cannot provision TDE-enabled vPDB in Delphix-writable keystore location due to directory permissions.
DLPX-74119	Drop database fails if default database is set to any other than master.
DLPX-74164	Sync fails with db.aseldb.source.dump_history.incomplete_stripes after dump history file is purged and Use dump history is enabled for the dSource.
DLPX-74233	During failover of a namespace, if there is a collision between an environment in the namespace with one on the target engine, the namespace environment will get renamed if its host does not match that of the environment on the target.

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-73390	Improve replication receive throughput.
DLPX-73393	Improve write performance under extreme disk fragmentation.
DLPX-73280	Improve write performance under extreme disk fragmentation.

Release 6.0.6.1 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-73848	Fixed an issue that can cause the management service to fail to start after upgrade on systems that have had SNMP enabled.
DLPX-73859	Fixed a file descriptor leak triggered by faults and alerts that can cause the management service to fail.

Release 6.0.6.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-47065	VDB recovery failed when files other than archive logs were detected by Oracle.
DLPX-47493	Fixed the bug where VDB directory under the DelphixConnector directory was not being removed from the target host on MSSQL VDB deletion.
DLPX-48046	Added sorting parameter to network test APIs.
DLPX-61405	Replication may send more data than expected if masking involves dropping large DBF files.
DLPX-61525	The height of the storage configuration list was limited to show 3 disks at a time. It will now dynamically grow with the number of disks.
DLPX-63603	Increased connector timeout from 10 minutes to 30 minutes to avoid unnecessary faults due to timeout during Validated Sync operation.
DLPX-67368	Delphix Engine hostname change is now immediately reflected in Splunk events.

Bug number	Description
DLPX-67593	Fixed an issue that caused the management service to remain offline following an out-of-space condition.
DLPX-68531	Introduced better handling of UniversalConnectionPoolException errors during SnapSync.
DLPX-69759	Oracle environment discovery failed due to an unhandled exception occurring at insert into dlpx_faults.
DLPX-69852	Fixed a bug that caused network configuration problems when removing and adding additional NICs.
DLPX-70426	Redaction of usernames took forever on tables with millions of entries.
DLPX-70583	move-to-asm.sh fails if timing is set in glogin.sql.
DLPX-70638	Removed Failed Actions section of Actions sidebar, in favor of manually dismissing from Running Actions and falling to Finished Actions section.
DLPX-70653	Removal of all instances in a RAC VDB should not be allowed.
DLPX-70808	Fixed issue related to the creation of empty DisableBroker.sql on the Windows machine in case DisableBroker.sql execution fails in the first attempt.
DLPX-70896	Added more detailed error message for when the Delphix Engine fails to push a script to Windows host.
DLPX-70919	Fixed an issue that causes job progress to not update in Self-Service.
DLPX-70928	Fixed a bug that results in a Delphix Engine remaining powered on following a shutdown from the user interface.

Bug number	Description
DLPX-71093	For AG databases, a full backup is not required even recovery fork guid changed but the LSN chain didn't break because of transactional log backups.
DLPX-71097	Unable to ignore snl.bct.needed warnings if Block Change Tracking is legitimately disabled on an Oracle dSource.
DLPX-71153	Recovery of PDB should fail if the database is down after offlining datafiles.
DLPX-71370	While deleting initiator in Windows environment deletion operation, delete all the views as well for that initiator.
DLPX-71685	VDB is auto disabled if the hook fails.
DLPX-71865	Reduced the size of support bundles.
DLPX-71961	When a PDB is selected for replication, its CDB and all other PDBs in the parent CDB get automatically selected for replication. Going forward, in the above scenario, while the CDB will get selected, its other PDBs will no longer get selected.
DLPX-72031	Fixed VDB refresh operations failures due to 'DB STARTUP' background process spid greater than 50.
DLPX-72066	Migrate VDB verifies against the old configuration, rather than new.
DLPX-72083	Fix an issue that causes a fully-qualified hostname to be changed on upgrade from 5.3 to 6.0.
DLPX-72131	Added namespace support for HashiCorp password vaults.
DLPX-72265	doCreateTempfiles.sh.template exits with code 0 on failure.
DLPX-72340	Incomplete recovery not detected during provisioning.

Bug number	Description
DLPX-72386	Unlock Solaris x86 Solaris -> Linux x86 provisioning.
DLPX-72452	For clusters with long hostnames, vPDB sync fails with exception.oracle.accessor.instances.missing.
DLPX-72495	Fixed a bug that prevents the application from coming up after an upgrade
DLPX-72686	Delphix no longer logs environment variables in logs on connected hosts since this could leak sensitive information such as passwords that are sometimes stored as environment variables on database hosts such as for the ASE database.
DLPX-72730	Fixed a Snapsync performance issue.
DLPX-72790	SnapSync job fails with 'internal error during execution' due to ORA-01652.
DLPX-72862	The scenario which was causing the null pointer has been fixed now.
DLPX-73300	Validation of connection to a container for PDBs should allow connecting to CDB$ROOT.
DLPX-73311	Added platform detection for ESX 7.0u1.
DLPX-73449	Replication of policies between two engines, in a loop, could lead to OOM exceptions.

Fixes that take effect after an optional reboot

Bug number	Description
DLPX-72990	Addressed a minor CVSS 5.9 security issue with no known attack vectors.
DLPX-73067	Fix for CVE-2020-10753.
DLPX-73069	Fix for CVE-2020-12059.

Bug number	Description
DLPX-73070	Fix for CVE-2020-1760.

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-71924	Fixed a bug that causes support bundle collection to fail with an internal error.
DLPX-72918	Fixed a system crash that can happen when replicating a masked VDB using SDD.
DLPX-73147	Fixed a bug that can cause a replication source to crash if it had run replication while running on 5.0.

Release 6.0.5.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-23360	The unistallation of the Delphix Connector installer should succeed even if one of the component connector services doesn't exist.
DLPX-69155	Reduced the time taken to generate support bundles in some cases.
DLPX-67316	Recreating a controlfile against an Oracle source may yield misleading error during snapshot.
DLPX-70766	JDBC driver updated to resolve intermittent JDBC connection failures due to JDBC SSL bug.
DLPX-70785	Options passed to VDB mounts on target AIX hosts did not include read and write size values. This fix adds the rsize and wsize parameters to mount command depending on the maximum values host is configured to support.

Bug number	Description
DLPX-70741	Enabling Validated Sync while SAP ASE SnapSync job is running leaves staging database unrecoverable.
DLPX-69865	Fixed a bug that causes a network interface to become unconfigured if its MAC address changes.
DLPX-71233	If LogSync is suspended when performing SnapSync of a standby database in real-time apply, SnapSync attempts to backup the archived logs which can cause SnapSync to become unresponsive.
DLPX-69800	UEM/Hostchecker directory ownership checks fail on HPUX environment with long usernames.
DLPX-69807	Provided mechanism for the user to bypass corrupted/ incomplete jdbc libraries.
DLPX-66585	Bundle ID "fault.environment.configuration.file.owner" reports insufficient host address.
DLPX-65739	createDelphixDBUser.sh fails when "@" used in the password.
DLPX-70973	SAP ASE database provisioning fails if the source database has holes in log fragments.
DLPX-71532	Improved error handling for Oracle memory configuration errors.
DLPX-62987	Allowed assigning privileges over replicated objects through the UI.
DLPX-71593	TIMEFLOW_REPAIR incorrectly skips a log because of "wrong database".
DLPX-71751	Added NFSv4 support on AIX for Oracle and SAP ASE.
DLPX-71736	Dynamically disable RPC services if NFSv3 is no longer in use.

Bug number	Description
DLPX-71772	Network DSP Test between versions 5.3 and (6.0.3, 6.0.4) is fixed.
DLPX-71513	Replicate non-data objects like delphix engine users, authorizations, roles, permissions, policies and DB config templates.
DLPX-71305	Unable to load dummy recovery database dump due to SAP ASE error 15728.
DLPX-71172	Enabling SAP ASE dump history causes IllegalStateException in getDumpListFromLastRestoreDateAndFiles due to timestamp mismatch because of TZ.
DLPX-71178	SAP ASE internal error raised when dump history file is purged using sp_dump_history.
DLPX-65101	Fixed a race condition between a DB_DELETE job and the Oracle retention policy worker for the same container that can lead to a deadlock between the job and the worker.
DLPX-71918	Fixed an issue that causes the Delphix Engine UUID to change upon rebooting in IBM Cloud.
DLPX-71141	Fixed an issue where upgrading an Oracle dSource or changing the environment user in a linked Oracle dSource fails with the error "SOURCE_UPGRADE job for xxx failed due to an internal error during execution."
DLPX-71611	Updated UI time zone library to IANA 2020a.
DLPX-70349	Fixed a memory leak during incremental replication.
DLPX-72038	Fixed an issue that prevents 5.3.x - 6.0.x upgrade if a static route exists that goes over a DHCP interface.
DLPX-72148	Fixed a bug of always order hooks alphabetically rather than the running order set by users.

Bug number	Description
DLPX-71971	Allowed Enable/Disable of VDB if its current Timeflow has at least one snapshot.
DLPX-72115	Changed the Time Point field on a VDB back to reflecting the point on the parent the VDB was created from, but displayed in the timezone of the parent.
DLPX-71995	6.0.4.0 can no longer interact with 5.3.x remote Masking Engines.

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-70675	Fixed a bug that causes the system to become unresponsive after expanding multiple storage devices.

Release 6.0.4.2 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-72155	Fixed an issue that can render a Delphix Engine unbootable if a reboot occurs after upgrade verification but before the upgrade is applied.
DLPX-71141	Fixed an issue where upgrading an Oracle dSource or changing the environment user in a linked Oracle dSource fails with the error "SOURCE_UPGRADE job for xxx failed due to an internal error during execution."

Release 6.0.4.1 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-71930	Fix a bug that causes feature flags to be disabled when upgrading to 6.0.4.0.

Release 6.0.4.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-68173	Resolved an issue where temporary database backup/ device files created for cleaning up the target database were not being deleted.
DLPX-68773	The management stack runs out of memory as the environment monitor does not purge stale objects.
DLPX-69573	Allow linking an Oracle PDB with a lowercase name.
DLPX-69634	Allow provisioning an Oracle PDB with a lowercase name.
DLPX-69962	After detaching a PDB, perform unplug/plug, and attach again, if disabled is performed before SnapSync, the PDB can no longer be enabled.
DLPX-66045	Prevent Self-Service Container branches getting into an unusable state by blocking deleting the last segment of branches.
DLPX-7037	Snapsync performs an unnecessary checkpoint.
DLPX-68277	Users will see the detailed error message upon connection failure to Delphix connector during OS user validation and there will also be a "More" button with an error message which will open an error popup with all error details.

Bug number	Description
DLPX-70288	On the "Add Environment" screen when OS user validation will get fail, they will see the "More" button along with the error message. When the user clicks the button, an error popup opens with all details of the error and suggested action.
DLPX-70832	NFSv4 support for appdata sources running on AIX.
DLPX-68495	Fixed GUI reporting conflict information when creating a Retention Policy.
DLPX-70788	Added Environment User field for MSSQL sources in Datasets -> Configuration -> Source tab -> Staging Environment section.
DLPX-58047	Fixed bug where the sort sequence was incorrect. Fixed in Hook Operation Templates.
DLPX-67931	Provision against VPDB after create/drop a new tablespace failed with exception.oracle.targetscripts.rename.datafiles.
DLPX-59910	Comps.xml associated with Oracle Homes are marked as unparseable if they are longer than 65535 characters.
DLPX-55476	CLI provisioning fails when the mount point provided includes quotes around the path.
DLPX-71168	Changed type to text and spaced "Secret Key" and "Username Key".
ORB-3285	Support using `api.delphix.com` as a proxy for verifying the Cloud Agent binary's code signature certificate.
DLPX-71006	Allow provisioning across patch versions for Oracle versions on or after 18.X.
DLPX-71334	Migrate NTP configuration when upgrading between 5.3 and 6.0.

Bug number	Description
ORB-3286	Communication with Central management servers is now routed through the web proxy when one is configured for the Engine.
ORB-3117	Summary: Increase an action's failure message size to 256 characters so users can view large failure messages.

Release 6.0.3.1 changes

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-71339	Fixed an issue that can cause the Virtualization Management service to become inaccessible when the system memory became highly fragmented.

Release 6.0.3.0 changes

Fixes that take effect immediately after upgrade

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-68995	Improved performance of dataset deletion.
DLPX-68997 DLPX-68999	Improved single connection replication throughput.
DLPX-70697 DLPX-70703	Addressed an issue that causes long periods of I/O unresponsiveness.
DLPX-69953	Fixed a bug that can cause a Windows iSCSI initiator to fail to connect to the Delphix Engine.

Bug number	Description
DLPX-70512	Fix a hang in the I/O subsystem that can cause the Delphix Engine to become unresponsive.

Release 6.0.2.1 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-70065	Provisioning a VDB from a dSource or another VDB will fail if the following conditions are met: • Delphix Engine has at least one dSource and a VDB created using a Python plugin prior to the upgrade • Delphix Engine was upgraded to 6.0.2 • Provisioning was attempted from the UI after the completion of the upgrade
DLPX-69350	Fixed an issue that the time point attribute of a VDB is not shown.

Fixes that take effect after an optional reboot (Activated after optional Reboot)

Bug number	Description
DLPX-69864	Fixed an issue that causes MSSQL operations to hang after the reception of an iSCSI LUN reset.

Release 6.0.2.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-62806	Fixed an issue where provision against PDB after unplug/replug against the same linked PDB fails with exception.oracle.targetscripts.controlfile.create.
DLPX-67567	Oracle Source Continuity creates an unnecessary source-archive file system on zfs.

Bug number	Description
DLPX-27807	LogSync may fall behind when connected to an Oracle physical standby database in Real-Time Apply mode.
DLPX-68385	Customer provided Oracle Java missing in the search path for Java on hosts.
DLPX-62782	Reducing the number of nodes for RAC VDB and VDB in NOARCHIVELOG mode may result in ORA-00258 errors during VDB enable operation.
DLPX-62738	Better error message when plugins are uploaded out of sequence.
DLPX-68722	The product now recognizes VMware with BIOS date of 12/12/2018 as VMware 6.7.0u2.
DLPX-68579	SnapSync of Oracle 19c DB with encrypted tablespace fails with fatal exception "Block header 91 is not empty".
DLPX-68689	Fixed the issue where a huge number of error messages from ASE caused OutOfMemory Error.
DLPX-68957	Always On AG discovery will not fail in a multi-subnet environment.
DLPX-63088	Can now recover multiple Self-Service containers at the same time.
DLPX-47977	Improved handling of snapshot standby.
DLPX-64125	SnapSync failed with exception.oracle.dsource.sync.no_hosts.rac on RAC clusters with very long hostnames.
DLPX-62584	PDB enable failed after migration if mountBase has a trailing slash.
DLPX-68657	Virtualization can now fetch jobs from Masking engines configured with HTTP redirection.

Bug number	Description
DLPX-69121	It is no longer mandatory to have at least one enabled system administrator with local credentials.
DLPX-68167	Fixed an issue where too many requests were being sent for Faults from the Datasets pages.
DLPX-69082	Large stderr produced by failed rsync jobs are truncated to prevent Java OutOfMemory errors.
DLPX-58600	Datasets filter updated so that all items within a group that matches the filter string are displayed, even if the items contained in the group do not match the filter string.
DLPX-65896	VDB deletion failed due to the inability to delete LogSync worker.
DLPX-57903	Improved diagnosability of PDB discovery issues.
DLPX-68878	Fixed issue where start/stop buttons were not being displayed in the RAC instances configuration table.
DLPX-69271	Enabled replication smart failover by default.
DLPX-66715	The user-visible name for Oracle cluster objects is being replaced with the Oracle cluster name. For Windows clusters, the user-visible name is being replaced with the cluster address.
DLPX-68929	Changed default replication settings for better out of the box performance.
DLPX-68930	Improved replication throughput when sending multiple timeflows.
DLPX-69245	Fixed a memory leak that occurs when experiencing connectivity errors.
DLPX-69377	At least one non LDAP system user should be enabled when the LDAP server is being disabled.

Bug number	Description
DLPX-68575	LDAP principal fields were not being redacted in phone-home bundles.
DLPX-68528	Self Service Recover operation failed due to missing Timeflow.

Fixes that take effect after an optional reboot

Bug number	Description
DLPX-66808	Re-introduced console splash screen with IP address and service states.

Release 6.0.1.1 changes

Fixes that take effect after an optional reboot (Optional on Reboot)

Bug number	Description
DLPX-69203	Improved synchronous write performance over iSCSI.
DLPX-69167	Improved SQL Server data ingestion performance by leveraging asynchronous writes on underlying storage.
DLPX-69298	Eliminated possible data corruption on SQL server and vFiles over iSCSI that can occur when a Delphix Engine reboots.

Release 6.0.1.0 changes

Fixes that take effect immediately after upgrade

Bug number	Description
DLPX-60689	For SAP ASE, instead of using the DBCC CHECKALLOC command to fix DBID mismatch issue, the MOUNT command with FIXDBID and ALLOW_DBID_MISMATCH clauses will be used, to improve performance.

Bug number	Description
DLPX-65831	VDB snapshots need to clean unneeded ZFS datafiles.
DLPX-63949	Improved boot time after 5.3 to 6.0 migration by optimizing metadata indexing.
DLPX-66261	Upgrades to 6.0.0.0 will only be supported from a release greater than or equal to 5.3.6.0.
DLPX-66486	Snapshot of a linked database can end up with extra datafiles that do not belong to the database which might cause VDB on VDB provision to fail during rename of datafiles.
DLPX-66558	Cluster environment discovery was incomplete if the host locale was not English.
DLPX-66804	DB_LINK using incorrect user when RAC node also configured as a standalone environment.
DLPX-66768	vPDB save state lead to rollback or child provisioning failures.
DLPX-66823	Unable to link database with CL8MSWIN1251 charset.
DLPX-64538	Fixed a bug causing the timezone selector to only be visible when manually setting the time.
DLPX-66809	Removed the Windows Diagnostics Files and Directories on successful Diagnostics upload.
DLPX-67279	Provision failed when the source was from a RAC Oracle Standard Edition database and the target was Oracle Standard Edition.
DLPX-67451	Fixed an issue that sporadically caused replication to fail with an internal error.
DLPX-67454	Delphix Engine should select the highest version ojdbc driver available at ORACLE_HOME/ojdbc/lib.

Bug number	Description
DLPX-66077	Ensures child worker threads are gracefully exited when parent linked source sync job has completed/ terminated.
DLPX-45983	MSSQL Validated sync will resume when storage usage falls below the threshold if storage threshold enforcement failed in the past.
DLPX-67560	Fixed an issue where MT provision may result in ORA-02058 due to un-purged or inflight 2PC transactions on dSource.
DLPX-67594	Old timeflows and snapshots are not getting removed by snapshot retention.
LX-2020	Report the correct amount of memory allocated to EC2 Nitro instances.
DLPX-67413	Fixed an issue where VDB point in time provisioning might fail if Oracle database environment is configured in a non-English locale.
DLPX-67684	PDB provisioning failed if the source had shutdown triggers.
DLPX-67575	Fixed failure during point in time 'Virtual to Physical' provisioning.
DLPX-67668	After setting the database online give it some extra time to startup completely, before doing any further operation on it.
DLPX-67759	Redact sensitive information from phone-home data.
DLPX-64638	Validated sync stops working if Delphix cannot connect to the backup server.
DLPX-65559	Even when the staging instance is down, attempt counter to detect backup files keeps on increasing and eventually, it stops detecting backups.

Bug number	Description
DLPX-56537	When a target host is used by a large number of dSources for staging or has a large number of objects, the performance of Delphix operations like validated sync, refresh, rewind, etc can be slow due to Powershell processes being serialized.
DLPX-67894	Removing cluster resource without removing its dependency can result in cluster failure. So, added retryer logic while fetching the resource dependencies (Get-ClusterResourceDependency) and ultimately fail the operation after all the retries.
DLPX-67813	Unsupported SQL server backup type gets picked while validated sync and the operation fails while looking for the backup. So, introduced a tunable filter to automatically skip SQL backups taken by backup software not supported by yet Delphix.
DLPX-67925	Added env host connectivity toolkit support for SLES on Power9.
DLPX-67934	Retries to fetch image identifiers during Netbackup restore if there is a mismatch between MSDB and Netbackup Master.
DLPX-67655	Fixed an issue where retention enforcement can generate user-visible errors while attempting to delete snapshots with dependencies after PDB migration to new CDB.
LX-1944	EBS NVMe devices can now be used in Delphix Engines.
DLPX-68022	Fixed an issue where hostchecker 'Check Oracle DB Instance' fails on HPUX and AIX.
DLPX-68124	PDBs with lower/mixed case names will not enable after an upgrade.
DLPX-68126	Fixed a bug that limits the number of disks that can be added in GCP.

Bug number	Description
DLPX-67421	Update the primary db file names in a transaction with the Timeflow creation to make sure whenever a Timeflow is created successfully we have its primary file information.
DLPX-67440	Skip VDBs having its current Timeflow as null from 'PrimaryDbFileAvailabilityCheck' as these VDBs doesn't undergo queisceing and are recoverable by refreshing them.
DLPX-61818	Linking wizard - Target Environment step - Privileged Credentials authenticates on the selected target now.
DLPX-68117	Some non-Admin users, lack all permissions, are unable to login to upgraded engine.
DLPX-67290	A wrong version input by user while manually adding a SQL Server instance, created issues in provisioning VDBs. SQL Server version will now be auto-discovered for manually added instances on adding or refreshing the environment.
DLPX-66238	Updated error message to let know user that non discovered CDBs are filtered out from the list when linking a detached source.
DLPX-68457	When a target host is used by a large number of dSources for staging or has a large number of objects, the performance of Delphix operations like validated sync, refresh, rewind, etc can be slow due to Powershell processes being serialized.
DLPX-68484	Fixed the issue where 'lstart' column value of sysusages table was beyond the range of Integer data type by taking the Long data type to store the lstart value.
DLPX-68500	Fixed an issue where the NTP service is not started following a reboot.
DLPX-68290	Support bundle generation can be time-consuming if the engine has a large number of snapshots to process.

Bug number	Description
DLPX-67792	Fixed issue in grids in which the selection checkbox was unclickable.
DLPX-67555	Provision vPDB/vCDB fails with ORA-45900 if the parameter enable_pluggable_database is omitted when specifying database parameters for new vCDB.

Fixes that take effect after an optional reboot

Bug number	Description
DLPX-67782	Engines running 5.3 on EC2 i3 can now be migrated to 6.0.
DLPX-67961	Fixed an issue that prevents ssh access after switching to a static IP address.
DLPX-65948	Fixed a bug that could cause replication jobs to fail with internal errors
DLPX-68025	Improved boot time after 5.3 to 6.0 upgrade by reducing the overhead of setting ZFS properties.
DLPX-67868	Fixed a bug that can cause the management service to run out of memory when disabling the Splunk integration.

Release 6.0.0.0 changes

Fixes that take effect immediately after upgrade

Bug Numer	Description
DLPX-27433	The analytics GUI network graph shows newly added NIC information without requiring a management service restart.
DLPX-33998	If you add a hook script via the CLI, newlines are removed erroneously.

Bug Numer	Description
DLPX-40094	Correctly set the default type for the parameters to all operations in the CLI according to the container type.
DLPX-43215	Exclude sybsecurity from the list of auto-discovered databases.
DLPX-48712	Java 6 packages are no longer included in the product image.
DLPX-48280	When a user is set with the Provisioner role the 'provision' button does not appear, meaning anyone set with this role only is unable to provision VDBs.
DLPX-53996	The Delphix Engine does not provide instructions to browsers to avoid caching HTTP responses (pages).
DLPX-54740	Ensure Windows mount points are always unmounted as part VDB refreshes to prevent future VDB refreshes from failing due to "ERROR_ASSIGN_MOUNTPATH: failed to assign mount path for disk at="">>, error="">,>
DLPX-55282	In environments where the vPDB has been provisioned using a Delphix provisioned virtual CDB, shutting down the virtual PDB causes it to get into an incorrect "Cannot monitor" state, this has now been fixed to show the correct "Stopped" state.
DLPX-55598	Fixed an issue where vPDB refresh/rollback triggers spurious vCDB restart jobs, after vPDB+vCDB auto-restart.
DLPX-55829	Validated Sync can fail when monitoring ASE backup servers started by using the $DSLISTEN environment variable instead of the "-S" argument. This can be worked around by accessing $DSLISTEN in the RUN_xxxxx script and pass it down as -S.
DLPX-55958	VDBs with no snapshots failed to re-enable after a Delphix Engine upgrade, this has now been fixed.
DLPX-57454	Display underlying ssh error when environment host connections fail.

Bug Numer	Description
DLPX-58519	Enable Oracle LiveSource when LiveSource is in RESYNC_NEEDED state currently re-start Oracle Redo Apply. Oracle Redo Apply should not be restarted in this state.
DLPX-58760	Fixed a TCP port leak in the network throughput test feature.
DLPX-58845	Provisioning vFiles to the same host using different OS Environment Users no longer fails.
DLPX-59772	The API to list all snapshots consumes a significant amount of memory when there are more than 100,000 snapshots on the engine.
DLPX-60356	Fixed an issue where Oracle remote listener registration fails if set to empty string.
DLPX-60603	Network settings dialog now displays actual MTU value rather than a checkbox.
DLPX-60907	Fixed an issue where the Environment Monitor on Redhat 6.9 and 6.10 might throw unidentified version errors.
DLPX-60979	When user configures connection strings manually, these connect strings can end up connecting to incorrect PDBs/CDBs causing invalid snapshots. Verify that each connection to a PDB/CDB connects to the expected PDB/CDB.
DLPX-60993	Delphix backups create controlfile records; in rare circumstances, these records can cause invalid snapshots. To avoid this problem, remove Delphix backups control file records when using SCN-based SnapSyncs once a SnapSync completes successfully.
DLPX-62094	Allow certificates to expire after issuer certificate expiration.
DLPX-62241	Reduce SSH connections by temporarily preserving and reusing existing Delphix<->host connections where possible.

Bug Numer	Description
DLPX-62781	Spurious job event "DISCOVERED_TO_MANUAL_ORACLE_CLUSTER_NODES" no longer shows up for non-Oracle RAC environment refreshes.
DLPX-62892	In Oracle versions 18c and 19c, an Oracle bug can prevent the datafile headers from being updated for a standby database when managed recovery is running, resulting in failed SnapSync operations. Alert the user that an Oracle patch might be needed.
DLPX-62962	Removed unneeded EMPTY_RENEGOTIATION cipher
DLPX-62998	Fixed an issue where stale file mounts may be leftover when vPDB provision fails.
DLPX-63469	Initial setup now fails if the system was not provisioned with enough storage.
DLPX-63600	Network settings dialog now displays actual MTU value rather than a checkbox.
DLPX-64641	Fixed an issue where the last snapshot of a vPDB Timeflow can be deleted after the vPDB has been disabled, thus leaving the vPDB in a state with no provisionable snapshots.
DLPX-64711	Allow provisioning to complete when source CDB includes PDBs in a broken state.
DLPX-66020	Provision should remove files present in datafile filesystem that are not part of the database when provisioning a VDB from a VDB.
DLPX-67299	ASE environment discovery will not fail if there is a mismatch of "dataserver name argument" and value of "@@servername".

Fixes that take effect after an optional reboot

Bug number	Description
DLPX-57384	Fixed a system hang caused by a deadlock in ZFS.

Fixes that take effect after an optional reboot

Bug number	Description
DLPX-57384	Fixed a system hang caused by a deadlock in ZFS.

Known issues

Version 14.0.0.0

Key	Summary	Workaroound
DLPX-56944	In Solaris 5.11 environments where Delphix OS user shell is set to `csh`, host parameters may be improperly parsed, leading to environment refresh and other job failures.	Contact Delphix Support for corrective actions.
DLPX-56978	Certain usernames are not available for use, as they are reserved system words (e.g. "root", "postgres", "delphix").	None
DLPX-56979	LDAP server test fails if authentication is set to DIGEST-MD5, but setup still works correctly.	None
DLPX-57142	The Job dashboard does not display the user that invoked each Job.	None
DLPX-57412	CLI parameters must be used exactly as described in documentation, including spelling and capitalization, or they are ignored.	None
DLPX-57673	Support bundle generation can be time consuming if the engine has a large number of core files to process.	None
DLPX-57823	Changing the Compliance Engine used by a VDB's masking job can lead to an internal error.	None
DLPX-58185	Changing a Custom Policy while it is running on an object can cause the Policy execution to fail.	None
DLPX-58226	Completed cleanup jobs may still show as running, even after an upgrade.	None

Key	Summary	Workaroound
DLPX-59473	It is not possible to set a password policy that prevents an administrator from re-using a previous password, though this can be set for other users.	None
DLPX-66155	Restarting a Delphix Engine during a network throughput test is not recommended as it may lead to a system hang.	None
DLPX-66860	For SSO/SAML, ADFS requires explicit rule to transform `emailAddress` attribute into `nameid`.	Create an explicit rule in ADFS that transforms the `emailAddress` attribute into a `nameid`. The rule type must be "Transform an incoming claim". The incoming claim type must be "Email address" and the outgoing claim type "Name ID". The `nameid` format must be "Email address".
DLPX-77849	An excessive number of connections to the SQL Server instance can cause infrastructure issues, leading the LSASS.exe to crash and the host to reboot.	Rather than using a Windows domain user for authentication, switch the dSource to use a database user. It seems SQL Server may be able to handle the massive number of connections that Delphix is establishing for each dSource better than Windows LSASS.EXE.
DLPX-77986	On a sybase host with multiple Sybase instances, If access to the first instance picked up for discovery fails due to invalid credentials, the discovery job will exit immediately preventing discovery of the remaining instances.	Fix any credential failures.
DLPX-78589	During the upgrade, when Delphix was trying to run the ASE "UNMOUNT" command while trying to quiesce VDBs, UNMOUNT command got hung as this command does not run under a timeout and due to this, the upgrade job stalled.	When the ASE UNMOUNT command is hung, the ASE instance must be restarted.
DLPX-80193	A vague error message appears in a case where provisioning fails due to a database being in read-only mode.	Change source database to read-write, create a backup and sync to the dSource and provision VDB.

Key	Summary	Workaroound
DLPX-81300	Windows Environment Add/Refresh operations may fail if the iSCSI Initiator Name is not a valid IQN.	Set the initiator name to a valid name, e.g. iqn.1991-05.com.microsoft:10-43-47-42.qa-ad.delphix.com. The “Default” button can also be selected in Windows iSCSI configuration to reset to the default (valid) name.
DLPX-81478	If a transaction log is taken using the `standby_access` option for a SAP ASE database, the validated sync worker will not be able to restore that log, and will fail with Delphix fatal exception (as Delphix currently does not support this option).	Change the backup script creating transaction logs to Ingest FULL backups.
DLPX-81559	After upgrading an Oracle VDB from 12c to 19c, VDB refresh fails with "Failed to mount database instance" due to ORA-01130 or ORA-00201 errors.	Update the VDB parameters (either directly or via a VDB Config Template). See this knowledge base article.
DLPX-83643	After engine upgrade or after disable and enable of vPDB, NFS version in GUI/API may be incorrect; for a vPDB in a vCDB, it was mounted with NFSv3 and now the host supports NFSv4 mounts. This condition corrects itself on the next refresh operation.	Perform a vPDB refresh operation to update the NFS protocol version to 4.
DLPX-84075	SQL Server VDB Refresh operations may fail if the PowerShell command `[IO.Path]::GetTempFileName()` returns no value.	Go to the temp directory below and delete all files from it: `C:\Window\ServiceProfiles\NetworkService\appdata\local\temp`
DLPX-84253	Users should be able to delete Delphix-generated CA certificate if no dependencies exist.	Mark related faults as Ignored, but note that the CA certificate will still appear in Truststore.
DLPX-84598	A sysadmin user cannot view actions initiated by a different sysadmin user.	None
DLPX-84710	Improved handling of missing networking devices after instance type migration required.	None

Key	Summary	Workaroound
DLPX-86181	Provision, refresh, rewind, or start operations on a virtual database on a Solaris host with NFSv4 enabled may become stuck indefinitely due to mount process getting stuck on the Solaris NFS client.	Switch the NFS protocol to v3 using the Delphix engine CLI, manually terminate the stuck mount processes on the Solaris host and re-attempt the failed Delphix operation.
DLPX-86894	After a canceled refresh operation, VDB refresh or delete may fail due to error code `exception.oracle.vdb.no.virtual.datafiles.found`. As a workaround, disable the VDB and try the operation again.	Disable the VDB and re-attempt the failed operation.
DLPX-87320	Exporting a VDB or vPDB to ASM fails with "ORA-32771: cannot add file to bigfile tablespace" when the database has a bigfile temporary tablespace	If using the `move-to-asm` script, use an older version of script from Delphix engine v6.0.15.0 or earlier. No workaround if attempting to export a VDB or vPDB to a physical ASM or Exadata database using the `database export` CLI.

Version 13.0.0.0

Key	Summary	Workaround
DLPX-56944	In Solaris 5.11 environments where the Delphix OS user shell is set to `csh`, host parameters may be improperly parsed, leading to ‘environment refresh’ and other job failures.	Contact Delphix Support for corrective actions.
DLPX-56978	Certain usernames are not available for use, as they are reserved system words (e.g. "root", "postgres", "delphix").	None
DLPX-56979	LDAP server test fails if authentication is set to DIGEST-MD5, but setup still works correctly.	None
DLPX-57142	The Job dashboard does not display the user that invoked each job.	None
DLPX-57412	CLI parameters must be used exactly as described in documentation, including spelling and capitalization, or they are ignored.	None

DLPX-57673	Support bundle generation can be time consuming if the engine has a large number of core files to process.	None
DLPX-57823	Changing the Compliance engine used by a VDB's masking job can lead to an internal error.	None
DLPX-58185	Changing a Custom Policy while it is running on an object can cause the Policy execution to fail.	None
DLPX-58226	Completed cleanup jobs may still show as running, even after an upgrade.	None
DLPX-59473	It is not possible to set a password policy that prevents an Administrator from re-using a previous password, though this can be set for other users.	None
DLPX-66155	Restarting a Delphix Engine during a network throughput test is not recommended, as it may lead to a system hang.	None
DLPX-66860	For SSO/SAML, ADFS requires an explicit rule to transform the emailAddress attribute into nameid.	Create an explicit rule in ADFS that transforms the emailAddress attribute into a nameid. The rule type must be "Transform an incoming claim". The incoming claim type must be "Email address" and the outgoing claim type is "Name ID". The nameid format must be "Email address".
DLPX-77849	An excessive number of connections to the SQL Server instance can cause infrastructure issues, leading the LSASS.exe to crash and the host to reboot.	Rather than using a Windows domain user for authentication, switch the dSource to use a database user. It seems SQL Server may be able to handle the massive number of connections that Delphix is establishing for each dSource better than Windows LSASS.exe.
DLPX-77986	On a Sybase host with multiple Sybase instances, if access to the first instance picked up for discovery fails due to invalid credentials, the discovery job will exit immediately (preventing discovery of the remaining instances).	Fix any credential failures.

DLPX-78589	During an upgrade, when Delphix tried to run the ASE "UNMOUNT" command, while trying to quiesce VDBs, the UNMOUNT command got hung (as this command doesn't run under a timeout); due to this, the upgrade job stalled.	When the ASE UNMOUNT command is hung, the ASE instance must be restarted.
DLPX-80193	Vague error message appears in a case where provisioning fails due to a database being in read-only mode.	Change source the database to read-write, create a backup, sync to the dSource, and provision VDB.
DLPX-81300	Windows Environment Add/Refresh operations may fail if the iSCSI Initiator Name is not a valid IQN.	Set the initiator name to a valid name, eg: iqn.1991-05.com.microsoft:10-43-47-42.qa-ad.delphix.com. The “Default” button can also be selected in the Windows iSCSI configuration to reset to the default (valid) name.
DLPX-81478	If a transaction log is taken using the `standby_access` option for an SAP ASE database, the validated sync worker will not be able to restore that log and will fail with a Delphix fatal exception (as Delphix currently does not support this option).	Change the backup script creating transaction logs to ingest FULL backups.
DLPX-81559	After upgrading an Oracle VDB from 12c to 19c, a VDB refresh fails with, "Failed to mount database instance" due to ORA-01130 or ORA-00201 errors.	Update the VDB parameters (either directly or via a VDB Config Template). See this knowledge base article.
DLPX-83643	After engine upgrade or after disable and enable of a vPDB, the NFS version in the GUI/API may be incorrect if it's a vPDB in a vCDB; this means it was mounted with NFSv3, and now the host supports NFSv4 mounts. This condition corrects itself on the next refresh operation.	Perform a vPDB refresh operation to update the NFS protocol version to 4.
DLPX-84075	SQL Server VDB Refresh operations may fail if the PowerShell command `[IO.Path]::GetTempFileName()` returns no value.	Go to the following temp directory and delete all the files from it: `C:\Window\ServiceProfiles\NetworkService\appdata\local\temp`
DLPX-84253	Unable to delete Delphix-generated CA certificate if no dependencies exist.	Mark the fault Ignored, but the CA certificate will still appear in Truststore.

DLPX-84598	A sysadmin user cannot view actions initiated by a different sysadmin user.	None
DLPX-84710	Improved handling of missing networking devices after instance type migration required.	None
DLPX-85770	If a Oracle VDB/vPDB is already enabled, a Self-Service container `start` operation or the VDB/vPDB `enable` operation may fail with `exception.oracle.vdb.database.exists.enable.not.allowed` / `exception.oracle.vdb.pdb.exists.enable.not.allowed`.	Retry the failed VDB/vPDB start or self-service container start operation, it should succeed without any error.
DLPX-86181	Provision, refresh, rewind or start operations on a virtual database on a Solaris host with NFSv4 enabled may become stuck indefinitely due to the mount process getting stuck on the Solaris NFS client.	Switch the NFS protocol to v3 using the Delphix engine CLI, manually terminate the stuck mount processes on the Solaris host and re-attempt the failed Delphix operation.
DLPX-87006	Refresh operation fails for the self service containers having ordered sources.	None. Contact Delphix support for corrective actions.

Version 12.0.0.0

Key	Summary	Workaround
DLPX-56944	In Solaris 5.11 environments where the Delphix OS user shell is set to csh, host parameters may be improperly parsed, leading to environment refresh and other job failures.	Contact Delphix Support for corrective actions.
DLPX-56978	Certain usernames are not available for use, as they are reserved system words (e.g. "root", "postgres", "delphix").	None
DLPX-56979	LDAP server test fails if authentication is set to DIGEST-MD5, but setup still works correctly.	None
DLPX-57142	The Job dashboard does not display the user that invoked each Job.	None

Key	Summary	Workaround
DLPX-57412	CLI parameters must be used exactly as described in documentation, including spelling and capitalization, or they are ignored.	None
DLPX-57673	Support bundle generation can be time consuming if the engine has a large number of core files to process.	None
DLPX-57823	Changing the masking engine used by a VDB's masking job can lead to an internal error.	None
DLPX-58185	Changing a Custom Policy while it is running on an object can cause the Policy execution to fail.	None
DLPX-58226	A completed Cleanup Job, after upgrade, still shows as "running".	None
DLPX-59473	It is not possible to set a password policy that prevents an Administrator from re-using a previous password, though this can be set for other users.	None
DLPX-66155	Restarting a Delphix Engine during a network throughput test is not recommended, as it may lead to a system hang.	None
DLPX-66860	For SSO/SAML, ADFS requires explicit rule to transform emailAddress attribute into nameid.	Create an explicit rule in ADFS that transforms the emailAddress attribute into a nameid. The rule type must be "Transform an incoming claim". The incoming claim type must be "Email address" and the outgoing claim type is "Name ID". The nameid format must be "Email address".
DLPX-77849	An excessive number of connections to SQL Server Instances causes infrastructure issues and leads LSASS.exe to crash, and host to reboot.	Rather than using a Windows domain user for authentication, switch the dSource to use a database user. It seems SQL Server may be able to handle the massive number of connections that Delphix is establishing for each dSource better than Windows LSASS.EXE.

Key	Summary	Workaround
DLPX-77986	On a sybase host with multiple Sybase instances, If access to the first instance picked up for discovery fails due to invalid credentials, the discovery job will exit immediately preventing discovery of the remaining instances.	Fix any credential failures
DLPX-78589	During the upgrade, when Delphix was trying to run the ASE "UNMOUNT" command while trying to quiesce VDBs, the UNMOUNT command gets hung (as this command doesn't run under a timeout and due to this, the upgrade job stalled).	When the ASE UNMOUNT command is hung, the ASE instance must be restarted.
DLPX-80193	Provide a proper error message in case provisioning fails due to database being in read-only mode.	Change source database to read-write, create a backup and sync to the dSource, and provision VDB.
DLPX-80920	A failed Delphix Engine upgrade can cause plugin operations to fail with "grpc_status 14".	Verify the upgrade and then apply the upgrade. If the issue occurs after a failed upgrade, restarting the management stack will resolve the issue.
DLPX-81300	Windows environment Add/Refresh operations may fail if the iSCSI Initiator Name is not a valid IQN.	Set initiator name to a valid name, eg: iqn.1991-05.com.microsoft:10-43-47-42.qa-ad.delphix.com The “Default” button can also be selected in the windows iSCSI configuration to reset to the default (valid) name.
DLPX-81478	If a transaction log is taken using the standby_access option for a SAP ASE database, the validated sync worker will not be able to restore that log, and will fail with Delphix fatal exception (as Delphix currently does not support this option).	Change the backup script creating transaction logs to ingest FULL backups.
DLPX-81559	After upgrading an Oracle VDB from 12c to 19c, VDB refresh fails with "Failed to mount database instance", due to ORA-01130 or ORA-00201 errors.	Update the VDB parameters (either directly or via a VDB Config Template). See this knowledge base article.

Key	Summary	Workaround
DLPX-83643	After an engine upgrade or after the disable/enable of a vPDB, the NFS version in the GUI/API may be incorrect if it's a vPDB in a vCDB or it was mounted with NFSv3 (and now the host supports NFSv4 mounts). This condition corrects itself on the next refresh operation.	Perform a vPDB refresh operation to update the NFS protocol version to 4.
DLPX-84075	SQL Server VDB Refresh operations may fail if the PowerShell command [IO.Path]::GetTempFileName() returns no value.	Go to the below mentioned temp directory, and delete all the files from it: C:\Window\ServiceProfiles\NetworkService\appdata\local\temp.
DLPX-84253	Users should be able to delete the Delphix-generated CA certificate, if no dependencies exist.	Users may mark the fault Ignored, but will need to be aware the CA certificate will still appear in Truststore.
DLPX-84598	Sysadmin users cannot view actions initiated by different sysadmin users.	None
DLPX-84710	Improved handling of missing networking devices after instance type migration.	None
DLPX-85469	JSON file masking does not support the use of a multi-column algorithm on (a) Fields in two or more different arrays (b) Fields at different levels in a single multi-dimensional array.	None
DLPX-85770	If an Oracle VDB/vPDB is already enabled, a self-service container start operation or the VDB/vPDB enable operation may fail with: exception.oracle.vdb.database.exists.enable.not.allowed/ exception.oracle.vdb.pdb.exists.enable.not.allowed	Retry the failed VDB/vPDB start or self-service container start operation, it should succeed without any error.
DLPX-86109	Oracle VDB unquiesce/enable may fail after a failed quiesce/disable on 10.0.0.X or 11.0.0.X.	To get disabled Oracle VDBs back to an enabled state, manually disable sources via the Delphix CLI and enable them back via the Delphix CLI/UI.
DLPX-86181	Provision, refresh, rewind, or start operations on a virtual database on a Solaris host with NFSv4 enabled may become stuck indefinitely due to mount process getting stuck on the Solaris NFS client.	Switch the NFS protocol to v3 using the Delphix engine CLI, manually terminate the stuck mount processes on the Solaris host and re-attempt the failed Delphix operation.

Key	Summary	Workaround
DLPX-86344	A failed Delphix Engine upgrade can cause plugin operation to fail with grpc_status 14.	Verify the upgrade and then apply the upgrade. If the issue is hit after a failed upgrade, Restarting the management stack will resolve the issue.
DLPX-86842	If there is any detached or unlinked Oracle dSource, upgrade failure may occur during `verify upgrade` job while upgrading to 12.0.0.0.	Re-link or delete the dSources before trying to upgrade to 12.0.0.0, OR upgrade to version 11.0.0.0 or 13.0.0.0.
DLPX-87006	Refresh operation fails for the self service containers having ordered sources.	None. Contact Delphix support for corrective actions.

Version 11.0.0.0

Key	Summary	Workaround
DLPX-56978	Certain usernames are not available for use, as they are reserved system words (e.g. "root", "postgres", "delphix").	None
DLPX-56979	LDAP server test fails if authentication is set to DIGEST-MD5, but setup still works correctly.	None
DLPX-57142	The Job dashboard does not display the user that invoked each job.	None
DLPX-57412	CLI parameters must be used exactly as described in documentation, including spelling and capitalization, or they are ignored.	None
DLPX-57673	Support bundle generation can be time consuming if the engine has a large number of core files to process.	None
DLPX-57823	Changing the Continuous Compliance engine used by a VDB's masking job can lead to an internal error.	None

Key	Summary	Workaround
DLPX-58185	Changing a Custom Policy while it is running on an object can cause the Policy execution to fail.	None
DLPX-58226	Completed Cleanup Job, after upgrade, will still show as running.	None
DLPX-59473	It is not possible to set a password policy that prevents an Administrator from re-using a previous password, though this can be set for other users.	None
DLPX-66155	Restarting a Delphix engine during a network throughput test is not recommended, as it may lead to a system hang.	None
DLPX-66860	For SSO/SAML, ADFS requires an explicit rule to transform emailAddress attribute into nameid.	Create an explicit rule in ADFS that transforms the emailAddress attribute into a nameid. The rule type must be "Transform an incoming claim". The incoming claim type must be "Email address" and the outgoing claim type "Name ID". The nameid format must be "Email address".
DLPX-77849	Excessive number of connections to SQL Server instance causes infrastructure issues and leads the LSASS.exe to crash and the host to reboot.	Rather than using a Windows domain user for authentication, switch the dSource to use a database user. SQL Server should be able to handle the massive number of connections that Delphix is establishing for each dSource better than LSASS.EXE.
DLPX-77986	On a Sybase host with multiple Sybase instances, if access to the first instance picked up for discovery fails due to invalid credentials, the discovery job will exit immediately, preventing discovery of the remaining instances.	Fix any credential failures.
DLPX-78589	When Delphix tries to run the ASE "UNMOUNT" command (while trying to quiesce VDBs) during the upgrade, the UNMOUNT command gets hung up (this command does not have a timeout). Because of this, the upgrade job stalled.	When the ASE UNMOUNT command is hung up, the ASE instance must be restarted.

Key	Summary	Workaround
DLPX-80193	Provides a proper error message in case provisioning fails due to database being in read-only mode.	Change the source database to read-write, create a backup, and sync to the dSource, then provision the VDB.
DLPX-80920	A failed Delphix engine upgrade can cause plugin operation to fail with "grpc_status 14".	Verify the upgrade and then apply the upgrade. If the issue occurs after a failed upgrade, restarting the management stack will resolve the issue.
DLPX-81300	Windows Environment Add/Refresh operations may fail if the iSCSI Initiator Name is not a valid IQN.	Set the initiator name to a valid name, eg: iqn.1991-05.com.microsoft:10-43-47-42.qa-ad.delphix.com. The “Default” button can also be selected in the Windows iSCSI configuration to reset to the default (valid) name.
DLPX-81478	If the transaction log is taken using the `standby_access` option for a SAP ASE database, the validated sync worker will not be able to restore that log and will fail with a fatal exception, as Delphix currently does not support this option.	Change the backup script creating transaction logs. Ingest FULL backups.
DLPX-81559	After upgrading an Oracle VDB from 12c to 19c, VDB refresh fails with "Failed to mount database instance", due to ORA-01130 or ORA-00201 errors.	Update the VDB parameters (either directly or via a VDB config template). See this knowledge base article
DLPX-83430	Initial configuration of Syslog breaks most of the pre-existing appenders.	Restart the management stack.
DLPX-83643	After an engine upgrade or after disable and enable of vPDB, the NFS version in the GUI/API may be incorrect if it's a vPDB in a vCDB, it was mounted with NFSv3 and now the host supports NFSv4 mounts. This condition corrects itself on the next refresh operation.	Perform a vPDB refresh operation to update the NFS protocol version to 4.
DLPX-84075	SQL Server VDB Refresh operations may fail if the PowerShell command [IO.Path]::GetTempFileName() returns no value.	Go to the below mentioned temp directory and delete all files from it: C:\Window\ServiceProfiles\NetworkService\appdata\local\temp.
DLPX-84253	Users should be able to delete Delphix-generated CA certificate if no dependencies exist.	User may mark the fault as ignored, but will need to be aware the CA certificate will still appear in TrustStore.

Key	Summary	Workaround
DLPX-84598	A sysadmin user cannot view actions initiated by different sysadmin user	None
DLPX-85578	Replaced the Win32_Volume class output with the mountvol output to fetch volumeId for Delphix iSCSI mount points.	None
DLPX-85770	If an Oracle VDB/vPDB is already enabled, a Self-Service container start operation or the VDB/vPDB enable operation may fail with exception.oracle.vdb.database.exists.enable.not.allowed/ exception.oracle.vdb.pdb.exists.enable.not.allowed.	Retry the failed VDB/vPDB start or Self-Service container start operation, it should succeed without any error.
DLPX-86109	Oracle VDB unquiesce/enable may fail after a failed quiesce/disable on 10.0.0.X or 11.0.0.X.	To get disabled Oracle VDBs back to Enabled state, manually disable such sources via the Delphix CLI and enable them back via the Delphix CLI/UI.
DLPX-86181	Provision, refresh, rewind, or start operations on a virtual database on a Solaris host with NFSv4 enabled may become stuck indefinitely due to the mount process getting stuck on the Solaris NFS client.	Switch the NFS protocol to v3 using the Delphix engine CLI, manually terminate the stuck mount processes on the Solaris host and re-attempt the failed Delphix operation.

Version 10.0.0.0

Key	Summary	Workaround
DLPX-56944	In Solaris 5.11 Environments where Delphix OS user shell is set to csh, host parameters may be improperly parsed leading to Environment refresh and other job failures.	Contact Delphix Support for corrective actions.
DLPX-56978	Certain usernames are not available for use as they are reserved system words (e.g. "root", "postgres", "delphix")	None
DLPX-56979	LDAP server test fails if authentication is set to DIGEST-MD5, but setup still works correctly	None

Key	Summary	Workaround
DLPX-57142	The Job dashboard does not display the user that invoked each Job	None
DLPX-57412	CLI parameters must be used exactly as described in documentation, including spelling and capitalization, or they are ignored	None
DLPX-57673	Support bundle generation can be time consuming if the engine has a large number of core files to process	None
DLPX-57823	Changing the masking engine used by a VDB's masking job can lead to an internal error	None
DLPX-58185	Changing a Custom Policy while it is running on an object can cause the Policy execution to fail	Disable/enable after fixing the problem
DLPX-58226	Completed Cleanup Job, After Upgrade, Still Shows as Running	None
DLPX-59473	It is not possible to set a password policy that prevents an Administrator from re-using a previous password, though this can be set for other users	None
DLPX-66155	Restarting a Continuous Data Engine during a network throughput test is not recommended as it may lead to a system hang.	Create an explicit rule in ADFS that transforms the emailAddress attribute into a nameid. The rule type must be "Transform an incoming claim". The incoming claim type must be "Email address" and the outgoing claim type "Name ID". The nameid format must be "Email address".
DLPX-66860	For SSO/SAML, ADFS requires explicit rule to transform emailAddress attribute into nameid.	Rather than using a Windows domain user for authentication switch the dSource to use a database user. It seems SQL Server may be able to handle the massive number of connections that Continuous Data is establishing for each dSource better than Windows LSASS.EXE.

Key	Summary	Workaround
DLPX-77849	Excessive number of connections to SQL Server Instance observed by Delphix Environment User. This causes infra issues and leads to LSASS.exe to crash and host to reboot	Rather than using a Windows domain user for authentication switch the dSource to use a database user. It seems SQL Server may be able to handle the massive number of connections that Delphix is establishing for each dSource better than Windows LSASS.EXE.
DLPX-77986	On a SAP ASE host with multiple SAP ASE instances, If access to the first instance picked up for discovery fails due to invalid credentials, the discovery job will exit immediately preventing discovery of the remaining instances.	Fix any credential failures
DLPX-78589	During the upgrade, when Delphix was trying to run the SAP ASE "UNMOUNT" command while trying to quiesce VDBs, UNMOUNT command hung as this command doesn't run under a timeout and due to this upgrade job stalled.	When the ASE UNMOUNT command is hung, the ASE instance must be restarted.
DLPX-80193	Provide proper error message in case provisioning fails due to database is in read-only mode.	Change source database to read-write, create a backup and sync to the dSource and provision VDB.
DLPX-80920	A failed Delphix engine upgrade can cause plugin (for plugins using docker runtime) operation to fail with grpc_status 14.	1. Verify the upgrade and then apply the upgrade. 2. If the issue is hit after a failed upgrade, Restarting the management stack will resolve the issue.
DLPX-81300	Windows Environment Add/Refresh operations may fail if the iSCSI Initiator Name is not a valid IQN	Set initiator name to a valid name, eg: iqn.1991-05.com.microsoft:10-43-47-42.qa-ad.delphix.com. The “Default” button can also be selected in windows iscsi configuration to reset to the default (valid) name.
DLPX-81478	If transaction log is taken using `standby_access` option for SAP ASE database, the validated sync worker will not be able to restore that log and will fail with Delphix Fatal Exception as Delphix currently doesnt support this option.	Change the backup script creating transaction logs Ingest FULL backups

Key	Summary	Workaround
DLPX-81559	After upgrading an Oracle VDB from 12c to 19c, the VDB refresh fails with "Failed to mount database instance" due to ORA-01130 or ORA-00201 errors.	Update the VDB parameters (either directly or via a VDB Config Template). For more information, please see KBA6234
DLPX-83643	If you upgrade the engine or disable and then enable a virtual PDB, the displayed NFS version in the GUI or API might be incorrect. This happens if the virtual PDB is inside a virtual CDB and was previously mounted with NFSv3, but the host now supports NFSv4 mounts. However, the issue will automatically resolve itself during the next refresh operation.	Perform a vPDB refresh operation to update the NFS protocol version to 4.
DLPX-84075	SQL Server VDB Refresh operations may fail if the PowerShell command [IO.Path]::GetTempFileName() returns no value"	Go to the below mentioned temp directory, and delete all the files from it: C:\Window\ServiceProfiles\NetworkService\appdata\local\temp
DLPX-84598	sysadmin user cannot view actions initiated by different sysadmin user	Perform an export with the same unique name as of the VDB and then manually change the physical database's unique name.
DLPX-85493	MSSQL: Linking and AttachSource operations will fail with unnecessary permissions of source user on the staging host and staging database. This issue can be seen from 6.0.17.0 onwards.	Adding source user to the staging instance with read permission on master database and Giving write permission on connector directory path {Connector_Installation}/sourceValidation

Version 9.0.0.0

Key	Summary	Workaround
DLPX-84977	For RAC environments, export of an Oracle VDB to a database with a new unique name is not supported.	Perform the export with the same unique name as of the VDB and then manually change the physical database's unique name.
DLPX-84423	Export operation does not retain the read-only mode in the newly converted physical database after an in-place conversion of a read-only Oracle VDB or vPDB	None

Key	Summary	Workaround
DLPX-84598	sysadmin user cannot view actions initiated by different sysadmin user	None
DLPX-84655	A false warning message that listener registration was not successful is posted when enabling a VDB or a vCDB, users can ignore this message.	Ignore the warning message.
DLPX-84679	Unable to add Staging push PDB if Staging Environment has more than one repository.	From the Add dSource Wizard dSource Configuration screen for Staging Push. • Before selecting PDB as the Database Type, select CDB. • Select the correct Staging Environment and repository. • Select PDB for the Database Type. • Now CDB will be shown under Container Database • Fill in the remaining PDB details
DLPX-85076	Instance init file is not copied to $ORACLE_BASE_CONFIG/dbs during staging push dSource linking	Specify the pfile parameter in the startup command as `pfile='/<MOUNT_BASE>/<DATABASE_UNIQUE_NAME>/script/<DATABASE_SID>/init<DATABASE_SID>.ora'`

Version 8.0.0.0

Key	Summary	Workaround
DLPX-84679	Unable to add Staging push PDB if Staging Environment has more than one repository.	From the Add dSource Wizard dSource Configuration screen for Staging Push. • Before selecting PDB as the Database Type, select CDB. • Select the correct Staging Environment and repository. • Select PDB for the Database Type. • Now CDB will be shown under Container Database • Fill in the remaining PDB details

Key	Summary	Workaround
DLPX-85076	Instance init file is not copied to $ORACLE_BASE_CONFIG/dbs during staging push dSource linking	Specify the pfile parameter to startup command as `pfile parameter=/<MOUNT_BASE>/<DATABASE_UNIQUE_NAME>/script/<DATABASE_SID>/init<DATABASE_SID>.ora`

Version 7.0.0.0

⚠ **Upgrades from versions < 6.0.17.0 to any version between 6.0.17.0 and 7.0.0.0 on a replication target engine** may fail due to the management services being down, which will require a support call. This applies to “Delay the Reboot” or “Apply Now” upgrades.

Key	Summary	Workaround
DLPX-84255	In a Single Engine Continuous Vault product, adding a new Sybase dSource to a locked group may result in the background environment monitoring process to stop working.	No workaround.
DLPX-84495	If upgrading from versions < 6.0.17.0 to any version between 6.0.17.0 and 7.0.0.0, a support call is needed.	Call support.
DLPX-84679	Unable to add Staging push PDB if Staging Environment has more than one repository	From the Add dSource Wizard dSource Configuration screen for Staging Push. • Before selecting PDB as the Database Type, select CDB. • Select the correct Staging Environment and repository. • Select PDB for the Database Type. • Now CDB will be shown under Container Database • Fill in the remaining PDB details

Key	Summary	Workaround
DLPX-85076	Instance init file is not copied to $ORACLE_BASE_CONFIG/dbs during staging push dSource linking	Specify the pfile parameter to startup command as `pfile parameter=/<MOUNT_BASE>/<DATABASE_UNIQUE_NAME>/script/<DATABASE_SID>/init<DATABASE_SID>.ora`

Version 6.0.17.0

Key	Summary	Workaround
DLPX-66860	For SSO/SAML, ADFS requires explicit rule to transform emailAddress attribute into nameid.	Create an explicit rule in ADFS that transforms the {{emailAddress}} attribute into a {{nameid}}. The rule type must be "Transform an incoming claim". The incoming claim type must be "Email address" and the outgoing claim type "Name ID". The nameid format must be "Email address".
DLPX-83622	In 6.0.15.0 and 6.0.16.0, Fluentd gems from plugins were installed in the base Fluentd. These left over gems can cause Fluentd to fail post-upgrade.	No workaround.
DLPX-83643	After engine upgrade or after a disable and enable of a vPDB, the NFS version in GUI/API may be incorrect (if it's a vPDB in a vCDB, it was mounted with NFSv3 and now the host supports NFSv4 mounts). This condition corrects itself on the next refresh operation.	Perform a vPDB refresh operation to update the NFS protocol version to 4.
DLPX-83575	The port in the connection string for a vPDB in a Linked CDB may be shown incorrectly when it is registered to a non-default listener.	Use the port in the connection string of the CDB to connect.

Version 6.0.16.0

Key	Summary	Workaround
DLPX-82448	Python2 has been deprecated and is being removed from the platform. During this removal, there are still trace artifacts of legacy Python versions being found on the engine.	No workaround.
DLPX-81559	After upgrading an Oracle VDB from 12c to 19c, the VDB refresh fails with "Failed to mount database instance" due to ORA-01130 or ORA-00201 errors.	Update the VDB parameters (either directly or via a VDB Config Template). For more information, please see KBA6234
DLPX-82169	Provisioning a TDE-enabled vPDB to a CDB with a different patch level than the source CDB will fail.	Patch the dSource to match the target and provision again. If this is not an option, run datapatch on the vPDB, update the timeflow in MDS to a CONFIGURED state, restart the vPDB, and run SnapSync. This second option requires an engineering escalation.

Version 6.0.15.0

Key	Summary	Workaround
DLPX-81610	If a tablespace is encrypted before an incremental snapshot is taken, provisioning from that snapshot can lead to the tablespace containing partially unencrypted data, which will not be accessible.	Take a snapshot with parameter "Force Full Backup" and provision a new vPDB/VDB with this snapshot.

Version 6.0.14.0

Key	Summary	Workaround
DLPX-80489	TDE-enabled vPDB provisions failing with LOCAL_AUTOLOGIN configuration	Use a regular autologin wallet in the target CDB.
DLPX-80822	vPDB provision from a fully encrypted shared undo parent to local undo CDB results in a vPDB with new UNDO TS which is not encrypted	Encrypt the local undo space manually after provision, or with the Configure Clone hook point.

Key	Summary	Workaround
DLPX-81125	MSSQL Export fails when performed on a target with a Microsoft SQL Server instance running with the network service user	If you update the instance owner of the SQL server, you should refresh the environment to reflect the new user in the MDS.
DLPX-81128	Masking: Unable to create/edit a profile set from GUI	Profile set can be add/edited using API

Version 6.0.13.0

Key	Summary	Workaround
DLPX-38908	LogSync should automatically resolve faults for transient issues.	Manually resolve the faults generated when Oracle LogSync encounters transient issues.
DLPX-57078	Job cancel requests during provisioning are not processed until the end of the step.	Find and kill the Oracle `pmon` process for the instance that is being provisioned.
DLPX-64386	LogSync thread hangs when trying to remove temporary RMAN command file from source host toolkit directory.	Remove the `rm` alias for the Delphix OS user.
DLPX-76382	Force disable should succeed despite environmental problems.	No workaround.
DLPX-78412	SCM/Talaria failure reason should be communicated in fault.	No workaround.
DLPX-78986	Prevent DSP connections for disabled cluster nodes.	No workaround.
DLPX-79212	While restoring full backups of file stream type database on open staging DB and taking snapshots, snapshots are consuming full space instead of referring to each snapshot.	No workaround.
DLPX-79355	V2P export throws an error when suspended and resumed at the Opening Database step.	Cancel the V2P export job and start the V2P export operation all over again.
DLPX-79502	SnapSync fails if more than 1000 tempfiles exist in the whole CDB.	No workaround.

Key	Summary	Workaround
DLPX-79596	DB files are getting deleted for staging DB if staging push dSource is forced disabled after performing a restore outside Delphix and then enabled again.	No workaround.
DLPX-79833	You can no longer select performance metric resolution for fluentd configurations.	No workaround.
DLPX-80385	Fluentd does not support deferred upgrades when upgrading to 6.0.13.0, and beyond, when starting from a prior release and using the Historic Splunk Insight solution.	Reboot after upgrade.

Version 6.0.12.0

Key	Summary	Workaround
DLPX-67604	Manually recovering a database after V2P from a snapshot of dSource fails with an error.	No workaround.
DLPX-69775	Updating Oracle credentials with an empty string throws an error when Simplified Connection Management is enabled.	Use the Delphix CLI to unset the user.
DLPX-75209	Network configuration is lost when changing EC2 instance type.	On Nitro-based instance types, use the virtual machine's virtual serial console and login to the sysadmin CLI, then add a DHCP address to the network interface.
DLPX-75878	The JDBC connection string for an Oracle vPDB does not get updated after a listener port change.	A second environment refresh will update the connection string for the vPDB.
DLPX-77231	When source discontinuity on the dSource is followed by resync on the livesource, one or more livesource workers may fail to start. This prevents livesource status from getting updated and the first snapshot from being taken after a resync.	No workaround.
DLPX-78689	Oracle vPDB snapshot job fails after doing reset logs on a linked CDB.	Disable the vPDB and re-enable it to clear out the snapshot job errors.

Key	Summary	Workaround
DLPX-78700	Oracle vPDB source enable jobs for Oracle 21c are taking more than 15 minutes to complete. This issue is only seen during the first Oracle vPDB enable operation on Oracle 21c.	No workaround.

Version 6.0.11.0

Key	Summary	Workaround
DLPX-44544	A SnapSync of an Oracle standby dSource in Real-Time Apply mode sometimes calculates the snapshot's timestamp incorrectly. This can cause ORA-01194 or ORA-01152 errors when provisioning to a timestamp after the snapshot.	To provision from a snapshot of an Oracle standby dSource in Real-Time Apply mode, provision by SCN instead of timestamp.
DLPX-57971	The latest snapshot of a LiveSource may take a long time to show the SCN/timestamp range on its card in the GUI.	No workaround.
DLPX-72123	Detaching or deleting an Oracle dSource may fail on RAC environments due to the failure of deletion of RMAN backups on RAC and the operation needs to be retried with the force option.	If the detach or delete operation on an Oracle dSource fails, retry the command using the force option.
DLPX-75209	Network configuration is lost when changing EC2 instance type.	On Nitro-based instance types, use the virtual machine's virtual serial console and login to the sysadmin CLI, then add a DHCP address to the network interface.
DLPX-77664	Oracle SnapSync fails with "RMAN-06183: datafile or datafile copy (file="" number="">>) larger="" than="" maxsetsize"="" if="" a="" datafile="" resized="" in="" the="" middle="" of="">)>	One of the following three workarounds can be applied: 1. Re-issue SnapSync. 2. Reduce the frequency of datafile resizes, or 3. Ensure datafile resize operations are not being performed while the SnapSync operation is in progress.

Key	Summary	Workaround
DLPX-77869	Unable to drop and recreate descending order (or any other functional) index as part of Oracle Connector masking/reidentification/ tokenization jobs.	Oracle interprets descending order as functional indexes. There is no workaround. This is a known limitation of Drop Indexes for Oracle connectors that will be resolved in a future release.
DLPX-78015	V2P export with absolute data files is failing with an internal error.	No workaround.
DLPX-78263	A SnapSync of an Oracle standby dSource in Real-Time Apply mode sometimes fails with "exception.oracle.snl.linkedsource.current_scn.invalid" if the rate of change in the database is low.	Force a log switch on all primary instances/ nodes and then try another SnapSync.

Version 6.0.10.0

Key	Summary	Workaround
DLPX-77467	Loading the setup app dashboard (as sysadmin) renders a server error popup with instruction to contact Delphix Support. This message can be ignored. However, it is known that this error impairs the ability to configure web proxy, PhoneHome, and SMTP servers via the GUI.	These settings can still be configured via the CLI. This issue will be fixed in the next version release.
DLPX-64082	Oracle provisioning scripts have hard-coded timeouts.	Retry the provisioning operation to see if it succeeds. Otherwise, contact Delphix Support.
DLPX-72369	RAC migration for VDB with a deleted parent may fail with error "Cannot update RAC instances if virtual source 'xxx' has a deleted parent."	No workaround.
DLPX-74896	Race condition during refresh may result in incorrect MDS entry for parent snapshot.	No workaround.
DLPX-75148	DSP throughput tests do not work when from-version is < 6.0.6.0 and target-version is >= 6.0.6.0.	No workaround, however, the user will need to upgrade the source version.

Key	Summary	Workaround
DLPX-75215	Switching AWS instance types can leave the Delphix engine with no network configuration.	On Nitro-based instance types, use the virtual machine's virtual serial console and login to the sysadmin CLI, then add a DHCP address to the network interface.
DLPX-75995	On Windows 2019 Server, with KB4598230 cumulative update, adding or refreshing an environment fails when PowerShell transcription is enabled.	The user has to turn off the transcription if the staging is on Windows 2019 Server with recent updates.
DLPX-77231	When source discontinuity on the dSource is followed by resync on the Oracle LiveSource, one or more LiveSources workers may fail to start. This prevents the LiveSource status from getting updated and the first snapshot from being taken after a resync.	No workaround.

Version 6.0.9.0

Key	Summary	Workaround
DLPX-72186	CDB logfile retention works incorrectly if a PDB has multiple timeflows pointing to the same CDB timeflow.	No workaround.
DLPX-74749	Oracle ENVIRONMENT_REFRESH_AND_DISCOVER job may fail with "The object OraclePDBConfig does not exist on the system".	Manually run an environment refresh after the ENVIRONMENT_REFRESH_AND_DISCOVER job fails.
DLPX-75517	Provisioning an Oracle vPDB to a vCDB fails with "`ORA-00959: tablespace 'TEMP' does not exist`" if there is no temporary tablespace named `TEMP` within the production `PDB$SEED` database.	One of the following two workarounds can be applied: 1. Re-attempt the vPDB provision to a linked CDB. The provision should succeed. 2. Manually create (or rename) the `TEMP` tablespace in the production `PDB$SEED` database, take a new snapshot of the vPDB, and provision from that snapshot to the vCDB.

Key	Summary	Workaround
DLPX-75737	Retention saves unnecessary logs if the bookmark falls exactly on a snapshot end SCN or snapshot end timestamp.	No workaround.
DLPX-75995	On Windows 2019 Server, with KB4598230 cumulative update, adding or refreshing an environment fails when PowerShell transcription is enabled.	The user has to turn off the transcription if the staging is on Windows 2019 Server with recent updates.
DLPX-76388	Entering key pairs directly into hook environment variables (a new feature in 6.0.9.0) results in an internal error.	Use password variables instead, or use a password vault. See Passing Credentials Securely to Hook Operations

Version 6.0.8.0

Key	Summary	Workaround

DLPX-64307	Environment refresh should ignore cluster discovery for VDBs	The following workaround should resolve the issue without Delphix support intervention.
		For RAC databases
		1. Add the instances to the clusterware configuration, ensuring you add all instances configured via Delphix. For example:
		`srvctl add instance -db <Database Name> -instance <Instance Name> -node <Node Name>`
		2. Refresh the environment. At this point, Delphix will add the instances back to the Delphix configuration, and you should be able to perform actions such as stop, start, disable, enable, etc.
		From here, we recommend removing the clusterware configuration.
		3. Stop the VDB via the Delphix GUI or CLI.
		4. Remove the instances from clusterware
		`srvctl remove instance -db <Database Name> -instance <Instance Name>`
		5. Remove the database from clusterware
		`srvctl remove database -db <Database Name>`
		6. Refresh the environment.
		7. Start the VDB via the Delphix GUI or CLI.
		For RACOne databases
		1. Stop the instance.
		`sqlplus "/as sysdba" shutdown immediate`

Key	Summary	Workaround

		2. Ensure the RACOne configuration has the "instance prefix" configuration set.
		`[oracle@mwtestx1 ~]$ srvctl config database -d <vdb name> | grep "Instance name prefix"`
		`Instance name prefix: VDB`
		3. If this is not set, then set it using
		`srvctl modify database -db <vdb name> -instance <instance prefix>`
		4. Set the pfile location to allow clusterware to start the instance
		`srvctl modify database -db <vdb name> -spfile <path to pfile>`
		The pfile is located in
		`<Delphix Mount base>/<VDB Name>/datafile/spfile.ora`
		5. Start the instance using clusterware
		`srvctl start database -d vdb`
		When querying the status of the database, it should now show a running instance. As an example:
		`[oracle@mwtestx1 ~]$ srvctl status database -d VDB Instance VDB_1 is running on node mwtestx1 Online relocation: INACTIVE`
		6. Refresh the environment. At this point, Delphix will add the instances back to the Delphix configuration, and you should be able to perform actions such as stop, start, disable, enable, etc.

Key	Summary	Workaround
		From here, we recommend removing the clusterware configuration. 7. Stop the VDB via the Delphix GUI or CLI. 8. Remove the database from clusterware `srvctl remove database -db <Database Name>` 9. Remove the database from clusterware `srvctl remove database -db <Database Name>` 10. Refresh the environment. 11. Start the VDB via the Delphix GUI or CLI.
DLPX-72186	CDB logfile retention works incorrectly if a PDB has multiple timeflows pointing to the same CDB timeflow	No workaround.
DLPX-73409	Duplicate listener entry gets generated in MDS if Oracle listener is manually started with non-uppercase name	Follow these steps: 1. Restart the listener without a name parameter so it displays in uppercase. 2. Update VDBs to use uppercase LISTENER. 3. Refresh the environment. lowercase listener will be removed and LISTENER updated appropriately.
DLPX-74749	Oracle ENVIRONMENT_REFRESH_AND_DISCOVER job may fail with "The object OraclePDBConfig does not exist on the system"	Manually run an environment refresh after the ENVIRONMENT_REFRESH_AND_DISCOVER job fails.
DLPX-74860	Provisioning a 2nd generation VDB from a dSource with imported RO transportable tablespaces fails with ORA-19654	No workaround.
DLPX-74882	Masking's SFTP client no longer compatible with SolarWinds and Goanyware SFTP servers	No workaround.

Key	Summary	Workaround
DLPX-75517	Provisioning an Oracle vPDB to a vCDB fails with " `ORA-00959: tablespace 'TEMP' does not exist` " if there is no temporary tablespace named `TEMP` within the production `PDB$SEED` database	One of the following two workarounds can be applied: 1. Re-attempt the vPDB provision to a linked CDB. The provision should succeed. 2. Manually create (or rename) the `TEMP` tablespace in the production `PDB$SEED` database, take a new snapshot of the vPDB and provision from that snapshot to the vCDB.

Version 6.0.7.0

Key	Summary	Workaround
DLPX-74457	Cluster discovery for Oracle RAC partially fails on Solaris 10	Switch the default login shell for the Delphix OS user from `/bin/sh` to `/bin/bash`.
DLPX-74749	Oracle ENVIRONMENT_REFRESH_AND_DISCOVER job may fail with "The object OraclePDBConfig does not exist on the system"	Manually run an environment refresh from the Delphix UI if the ENVIRONMENT_REFRESH_AND_DISCOVER job fails with the mentioned error.
DLPX-76718	Time required to display the point-in-time pop-over on timelines for Self-Service Templates with replicated objects can degrade linearly with each replication.	No workaround but reducing replication frequency can reduce the impact.

Version 6.0.6.0

Key	Summary	Workaround
DLPX-73224	When provisioning from a non-multitenant source to a virtual pluggable database (vPDB), the post-plug hook script (dx-post-plug-hook.sh) should exit with the vPDB either closed or open unrestricted. If the vPDB is left open with restricted access subsequent Snapshots of the vPDB will fail with "oracle.ucp.UniversalConnectionPoolException" and " java.sql.SQLException: ORA-01035 ORACLE only available to users with RESTRICTED SESSION privilege" .	To prevent the issue: Ensure that the vPDB is either closed/mounted or open unrestricted when exiting from dx-post-plug-hook.sh. After it has happened: close and reopen the vPDB.
DLPX-72749	Provisioning a TDE-enabled vPDB with system encrypted tablespaces fails with the error "ORA-28374: typed master key not found".	No workaround, contact Delphix Support (TBD)
DLPX-72655	Provisioning an Oracle TDE-enabled vPDB fails intermittently if the dSource is encrypted after linking	Enabling TDE on an existing non-encrypted dSource is not supported. Detach, rename the Delphix dSource name and re-attach it as a new TDE-enabled dSource before re-attempting the provisioning operation.
DLPX-73224	When provisioning from a non-multitenant source to a virtual pluggable database (vPDB), the post-plug hook script (dx-post-plug-hook.sh) should exit with the vPDB either closed or open unrestricted. If the vPDB is left open with restricted access subsequent Snapshots of the vPDB will fail with "oracle.ucp.UniversalConnectionPoolException" and " java.sql.SQLException: ORA-01035 ORACLE only available to users with RESTRICTED SESSION privilege"	To prevent the issue: Ensure that the vPDB is either closed/mounted or open unrestricted when exiting from dx-post-plug-hook.sh. After it has happened: close and reopen the vPDB.
DLPX-73357	Rewinding an Oracle TDE-enabled vPDB to the snapshot before vPDB migration may fail with `exception.oracle.tde.export.keys.failed` if all of the steps of the keystore merge procedure are not followed correctly	Verify that the CDB is restarted after merging the keys of the old target CDB into the new target CDB and re-attempt the rewind operation. If the rewind still fails, provision a new vPDB from the required snapshot instead of the rewind operation.

Key	Summary	Workaround
DLPX-72181	The "Restore Self-Service data container from the bookmark of sibling data container" operation may fail in an Oracle TDE environment with `exception.oracle.tde.export.keys.failed` if the keystore merge procedure is not followed correctly. In this case, the warning "Refresh operation on the TDE-configured container <target container> from the Timeflow of another container <sibling container> requires merging of the TDE keystores." will be displayed in the Delphix UI events log	Verify that the procedure for merging the sibling keystores is followed correctly and re-attempt the self-service operation.
DLPX-73742	Provisioning an Oracle TDE-enabled vPDB may fail with the error "ORA-28367: wallet does not exist" if the TDE wallet for the target linked CDB is stored on ASM storage	Storing the TDE wallet on ASM storage is currently unsupported. Modify `sqlnet.ora` to point to the keystore location outside of the ASM diskgroup and re-attempt the provisioning operation.
DLPX-73788	Provisioning or enabling an Oracle TDE-enabled vPDB fails when $ORACLE_BASE is used in sqlnet.ora	Any environment variable referenced in sqlnet.ora must always be set in the environment for the Delphix OS user. Ensure that the environment variable $ORACLE_BASE is set in the shell initialization file for the Delphix OS user and re-attempt the operation.
DLPX-73789	If WALLET_ROOT initialization parameter is configured on a TDE-enabled dSource PDB, provisioning may fail since the auxiliary CDB instance uses dSource keystore location.	If the provision fails, there are 2 workarounds: 1. Provide permissions to the Delphix OS user to create files in the location identified by WALLET_ROOT directory of the source keystore on the target host. 2. Modify the dSource database to not use WALLET_ROOT to identify the TDE keystore.

Version 6.0.2.0

Key	Summary	Workaround
DLPX-69638	Masking job created on engine 6.0.1.1 or prior is failing after the upgrade to version 6.0.2.0 or later	Masking jobs created in 6.0.1.x using a Hana JDBC driver will need to be updated to grant the following permission {"java.io.FilePermission" "/", "read"} in 6.0.2.0. All drivers created in and after 6.0.2.0 will be granted this permission by default.

Version 6.0.0.0

Key	Summary	Workaround
DLPX-60397	If a mapping algorithm is included in multiple jobs, only one job should be run at a time. If multiple jobs are run at the same time, then the mapping algorithm might contain multiple mappings to the same value or the jobs might deadlock.	Only run one job at a time.
DLPX-60947	Self-Service template with replica VDB is not updated with new Timeflow on incremental replication update	The latest replica VDB data can still be accessed by doing a Self-Service container Refresh, rather than a point-in-time restore from the template.
DLPX-61079	Certificate import validation may incorrectly reject a root CA certificate	Support must manually import the certificate into the truststore.
DLPX-61405	Masking operation should wait for zfs delete queue to drain	Replication may send more data than expected if masking involves dropping large DBF files.
DLPX-64493	V5 API /roles endpoint missing certain items	View and set these privileges through the GUI
DLPX-66155	Failed DSP engine test leads to multiple blocked client.jar processes on target hosts.	Restarting a Delphix Engine during a network throughput test is not recommended as it may lead to a system hang.

Key	Summary	Workaround
DLPX-66860	ADFS does not like NameIDPolicy sent by SSO app	Create an explicit rule in ADFS that transforms the {{emailAddress}} attribute into a {{nameid}}. The rule type must be "Transform an incoming claim". The incoming claim type must be "Email address" and the outgoing claim type "Name ID". The nameid format must be "Email address".
DLPX-66973	Date format is changed after importing the environment	Either (a) use the GUI import feature and then review the imported date formats for correctness or (b) use EngineSync to export/ import jobs, which will not alter the date format.

API changes

This section covers the following topics:

API Changes in Delphix 14.0.0.0

In Delphix 14.0, the new API version is 1.11.25. This section describes all API changes since API version 1.11.24, released with Delphix 13.0. All URL paths are relative to `/resources/json/delphix`.

What's Changed?

API Object	Path	Type	Name	Change
`FluentdPlugin`	/service/fluentd/plugins	Property	`gems`	Property gems have been removed.
`Host`	/host	Property	`nfsAddressList`	Property nfsAddressList has been updated for property format from host to hostOrCIDRAddress.

What's New?

API Object	Path	Type	Name	Description
`Host`	/host	API Type	`nfsEncryptionPort`	Property nfsEncryptionPort have been added.
`MSSqlAdditionalDatabaseParameters`	NA (value type)	API Type	`MSSqlAdditionalDatabaseParameters`	Collection of all the parameters that are not inherited during provisioning of a vdb.
`MSSqlSnapshot`	NA (value type)	API Type	`additionalDataBaseParameters`	Collection of all the parameters that are not inherited during provisioning of a vdb.
`OciObjectStoreTest`	NA (value type)	API Type	`OciObjectStoreTest`	An Oracle Cloud object store connectivity test object.
`OciObjectStore`	NA (value type)	API Type	`OciObjectStore`	An Oracle Cloud object store.

`SourceEnvironment`	/environment	API Type	`nfsEncryptionEnabled`	Flag indicating whether nfs is encrypted or not.
`UnixRuntimeMountInformation`	NA (value type)	API Type	`nfsEncryptionEnabled`	The flag for NFS Encryption.

API Changes in Delphix 13.0.0.0

In Delphix 13.0, the new API version is 1.11.24. This section describes all API changes since API version 1.11.23, which was released with Delphix 12.0. All URL paths are relative to /resources/json/delphix.

What's Changed?

API Object	Path	Type	Name	Change
`OracleRefreshParameters`	NA (value type)	Property	`force`	Property force has been added.
`OracleRollbackParameters`	NA (value type)	Property	`force`	Property force has been added.
`TimeflowSnapshot`	/snapshot	Property	`operations`	Property parameters has been removed from property operations

What's New?

API Object	Path	Type	Name	Description
`ContainerStorageInfo`	NA (value type)	API Type	`ContainerStorageInfo`	Container Storage Information.
`Container`	/database	API Type	`containerStorageInfo`	Container Storage Information.

API Changes in Delphix 12.0.0.0

In Delphix 12.0, the new API version is 1.11.23. This section describes all API changes since API version 1.11.22, which was released with Delphix 11.0. All URL paths are relative to `/resources/json/delphix`.

What's Changed?

API Object	Path	Type	Name	Change
`KeyPairCredential`	NA (value type)	Property	`publicKey`	Property create has been updated from required to optional
`OracleBaseExternalLinkData`	NA (value type)	API Type	`sourcingPolicy`	Property sourcingPolicy has been added
`OracleExportDBInPlaceTransferStrategy`	NA (value type)	API Type	`OracleExportDBInPlaceTransferStrategy`	Property operationsPostV2P has been added
`OracleExportPDBInPlaceTransferStrategy`	NA (value type)	API Type	`OracleExportPDBInPlaceTransferStrategy`	Property operationsPostV2P has been added
`OracleDatabaseContainer`	NA (value type)	Property	`sourcingPolicy`	Ref has been updated from delphix-oracle-sourcing-policy.json to delphix-oracle-base-sourcing-policy.json
`OracleSourcingPolicy`	NA (value type)	Property	`OracleSourcingPolicy`	Ref has been updated from delphix-sourcing-policy.json to delphix-oracle-base-sourcing-policy.json
`OracleVirtualSource`	NA (value type)	API Type	`OracleVirtualSource`	Property allowRefreshRewindPostV2P has been added.
`ValidateNTPParameters`	NA (value type)	Property	`address`	Format has been updated from hostname to host.

What's New?

`OracleBaseSourcingPolicy`	NA (value type)	API Type	`OracleBaseSourcingPolicy`	Oracle Database policies for managing SnapSync and LogSync across sources for a Oracle container.
`OracleStagingSourcingPolicy`	NA (value type)	API Type	`OracleStagingSourcingPolicy`	Database policies for managing LogSync for Oracle Staging push container.
`SuperuserSessionLogDeleteRecord`	NA (value type)	API Type	`SuperuserSessionLogDeleteRecord`	Represents a deletion of a Delphix superuser session log file.
`SuperuserSessionLogDownloadRecord`	NA (value type)	API Type	`SuperuserSessionLogDownloadRecord`	Represents a download of a Delphix superuser session log file.
`SuperuserSession`	/superuser/session	API Type	`SuperuserSession`	Audit logs for superuser sessions.
`OracleExportOperationsPostV2P`	NA (value type)	API Type	`OracleExportOperationsPostV2P`	Describes operations allowed on virtual source post V2P.
`OracleExportTransferStrategy`	NA (value type)	API Type	`OracleExportTransferStrategy`	The transfer strategy for exporting a database whether in-place or TimeFlow point based.
`SystemInfo`	system	API Type	`SystemInfo`	Property maxNativeMemoryGb has been added

API changes in Delphix 11.0.0.0

In Delphix 11.0, the new API version is 1.11.22. This section describes all API changes since API version 1.11.21, which was released with Delphix 10.0. All URL paths are relative to `/resources/json/delphix`.

What's changed?

API object	Path	Type	Name	Change
`AzureAuthentication`	NA (value type)	API Type	`AzureAuthentication`	Properties have been added tenantId and clientId.
`CpuUtilDatapoint`	NA (value type)	API Type	`CpuUtilDatapoint`	Removed property dtrace.
`AzureSecretsAuthentication`	NA (value type)	API Type	`AzureSecretsAuthentication`	Removed properties tenantId and clientId.
`AzureSecretsAuthentication`	NA (value type)	Property	`clientSecret`	Description has been updated for property clientSecret
`CipherSuite`	/service/tls/cipherSuite	Property	`name`	enum property values have been updated.
`Container`	/database	Property	`delete, operations, rollback, sync`	PgSQLDatabaseContainer and MySQLDatabaseContainer have been removed from defaultType.
`DeleteParameters`	NA (value type)	API Type	`DeleteParameters`	The description has been updated.
`DeletionDependency`	NA (value type)	Property	`size`	The description has been updated for property size.
`KeyPairCredential`	NA (value type)	Property	`privateKey`	The format has been updated from password to pemKey.

API object	Path	Type	Name	Change
`KeyPairCredential`	NA (value type)	Property	`publicKey`	Property format has been updated from password to hostKey and property create has been updated from required to optional.
`NamedKeyPairCredential`	NA (value type)	Property	`privateKey`	The property format has been updated from password to pemKey.
`NamedKeyPairCredential`	NA (value type)	Property	`publicKey`	The property format has been updated from password to hostKey.
`OracleExportTimeflowFilesystemLayout`	NA (value type)	API Type	`OracleExportTransferStrategy`	Property rmanFileSectionSizeInGb has been added.
`PemClientCertificate`	NA (value type)	Property	`privateKey`	The property format has been updated from password to pemKey.
`RefreshParameters`	NA (value type)	API Type	`RefreshParameters`	The description has been updated.
`RollbackParameters`	NA (value type)	API Type	`RollbackParameters`	The description has been updated.
`SourceConfig`	/ sourceconfig	Property	`operations`	PgSQLDBClusterConfig has been removed from defaultType.
`SourceDisableParameters`	NA (value type)	API Type	`SourceDisableParameters`	The description has been updated
`SourceEnableParameters`	NA (value type)	API Type	`SourceEnableParameters`	The description has been updated

API object	Path	Type	Name	Change
`SourceIngestionData`	NA (value type)	Property	`containerType`	MYSQL_DB_CONTAINER and PGSQL_DB_CONTAINER have been removed from enum.
`SourceStartParameters`	NA (value type)	API Type	`SourceStartParameters`	The description has been updated
`SourceStopParameters`	NA (value type)	API Type	`SourceStopParameters`	The description has been updated
`SourceTypeAggregateIngestedSize`	NA (value type)	Property	`containerType`	MYSQL_DB_CONTAINER and PGSQL_DB_CONTAINER have been removed from enum.
`Source`	/source	Property	`operations, disable, start, stop`	PgSQLLinkedSource, PgSQLStagingSource, PgSQLVirtualSource, MySQLLinkedSource, MySQLStagingSource, MySQLVirtualSource
`User`	/user	Property	`publicKey`	Property redact-in-logs have been removed.

What's new?

API object	Path	Type	Name	Description
`OracleExportTimeflowFilesystemLayout`	NA (value type)	API Type	`OracleExportTransferStrategy`	Property rmanFileSectionSizeInGb has been added.
`OracleStagingSource`	NA (value type)	API Type	`OracleStagingSource`	Property datafileMountPath and archiveMountPath has been added

API object	Path	Type	Name	Description
`TimeConfig`	/service/time	API Type	`TimeConfig`	Property rootOperations has been added.
`ValidateNTPParameters`	NA (value type)	API Type	`ValidateNTPParameters`	Validate an NTP server by querying it for the time.
`AzureCertificateAuthentication`	NA (value type)	API Type	`AzureCertificateAuthentication`	Client certificate for authenticating to an Azure vault.

API changes in Delphix 10.0.0.0

In Delphix 10.0.0.0, the new API version is 1.11.21. This section describes all API changes since API version 1.11.20, which was released with Delphix 9.0. All URL paths are relative to

`/resources/json/delphix.`

What's changed?

API object	Path	Type	Name	Change
`NfsConfig`	/service/nfs	Property	`mountVersion`	Property mountVersion (enum) has been updated to include NFSv4.
`OracleVirtualCdbSource`	NA (value type)	API Type	`OracleVirtualCdbSource`	Removed property nodeListeners.
`PasswordVaultTestParameters`	NA (value type)	API Type	`PasswordVaultTestParameters`	Removed properties host and port.
`SystemInfo`	/System	Property	`maxHeapSizeGb`	Maximum heap size of the management application.

What's new?

API object	Path	Type	Name	Description
`AzureAuthentication`	NA (value type)	API Type	`AzureAuthentication`	Parameters for authenticating to an Azure vault.
`AzureSecretsAuthentication`	NA (value type)	API Type	`AzureSecretsAuthentication`	Secret credential for authenticating to an Azure vault.

API object	Path	Type	Name	Description
`AzureVaultCredential`	NA (value type)	API Type	`AzureVaultCredential`	The Azure vault based security credential.
`AzureVaultTestParameters`	NA (value type)	API Type	`AzureVaultTestParameters`	Azure password vault test configuration.
`AzureVault`	NA (value type)	API Type	`AzureVault`	Azure password vault configuration.
`HostedVaultTestParameters`	NA (value type)	API Type	`HostedVaultTestParameters`	Hosted password vault test configuration.

API changes in Delphix 9.0.0.0

In Delphix 9.0, the new API version is 1.11.20. This section describes all API changes since API version 1.11.19, that was released with Delphix 8.0.0.0. All URL paths are relative to `/resources/json/delphix.`

What's changed?

API object	Path	Type	Name	Change
`MSSqlVirtualSource`	NA (value type)	Property	`configParams`	Can now be updated via CLI.
`CyberArkPasswordVault`	NA (value type)	API Type	`CyberArkPasswordVault`	Returns a list of password vault objects on the system. Limits CLI visibility for create, update and delete of password vault objects to system users only.
`NamespaceFailoverParameters`	NA (value type)	API Type	`NamespaceFailoverParameters`	Removed property smartFailover.
`PasswordVault`	/service/passwordVault	API Type	`PasswordVault`	Removed property host and port.
`TimeZone`	/timezone	Property	`id`	Property id (enum) has been updated to include America/Rosario,Europe/Belfast.

What's new?

API object	Path	Type	Name	Description
`MSSqlVirtualSource`	N/A (value type)	Property	`configTemplate`	Optional database template to use for provisioning, refresh and enable. If set, configParams will be ignored on the provision or refresh.
`HostedPasswordVault`	N/A (value type)	PropertyAPI Type	`HostedPasswordVault`	Password vaults with host and port.
`UnixHost`	N/A (value type)	Property	`toolkitPath`	Added property format with value unixabsolutepath.
`OracleDBExportParameters`	N/A (value type)	API Type	`OracleDBExportParameters`	Parameters to use as input to export Oracle non-MT databases.
`OracleEnhancedExportParameters`	N/A (value type)	API Type	`OracleEnhancedExportParameters`	The enhanced parameters to use as input to export Oracle databases.
`OraclePDBExportParameters`	N/A (value type)	API Type	`OraclePDBExportParameters`	Parameters to use as input to export Oracle PDB databases.
`OracleExportPDBInPlaceTransferStrategy`	N/A (value type)	API Type	`OracleExportPDBInPlaceTransferStrategy`	Convert a virtual PDB to a physical PDB in-place.

API object	Path	Type	Name	Description
`OracleExportDBInPlaceTransferStrategy`	N/A (value type)	API Type	`OracleExportDBInPlaceTransferStrategy`	Convert a non-MT virtual DB to a physical DB in-place.
`OracleExportASMStorageStrategy`	N/A (value type)	API Type	`OracleExportASMStorageStrategy`	Storage strategy for exporting database files to ASM.
`OracleASMLayout`	N/A (value type)	API Type	`OracleASMLayout`	ASM diskgroups for datafiles, archive logs/ redo logs.

API changes in Delphix 8.0.0.0

In Delphix 8.0.0.0, the new API version is 1.11.19. This section describes all API changes since APIversion 1.11.18, which was released with Delphix 7.0.0.0. All URL paths are relative to `/resources/json/delphix.`

What's changed?

API object	Path	Type	Name	Change
`FluentdConfig`	/service//fluentd/configuration	Property	`performanceMetricsResolution`	Can now set property via CLI.
`NamespaceFailoverParameters`	N/A (value type)	Property	`getFailbackCapability`	Return object has changed from string to object.
`RepavePrepareParameters`	N/A (value type)	API Type	`RepavePrepareParameters`	Removed properties: • quiesceSources • enableSourcesOnFailure • ignoreMaskingJobsInProgress Added property: • unquiesceSourcesOnFailure
`SNMPV2Config`	/service/snmp/v2	Property	`severity`	Added enum value AUDIT.
`Source`	/source	Property	`update`	Restrict CLI visibility to domain users only.
`Source`	/source	Property	`operations`	Restrict CLI visibility to domain users only.
`Source`	/source	Property	`disable`	Restrict CLI visibility to domain users only.

API object	Path	Type	Name	Change
`Source`	/source	Property	`stop`	Restrict CLI visibility to domain users only.
`Source`	/source	Property	`lock`	Restrict CLI visibility to domain users only.
`SsoSuccessfulLoginRecord`	N/A (value type)	API Type	`SsoSuccessfulLoginRecord`	Redact email from logs.
`User`	/user	Property	`isDefault`	Can no longer set property via CLI.

What's new?

API object	Path	Type	Name	Description
`CloudEndpoint`	N/A (value type)	API Type	`CloudEndpoint`	A mapping of a recommend region and endpoint for use with Cloud Engines.
`FailbackCapability`	N/A (value type)	API Type	`FailbackCapability`	A replica namespace's fallback capability.
`OracleLinkedSourceOperations`	N/A (value type)	API Type	`OracleLinkedSourceOperations`	Describes operations which are performed on linked sources at various times.
`SystemInfo`	/system	Property	`rootOperations`	Added getRecommendedCloudStorageEndpoints to get list of recommended endpoints and regions for Cloud Engine setup.

API object	Path	Type	Name	Description
`OracleBaseLinkData`	N/A (value type)	Property	`operations`	Added preLogSync property and new optional fields.
`OracleLinkedSource`	N/A (value type)	Property	`operations`	Added preLogSync property and new optional fields.

API changes in Delphix 7.0.0.0

In Delphix 7.0.0.0, the new API version is 1.11.18. This section describes all API changes since APIversion 1.11.17, which was released with Delphix 6.0.17.0. All URL paths are relative to `/resources/json/delphix.`

What's changed?

API object	Path	Type	Name	Change
`Domain`	N/A (value type)	API Type	`lock`	Restrict CLI visibility to domain users only.
`MSSqlExportParameters`	N/A (value type)	Property	`prodSyncConfigParam`	Property removed.
`NettyVersionInfo`	/netty	Property	`NettyVersionInfo`	API object removed.
`OracleExportAsmParameters`	N/A (value type)	Property	`OracleExportAsmParameters`	API object removed.
`SystemPackage`	/system/package	Property	`SystemPackage`	API object removed.

What's new?

API Object	Path	Type	Name	Description
`Domain`	N/A (value type)	Property	`cliVisibility`	Restrict CLI visibility to system or domain users.
`FluentdAttribute`	N/A (value type)	API Type	`FluentdAttribute`	Fluentd attribute.

API Object	Path	Type	Name	Description
`FluentdConfig`	N/A (value type)	API Type	`FluentdConfig`	Fluentd configuration information.
`FluentdPlugin`	/service/fluentd/plugins	API Type	`FluentdPlugin`	Upload and manage fluentd plugins.
`FluentdRegularAttribute`	N/A (value type)	API Type	`FluentdRegularAttribute`	Fluentd attribute with a plain value.
`FluentdSecretAttribute`	N/A (value type)	API Type	`FluentdSecretAttribute`	Fluentd attribute with a secret value.
`MSSqlSnapshot`	N/A (value type)	Property	`emptySnapshot`	Readonly - True if the staging push dSource snapshot is empty.
`OracleStagingSource`	N/A (value type)	Property	`customEnvVars`	Custom environment variables for Oracle databases.
`OracleBaseStagingLinkData`	N/A (value type)	API Type	`OracleBaseStagingLinkData`	Represents common parameters to link an Oracle database using a staging database.
`OracleStagingPushSyncParameters`	N/A (value type)	API Type	`OracleStagingPushSyncParameters`	The parameters to use as input to sync a staging Oracle database.
`OracleStagingPushSyncStrategy`	N/A (value type)	API Type	`OracleStagingPushSyncStrategy`	Oracle specific parameters for staging push sync strategy.
`OracleSyncFromStagingParameters`	N/A (value type)	API Type	`OracleSyncFromStagingParameters`	The parameters to use as input to sync from a Staging Oracle database.

API Object	Path	Type	Name	Description
`OracleLinkFromStaging`	N/A (value type)	API Type	`OracleLinkFromStaging`	Represents parameters to link a non-pluggable Oracle database using a staging database.
`OraclePDBLinkFromStaging`	N/A (value type)	API Type	`OraclePDBLinkFromStaging`	Represents parameters to link a pluggable Oracle database using a staging database.
`OracleSourceLessSyncStrategy`	N/A (value type)	API Type	`OracleSourceLessSyncStrategy`	Base type for Oracle source less sync strategy and associated parameters.
`S3ObjectStoreRepaveApplyParameters`	N/A (value type)	API Type	`S3ObjectStoreRepaveApplyParameters`	An Amazon Simple Storage Service (Amazon S3) object store.
`BlobObjectStoreRepaveApplyParameters`	N/A (value type)	API Type	`BlobObjectStoreRepaveApplyParameters`	An Azure blob object store.
`BlockStorageRepaveApplyParameters`	N/A (value type)	API Type	`BlockStorageRepaveApplyParameters`	Block Storage.
`RepaveApplyParameters`	N/A (value type)	API Type	`RepaveApplyParameters`	The parameters to use as input to repave apply.
`RepavePrepareParameters`	N/A (value type)	API Type	`RepavePrepareParameters`	The parameters to use as input to repave prepare.

API changes in Delphix 6.0.17.0

In Delphix 6.0.17.0, the new API version is 1.11.17. This section describes all API changes since API version 1.11.16, which was released with Delphix 6.0.16.0. All URL paths are relative to `/resources/json/delphix`.

What's changed?

API object	Path	Type	Name	Change
`AppDataExportParameters`	N/A (value type)	API Type	`AppDataExportParameters`	Object now subclasses GenericExportParameters
`ASEAttachData`	N/A (value type)	Property	`stagingPreScript`	Property removed.
`ASEAttachData`	N/A (value type)	Property	`stagingPostScript`	Property removed.
`ASELinkData`	N/A (value type)	Property	`stagingPreScript`	Property removed.
`ASELinkData`	N/A (value type)	Property	`stagingPostScript`	Property removed.
`ASEStagingSource`	N/A (value type)	Property	`preScript`	Property removed.
`ASEStagingSource`	N/A (value type)	Property	`postScript`	Property removed.
`DbExportParameters`	N/A (value type)	API Type	`DbExportParameters`	Object now subclasses GenericExportParameters
`ExportParameters`	N/A (value type)	Property	`timeflowPointParameters`	Property removed and added to GenericExportParameters

API object	Path	Type	Name	Change
`ExportParameters`	N/A (value type)	Property	`sourceConfig`	Property removed and added to GenericExportParameters
`OracleHostParameters`	N/A (value type)	Property	`tdeKeystoresRootPath`	Restrict format to valid unix path
`OracleVirtualPdbSource`	N/A (value type)	Property	`targetVcdbTdeKeystorePath`	Remove maxLength constraint. Add minLength constraint of 1 character.
`RefreshParameters`	N/A (value type)	API Type	`RefreshParameters`	Object now subclasses ReprovisionParameters
`RollbackParameters`	N/A (value type)	API Type	`RollbackParameters`	Object now subclasses ReprovisionParameters
`TimeflowPointSemantic`	N/A (value type)	Property	`location`	Add OLDEST_SNAPSHOT to acceptable values

What's new?

API object	Path	Type	Name	Description
`Container`	/database	Property	`cdbContainer`	Restrict listing to only include the PDB datasets belonging to the specified CDB dataset reference. This option is mutually exclusive with all the other options.
`Container`	/database	Object Operations	`deprovision`	Deprovisions a container.

API object	Path	Type	Name	Description
`Container`	/database	Object Operations	`reprovision`	Reprovisions a container. This should only be called after a deprovision using the ReprovisionParameters returned from that operation.
`EventsConfig`	/service/events	API Type	`EventsConfig`	Event Message Configuration.
`GenericExportParameters`	N/A (value type)	API Type	`GenericExportParameters`	The parameters to use as input to export requests.
`JobRetentionConfig`	/job/retention	API Type	`JobRetentionConfig`	Configuration for job retention.
`OracleStagingSource`	N/A (value type)	Property	`allowAutoStagingRestartOnHostReboot`	Indicates whether Delphix should automatically restart this staging source when target host reboot is detected.
`ReprovisionParameters`	N/A (value type)	API Type	`ReprovisionParameters`	The input parameters to refresh and rollback requests.
`SourceIngestionData`	N/A (value type)	Property	`containerTypeLabel`	Human readable description of containerType.
`SystemInfo`	/system	Property	`cloudRegion`	The region of the current system if hosted on an applicable cloud service provider.
`SystemInfo`	/system	Property	`smtpConfigured`	Whether SMTP has been configured."

API changes in Delphix 6.0.16.0

In Delphix 6.0.16.0, the new API version is 1.11.16. This section describes all API changes since API version 1.11.15, which was released with Delphix 6.0.15.0. All URL paths are relative to `/resources/json/delphix`.

What's changed?

API object	Path	Type	Name	Change
`NetworkRoute`	/network/route	Property	`gateway`	Parameter has been made optional.
`OracleBaseSourceRuntime`	N/A (value type)	Property	`databaseMode`	UNMOUNTED mode added.

What's new?

API object	Path	Type	Name	Description
`FluentdConfig`	N/A (value type)	Property	`performanceMetricsResolution`	Performance metrics resolution.
`HostNfsChecksParameters`	N/A (value type)	API Type	`HostNfsChecksParameters`	Mechanism to verify user can mount/unmount on a remote host.
`OracleStagingSourceParameters`	N/A (value type)	Property	`physicalStandby`	Whether this staging database will be configured as a physical standby.
`OracleStagingSource`	N/A (value type)	Property	`physicalStandby`	Whether this staging database will be configured as a physical standby.
`PasswordResetConfig`	N/A (value type)	API Type	`PasswordResetConfig`	Password Reset Configuration

API object	Path	Type	Name	Description
`PasswordResetRequestParameters`	N/A (value type)	API Type	`PasswordResetRequestParameters`	Parameters in a password reset request.
`PasswordResetValidationParameters`	N/A (value type)	API Type	`PasswordResetValidationParameters`	Self-service password Reset validation parameters.
`PasswordResetValidationResult`	N/A (value type)	API Type	`PasswordResetValidationResult`	Self-service password Reset validation result.
`PluginManifest`	N/A (value type)	Property	`hasLinkedSourceSize`	Indicates whether linked.source_size() operation has been implemented.
`PluginManifest`	N/A (value type)	Property	`hasVirtualSourceSize`	Indicates whether virtual.source_size() operation has been implemented.
`SourceEnvironment`	/environment	Property	`nfsChecks`	Tests that the environment user can run mount and unmount successfully on the host.

API changes in Delphix 6.0.15.0

In Delphix 6.0.15.0, the new API version is 1.11.15. This section describes all API changes since API version 1.11.14, which was released with Delphix 6.0.14.0. All URL paths are relative to `/resources/json/delphix`.

What's changed?

API object	Path	Type	Name	Change
`DatabaseTemplate`	N/A (value type)	Property	`sourceType`	MySQLVirtualSource and PgSQLVirtualSource removed from acceptable values. OracleLinkedSource added to acceptable values.
`S3ObjectStoreTestResult`	N/A (value type)	API Type	`timestamp`	Object removed and replaced by ObjectStoreTestResult
`ObjectStore`	/storage/objectStorage	Property	`root`	Change root path to /resources/json/delphix/storage/objectStorage
`S3ObjectStore`	N/A (value type)	Standard operations	`read` `update`	Properties removed and added to ObjectStore

`S3ObjectStore`	N/A (value type)	Root operations	`testConnection`
			`cacheHitsReport`
			`clearCacheHits`
				Properties removed and added to ObjectStore

`TimeZone`	N/A (value type)	Property	`sendSocketBuffer`	Add Pacific/Kanton timezone

What's new?

API object	Path	Type	Name	Description
`BlobObjectStoreAccess`	N/A (value type)	API Type	`BlobObjectStoreAccess`	Blob object store access
`BlobObjectStoreAccessKey`	N/A (value type)	API Type	`BlobObjectStoreAccessKey`	Blob object store access key
`BlobObjectStoreAccessManagedIdentities`	N/A (value type)	API Type	`BlobObjectStoreAccessManagedIdentities`	Blob object store access through Managed Identities
`ObjectStoreCacheHitsReport`	N/A (value type)	API Type	`ObjectStoreCacheHitsReport`	A cache hits report for an object store
`ObjectStoreTestResult`	N/A (value type)	API Type	`ObjectStoreTestResult`	An object store connectivity test result
`ObjectStoreTest`	N/A (value type)	API Type	`ObjectStoreTest`	An object store connectivity test object

`ObjectStore`	/storage/objectStorage	Standard operations	`read`
			
			`update`
				Retrieve the specified ObjectStore object
				Update the specified ObjectStore object

API object	Path	Type	Name	Description

`ObjectStore`	/storage/objectStorage	Root operations	`testConnection`
			
			`cacheHitsReport`
			
			`clearCacheHits`
				Test connectivity to an object store
				
				Get a ZettaCache hits-by-size report
				
				Clear the accumulated ZettaCache hits-by-size data

API changes in Delphix 6.0.14.0

In Delphix 6.0.14.0, the new API version is 1.11.14. This section describes all API changes since API version 1.11.13, which was released with Delphix 6.0.13.0. All URL paths are relative to `/resources/json/delphix`.

What's changed?

API object	Path	Type	Name	Change
`CompatibilityCriteria`	N/A (value type)	Property	`processor`	Constraints removed
`JobEvent`	N/A (value type)	Property	`timestamp`	Description changed to "Time the event last occurred."
`MSSqlExportParameters`	N/A (value type)	Property	`filesystemLayout`	Description changed to note this filesystem configuration is specific to anexported MSSQL database.
`NetworkDSPTestParameters`	N/A (value type)	Property	`queueDepth`	Changed from 32 to 64
`NetworkDSPTestParameters`	N/A (value type)	Property	`blockSize`	Changed from 65536 to 1048576
`NetworkDSPTestParameters`	N/A (value type)	Property	`sendSocketBuffer`	Changed from 262144 to 1048576
`NetworkDSPTestParameters`	N/A (value type)	Property	`receiveSocketBuffer`	Changed from 262144 to 1048576
`OracleBaseDBConfig`	N/A (value type)	Property	`user`	Field changed to be nullable
`OracleBaseDBConfig`	N/A (value type)	Property	`credential`	Field changed to be nullable

API object	Path	Type	Name	Change
`OracleVirtualPdbSource`	N/A (value type)	Property	`tdeKeyIdentifier`	minLength changed from 1 to 34
`Schedule`	N/A (value type)	Property	`cutoffTime`	Added to description: “A value of 0 indicates no cutoff”

What's new?

API object	Path	Type	Name	Description
`AppDataLinkedSource`	N/A (value type)	Property	`locked`	Whether the source is protected from deletion and other data-losing actions.
`ASELinkedSource`	N/A (value type)	Property	`locked`	Whether the source is protected from deletion and other data-losing actions.
`ASEStagingSource`	N/A (value type)	Property	`locked`	Whether the source is protected from deletion and other data-losing actions.
`JobEvent`	N/A (value type)	Property	`startTimestamp`	Time the event first occurred.
`MSSqlFileMappingParameters`	N/A (value type)	API Type	`MSSqlFileMappingParameters`	The parameters to use as input to provide File Mapping for MSSQL databases.
`MSSqlLinkedSource`	N/A (value type)	Property	`locked`	Whether the source is protected from deletion and other data-losing actions.
`MSSqlStagingSource`	N/A (value type)	Property	`locked`	Whether the source is protected from deletion and other data-losing actions.

API object	Path	Type	Name	Description
`MSSqlTimeflowFilesystemLayout`	N/A (value type)	API Type	`MSSqlTimeflowFilesystemLayout`	A filesystem layout that matches the filesystem of a Delphix MSSQL TimeFlow.
`OracleLinkedSource`	N/A (value type)	Property	`locked`	Whether the source is protected from deletion and other data-losing actions.
`OracleStagingSource`	N/A (value type)	Property	`locked`	Whether the source is protected from deletion and other data-losing actions.
`Source`	/source	Root operation	`lock`	Protects source from deletion and other data-losing actions. Cannot be undone.

API changes in Delphix 6.0.13.0

In Delphix 6.0.13.0, the new API version is 1.11.13. This section describes all API changes since API version 1.11.12, which was released with Delphix 6.0.12.0. All URL paths are relative to `/resources/json/delphix`.

What's changed?

API object	Path	Type	Name	Change
`MSSqlNoBackupSyncParameters`	API Type	N/A (value type)	`MSSqlNoBackupSyncParameters`	STAGING_PUSH_INGESTION feature flag removed
`MSSqlSourcelessSyncStrategy`	API Type	N/A (value type)	`MSSqlSourcelessSyncStrategy`	STAGING_PUSH_INGESTION feature flag removed
`MSSqlStagingPushSyncStrategy`		N/A (value type)	`MSSqlStagingPushSyncStrategy`	STAGING_PUSH_INGESTION feature flag removed
`NetworkInterface`	API Type	/network/interface	`NetworkInterface`	Removed Domain User access on CLI
`NetworkRoute`	API Type	/network/route	`NetworkRoute`	Removed Domain User access on CLI
`NetworkRoute`	Property	/network/route	`protocol`	Changed from "readonly" to "optional" when creating.
`OAuth2Config`	API Type	/service/oauth2	`OAuth2Config`	Added Domain User access from CLI. Provide System user only access for updates and root operations.

API object	Path	Type	Name	Change
`ObjectStore`	API Type	N/A (value type)	`ObjectStore`	DELPHIX_DATA_BANK feature flag removed. Added update operation.
`OracleDBConfig`	Properties	N/A (value type)	`databaseName` `uniqueName`	Replace property pattern with format - oracleDatabaseName oracleDbUniqueName
`OracleInstance`	Property	N/A (value type)	`instanceName`	Replace property pattern with format - oracleInstanceName
`OraclePDBConfig`	Property	N/A (value type)	`databaseName`	Replace property pattern with format - oraclePDBName
`S3ObjectStoreAccessInstanceProfile`	API Type	N/A (value type)	`S3ObjectStoreAccessInstanceProfile`	DELPHIX_DATA_BANK feature flag removed.
`S3ObjectStoreAccessKey`	API Type	N/A (value type)	`S3ObjectStoreAccessKey`	DELPHIX_DATA_BANK feature flag removed.
`S3ObjectStoreAccess`	API Type	N/A (value type)	`S3ObjectStoreAccess`	DELPHIX_DATA_BANK feature flag removed.
`S3ObjectStoreTestResult`	API Type	N/A (value type)	`S3ObjectStoreTestResult`	DELPHIX_DATA_BANK feature flag removed.

API object	Path	Type	Name	Change
`S3ObjectStoreTest`	API Type	N/A (value type)	`S3ObjectStoreTest`	DELPHIX_DATA_BANK feature flag removed.
`S3ObjectStore`	API Type	/storage/objectStorage	`S3ObjectStore`	DELPHIX_DATA_BANK feature flag removed. "Update" operation added for all the properties.

What's new?

API object	Path	Type	Name	Description
`FluentdAttributeDefinition`	N/A (value type)	API Type	`FluentdAttributeDefinition`	Fluentd attribute definition.
`FluentdAttribute`	N/A (value type)	API Type	`FluentdAttribute`	Fluentd attribute.
`FluentdConfig`	/service/fluentd/configuration	API Type	`FluentdConfig`	Fluentd configuration information.
`FluentdPlugin`	/service/fluentd/plugins	API Type	`FluentdPlugin`	Upload and manage fluentd plugins.
`FluentdRegularAttribute`	N/A (value type)	API Type	`FluentdRegularAttribute`	Fluentd attribute with a plain value.
`FluentdSecretAttribute`	N/A (value type)	API Type	`FluentdSecretAttribute`	Fluentd attribute with a secret value.

API object	Path	Type	Name	Description
`MaskingServiceConfig`	/maskingjob/serviceconfig	Property	`maxJobFetchCount`	Maximum number of jobs to fetch from masking service. Defaults to 500.
`NettyVersionInfo`	/netty	API Type	`NettyVersionInfo`	View netty version and switch between legacy and latest netty version.
`ObjectStore`	N/A (value type)	Property	`configured`	States whether an object store has been configured.
`OracleBaseStagingLinkData`	N/A (value type)	API Type	`OracleBaseStagingLinkData`	Represents common parameters to link an Oracle database using a staging database.
`OracleLinkFromStaging`	N/A (value type)	API Type	`OracleLinkFromStaging`	Represents parameters to link a non-pluggable Oracle database using a staging database.
`OraclePDBLinkFromStaging`	N/A (value type)	API Type	`OraclePDBLinkFromStaging`	Represents parameters to link a pluggable Oracle database using a staging database.
`OracleSourceLessSyncStrategy`	N/A (value type)	API Type	`OracleSourceLessSyncStrategy`	Base type for Oracle source less sync strategy and associated parameters.
`OracleStagingPushSyncParameters`	N/A (value type)	API Type	`OracleStagingPushSyncParameters`	The parameters to use as input to sync a staging Oracle database.

API object	Path	Type	Name	Description
`OracleStagingPushSyncStrategy`	N/A (value type)	API Type	`OracleStagingPushSyncStrategy`	Oracle specific parameters for staging push sync strategy.
`OracleStagingSourceParameters`	N/A (value type)	Property	`instanceName`	The name (sid) of the instance.
`S3ObjectStore`	/storage/objectStorage	Root operations	`cacheHitsReportclearCacheHits`	Get print out of the ZettacCache hit-by-size histogram. Clear the current ZettacCache hit-by-size histogram.
`S3ObjectStoreTestResult`	N/A (value type)	Property	`errorMessage`	Error message from connectivity test.

API changes in Delphix 6.0.12.0

In Delphix 6.0.12.0, the new API version is 1.11.12. This section describes all API changes since API version 1.11.11, which was released with Delphix 6.0.11.0. All URL paths are relative to `/resources/json/delphix`.

What's changed?

API object	Path	Type	Name	Change
`AbstractToolkit`	/toolkit	property	`language`	Removed.
`CipherSuite`	/service/tls/cipherSuite	property	`name`	Removed values for enum - "TLS_RSA_WITH_AES_256_CBC_SHA", "TLS_RSA_WITH_AES_128_CBC_SHA", "TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA", "TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA", "TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA", "TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA", "TLS_DHE_RSA_WITH_AES_256_CBC_SHA", "TLS_DHE_RSA_WITH_AES_128_CBC_SHA", "TLS_DHE_DSS_WITH_AES_256_CBC_SHA", "TLS_DHE_DSS_WITH_AES_128_CBC_SHA".
`HttpConnectorConfig`	/service/httpConnector	property	`tlsVersions`	Removed values for enum - "SSLv3", "TLSv1", "TLSv1_1"
`OAuth2Config`	/service/oauth2	property	`userMatchingFieldType`	Updated description.

API object	Path	Type	Name	Change

`OracleLinkedSource`	N/A (value type)	property	`config`
			`backupLevelEnabled`
			`rmanChannels`
			`filesPerSet`
			`checkLogical`
			`encryptedLinkingEnabled`
			`compressedLinkingEnabled`
			`bandwidthLimit`
			`numberOfConnections`
				Removed.

`OracleSource`	N/A (value type)	property	`config`	Removed.
`StorageDevice`	/storage/device	property	`list` `operations`	Removed CliDisplay parameters - "bootDevice", "allocating". Removed operations - "remove", "removeVerify".

API object	Path	Type	Name	Change
`UnixHost`	N/A (value type)	property	`oracleJdbcKeystorePassword`	Removed.
`User`	/user	property	`apiUser`	Renamed to allowPasswordAuthentication (see in What's New?)

What's new?

API object	Path	Type	Name	Description
`ConfiguredStorageDevice`	N/A (value type)	property	`allocating`	True if the device is being used for allocations. Allocations will be disabled automatically if the device is going to be removed. The allocating property will also be false for devices that are not configured.
`MSSqlNoBackupSyncParameters`	N/A (value type)	API Type	`MSSqlNoBackupSyncParameters`	The parameters to use as input to sync MSSQL databases without a backup.
`MSSqlSourcelessSyncStrategy`	N/A (value type)	API Type	`MSSqlSourcelessSyncStrategy`	MSSQL specific parameters for sourceless sync strategy.
`MSSqlStagingPushSyncStrategy`	N/A (value type)	API Type	`MSSqlStagingPushSyncStrategy`	MSSQL specific parameters for staging push sync strategy.
`OracleHostParameters`	N/A (value type)	API Type	`OracleHostParameters`	Oracle specific host parameters.

API object	Path	Type	Name	Description
`OracleLinkedSource`	N/A (value type)	property	`syncStrategy` `overrideMapsTo`	Parameters used to sync the container. These are persisted and used for every sync operation. Override 'mapsTo' property from source.js for the query parameters defined in 'list' section.
`OracleManagedSource`	N/A (value type)	property	`config`	Reference to the configuration for the source.
`Plugin`	N/A (value type)	property	`language`	Implementation language for workflows in this plugin.
`StorageDeviceRemovalStatus`	/storage/remove	property	`rootOperations`	Add "start" and "verify" root operations to • remove storage devices • verify storage devices can be removed.
`StorageDeviceRemoveParameters`	N/A (value type)	API Type	`StorageDeviceRemoveParameters`	The parameters to use as input when removing devices.
`UnixHost`	N/A (value type)	property	`oracleHostParameters`	The Oracle specific parameters associated with the host.
`User`	/user	property	`allowPasswordAuthentication`	If the Delphix Engine has been configured for SAML SSO, only users with this property set can authenticate with a username and password for API or CLI access.

API changes in Delphix 6.0.11.0

In Delphix 6.0.11.0, the new API version is 1.11.11. This section describes all API changes since API version 1.11.10, which was released with Delphix 6.0.10.0. All URL paths are relative to `/resources/json/delphix`.

What's changed?

API object	Path	Type	Name	Change
`AdvancedSettingsInfo`	/system/advancedSettings	API Type	`AdvancedSettingsInfo`	Hidden behind feature flag SYSTEMTUNABLEINTERFACE
`DelphixManagedBackupIngestionStrategy`	N/A (value type)	API Type	`DelphixManagedBackupIngestionStrategy`	Removed.
`ExternalBackupIngestionStrategy`	N/A (value type)	API Type	`ExternalBackupIngestionStrategy`	Removed.
`IngestionStrategy`	N/A (value type)	API Type	`IngestionStrategy`	Removed.
`MSSqlDBConfig`	N/A (value type)	property	`mssqlUser`	Removed.

API object	Path	Type	Name	Change

`MSSqlLinkedSource`	N/A (value type)	property	`config`
			`sharedBackupLocations`
			`ingestionStrategy`
			`mssqlNetbackupConfig`
			`mssqlCommvaultConfig`
				Removed.

`MSSqlSourceSyncStrategy`	N/A (value type)	property	`config`	Optional while creating.
`NoBackupIngestionStrategy`	N/A (value type)	API Type	`NoBackupIngestionStrategy`	Removed.
`OracleLinkFormExternal`	N/A (value type)	property	`config`	Extends reference changed from OracleBaseLinkData to OracleBaseExternalLinkData. Config property removed.
`OraclePDBLinkFromExternal`	N/A (value type)	property	`config`	Extends reference changed from OracleBaseLinkData to OracleBaseExternalLinkData. Configproperty removed.
`OracleTimeflow`	N/A (value type)	property	`tdeUUID`	No Longer behind feature flag ORACLEMTTDE.

API object	Path	Type	Name	Change

`OracleVirtualPdbSource`	N/A (value type)	property	`parentTdeKeystorePath`
			`parentTdeKeystorePassword`
			`tdeExportedKeyFileSecret`
			`tdeUUID`
				No Longer behind feature flag ORACLEMTTDE.

`Source`	N/A (value type)	property	`config`	nameParent changed from config to container. Configproperty removed from properties and list operation.

`SsoFailedLoginRecord`	N/A (value type)	property	`userAgent`
			`originIp`
			`reason`
				Extends reference changed from TypedObject to FailedLoginRecord.
				All properties removed.

`SsoSuccessfulLoginRecord`	N/A (value type)	property	`userAgent` `originIp`	Extends reference changed from TypedObject to LoginRecord. Properties removed.

What's new?

API object	Path	Type	Name	Description
`AppDataCompatibilityCriteria`	N/A (value type)	API Type	`AppDataCompatibilityCriteria`	The compatibility criteria to use for filtering the list of available AppData repositories.

API object	Path	Type	Name	Description
`AppDataSource`	N/A (value type)	property	`config`	Reference to the configuration for the source.
`ASESource`	N/A (value type)	property	`config`	Reference to the configuration for the source.
`FailedLoginRecord`	N/A (value type)	API Type	`FailedLoginRecord`	Represents an successful login using an external identity provider.
`LoginRecord`	N/A (value type)	API Type	`LoginRecord`	Represents a failed SSO login.
`MSSqlDelphixManagedSyncStrategy`	N/A (value type)	property	`config`	Reference to the configuration for the source.
`MSSqlExternalManagedSourceSyncStrategy`	N/A (value type)	property	`config`	Reference to the configuration for the source.
`MSSqlLinkedSource`	N/A (value type)	property	`syncStrategy` `overrideMapsTo`	Sync strategy for this source. Override mapsTo property for the query parameter defined in list at the root type.
`MSSqlStagingSource`	N/A (value type)	property	`config`	Reference to the configuration for the source.
`MSSqlUser`	N/A (value type)	property	`config`	Reference to the configuration for the source.
`MySQLSource`	N/A (value type)	property	`config`	Reference to the configuration for the source.

API object	Path	Type	Name	Description
`OAuth2Config`	/service/oauth2	API Type	`OAuth2Config`	OAuth2 Configuration.
`OAuth2FailedLoginRecord`	N/A (value type)	API Type	`OAuth2FailedLoginRecord`	Represents a failed OAuth2 login.
`OAuth2SuccessfulLoginRecord`	N/A (value type)	API Type	`OAuth2SuccessfulLoginRecord`	Represents a successful OAuth2 login.
`OracleBaseExternalLinkData`	N/A (value type)	API Type	`OracleBaseExternalLinkData`	Represents common parameters to link all externally managed Oracle databases.
`OracleSourceBasedSyncStrategy`	N/A (value type)	API Type	`OracleSourceBasedSyncStrategy`	Base type for Oracle source based sync strategy and associated parameters.
`OracleSource`	N/A (value type)	property	`config`	Reference to the configuration for the source.
`OracleSyncStrategy`	N/A (value type)	API Type	`OracleSyncStrategy`	Base type for Oracle sync strategy and associated parameters.
`OracleVirtualPdbSource`	N/A (value type)	property	`tdeKeyIdentifier`	ID of the key created by Delphix, as recorded in encryption_keys.key_id.
`OracleVirtualSource`	N/A (value type)	property	`newDBID`	Indicates whether Delphix will generate a new DBID during VDB provision or refresh.

API object	Path	Type	Name	Description
`OsAdminStatus`	/osadmin/engineStatus	API Type	`OsAdminStatus`	Information for the current state of the Delphix Engine.
`PgSQLSource`	N/A (value type)	property	`config`	Reference to the configuration for the source.
`PluginManifest`	N/A (value type)	property	`hasVirtualCleanup`	Indicates whether virtual.cleanup() operation has been implemented.
`ReplicationSourceState`	N/A (value type)	property	`lastKnownTargetVersion`	The Delphix version of the target engine during the last replication using the spec.
`ReplicationTargetState`	N/A (value type)	property	`lastKnownSourceVersion`	The Delphix version of the source engine during the last replication of the namespace.

API changes in Delphix 6.0.10.0

In Delphix 6.0.10.0, the new API version is 1.11.10. This section describes all API changes since API version 1.11.9, which was released with Delphix 6.0.9.0. All URL paths are relative to `/resources/json/delphix`.

What's changed?

API object	Path	Type	Name	Change
`CredentialsEnvVars`	N/A (value type)	API Type	`CredentialsEnvVars`	Extends reference changed from TypedObject to IdentifiableArrayElement.
`CredentialsEnvVars`	N/A (value type)	property	`baseVarName` `credentials`	Optional for an update. Optional for update.
`IscsiTarget`	/storage/iscsi/target	property	`state`	Added enum value ERROR.
`KeyPairCredential`	N/A (value type)	property	`privateKey` `publicKey`	Required for create. Optional for update.
`NetworkRoute`	/network/route	property	`list`	Add enum value protocol in "cliDisplay".
`Operation`	N/A (value type)	API Type	`Operation`	Extends reference changed from TypedObject to IdentifiableArrayElement.
`OracleDatabaseContainer`	N/A (value type)	property	`racMaxInstanceLag`	FeatureFlag 'ORACLERACMAXINSTANCELAG' is removed. default value changed to 3.
`OracleExportParameters`	N/A (value type)	property	`filesystemLayout`	Extends reference changed from TimeFlowFilesystemLayout to OracleExportTimeflowFilesystemLayout

API object	Path	Type	Name	Change
`SystemInitializationParameters`	N/A (value type)	property	`devices`	Optional while creating.
`TimeZone`	/timezone	property	`id`	Removed values for enum - "ACT", "AET", "AGT", "ART", "AST", "SST", "US/Pacific-New", "VST".

What's new?

API object	Path	Type	Name	Description
`BlobObjectStoreAccess`	N/A (value type)	API Type	`BlobObjectStoreAccess`	Blob object store access
`BlobObjectStoreAccessKey`	N/A (value type)	API Type	`BlobObjectStoreAccessKey`	Blob object store access key
`BlobObjectStoreAccessManagedIdentities`	N/A (value type)	API Type	`BlobObjectStoreAccessManagedIdentities`	Blob object store access through Managed Identities
`ObjectStoreCacheHitsReport`	N/A (value type)	API Type	`ObjectStoreCacheHitsReport`	A cache hits report for an object store
`ObjectStoreTestResult`	N/A (value type)	API Type	`ObjectStoreTestResult`	An object store connectivity test result
`ObjectStoreTest`	N/A (value type)	API Type	`ObjectStoreTest`	An object store connectivity test object

API object	Path	Type	Name	Description

`ObjectStore`	/storage/objectStorage	Standard operations	`read`
			
			`update`
				Retrieve the specified ObjectStore object
				Update the specified ObjectStore object

`ObjectStore`	/storage/objectStorage	Root operations	`testConnection`
			
			`cacheHitsReport`
			
			`clearCacheHits`
				Test connectivity to an object store
				Get a ZettaCache hits-by-size report
				Clear the accumulated ZettaCache hits-by-size data

API changes in Delphix 6.0.9.0

In Delphix 6.0.9.0, the new API version is 1.11.9. This section describes all the API changes since API version 1.11.8, which was released with Delphix 6.0.8.0. All URL paths are relative to `/resources/json/delphix`.

What changed

API object	Path	Type	Name	change
`AppDataProvisionParameters`	N/A (value type)	property	`sourceConfig`	Type changed from AppDataSourceConfig to AppDataDirectSourceConfig.
`BaseSupportBundleParameters`	N/A (value type)	property	`bundleType`	Added enum value DOCKER_LOG.
`CipherSuite`	/service/tls/cipherSuite	property	`name`	Added enum values TLS_AES_128_CCM_8_SHA256, TLS_AES_128_CCM_SHA256, TLS_AES_128_GCM_SHA256, TLS_AES_256_GCM_SHA384, and TLS_CHACHA20_POLY1305_SHA256.
`HttpConnectorConfig`	/service/httpConnector	property	`tlsVersions`	Added enum value TLSv1_3.
`Namespace`	/namespace	property	`locked`	No longer behind a feature flag.
`OracleCluster`	/environment	property	`name` `version`	No longer read-only Deleted.
`ReplicationSpec`	N/A (value type)	property	`lockedProfile`	No longer behind a feature flag.

API object	Path	Type	Name	change
`WindowsCluster`	/ environment	property	`name`	No longer read-only.

What is new

API object	Path	Type	Name	Description
`AlertProfile`	/alert/ profile	property	`user`	User to which the alert profile is assigned. Defaults to the logged-in user.
`CredentialsEnvVars`	N/A (value type)	API type	`CredentialsEnvVars`	Credentials to be placed in environment variables for an operation.
`IscsiTargetDiscoverParameters`	N/A (value type)	property	`chapUsername` `chapPassword` `chapUsernameIn` `chapPasswordIn`	CHAP username to be used for iSCSI Discovery. CHAP password to be used for iSCSI Discovery. Target/Mutual CHAP username to be used for iSCSI Discovery (bidirectional authentication). Target/Mutual CHAP password to be used for iSCSI Discovery (bidirectional authentication).
`IscsiTarget`	/storage/ iscsi/ target	property	`chapUsername` `chapPassword` `chapUsernameIn` `chapPasswordIn`	CHAP username to be used for iSCSI Target authentication. CHAP password to be used for iSCSI Target authentication. Target/Mutual CHAP username (bidirectional authentication). Target/Mutual CHAP password (bidirectional authentication).

API object	Path	Type	Name	Description
`SourceOperation`	N/A (value type)	property	`credentialsEnvVarsList`	List of environment variables that will contain credentials for this operation.
`SystemVersion`	/system/version	property	`hotfixVersion`	The hotfix version.

API changes in Delphix 6.0.8.0

In Delphix 6.0.8.0, the new API version is 1.11.8. This section describes all API changes since API version 1.11.7, which was released with Delphix 6.0.7.0. All URL paths are relative to `/resources/json/delphix`.

What changed

API object	Path	Type	Name	Change
`AttachData`	N/A (value type)	property	`config`	Deleted.
`Container`	/database	operation	`undo`	HTTP method changed from GET to POST.
`LinkData`	N/A (value type)	property	`config`	Deleted.
`MSSqlAttachData`	N/A (value type)	property	`backupLocationCredentials` `backupLocationUser/source` `config` `externalFilePath` `ingestionStrategy` `mssqlCommvaultConfig` `mssqlNetbackupConfig` `mssqlUser` `sharedBackupLocations`	Deleted. (See the new syncStrategy property in the table below.)

API object	Path	Type	Name	Change
`MSSqlLinkData`	N/A (value type)	property	`backupLocationCredentials` `backupLocationUser` `config` `externalFilePath` `ingestionStrategy` `mssqlCommvaultConfig` `mssqlNetbackupConfig` `mssqlUser` `sharedBackupLocations`	Deleted. (See the new syncStrategy property in the table below.)
`MSSqlLinkedSource`	/source	property	`backupLocationCredentials` `backupLocationUser` `externalFilePath`	Deleted.
`NamespaceFailoverParameters`	N/A (value type)	property	`smartFailover`	Default changed to true.
`OracleSyncFromExternalParameters`	N/A (value type)	property	`filesForFullBackup`	No longer behind a feature flag.

API object	Path	Type	Name	Change
`SourceIngestionData`	N/A (value type)	property	`containerType`	Removed obsolete value VMWARE_CONTAINER from enum.
`SourceRepository`	/repository	property	`linkingEnabled`	Deleted.
`SourceTypeAggregateIngestedSize`	N/A (value type)	property	`containerType`	Removed obsolete value VMWARE_CONTAINER from enum.
`Source`	/source	property	`status`	Removed value DETACHED from enum.
`VMware*`		API types	`VMware*`	Deleted all types whose names start with VMware (obsolete).

What is new

API object	Path	Type	Name	Description
`AdvancedSettingsInfo`	/system/advancedSettings	API type	`AdvancedSettingsInfo`	Set advanced virtualization, OS, network, and service tunables.
`ASESnapshot`	/snapshot	property	`dbEncryptionKey`	Name of database encryption key present for this snapshot.
`IscsiInitiator`	/storage/iscsi/initiator	API type	`IscsiInitiator`	Endpoint for iSCSI initiator information.
`MSSqlAttachData`	N/A (value type)	property	`syncStrategy`	Configuration that determines what sync strategy the source will use.

API object	Path	Type	Name	Description
`MSSqlDelphixManagedSyncStrategy`	N/A (value type)	API type	`MSSqlDelphixManagedSyncStrategy`	MSSQL specific parameters for delphix managed source based sync strategy.
`MSSqlExternalManagedSourceSyncStrategy`	N/A (value type)	API type	`MSSqlExternalManagedSourceSyncStrategy`	MSSQL specific parameters for externally managed source based sync strategy.
`MSSqlLinkData`	N/A (value type)	property	`syncStrategy`	Configuration that determines what sync strategy the source will use for linking.
`MSSqlSourceSyncStrategy`	N/A (value type)	API type	`MSSqlSourceSyncStrategy`	MSSQL specific parameters for source based sync strategy.
`MSSqlSyncStrategy`	N/A (value type)	API type	`MSSqlSyncStrategy`	Base type for mssql specific parameters for sync strategy.
`Namespace`	/namespace	property	`locked`	Indicates the namespace is locked. (Behind the DATA_VAULT feature flag.)
`OracleDatabaseContainer`	/container	property	`racMaxInstanceLag`	Maximum number of log sequences that a node can lag behind on RAC before considering instance offline. (Behind the ORACLERACMAXINSTANCELAG feature flag.)
`ReplicationSpec`	N/A (value type)	property	`lockedProfile`	Indicates the replication profile is locked. (Behind the DATA_VAULT feature flag.)

API object	Path	Type	Name	Description
`SyncStrategy`	N/A (value type)	API type	`SyncStrategy`	Base type for specific parameters for sync strategy.
`Tunable`	N/A (value type)	API type	`Tunable`	The name of the tunable and its value.
`TunableIdentifier`	N/A (value type)	API type	`TunableIdentifier`	The subsystem and name for a tunable.

API changes in Delphix 6.0.7.0

In Delphix 6.0.7.0, the new API version is 1.11.7. This section describes all API changes since API version 1.11.6, which was released with Delphix 6.0.6.0. All URL paths are relative to `/resources/json/delphix`.

What's changed

API object	Path	Type	Name	Change
`PgSQLDBClusterConfigConnectivity`	N/A (value type)	API type	`PgSQLDBClusterConfigConnectivity`	Deleted.
`OracleBaseDBConfig`	/sourceconfig	property	`nonSysCredentials`	Changed type from PasswordCredential to Credential.
`OracleLinkFromExternal`	N/A (value type)	property	`nonSysCredentials`	Changed type from PasswordCredential to Credential.
`ReplicationSecureList`	N/A (value type)	API type	`ReplicationSecureList`	No longer behind feature flag MDD.
`SNMPV3Manager`	/service/snmp/v3/manager	property	`username`	Introduced a minimum length of 1.
`Source`	/source	property	`status`	Added DETACHED to the list of possible values.

What 's new

API object	Path	Type	Name	Description
`AbstractToolkit`	/toolkit	root operation	`schemaDefinitions`	Get the platform's JSON schema definitions that plugin schemas can reference.
`ASESyncParameters`	N/A (value type)	property	`dropAndRecreateDevices`	If this parameter is set to true, it will drop the older devices and create new devices during manual sync operations instead of trying to remap the devices. This might increasethe space utilization on Delphix Engine.
`NamedKeyPairCredential`	N/A (value type)	API type	NamedKeyPairCredential	Username and key-pair credential.
`NamedPasswordCredential`	N/A (value type)	API type	NamedPasswordCredential	Pair of username and password credential.
`OracleSyncFromExternalParameters`	N/A (value type)	property	filesForFullBackup	List of datafiles to take a full backup of. This would be useful in situations where certain datafiles could not be backed up during previous SnapSync due to corruption or because they went offline.
`SsoConfig`	/service/sso	property	`entityId`	Audience Restriction (SP entity ID, Partner's Entity ID) of this engine as an SSO service provider.

API object	Path	Type	Name	Description
`SsoConfig`	/service/sso	property	`responseSkewTime`	Maximum time difference allowed between a SAML response and the engine's current time, in seconds. If not set, it defaults to 86,400 seconds (one day).
`SsoConfig`	/service/sso	property	`maxAuthenticationAge`	How far in the past to accept authentications to the identity provider, in seconds. If not set, it defaults to 120 seconds.
`SupportBundleConfiguration`	/service/support/bundle	property	`maxActions`	Maximum number of actions not referenced anywhere else to include from the metadata store. The most recent such actions are included.
`SupportBundleConfiguration`	/service/support/bundle	operation	`read`	Retrieve the specified SupportBundleConfiguration object.
`SupportBundleConfiguration`	/service/support/bundle	operation	`update`	Update the specified SupportBundleConfiguration object.

API changes in Delphix 6.0.6.0

In Delphix 6.0.6.0, the new API version is 1.11.6. This section describes all API changes since API version 1.11.5, which was released with Delphix 6.0.5.0. All URL paths are relative to `/resources/json/delphix`.

What's changed

API object	Path	Type	Name	Change
`OracleMultitenantProvisionParameters`	N/A (value object)	object property	`timeflowPointParameters`	The source timeflow point can now reference an Oracle non-multitenant database. In which case the non-multitenant database will be provisioned from the timeflow point to a virtual pluggable database.

API changes in Delphix 6.0.5.0

In Delphix 6.0.5.0, the new API version is 1.11.5. This section describes all API changes since API version 1.11.4, which was released with Delphix 6.0.4.0. All URL paths are relative to `/resources/json/delphix`.

What's changed

API object	Path	Type	Name	Change
`CloudStatus`	/service/cloud	property	`delphixDataServicesComponentStatus`	Made read-only for update.
`CloudStatus`	/service/cloud	property	`delphixDataServicesComponentInfo`	Made read-only for update.
`CloudStatus`	/service/cloud	root operation	`enable`	Added optional payload CloudEnableParameters.
`OracleDBConfig`	/sourceconfig	property	`tdeKeystorePassword`	Impose a maximum length of 256.
`OracleInstance`	N/A (value object)	property	`instanceNumber`	Change type from number to integer.
`OracleSnapshot`	/snapshot	property	`redoLogSizeInBytes`	Change type from number to integer.
`OracleStartParameters`	N/A (value object)	property	`instances`	Change type from number to integer.
`OracleStopParameters`	N/A (value object)	property	`instances`	Change type from number to integer.
`TimeZone`	/timezone	property	`id`	Added enum values America/Nuuk and Asia/Qostanay.

What's new

API object	Path	Type	Name	Description
`CloudEnableParameters`	N/A (value object)	value type	`CloudEnableParameters`	Parameters to the Cloud Enable operation
`CloudStatus`	/service/cloud	property	`proxyMode`	Whether an HTTP proxy must be used to connect to Central Management.
`CloudStatus`	/service/cloud	property	`proxyConfiguration`	Proxy configuration for communication with Delphix Central Management. This property is ignored unless the 'proxyMode' property is set to CLOUD_SPECIFIC_SETTING.
`CloudStatus`	/service/cloud	operation	`update`	Update the specified CloudStatus object. Payload is a CloudStatus.
`HostConfiguration`	N/A (value object)	property	`powerShellVersion`	The PowerShell version installed on the windows target host.
`OracleSTConvertedToPDBAttachData`	N/A (value object)	value type	`OracleSTConvertedToPDBAttachData`	Sub-type of OraclePDBAttachData.
`OracleTimeflow`	/timeflow	property	`tdeUUID`	Unique identifier for TimeFlow-specific TDE objects that reside outside of Delphix storage.
`OracleVirtualPdbSource`	/source	property	`parentTdeKeystorePassword`	The password of the keystore specified in parentTdeKeystorePath.

API object	Path	Type	Name	Description
`OracleVirtualPdbSource`	/source	property	`tdeExportedKeyFileSecret`	Secret to be used while exporting and importing vPDB encryption keys if Transparent Data Encryption is enabled on the vPDB.
`OracleVirtualPdbSource`	/source	property	`tdeUUID`	Unique identifier for PDB-specific TDE objects that reside outside of Delphix storage.
`StaticHostAddress`	/service/ host/ address	new API	`StaticHostAddress`	Static mapping of hostname to IP address.
`WindowsHost`	/host	property	`connectorVersion`	The Windows Connector version that is installed on the provided host.
`WindowsHost`	/host	property	`connectorDotNetFrameworkVersion`	The .NET Framework version used for Windows Connector Service.

API changes in Delphix 6.0.4.0

In Delphix 6.0.4.0, the new API version is 1.11.4. This section describes all API changes since API version 1.11.3, which was released with Delphix 6.0.3.0. All URL paths are relative to `/resources/json/delphix`.

What's changed

API object	Path	Type	Name	Change
`OracleDBConfigConnectivity`	N/A (value object)	object property	`password`	Name changed to credentials Type changed to Credential
`OracleBaseAttachData`	N/A (value object)	object property	`oracleFallBackCredentials`	Type changed to Credential
`OracleBaseDBConfig`	/sourceconfig	object property	`credentials`	Type changed to Credential
`OracleBaseLinkData`	N/A (value object)	object property	`oracleFallBackCredentials`	Type changed to Credential
`OracleDatabaseContainer`	/database	object property	`tdeProvisioningEnabled`	Deleted
`OracleManagedSource`	/source	object property	`mountBase`	Format is now unixrestrictedpath
`OracleVirtualPdbSource`	/source	object property	`parentTdeKeystorePath`	Format is now unixrestrictedpath
`UnixRuntimeMountInformation`	N/A (value object)	object property	`nfsVersionReason`	Added enum value INCOMPLETE_V4_CONFIG

API object	Path	Type	Name	Change
`X500DistinguishedNameComposite`	N/A (value object)	object property	`dname`	Minimum length is now 1

What's new

API object	Path	Type	Name	Description
`X509Certificate`	/service/certificate	new API	`X509Certificate`	X509 Certificate.

API changes in Delphix 6.0.3.0

In Delphix 6.0.3.0, the new API version is 1.11.3. This section describes all API changes since API version 1.11.2, which was released with Delphix 6.0.2.0. All URL paths are relative to `/resources/json/delphix`.

What's changed

API object	Path	Type	Name	Change
ApplyVersionParameters	N/A (value object)	object property	verify	Default changed from true to false.
OracleBaseAttachData	N/A (value object)	object property	dbUser dbCredentials	Name changed to oracleFallbackUser. The database user. Optional if bequeath connections are enabled (to be used in case of bequeath connection failures). Name changed to oracleFallbackCredentials. The password for the database user. Optional if bequeath connections are enabled (to be used in case of bequeath connection failures).
OracleBaseLinkData	N/A (value object)	object property	dbUser dbCredentials	Name changed to oracleFallbackUser. The database user. Optional if bequeath connections are enabled (to be used in case of bequeath connection failures). Name changed to oracleFallbackCredentials. The password for the database user. Optional if bequeath connections are enabled (to be used in case of bequeath connection failures).
Plugin	/toolkit	object property	status	Deleted

What's new

API object	Path	Type	Name	Description
AbstractToolkit	/toolkit	object property	status	The status of the toolkit. ACTIVE indicates toolkit is actively referenced and in use. INACTIVE means toolkit needs to go through a manual upgrade operation before it can be used.
Certificate	/service/tls/caCertificate /service/tls/endEntityCertificate	object property	isCertificateAuthority	Whether this certificate is a Certificate Authority (CA).
ConfiguredStorageDevice	/storage/device	object property	fragmentation	Percent fragmentation for this device.
CyberArkVaultCredential	N/A (value object)	subclass of Credential	CyberArkVaultCredential	CyberArk vault based security credential.
DNSConfig	/service/dns	object property	source	The source of the DNS configuration (STATIC or DHCP).
HashiCorpVaultCredential	N/A (value object)	subclass of Credential	HashiCorpVaultCredential	HashiCorp vault based security credential.
LicenseInfo	/license	object	LicenseInfo	Retrieve all external licenses.
NetworkDSPTestParameters	N/A (value object)	object property	xportScheduler	The transport scheduler to use.
OracleInstall	/repository	object property	oracleBaseConfig oracleBaseHome	The Oracle Base Config directory. The Oracle Base Home directory.

API object	Path	Type	Name	Description
OracleVirtualPdbSource	/source	object property	parentTdeKeystorePath	Path to a copy of the parent's Oracle transparent data encryption keystore on the target host. Required to provision from snapshots containing encrypted database files.
PasswordVault	/service/passwordVault	new API	PasswordVault	Password vault configuration.
Plugin	/toolkit	object property	luaName minimumLuaVersion	The name of the LUA toolkit that this plugin can upgrade. The minimum version (in major.minor format) of a LUA toolkit that this plugin can upgrade.
RunDefaultPowerShellOnSourceOperation	N/A (value object)	subclass of SourceOperation	RunDefaultPowerShellOnSourceOperation	A user-specifiable operation that runs a PowerShell command (using default version) on the target host.
SnapshotCapacityData	/capacity/snapshot	object property list parameter	namespace snapshotTimezone snapshotLatestChange namespace	Reference to the namespace to which this snapshot belongs. Time zone of the source database at the time the snapshot was taken. The timestamp of the latestChangePoint of the associated snapshot. The namespace to list snapshot data for. If null, will limit returned snapshots to the default namespace.

API object	Path	Type	Name	Description
StatisticSlice	/analytics	getData parameter	count	The number of data points to return. Mds data points will be combined in order to to meet this requirement. When count is specified at least two of the other getData parameters must be specified.
SystemInfo	/system	object property	poolFragmentation	Percent fragmentation for the domain0 pool.

Support matrices

This section covers the following topics:

Oracle matrix

Source and Target OS and DBMS Compatibility

- Source and Target must have the same major Oracle version and Edition (Enterprise/Standard). It is recommended to also have the same minor versions and patches.
  - If the minor version (specified with the second digit with Oracle version 18+, or the dated patch level with Oracle version 12.2) or patches differ between the source and target, then a plug-in violation will be received when opening the vPDB on the target, and will need to run datapatch manually or via a hook to resolve it.
- Source and Target operating systems must be compatible. E.g., AIX -> AIX, Solaris SPARC -> Solaris SPARC, x86_64 Linux -> x86_64 Linux.
- In addition, x86_64 Solaris -> x86_64 Linux and x86_64 Linux -> x86_64 Solaris is supported.

Disclaimers and support notes

Delphix Support Policies specifically list **major** and **minor** release coverage. If a minor release is listed as covered, all patch releases under that minor release are certified. Support applies to corresponding versions of Community Enterprise Operating System (CentOS) / Oracle Linux (OL), 64-bit OS support only.

The Oracle version in the support matrix applies to both Oracle DB and Oracle Grid. Delphix supports the same set of features and functionality for supported non-multitenant database versions. Currently, Delphix does not support the Oracle 12c feature of THREADED_EXECUTION being set to TRUE, because this disables OS authentication.

Summary of Delphix features that are unsupported for the Oracle12c and higher Multi-tenant configuration:

- Existing Virtual Container Database (vCDB) as target for provisioning an additional vPDB for Oracle version 12.1.0.1
- Customize VDB settings/initialization parameters. Includes the following:
  - Customize init.ora database parameter during provisioning
  - Config templates
  - Online redo log size
  - Number of RAC VDB instances
  - Online redo log groups
  - Archive log mode
  - Setting new DBID
  - Customize local listeners
- Virtual to Physical (V2P) Support
- Resumable initial SnapSync
- Validated Sync
- Source continuity for dSource upgraded from Oracle 12c non-multitenant to multitenant database
- Cross-platform provisioning (XPP) to the virtual database
- Oracle LiveSources

Delphix supports systems that are part of an Exadata or Exadata Cloud-at-Customer (ExaCC) cluster, provided that the operating system and DBMS version are in the supported list. Oracle Exadata, Exadata Cloud, and Exadata Cloud-at-Customer (ExaCC) offer several features that are reliant on the underlying Exadata storage. Delphix supports creating dSources from Exadata source clusters and provisioning VDBs and vPDBs to Exadata target clusters.

The following Exadata-only features will not be available in child VDBs running in Delphix storage:

- Smart Scan - allows SQL processing to be offloaded to Exadata storage cells. Queries that leverage Smart Scan will execute successfully in VDBs but query performance may be impacted.

- Smart Flash Cache - allows frequently used data blocks to be stored in the flash cache of Exadata storage cells to allow for faster access. Queries against objects that are normally cached in Smart Flash Cache will execute successfully in VDBs but query performance may be impacted. As the virtualization engine uses memory to cache frequently used data blocks, allocating extra memory to the engine may help to mitigate this performance impact.
- Hybrid Columnar Compression (HCC) - compresses data and stores it in a columnar format, allowing Exadata storage cells to deliver faster query performance while also saving on storage. Columns compressed using HCC cannot be used by queries executed in VDBs. Affected columns can be decompressed though care should be taken to evaluate the impact of doing so on VDB refresh times and Delphix storage usage. In Delphix VDBs, queries against partitions or tables using HCC will fail with the following error "*ORA-64307: Exadata Hybrid Columnar Compression is not supported for tablespaces on this storage type*".

Color	Supported?
Y	Yes
N	No
NA	Not Applicable

Oracle database editions

Delphix supports the following Oracle database editions:

- Enterprise
- Standard

Red Hat Enterprise Linux (RHEL)

Support for RHEL 8.x requires installing the `libncurses.5` library onto the host. Please refer to KBA 5622 for the required steps.

Supported OS version	Supported DBMS version					
	11gR2	12cR1	12cR2	18c	19c	21c
RHEL 5.5	Y	NA	NA	NA	NA	NA
RHEL 5.6	Y	Y	NA	NA	NA	NA
RHEL 5.7	Y	Y	NA	NA	NA	NA
RHEL 5.8	Y	Y	NA	NA	NA	NA

RHEL 5.9	Y	Y	NA	NA	NA	NA
RHEL 5.10	Y	Y	NA	NA	NA	NA
RHEL 6.0	Y	Y	NA	NA	NA	NA
RHEL 6.1	Y	Y	NA	NA	NA	NA
RHEL 6.2	Y	Y	NA	NA	NA	NA
RHEL 6.3	Y	Y	NA	NA	NA	NA
RHEL 6.4	Y	Y	Y	Y	NA	NA
RHEL 6.5	Y	Y	Y	Y	NA	NA
RHEL 6.6	Y	Y	Y	Y	NA	NA
RHEL 6.7	Y	Y	Y	Y	NA	NA
RHEL 6.8	Y	Y	Y	Y	NA	NA
RHEL 6.9	Y	Y	Y	Y	NA	NA
RHEL 6.10	Y	Y	Y	Y	NA	NA
RHEL 7.0	Y	Y	Y	Y	NA	NA
RHEL 7.1	Y	Y	Y	Y	NA	NA
RHEL 7.2	Y	Y	Y	Y	NA	NA
RHEL 7.3	Y	Y	Y	Y	NA	NA
RHEL 7.4	Y	Y	Y	Y	Y	NA
RHEL 7.5	Y	Y	Y	Y	Y	NA
RHEL 7.6	Y	Y	Y	Y	Y	NA

RHEL 7.7	Y	Y	Y	Y	Y	NA
RHEL 7.8	Y 6.0.2+	Y 6.0.2+	Y 6.0.2+	Y 6.0.2+	Y 6.0.2+	NA
RHEL 7.9	Y 6.0.4+	Y 6.0.4+	Y 6.0.4+	Y 6.0.5+	Y 6.0.4+	NA
RHEL 8.0	NA	NA	NA	NA	Y 6.0.3+	NA
RHEL 8.1	NA	NA	NA	NA	Y 6.0.3+	NA
RHEL 8.2	NA	NA	NA	NA	Y 6.0.3+	NA
RHEL 8.3	NA	NA	NA	NA	Y 6.0.7+	Y 6.0.11+
RHEL 8.4	NA	NA	NA	NA	Y 6.0.10+	Y 6.0.11+
RHEL 8.5	NA	NA	NA	NA	Y 6.0.14+	Y 6.0.14+
RHEL 8.6	NA	NA	NA	NA	Y 6.0.15+	Y 6.0.15+
RHEL 8.7	NA	NA	NA	NA	Y 7.0.0+	Y 7.0.0+
RHEL 8.8	NA	NA	NA	NA	Y 12.0.0+	Y 12.0.0+

SUSE Linux Enterprise Server (SLES)

Support for SLES 15 requires installing the `libncurses.5` library onto the host. Please reference KBA 5622 for actionable steps.

Supported OS version	**Supported DBMS version**					
	11gR2	**12cR1**	**12cR2**	**18c**	**19c**	**21c**
SLES 11	Y	NA	NA	NA	NA	NA
SLES 11 SP1	Y	NA	NA	NA	NA	NA
SLES 11 SP 2	Y	Y	NA	NA	NA	NA

SLES 11 SP 3	Y	Y	NA	NA	NA	NA
SLES 11 SP 4	Y	Y	NA	NA	NA	NA
SLES 12	Y	Y	NA	NA	NA	NA
SLES 12 SP 1	Y	Y	Y	Y	NA	NA
SLES 12 SP 2	Y	Y	Y	Y	NA	NA
SLES 12 SP 3	Y	Y	Y	Y	Y	NA
SLES 12 SP 4	Y	Y	Y	Y	Y	NA
SLES 12 SP 5	NA	Y 6.0.4+	Y 6.0.4+	N	Y 6.0.4+	NA
SLES 15	N	N	Y	Y	Y	NA
SLES 15 SP 1	N	N	N	N	Y	Y 6.0.11+
SLES 15 SP 2	N	N	N	N	Y 6.0.11+	Y 6.0.11+
SLES 15 SP 3	N	N	N	N	Y 6.0.11+	Y 6.0.11+

Solaris SPARC

Supported OS version	**Supported DBMS version**					
	11gR2	**12cR1**	**12cR2**	**18c**	**19c**	**21c**
Solaris 10 U9[1]	Y	NA	NA	NA	NA	NA
Solaris 10 U10[1]	Y	Y	NA	NA	NA	NA
Solaris 10 U11	Y	Y	N	Y	NA	NA
Solaris 11	Y	Y	NA	NA	NA	NA
Solaris 11 U1	Y	Y	NA	NA	NA	NA

Solaris 11 U2	Y	Y	Y	Y	NA	NA
Solaris 11 U3	N	Y	Y	Y	Y	NA
Solaris 11 U4	N	N	Y	Y	Y	NA

1 - Solaris 10 U9 and U10 require libc.so.1 version 1.22.7 or newer. This version of libc.so.1 can be found in Solaris 10 SPARC kernel patch 144500, or a newer kernel patch.

Solaris x86

Supported OS version	**Supported DBMS version**					
	11gR2	**12cR1**	**12cR2**	**18c**	**19c**	**21c**
Solaris 10 U9[1]	Y	NA	NA	NA	NA	NA
Solaris 10 U10[1]	Y	Y	NA	NA	NA	NA
Solaris 10 U11	Y	Y	NA	Y	NA	NA
Solaris 11	Y	Y	NA	NA	NA	NA
Solaris 11 U1	Y	Y	NA	NA	NA	NA
Solaris 11 U2	Y	Y	Y	Y	NA	NA
Solaris 11 U3	N	Y	Y	Y	Y	NA
Solaris 11 U4	N	N	Y	Y	Y	NA

[1] - Solaris 10 U9 and U10 require libc.so.1 version 1.22.7 or newer. This version of libc.so.1 can be found in Solaris 10 X86 kernel patch 144501, or a newer kernel patch.

Hewlett Packard Unix (HP-UX)

Supported OS version	Supported DBMS version		
	11gR2	12cR1	12cR2
HP-UX 11.31	Y	Y	Y

Advanced Interactive eXecutive (AIX)

Supported OS version	Supported DBMS version					
	11gR2	12cR1	12cR2	18c	19c	21c
AIX 6.1	Y	Y	N	N	NA	NA
AIX 7.1	Y	Y	Y	Y	Y	NA
AIX 7.2	Y	Y	Y	Y	Y	NA

⚠ Required HP-UX patch for Target Servers
PHNE_37851 - resolves a known bug in HP-UX NFS client prior to HP-UX 11.31.

SQL Server matrix

- Delphix Support Policies specifically list Major and Minor release coverage. If a minor release is listed as covered, then all patch releases under that minor release are certified.
- Delphix only supports 64-bit operating systems.

Key:

Color	Supported?
Y	Yes
NA	Not Applicable

Windows server

Supported OS version	Supported DBMS version					
	SQL Server 2012	SQL Server 2014	SQL Server 2016	SQL Server 2017	SQL Server 2019	SQL Server 2022
Windows Server 2012	Y	Y	Y	Y 5.2.3+	NA	NA
Windows Server 2012 R2	Y	Y	Y	Y 5.2.3+	NA	NA
Windows Server 2016	Y 5.1.8+	Y 5.1.8+	Y 5.1.8+	Y	Y 5.3.7+	Y 8.0.0+
Windows Server 2019	NA	NA	Y 5.3.3+	Y 5.3.3+	Y 5.3.7+	Y 8.0.0+
Windows Server 2022	NA	NA	NA	Y 6.0.13+	Y 6.0.13+	Y 8.0.0+

Windows support

Delphix supports both "Windows with Desktop Experience" and Windows Core (Windows without Desktop Experience) server installations.

Windows support

After applying Windows updates or .NET updates for Windows Server 2016 or 2019, servers used to host Delphix VDBs may experience the following symptoms:

- CPU usage on the Windows target host may increase significantly, possibly reaching 100%.
- PowerShell may take up to 10 seconds to launch from the command line.

For more information on how to resolve this issue, see KBA6024

64-bit Windows only

Delphix supports only 64-bit versions of Windows on VDB Target Hosts and Staging Target Hosts. This restriction does not apply to Source Hosts.

Check OS compatibility

The Windows Server OS versions on the Staging Target Hosts and VDB Target Hosts must be compatible. Please refer to the compatibility matrices below.

SQL server failover clusters

There are additional restrictions on supported Windows and SQL Server versions for SQL Server Failover Cluster target environments.

For details, see Adding a SQL Server Failover Cluster Target Environment

Supported SQL versions

SQL server version	Delphix version
SQL Server 2012 (11.0)	Delphix 3.1.2 and beyond
SQL Server 2014 (12.0)	Delphix 4.1.3 and beyond
SQL Server 2016 (13.0)	Delphix 5.1.4 and beyond
SQL Server 2017 (14.0)	Delphix 5.2.3 and beyond
SQL Server 2019 (15.0)	Delphix 5.3.7 and beyond
SQL Server 2022 (16.0)	Delphix 8.0.0 and beyond

Supported SQL server editions

- Standard

- Enterprise
- Developer

SQL server editions

For Staging Target Hosts, you can use SQL Server Standard Edition only if the source databases do not use SQL Server Enterprise Edition specific features, such as partitioned tables.

However, it is important for VDB target hosts to use the same edition of SQL Server software as the source database so that all features available in the source are also available in the VDB.

Supported Windows connector versions

You can always download the current Windows connector version from the Delphix Engine from the following locations:

Delphix Dynamic Data Platform versions before 6.0.5.0:

```
http://<name of your Delphix Engine>/connector/DelphixConnectorInstaller.msi
```

Delphix Dynamic Data Platform versions 6.0.5.0 onwards:

```
http://<name of your Delphix Engine>/connector/DelphixConnectorInstaller.exe
```

The Delphix Connector and Engine are generally backward-compatible.

Delphix connector version 1.18.0 and beyond is compatible with Delphix Engine 6.0.5.0 and later releases.

Delphix connector version 1.17.0 or earlier is compatible with Delphix Engine 6.0.4.0 and earlier releases

Checking the windows connector version

After installing the connector, you can perform the following steps to determine the Windows connector version.

1. Navigate to the Windows Environment.
2. In the search box on the taskbar, type Add or Remove Programs or launch **Start** > **ControlPanel** > **Programs** > **Programs and Features**.
3. On the Add or Remove Programs screen, locate Delphix Connector and then click on the software to check the version installed.

Windows connector matrix

The following table details the supported Windows Connector and JRE versions for the corresponding Delphix Dynamic Data Platform versions.

Delphix dynamic data platform version	Windows connector version	JRE version
4.2	1.2.0	1.7.0_71 (4.2.5.1)

Delphix dynamic data platform version	Windows connector version	JRE version
4.3	1.3.0	1.8.0_40 (4.3.5.1)
5.1.0.0-5.1.4.0	1.4.0	1.8.0_40
5.1.5.0-5.1.9.0	1.5.0	1.8.0_112
5.2.0.0-5.2.2.0	1.9.0	1.8.0_131 (5.2.2.1)
5.2.3.0-5.2.4.0	1.10.0	1.8.0_162
5.2.5.0-5.3.3.0	1.12.0	1.8.0_202
5.3.4.0	1.13.0	1.8.0_202
5.3.5.0	1.14.0	1.8.0_202
5.3.6.0-5.3.7.0	1.15.0	1.8.0_202
5.3.8.0-6.0.0.0	1.16.0	1.8.0_202
6.0.1.0-6.0.4.0	1.17.0	OpenJDK 8u242
6.0.5.0-6.0.8.0	1.18.0	OpenJDK 8u262-b10
6.0.9.0-6.0.10.0	1.20.0	OpenJDK 8u282-b08
6.0.11.0	1.21.0.0	OpenJDK 8u302-b08
6.0.12.0	1.22.0.0	OpenJDK 8u302-b08
6.0.12.1	1.22.0.0	OpenJDK 8u302-b08
6.0.13.0	1.23.0.0	OpenJDK 8u302-b08
6.0.13.1	1.23.0.0	OpenJDK 8u302-b08
6.0.14.0	1.23.0.0	OpenJDK 8u302-b08

Delphix dynamic data platform version	Windows connector version	JRE version
6.0.15.0	1.24.0.0	OpenJDK8u332-b09
6.0.16.0	1.24.0.0	OpenJDK8u332-b09
6.0.17.0	1.27.0.0	OpenJDK 8u345-b01
7.0.0.0	1.27.0.0	OpenJDK 8u345-b01
8.0.0.0	1.27.0.0	OpenJDK 8u345-b01
9.0.0.0	1.28.0.0	OpenJDK 8u362-b09
10.0.0.0	1.29.0.0	OpenJDK 8u362-b09
11.0.0.0	1.30.0.0	OpenJDK 8u362b09
12.0.0.0	1.30.0.0	OpenJDK 8u362b09
13.0.0.0	1.30.0.0	OpenJDK 8u362b09
14.0.0.0	1.31.0.0	OpenJDK 8u362b09

The following table details the .Net connector version compatibility for the corresponding Windows Connector versions.

.Net connector version	Windows connector version
3.5	1.17.0 and earlier
3.5 and 4.0+	1.18.0 and higher

File checksum / hashes

The following table provides the MD5 and SHA256 file hash for recent Delphix versions, in the instance that a security policy requires an exception for software installation.

Delphix dynamic data platform version	Connector version	MD5 Hash	SHA256 Hash
6.0.0.0	1.16.0	84616c4f86c0d871127feb26b9b8bd52	bc2e5e4b26c33d739f7be12f53637979481e1371325e0b5c7645462f6cf5f4f3
6.0.1.0	1.17.0	bb6544dcff3f22d6bf9e20429b1cdb0c	8d914b77d2c5aa0695c3fdbe7d7394e8ae051c091f6f5ee59185cb1d9d157542
6.0.1.1	1.17.0	dd3ea3bedf92a1f8d7e89228315ee315	75ed8aa2cf2689916e0b92de6c2aa2fa760003d62ff07e99978f95dcc9f6abc
6.0.2.0	1.17.0	4770f8c1ad46430a1e4b8764ad687ff5	a98f1c1f90d5053f62d74c1a74bc27b3b48f68a3e067dce04ce9ab2ccf484241
6.0.2.1	1.17.0	97ec3d13db50ad843625b7757c1a2360	7c2ab497ae4ac27b907611e976d7068ed8eaf54e52b836591c082bbe01079512
6.0.3.0	1.17.0	3154c7a5fd7bcb4f0e2c66d328290e6a	efb69b2df884705e488d3405a3958c3bb5457142cbfc71990559bb549c2e8c37
6.0.3.1	1.17.0	2f73be2cf8ceef97cf519c716237462d	7fa7bc269bbc944568231f93ac17ba919b494aa43009967245b75f4abe576a73
6.0.4.0	1.17.0	6a4b7f0a5d180438ff199b2b464506cd	971b88456574e8e11906b5e6850ea46f012490dcb789a5af73f37d71158ffebf
6.0.4.1	1.17.0	ea9a564d0cae1066cd9607824292fbd7	674f9896360a351b314d6aa6e624f9270ef9e3e8d113baddf028581f1ddddec9
6.0.4.2	1.17.0	c5f855c1fa26f5cd3d84d70a94fc0480	07d71acafd559e15782e7145225734adf5e50356d2f8abe5a52bbd90f3a44cbe
6.0.5.0	1.18.0	a3ac768989a6c1dae41adf9452548236	71525c31edcf321d2c0220f4edf8932e249e2cd14b261054af1859574dc9e5eb
6.0.6.0	1.18.0	acb09cbd754e9c25d1afa17b35321cd4	3a51bd3b05148dc3563c5b36d83d44ac725295c23c4b7e9c28f54263058fe2b4
6.0.6.1	1.18.0	32202760adb743a62851f8b698a31bf3	ac16e8a5c37807c2fa775ecc18f9e0387f5bd9d235c9e8854275c99d5bf865a1

Delphix dynamic data platform version	Connector version	MD5 Hash	SHA256 Hash
6.0.7.0	1.18.0	2fed8da4b986b95a7efb9a9a148239ba	2c2a83bad568617bf5dd36b84ad0de22b811f113acac45124ed0effbeb6aef17
6.0.8.0	1.18.0	5bf0b950cdcad8c39db60e1a7406d985	eae5958878a22b4f4f0456210c031ef9c11d064a05ca6cb85535f0e7dec9c0a1
6.0.8.1	1.18.0	a3ac768989a6c1dae41adf9452548236	71525c31edcf321d2c0220f4edf8932e249e2cd14b261054af1859574dc9e5eb
6.0.9.0	1.20.0	127af1d95372f84feb373ab4f7b280b2	b7c7aaceb2b04aebfbd0ba2521dd15520159d54877b0543567b892ee2584c7ce
6.0.10.0	1.20.0	e8a39a57e5a77d106d8040a4537bd8f0	7d4501bf5f1cf22a7039e8f7429d4b42c0d11b1360434380b174830f1155646e
6.0.10.1	1.20.0	baf1223b73e626798dd404bd4d18a3c9	b827d9213e933fce8389322fe25c8d2e1db57ab66709d236b9a8cd5e254bb89d
6.0.11.0	1.21.0.0	412c4c32c7aefb973c21b573dca66d2a	9b689a21249456ddf623042cd2595820531cfbfca0703e713d9795036eef49b7
6.0.12.0	1.22.0.0	dc7a97c46ef9e2d91285dcb0712eaf61	db544774ac3eb3235bf90deef3e93962f3955e86bd9f14309ecb0fba9d41dd7f
6.0.12.1	1.22.0.0	8cf9e955438c6393031c5596ca0420cd	130249658f68eab2a2263deceed6a9de6dd54eead66da68886272f3f40b2f9af
6.0.13.0	1.23.0.0	31cc5d3869e4c7f93d38014e5bd40b27	de1c2fd4c79dcb7391a51774fba1821456ecc031283db5b497145cce383005f2
6.0.13.1	1.23.0.0	3236df25c2e0ab6588bcb54974228b52	c18adb5e0cce7e40e9fa9552b022973ae468c746aca311df2e58c46ef09749ac
6.0.14.0	1.23.0.0	5913ddb3336d571e3ecec1327dcc441c	11fc899f72239061ff5eaf8d65dcd6de467ba1adcaa7f70c1143f04f22cf14d64
6.0.15.0	1.24.0.0	df30e6dc33ef48c839fafdc10726c3a6	e9f0a139d759b366165fc8ac10361a1a9a0518692f8d4a08cae64c889fd9e753

Delphix dynamic data platform version	Connector version	MD5 Hash	SHA256 Hash
6.0.16.0	1.24.0.0	bd3a63d94b7cf6ec2e3c4b7b0d72415f	81a91fb9383d125ca123ca959ea69fe980d01dcfe35474774618b34d2f2a8a84
6.0.17.0	1.27.0.0	ae36f9bdaef53fc967f6504f679ba6c7	f036069d6ec056a0a5ab7e38237d05713f207e84c27dcb65c049bafcc9670214
7.0.0.0	1.27.0.0	3257ed8180aaa53b486afe5b78cf00c5	742cc206acc59942d4773943a6ecaa978be58c95aa3de65b3621b9fa9b791817
8.0.0.0	1.27.0.0	911b83c5ad4fbdb907c11ab3d0ddac18	ea0168077bc104ab59b42b6d76f86000f1f0f3b9ee185be1df1a0f6a5cb1c91f
9.0.0.0	1.28.0.0	64a245d517f05e72ca80453a6d0bf43c	e8f038626c3e1fd9e5bd1c3a2b30df6e0b5e192713005a03ca3a3d0ab329badf
10.0.0.0	1.29.0.0	aa88f148d7fbf2988522841aac780020	b6b5d54be508d9efc2362898a8a8622462fdc42ac15fc3e63d4783edde4c61ee
11.0.0.0	1.30.0.0	47a1c9c5817e64f3f68b17e766cac0a	6266b1f4084c2ea48d922ab5760adad40a730745cd27bfc4bcdcadae7db44bd0
12.0.0.0	1.30.0.0	a5ba7f208b474736ee4e0595155d018e	1622ff824abd56ee8921a75bde4e7012075c31b96aae7fb88ee5cb3d3a3af271
13.0.0.0	1.30.0.0	4e2494ca87ff40f4a17461c3c336dd9d	1206eb0797cdd5356be39d9822fe4944ada5426734b4f826e5bb3b47c79deb4e
14.0.0.0	1.31.0.0	bf6a02d54829d974addf7e54744e4ee7	f185bdca7bd771dda49005d21d03f9cf57dac0f7a909eaab38fab25cbec59dc0

Source environment operating system compatibility

Source environments can be running any supported Windows operating system version. There are no compatibility requirements between the source environment's operating system and that of the target environments. For more information, see Overview of Setting Up SQL Server Environments

Staging and target environment operating system version compatibility

The operating system version on the target environment that will contain the virtual databases must be equal to or higher than that on the staging target.

Source and staging environment SQL server version compatibility

The SQL Server version on the staging environment must be equal to that on the source environment.

> **Provisioning to higher SQL versions when the source is SQL server 2005**
>
> For SQL Server 2005, direct provisioning to higher SQL Server versions is only supported for provisioning to SQL Server 2012 or higher. You can first provision a VDB to SQL Server 2005 and then upgrade it to a higher version by following the steps outlined in Upgrading SQL Server VDBs

Supported SQL server backup software

Delphix currently supports the following software for dSource backups:

- SQL Server native backups - all formats, including but not limited to
  - Striped
  - Compressed
  - Append
  - Overwrite
- Veritas NetBackup
  - Delphix currently supports NetBackup v7.7.3, v8.0, and v8.1.
  - The version of the NetBackup client on the staging environment must be the same as that on the source.
  - If the dSource is backed up with NetBackup, the source and staging environments must have a NetBackup client installed.
    VDB target environments do not need to have NetBackup installed.
  - Logsync (point-in-time provision) is currently not supported.
  - See Linking a dSource from a NetBackup SQL Server Backup for more NetBackup requirements and additional information about NetBackup.
- Commvault
  - Delphix currently supports Commvault v11.
  - The version of Commvault SQL Agent on the staging environment must be the same as that on the source.
  - If the dSource is backed up with Commvault, the source and staging environment must have Commvault SQL Agent installed.
    VDB target environments do not need to have Commvault SQL Agent installed.
  - Commvault SQL Agent on staging environment must be registered to the CommServe server, which is managing source database backups.
  - Logsync (point-in-time provision) is currently not supported.
  - See Linking a dSource from a Commvault SQL Server Backup for more Commvault requirements and additional information about Commvault.
- Quest/NetVault LiteSpeed
  - Delphix currently supports LiteSpeed v5.0.0.0 to v8.9.
  - The LiteSpeed version on the staging environment must be the same or higher than that on the source.
  - If the dSource is backed up with LiteSpeed, the source and the staging environments must also have LiteSpeed installed.
    VDB target environments do not need to have LiteSpeed installed.
  - On source SQL Server Instance, the dSource user must be granted execute privilege on extended stored procedure master.dbo.xp_sqllightspeed_version.
- Red Gate SQL Backup Pro
  - Delphix currently supports SQL Backup Pro v7.3 to v7.7, v8.x, v9.x, and v10.0.
  - The version of SQL Backup Pro on the staging environment must be the same as that on the source.

- On source SQL Server Instance, the dSource database user must be granted execute privilege on extended stored procedure master.dbo.sqbutility.

If the dSource is backed up with SQL Backup Pro, the source and the staging environments must have SQL Backup Pro installed.
VDB target environments do not need to have SQL Backup Pro.

Source and provisioning environment SQL server compatibility matrix

When provisioning a VDB, the SQL Server version on the target should be equal to or higher than that on the source.

	Provisioning target environment					
	SQL Server 2012	**SQL Server 2014**	**SQL Server 2016**	**SQL Server 2017**	**SQL Server 2019**	**SQL Server 2022**
Source Environment						
SQL Server 2012	**X**	**X**	**X**	**X**	**X**	**X**
SQL Server 2014		**X**	**X**	**X**	**X**	**X**
SQL Server 2016			**X**	**X**	**X**	**X**
SQL Server 2017				**X**	**X**	**X**
SQL Server 2019					**X**	**X**
SQL Server 2022						**X**

SAP ASE matrix

This topic describes supported operating systems and database versions for SAP ASE.

Supported DBMS versions

- ASE 15.5
- ASE 15.7
- ASE 16.0

Supported operating systems and database versions for SAP ASE

Source and Target OS and DBMS compatibility
The source and target must be running the same DBMS/Operating System combination, (*although users can run different patch/sp levels*) in order to successfully provision a VDB to the target. For example, if the source is running SAP ASE 16, the target can be running ASE 16SP1. However, for ASE 15.7 the versions of the source and target should both be in the 15.7 SP100 to 15.7 SP140 or 15.7 ESD#1 to 15.7 SP64 ranges. There can be no mixing of versions between these two ranges. If the target is used as a staging server, the same rule applies (prior to Delphix 5.2.5.0, the staging server had to match the source server's version down to the SP level). The Operating System platform must be the same between the source and target, even when the operating system version may differ. For example, if the source is running Red Hat Enterprise Linux 6.2 x86_64 then the target could be running Red Hat Enterprise Linux 6.4 x86_64, but not Solaris 10 SPARC.
ASE itself may or may not allow the **MOUNT** command to work between versions. Be sure to check that the **UNMOUNT** and **MOUNT** commands work between your desired ASE versions.
So for example, if we upgraded the ASE staging instance to ASE 15.7 SP140 but we keep the ASE instance hosting the VDBs at ASE 15.7 SP135, we would want to make sure that we can **UNMOUNT** a database from ASE 15.7 SP140 and then **MOUNT** it on ASE 15.7 SP135:
isql -Usa -Psybase -SASE157SP140 1> UNMOUNT DATABASE testdb to "/tmp/manifest" 2> GO isql -Usa -Psybase -S ASE157SP135 1> MOUNT DATABASE testdb from "/tmp/manifest" 2> go Msg 14580, Level 16, State 1: Server 'ASE157SP135', Line 1: You cannot mount the database(s) because at least one database contains functionality that is available only on the server on which it originated.

In the above example, ASE would not allow the database "testdb" to be mounted on the older release of ASE.

Required HP-UX patch for target servers
PHNE_37851 - resolves a known bug in HP-UX NFS client prior to HP-UX 11.31.

- Delphix Support Policies specifically list Major and Minor release coverage. If a minor release is listed as covered, then all patch releases under that minor release are certified.
- Delphix only supports 64-bit operating systems.

Key:

Color	Supported?
Y	Yes

Color	Supported?
N	No
NA	Not Applicable

Red Hat Enterprise Linux (RHEL)

Support for RHEL 8.x requires installing the `libncurses.5` library onto the host. Please reference KBA 5622 for actionable steps.

Supported OS version	Supported DBMS version		
	15.5	15.7	16
RHEL 5.5	Y	Y	**NA**
RHEL 5.6	Y	Y	NA
RHEL 5.7	Y	Y	NA
RHEL 5.8	Y	Y	NA
RHEL 5.9	Y	Y	NA
RHEL 5.10	Y	Y	NA
RHEL 5.11	Y	Y	NA
RHEL 6.0	N	N	NA
RHEL 6.1	N	N	N
RHEL 6.2	Y	Y	N
RHEL 6.3	Y	Y	N
RHEL 6.4	Y	Y	N
RHEL 6.5	Y	Y	Y

RHEL 6.6	Y	Y	Y
RHEL 6.7	Y	Y	Y
RHEL 6.8	Y	Y	Y
RHEL 6.9	Y	Y	Y
RHEL 6.10	Y	Y	Y
RHEL 7.0	NA	Y	Y
RHEL 7.1	NA	Y	Y
RHEL 7.2	NA	Y	Y
RHEL 7.3	NA	Y	Y
RHEL 7.4	NA	Y	Y
RHEL 7.5	NA	Y	Y
RHEL 7.6	NA	Y	Y
RHEL 7.7	NA	Y	Y
RHEL 7.8	NA	Y 6.0.3+	Y 6.0.3+
RHEL 7.9	NA	Y 6.0.4+	Y 6.0.4+
RHEL 8.0	NA	NA	Y
RHEL 8.1	NA	NA	Y 6.0.4+
RHEL 8.2	NA	NA	Y 6.0.4+
RHEL 8.3	NA	NA	Y 6.0.7+

SUSE Linux Enterprise Server (SLES)

Support for SLES 15 requires installing the `libncurses.5` library onto the host. Please reference KBA 5622 for actionable steps.

Supported OS version	Supported DBMS version		
	15.5	15.7	16
SLES 11	N	N	N
SLES 11 SP1	N	N	N
SLES 11 SP 2	N	N	N
SLES 11 SP 3	N	N	N
SLES 11 SP 4	N	N	N
SLES 12	N	N	N
SLES 12 SP 1	N	N	N
SLES 12 SP 2	N	N	N
SLES 12 SP 3	N	N	N
SLES 12 SP 4	N	N	Y 6.0.4+

Solaris SPARC

Supported OS Version	Supported DBMS version		
	15.5	15.7	16
Solaris 10 U9	Y	Y	N
Solaris 10 U10	Y	Y	N
Solaris 10 U11	Y	Y	N

Solaris 11	N	N	N
Solaris 11 U1	N	N	N
Solaris 11 U2	N	N	N
Solaris 11 U3	N	Y	N
Solaris 11 U4	N	Y 6.0.1.0+	Y 6.0.1.0+

Solaris x86

Supported OS version	**Supported DBMS version**		
	15.5	**15.7**	**16**
Solaris 10 U9[1]	Y	Y	N
Solaris 10 U10[1]	Y	Y	N
Solaris 10 U11	NA	Y	N
Solaris 11	NA	Y	Y
Solaris 11 U1	NA	Y	Y
Solaris 11 U2	NA	Y	Y
Solaris 11 U3	NA	Y	Y
Solaris 11 U4	NA	N	N

Hewlett Packard Unix (HP-UX)

	Supported DBMS version		
Supported OS version	**15.5**	**15.7**	**16**
HP-UX 11.31	N	N	N

Advanced Interactive executive (AIX)

	Supported DBMS version		
Supported OS version	**15.5**	**15.7**	**16**
AIX 6.1	N	Y	N
AIX 7.1	N	Y	Y 6.0.1.0+
AIX 7.2	N	Y	Y

IBM Db2 matrix

Supported Db2 Database Editions

1. Enterprise Server Edition
2. Advanced Enterprise Server Edition
3. Advanced Edition
4. Standard Edition
5. For the supported Db2 versions, Delphix supports the corresponding Db2 Developer edition with Db2 plugin support

- Delphix Support Policies specifically list Major and Minor release coverage. If a minor release is listed as covered, then all patch releases under that minor release are certified.
- Db2 versions below 10.5 i.e (10.1, 9.8) are NOT supported by the plugin.
- Db2 supports only 64-bit OS

Key:

Color	Supported?
Y	Yes
N	No

Red Hat Enterprise Linux (RHEL)

Support for RHEL 8.x requires installing the `libncurses.5` library onto the host. Please reference KBA 5622 for actionable steps.

Db2 plugin supports below mentioned OS and Db2 versions

Supported OS version	Supported DBMS version		
	10.5	11.1	11.5
RHEL 6.9	Y	Y	N
RHEL 7.0	Y	Y	N
RHEL 7.1	Y	Y	N
RHEL 7.2	Y	Y	N

RHEL 7.3	Y	Y	N
RHEL 7.4	Y	Y	N
RHEL 7.5	Y	Y	N
RHEL 7.6	Y	Y	N
RHEL 7.7	N	Y	N
RHEL 7.8	N	Y	Y 6.0.5+
RHEL 7.9	N	Y	Y
RHEL 8.0	N	Y	Y
RHEL 8.1	N	Y	Y
RHEL 8.2	N	N	Y
RHEL 8.3	N	N	Y
RHEL 8.4	N	N	Y
RHEL 8.5	N	N	Y
RHEL 8.6	N	N	Y 6.0.16+
RHEL 8.7	N	N	Y
RHEL 8.8	N	N	Y 12.0.0.0

Advanced Interactive eXecutive (AIX)

Db2 plugin supports below mentioned OS and Db2 versions

	Supported DBMS Version		
Supported OS Version	**10.5**	**11.1**	**11.5**
AIX 7.1	Y	Y	Y

AIX 7.2	Y	Y	Y

Plugin/Delphix Engine compatibility

Plugins should be installed on compatible Delphix Engines per the table below:

> The plugin versions starting 2.1.0 till 2.7.3 are on extended support. Please contact Delphix Support for any queries.

Delphix Engine	3.0.0	3.0.1	3.0.2	3.1.0	3.1.1	4.0.0	4.0.1	4.1.0	4.1.1	4.1.2	4.1.3	4.2.0	4.2.1	4.3.0	4.3.1	4.3.2	4.4.0	4.4.1	4.4.2	4.5.0	4.5.1	4.6.0	4.6.1
6.0.0.x	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6.0.1.x	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6.0.1.x	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6.0.3.x	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6.0.4.x	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N

6.0.5.x	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6.0.6.x	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6.0.7.x	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6.0.8.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6.0.9.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6.0.10.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6.0.11.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6.0.12.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N
6.0.13.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N

6.0.14.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N
6.0.15.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N
6.0.16.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
6.0.17.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
7.0.0.x	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
8.0.0.x	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
9.0.0.x	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
10.0.0.x	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
11.0.0.x	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

12.0.0.x	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
13.0.0.x	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
14.0.0.x	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

⚠ In the case of HADR, all source, staging, and target should be on the same fix pack version.

Plugin upgrade path

Use the table below to determine the most efficient upgrade path from your current version to the latest version of the Db2 Plugin.

Your Version	Recommended upgrade path to Db2 4.6.1
3.0.0 3.0.1 3.0.2	Upgrade to 3.1.0.
3.1.0 3.1.1	Upgrade to 4.0.0 or 4.0.1.
4.0.0 4.0.1	Upgrade directly to 4.1.0, 4.1.1, 4.1.2, 4.1.3, 4.2.0, 4.2.1, or 4.3.0, 4.3.1, 4.3.2, 4.4.0, 4.4.1, 4.4.2,4.5.0, 4.5.1, 4.6.0,4.6.1.
4.1.0 4.1.1 4.1.2 4.1.3	Upgrade directly to 4.2.0, 4.2.1, or 4.3.0, 4.3.1, 4.3.2, 4.4.0, 4.4.1, 4.4.2,4.5.0,4.5.1,4.6.0,4.6.1.
4.2.0 4.2.1	Upgrade directly to 4.3.0, or 4.3.1, 4.3.2, 4.4.0, 4.4.1, 4.4.2,4.5.0,4.5.1, 4.6.0,4.6.1.

Your Version	Recommended upgrade path to Db2 4.6.1
4.3.0 4.3.1 4.3.2	Upgrade directly to 4.4.0, 4.4.1,4.4.2,4.5.0,4.5.1, 4.6.0,4.6.1.
4.4.0 4.4.1 4.4.2	Upgrade directly to 4.5.0, 4.5.1, 4.6.0,4.6.1.
4.5.0 4.5.1	Upgrade directly to 4.5.0, 4.5.1, 4.6.0,4.6.1.
4.6.0	Upgrade directly to 4.6.1.

Unsupported DB2 features

- V2P
- Migrating a DB2 Database

PostgreSQL matrix

Delphix Support Policies specifically list Major and Minor release coverage. If a minor release is listed as covered, then all patch releases under that minor release are certified.

Key:

Color	Supported?
Y	Yes
N	No

RHEL/CentOS/SUSE/Ubuntu

Supported OS Version	Supported DBMS Version (Open-source & EDB)								
	9.4.x (Open-source & EDB)	9.5.x (Open-source & EDB)	9.6.x (Open-source & EDB)	10.x (Open-source & EDB)	11.x (Open-source & EDB)	12.x (Open-source & EDB)	13.x (Open-source & EDB)	14.x (Open-source & EDB)	15.x (Open-source & EDB)
RHEL/ CentOS 7.3	Y	Y	Y	Y	Y	Y	N	N	N
RHEL/ CentOS 7.4	Y	Y	Y	Y	Y	Y	N	N	N
RHEL/ CentOS 7.5	Y	Y	Y	Y	Y	Y	N	N	N
RHEL/ CentOS 7.6	Y	Y	Y	Y	Y	Y	N	N	N
RHEL/ CentOS 7.7	Y	Y	Y	Y	Y	Y	N	N	N
RHEL/ CentOS 7.8	N	N	Y	Y	Y	Y	N	N	N
RHEL 7.9	N	N	Y	Y	Y	Y	Y	Y	N

Centos 7.9	N	N	N	N	N	Y	Y	Y	N
RHEL 8.0	N	N	N	N	N	Y	Y	Y	Y
RHEL 8.1	N	N	N	N	N	Y	Y	Y	Y
RHEL 8.2	N	N	N	N	N	Y	Y	Y	Y
RHEL 8.3	N	N	N	N	N	Y	Y	Y	Y
RHEL 8.4	N	N	N	N	N	Y	Y	Y	Y
RHEL 8.5	N	N	N	N	N	Y	Y	Y	Y
RHEL 8.6	N	N	N	N	N	Y	Y	Y	Y
RHEL 8.7	N	N	N	N	N	Y	Y	Y	Y
RHEL 8.8	N	N	N	N	N	Y	Y	Y	Y
RHEL 8.9	N	N	N	N	N	Y	Y	Y	Y
RHEL 9.0	N	N	N	N	N	N	N	N	Y
SUSE 11	Y	N	N	N	N	N	N	N	N
SUSE 12	Y	Y	Y	Y	Y	N	N	N	N
SUSE 15	N	N	N	N	N	N	Y	N	N
Ubuntu 16.04	N	N	N	Y	N	N	N	N	N
SUSE 15SP3	N	N	N	N	N	N	Y	N	N

where **x** represents the minor version.

- All the available PostgreSQL Database minor versions will be supported for the above-mentioned major versions.
- SUSE 15SP3 is supported on Power 8 hardware.

- Make sure to install libncurses.5 or libncurses.6 in the host, for supporting RHEL and SUSE. You can refer KBA 5622 for actionable steps.

Cloud vendors

Supported cloud vendors	PostgreSQL 11	PostgreSQL 12	PostgreSQL 13	PostgreSQL 14
Amazon RDS PostgreSQL	Y	Y	Y	Y
Amazon Aurora PostgreSQL	Y	Y	Y	Y
Azure Flexible Server	Y	Y	Y	Y

Only a single DB ingestion feature is supported for the Cloud Vendors (Amazon RDS/Aurora and Azure Single/Flexible Server) PostgreSQL through the pg_dump/pg_restore method.

Plugin/Delphix Engine compatibility

Plugins should be installed on compatible Delphix Engines per the table below:

The plugin versions from 1.0.1 till 1.3.2 are currently in extended support. Please reach out to Delphix customer support for any queries.

Currently, PostgreSQL Plugin 1.4.1 is not supported on SUSE 12.x.

Delphix Engine Version	Plugin Version															
	1.4.0	1.4.1	1.4.2	1.4.3	1.4.4	1.5.0	1.5.1	2.0.0	2.1.0	2.1.1	3.0.0	3.1.0	3.2.0	4.0.0	4.1.0	4.1.1
6.0.0.x	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.1.x	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.2.x	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.3.x	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N

6.0.4.x	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.5.x	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.6.x	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.7.x	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.8.x	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.9.x	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.10.x	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.11.x	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.12.x	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.13.x	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.14.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N
6.0.15.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N
6.0.16.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
6.0.17.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
7.0.0.x	N	N	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y
8.0.0.x	N	N	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y
9.0.0.x	N	N	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y

10.0.0.x	N	N	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y
11.0.0.x	N	N	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y
12.0.0.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
13.0.0.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
14.0.0.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Plugin upgrade path

Path A - New plugin installation for PostgreSQL customers:

- Upload 4.0.0 Plugin for PostgreSQL 9.4.x, 9.5.x, 9.6.x,10.x, 11.x, 12.x, 13.x,14.x and 15.x customers.

Path B - Existing PostgreSQL customers with different plugin versions:

Your version	**Recommended upgrade path to 4.1.1**
1.0.1 1.0.2	Upgrade to 1.1.0 and follow the upgrade path below.
1.1.0	Upgrade to 1.2.0 and follow the upgrade path below.
1.2.0	Upgrade to 1.3.0 and follow the upgrade path below.
1.3.0	Upgrade to 1.4.1, 1.4.2, 1.4.3, or 1.4.4 and follow the path below.
1.4.1 1.4.2 1.4.3	Upgrade to 1.5.0 or 1.5.1and follow the path below.
1.5.0 1.5.1	Upgrade to 2.0.0 and follow the upgrade path below.

Your version	Recommended upgrade path to 4.1.1
2.0.0 2.1.0	Upgrade to 3.0.0 and follow the upgrade path below.
3.0.0 3.1.0 3.2.0	Upgrade to 4.0.0 and follow the upgrade path below.
4.0.0 4.1.0	Upgrade directly to the latest version 4.1.1.

For PostgreSQL customers with plugin version 1.0.1 or 1.0.2:

Plugin upgrade path: 1.0.1 → 1.1.0 → 1.2.0 → 1.3.0 → 1.4.1/1.4.2/1.4.3/1.4.4 → 1.5.0/1.5.1 → 2.0.0/2.1.0 → 3.0.0 → 3.1.0 → 3.2.0 → 4.0.0→ 4.1.0→ 4.1.1.

Plugin upgrade path 1.0.2 → 1.1.0 → 1.2.0 → 1.3.0 → 1.4.1/1.4.2/1.4.3/1.4.4 → 1.5.0/1.5.1 → 2.0.0/2.1.0 → 3.0.0 → 3.1.0 → 3.2.0 → 4.0.0→ 4.1.0→ 4.1.1.

For PostgreSQL customers with plugin version 1.1.0

Plugin upgrade path: 1.1.0 → 1.2.0 → 1.3.0 → 1.4.1/1.4.2/1.4.3/1.4.4 → 1.5.0/1.5.1 → 2.0.0/2.1.0 → 3.0.0 → 3.1.0 → 3.2.0 → 4.0.0→ 4.1.0→ 4.1.1.

For PostgreSQL customer with plugin version 1.2.0

Plugin upgrade path: 1.2.0 → 1.3.0 → 1.4.1/1.4.2/1.4.3/1.4.4 → 1.5.0/1.5.1 → 2.0.0/2.1.0 → 3.0.0 → 3.1.0 → 3.2.0 → 4.0.0→ 4.1.0→ 4.1.1.

For PostgreSQL customer with plugin version 2.0.0

Plugin upgrade path: 2.0.0/2.1.0 → 3.0.0 → 3.1.0 → 3.2.0 → 4.0.0→ 4.1.0→ 4.1.1.

Upgrading existing objects

In order to upgrade the existing objects with 1.1.0 (i.e 1.0.1 → 1.1.0 or 1.0.2 → 1.1.0), just after the upgrade, it is mandatory to update below two parameters of dSource using the "custom Configuration" option on the UI :

- Source IP Address
- Source Port

A critical fault will show up on dSource just after the upgrade. It is because the engine is not able to find the source config from which dSource was created. As we have moved to manual discovery in this release, we need to mark this fault as resolved on the status tab of dSource.

Unsupported PostgreSQL versions and features

- PostgreSQL versions below 9.4 i.e (9.2, 9.3) are NOT supported by the plugin.
- It is not possible to access the staging server with PostgreSQL 9.4 and PostgreSQL 9.5 versions.
  The following points describe why staging server is not accessible with PostgreSQL 9.4 and PostgreSQL 9.5 versions.
  a. On staging server, PostgreSQL plugin setups warm standby till version 9.x, which indicates that you cannot access PostgreSQL database on staging instance with PostgreSQL 9.4 and PostgreSQL 9.5

versions or even dSource for READ-ONLY workload. However, in later versions it is possible to trigger READ-ONLY query on dsource.

b. In PostgreSQL version 9.x, the plugin sets *wal_level* parameter on staging server PostgreSQL instance to *archive*. However for PostgreSQL version 10 and later, PostgreSQL plugin sets *wal_level* parameter on staging server PostgreSQL instance to *hot_standby* or *replica*.

1. **Warm Standby/Log Shipping** is a HA solution which 'replicates' a database cluster to an archive or a warm (can be brought up quickly, but not available for querying) standby server. Overhead is very low and it's easy to set up. This is a simple and appropriate solution if all you care about is continuous backup and short failover times.
2. **Hot Standby/Streaming Replication** provides asynchronous binary replication to one or more standbys. Standbys may also become hot standbys meaning they can be queried as a read-only database. This is the fastest type of replication available as WAL data is sent immediately rather than waiting for a whole segment to be produced and shipped. In hot_standby level, the same information is logged as with archive, plus information needed to reconstruct the status of running transactions from the WAL. To enable read-only queries on a standby server, wal_level must be set to 'hot_standby' or higher on the primary, and hot_standby must be enabled in the standby. Dsource created in PostgreSQL 9.4 & 9.5 version is a warm standby so it is not accessible. *wal_level* determines how much information is written to the WAL. The default value is minimal, which writes only the information needed to recover from a crash or immediate shutdown. *archive* adds logging required for WAL archiving; hot_standby further adds information required to run read-only queries on a standby server; and, finally logical adds information necessary to support logical decoding. Each level includes the information logged at all lower levels. This parameter can only be set at server/instance level.

- There are two features that are not supported by the plugin:
  a. Unlogged Tables: All the tables which are not logged (unlogged) will not be supported by the Plugin.
  b. Point In Time recovery: Currently, the Plugin doesn't support Point in Time recovery of the PostgreSQL database.

SAP HANA matrix

Plugin installation

Plugins to support SAP HANA virtualization are not automatically included in the Virtualization Engine deployment and are installed by Delphix Professional Services or Technical Support. A support case can be opened requesting installation by following this link.

Migration and compatibility

This section describes the SAP HANA (DBMS) versions that are supported by the SAP HANA 2.0 Plugin, as well as the compatible operating systems (OS), for use on the source and target environments.

Delphix Support Policies specifically list Major and Minor release coverage. If a minor release is listed as covered, then all patch releases under that minor release are certified.

Supported DBMS versions

SAP HANA version (Standard and enterprise editions)	**Support package stack (SPS)**
2.0	04
2.0	05

A target may have a higher SPS release than a source; the reverse is not supported.

Key:

Color	**Supported?**
Y	Yes
N	No

Supported operating system

Supported OS version	**Supported DBMS version**	
	SAP HANA 2.0 SPS 04 (Standard and enterprise editions)	**SAP HANA 2.0 SPS 05 (Standard and enterprise editions)**

RHEL 7.9	Y	Y
RHEL 8.1	Y	Y
RHEL 7.3/7.4/7.5	Y	N
RHEL 7.6	Y	Y
SUSE 12 SP 03	Y	N
SUSE 15 SP 02	N	Y

Plugin/Delphix Engine compatibility

The plugin should be installed on compatible Delphix Engines per the table below:

Delphix Engine	**Lua-based plugin**						**vSDK-based Plugin**								
	4.1.1	**4.1.2**	**4.2.0**	**4.3.0**	**4.3.1**	**4.3.2**	**6.0.0**	**6.0.1**	**6.1.0**	**6.1.1**	**6.2.0**	**6.2.1**	**6.3.0**	**6.3.1**	**6.3.2**
6.0.0.x	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.1.x	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.1.x	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.3.x	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.4.x	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.5.x	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.6.x	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.7.x	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N
6.0.8.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N
6.0.9.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N

6.0.10.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N
6.0.11.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N
6.0.12.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N
6.0.13.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N
6.0.14.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N
6.0.15.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N
6.0.16.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N
6.0.17.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N
7.0.0.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N
8.0.0.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N
9.0.0.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N
10.0.0.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
11.0.0.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
12.0.0.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
13.0.0.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

14.0.0.x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Plugin upgrade path

> ⚠ Upgrade from SAP HANA Direct (Lua-based) plugin to SAP HANA Staged (VSDK-based) plugin is not supported.

SAP HANA (Lua-based) plugin upgrade path

Your version	Recommended upgrade path to 4.3.2
3.4.5	Upgrade to version 3.5.1 and then upgrade to 4.0.0 and then to 4.1.1 and then to 4.1.2 and then to 4.2.0
3.5.0	Upgrade to version 3.5.1 and then upgrade to 4.0.0 and then to 4.1.1 and then to 4.1.2 and then to 4.2.0
3.5.1	Upgrade to version 4.0.0 and then to 4.1.1 and then to 4.1.2 and then to 4.2.0
4.0.0	Upgrade to version 4.1.1 and then to 4.1.2 and then to 4.2.0
4.1.1	Upgrade directly to version 4.1.2 and then to 4.2.0
4.1.2	Upgrade directly to version 4.2.0
4.2.0	Upgrade directly to versions 4.3.1 and 4.3.2

SAP HANA (VSDK-based) plugin upgrade path

To upgrade from the existing SAP HANA Staged (VSDK-based) plugin follow the recommended path below.

Your version	Recommended upgrade path to 6.3.2
6.0.0 6.0.1	Upgrade to 6.1.0 and follow the path below
6.1.0 6.1.1	Upgrade to 6.2.0 and follow the path below
6.2.0 6.2.1	Upgrade directly to 6.3.2

Your version	Recommended upgrade path to 6.3.2
6.3.0 6.3.1	Upgrade directly to 6.3.2

Compatibility Limitations

The following functionalities, users are **not** supported:

- Multiple duplicate services.
- Point-in-Time (PiT) recovery.
- Scale-out source and target environments.
- High isolation level.
- Partitioning.
- Virtual to Physical (V2P).
- We have not validated SAP HANA Stream Replication (HSR), but it should work for the Staging Push ingestion mechanism.
- Pull-based ingestion mechanism using third-party backups tools.
- More than one instance of the same SAP HANA service.
- Switching between the Staging Push to Pull model.
- Non-database users/Low Privileged users/Users for which privileges need to be elevated.

Oracle E-Business Suite (EBS) matrix

Plugin installation

Plugins to support EBS virtualization are not automatically included in the Virtualization Engine deployment and can be installed by the end-user after uploading the plugin through the Manage > Plugin screen.

Migration and compatibility

Supported versions

- Oracle SI dbTechStack and Database

EBS 12.2

- Oracle11gR2 (Minimum DB version supported 11.2.0.4)
- Oracle12cR1 (Minimum DB version supported 12.1.0.2)
- Oracle19c MT (Minimum DB version supported 19.3.0.0)

Oracle EBS appsTier

- Single-node appsTier
- Multi-node appsTier with shared APPL_TOP

⚠ The Delphix Engine does not provide support for provisioning a multi-node appsTier where the APPL_TOP is not shared between nodes.

- Delphix Support Policies specifically list Major and Minor release coverage. If a minor release is listed as covered, then all patch releases under that minor release are certified.
- Support applies to corresponding versions of Community Enterprise Operating System (CentOS) / Oracle Linux (OL)
- 64-bit OS support only

ⓘ For specific plugin version release notes, see EBS 12.2 Plugin Release Notes.

Key:

Color	Supported?
Y	Yes
N	No
NA	Not Applicable

Red Hat Enterprise Linux (RHEL)

Supported OS version	Supported EBS 12.2 Apps version	
	Oracle 11g/12c	Oracle 19c
RHEL 5.5	Y	NA
RHEL 5.6	Y	NA
RHEL 6.4	Y	NA
RHEL 6.5	Y	NA
RHEL 6.6	Y	NA
RHEL 6.7	Y	NA
RHEL 6.8	Y	NA
RHEL 6.9	Y	NA
RHEL 7.0	Y	NA
RHEL 7.1	Y	NA
RHEL 7.2	Y	NA
RHEL 7.3	Y	NA
RHEL 7.4	Y	NA
RHEL 7.5	Y	Y
RHEL 7.6	Y	Y
RHEL 7.7	Y	Y
RHEL 7.8	Y	Y
RHEL 7.9	Y	Y

RHEL 8.0	**N**	Y
RHEL 8.1	**N**	Y
RHEL 8.2	**N**	Y
RHEL 8.3	**N**	Y
RHEL 8.4	**N**	Y
RHEL 8.5	**N**	Y
RHEL 8.6	**N**	Y
RHEL 8.7	**N**	Y
RHEL 8.8	**N**	Y

Oracle E-Business Suite Installation and Upgrade Notes Release 12 (12.2) for Linux x86-64 (Doc ID 1330701.1)

Solaris SPARC (x86 and SPARC)

Supported OS version	**Supported EBS 12.2.x Apps version**	
	Oracle 12.1.0.2	**Oracle 19c**
Solaris 11 U4	**Y**	**Y**

Delphix Engine version 6.0.17.x along with EBS plugin version 4.2.1 is certified with 19c database, EBS apps 12.2.x and Solaris 11 U4.

Plugin/Delphix engine compatibility

Supported Delphix Engine	Plugin Version for EBS 12.2 with Oracle 12c															Plugin Version for EBS 12.2 with Oracle 19c											
	1.8.2	1.8.3	1.8.4	2.0.0	2.0.1	3.0.0	3.0.1	3.0.2	3.0.3	4.0.2	4.1.0	4.2.0	4.2.1	4.2.2	4.3.0	1.8.2	1.8.3	1.8.4	2.0.0	2.0.1	3.0.0	3.0.1	3.0.2	3.0.3	4.0.2	4.1.0	4.2.0
6.0.0.0	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6.0.1.0	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6.0.2.0	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N

6.0.3.0	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6.0.4.0	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6.0.5.0	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6.0.6.0	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6.0.7.0	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6.0.8.0	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N

6.0.9.0	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N
6.0.10.0	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	Y	Y	N	N	N	N	N
6.0.11.0	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	Y	Y	Y	Y	N	N	N
6.0.12.0	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	Y	Y	Y	Y	N	N	N
6.0.13.0	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	Y	Y	Y	Y	Y	Y	N

6.0.140	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	Y	Y	Y	Y	Y	Y	N
6.0.150	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y
6.0.160	N	N	N	N	N	N	N	N	N	Y	Y	N	N	N	N	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y
6.0.170	N	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y
7.0.0.0	N	N	N	N	N		N	N	N	N	N	N	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N

8.0.0.0	N	N	N	N	N	N	N	N	N	N	N	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	Y
9.0.0.0	N	N	N	N	N	N	N	N	N	N	N	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	Y
10.0.0.x	N	N	N	N	N	N	N	N	N	N	N	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	Y
11.0.0.x	N	N	N	N	N	N	N	N	N	N	N	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	Y
12.0.0.x	N	N	N	N	N	N	N	N	N	N	N	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	Y

13.0.0.x	N	N	N	N	N	N	N	N	N	N	N	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	Y
14.0.0.x	N	N	N	N	N	N	N	N	N	N	N	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	Y

Plugin upgrade path for EBS 12.2 customers

The plugin versions for AppsTier and DB Tech Stack need to be the same, at installation and through the upgrade. Installation and usage of different versions of these plugins are not supported and may encounter operational issues.

New plugin installation for EBS customers

Upload 3.0.0/3.0.1/3.0.2/3.0.3/4.0.2/4.1.0/4.2.0/4.2.1/4.2.2/4.3.0 plugin for EBS 12.2 customers.

EBS plugin version 4.3.0 can handle provision of virtual dbTechStack if AD/TXK codelevel is 13 or later where adpreclone.pl is shipped with version 120.31.12020000.27 or later

Existing EBS customers with different plugin version

Your version	**Recommended upgrade path to 4.3.0**
1.8.2 1.8.3 1.8.4	Upgrade to 2.0.0 or 2.0.1 and follow the upgrade path below.
2.0.0 2.0.1	Upgrade to 3.0.0, 3.0.1, 3.0.2, or 3.0.3 the latest plugin version.

Your version	Recommended upgrade path to 4.3.0
3.0.0 3.0.1 3.0.2 3.0.3	Upgrade directly to version 4.0.2.
4.0.2	Upgrade directly to version 4.1.0.
4.1.0	Upgrade directly to version 4.2.0.
4.2.0 4.2.1 4.2.2	Upgrade directly to version 4.3.0.

For example, upgrade path for EBS 12.2 customer with existing plugin version 1.8.2 who wants to upgrade to 3.0.0 plugin version : 1.8.2 or 1.8.3 or 1.8.4 → 2.0.0 or 2.0.1 → 3.0.0, 3.0.1, 3.0.2, or 3.0.3 → 4.0.2 → 4.1.0 → 4.2.0 or 4.2.1 or 4.2.2→ 4.3.0.

- Customers should skip a minor plugin 1.8.2/1.8.3/1.8.4 version upgrade.
- If EBS 12.2 customers are on plugin version 1.8.2, the plugin upgrade path will be : 1.8.2 → 2.0.0 or 2.0.1 → 3.0.0, 3.0.1, 3.0.2 or 3.0.3 → 4.0.2 → 4.1.0→ 4.2.0 or 4.2.1 or 4.2.2→ 4.3.0.

Unsupported EBS versions and features

- Support SSL enabled Environment
- V2P

MySQL matrix

Connector/Delphix Engine compatibility

Delphix Engine Version	Connector Version				
	2.0.26	**2.1.0**	**3.0.0**	**4.0.0**	**4.1.0**
6.0.4.0	Y	N	N	N	N
6.0.5.0	Y	N	N	N	N
6.0.6.0	Y	N	N	N	N
6.0.7.0 to 6.0.16.0	N	Y	N	N	N
6.0.16.0 and above	N	Y	Y	Y	Y

MySQL connector source and target compatibility

When provisioning a VDB, the MySQL version on the target should be less than that on the source.

Source Environment	Provisioning Target Environment			
	MySQL Community 5.7	**Percona Server 5.7**	**MySQL Community 8.0.x**	**Percona Server 8.0.x**
MySQL Community 5.7	Y	Y	N	N
Percona Server 5.7	Y	Y	N	N
MySQL Community 8.0.x	N	N	Y	Y
Percona Server 8.0.x	N	N	Y	Y

Supported DBMS version

Supported DBMS Version	Connector version				
	2.0.26	2.1.0	3.0.0	4.0.0	4.1.0
MySQL Community 5.7.x	Y	Y	Y	Y	Y
Percona Server 5.7.x	N	N	Y	Y	Y
MySQL Community 8.0.x	N	N	N	Y	Y
Percona Server 8.0.x	N	N	N	Y	Y

Supported operating system

Supported OS Version	Connector version				
	2.0.26	2.1.0	3.0.0	4.0.0	4.1.0
RHEL 6.9 / 7.x	Y	Y	Y	Y	Y

Future releases may add support for additional versions.

Connector upgrade path

To upgrade to the latest MySQL connector follow the recommended path below.

Your version	Recommended upgrade path to 4.1.0
2.0.26 2.0.33	Upgrade to 2.1.0 and follow the upgrade path below.
2.1.0	Upgrade to 3.0.0 and follow the upgrade path below.
3.0.0	Upgrade directly to 4.0.0 and follow the upgrade path below.
4.0.0	Upgrade directly to 4.1.0.

Size in TB / sizing

No known limitations.

Frequency of refreshes

No known limitations.

Source backup policy

Here note any possible issues of the plugin conflicting with or relying on Source Backup Policy.

Ex: This EDSI uses backup ingestion, and relies on the production backup schedule to determine it's snapsync granularity.

Ex: This EDSI uses backup ingestion, but for RDS integration we use a hook and create our own backups.

Use case (PIT Provisioning, Dev/Test)

Here note any possible issues with certain use cases

Ex: Does not support Point-in-Time Provisioning.

Ex: Only useful for migration to the cloud, not appropriate for Dev/Test, Compliance, or AI/ML training.

Delphix option (SDD, API Calls, GUI Icons, GUI PIT Prov)

Here note any Delphix Options which don't work, or which do work but not be readily apparent.

Ex: Supports SDD, API Calls. Does not Support Fancy GUI Icons or GUI PIT Provisioning.

Timestamp on snapshot metadata (Affects other products too (HANA, but not Oracle/SQLServer)

Here note issues/workarounds relating to Timestamps of Snapshot.

Ex: This plugin ingests Backups at any time, so the time flow may not go in linear order. Time of Backup is not readily available.

Architecture options

Clustering (Active/Active vs Active/Passive, different Vendors)

Plugin has not been tested & certified with clustered configurations.

Sharding

Partial

	Un-Sharded VDB	**Sharded VDB**
Un-Sharded Source	**Y**	**N**
Sharded Source	**N**	**N**

Authentication type (LDAP, Kerberos,)

LDAP and Kerberos are not supported at this time, but please contact Ranzo Taylor if you have a customer who requires this.

Ingestion type

mysqldump works, RAW/mysqlbackup does not

- Manual Backup Ingestion
- Simple Tablespace Backup
- Replication

Backup vendor (Commvault, Native, Replication)

Supports Native Backups & Replication

Encryption (At Rest, TLS/SSL, Encrypted Backups)

- Encryption at Rest has not been tested.
- Encrypted Backups have not been tested.

Compatibility limitations

The following functionalities are not supported:

- Pull-based ingestion mechanism using Backup & Recovery.
- Pull-based ingestion mechanism using replication, except Binary Log (binlog).
- Pull-based ingestion mechanism using only backups is not supported.
- Partitioning of the databases.
- Selective replication of databases during dSource creation.
- Windows Operating System

Questions?

If you would like to see a new feature or supported platform reach out to us via the Delphix Support or the Delphix Community Portal.

Kerberos support matrix

Delphix Support Policies specifically list Major and Minor release coverage. If a minor release is listed as covered, then all patch releases under that minor release are certified.

Key:

Color	Supported?
Y	Yes
N	No
NA	Not Applicable

Oracle - Red Hat Enterprise Linux (RHEL)

Supported OS Version	Oracle 10g	Oracle 11gR1	Oracle 11gR2	Oracle 12cR1	Oracle 12cR2	Oracle 18c
	Supported DBMS Version					
RHEL 5.5	Y	Y	Y	NA	NA	NA
RHEL 5.6	Y	Y	Y	Y	NA	NA
RHEL 5.7	Y	Y	Y	Y	NA	NA
RHEL 5.8	Y	Y	Y	Y	NA	NA
RHEL 5.9	Y	Y	Y	Y	NA	NA
RHEL 5.10	Y	Y	Y	Y	NA	NA
RHEL 5.11	Y	Y	Y	Y	NA	NA
RHEL 6.0	NA	NA	Y	Y	NA	NA
RHEL 6.1	NA	NA	Y	Y	NA	NA

RHEL 6.2	NA	NA	Y	Y	NA	NA
RHEL 6.3	NA	NA	Y	Y	NA	NA
RHEL 6.4	NA	NA	Y	Y	Y	Y
RHEL 6.5	NA	NA	Y	Y	Y	Y
RHEL 6.6	NA	NA	Y	Y	Y	Y
RHEL 6.7	NA	NA	Y	Y	Y	Y
RHEL 6.8	NA	NA	Y	Y	Y	Y
RHEL 6.9	NA	NA	Y	Y	Y	Y
RHEL 6.10	NA	NA	Y	Y	Y	Y
RHEL 7.0	NA	NA	Y	Y	Y	Y
RHEL 7.1	NA	NA	Y	Y	Y	Y
RHEL 7.2	NA	NA	Y	Y	Y	Y
RHEL 7.3	NA	NA	Y	Y	Y	Y
RHEL 7.4	NA	NA	Y	Y	Y	Y
RHEL 7.5	NA	NA	Y	Y	Y	Y

SAP ASE - Red Hat Enterprise Linux (RHEL)

Supported OS Version	**Supported DBMS Version**		
	ASE 15.5	**ASE 15.7**	**ASE 16**
RHEL 5.5	Y	Y	**NA**
RHEL 5.6	Y	Y	**NA**

Supported OS Version	Supported DBMS Version		
RHEL 5.7	Y	Y	NA
RHEL 5.8	Y	Y	NA
RHEL 5.9	Y	Y	NA
RHEL 5.10	Y	Y	NA
RHEL 5.11	Y	Y	NA
RHEL 6.0	N	N	NA
RHEL 6.1	N	N	N
RHEL 6.2	Y	Y	N
RHEL 6.3	Y	Y	N
RHEL 6.4	Y	Y	N
RHEL 6.5	Y	Y	Y
RHEL 6.6	Y	Y	Y
RHEL 6.7	Y	Y	Y
RHEL 6.8	Y	Y	Y
RHEL 6.9	Y	Y	Y
RHEL 6.10	Y	Y	Y
RHEL 7.0	NA	Y	Y
RHEL 7.1	NA	Y	Y
RHEL 7.2	NA	Y	Y

Supported OS Version	Supported DBMS Version		
RHEL 7.3	NA	Y	Y
RHEL 7.4	NA	Y	Y
RHEL 7.5	NA	Y	Y

DB2 support matrix

- ESE: Enterprise Server Edition
- AESE: Advanced Enterprise Server Edition
- For the supported DB2 versions, Delphix supports the corresponding DB2 Developer edition where the vendor, IBM, supports it
- 64-bit OS support only

Red Hat Enterprise Linux (RHEL)

Supported OS Version	Supported DBMS Version			
	AESE 10.5	AESE 10.5	ESE 11.1	AESE 11.1
RHEL 6.0	N	N	N	N
RHEL 6.1	N	N	N	N
RHEL 6.2	N	N	N	N
RHEL 6.3	N	N	N	N
RHEL 6.4	Y	Y	N	N
RHEL 6.5	Y	Y	N	N
RHEL 6.6	Y	Y	N	N
RHEL 6.7	Y	Y	Y	Y
RHEL 6.8	Y	Y	Y	Y

RHEL 6.9	Y	Y	Y	Y
RHEL 7.0	Y	Y	Y	Y
RHEL 7.1	Y	Y	Y	Y
RHEL 7.2	Y	Y	Y	Y
RHEL 7.3	Y	Y	Y	Y
RHEL 7.4	Y	Y	Y	Y

DB2 11.1 Developer Edition has been certified on RHEL 6.7.

DB2 on Advanced Interactive eXecutive (AIX)

Supported OS Version	**Supported DBMS Version**			
	AESE 10.5	**AESE 10.5**	**ESE 11.1**	**AESE 11.1**
AIX 6.1	N	N	NA	NA
AIX 7.1	Y	Y	Y	Y
AIX 7.2	Y	Y	Y	Y

Data source certifications

Legal notice

IMPORTANT: The policies outlined in this document are subject to change. The policies and timelines are provided as rough estimates when certifications of interoperability/connectivity with third-party products may be completed and are subject to whether such interoperability/connectivity is part of the Delphix product roadmap and support policy and should not be regarded as a commitment.

Speak with Delphix Product Management if you need more specific information or are unsure of these policies.

Goals

The goal of this certification cadence is to maintain a cadence for certifications of new software releases.

The objectives are:

1. To set reasonable expectations for our organization and customers regarding how and when we will address certifications
2. To define an intended timeframe for certification of new data sources versions and operating systems versions to maintain an up-to-date support matrix

Certification expectations:

Please note that certification only covers previously supported functionality of the Delphix Engine. Any new features and functionality introduced in the database or operating system (DB/OS) will be tracked through the regular release planning and roadmap process and considered as features/functionality/fixes.

Specifically, the following paths exist with varying delivery timeline commitments, in the following priority order:

1. Security Response Policy: In case security issues are discovered in a certain DB and/or OS version, we will follow our security response policy, which supersedes the goals described in this document.
2. Certifications: As described in the document.
3. Roadmap Features: No policies outlined. This follows the regular roadmap planning process.

Data sources

This documentation currently applies only to the following data source integrations:

- Oracle
- SQL Server
- SAP ASE

Cadence for data source database version changes

Release type	Certification details	Goal
A new version of a database with all currently supported operating systems	Certification with all supported operating systems	8 months from new version GA date

Cadence for operating system version changes

Release type	Certification details	Goal
A new major operating system version with all currently supported databases	Certification with all supported versions of the database	6 months from new version GA date
A new update* operating system version with all currently supported databases	Certification with all supported versions of the database	4 months from new version GA date

*Terminology for an 'update' version varies across operating systems. See the Appendix for more information.

Appendix

Product lifecycle of Oracle related DB/OS vendors:

Software	Product lifecycle	Notes
Oracle Database	Link	12.1 is Oracle Enterprise Edition (EE)
Red Hat Enterprise Linux	Link	
Solaris (x86 and SPARC)	Link	
SuSE Linux (SLES)	Link	
IBM AIX	Link	
HP-UX	Link	

Product lifecycle of SQL server related DB/OS vendors:

Software	Product Lifecycle
Microsoft SQL Server	Link
Microsoft Windows	Link

Product lifecycle of SAP ASE related DB/OS vendors:

Software	Product Lifecycle
SAP ASE	Link
Red Hat Enterprise Linux	Link
Solaris (x86 and SPARC)	Link
SuSE Linux (SLES)	Link
IBM AIX	Link

Footnote 1: Update version terminology

Operating System releases typically include updated versions, the terminology for which varies depending on the vendor. Below are the various operating systems which we support, as well as the terminology for updates to use with the document above.

Vendor	Update version terminology	Definition
Red Hat Enterprise Linux	Update version	Installed on top of major releases to provide larger-scale point-release updates via Red Hat Network, after initial installation via traditional methods (PXE, DVD, etc.).
Solaris (x86 and SPARC)	Update release	Update release number. The update release number for this Oracle Solaris release. The update value is 0 for the first customer shipment of an Oracle Solaris release, 1 for the first update of that release, 2 for the second update of that release, and so forth.
SuSE Linux (SLES)	Service Pack (SP) release	Service Packs are minor updates and enhancements, usually with security fixes, that are applied to the major version of the OS.
IBM AIX	Service Pack (SP) release	A Technology Level (TL) gives new features, while an SP contains fixes for problems that are critical and can't wait until the next TL. Service Packs are limited to minimal corrections that don't change the way things work or add any new functionality.

Vendor	Update version terminology	Definition
HP-UX	Patch release	Patch sets are groups of patches that should be installed to support common HPE products and sub-systems.
Windows	New Release (R) version or Service Pack (SP) release	A service pack (SP) is a collection of updates and fixes, called patches, for an operating system or a software program. Many of these patches are often released before a larger service pack, but the service pack allows for an easy, single installation.

vFiles matrix

Unix environments

Delphix Support Policies specifically list Major and Minor release coverage. If a minor release is listed as covered, then all patch releases under that minor release are certified.

Color	Supported?
Y	Yes

Red Hat Enterprise Linux (RHEL)

Supported OS Version	6.0+
	Delphix AppData Version (x86_64)
RHEL 5.5	Y
RHEL 5.6	Y
RHEL 5.7	Y
RHEL 5.8	Y
RHEL 5.9	Y
RHEL 5.10	Y
RHEL 5.11	Y
RHEL 6.0	Y
RHEL 6.1	Y
RHEL 6.2	Y
RHEL 6.3	Y
RHEL 6.4	Y

Supported OS Version	6.0+
RHEL 6.5	Y
RHEL 6.6	Y
RHEL 6.7	Y
RHEL 6.8	Y
RHEL 6.9	Y
RHEL 6.10	Y
RHEL 7.0	Y
RHEL 7.1	Y
RHEL 7.2	Y
RHEL 7.3	Y
RHEL 7.4	Y
RHEL 7.5	Y
RHEL 7.6	Y
RHEL 7.7	Y
RHEL 7.8	Y
RHEL 7.9	Y 6.0.7+
RHEL 8.0	Y
RHEL 8.1	Y 6.0.3+
RHEL 8.2	Y 6.0.3+

Supported OS Version	6.0+
RHEL 8.3	Y 6.0.8+
RHEL 8.4	Y 6.0.11+
RHEL 8.5	Y 6.0.14+
RHEL 8.6	Y 6.0.15+
RHEL 8.7	Y 13.0.0+
RHEL 8.8	Y 13.0.0+

Oracle Enterprise Linux

Supported OS Version	6.0+
	Delphix AppData Version (x86_64)
Oracle Enterprise Linux 5.5	Y
Oracle Enterprise Linux 5.6	Y
Oracle Enterprise Linux 5.7	Y
Oracle Enterprise Linux 5.8	Y
Oracle Enterprise Linux 5.9	Y
Oracle Enterprise Linux 5.10	Y
Oracle Enterprise Linux 5.11	Y
Oracle Enterprise Linux 6.0	Y
Oracle Enterprise Linux 6.1	Y
Oracle Enterprise Linux 6.2	Y

Supported OS Version	6.0+
Oracle Enterprise Linux 6.3	Y
Oracle Enterprise Linux 6.4	Y
Oracle Enterprise Linux 6.5	Y
Oracle Enterprise Linux 6.6	Y
Oracle Enterprise Linux 6.7	Y
Oracle Enterprise Linux 6.8	Y
Oracle Enterprise Linux 6.9	Y
Oracle Enterprise Linux 6.10	Y
Oracle Enterprise Linux 7.0	Y
Oracle Enterprise Linux 7.1	Y
Oracle Enterprise Linux 7.2	Y
Oracle Enterprise Linux 7.3	Y
Oracle Enterprise Linux 7.4	Y
Oracle Enterprise Linux 7.5	Y
Oracle Enterprise Linux 7.6	Y
Oracle Enterprise Linux 7.7	Y
Oracle Enterprise Linux 7.8	Y
Oracle Enterprise Linux 7.9	Y 6.0.7+
Oracle Enterprise Linux 8.0	Y

Supported OS Version	6.0+
Oracle Enterprise Linux 8.1	Y
Oracle Enterprise Linux 8.2	Y 6.0.5+

SUSE Linux Enterprise Server (SLES)

Delphix AppData Version	6.0+
Supported OS Version (x86_64)	
SLES 11	Y
SLES 11 SP1	Y

Solaris

Delphix AppData Version	6.0+
Supported OS Version (SPARC x86_64)	
Solaris 10	Y
Solaris 11	Y

AIX

Delphix AppData Version	6.0+
Supported OS Version (Power)	
AIX 6.1	Y
AIX 7.1	Y
AIX 7.2	Y

HP-UX

Delphix AppData Version	**6.0+**
Supported OS Version (IA64)	
HP-UX 11i v3 (11.31)	Y

Windows environments

Delphix AppData Version	**6.0+**
Supported OS Version	
Windows Server 2012	Y
Windows Server 2012 R2	Y
Windows Server 2016	Y
Windows Server 2019	Y
Windows Server 2022	Y

Select connectors matrix

Color	Supported?
Y	Yes
N	No
NA	Not Applicable

Couchbase

Data source version and OS version

	RHEL 6.x	RHEL 7.0	RHEL 8.0	Windows
Couchbase 6.0	Y	Y	Y	N
Couchbase 6.5	Y	Y	Y	N
Couchbase 6.6	Y	Y	Y	N

Size in TB / sizing

- No known restrictions.
- Memory, Network, IOPS metrics from Couchbase GUI

Frequency of refreshes

No specific limitations

Source backup policy

- Supports backups using Couchbase Backup Manager (cbbkpmgr)
- Supports Couchbase XDCR technology

Use case (PIT Provisioning, Dev/Test)

- Does not support Point-in-Time Provisioning.
- Not appropriate for high throughput performance requirement.

Delphix option (SDD, API Calls, GUI Icons, GUI PIT Prov)

- V2P is not supported.

Timestamp on snapshot metadata

This plugin ingests Backups at any time, so the time flow may not go in linear order. Time of Backup is not readily available.

Architecture options

- Only 1 VDB per host is supported (couchbase restriction of 1 instance per Node)
- Dedicated 1 Staging host is needed for each dSource created using XDCR technology

	Community Edition **XDCR only**	**Enterprise Edition**
Couchbase 6.0	**Y**	**Y**
Couchbase 6.5	**Y**	**Y**
Couchbase 6.6	**Y**	**Y**

Authentication type (LDAP, Kerberos,)

- Kerberos is not supported.
- LDAP not tested.

Ingestion type (Backup, Dump, Replication, Snapshot)

- Couchbase Backup Manager (cbbkpmgr)
- XDCR

Link to documentation

https://delphix.github.io/couchbase-plugin

Mongo

Data Source version and OS version

Supported Mongo Version	**mongoDB 4.2.x**	**mongoDB 4.4.x**	**mongoDB 5.0.x**	**mongoDB 6.0.x**
RHEL 6.x	**Y**	**Y**	**Y**	**Y**
RHEL 7.x	**Y**	**Y**	**Y**	**Y**
RHEL 8.x	**Y**	**Y**	**Y**	**Y**
CentOS 6.x	**Y**	**Y**	**Y**	**Y**

Supported Mongo Version	mongoDB 4.2.x	mongoDB 4.4.x	mongoDB 5.0.x	mongoDB 6.0.x
CentOS 7.x	Y	Y	Y	Y
CentOS 8.x	Y	Y	Y	Y
Windows	N	N	N	N

Engine Compatibility Matrix

MongoDB Version	Mongopy 1.0.0	Mongopy 1.0.1	Mongopy 1.0.2	Mongopy 1.1.0	Mongopy 1.2.0	Mongopy 1.2.1	Mongopy 1.3.0
6.0.7.x-6.0.11.x	Y	Y	N	N	N	N	N
6.0.12.x-6.0.15.x	Y	Y	Y	Y	N	N	N
7.x-12.x	N	N	N	N	Y	Y	Y

Mongo/Plugin Version Compatibility Matrix

MongoDB Version	Mongopy 1.0.0	Mongopy 1.0.1	Mongopy 1.0.2	Mongopy 1.1.0	Mongopy 1.2.0	Mongopy 1.2.1	Mongopy 1.3.0
4.x-5.x	Y	Y	Y	Y	Y	Y	Y
6.x	N	N	N	N	Y	Y	Y

Size in TB / sizing

- Not tested with large datasets.
- Memory, Network, IOPS metrics from Mongo OPS Manager

Frequency of refreshes

No specific limitations

Source backup policy

- Sharded Mongo always ingest full. Non-Sharded mongo ingestion may be full or incremental based on the mechanism used for ingestion.
- Sharded : Full
- Non-Sharded (Using Backups) - Full
- Non-sharded (Using replicaset member) - incremental

- Mongodump - Full or incremental based on type of source and type of ingestion

Use case (PIT Provisioning, Dev/Test)

- Does not support Point-in-Time Provisioning.
- Not appropriate for high throughput performance requirement.

Delphix option (SDD, API Calls, GUI Icons, GUI PIT Prov)

- V2P is not supported.

Timestamp on snapshot metadata

This plugin ingests Backups at any time, so the time flow may not go in linear order. Time of Backup is not readily available.

Architecture options

Sharding

Supported

	Unsharded VDB	Sharded VDB
Unsharded Source >= 3.6	Y	N
Sharded Source >= 4.2	N	Y

Example: Sharded sources are supported, but require backup vendor MongoOPS Manager to create backup.

Authentication type (LDAP, Kerberos,)

- LDAP and SCRAM supported
- Kerberos not supported

Ingestion type (Backup, Dump, Replication, Snapshot)

- MongoOPS Manager Backup Files
- Mongodump
- Replicaset

Encryption (At Rest, TLS/SSL, Encrypted backups)

Supported

Link to documentation

https://delphix.github.io/mongo-plugin/

Microsoft SQL backup ingestion

Data source version and OS version

	Windows 2012	Windows 2012 R2	Windows 2016	Windows 2019	Windows 2022
SQLServer 2012	Y	Y	Y	N	N
SQLServer 2014	Y	Y	Y	N	N
SQLServer 2016	Y	Y	Y	Y	N
SQLServer 2017	Y	Y	Y	Y	Y
SQLServer 2019	N	N	Y	Y	Y
SQLServer 2022	N	N	Y	Y	Y

Size in TB / sizing

No sizing limitation

Frequency of refreshes

No limitation

Source backup policy

dSource sync is managed by customers and can apply backup on staging whenever required.

Use case (PIT provisioning, Dev/Test)

Does not support Point-in-Time Provisioning.

Delphix option (SDD, API Calls, GUI Icons, GUI PIT Prov)

V2P is not supported

Timestamp on snapshot metadata (Affects other products too (HANA, but not Oracle/SQLServer)

This plugin ingests Backups at any time, so the time flow may not go in linear order.

Architecture options

Failover clusters and availability groups

- Does not support Windows Failover Clusters or Availability Groups for Windows Staging and Target environments.
- Authentication Type (LDAP, Kerberos,)

Ex: LDAP and Kerberos are not supported at this time, but please contact Ranzo Taylor if you have a customer who requires this.

Ingestion type (Backup, Dump, Replication, Snapshot)

Full backups, Differential backups, Log Shipping, Transaction replication

Backup vendor (Commvault, Native, Replication)

All backup vendors are supported

Encryption (At Rest, TLS/SSL, Encrypted Backups)

- Encryption at Rest should work, but has not been tested.
- Encrypted Backups should work, and has not been tested.

Link to documentation

https://delphix.github.io/mssql_plugin_doc

MySQL Linux

Connector/Delphix engine compatibility

Delphix Engine Version	Connector Version				
	1.0.0	2.0.26	2.1.0	3.0.0	4.0.0
6.0.4.0	Y	Y	N	N	N
6.0.5.0	Y	Y	N	N	N
6.0.6.0	Y	Y	N	N	N
6.0.7.0 to 6.0.16.0	N	N	Y	N	N
6.0.16.0 and above	N	N	Y	Y	Y

Supported DBMS version

Supported DBMS Version	Connector version				
	1.0.0	2.0.26	2.1.0	3.0.0	4.0.0
MySQL Community 5.7.x	Y	Y	Y	Y	Y
Percona Server 5.7.x	N	N	N	Y	Y

Supported DBMS Version	Connector version				
MySQL Community 8.0.x	N	N	N	N	Y
Percona Server 8.0.x	N	N	N	N	Y

Supported operating system

Supported OS Version	Connector version				
	1.0.0	2.0.26	2.1.0	3.0.0	4.0.0
RHEL 6.9 / 7.x	Y	Y	Y	Y	Y

Size in TB / sizing

No known limitations.

Frequency of refreshes

No known limitations.

Source backup policy

Here note any possible issues of the plugin conflicting with or relying on Source Backup Policy.

Ex: This EDSI uses backup ingestion, and relies on the production backup schedule to determine it's snapsync granularity.

Ex: This EDSI uses backup ingestion, but for RDS integration we use a hook and create our own backups.

Use case (PIT Provisioning, Dev/Test)

Here note any possible issues with certain use cases

Ex: Does not support Point-in-Time Provisioning.

Ex: Only useful for migration to the cloud, not appropriate for Dev/Test, Compliance, or AI/ML training.

Delphix option (SDD, API Calls, GUI Icons, GUI PIT Prov)

Here note any Delphix Options which don't work, or which do work but not be readily apparent.

Ex: Supports SDD, API Calls. Does not Support Fancy GUI Icons or GUI PIT Provisioning.

Timestamp on snapshot metadata (Affects other products too (HANA, but not Oracle/SQLServer)

Here note issues/workarounds relating to Timestamps of Snapshot.

Ex: This plugin ingests Backups at any time, so the time flow may not go in linear order. Time of Backup is not readily available.

Architecture options

Clustering (Active/Active vs Active/Passive, different Vendors)

Plugin has not been tested & certified with clustered configurations.

Sharding

Partial

	Un-Sharded VDB	Sharded VDB
Un-Sharded Source	Y	N
Sharded Source	N	N

Authentication type (LDAP, Kerberos,)

LDAP and Kerberos are not supported at this time, but please contact Ranzo Taylor if you have a customer who requires this.

Ingestion type

mysqldump works, RAW/mysqlbackup does not

- Manual Backup Ingestion
- Simple Tablespace Backup
- Replication

Backup vendor (Commvault, Native, Replication)

Supports Native Backups & Replication

Encryption (At Rest, TLS/SSL, Encrypted Backups)

- Encryption at Rest has not been tested.
- Encrypted Backups have not been tested.

Link to documentation

MySQL environments and data sources

Oracle backup ingestion

Data source version and OS version

	RHEL 6.x	RHEL 7.0	Windows
Oracle 9i/10g/11r1	N	N	N
Oracle >= 11gR2	Y	Y	N

Size in TB / sizing

No Restrictions.

Frequency of refreshes

No specific limitations

Source backup policy

No limitations for disk.

For Tape Backup - Autobackup should be enabled.

Use case (PIT Provisioning, Dev/Test)

- Disk - Archivelog need to be made available on target host for Point-in-Time Provisioning.
- Tape - Tape library installed on target and authorized to access tape from target host.
- Support using timestamp.

Delphix option (SDD, API Calls, GUI Icons, GUI PIT Prov)

- V2P is not supported.

Timestamp on snapshot metadata

This plugin ingests Backups at any time, so the time flow may not go in linear order. Time of Backup is not readily available.

Architecture Options

Full	Y	Y	Y	Y	Y	NA
Partial	Y	Y	Y	Y	Y	NA
PDB	Y	Y	Y	Y	Y	N
Partial PDB	Y	Y	Y	Y	Y	N
Use Existing CDB	NA	NA	NA	N	NA	N
Seed	NA	NA	NA	NA	N	NA

RAC

Manual Addition, Selection of nodes and Checks needed to create successful RAC VDBs.

Ingestion type (Backup, Dump, Replication, Snapshot)

- RMAN database backup

Backup vendor (Oracle RMAN, ZDLRA, DDBOOST, NETBACKUP, Generic Tape Drivers)

- Oracle RMAN
- ZDLRA
- DDBOOST
- NETBACKUP
- Other Tape Vendors not listed above not tested but most probably will work

Link to documentation

https://delphix.github.io/obi-plugin/NOT TESTED

SAPIQ

Data source version and OS version

Supported OS Version	SAPIQ 16.0	SAPIQ 16.1
RHEL 7.0	Y	Y
RHEL 7.4	Y	Y
RHEL 7.9	Y	Y
Windows x.x	N	N

Size in TB / sizing

Tested at the customer with 600-800GB ingestion size.

Frequency of refreshes

No limitation

Source backup policy

Customers will manage backup located on the staging host required to create dSource.

Use case (PIT Provisioning, Dev/Test)

Does not support Point-in-Time Provisioning.

Delphix option (SDD, API Calls, GUI Icons, GUI PIT Prov)

- V2P is not supported.
- Password Vault is not supported.
- SDD is not supported.

Timestamp on snapshot metadata

This plugin ingests backups at any time, so the time flow may not go in linear order. Time of Backup is not readily available.

Architecture options

Supports SAPIQ Simplex Architecture for VDBs.

Architecture	SAPIQ 16.0	SAPIQ 16.1
Simplex	Y	Y
Multiplex	N	N

- IQ Plugin converts multiplex architecture to simplex during VDB provision.
- Ingest SAPIQ source databases which are backed up using SAPIQ built-in backup mechanism.
- Data Ingestion to Delphix Engine will always start with a full DB backup.
- As best practices, it is recommended to keep a similar SAPIQ version in all environments (source, staging and target).

Authentication type (LDAP, Kerberos)KE

- Kerberos not supported.
- LDAP not tested.

Ingestion type (Backup, Dump, Replication, Snapshot)

Full backups for Simplex Architecture Tested.

Backup Vendor

Tested with manual SAPIQ backups.

Encryption (At Rest, TLS/SSL, Encrypted Backups)

- Encryption at Rest has not been tested.
- Encrypted Backups have not been tested.

Link to documentation

https://delphix.github.io/SAPIQ/

Cockroach

Data source version and OS version

Supported OS Version	CRDB v21.1.x	CRDB v21.2.x	CRDB v22.1.x	CRDB v22.2.x
RHEL 7.x	Y	Y	Y	Y
RHEL 8.x	Y	Y	Y	Y
Windows	N	N	N	N

Architecture options

Architecture	Disk	Seed	AWS	GCP	Azure
Full	Y	Y	Y	N	N
Incremental	Y	Y	Y	N	N

Engine compatibility matrix

Engine Version	CRDB v1.0.0	CRDB v1.1.0	CRDB v1.1.1	CRDB v1.1.2
6.0.12.0 - 6.0.17.x	Y	Y	Y	Y
7.0.0.0 - 13.0.0.0	Y	Y	Y	Y

PostgreSQL JDBC driver matrix

PostgreSQL Driver Version	Supported
42.3.2	Y
42.3.6	Y
42.4.0	Y

JDBC masking support matrix

Masking Mode	Feature	Availability
In-Place Masking	Multi-Tenant	Available
	Streams/Threads	Available
	Batch Update	Available
	Drop Indexes	Unavailable
	Disable Constraint	Unavailable

Masking Mode	Feature	Availability
	Disable Trigger	Unavailable NA for CockroachDB
	Identity Column Support	Available
On-The-Fly Masking		Unavailable
Profiling	Multi-Tenant	Available
	Streams	Available

- Drop Indexes and Disable Constraint are not supported natively while masking CockroachDB. Use pre/post scripts if you want to use these features.
- CockroachDB supports rowid natively for tables with no primary key defined. It will automatically create a primary key over a hidden, INT-typed column named rowid.
- To create a successful JDBC connection to CockroachDB cluster running in secure mode, please enable the password authentication without TLS using flag `--accept-sql-without-tls` wi th cockroach start command.
- To mask the CockroachDB data source where CDC (change data capture) feature is enabled, disable all the running change-feed jobs using the below command

```
SHOW CHANGEFEED JOBS; PAUSE JOBS (WITH x AS (SHOW CHANGEFEED JOBS)
SELECT * FROM x WHERE status = ('running'));
```

Copy

- Resume CDC job once CockroachDB masking is completed.

```
RESUME JOBS (WITH x AS (SHOW CHANGEFEED JOBS) SELECT * FROM x WHERE
status = ('paused'));
```

Ingestion type

- AWS S3 Backup Ingestion
- Seed Ingestion

Limitations

- CockroachDB Ingestion from Azure, GCP, and Disk is not supported.
- CockroachDB Point-in-time restore is not supported.
- Delphix V2P is not supported.

- Delphix Password Vault is not supported.

Link to documentation

https://delphix.github.io/crdb/

Salesforce

JDBC masking

	Feature	Availability
In-Place Masking Mode	Multi-Tenant	Available
	Streams/Threads	Available
	Batch Update	Available
	Drop Indexes	Unavailable
	Disable Trigger	Unavailable
	Disable Constraint	Unavailable
	Identity Column Support	Available
On-The-Fly Masking Mode		Unavailable
Profiling	Multi-Tenant	Available
	Streams	Available

Delphix compliance accelerator

Platform \ dxac	dxac v2.0.0	dxac v2.1.0
Delphix Continuous Compliance Engine	6.0.10 - 6.0.17	7.0.0+ (OEM enabled)
Continuous Compliance API	5.1.9	5.1.9
Salesforce API	v51+	v51+
MacOS	11.6.0+	11.6.0+

Platform \ dxac	dxac v2.0.0	dxac v2.1.0
Windows	10	10
RHEL	7.9, 8.x	7.9, 8.x
CDATA Salesforce Driver Version	21.0.8137, 21.0.8201.0	22.0.8301

Delphix rehearsal tool

Platform \ dxrt	dxrt v3.2.0
Salesforce API	v56+
MacOS	11.6.0+
Windows	10
RHEL	8.x

Salesforce package manager

Platform \ Package Manager	Package Manager v1.0.0
Delphix Continuous Compliance Engine	6.0.10+
Continuous Compliance API	5.1.9
Salesforce API	v51+
CDATA Salesforce Driver Version	21.0.8201.0

Data protect & version

Platform \ Data Protect & Version	v3.0.0
Delphix Engine	6.0.15+
Salesforce Platform	Spring ‘23 (v.57), Winter ‘23 (v.56)
Salesforce API	53

Platform \ Data Protect & Version	v3.0.0
Staging Host	RHEL 7.9, 8.2

Link to documentation

Select Connector for Salesforce Documentation

SAP

JDBC masking

	Feature	Oracle	HANA
In-Place Masking Mode	Multi-Tenant	Available	Available
	Streams/Threads	Available	Available
	Batch Update	Available	Available
	Drop Indexes	Available	Unavailable
	Disable Trigger	Available	Unavailable
	Disable Constraint	Available	Unavailable
	Identity Column Support	Available	Unavailable
On-The-Fly Masking Mode		Available	Available
Profiling	Multi-Tenant	Available	Available
	Streams	Available	Available

Delphix compliance accelerator

Platform \ dxac	dxac v2.1.0
Delphix Continuous Compliance Engine	6.0.15+
Continuous Compliance API	5.1.9

Platform \ dxac	dxac v2.1.0
MacOS	11.6+
Windows	10
RHEL	7.9, 8.x

SAP source version

Applications	Database	Driver
SAP S/4 HANA 1909	HANA 2.0 SP04, SP05, and SP06	SAP NDBC Driver 2.12.7
SAP S/4 HANA 2020	HANA 2.0 SP04, SP05, and SP06	SAP NDBC Driver 2.12.7
SAP ECC	Oracle	Delphix Compliance Engine Native Connector

Link to documentation

SAP Compliance Accelerator Documentation

Cassandra

Delphix engine compatibility

Engine Version	Cassandra v1.0.0	Cassandra v1.1.0
6.0.16.0	Y	Y
6.0.17.0	Y	Y
7.0.0.0	Y	Y
10.0.0.0	Y	Y

Supported Database / OS

Supported DBMS Version	DataStax Cassandra v5.1.16	DataStax Cassandra v6.8.25 - v6.8.33	Apache Cassandra v4.0.7 - v4.1.1
RHEL 7.4	Y	Y	Y
RHEL 7.9	Y	Y	Y
RHEL 8.0	Y	Y	Y
Windows	N	N	N

Supported Ingestion

Cassandra Distribution	Staging Push	Disk	AWS
DataStax Enterprise	Y	N	N
Open-Source Apache	Y	N	N

Link to documentation

https://delphix.github.io/cassandra/

Upgrade matrix

This section lists all the Delphix Engine versions that a user can upgrade to the required or the latest version.

Self-service upgrades of the Delphix Engine to version 6.0.x.x from version 5.3.0 or older is not supported.
Versions 5.3.6.0 - 5.3.9.0 can only be upgraded as far as 6.0.6.1, and can then be upgraded to a newer version.

Starting with version 6.0.10.0, direct upgrades will only be supported to versions released in the last 12 months, per Delphix policy.
Version 6.0.4.1 and above can upgrade directly to version 6.0.10.1.
Versions 6.0.0.0 - 6.0.4.0 can only be upgraded as far as 6.0.7.0, and can then be upgraded to 6.0.10.1.

Version	Date Released	Can Upgrade To	VDB Downtime Required
> 5.3.5.0	Aug 8th, 2019	Contact Support	Yes
5.3.6.0	Oct 10th, 2019	6.0.0.0 - 6.0.6.1	Yes
5.3.7.0	Dec 9th, 2019	6.0.1.0 - 6.0.6.1	Yes
5.3.7.1	Jan 13th, 2020	6.0.1.0 - 6.0.6.1	Yes
5.3.8.0	Feb 10th, 2020	6.0.1.0 - 6.0.6.1	Yes
5.3.8.1	Feb 26th, 2020	6.0.1.0 - 6.0.6.1	Yes
5.3.9.0	Apr 14th, 2020	6.0.2.0 - 6.0.6.1	Yes
6.0.0.*	Jan 22nd, 2020	6.0.0.* - 6.0.7.*	Optional
6.0.1.*	Mar 19th, 2020	6.0.1.* - 6.0.7.*	Optional
6.0.2.*	May 7th, 2020	6.0.2.* - 6.0.7.*	Optional
6.0.3.*	Jul 31st, 2020	6.0.3.* - 6.0.7.*	Optional
6.0.4.*	Sep 10th, 2020	6.0.4.* - 6.0.7.*	Optional
6.0.5.*	Nov 5th, 2020	6.0.5.* - 6.0.11.*	Optional

Version	Date Released	Can Upgrade To	VDB Downtime Required
6.0.6.*	Jan 21st, 2021	6.0.6.* - 6.0.12.*	Optional
6.0.7.*	Mar 15th, 2021	6.0.7.* - 6.0.13.*	Optional
6.0.8.*	May 13th, 2021	6.0.8.* - 6.0.14.*	Optional
6.0.9.*	Jul 13th, 2021	6.0.9.* - 6.0.15.*	Optional
6.0.10.*	Sep 9th, 2021	6.0.10.* - 6.0.16.*	Optional
6.0.11.*	Nov 11th, 2021	6.0.11.* - 6.0.17.*	Optional
6.0.12.*	Jan 14th, 2022	6.0.12.* - 7.0.0.*	Optional
6.0.13.*	Mar 10th, 2022	6.0.13.* - 9.0.0.*	Optional
6.0.14.*	May 20th, 2022	6.0.14.* - 11.0.0.*	Optional
6.0.15.*	Jul 7th, 2022	6.0.15.* - 13.0.0.*	Optional
6.0.16.*	Sep 8th, 2022	6.0.16.* - 14.0.0.*	Optional
6.0.17.*	Nov 15th, 2022	6.0.17.* - 14.0.0.*	Optional
7.0.0.*	Jan 12th, 2023	7.0.0.* - 14.0.0.*	Optional
8.0.0.*	Feb 13th, 2023	8.0.0.* - 14.0.0.*	Optional
9.0.0.*	Mar 13th, 2023	9.0.0.* - 14.0.0.*	Optional
10.0.0.*	Apr 13th, 2023	10.0.0.* - 14.0.0.*	Optional
11.0.0.*	May 24th, 2023	11.0.0.* - 14.0.0.*	Optional
12.0.0.*	Jun 21st, 2023	12.0.0.* - 14.0.0.*	Optional
13.0.0.*	Jul 19th, 2023	13.0.0.* - 14.0.0.*	Optional

Version	Date Released	Can Upgrade To	VDB Downtime Required
14.0.0.*	Aug 23rd, 2023	14.0.0*	N/A

The * in the above table represents patch releases. When upgrading from X.Y.Z.x to X.Y.Z.x, the target version must be greater than the source version. Please refer to the How to Upgrade article for steps, or check with Delphix Support to perform the upgrade.

1. VDB Downtime is caused by a reboot of the Delphix Engine when DelphixOS is modified by an upgrade.
2. VDB Downtime may be optional for an upgrade when a release contains DelphixOS changes that are also optional.

Tested browser and operating systems

This topic describes the Web browsers and operating systems that have been tested for use with the Delphix Management application.

The following table lists the supported browsers and operating systems.

OS	Browsers
Windows 11	Microsoft Edge, Firefox, Chrome
Windows 10	Microsoft Edge, Firefox, Chrome
Windows 8.1	Microsoft Edge, Firefox, Chrome
Windows 7	Microsoft Edge, Firefox, Chrome
Mac OS X	Microsoft Edge, Firefox, Chrome

⚠ Delphix supports the latest version of Microsoft Edge, Firefox, and Chrome as well as the immediate prior version.

Deprecated and end-of-life features

Release 11.0.0.0

End-of-life feature

- ESXi 6.0 Support

Release 10.0.0.0

Deprecated features

- **ESXi 6.5 and 6.7**
  VMware's end of technical guidance for ESXi 6.5 and 6.7 is on 11/15/2023. To align our support, Delphix will end support of ESXi 6.5 and 6.7 in version 15.0.0.0 (November 2023 release).

Release 9.0.0.0

Deprecated features

- **Oracle**
  Oracle `move-to-asm.sh` scripted procedure has been deprecated in favor of the new feature "**Export an Oracle virtual DB or virtual PDB to a physical Oracle ASM or Exadata database**". For more details on this new feature, refer to the Export an Oracle VDB or vPDB to a Physical ASM or Exadata Database section. The `move-to-asm.sh` scripted procedure is still supported in 9.0.0.0, but is discouraged by Delphix and will be eventually removed in future versions of the Delphix Continuous Data Engine.

Release 6.0.17.0

End-of-life features

- **Oracle 11.1**
  Details of the Oracle database EOL can be found in the Oracle Lifetime Support Policy PDF.

Release 6.0.12.0

End-of-life features

- **Internet Explorer 11 support**
  Internet Explorer 11 is no longer supported by Delphix. The users are requested to refer to the list of supported browsers.
- **EBS 12.1 support**
  Starting 6.0.12.0 and beyond, EBS 12.1 is no longer supported by Delphix. The users are requested to upgrade to EBS 12.2 and start using this plugin.

- **SAP HANA 1.0 support**
  Starting 6.0.12.0 and beyond, SAP HANA 1.0 is no longer supported by Delphix. The users are requested to upgrade to HANA 2.0 and start using this plugin.

Release 6.0.11.0

Deprecated features

- **Oracle 11.1**
  Details of the Oracle database EOL can be found in the Oracle Lifetime Support Policy PDF.
- **TLS 1.0 and 1.1**
  These versions of TLS are known to be vulnerable, enterprise use is heavily discouraged.

End-of-life features

- **Oracle 10g Support**
- **TLS 1.0 and 1.1**
  These versions of TLS are known to be vulnerable, enterprise use is heavily discouraged.

Release 6.0.10.0

End-of-life features

- **EBS12.1**
  EBS 12.1 remains deprecated with a planned EOL (and removal from the product) in December 2021. This is in line with Oracle's planned EOL for EBS 12.1 later this year. The users are encouraged to upgrade to Delphix's EBS 12.2.
- **Masking ruleset edit options**
  The Table Suffix, Add Column, Join Table, and List options were deprecated in the 6.0.3.0 release. These options have reached the EOL in the 6.0.10.0 release and have been completely removed from the product.
- **Internet Explorer 11 support**
  Internet Explorer 11 is planned for EOL in December 2021. This is in line with Microsoft's planned EOL for Internet Explorer 11 later this year.

Release 6.0.9.0

End-of-life features

- Delphix Reporting (Formerly, *Mission Control*) is EOL starting 6.0.9.0 (July 1st, 2021). The users interested in continuing their reporting workflows are recommended to discuss Data Control Tower with their account teams.
- Also, EBS 12.1 remains deprecated with a planned EOL (and removal from the product) in December 2021. This is in line with Oracle's planned EOL for EBS 12.1 later this year. The users are encouraged to upgrade to Delphix's EBS 12.2.

Release 6.0.8.0

End-of-life features

- SAP ASE 15.0.3
- Windows Server 2008 R2
- SQL Server 2008
- SQL Server 2008 R2

Release 6.0.7.0

Masking End-of-life features

- ESX 5.5 support will be end-of-life in 6.0.7
- Masking Connectors: Db2 LUW and zOS v9, Db2 LUW and zOS v10, SQL Server 2005, 2008, 2008 R2

Release 6.0.5.0

The following list of OS and DB versions are now past their respective support windows. Delphix will deprecate these versions to stay in line with Microsoft, Oracle, and SAP lifecycle policies:

- Windows 2008 R2: Microsoft End of Extended Support (EoES) January 4th, 2020
- SQL Server 2008/2008 R2: Microsoft End of Extended Support (EoES) July 9th, 2019
- Oracle 10: Oracle End of Extended Support (EoES) July 2013
- SAP ASE 15.0.3: SAP End-of-Life (EoL) March 31st, 2015

Release 6.0.4.0

End-of-life features

- **Masking**:
    - Job Scheduler - As of this release, Delphix has removed the Job Scheduler feature. The introduction of Masking's REST API several releases ago allowed customers to schedule job executions using their preferred job scheduler. As a result, the integrated scheduler is seldom used.

Deprecated features

- **Masking**:
    - FTP, SFTP, and mount upload for XML and Cobol formats - FTP/SFTP/Mount-based format import were the original modes for XML and Cobol files, since then, Delphix has added the ability to upload a format file, which is far simpler to set up. After the introduction of "upload", we've seen a dramatic shift away from the legacy import modes in favor of the simplicity of "upload".
    - Row Type Feature - Originally geared for limiting masking to subsets of rows within a column, this feature was seldomly used. Its functionality, if desired, can still be replicated via the Custom SQL feature.
    - Redundant Settings for 'Edit Table' under Rule Sets - Table Suffix, Add Column, Join Table, and List - These settings are redundant and can be replicated with the Custom SQL setting.
    - 'HAVING' clause from Masking API - Deprecating due to low use. This feature, if desired, can be replicated with Custom SQL.

Release 6.0.2.0

End-of-life features

- AWS i.3 (EOL) January 22, 2020
- Microsoft Azure GSx (EOL) May 7, 2020

Release 6.0.0.0

End-of-life features

- **Removed Virtualization Features**:
  A few features will be removed with the 6.0 release. If these features are in use, the upgrade will not proceed.
    - Cross Platform Provisioning (XPP)

- AIX 6.1 Technology Levels 0-6 and AIX 7.1 Technology Levels 0-2 for Environments
- Windows Server 2008 for Environments

> Windows Server 2008 will not have an upgrade blocker.

- **Removed Masking Features**:
  Please note that Excel files can still be masked by first converting them to one of Delphix's supported file types (CSV, etc). Also, XML CLOBs can be masked by extracting their values into a table (example - using extractValue in Oracle).
  - Native XML CLOB masking: After upgrade, columns masked as XML CLOBs will have the NULL SL algorithm assigned.
  - DB2 9.1, 9.5, and other 9.x versions of LUW & Z/OS
  - "Create target" job option: After upgrade jobs using "create target" will be removed.
  - "Bulk data" job option: After upgrade, jobs using "bulk data" will be turned into non-bulk data jobs.
  - Native Microsoft Excel Masking: After upgrade, MS Excel connectors, rulesets and jobs will be removed.

Licenses and notices

The Delphix Dynamic Data Platform includes licensed, third-party products from the following companies. These products are copyrighted and all rights are reserved by the respective companies:

- Highcharts, © Highsoft

The Delphix Masking engine includes licensed, third-party products from the following companies. These products are copyrighted and all rights are reserved by the respective companies:

- Kendo UI, © Telerik

Starting with 6.0.3.0, the license info is available via a CLI/API on the engine when logged in as a system administrator.

```
engine> cd license
engine license> getLicense
engine license getLicense *> commit
```

Access to the source code of such third party open source components may be permitted or required in certain instances under the applicable open source licenses by sending an email to mailto:open-source@delphix.com

Data source integration (plugin) release notes

This section covers the following topics:

IBM Db2 release notes

This section covers the following topics:

- New features (IBM Db2)
- Fixed issues (IBM Db2)
- Known issues (IBM Db2)

To view the IBM Db2 support matrix, see IBM Db2 matrix.

New features (IBM Db2)

Release 4.6.1

This plugin release consists of only bug fixes.

Release 4.6.0

This plugin release has the following set of new features and bug fixes.

- **Staging push support for DPF databases**
  The Db2 Plugin 4.6.0 version introduces Staging push support for DPF databases. This allows you to ingest databases using a backup agent solution of your choice.
- **VDB Provision with user-defined mount base**
  This plugin version introduces support for user-defined mount base for VDBs, which allows you to provide a mount base of your choice during the provisioning of a VDB. **Note:** DIR_CACHE for the instance needs to be set to NO when users want to update the VDB dataset with the new mount base.
- **Optionally skip archive log validation**
  With this feature, you can skip the archive log validation process during Snapsync if you choose to do so.
- **Load copy files support for Staging push**
  The load copy files are now supported by the DB2 staging push automation script. This allows you to provide the load copy files along with the archive logs to successfully roll forward the database.

Release 4.5.1

This plugin release consists of only bug fixes.

Release 4.5.0

This plugin release has the following set of new features and bug fixes.

- **Support for multiple backup paths for dSource ingestion**
  The Db2 Plugin 4.5.0 version introduces the support for multiple backup paths instead of one for dSource ingestion. This allows you to provide single backup path for the dSource ingestion and add more backup paths with a dynamic UI to accommodate for the extra backup directories where the backup was split into.

Release 4.4.2

This plugin release consists of only bug fixes.

Release 4.4.1

This plugin release consists of only bug fixes.

Release 4.4.0

This plugin release has the following set of new features and bug fixes.

- **DB2 pull performance optimization**
  The Db2 Plugin 4.4.0 version introduces the Pull Performance Optimization feature, which attempts to optimize the dSource workflows by allowing users to provide the degree of parallelisation to run the database restore operation. This feature allows users to ingest large databases in a faster way, provided that there are sufficient resources for Db2 plugin to implement this parallelisation in the operation.
- **Source sizing implementation on the DSIL DB2 plugin**
  The source sizing feature aligns the data source experience with Delphix database standards that will improve your ingestion reporting experience on the per TB pricing model. This feature enables you to

calculate the dataset size based on the total allocated space on the mount point provided by the Delphix Engine for the dataset. For more details about this feature, see Source Sizing Implementation for Datasets.

Release 4.3.2

This plugin release consists of only bug fixes.

Release 4.3.1

This plugin release consists of only bug fixes.

Release 4.3.0

The DB2 plugin 4.3.0 introduces Staging push support for HADR dSources. You can now create and maintain dSources configured with HADR using the Staging push option.

Release 4.2.1

This plugin release consists of only bug fixes.

Release 4.2.0

This plugin release has the following set of new features and bug fixes.

- **Dsource snapshot with user-defined timestamp**
  The Db2 Plugin version 4.2.0 introduces the dsource snapshot with the user-defined timestamp feature. This feature enables the user to specify a timestamp during dSource snapshot so that staging will roll forward to the requested timestamp before performing the dSource snapshot. For more information, see Linking a Db2 dSource.

Release 4.1.3

This plugin release consists of only bug fixes.

Release 4.1.2

This plugin release consists of only bug fixes.

Release 4.1.1

This plugin release consists of only bug fixes.

Release 4.1.0

This plugin release has the following set of new features and bug fixes.

- **User-specified mount path For DB2 dSource**
  Users can specify the mount path of their choice on the target host to host the dSource dataset.
- **Staging push automation**
  A set of scripts is being provided that can be used to automate the steps related to the restore and rollforward operations on the dSource.

Release 4.0.1

The Db2 Plugin version 4.0.1 includes the below improvements.

- Added support for 11.1.4.6 fix pack.

Release 4.0.0

The Db2 Plugin version 4.0.0 includes the below improvements.

- Plugin generated log file will now be placed at `<toolkit path>/<Delphix_COMMON>/DB2_18f4ff11-b758-4bf2-9a37-719a22f5a4b8/logs/<instance name>`.
- More user-friendly log statements will be printed.
- The logging.properties and advanceConf.config configuration files are now merged into the db2_plugin.conf file.
- Dependency on redirect restore script to get the data path information is now removed.

Release 3.1.1

The Db2 Plugin version 3.1.1 introduces the support for full online backups, which are taken using the "exclude logs" syntax. This support is only applicable to the dSources created using the Backup and Logs ingestion mechanism on a single partition.

In Db2, users have the flexibility to exclude logs while taking the full online backups. The Db2 plugin will read the backup header from the user-provided backup file. The backup header is used to figure out whether the backup was taken using the "exclude logs" syntax or not. In case the plugin detects that it was taken using the "exclude logs" option, then the required changes will be made in the restore syntax.

Release 3.1.0

The Db2 Plugin version 3.1.0 introduces the **Staging Push** feature. This feature will allow users to control the restore of the staging database and they can also control the state of the restored staging database. With the **Staging Push** feature, the restore and rollforward operations will be managed by the user from the CLI. During dSource creation (or RESYNC) the plugin will create an ***empty mount point*** and will provide a restore and rollforward template that will help the user to create the redirect restore script. Once the dSource will be ready then the user can perform restore and rollforward operations from CLI. The restore must be as per the plugin provided template. After the restore, the staging database should remain in rollforward pending state, and prior to the next snapshot, the user can do a rollforward to a specific Point-in-Time (by providing a timestamp to rollforward command) against the available archive logs.

Release 3.0.2

This release addresses a limitation in Db2 wherein the VDB can accept the connections when refresh, disable and delete Delphix operations are in process which eventually can corrupt the VDBs. This plugin version will reduce such occurrences by introducing maintenance window in these operations during which the VDB will not accept external connections.

Release 3.0.1

The Db2 Plugin version 3.0.1 fixes the bug wherein the jobs on UI hung intermittently due to a known IBM Db2 bug.

Release 3.0.0

The Db2 Plugin version 3.0.0 introduces support to Db2 Database Partitioning Feature (DPF). This will help the customer to ingest the data to a staging dsource from a Db2 DPF enabled source.

- **Database partitioning feature**
  Database Partitioning Feature (DPF) lets you partition your database across multiple servers. Since you can add new machines and spread your database across them, this allows users to scale their database. This means more CPUs, more memory, and more disks from each of the additional machines are available for your database. DPF can be used to manage large databases for a variety of use cases including data warehousing, data mining, online analytical processing (OLAP), or online transaction processing (OLTP) workloads.

DPF enables the user to divide a database into database partitions, a database partition is a part of a database that consists of its own data, indexes, configuration files, and transaction logs. Each database partition can be configured on the different physical server having its own set of computing resources, including CPU and storage. When a query is processed, the request is divided so each database partition processes the rows that it is responsible for. DPF can maintain consistent query performance as the table grows by providing the capability to add more processing power in the form of additional database partitions. This capability is often referred to as providing linear scalability using Db2s shared-nothing architecture.
DPF is an approach to sizing and configuring an entire database system. Please follow the recommended practices for optimal performance, reliability, and capacity growth. Please refer to IBM documentation of DPF for more details in IBM knowledge center.

- **Pipelining logic for implementing parallel restores**
  The plugin employs a parallel pipeline methodology so that the restore operation of non-catalog partitions can be performed in parallel in the Database Partitioning Feature (DPF). The number of parallel restores is determined by the value of “restorePipelineLimit” (default value is 10) in <Toolkit directory>/advanceConfFlag.conf. The parameter “restorePipelineLimit” is configurable by end-users. The plugin performs parallel restore for all non-catalog partitions. For e.g. If the total number of non-catalog partitions is 15, and the "restorePipelineLimit" parameter is set to 10, the first set of 10 restores will happen in parallel. The plugin will track the restore of each partition present in the pipe. Whenever a restore of a partition completes, it will move out from the pipe and a new partition will enter into that pipe. Thus, the plugin ensures that the pipe will always have the user-configured number of partitions being restored (default=10).
- **Data synchronization**
  When “Customer Supplied Archive Logs” feature is used along with Db2 DPF feature, users will have to place the archive logs inside a folder with a name as NODE<Partition number> where <Partition number> is a four-digit number. For example, if the source environment has two partitions then the user-provided log path will have folder names NODE0000 and NODE0001. Both the folders will have respective archive logs. Snapshot operation will be used to apply the archive logs.
- **Advance configuration file**
  The advanced config file is created at <Toolkit directory>/advanceConfFlag.conf during discovery or environment refresh operation if the file does not pre-exist already. This file will have the following configurable parameters:
    - **Common group (*notCommonGroupFlag)***
        - This parameter sets the permission of "mnts", "logs" and "code" directories at <Toolkit dir>/DB2 location.
        - Set *notCommonGroupFlag to false,* if the primary group of the primary environment user (user which is used to do discovery) is shared with the instance users.
        - Set *notCommonGroupFlag to true,* if you don't want to share any group between the primary environment user and the instance users.
        - Once the necessary changes are made, refresh the environment in order to make the changes applicable.

  By default, the *notCommonGroupFlag* parameter is not specified in the file. This means that the plugin implicitly assumes its value as false. The valid values for this parameter are true or false.
    - **Allow source database on same instance (*allowSourceDBonSameInst*)**
      By default, the DB2 Plugin restricts the provisioning of a VDB on a DB2 instance which contains a database with the same name as the VDB's source database
      For example:
        - When provisioning a VDB from a VDB then source VDB is treated as the source database.
        - When provisioning a VDB from a dSource then a staging database is treated as the source database.

  Set *allowSourceDBonSameInst* to true, if the user wants to provision a VDB on an instance which contains a database with the same name as the VDB's source database. The valid values for this parameter are true or false.

- **Restore pipeline limit (*restorePipelineLimit)***
  Db2 DPF Plugin performs restoration of all non-catalog partitions in parallel.
  The parameter *restorePipelineLimit* allows the user to configure the number of partitions that should be restored in parallel. When the new advanceConfFlag.conf file is created during environment refresh or discovery operation, the default value of *restorePipelineLimit* is set to 10. If the advanceConfFlag.conf file is already present, the user will have to set the value of the `restorePipelineLimit` parameter to the desired value. The valid value for this field is any positive integer.

A default config file created during discovery or environment refresh operation: “

```
# If the user is not sharing a common group between primary environment user
and instance user.

# Then the user needs to set parameter notCommonGroupFlag as true.

# notCommonGroupFlag=true

# During provision operation check if instance contains a database name which
is identical to the source DB name used for provisioning operation.

# By default plugin will never allow creating a VDB on the instance where
source database (or other database which is identical in name with source DB)
already exists.

# If the user still wants to create a VDB on the instance which contains a
database name which is identical to the source DB name used for provisioning
operation,

# then the user needs to set parameter allowSourceDBonSameInst as true.

# allowSourceDBonSameInst=true

# Limit for parallel restores

restorePipelineLimit=10
```

Fixed issues (IBM Db2)

Release 4.6.1

Key	Summary
DB2-2190	Instead of failing the process, the housekeeping job should issue a warning.

Release 4.6.0

Key	Summary
DB2-2068	"VDB Name" requested to be entered twice during provisioning.
DB2-2117	Update LOGARCHCOMPR1 and LOARCHCOMPR2 to off during dSource creation
DB2-2155	db2greg should run from the latest installation of db2 on staging/target for environment discovery
DB2-2156	All DB2 dSources with Backup+logs should only rollforward to PIT
DB2-2178	MRT query fails with SQL1262N on staging hosts that are ahead of UTC timezone
DB2-2179	If DB2LOADREC registry value already set raise error.

Release 4.5.1

Key	Summary
DB2-2112	DB2 pre-snapshot phase runs rollforward to EOL on RF pit configuration

Release 4.5.0

There are no fixed issues in this release.

Release 4.4.2

Ticket number	Summary
DB2-2004	Some DB2 VDB deletes/refresh when canceled leave database running on instance in strange state

Release 4.4.1

Ticket number	Summary
DB2-2044	Db2 plugin shouldn’t implement Source sizing for VDB datasets
DB2-2045	VDB provisioning fails with SQl5153N due to LOGSECOND value
DB2-2053	Provisioning VDB failed during db2relocatedb due to multiple storage group paths

Release 4.4.0

Ticket number	Summary
DB2-2033	Resync fails on staging hosts with fuser command
DB2-2041	HADR linking is failing for low privilege user environment

Release 4.3.2

Ticket number	Summary
DB2-1682	Db2 vSDK plugin should support all product editions
DB2-1971	Environment addition and other workflows fail when .profile is configured with banners
DB2-1992	Restore on dpf environment continues even when one or more nodes' restore fails in dSource creation
DB2-1997	Db2 staging push doesn't test for missing tablespaces/datapaths while running db2dart

Release 4.3.1

Ticket number	Summary
DB2-1973	Make sure the data paths are configured under "db_mount/DB_NAME" in staging push

Release 4.3.0

Ticket number	Summary
DB2-1952	`remove_event_bin_files` may fail due to regular expression mismatch for DIAGPATH value

Release 4.2.1

There are no fixed issues in this release.

Release 4.2.0

Ticket number	Summary
DB2-1901	VDB provision from dsource with rollforward to PIT is running `rollforward to end of logs and stop` instead of `rollforward stop`
DB2-1911	DB2 rollforward command failing with syntax error for `dpf` environments
DB2-1913	The rollforward command should work with timezone only when timestamp is provided.
DB2-1920	VDB provision fails with `Failed to execute rollforward command on database BDN9NAJZ` in case of dsource with backup and log ingestion.

Release 4.1.3

Ticket number	Summary
DB2-1812	CLI operations during snapsync may fail if commands return error messages in a different locale
DB2-1856	VDB provision fails with an issue during db2relocatedb
DB2-1875	VDB behavior when we provision them using the first snapshot vs the latter ones
DB2-1921	Fix permissions for “/toolkit_dir/DB2” along with mnts/ and logs
DB2-1904	Staging push vSDK: snapsync operation on dSource fails with TypeError: unhashable type: ‘SnapshotDefinitionDataPaths’

Release 4.1.2

Ticket number	Summary
DB2-1826	Moved Staging Push redirect restore template out of the plugin and added contextual values to plugin logs
DB2-1796	Unable to get the complete content of redirect restore script in case the script size is more than 1MB
DB2-1824	DB2 VDB provisioning fails during relocateDB

Release 4.1.1

There are no fixed issues in this release.

Release 4.1.0

There are no fixed issues in this release.

Release 4.0.1

Key	Summary
DB2-1684	[Staging Push]Enable operation on dSource should generate the Restore and Rollforward template file if it is missing

Release 4.0.0

There are no fixed issues in this release.

Release 3.1.1

There are no fixed issues in this release.

Release 3.1.0

There are no fixed issues in this release.

Release 3.0.2

There are no fixed issues in this release.

Release 3.0.1

There are no fixed issues in this release.

Release 3.0.0

There are no fixed issues in this release.

Known issues (IBM Db2)

Release 4.6.1

Key	Summary	Workaround
DB2-2056	Upgrade from lua version to VSDK has issues with VDB Snapsync	Re-provisioning the database with the dSource's snapshot fixes the issue.
DB2-2124	If the storage paths are subdirectories of another storage path, the instance name at target and true prod cannot be the same.	Create instance name on target different than true production

Release 4.6.0

Key	Summary	Workaround
DB2-2056	Upgrade from lua version to VSDK has issues with VDB Snapsync	Re-provisioning the database with the dSource's snapshot fixes the issue.
DB2-2124	If the storage paths are subdirectories of another storage path, the instance name at target and true prod cannot be the same.	Create instance name on target different than true production

Release 4.5.1

Key	Summary	Workaround
DB2-2107	db2ckbkp fails for Backup paths with spaces	Remove blank spaces from backup paths
DB2-2124	The instance name at target and true prod cannot be the same if the storage paths are sub directories of another storage path.	Create instance name on target different than true production

Release 4.5.0

Key	Summary	Workaround
DB2-2107	db2ckbkp fails for Backup paths with spaces	Remove blank spaces from backup paths

Release 4.4.2

There are no known issues in this release.

Release 4.4.1

There are no known issues in this release.

Release 4.4.0

There are no known issues in this release.

Release 4.3.2

There are no known issues in this release.

Release 4.3.1

There are no known issues in this release.

Release 4.3.0

There are no known issues in this release.

Release 4.2.1

There are no known issues in this release.

Release 4.2.0

There are no known issues in this release.

Release 4.1.3

There are no known issues in this release.

Release 4.1.2

There are no known issues in this release.

Release 4.1.1

There are no known issues in this release.

Release 4.1.0

There are no known issues in this release.

Release 4.0.1

There are no known issues in this release.

Release 4.0.0

There are no known issues in this release.

Release 3.1.1

There are no known issues in this release.

Release 3.1.0

There are no known issues in this release.

Release 3.0.2

There are no known issues in this release.

Release 3.0.1

There are no known issues in this release.

Release 3.0.0

Key	Summary	Workaround / Comments
DB2-1318	Jobs on UI might hang intermittently due to a known IBM Db2 bug	This issue will be only observed when multiple datasets jobs, within the same Db2 instance, will get executed at the same time.

MySQL release notes

This section covers the following topics:

- New features (MySQL)
- Fixed issues (MySQL)

To view the MySQL support matrix, see MySQL matrix.

New features (MySQL)

Release 4.1.0

This connector version supports blocking of creation/rollback and refresh of VDB on target environment when MySQL version installed is not the same as the MySQL version installed on the environment of dSource.

> To install the connector, refer to MySQL Connector Installation.

Release 4.0.0

This connector version supports MySQL v8 across all certified editions.

Release 3.0.0

This connector version has the following new features:

- The MySQL connector version 3.0.0 has been enhanced for improved security and stability. We have added fixes for the following:
  - Dataset creation workflows can detect if the provided port number is already in use by an existing MySQL instance and notify the user.
  - Corrected the reported dSource and VDB statuses.
  - Fixed the **Snapshot with Parameters** functionality of the Manual Backup Ingestion to work smoothly and without errors.
  - Improved UI descriptions, UI and connector validations, and error messages.
- Added support for Percona Server for MySQL version 5.7 using Manual Backup Ingestion and Binary Log Replication ingestion models.
- Integrated support for MySQL and Percona Server for MySQL version 5.7 in a consolidated release stream. If you have installed divergent or custom branches of the MySQL connector you should look to migrate to this production stream. Future features and bugs will be resolved here.
- Implemented various security fixes.

Release 2.1.0

The MySQL connector 2.1.0 release consists of security improvements.

Release 2.0.26

This connector version has the following new features:

- Selecting Databases: Ability to select specific databases in the Source MySQL server when creating a dSource.
- Delphix generated config file: If a `my.cnf` file is not provided while creating a dSource, Delphix now creates one.
- New Exit Return Codes: MySQL connector 2.0.26 introduces more granular error exit codes.

> Delphix began supporting MySQL connector version from v2.0.26.

Fixed issues (MySQL)

Release 4.1.0

Ticket number	Summary
MYSQL-118	The MySQL Instance name is changed to Source Config name to reduce confusion during configuration and standardize across our connectors.

Release 4.0.0

Ticket number	Summary
MYSQL-138	The length of the MySQL Instance name has now been changed to support up to 255 characters.
MYSQL-81	Changed the command to change the root password which facilitates seamless working of the linking process.
MYSQL-72	Changed the commands to take backups as they were deprecated.

Release 3.0.0

There are no fixed issues in this release.

Release 2.1.0

There are no fixed issues in this release.

Release 2.0.26

There are no fixed issues in this release.

Oracle E-Business Suite (EBS) 12.2 release notes

This section covers the following topics:

- New features (EBS)
- Fixed issues (EBS)
- Known issues (EBS)

To view the EBS support matrix, see Oracle E-Business Suite (EBS) matrix.

New features (EBS)

Release 4.3.0

This plugin release has the following set of new features.

- **EBS Intelligent AppsTier linking**
  This will remove the manual intervention while you are performing cutover from RUN FS(fs1) to PATCH FS(fs2) and now new PATCH FS(fs1), new RUN FS(fs2) and you don't have to make any changes in dSource configuration so linking will work seamlessly.
  With 4.3.0 or later, we do not need to specify the "relative paths" of files to exclude in the Paths to Exclude list. Delphix Engine will automatically detect the RUN FS.
- **EBS HealthChecker utility**
  The HealthChecker will perform various pre-requisite checks reporting to ensure OS libraries, utilities, OS user permissions, oracle central inventory, OS resources (CPU, RAM, swap space, storage) etc. are as per Oracle vendor EBS sizing guidelines. if anything is missing, it will be reported and must be fixed before Delphix Virtualization workflows are triggered by end-user
  The HealthChecker ensures Delphix's linking, provision, and refreshes of EBS workflows run smoothly.
- **Performance improvement**
  The latest 4.3.0 plugin version has an improved performance, due to the removal of explicit status checks of WLS Managed servers during the virtual AppsTier provision and refresh phase.
- **Improvement in the re-run of failed VDB provision or refresh workflows**
  With this improvement, you don't have to delete a failed VDB. Instead, you can disable the VDB with the force option, clean up the target server after receiving the 'stale file handle' and stale processes problems, and then re-try the provision or refresh workflow.

Release 4.2.2

This plugin release consists of only bug fixes.

Release 4.2.1

This plugin release consists of only bug fixes.

Release 4.2.0

This plugin release has the following set of new features.

- **EBS database port customization**
  This feature supports the provisioning of VDBs in EBS dbTechStack & AppsTier plugins to run on a customized port. Using this feature you can provision database binaries(ORACLE_HOME) cloning and listener process to run on a custom DB port and Application tier to connect to that port. The EBS Database Port Customization provide flexibility to choose the desired port for running the Delphix-created VDBs & listeners.

> ⓘ This feature is supported only in Oracle 19c Database.

Release 4.1.0

The EBS Plugin version 4.1.0 introduces the hooks utility feature and also addresses the handling of an additional prompt for ISG Datasource password during AppsTier provisioning operation for `adcfgclone` execution.

- **Hooks utility feature**
  This feature provides flexibility to end-users where they can provide a command with configure/pre-

snapshot as an option in configuring and pre-snapshot hooks during DB provisioning. This is a huge improvement in terms of user experience in comparison to the conventional approach where end-users had to add entire content (by copying it from the user documentation) into the various hooks. The hooks utility also manages the logs for hooks execution; it'll be helpful in scenarios where end users want to track the hook execution. We have not removed or restricted the existing approach of providing hooks content from user documentation, the hooks utility can be used as an alternative to the existing approach.

Release 4.0.2

This plugin release consists of only bug fixes.

Release 3.0.3

This plugin release consists of only bug fixes.

Release 3.0.2

This is a patch release that aims to provide the following updates to EBS 12.2 plugin.

- Updated Hooks for 19c MT low and high privileged users.
- Fixed an issue where 19c DBTechStack provision fails during `adcfgclone` with java exception.

Release 3.0.1

This is a patch release that aims to provide the following updates to EBS 12.2 plugin.

- Added support for RHEL 8.3
- Updated Hooks for 19c MT low and high privileged users.

Release 3.0.0

EBS Plugin version 3.0.0 aims to support Oracle 19c MT architecture. Moreover, a new field has been added on the provisioning UI. This will provide the flexibility to the end-user to provide the maximum time allowed to virtualize the application tier.

Release 2.0.1

EBS Plugin version 2.0.1 aims to successfully stop application services while provisioning or refresh operation. Hence, creating successful snapshots.

Release 2.0.0

EBS Plugin version 2.0.0 aims to implement Sudo Privileges feature for both EBS 12.2 Database Tier and Application Tier.

Release 1.8.4

EBS Plugin version 1.8.4 aims to implement Weblogic password complexity validation as per the Oracle standards.

Release 1.8.3

EBS Plugin version 1.8.3 introduces a new feature while refreshing EBS appsTier where a Pre Refresh hook will help in aborting concurrent manager services and reduce the time taken to gracefully stop the EBS application.. The password required to execute the same will be pulled from a text file present on the target appsTier host which will be created automatically during appsTier provisioning.

Release 1.8.2

EBS Plugin version 1.8.2 introduces a new feature where the user will not need to specify the source apps schema password in EBS DB hooks provision wizard. This password can now be pulled from a text file present on the target DB host which will be created automatically during DBTechStack provisioning.

Fixed issues (EBS)

Release 4.3.0

Ticket number	Summary
EBS-1319	In DBTECHSTACK provision, Plugin should not run adpreclone.pl dbTechStack command
EBS-1312	WLS password is saved in updatedspassword file
EBS-1255	Unable to refresh a failed provision
EBS-874	Detach oracle home and delete apps password file during unconfigure/delete operation and cleanup before provision

Release 4.2.2

Ticket number	Summary
EBS-1292, HF-1225	Plugin was using wrong Autoconfig script location if `s_jdbc_connect_descriptor_generation` was not true in CONTEXT_FILE
EBS-1280	Strong password check for WLS password as per EBS requirement
EBS-1169	Source apps password file removed from all remote hosts to improve security.
EBS-1285	Implementation of AppData Bundle Redaction parameters in EBS plugin
EBS-1256	Expect utility availability is checked on additional application nodes for smooth multi-node provisioning.

Release 4.2.1

Ticket number	Summary
EBS-1265	Hooks utility fails to get the short hostname if target server is SunOS SOLARIS
EBS-1278	Handling detach ORACLE HOME when inventory.xml has multiple HOMES

EBS-1259	If someone is upgrading from Lua(2.0.x) to vSDK version(4.x.x) and using oracle 11g, the end user will not be able to take snapshot in Delphix Engine
EBS-1275	Fix lsof issue in Solaris and Linux

Release 4.2.0

There are no fixed issues in this release.

Release 4.1.0

There are no fixed issues in this release.

Release 4.0.2

There are no fixed issues in this release.

Release 3.0.3

Key	Summary
EBS-1027	EBS 12.2 Plugin will validate user-provided ORACLE_HOME during DbTier provisioning
EBS-1028	Enhancements identified in dbTechStack code
EBS-1038	Fix shellchecks warnings/errors in hooks and sudoers entry for target dbtier
EBS-1066	Make the check stricter for listener process in DBTechStack status module and fix shellchecks

Release 3.0.2

Key	Summary
EBS-890	19C DBTechStack provision fails during adcfgclone with java exception

Release 3.0.1

There are no fixed issues in this release.

Release 3.0.0

Key	Summary
EBS-767	EBS appsTier linking failed while identifying Delta patch level information from low-privilege OS user

Key	Summary
EBS-779	Continued timeouts - toolkit.ebs122-app-1-8-2.errorTimeout {admanagedsrvctl}
EBS-787	Toolkit should perform prior dependency/pre-requisite check in target host before starting to provision dBtechStack and Appstier
EBS-788	Error not being shown to the customer
EBS-798	GUI shows vfiles ORACLE_HOME as stopped however it is not
EBS-778	Expect timeout set to 8 hours which is too short
EBS-798	GUI shows vfiles ORACLE_HOME as stopped however it is not
EBS-816	EBS 12.2 disable Appstier vdb failed with Delphix cannot stop all processes under /u01/oracle/VIS on host

Release 2.0.1

Key	Summary
EBS-783	Provisioning VDB from Appstier dSource fails with adpreclone connection with Oracle WebLogic Administration Server

Release 2.0.0

Key	Summary
EBS-757	EBS plugin should check for data accessibility during status check.

Release 1.8.4

Key	Summary
EBS-712	WLS Password Complexity Issue - Merge ESCL_2983 into EBS 12.2 Plugin.
EBS-713	Multi-line support for SQL query O/P - merge ESCL-2988 into EBS 12.2 Plugin.

Release 1.8.3

Key	Summary
EBS-663	Abort EBS ConcMgr services before stopping EBS appsTier services during refresh.

Release 1.8.2

Key	Summary
EBS_566	Display port pool number in the error toolkit.ebs122-app-1-7-0.portPoolNotFreeError.
EBS-597	Delphix Hooks Password-Storing Apps password to file for EBS appsTier

Known issues (EBS)

Release 4.3.0

There are no known issues in this release.

Release 4.2.2

There are no known issues in this release.

Release 4.2.1

There are no known issues in this release.

Release 4.2.0

There are no known issues in this release.

Release 4.1.0

There are no known issues in this release.

Release 4.0.2

There are no known issues in this release.

Release 3.0.3

There are no known issues in this release.

Release 3.0.2

There are no known issues in this release.

Release 3.0.1

Key	Summary
EBS-836	"ebs_<PDB_SID>" listener services are getting disappeared on VDB stop/start. As a workaround, a new Third Configure Clone Hook has been added.

Release 3.0.0

Key	Summary
EBS-836	"ebs_<PDB_SID>" listener services are getting disappeared on VDB stop/start. As a workaround, a new configure clone hook has been added.

Release 2.0.1

There are no known issues in this release.

Release 2.0.0

There are no known issues in this release.

Release 1.8.4

There are no known issues in this release.

Release 1.8.3

Key	Summary
EBS-692	EBS link operation fails when source EBS environment is linked with BYOOJ
DLPX-69617	OperationRunnerImpl incorrectly assumes toolkit JDK always exists

Release 1.8.2

Key	Summary
EBS-692	EBS link operation fails when source EBS environment is linked with BYOOJ
DLPX-69617	OperationRunnerImpl incorrectly assumes toolkit JDK always exists

PostgreSQL release notes

This section covers the following topics:

- New features (PostgreSQL)
- Fixed issues (PostgreSQL)
- Known issues (PostgreSQL)

To view the PostgreSQL support matrix, see PostgreSQL matrix.

New features (PostgreSQL)

Release 4.1.1

This plugin release consists of only bug fixes.

Release 4.1.0

This plugin release has the following set of new features.

- **Source sizing for dSource with ZFS**
  With PostgreSQL Plugin 4.1.0 and onwards, dSource database size will be calculated using `du` command on dSource mount path. For more information, see Source sizing implementation for dSource.
- **Azure Database for PostgreSQL - Flexible Server Certification**
  Added support for the PaaS-to-IaaS ingestion model of Azure Flexible Server 11, 12, 13, and 14 versions.

Release 4.0.0

This plugin release has the following set of new features.

- **Provisioning a PostgreSQL VDB PiT with external logs**
  With the latest PostgreSQL Plugin 4.0.0 and onwards, you can provision a VDB to a point in time by providing the WAL logs on the target host. For detailed information, see Provisioning a PostgreSQL VDB PiT with External Logs.
- **pg_dumpall support for single database ingestion**
  PostgreSQL plugin 4.0.0, supports `pg_dumpall` which has been added with Single Database Ingestion to get the global objects like roles and tablespaces.

Release 3.2.0

This plugin release has the following set of new features.

- **Enhancement to the external backup mechanism**
  With the 3.2.0 release, while using External Backup Mechanism it is now **not** mandatory for you to provide the backups as ZIP files. This enhancement thus removes the mandatory requirement of `pg_basebackup` to be encapsulated in `PostgreSQL_T<YYYYMMDD>.zip format`. You can now store the backups (as tars) on the staging server directly, without providing the backups as ZIP files (reducing the time spent on creating ZIP files) using the following additional ways:
  - Within folders (named `PostgreSQL_T<YYYYMMDDhhmmss>`) created inside the provided backup path.
  - Within the provided backup path - `PostgreSQL Backup File Path`

  Switching to the new mechanism is easy, just store the full backup within the same backup path and run a resync operation.
  The plugin still supports the usage of zip files.
- **Introducing a LOG_LEVEL parameter in the PostgreSQL plugin configuration**
  PostgreSQL plugin supports remote side logging, with all plugin logs available within the scratch path on the remote host (`<tookit_path>/Delphix_COMMON_*/plugin/postgres-vsdk_*`). You can tweak the log parameters related to log rotation within the plugin configuration file called `dlpx_postgres_config.ini`. The 3.2.0 version, introduces a new parameter called **LOG_LEVEL**, with the capability of handling the amount of logs being generated on the remote host. Accepted values are **DEBUG, INFO, WARNING, and ERROR**, with **INFO** being the default value.

Release 3.1.0

This plugin release has the following set of new features.

- **Source sizing implementation on PostgreSQL plugin**
  This feature enables you to check the size of PostgreSQL cluster or database being virtualized from the Delphix Engine. The calculation of the dataset size is based on the ingestion method used. For staging pull methods, if the whole cluster is ingested then the size of the cluster will be calculated or else the size of the ingested single database will be calculated. Source sizing cannot be implemented when staging push method is used.

⚠ This feature requires the plugin to run SQL queries on the datasets. If the plugin cannot login to the dataset for any reason, then source sizing cannot be implemented. For more details about this feature, see Source Sizing Implementation for dSource.

If you are using privilege elevation and upgrading to PostgreSQL 3.1.0 version, then you will have to change the sudoers entry. For more details see, Prerequisites for Privilege Elevation using DLPX_DB_EXEC Script.

Release 3.0.0

This plugin release has the following set of new features.

- **Staging push**
  The PostgreSQL Plugin 3.0.0 release introduces a new data ingestion mechanism i.e. Staging Push that enables you to push data into the Delphix-provided mount point on your own. It is an alternative Delphix dSource data ingestion approach that enables you to push the data in Delphix, which is different from the conventional pull model where Delphix pulls the data.
  Delphix relinquishes the responsibility to prepare the data (ie: recover to a specific state) on the staging push. It is different from conventional staging where Delphix controls the state of recovery of the staging database.
  There are two mechanisms to create dSource on the staging server.
    - Pull architecture: The Postgres plugin pulls the data into the Delphix-provided mount location.
    - Staging Push architecture: You push the data into the Delphix-provided mount point on your own.

  For more details, see Staging Push Implementation.

Release 2.1.1

This plugin release consists of only bug fixes.

Release 2.1.0

This plugin release has the following set of new features.

- **Privilege elevation**
  The Privilege Elevation feature allows you to use a low privileged OS user to create a dSource or VDB. To use this feature, you must provide a username in the **Privileged OS account (optional)** field on the dSource creation page. The plugin will then use the `dlpx_db_exec` script to perform the privilege elevation. The script has to be updated via the engine API before the creation/discovery of an environment.

Release 2.0.0

The PostgreSQL Plugin version 2.0.0 includes the following improvements and features:

- Plugin generated log file will now be placed at <toolkit path>/Delphix_COMMON_<long id>/plugin/postgres-vsdk_12c9a11f-fab6-4792-a5da-49d726c004df/<os user> instead of <toolkit path>/postgres/logs.
- Improved logging

- The plugin logs will be part of the support bundle.
- The plugin logs will be available on the target host as well as on the Delphix Engine host.
- Introduction of a new generic config ini file dlpx_postgres_config.ini which will be present at the scratch path. The previous log configure file dlpx_pg_logrotate.conf will no longer be available and the contents of this file will be moved to the generic config ini file-*dlpx_postgres_config.ini.*
- The PostgreSQL plugin will now have one more ingestion mechanism - Single Database Ingestion that allows you to virtualize a single database.

Release 1.5.1

This release includes the Streaming Replication Monitoring feature that monitors the streaming replication by reading the Postgres database logs every minute.

Release 1.5.0

This plugin release consists of only bug fixes.

Release 1.4.4

This plugin release consists of only bug fixes.

Release 1.4.3

This plugin release consists of only bug fixes.

Release 1.4.2

This plugin release consists of only bug fixes.

Release 1.4.1

This plugin release consists of only bug fixes.

Release 1.4.0

PostgreSQL Plugin version 1.4.0 contains the following certifications :

- Operating System - RHEL/CentOS 7.8 version Support with PostgreSQL 12.x.

Fixed issues (PostgreSQL)

Release 4.1.1

Key	Summary
POST-690	Incorrect error: The mount path has been changed from the UI
POST-936	def locate_files & def __find_repo_using_env_variable are not able to suppress the warning "Permission denied"

Release 4.1.0

Key	Summary
POST-720	Postgres staging and target database should use the same local as the source database
POST-85	The Plugin post-snapshot workflow breaking for non-english locales

Release 4.0.0

There are no fixed issues in this release.

Release 3.2.0

Ticket number	Summary
POST-293	Compressed backup and backup timestamp
POST-470	Pg_vSDK >> Plugin should add a parameter to control debug log ON/OFF in `dlpx_pg_logrotate.conf`
POST-497	Ingestion from backup should be more efficient
POST-699	Removing source sizing for VDB

Release 3.1.0

Ticket number	Summary
POST-596	New mechanism of creating files to remove wildcard entry for `echo` in sudoers file.

Ticket number	Summary
POST-611	Ubuntu Certification >> Linking is failing when pg_hba.conf and pg_ident.conf files are not at data directory
POST-642	dSource Single DB Ingestion >> User should not be allowed to change database name parameter, unless it's a RESYNC operation
POST-668	Staging Push >> Failure to provision a VDB using privilege elevation from Staging Push snapshot with provisioning allowed

Release 3.0.0

Ticket number	Summary
POST-537	`postgresql.auto.conf` and `recovery.signal` config files needs to be handled in VDB creation
POST-546	`logging_collector` should be enabled for the dSource
POST-563	SP >> Snapshot operation is failing if the port value is quoted with the strings in config file
POST-565	PG 2.1.1 >> Variable value is missing in error
POST-566	Time difference issue between Delphix Engine and Staging Environment Host
POST-571	Environment variables between low-privileged user & high-privileged user are not translated

Release 2.1.1

Ticket number	Summary
POST-555	Logging of large content is failing at staging/target side
POST-562	Echo command in .bash_profile of the user blocks several Plugin operations

Release 2.1.0

Ticket number	Summary
POST-494	Using dlpx_db_exec for privilege elevation
POST-511	`max_connections` parameter on staging should be set according to the source
POST-518	`netstat` utility is not available in SuSE 15

Release 2.0.0

PostgreSQL version 2.0.0 contains the following bug fixes and enhancements:

Ticket number	Summary
POST-300	pg_basebackup failure not captured
POST-303	Default value for PostgreSQL Mount Path is wrong
POST-306	Delphix GUI allows PostgreSQL VDB port to be changed while the VDB is running
POST-422	Snapshots created for dSource despite failure to start staging database
POST-428	Instance failure reason not captured
POST-431	Unable to provision PostgreSQL VDB with parameters that use the strftime specification formatting
POST-441	Postgres dsource creation with existing backup fails due to missing wal file

Release 1.5.1

Ticket number	Summary
POST-418	Replication monitoring - Lua plugin

Release 1.5.0

Ticket number	Summary
POST-202	If the port is not present in postgresql.conf on the source, it's not provided in staging and VDBs
POST-283	Security enhancement - Plugin overrides listen_addresses parameter in posgresql.conf to "localhost" on staging & target server
POST-292	dSource configuration parameters don't accept empty values
POST-295	The staging server is created using a copy of the production parameters file
POST-297	Postgres Plugin Should check if the toolkit directory is accessible by the OS user
POST-298	primary_conninfo' parameter is not getting commented out in vdb flow
POST-302	Allow users to set empty values for a parameter in a configuration file
POST-312	delphix_postgres_debug.log is too large to manage
POST-316	Propagation of Hotfix 1112 code changes to plugin 1.5.0 release

Release 1.4.4

Key	Summary
POST-291	Error Handling for Authentication Failure in `pg_basebackup` command

Release 1.4.3

Key	Summary
POST-284	The plugin doesn't check if the staging Postgres database is in a stable state or not

Release 1.4.2

Key	Summary
POST-274	`data_directory` parameter in the postgresql.conf file is causing failure in dSource and VDB flow

Key	Summary
POST-275	Linking Fails if the parameter which needs to be configured is commented through multiple `#`
POST-276	Linking Fails if the mandatory parameter required to setup replication is commented in config file

Release 1.4.1

Key	Summary
POST-269	Postgres DB_SYNC fails - ESCL-3179
POST-270	Plugin should handle postgresql.auto.conf entries during dsource creation

Release 1.4.0

There are no fixed issues in this release.

Known issues (PostgreSQL)

Release 4.1.1

There are no known issues in this release.

Release 4.1.0

There are no known issues in this release.

Release 4.0.0

There are no known issues in this release.

Release 3.2.0

There are no known issues in this release.

Release 3.1.0

There are no known issues in this release.

Release 3.0.0

There are no known issues in this release.

Release 2.1.1

There are no known issues in this release.

Release 2.1.0

There are no known issues in this release.

Release 2.0.0

There are no known issues in this release.

Release 1.5.1

There are no known issues in this release.

Release 1.5.0

There are no known issues in this release.

Release 1.4.4

There are no known issues in this release.

Release 1.4.3

There are no known issues in this release.

Release 1.4.2

There are no known issues in this release.

Release 1.4.1

There are no known issues in this release.

Release 1.4.0

There are no known issues in this release.

SAP HANA release notes

This section covers the following topics:

- New features (SAP HANA)
- Fixed issues (SAP HANA)
- Known issues (SAP HANA)

To view the SAP HANA support matrix, see SAP HANA matrix.

New features (SAP HANA)

Release 6.3.2

This plugin release consists of only bug fixes.

Release 6.3.1

This plugin release consists of only bug fixes.

Release 6.3.0

This plugin release has the following set of new features.

- **Source sizing implementation on HANA plugin**
  The SAP HANA 2.0 Plugin version 6.3.0 introduces the source sizing feature, that aligns the data source experience with Delphix database standards which will improve your ingestion reporting experience on the per TB pricing model. This feature enables you to calculate the dataset size based on the total allocated space on the mount point provided by the Delphix Engine for the dataset. Source sizing is not implemented for VDB.

Release 6.2.1

The SAP HANA 2.0 Plugin version 6.2.1 validates the SAP HANA's Staging Push solution with third party backup vendors.

Release 6.2.0

The SAP HANA 2.0 Plugin version 6.2.0 introduces the **Staging Push** feature. This feature provides a new data ingestion mechanism that helps users to push data into the Delphix provided mount point on their own.

Release 6.1.1

This patch release contains an enhancement that allows the `create_backup_metadata.sh` script to be executed in a non-interactive mode as well.

Release 6.1.0

This plugin release has the following set of new features.

- **SAP HANA port control**
  With this release, a solution has been introduced that keeps the port numbers consistent throughout the VDB life cycle so that the connections made to the VDBs are not disrupted during their life cycle.

Release 6.0.1

This plugin release consists of only bug fixes.

Release 6.0.0

This is a conversion from SAP HANA direct plugin to vSDK-based, staged plugin. This plugin will support only customer-provided/External backups ingestion mechanism.

Release 4.3.2

This plugin release consists of only bug fixes.

Release 4.3.1

This plugin release consists of only bug fixes.

Release 4.3.0

Below features are being provided:

1. **Non-ingestion of backups into Delphix**: This means that database backups will not be transferred into the Delphix Engine during any of the operation. This will be driven on the basis of a checkbox that can be selected by the customer during dSource creation.
2. Support provided for log backups along with the data backups (only when non-ingestion has been chosen during dSource creation).
3. Ability to use Delphix provided storage for all virtualized tenant services. This is valid even for the user-created services outside the plugin.
4. Multiple HANA installations on the same VM will also be supported.
5. Support for Self-Service (Jetstream) has been certified with this release.
6. Certification of plugin support with HANA Standard and Enterprise Edition 2.0 SP05 on RHEL 8.1.

Release 4.2.0

With this release, we are including the support for basepath_databackup parameter. Backups created via plugin will now be created on the PATH specified by this parameter. This parameter can be defined at the system and at the tenant level. If this parameter has not been specified then default location will be used for creating the backups.

Release 4.1.2

This plugin release consists of only bug fixes.

Release 4.1.1

This plugin release consists of only bug fixes.

Fixed issues (SAP HANA)

Release 6.3.2

Delphix ticket number	Summary
CTCHANA-1936	Handling services not having volumes information in m_volumes for VDB creation.

Release 6.3.1

Delphix ticket number	Summary
CTCHANA-1914	Refining the plugin discovery logic to search for HANA instances.

Release 6.3.0

Delphix ticket number	summary
CTCHANA-1906	Changing the plugin's environment discovery logic to refer "HDB --version" instead of "landscapeDescription.xml"
CTCHANA-1900	Removal of tenant parameters from UI and plugin
CTCHANA-1891	Incorrect execution time for VDB

Release 6.2.1

There are no fixed issues in this release.

Release 6.2.0

There are no fixed issues in this release.

Release 6.1.1

Delphix ticket number	Summary
CTCHANA-1661	v6.0.0.0 create_backup_metadata.sh needs ability to run non-interactively
CTCHANA-1667	HANA Recovery failure on staging host yields difficult to decipher error

Release 6.1.0

Delphix ticket number	Summary
CTCHANA-821	HANA SQL Port value should remain consistent for the life of VDB.

Release 6.0.1

There are no fixed issues in this release.

Release 6.0.0

There are no fixed issues in this release.

Release 4.3.2

Delphix ticket number	Summary
CTCHANA-1666	Backporting the password change through hooks issue to the master branch
CTCHANA-1660	remove dependency on /etc/SuSE-release for suse o/s platforms

Release 4.3.1

There are no fixed issues in this release.

Release 4.3.0

There are no fixed issues in this release.

Release 4.2.0

There are no fixed issues in this release.

Release 4.1.2

There are no fixed issues in this release.

Release 4.1.1

There are no fixed issues in this release.

Known issues (SAP HANA)

Release 6.3.2

Delphix ticket number	Summary	Workaround/Comments
DLPX-75383	Refresh/Disable not working after failed delete operation	Force delete the VDB. Manually remove the mount point
DLPX-75385	Undo Refresh operation after a failed Refresh job leads to 502 Bad gateway and management stack restart	None
DLPX-67911	Failed enable operation leaving a mount point and next delete operation creating a stale file handle	Manually remove the mount point
CTCHANA-1599	Mount Point change not detected by the plugin may lead to multiple mounts	Mount point not in use can be removed manually
CTCHANA-1598	Plugin loses control in case of user cancel operation.	Manual cleanup can be done in case the plugin leaves any tenant database/ files on the VM
CTCHANA-1586	Refresh after a Force disable on a VDB fails with 'mount not empty' error	Manually remove the mount before proceeding with the Refresh
CTCHANA-1565	Stop/Start operation on a VDB post failed post_refresh hook fails with inappropriate error	None
CTCHANA-1514	dSource tenant database can be accessed and written onto	None
CTCHANA-1610	'test_force_disable_enable_vdb' fails across multiple images	None

Release 6.3.1

Delphix ticket number	Summary	Workaround/Comments
DLPX-75383	Refresh/Disable not working after failed delete operation	Force delete the VDB. Manually remove the mount point

Delphix ticket number	Summary	Workaround/Comments
DLPX-75385	Undo Refresh operation after a failed Refresh job leads to 502 Bad gateway and management stack restart	None
DLPX-67911	Failed enable operation leaving a mount point and next delete operation creating a stale file handle	Manually remove the mount point
CTCHANA-1599	Mount Point change not detected by the plugin may lead to multiple mounts	Mount point not in use can be removed manually
CTCHANA-1598	Plugin loses control in case of user cancel operation.	Manual cleanup can be done in case the plugin leaves any tenant database/ files on the VM
CTCHANA-1586	Refresh after a Force disable on a VDB fails with 'mount not empty' error	Manually remove the mount before proceeding with the Refresh
CTCHANA-1565	Stop/Start operation on a VDB post failed post_refresh hook fails with inappropriate error	None
CTCHANA-1514	dSource tenant database can be accessed and written onto	None
CTCHANA-1610	'test_force_disable_enable_vdb' fails across multiple images	None

Release 6.3.0

Delphix ticket number	Summary	Workaround/Comments
DLPX-75383	Refresh/Disable not working after failed delete operation	Force delete the VDB. Manually remove the mount point
DLPX-75385	Undo Refresh operation after a failed Refresh job leads to 502 Bad gateway and management stack restart	None
DLPX-67911	Failed enable operation leaving a mount point and next delete operation creating a stale file handle	Manually remove the mount point

Delphix ticket number	Summary	Workaround/Comments
CTCHANA-1599	Mount Point change not detected by the plugin may lead to multiple mounts	Mount point not in use can be removed manually
CTCHANA-1598	Plugin loses control in case of user cancel operation.	Manual cleanup can be done in case the plugin leaves any tenant database/files on the VM
CTCHANA-1586	Refresh after a Force disable on a VDB fails with 'mount not empty' error	Manually remove the mount before proceeding with the Refresh
CTCHANA-1565	Stop/Start operation on a VDB post failed post_refresh hook fails with inappropriate error	None
CTCHANA-1514	dSource tenant database can be accessed and written onto	None
CTCHANA-1610	'test_force_disable_enable_vdb' fails across multiple images	None

Release 6.2.1

Delphix ticket number	Summary	Workaround/Comments
DLPX-75383	Refresh/Disable not working after failed delete operation	Force delete the VDB. Manually remove the mount point
DLPX-75385	Undo Refresh operation after a failed Refresh job leads to 502 Bad gateway and management stack restart	None
DLPX-67911	Failed enable operation leaving a mount point and next delete operation creating a stale file handle	Manually remove the mount point
CTCHANA-1599	Mount Point change not detected by the plugin may lead to multiple mounts	Mount point not in use can be removed manually
CTCHANA-1598	Plugin loses control in case of user cancel operation.	Manual cleanup can be done in case the plugin leaves any tenant database/files on the VM

Delphix ticket number	Summary	Workaround/Comments
CTCHANA-1586	Refresh after a Force disable on a VDB fails with 'mount not empty' error	Manually remove the mount before proceeding with the Refresh
CTCHANA-1565	Stop/Start operation on a VDB post failed post_refresh hook fails with inappropriate error	None
CTCHANA-1514	dSource tenant database can be accessed and written onto	None
CTCHANA-1610	'test_force_disable_enable_vdb' fails across multiple images	None

Release 6.2.0

Delphix ticket number	Summary	Workaround/Comments
DLPX-75383	Refresh/Disable not working after failed delete operation	Force delete the VDB. Manually remove the mount point
DLPX-75385	Undo Refresh operation after a failed Refresh job leads to 502 Bad gateway and management stack restart	None
DLPX-67911	Failed enable operation leaving a mount point and next delete operation creating a stale file handle	Manually remove the mount point
CTCHANA-1599	Mount Point change not detected by the plugin may lead to multiple mounts	Mount point not in use can be removed manually
CTCHANA-1598	Plugin loses control in case of user cancel operation.	Manual cleanup can be done in case the plugin leaves any tenant database/files on the VM
CTCHANA-1586	Refresh after a Force disable on a VDB fails with 'mount not empty' error	Manually remove the mount before proceeding with the Refresh
CTCHANA-1565	Stop/Start operation on a VDB post failed post_refresh hook fails with inappropriate error	None

Delphix ticket number	Summary	Workaround/Comments
CTCHANA-1514	dSource tenant database can be accessed and written onto	None
CTCHANA-1610	'test_force_disable_enable_vdb' fails across multiple images	None

Release 6.1.1

Delphix ticket number	Summary	Workaround/Comments
DLPX-75383	Refresh/Disable not working after failed delete operation	Force delete the VDB. Manually remove the mount point
DLPX-75385	Undo Refresh operation after a failed Refresh job leads to 502 Bad gateway and management stack restart	None
DLPX-67911	Failed enable operation leaving a mount point and next delete operation creating a stale file handle	Manually remove the mount point
CTCHANA-1599	Mount Point change not detected by the plugin may lead to multiple mounts	Mount point not in use can be removed manually
CTCHANA-1598	Plugin loses control in case of user cancel operation.	Manual cleanup can be done in case the plugin leaves any tenant database/files on the VM
CTCHANA-1586	Refresh after a Force disable on a VDB fails with 'mount not empty' error	Manually remove the mount before proceeding with the Refresh
CTCHANA-1565	Stop/Start operation on a VDB post failed post_refresh hook fails with inappropriate error	None
CTCHANA-1514	dSource tenant database can be accessed and written onto	None
CTCHANA-1610	'test_force_disable_enable_vdb' fails across multiple images	None

Release 6.1.0

Delphix ticket number	Summary	Workaround/Comments
DLPX-75383	Refresh/Disable not working after failed delete operation	Force delete the VDB. Manually remove the mount point
DLPX-75385	Undo Refresh operation after a failed Refresh job leads to 502 Bad gateway and management stack restart	None
DLPX-67911	Failed enable operation leaving a mount point and next delete operation creating a stale file handle	Manually remove the mount point
CTCHANA-1599	Mount Point change not detected by the plugin may lead to multiple mounts	Mount point not in use can be removed manually
CTCHANA-1598	Plugin loses control in case of user cancel operation.	Manual cleanup can be done in case the plugin leaves any tenant database/files on the VM
CTCHANA-1586	Refresh after a Force disable on a VDB fails with 'mount not empty' error	Manually remove the mount before proceeding with the Refresh
CTCHANA-1565	Stop/Start operation on a VDB post failed post_refresh hook fails with inappropriate error	None
CTCHANA-1514	dSource tenant database can be accessed and written onto	None
CTCHANA-1610	'test_force_disable_enable_vdb' fails across multiple images	None

Release 6.0.0

Delphix ticket number	Summary	Workaround/Comments
DLPX-75383	Refresh/Disable not working after failed delete operation	Force delete the VDB. Manually remove the mount point
DLPX-75385	Undo Refresh operation after a failed Refresh job leads to 502 Bad gateway and management stack restart	None

Delphix ticket number	Summary	Workaround/Comments
DLPX-67911	Failed enable operation leaving a mount point and next delete operation creating a stale file handle	Manually remove the mount point
CTCHANA-1599	Mount Point change not detected by the plugin may lead to multiple mounts	Mount point not in use can be removed manually
CTCHANA-1598	Plugin loses control in case of user cancel operation.	Manual cleanup can be done in case the plugin leaves any tenant database/ files on the VM
CTCHANA-1586	Refresh after a Force disable on a VDB fails with 'mount not empty' error	Manually remove the mount before proceeding with the Refresh
CTCHANA-1565	Stop/Start operation on a VDB post failed post_refresh hook fails with inappropriate error	None
CTCHANA-1514	Dsource tenant database can be accessed and written onto	None
CTCHANA-821	HANA SQL Port value should remain consistent for the life of VDB.	None
CTCHANA-1610	'test_force_disable_enable_vdb' fails across multiple images	None

Release 6.0.0

Delphix ticket number	Summary	Workaround/Comments
DLPX-75383	Refresh/Disable not working after failed delete operation	Force delete the VDB. Manually remove the mount point
DLPX-75385	Undo Refresh operation after a failed Refresh job leads to 502 Bad gateway and management stack restart	None
DLPX-67911	Failed enable operation leaving a mount point and next delete operation creating a stale file handle	Manually remove the mount point

Delphix ticket number	Summary	Workaround/Comments
CTCHANA-1599	Mount Point change not detected by the plugin may lead to multiple mounts	Mount point not in use can be removed manually
CTCHANA-1598	Plugin loses control in case of user cancel operation.	Manual cleanup can be done in case the plugin leaves any tenant database/ files on the VM
CTCHANA-1586	Refresh after a Force disable on a VDB fails with 'mount not empty' error	Manually remove the mount before proceeding with the Refresh
CTCHANA-1565	Stop/Start operation on a VDB post failed post_refresh hook fails with inappropriate error	None
CTCHANA-1514	Dsource tenant database can be accessed and written onto	None
CTCHANA-821	HANA SQL Port value should remain consistent for the life of VDB.	None

Release 4.3.2

Delphix ticket number	Summary	Workaround/Comments
HANA-821	HANA SQL Port value should remain consistent for the life of VDB.	None
DLPX-44359	Job cancellation may fail.	None
CTCHANA-930	Upgrade from 3.4.5 to 3.5.0/3.5.1/4.0.0/4.1.1/4.1.2/4.2.0 needs a manual deletion of the existing tenant databases for disable/enable to work.	Drop the tenant database manually before enabling the VDB.

Release 4.3.1

Delphix Ticket Number	Summary	Workaround/Comments
HANA-821	HANA SQL Port value should remain consistent for the life of VDB.	None.

Delphix Ticket Number	Summary	Workaround/Comments
DLPX-44359	Job cancellation may fail.	None.
CTCHANA-930	Upgrade from 3.4.5 to 3.5.0/3.5.1/4.0.0/4.1.1/4.1.2/4.2.0 needs a manual deletion of the existing tenant databases for disable/enable to work.	Drop the tenant database manually before enabling the VDB.

Release 4.3.0

Delphix ticket number	Summary	Workaround/Comments
HANA-821	HANA SQL Port value should remain consistent for the life of VDB.	None.
DLPX-44359	Job cancellation may fail.	None.
CTCHANA-930	Upgrade from 3.4.5 to 3.5.0/3.5.1/4.0.0/4.1.1/4.1.2/4.2.0 needs a manual deletion of the existing tenant databases for disable/enable to work.	Drop the tenant database manually before enabling the VDB.

Release 4.2.0

Delphix ticket number	Summary	Workaround/Comments
HANA-821	HANA SQL Port value should remain consistent for the life of VDB.	None.
DLPX-44359	Job cancellation may fail.	None.
CTCHANA-930	Upgrade from 3.4.5 to 3.5.0/3.5.1/4.0.0/4.1.1/4.1.2/4.2.0 needs a manual deletion of the existing tenant databases for disable/enable to work.	Drop the tenant database manually before enabling the VDB.

Release 4.1.2

Delphix ticket number	Summary	Workaround/Comments
HANA-821	HANA SQL Port value should remain consistent for the life of VDB.	None.

Delphix ticket number	Summary	Workaround/Comments
DLPX-44359	Job cancellation may fail.	None.
CTCHANA-930	Upgrade from 3.4.5 to 3.5.0/3.5.1/4.0.0/4.1.1/4.1.2 needs a manual deletion of the existing tenant databases for disable/enable to work.	Drop the tenant database manually before enabling the VDB.

Release 4.1.1

Delphix ticket number	Summary	Workaround/Comments
HANA-821	HANA SQL Port value should remain consistent for the life of VDB.	None.
DLPX-44359	Job cancellation may fail.	None.
CTCHANA-930	Upgrade from 3.4.5 to 3.5.0/3.5.1/4.0.0/4.1.1 needs a manual deletion of the existing tenant databases for disable/enable to work.	Drop the tenant database manually before enabling the VDB.

Overview

Overview Of Continuous Data

Learn the fundamentals of our data management platform. If you're new to Delphix, this section will help you understand and use our platform. Here, we will describe the core capabilities of Delphix, which include the following:

- **Delphix Engines**: what is a Delphix Engine and how do they work?
- **Database Linking**: what does linking a database mean and what is the impact?
- **Virtual Databases**: what are virtual databases and what is their architecture?

While using our platform, you will encounter new terminology and concepts that are important to understand. View our glossary as a reference for all concepts that are part of our product experience.

This section covers the following topics:

Delphix engine overview

What is Delphix?

Delphix is a data management platform that provides the ability to securely copy and share datasets. Using virtualization, you will ingest your data sources and create virtual data copies, which are full read-write capable database instances that use a small fraction of the resources a normal database copy would require.

Overview

The Delphix Engine links to source physical databases via standard APIs and asks the source databases to send copies of their entire file and log blocks to it. The copy of the source database stored in the Delphix Engine, along with all incremental updates, is referred to as the dSource in Delphix terminology.

After the initial loading, the Delphix Engine maintains synchronization with source databases based on a user-defined policy. Once linked, Delphix maintains a Timeflow of the source database - a record of snapshots and log changes. From any time within that Timeflow, a virtual database (referred to in Delphix terminology as a VDB) can be instantly provisioned from the Delphix Engine. VDBs are served from the shared storage footprint of the dSource database Timeflow, so no additional storage is required.

Multiple VDBs can be provisioned from any point in time in a Timeflow, down to the second. Once provisioned, a VDB is an independent, read-write database, and changes made to the VDB by users or applications are written to new, compressed blocks in Delphix storage. VDBs can be provisioned from other VDBs, and the data within VDBs refreshed from its parent VDB or dSource.

Getting started with data sources

Delphix enables agile data management, but our users first need to learn how to use our platform. Learn more about what data management looks like and how Delphix fits into the bigger picture. The following sections link to the Getting Started section of our Datasets category. This category also has explanations of other components of the Delphix platform, such as environments and policies.

Managing data sources and syncing data

This section explains how to manage your data sources and create dSources. It hopes to answer questions such as: when you link a database, what happens? How does Delphix stay in sync with changes to a data source, and is there any impact to that source?

Learn about how we link to your data sources to enable fast and secure data management.

Provisioning and managing virtual databases

This section details the virtual database (VDB) functionality and explains how you can create, manage, and utilize these data objects. This section will address questions such as: what is a virtual database, what makes them virtual? What does it mean to provision?

Learn all about the objects we create and refer to as virtual databases or VDBs.

Delphix product Information

Product overview

The Delphix Engine is delivered and maintained as a closed virtual software appliance. Like all virtual appliances, the Delphix Engine is a tightly integrated combination of a special-purpose operating system and business logic. A single engine can be configured for data virtualization or data masking.

The product is delivered as a closed appliance because there are dependencies between software components within the virtual appliance, which require end-to-end testing. As such, we do not provide administrative access to the operating system for any reason, including installing software, making customizations, or performing security scans. More details about the administrative model will be provided in later sections of this document.

Product packaging

The DevOps Data Platform is delivered and maintained as a virtual software appliance. The mode of product delivery is dependent on the hosting hypervisor platform described in the table below:

Hypervisor Platform	**Delivery Vehicle**	**Notes**
VMware ESXi	OVA image	
Amazon AWS EC2	AMI image	Manual deployment to EC2
Amazon AWS EC2	Guest Machine Appliance	Amazon Marketplace
Microsoft Azure	Guest Machine Appliance	Azure Marketplace
Google Cloud Platform	Guest Machine Appliance	
Hyper V	Hyper image	

Product updates, upgrades, and versions

Regardless of the initial packaging used to deploy the Delphix Engine, updates are supplied as a single upgrade image of the new release, and the same image can be used for any prior release from which an upgrade is supported. This upgrade image delivers a completely new appliance, including both the operating system and business logic components.

The upgrade process retains prior configuration and customer data such that no data or configuration is lost during the upgrade process. The upgrade process retains a copy of the previous version of the components for automatic recovery in case an upgrade fails. Installed versions older than the prior version are automatically deleted during the upgrade.

Delphix characterizes software as Major Release, Minor Release, and Maintenance Release in the Delphix Support Policies. These release types indicate the level of feature addition and change. For example, a Major Release may introduce significant new features, interface changes, and many bug fixes. A Maintenance Release or Patch Release

may deliver only a small number of bug fixes and no feature additions. In any case, each release delivers a complete upgrade image of the appliance.

There is no component patching of the Delphix Engine; fixes are delivered in new versions of the software as a new software appliance. There is no management of patches involved, and each Delphix Engine version is a consistent, tested virtual appliance.

Product administration

Administration of the Delphix Engine is effected through product interfaces only. These interfaces provide for the proper configuration and testing of customer infrastructure components, such as network addresses, storage, Domain Name Service (DNS) servers, authentication servers (LDAP), etc. The interfaces also control the business logic and control of the overall platform, including how customer data is used and provisioned by the system.

Although the special-purpose operating system may be accessed by Delphix Support and Engineering personnel for the purpose of diagnostics and problem remediation, there are no customer-accessible interfaces at the operating system level. Customers are not provided access to the underlying operating system nor can any custom software be installed on the appliance.

Product customization

The product has several endpoints allowing customization for improved integration with customer environments, local business workflow requirements, and alternative data sources.

Interface	Functional Area	Description
Data Plugins	Data Virtualization	Delivered by Delphix Services or Integration partners. These plugins allow for supporting additional data types including both structured and unstructured data.
Privilege Elevation Profiles	Data Virtualization	Delivered by Delphix Services or Integration Partners. These customizations allow for the use of privilege mechanisms other than sudo on Linux and Unix target environments. Sudo is the product default.
Hook Scripts	Data Virtualization	These customer-managed scripts allow for custom business logic to be applied to Oracle and SQL Server data sources and virtual databases. The scripts are not integrated into the appliance but are referenced and invoked by the product during data operations.

Interface	Functional Area	Description
Custom Algorithms	Data Masking	Delivered by Delphix Services or Integration Partners. Custom algorithms provide specialized data transformations to secure or anonymize sensitive data.

In addition to these endpoints, Delphix provides a robust set of application programming interfaces (APIs) that enable business automation and fully integrated data operations into client workflows.

Glossary of major Delphix concepts

This glossary is your guide to exploring Delphix terms and definitions.

Products

Term	Explanation
Delphix Virtualization	The Delphix product to deliver data on-demand to application developers and testers.
Continuous Compliance	The Delphix product to identify sensitive data and replace it with realistic but fictitious data.

Functional product components

Term	Explanation
Delphix Management	The user interface for Engine Administrators to configure the product
Delphix Self-Service	The user interface designed specifically for project teams, application developers and testers.

Delphix virtualization engine terms

Delphix virtualization concepts

Term	Explanation
Blocks or data blocks	Relational Database Management System (RDBMS) databases are based on files containing data blocks on disk. These are then backed up and restored with traditional database tools or virtualized with Delphix.
Dataset	A generic term for data in Delphix Virtualization. This can refer to sources or copies of data across any file/ database.

Term	Explanation
Data source	The set of data, typically a database, that you would like to create copies of. Data sources serve as the base from which you will create virtual copies. The source may be a database, a cluster of databases, or a file system. Not to be confused with a dSource, which is a virtualized, compressed duplicate of this database. (See below.)
dSource	A database that the Delphix Virtualization Engine uses to create and update virtual copies of your database. As a virtualized representation of your source data, it cannot be managed, manipulated, or examined by database tools. Because dSources are simply sourced data, you must provision a VDB in order to distribute/clone/test the data being pulled in. VDBs can also later be refreshed from the same or other points in time synced from the dSource.
Delphix Connector	A service that runs on a Windows host and enables the communication between the Delphix Virtualization Engine and the Windows Target Environment where it is installed.
Environment	The server and software required to run a data set. For example, a Linux system running Postgres. This may either be an individual instance, or a cluster like Oracle RAC. Environments can either be a source (where data comes from), staging (where data are prepared/masked) or target (where data are delivered and used by developers and testers).
Hooks	Mechanisms by which Delphix can run scripts or call external processes during common Delphix operations. For example, running a particular script during every VDB provision.

Term	Explanation
HostChecker	A standalone program that validates that environments are configured correctly before the Delphix Virtualization Engine uses them for syncing from data sources or provisioning VDBs. HostChecker should run before adding any Environment to your Delphix Virtualization Engine. It is available to download from the HostChecker subfolder at download.delphix.com.
Replication	Replication enables creating a copy of specific data sets on a second Delphix Virtualization Engine. This is used both for disaster recovery purposes as well as scale-out. Replication is configured on a source engine and the data are moved to a secondary engine. The source engine then sends incremental updates manually or according to a schedule.
Snapshots	Snapshots represent the state of a dataset at a specific moment in time. Snapshots are created from policies or are generated by manual creation. Snapshots allow you to choose a point in time from which to provision, refresh, or rollback. If you have LogSync enabled, you can provision refresh or rollback from a point in time between the snapshots endpoints. Note: Point in time provisioning is also dependent on log retention.
SnapSync	The standard process for importing data from a source into the Delphix Virtualization Engine. An initial SnapSync is performed to create a dSource on the Delphix Virtualization Engine. Incremental SnapSyncs are performed to provide additional points in time to the dSource on the Delphix Virtualization Engine.
LogSync	This feature enables the ingestion and retention of more granular (log-based) source change data. This more granular change data allows for VDB point-in-time provision, refresh, or rollback.

Term	Explanation
Validated Sync	The process that runs on a staging database within a Staging Environment executes database cleanup and preparation. This is recommended and results in faster and more predictable provisions and refreshes, and which executes either before a snapshot is taken (SQL Server) or after a snapshot is taken (Oracle).
Timeflow	The collection of snapshots for a particular data set. Virtual databases can be provisioned from any snapshots in a Timeflow.
Unstructured files	Data stored in a filesystem that is not part of a database Unstructured files can consist of anything from a simple directory to the root of a complex application like Oracle E-Business Suite. Like with other data types, you can configure a dSource to sync periodically with a set of unstructured files external to the Delphix Virtualization Engine.
Virtual Database (VDB)	A database provisioned from either a dSource or another VDB which is a full read/write copy of the source data. A VDB is created and managed by the Delphix Virtualization Engine.
Virtual dataset	A comprehensive term that includes VDBs and virtualized files.
V2P	Shorthand for “Virtual-to-Physical Provisioning”. This refers to the process of moving a dataset from Delphix back to a full-size database. This can be used in DR purposes or as part of a final performance test, matching the production systems.

TDE definitions

The following terms are used by Delphix and are summarized here for clarity.

Term	Definition
Artifact directory	Directory on the target system (not on Delphix storage) which stores keys needed to support Delphix workflows on TDE-enabled vPDBs. It is located under the toolkit directory.

Term	Definition
Auxiliary container database (CDB)	Provisioning an Oracle vPDB requires running recovery to bring the snapshotted datafiles into a consistent state. This needs to be done in the context of a container database, which is created on the target system. After the recovery is complete, the vPDB is unplugged and plugged into the target container, and the auxiliary container is deleted.
Exported keyfile	File located on the target Oracle host which contains keys that have been exported from the Keystore. It is encrypted with a secret that is specified when it is exported. The exported keyfile itself cannot be used as a Keystore, but its contents can be imported into a new Keystore.
Key rotation	Process for changing the master encryption key in the keystore via ADMINISTER KEY MANAGEMENT SET KEY. This does not remove the original key, rather it adds a new key to the wallet and future data will be encrypted with the new key.
Keystore/wallet	File found on the Oracle host which stores the keys used to encrypt and decrypt the internal table keys in a database. Every keystore has a password that is set when it is first created and must be supplied for operations on it.
Parent Keystore	Keystore with the keys used to encrypt the dSource PDB files.
Target Keystore	Keystore for the target CDB into which the TDE-encrypted vPDB is plugged.

Dataset operations

The terms below describe actions you can perform on a Delphix Virtualization Engine.

Term	Explanation
Link	The process of establishing a relationship between a data source and the Delphix Virtualization Engine. After linking a data source, the Delphix Virtualization Engine can import data periodically and manage it as it evolves over time. In the user interface, this accomplished via "Add dSource."
Mask	Masking replaces sensitive data with fictitious data in non-prod environments (such as VDBs). It provides realistic data with which to work while reducing security risks. For more details about masking, see Masking Terms below.
Migrating a VDB	Moving a VDB to a new Target Environment.
Provision	Create a new VDB from a dSource or VDB.
Refresh	Refreshing a VDB will re-provision it from the dSource. As with the normal provisioning process, you can choose to refresh the VDB from a Snapshot or a specific point in time. Refreshing a VDB will delete any changes that have been made to the VDB prior to the refresh operation; you are essentially resetting it to the state you select during the refresh process.
Rewind	Rewinding a VDB rolls it back to a previous point in its Timeflow. The VDB will no longer contain changes that occurred after the rewind point.
Staging Push	Allow users to control the staging database restore and the state of the restored staging database.
Disable	Remove all evidence of a specific VDB from a target environment. This action deletes the database and network pathway(s) on the target host. But it leaves the current VDB filesystem and snapshot(s) intact in Delphix Virtualization Engine so that VDB can later be enabled on a chosen target environment.
Enable	Establish the network pathway for the VDB between Delphix Virtualization Engine and the appropriate target environment, then start the specific VDB database.

Term	Explanation
Stop	Leave network pathway (e.g., NFS or iSCSI) in place on the target environment for VDB, but stop the specific VDB database on the target environment.
Start	Start a specific VDB database on the target environment. This will result in no action for the network pathway and is assumed to be in place already.
Delete	Remove all evidence of specific VDB from the target environment and the Delphix Virtualization Engine. This operation is intentionally destructive.

Delphix virtualization users and privileges

Object	User Privileges	Group Privileges
Reader	Access statistics on the dSource, VDB, or Snapshot such as usage, history, and space consumption	Access statistics on all dSources, VDBs, or Snapshots in the group such as usage, history, and space consumption
Provisioner	• Access statistics on the dSource, VDB, or Snapshot such as usage, history, and space consumption • Provision VDBs from owned dSources and VDBs	• Access statistics on all dSources, VDBs, or Snapshots in the group such as usage, history, and space consumption • Provision VDBs from all dSources and VDBs in the group

Object	User Privileges	Group Privileges
Owner	• Provision VDBs from owned dSources and VDBs • Perform Virtual to Physical (V2P) from owned dSources • Access the same statistics as a Reader • Refresh or Rollback VDBs • Take Snapshots of dSources and VDBs	• Provision VDBs from all dSources and VDBs in the group • Refresh or Rollback all VDBs in the group • Snapshot all dSources and VDBs in the group • Perform Virtual to Physical (V2P) from owned dSources • Apply custom policies to dSources and VDBs • Create template policies for the group • Assign Owner privileges for dSources and VDBs • Access the same statistics as a Provisioner, Data Operator, or Reader
Data operator	• Access statistics on the dSource, VDB, or snapshot such as usage, history, and space consumption • Refresh or rollback VDBs	• Access statistics on all dSources, VDBs, or snapshots in the group such as usage, history, and space consumption • Refresh or rollback all VDBs in the group
Administrator	Manage all data objects: dSources, virtual databases (VDBs), users, groups, and related policies and resources." The `admin` user manages the Delphix Virtualization Engine using either the browser-based Delphix Management application or the Command Line Interface (CLI).	Has privileges over all objects and users.

Object	User Privileges	Group Privileges
sysadmin	Can perform typical system administration duties such as: modifying NTP, SNMP, SMTP settings; managing storage; downloading support logs for the Delphix Virtualization Engine, and performing upgrades and patches. The sysadmin user launches the initial Delphix Setup configuration application and has access to the Command Line Interface (CLI).	Has privileges for storage, upgrades, network, etc.

Types of notification

Type	Notification
Event	Completion of some action in the Delphix Virtualization Engine. Examples include user-initiated tasks such as snapshots or VDB provisioning, policy-based tasks, and background monitoring and maintenance tasks.
Alert	Caused by a single event on a Delphix Virtualization Engine. Also known as a System Event, and viewable through the System Event Viewer. Examples include warnings on source/target environment settings, recoverable errors, or incorrect connection settings. Alert Levels: Informational, Warning, Critical
Fault	A persistent event on a Delphix Virtualization Engine that remains until the issue is resolved. The fault may be marked resolved automatically or require that it be resolved manually. Selecting to Ignore a fault will also ignore future faults of that exact type against the same object. System faults describe states and configurations that may negatively impact the functionality of the Delphix Virtualization Engine and which can only be resolved through active user intervention. Examples: Delphix Virtualization Engine storage failure, communication failures between the Delphix Virtualization Engine and a source or target environment/host Fault Levels: Warning, Critical

Delphix self-service terms

Term	Explanation
Administrator	Has full access to all report data and can configure and administer Delphix Self-Service. Additionally, can use the Delphix Virtualization Engine to: • add/delete reports • add/delete users • change tunable settings • add/delete tags • create and assign data templates and containers
Bookmark	A logical reference to a point in time on a branch. You can use it as a point from which to fork new branches. It can also be the target of policies – for example, you can arrange to keep this bookmark for two years. Bookmarks are a way to mark and name a particular moment of data on a timeline. You can restore the active branch's timeline to the moment of data marked with a bookmark. You can also share bookmarks with other Delphix Self-Service users, which allows them to restore their own active branches to the moment of data in your container. The data represented by a bookmark is protected and will not be deleted until the bookmark is deleted.
Branches	Branches are task-specific groupings you can create within a data container. A branch is used to track a logical task and contains a timeline of the historical data for that task. As you work within your data container, you can create more branches overtime to run or complete separate tasks. Branches represent a logical sequence of activity, separate from the underlying data lineage. This is the main concept introduced in the core engine and forms the basis of many higher-level primitives. Branches: • Can have only one timeline active at any time • Can be user-visible (e.g. exported to a user target) or implementation (e.g. just a staging source to run a series of transformations)
Branch group/target group	A collection of multiple Branches that are treated as a single entity. The system can determine compatibility automatically, or a template can be used to create more complex orchestration.

Term	Explanation
Branch timeline	A dynamic point-in-time interface for user actions within the Branch. Common activities include re-setting Data Sources to run a test, refreshing the Data Container with the most current source data, and bookmarking data to share or track interesting moments of time along the branch timeline.
Data container	Consists of one or more Data Sources, such as databases, application binaries, or other application data. Allows users to: • Undo any changes to their application data in seconds or minutes • Have immediate access to any version of their data over the course of their project • Share their data with other people on their team, without needing to relinquish control of their own container • Refresh their data from production data without waiting for an overworked DBA
Data template	Created by the Engine Administrator, data templates consist of the data sources users need in order to manage their data playground and their testing and/or development environments. Data templates serve as the parent for a set of data containers that the administrator assigns to Delphix Self-Service users. Additionally, data templates enforce the boundaries for how data is shared. Data can only be shared directly with other users whose containers were created from the same parent data template.
Data User	Delphix Self-Service data users have access to production data provided in a data container. The data container provides these users with a playground in which to work with data using the Self-Service Toolbar.

Product icon reference

This topic illustrates the icons that appear on dSources and virtual databases (VDBs) in the Delphix Engine graphic user interface and describes the meaning of each, along with tips for clearing those that represent errors.

Icon	Description
	Environment
	Host
	Cluster
	dSource
	VDB
	Masked VDB
	vFiles
	Live Source
	Universal control for adding, creating users and objects
	Provision a Virtual Database (VDB)
	Refresh a VDB
	Take a snapshot
	Rewind or roll back a VDB to a point in time
	Copy Virtual Database to a physical database (V2P)
	Open Log Sync to create a dataset from a point in time
	Filter control to narrow the results in a list view
	Toggle to expand the current state

Icon	Description
	Toggle to collapse current state
	Timeflow view selector
	There is a warning fault associated with the dataset
	There is a critical fault associated with the dataset

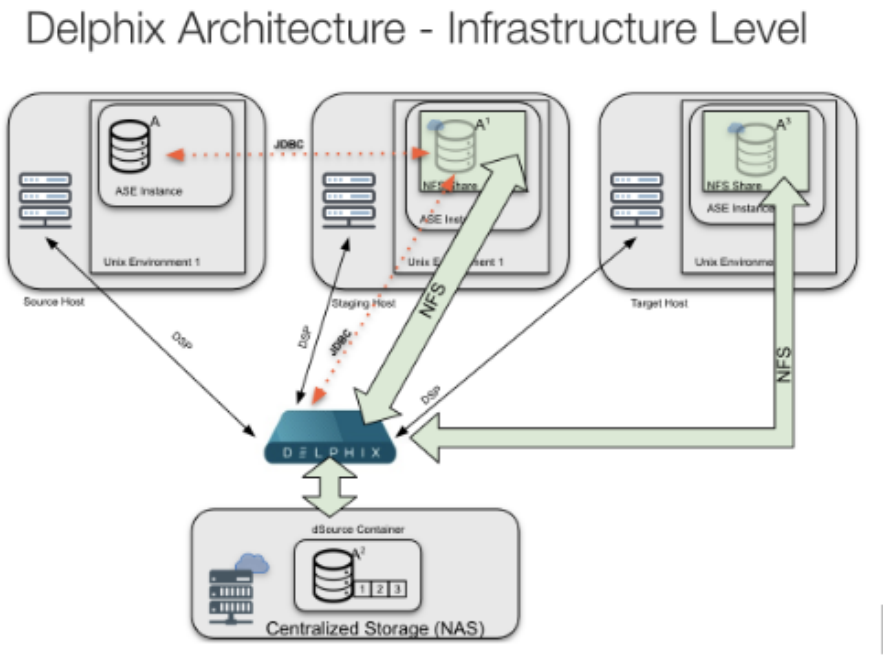
Deployment

This section covers the necessary requirements to deploy the Delphix Engine across multiple different environments. Specific technical requirements for both on-premise and cloud deployments can be found under each platform topic (VMware, AWS, Azure, Google Cloud, etc.). This section covers the following topics:

Standard deployment architecture

Overview

At a high level, Continuous Data functions to efficiently ingest data off of a source host by creating a compressed copy of that data, called a dSource, on the Delphix Engine. From there, the Delphix Engine can easily create multiple virtual databases (VDBs) on a target host with a marginal increase in storage. From an architecture perspective, the diagram below details how an engine interacts with an organizations infrastructure.



Architectural diagram of a common Delphix deployment (configurations vary depending on RDBMS platforms).

On an infrastructure level, Continuous Data relies on a Source, Staging, and Target Hosts, in addition to the Delphix Engine needing specific compute requirements (outlined in the platform-specific requirements section). Storage Requirements, like compute, vary by host type and are outlined in subsequent sections. Staging and Target host storage is provided from the Delphix Engine, which is mounted over the network similar to any Target host (NFS/ iSCSI).

Checklist of information required for installation and configuration

Overview

This article describes the information required for the initial installation and configuration of the Delphix Engine.

The **InstallWorkbooks** articles list all the values that need to be captured, as well as sample values. Choose the workbook for the corresponding hypervisor. In addition to required values, the InstallWorkbooks also lists useful data to document as part of the installation.

- InstallWorkbook AWS
- InstallWorkbook VMware
- InstallWorkbook Azure

Hypervisor specific options

As a virtual appliance, Delphix supports a number of different hypervisors and configurations. The choice of hypervisor and network configuration will determine what configurations are needed for the first step in the installation of Delphix.

Attribute	**Sample data**	**Used with the following hypervisor options**			
		VMware with Static IP	**VMware with DHCP**	**AWS**	**Azure**
Name	mydelphix1	*	*	*	*
Gateway IP	192.168.0.1	*	*	*	*
Root Volume Size	127GB	*	*	*	*
ESX Hostname	thx1138	*	*		
Domain	mydomain.com	*		*	*
DNS server IP(s), comma-separated	10.80.1.1, 10.80.1.2	*		*	*
Static IP and Subnet Mask in CIDR Notation	172.16.180.130/24	*			
Number of vSCSI Controllers	4	*			

Attribute	Sample data	Used with the following hypervisor options			
VMDK/RDMs and Sizes	SAN 03- 3 TB, SAN04-3 TB, SAN05-3 TB & SAN06-3 TB	*			
Virtual Private Cloud (VPC) Name	vpc-673822a23			*	
Subnet	subnet-748391e26			*	
Auto-assign Public IP	disable			*	
EC2 instance type	r3.8xlarge			*	
Security Groups	sg-78sdg99879, sg-8798s77009			*	
Root Volume Type	Magnetic			*	
Number of EBS volumes	4			*	
Size of EBS volumes	500G			*	
Type of EBS volumes	Provisioned IOPS SSD			*	
IOPS set on each volume	5,000			*	
Encryption on EBS volumes	TRUE			*	
Virtual Network (vnet)					*
Subnet					*
Network Interface w/ Static Public IP					*

Attribute	Sample data	Used with the following hypervisor options			
VM Size	Example: D8s_v3				*
Network Security Group					*
OS Disk	127 GB - Premium SSD				*
Data Disks	Premium SSDs				*

Optional information required for initial configuration

Class	Attribute	Sample data
Time	NTP Servers (Highly Recommended)	172.16.180.10, 172.16.180.11
	Timezone	US/Pacific
Serviceability (Highly Recommended)		
	Enable Phone Home? (Highly Recommended)	Yes
	Web Proxy Server IP Address	192.168.1.200
	Web Proxy Server Port	8080
	Web Proxy Server Username (If Required)	user
	Web Proxy Server Password (If Required)	***
	Enable SMTP Server (Highly Recommended)	Yes
	SMTP Server IP Address	192.168.22.22

Class	Attribute	Sample data
	SMTP Server Port Number	25
	SMTP Server From Email Address	delphix@hostname.mydomain.com
	SMTP Server User (Optional)	user
	SMTP Server Password (Optional)	***
	SMTP Use TLS Authentication? (Optional, Default=No)	No
	SMTP Send Timeout (Optional, Default=60)	60
Authentication (Highly Recommended)		
	LDAP Server IP Address	192.168.7.7
	LDAP Server Port	389 (636 for SSL)
	Protect LDAP traffic with SSL/TLS (true | false)	false
	LDAP SSL Authentication (SIMPLE | DIGEST_MD5)	SIMPLE
SNMP Integration		
	SNMP Server IP Address	192.168.77.156
	SNMP Server Port	162
	SNMP Community	mycommunity
	SNMP Use INFORM Instead of TRAP	false

Class	Attribute	Sample data
	SNMP Severity Level (INFORMATIONAL, WARNING, CRITICAL)	CRITICAL
Splunk integration (Highly recommended)		
	Splunk Server IP address	192.168.8.8
	Splunk Server HEC Port Number	8088
	Splunk HEC Token	12345678-1234-1234-1234-1234567890AB
	Index Name for Events	delphix_events
	Index Name for Metrics	delphix_metrics
ADMIN		
	Email Address	u12345@mydomain.com
SYSADMIN		
	Email Address	u12345@mydomain.com

Network connectivity requirements

Overview

This topic covers the general network and connectivity requirements for the Delphix Engine, including connection requirements, port allocation, and firewall and Intrusion Detection System (IDS) considerations. For platform-specific network and connectivity requirements, see relevant topics under the **Requirements** article for each platform.

General outbound from the Delphix engine port allocation

Protocol	Port Numbers	Use
TCP	25	Connection to a local SMTP server for sending email
TCP/UDP	53	Connections to local DNS servers
UDP	123	Connection to an NTP server
UDP	162	Sending SNMP TRAP messages to an SNMP Manager
TCP	443	HTTPS connections from the Delphix Engine to the Delphix Support upload server
TCP/UDP	636	Secure connections to an LDAP server
TCP	8415	Connections to a Delphix replication target (see Configuring Replication)
TCP	50001	Connections to source and target environments for network performance tests

General inbound to the Delphix Engine port allocation

Protocol	Port Number	Use
TCP	22	SSH connections to the Delphix Engine
TCP	80	HTTP connections to the Delphix GUI
UDP	161	Messages from an SNMP Manager to the Delphix Engine
TCP	443	HTTPS connections to the Delphix Management Application
TCP	8415	Delphix Session Protocol connections from all DSP-based network services including Replication, SnapSync for Oracle, V2P, and the Delphix Connector.
TCP	50001	Connections from source and target environments for network performance tests via the Delphix CLI

Firewalls and intrusion detection systems (IDS)

Production databases on source environments (for dSources) are often separated from the non-production environment by firewalls. Firewalls can add milliseconds to the latency between servers. Accordingly, for best performance, there should be no firewalls between the Delphix Engine and the virtual database (VDB) target environments. If the Delphix Engine is separated from a source environment by a firewall, the firewall must be configured to permit network connections between the Delphix Engine and the source environments for the application protocols (ports) listed above.

Intrusion detection systems (IDSs) should also be made permissive to the Delphix Engine deployment. IDSs should be made aware of the anticipated high volumes of data transfer between dSources and the Delphix Engine.

Setting up network access to the Delphix engine

Overview

This article outlines the procedure for setting up network access to the Delphix Engine. Follow the initial installation instructions in Installing the Delphix engine before addressing this procedure.

Procedure

1. Power on the Delphix Engine and open the Console.
2. Wait for the Delphix Management Service and Delphix Boot Service to come online. This might take up to 10 minutes during the first boot. Wait for the large orange box to turn green.
3. Press any key to access the sysadmin console.
4. Enter `sysadmin` for the username and the password.
5. A description of available network settings and instructions for editing will be shown.

```
Delphix Engine Network Setup

To access the system setup through the browser, the system must first be
configured for networking in your
environment. From here, you can configure the primary interface, DNS, hostname,
and default route. When DHCP is
configured, all other properties are derived from DHCP settings.

To see the current settings, run "get." To change a property, run "set
<property>=<value>." To commit your changes,
run "commit." To exit this setup and return to the standard CLI, run "discard."

    defaultRoute    IP address of the gateway for the default route -- for
 example, "1.2.3.4."

    dhcp            Boolean value indicating whether DHCP should be used for
 the primary interface. Setting this value
                    to "true" will cause all other properties (address,
hostname, and DNS) to be derived from the DHCP
                    response

    dnsDomain       DNS Domain -- for example, "delphix.com"

    dnsServers      DNS server(s) as a list of IP addresses -- for example,
"1.2.3.4,5.6.7.8."

    hostname        Canonical system hostname, used in alert and other logs --
for example, "myserver"

    primaryAddress  Static address for the primary interface in CIDR notation
-- for example, "1.2.3.4/22"

Current settings:
```

```
    defaultRoute: 192.168.1.1
    dhcp: false
    dnsDomain: example.com
    dnsServers: 192.168.1.1
    hostname: Delphix
    primaryAddress: 192.168.1.100/24
```

6. Configure the `hostname`. If using DHCP, this step can be skipped.

```
delphix network setup update *> set hostname=<hostname>
```

Use the same hostname entered during the server installation.

7. Configure DNS. If using DHCP, this step can be skipped.

```
delphix network setup update *> set dnsDomain=<domain>
delphix network setup update *> set dnsServers=<server1-ip>[,<server2-ip>,...]
```

8. Configure either a static or DHCP address:
DHCP configuration

```
delphix network setup update *> set dhcp=true
```

Static configuration

```
delphix network setup update *> set dhcp=false
delphix network setup update *> set primaryAddress=<address>/<prefix-len>
```

⚠ The static IP address must be specified in CIDR notation (for example, `192.168.1.2/24`).

9. Configure a default gateway. If using DHCP, this step can be skipped.

```
delphix network setup update *> set defaultRoute=<gateway-ip>
```

10. Commit the changes. Use the get command prior to committing to verify the desired configuration.

```
delphix network setup update *> commit
Successfully committed network settings. Further setup can be done through the
browser at:

        http://<address>

Type 'exit' to disconnect, or any other commands to continue using the CLI.
```

11. Check that the Delphix Engine can now be accessed through a Web browser by navigating to the displayed IP address, or hostname if using DNS.

12. Exit setup.

1.
```
delphix> exit
```

Installation and initial system configurations

This section covers the following topics:

- Initial setup
- Customizing the Delphix engine system settings
- Installing an OVA or AMI

Initial setup

Overview

Delphix runs as a virtual appliance deployed in various types of platforms, as outlined in these **Deployment** articles. When first logging into an instance, the setup wizard will help with initial configurations for network, storage, authentication, and more. This article describes each step in the setup process and will outline and the different options available.

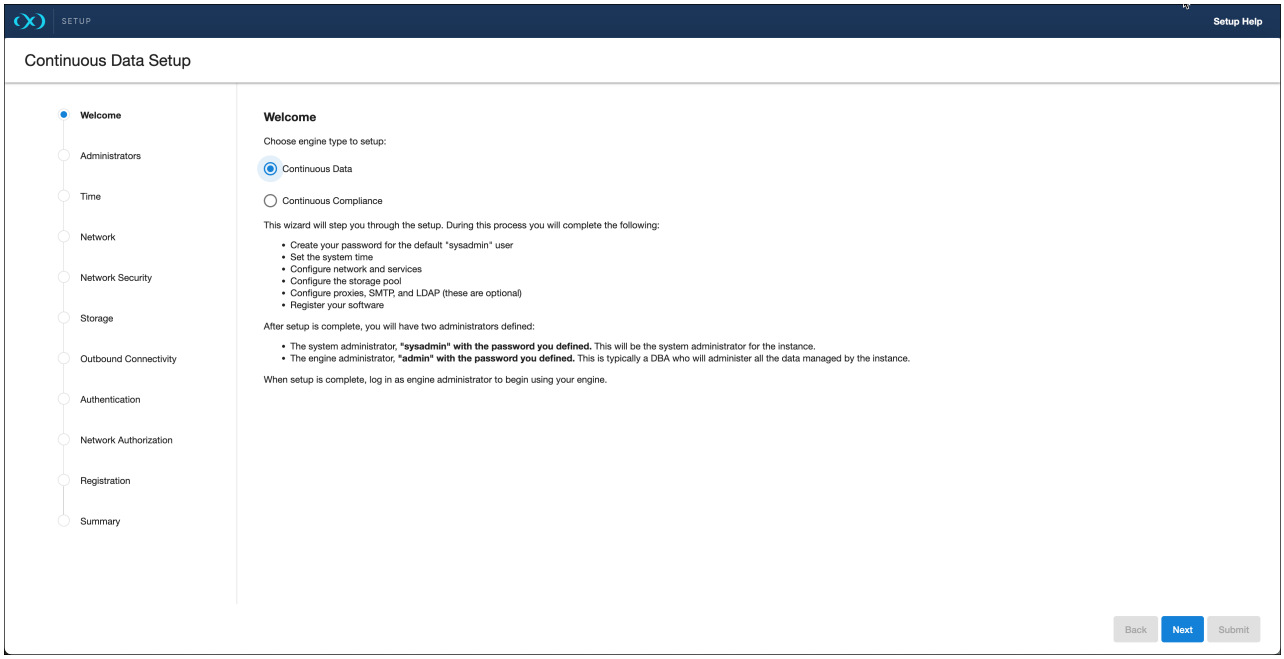
When first connecting to the Delphix Engine via any supported browser, enter the default sysadmin login; **username**: sysadmin, **password**: sysadmin. On the first login, there will be a prompt to change the default password for security purposes.

> A login failure issue could occur if the Delphix Engine clock is not in sync with the IdP (identity provider) clock. To resolve the issue, either use the NTP (network time) clock or set up the skew time property in the SSO (Single Sign-On) configuration.

Welcome

The Welcome section asks users to select the engine type being setup, whether Continuous Data or Continuous Compliance. This article explains the Continuous Data setup, the Continuous Compliance version can be found

The Welcome tab asks users to select the engine type for setup, and choose from Virtualization or Masking. This document explains the setup for Virtualization engines. For Masking engines, please visit the Masking documentation.



Administrators

Each **Continuous Data** engine has two default accounts:

- **System Administrator**: `sysadmin` with a user-defined password. This will be the system administrator for the instance.
- **Engine Administrator**: `admin` with a user-defined password. This is typically a DBA who will administer all the data managed by the instance.

Provide an email address and password for both users in the Administrator section.

Each **Continuous Compliance** engine has this default account:

- **Masking Administrator**: `admin` with a user-defined password. This will be the user responsible for setting up other users and handling administrative actions.

Time

The Delphix engine leverages its time setting to determine policies and actions that take place within the application. Manually set the time or choose from an NTP server, an explanation of these options are shown below.

Option	Notes
Set NTP Server (recommended)	After selecting this option, select an NTP server from the list, or select **Add NTP Server** to manually enter one or more server(s). When configuring a Delphix Engine on VMware, be sure to configure the NTP client on the host to use the same servers that are entered here.
Manually Select Time and Date	Select the **Use browser time and date** option to set the system time, or select the date and time by using the calendar and clock icons. If this option is selected, the date and time will persist as the local time, even if time zones are changed.

Network

The initial network configuration will be pre-populated based on the deployment platform used for Delphix. For VMware deployments, Delphix defaults to the VMXNET3 network adapter.

Select **Settings** for each Network interface to manage the following options:

Option	Notes
DHCP or Static network addressing	For Static addressing, enter an IP Address and Subnet Mask. The static IP address must be specified in CIDR notation (for example, 192.168.1.2/24).
Jumbo Frames	This setting is highly recommended. VMXNET3 supports Ethernet jumbo frames, which can be used to maximize throughput and minimize CPU utilization.
Routing	A default gateway will be specified in this section.

Option	Notes
DNS Services	Enter a DNS Domain Name and DNS Server to be used for this engine.

Network security

Delphix installs certificates signed by the engine's Certificate Authority. Users have the ability to manage their own certificates for HTTPS and DSP (Delphix Session Protocol) connections to and from the Delphix Engine. You can add or modify certificates and certificate signing requests (CSRs) via the **...** option.

When you update the Certificate Authority certificate, your HTTPS and DSP certificates will be automatically updated.

For more information please refer to Certificate management in the Security section of this documentation.

Storage

Storage for engines backed by block devices

The Delphix engine automatically discovers and displays storage devices. For each device, confirm that **Usage Assignment** is set to Data.

You can associate additional storage devices with the Delphix engine after initial configuration, as described in Adding, expanding, and removing storage devices.

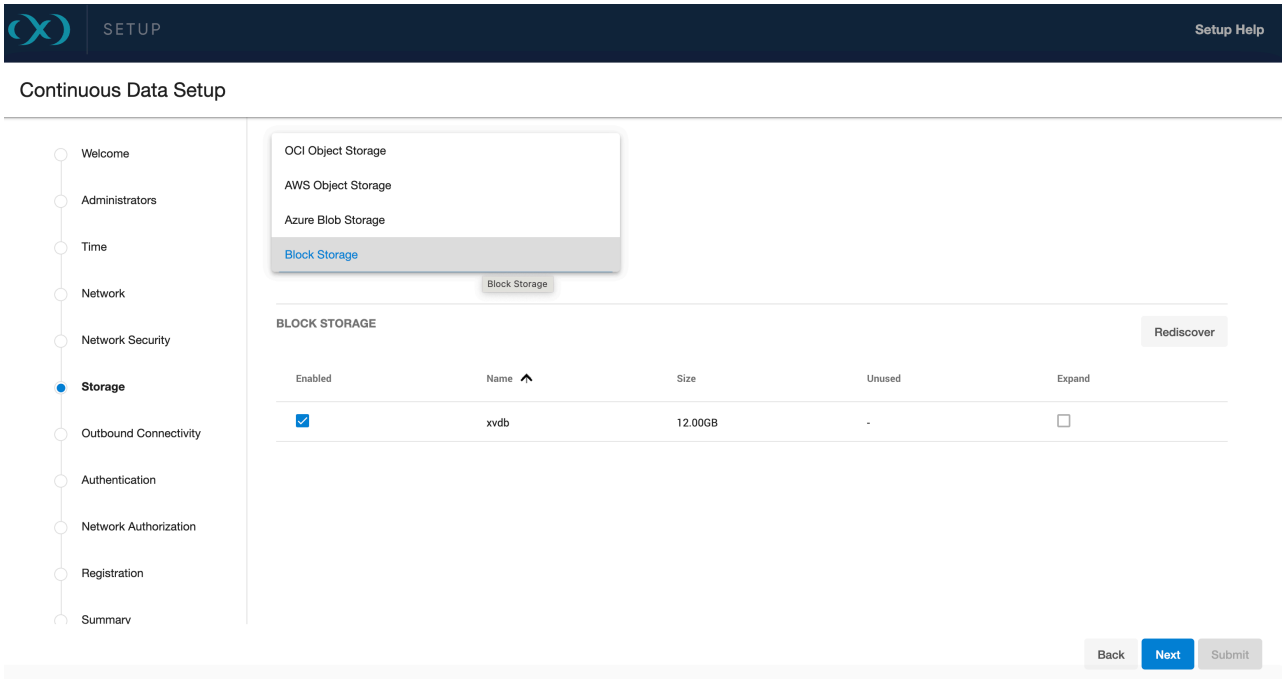
There are two options for storage disk usage assignment:

1. **Enabled**: Once you set the storage unit assignment for a disk and save the configuration, you cannot change it again.
2. **Unassigned**: These are disks being held for later use.

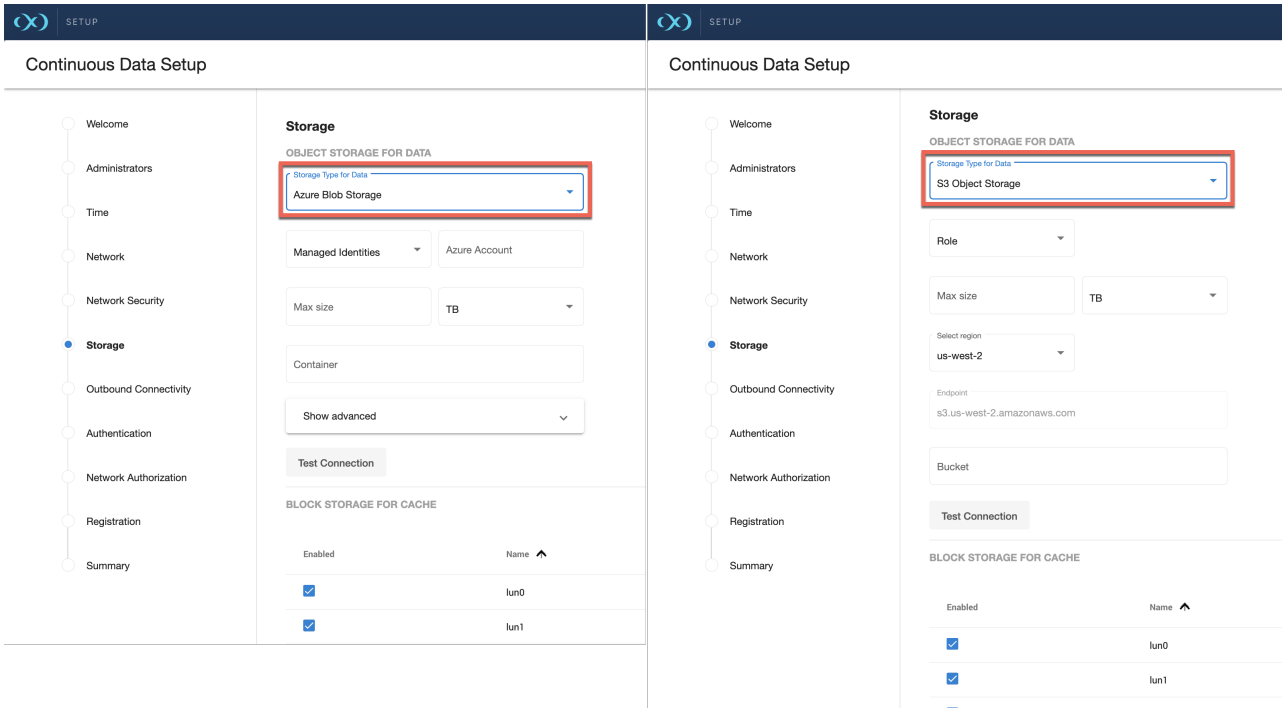
Configure at least four disks for the storage of user data. This makes the Delphix engine storage manager function more efficient since duplicated metadata can be distributed across multiple disks.

Delphix Elastic Data Engines (Engines backed by object storage)

- Select one of the four options from the drop-down menu: Block Storage, AWS Object Storage, OCI Object Storage or Azure Blob Storage. For on premise object storage select AWS Object Storage. Block storage usage is covered in the previous section, Elastic Data engines should select the corresponding storage type.
- Delphix only supports the Online access tier (hot tier of Standard GPv2) for Azure-Blob.
- For on premise object storage we currently support storage vendors that confirm to the following
    - s3 REST API compatibility
    - strong read-after-write consistency
    - supports s3 key id and secret access key authentication
    - perpetual key support



- Once one of these options is selected, enter the corresponding information with the fields that appear.
  - For AWS Object Storage, select whether the engine should access the storage bucket via Role (Instance profile) or via Access Key. For on-prem object storage only Access Key authentication is supported. For Azure-Blob, select whether the engine should access the storage bucket via Managed Identities or via Access Key. Make sure you do not have versioning on your blob/bucket, otherwise deleted space will not be reclaimed since it will be held by the versioning logic of the cloud vendor. For OCI object storage, Role (instance principals) is only supported.
  - Select the Region Enter the Base URL, Region, and Bucket that you want to use for the engine. For on-prem object storage, select Other from the Region dropdown menu, then enter the region and endpoint URL in the provided fields. For Azure-Blob, there is no region to configure. In addition, “endpoint” is an optional field (hidden in the Advanced tab). For OCI object storage, only the bucket information is needed.



- Enter the amount of data that you would like to store on the engine. The size would be similar to the total storage on a traditional engine. Delphix Elastic Data engines provide the following advantages:
  - Blob/Object storage only charges for the amount of storage used (In traditional EBS-backed engines, AWS charges based on provisioned storage). I.e., if the size was 10TB but the engine only uses 1TB, then AWS only charges for the 1TB of storage used.
  - Increasing or reducing this number after setup is simple. It involves editing this number from the sysadmin login (size cannot be reduced to a number lesser than what the Delphix engine is already using).
  - This number acts as a quota in case you do not want S3 storage and costs to grow beyond a certain number.
- Make sure to **Test** the connection to confirm that the VM can access the bucket.

Block storage for cache

- Block devices as cache are used to reduce latencies for frequently read data and as temporary storage for synchronous writes before the writes are sent to blob/object storage.
- **Sizing:** If you already know the size of the frequently accessed data, then size the cache equal to (size of frequently accessed data + Extra 10% for bookkeeping purposes), If not, start off with sizing the cache to 50% of the size of all dSources that will be added to the engine.
- For specific block storage requirements for each platform, refer to:
  - See, EBS Configuration section of Deployment for AWS EC2 .This is only applicable to AWS.
  - See, Storage configuration section of Deployment for Microsoft Azure
  - See, General storage configuration section of Deployment for OCI
  - See, General storage section of Deployment for VMware
- Setup the disks such that they can support the throughput of the engine. For example
  - AWS
    - gp3 disks are recommended as they offer good performance at a lower cost. At 500 IOPS per 1GiB ratio, a 32 GiB volume can be configured to have the gp3 volume 16K IOPS limit and 1000 MB/s throughput. For reference, r5n.8xlarge instance has a 30K IOPS and 850 MB/s throughput limits so two gp3 devices would be sufficient for the instance.
    - io2 disks have lower latency at a higher cost. However, the lower latency is not beneficial once the instance IOPS or throughput limit is reached.

- Azure
  - Ultra disks are recommended as they can be configured to have high IOPS and throughput at relatively small sizes. At 300 IOPS per GiB ratio, a 256 GB volume can have 76,800 IOPS and 4000 MB/s throughput.
  - Premium SSD disks can have high performance though they need to be much larger, e.g. P80 32TiB volume has 20K IOPS and 900 MB/s. The new Premium SSD v2 disks have better IOPS per GiB ratio though they're not available in all regions. SSD v2 is currently untested due to this unavailability.
- OCI
  - Ultra High Performance and Higher Performance volumes are supported.
  - Max IOPs and Throughput are proportional to the volume size. A minimum size of 200G using Higher Performance volumes is required.

Elastic Data engines on Azure - storage account and permission setup

Storage container

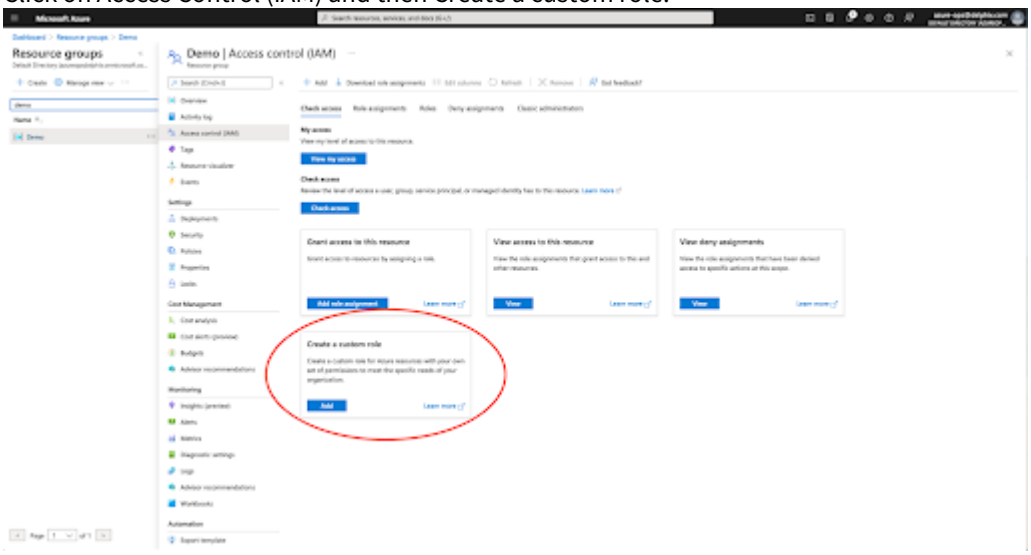
To have a dedicated Storage Account for your Delphix Elastic Data engine, create a new Storage Account. This can be done by navigating to Storage Accounts and clicking **Create**. Note that Azure places restrictions on ingress and egress per storage account (https://docs.microsoft.com/en-us/azure/storage/common/scalability-targets-standard-account).

Create a Storage Container in the Storage Account being used for your Delphix Engine. Navigate to the Storage Account you want to use, click **Containers** under the **Data Storage** section, and click **+ Container**. Take note of the Storage Container Name, the Storage Account name, and the Resource Group name.

Managed role

Create a custom role that gives access to the storage account. This role will be assigned to the virtual machine after creation.

1. Navigate to the Resource Group for the Storage Account that contains the Storage Container that was created above.
2. Click on Access Control (IAM) and then Create a custom role.



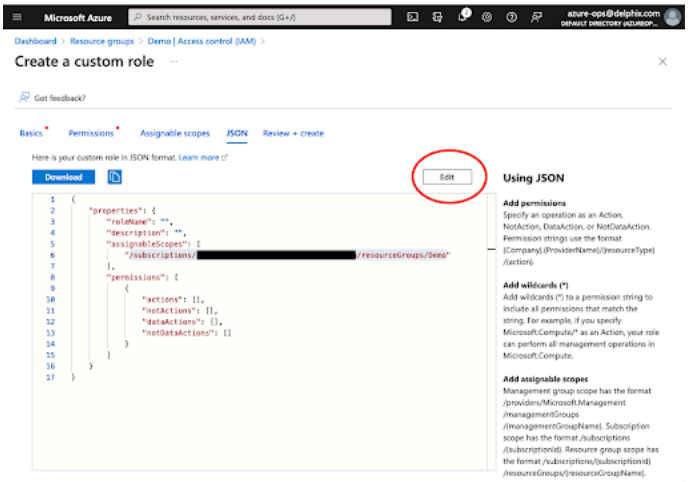
3. Create a role with the following permissions with a minimum scope of the storage account that your Delphix Engine will be using

Permission Type	Permission
Action	Microsoft.Storage/storageAccounts/blobServices/containers/read
DataAction	Microsoft.Storage/storageAccounts/blobServices/containers/blobs/write
DataAction	Microsoft.Storage/storageAccounts/blobServices/containers/blobs/read
DataAction	Microsoft.Storage/storageAccounts/blobServices/containers/blobs/move/action
DataAction	Microsoft.Storage/storageAccounts/blobServices/containers/blobs/add/action
DataAction	Microsoft.Storage/storageAccounts/blobServices/containers/blobs/delete

4. Alternatively, in the Create a custom role navigate to the JSON view of the role and click Edit in the upper right. Copy the JSON below with the bolded section replaced with your values, namely Your_Role_Name, Your_Subscription_ID, Your_Resource_Group_Name, Your_Storage_Account_Name. Be sure to click Save after adding the JSON.

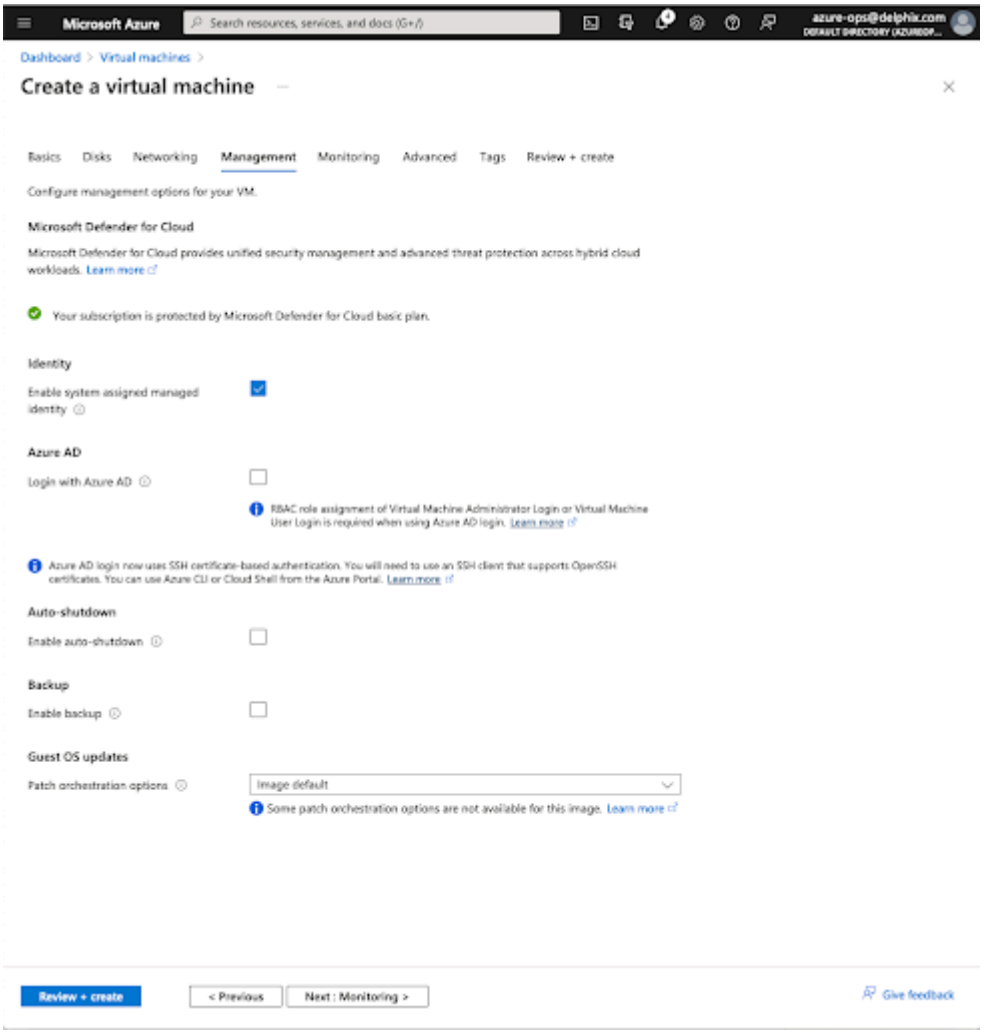
```
{
    "properties": {
        "roleName": "<Your_Role_Name>",
        "description": "Delphix object storage Azure role permissions",
        "assignableScopes": [
            "/subscriptions/<Your_Subscription_ID>/resourceGroups/
<Your_Resource_Group_Name>/providers/Microsoft.Storage/storageAccounts/
<Your_Storage_Account_Name>"
        ],
        "permissions": [
            {
                "actions": [
                    "Microsoft.Storage/storageAccounts/blobServices/containers/
read"
                ],
                "notActions": [],
                "dataActions": [
                    "Microsoft.Storage/storageAccounts/blobServices/containers/
blobs/write",
                    "Microsoft.Storage/storageAccounts/blobServices/containers/
blobs/read",
                    "Microsoft.Storage/storageAccounts/blobServices/containers/
blobs/move/action",
                    "Microsoft.Storage/storageAccounts/blobServices/containers/
blobs/add/action",
                    "Microsoft.Storage/storageAccounts/blobServices/containers/
blobs/delete"
                ],
```

```
                "notDataActions": []
            }
        ]
    }
}
```

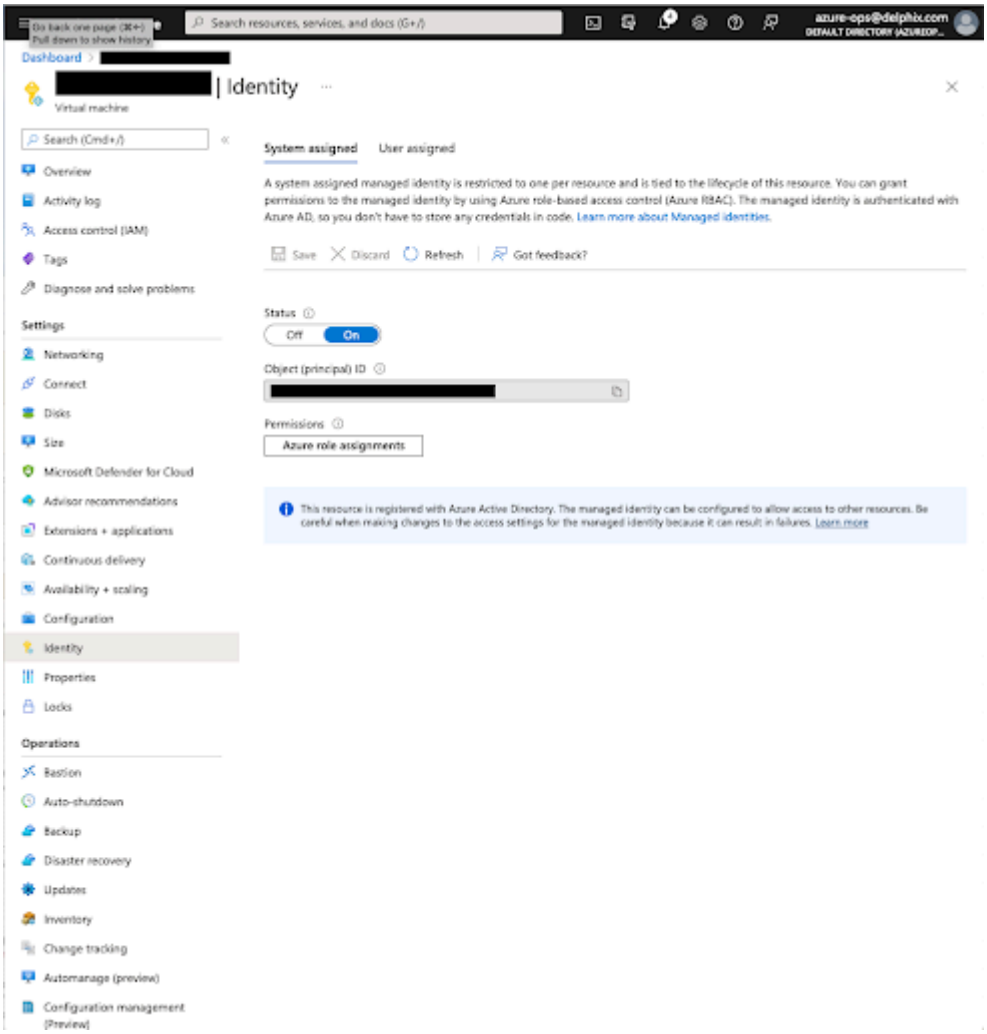


Virtual machine

1. When creating your virtual machine ensure that **system-assigned managed identity** is enabled for the VM.



2. After the virtual machine is created, you need to assign the role that was created above to the virtual machine. Navigate to the virtual machine and click on **Identity** and then **Azure role assignments**.



3. Click **Add role assignment**. For the scope, select **Storage**. Select your subscription, for **Resource** select the Storage Account you will be using, and for the **Role**, select the new role you created above.

Testing permissions

To test that the role permissions are working correctly, assign the role you created above to a generic Ubuntu VM. Install the Azure CLI following Azure's documentation https://learn.microsoft.com/en-us/cli/azure/install-azure-cli

Run the following commands to test access to the storage container that you wish to use:

```
delphix@demo-vm:~$ az login --identity
[
  {
    "environmentName": "AzureCloud",
    "homeTenantId": "<Tenant>",
    "id": "<ID>",
    "isDefault": true,
    "managedByTenants": [],
    "name": "Pay-As-You-Go",
    "state": "Enabled",
```

```
    "tenantId": "<Tenant>",
    "user": {
    "assignedIdentityInfo": "MSI",
    "name": "systemAssignedIdentity",
    "type": "servicePrincipal"
    }
  }
]

# List the storage containers
delphix@demo-vm:~$ az storage container list --account-name
<Your_Storage_Account_Name> --output table --auth-mode login
Name                                                        Lease Status    Last
Modified
----------------------------------------------------------  --------------
-------------------------
<Your_Storage_Container_Name>                                               2022-04-2
8T18:57:01+00:00

# List the contents of the new empty container
delphix@demo-vm:~$ - <Your_Storage_Account_Name> --container-name
<Your_Storage_Container_Name> --output table --auth-mode login

# Create a test file
delphix@demo-vm:~$ echo "This is a test" > test.txt

# Upload the file to the VM's storage container
delphix@demo-vm:~$ az storage blob upload --file test.txt --account-name
<Your_Storage_Account_Name> --container-name <Your_Storage_Container_Name> --output
table --auth-mode login
Finished[#############################################################]  100.0000%
Client_request_id                    Content_md5             Date
LastModified            Request_id                          Request_server_encrypted
Version
-----------------------------------  ------------------------
------------------------  ------------------------
------------------------------------  -------------------------  ----------
741a4bb4-c72a-11ec-95be-0022487dab78  /yKUEzaVYJiumlZCidG/Gw==  2022-04-28T19:36:09+0
0:00  2022-04-28T19:36:09+00:00  d254d8c6-a01e-00fe-1937-5bf5c0000000  True
2021-04-10

# List the contents of the storage container again
delphix@demo-vm:~$ az storage blob list --account-name <Your_Storage_Account_Name> --
container-name <Your_Storage_Container_Name> --output table --auth-mode login
Name    Blob Type   Blob Tier   Length  Content Type    Last Modified
Snapshot
--------  -----------  -----------  --------  --------------
-------------------------  ----------
test.txt  BlockBlob                15      text/plain      2022-04-28T19:36:09+00:00

# Download the storage blob
```

```
delphix@demo-vm:~$ az storage blob download  --account-name
<Your_Storage_Account_Name> --container-name <Your_Storage_Container_Name> --output
table --auth-mode login --name test.txt --file downloaded-test.txt
Finished[#############################################################]  100.0000%
Name    Blob Type   Blob Tier   Length  Content Type    Last Modified
Snapshot
--------  -----------  -----------  --------  --------------
-------------------------  ----------
test.txt  BlockBlob                 15      text/plain      2022-04-28T19:36:09+00:00
delphix@demo-vm:~$ cat downloaded-test.txt
This is a test

# Delete the storage blob
delphix@demo-vm:~$ az storage blob delete  --account-name <Your_Storage_Account_Name>
--container-name <Your_Storage_Container_Name> --output table --auth-mode login --
name test.txt
delphix@demo-vm:~$ az storage blob list --account-name <Your_Storage_Account_Name> --
container-name <Your_Storage_Container_Name> --output table --auth-mode login
```

Key based access

While managed identities are the recommended authentication method, users can also use static Storage Account access keys. Azure has detailed instructions here on how to manage Storage Account access keys here, including best practices on key security and rotation.

Elastic Data engines on AWS: Permissions

A role with at least the following permissions for that bucket assigned to the instance:

```
AWS: [
“s3:PutObject”,
“s3:GetObject”,
“s3:ListBucket”,
“s3:DeleteObject”
]
```

OCI: Permissions and Setup

Setup

OCI IAM has the capability to provide access to different services from an OCI instance, you don't need any API key or customer secrets or credentials to call the services.

OCI IAM Instance principals serves the same purpose as AWS IAM Role does.

Instance principals helps you in accessing OCI services from OCI instance.

For creating a dynamic group:

1. Login to the OCI Console
2. Go to Identity → Dynamic Groups → Create Dynamic Group
3. Specify Name and Description

4. You have a choice whether you want one of the matching rule to match or all the rules to be matched. You can select all the rules to be matched.

Creating Rules

When you create a dynamic group, you need to define some set of rules using which it would be determined which instances would be part of that dynamic group.

In this usecase, you can define 2 rules :

- Check the tag value on the instance and match it with some defined value.

```
tag.InstanceType.value='prod'
```

- All the instances present in a specific compartment.

```
instance.compartment.id = 'ocid1.compartment.oc1..aaaaaaxx'
```

Policy

After creating the group, we need to define a policy which specifies who can access the service. We need policy to tell OCI IAM to give access to the dynamic group which we created. For understanding OCI policies in detail How Policies Work (without Identity Domains).

For creating a policy:

- Log into OCI Console
- Go to Identities → Policies → Create PolicySpecify Name, Description, Compartment and policy statement. Here you are giving manage access to objects and not buckets. If needed, you can add another statement providing access to manage buckets also.
- Create the following policy:

```
Allow dynamic-group <dynamic group name> to manage buckets in compartment id
<compartment id> where target.bucket.name=<regex for bucket name>
Allow dynamic-group <dynamic group name> to inspect buckets in compartment id
<compartment id>
Allow dynamic-group <dynamic group name> to manage objects in compartment id
<compartment id> where target.bucket.name=<regex for bucket name>
```

Once all this is done, it would mean the instances launched in the specific compartment with specified tag would have access to manage object storage objects in that compartment.

Testing Permissions

You can use OCI CLI to demonstrate how you can use instance principals for authorising OCI requests without the need of managing any security credentials.

OCI CLI does not come preinstalled with different variants other than Oracle Linux 8 and Oracle Linux Cloud Developer 7. You can install it by following Quickstart.

Now you can make any call using oci cli by passing — auth instance_principal with the calls.

```
oci os bucket get -bn prodbucket --auth instance_principal
oci os object put -bn prodbucket --auth instance_principal --file object.txt
```

You could also define it as an environment variable.

```
OCI_CLI_AUTH=instance_principal
```

Instance principals could be used by OCI CLI and Python SDK/JAVA SDK. But does not works with Amazon S3 Compatibility API. For that you need customer secret keys.

Outbound connectivity

Web proxy

If a Web Proxy Server is necessary for your environment, select **Configure web proxy** and enter the hostname and credentials for that server.

Phone home

The support and phone home bundles contain metadata from the Delphix Engine, but do not include the customer data that has been ingested into the engine. Redaction of known PII data (i.e. names and email addresses for Delphix users) is done on-engine before bundles are uploaded, and again after bundles are uploaded to Delphix, to ensure that the latest redaction rules are applied to each bundle without requiring the engine to be continually upgraded. There may be some limited environment data in the bundle (i.e., IP addresses and database names) that are needed for debugging purposes. Support bundles are automatically deleted within 30 days after the support case is closed, or 30 days after upload, whichever comes later.

Enabling/disabling phone home

Enabling this option sends information to Delphix periodically over HTTPS (SSL). This data is securely managed by the internal team for product analysis and improvements. This feature requires a connection to the internet and will use the Web Proxy Server configuration.

Perform the following steps to enable/disable phone home.

1. Login to the Delphix Virtualization engine setup using the sysadmin credentials.
2. From the Outbound Connectivity widget, click **Modify**.
3. To enable phone home, select the checkbox before the **Enable phone home service** option. If enabled, this service will automatically send a stream of anonymous, non-personal metadata describing user interaction with the product's user interface.
4. To disable, deselect the checkbox before the **Enable phone home service** option.
5. Click **Save** to save for the settings.

User-click analytics

User-click analytics is a lightweight method to capture how users interact with Delphix product UIs, allowing Delphix to collect browser-based, user-click data. Delphix does not collect, transmit, or store any personally identifiable information (PII) such as email addresses, IP addresses, usernames, etc.

SMTP server

Select **Use an SMTP Server** and enter the server name or IP address to enable email notifications for events and alerts. When a critical fault occurs with the Delphix Engine, it will automatically send an email alert to the admin user. Make sure to configure the SMTP server so that alert emails can be sent to this user. See **System Faults** for more information.

Authentication

On the Authentication page, configure authentication protocols such as LDAP and SAML/SSO. See User and authentication management for further details.

LDAP

To avoid configuration issues, consult with the lightweight directory access protocol (LDAP) administrator before attempting to set up LDAP authentication of users for the Delphix engine. When configuring LDAP, provide an LDAP Server. Two authentication methods are currently supported: **SIMPLE** and **DIGEST_MD5**.

Select to **Protect LDAP traffic with SSL/TLS** if desired. This option requires an import of the server certificate. If LDAP has been set up as an authentication service for the Delphix Engine, add new users with LDAP as their authentication mechanism. For more information, see the User groups article.

SAML/SSO

To enable SAML/SSO, there are two properties to set:

1. **Audience Restriction**: The audience restriction must be set to the entity id configured in the Delphix Server via the Delphix Setup. Its default value is **https://Delphix Server ID**, where **Delphix Server ID** is a 36-character hexadecimal string of the form xxxxxxxx-xxxx-xxxx-xxxxxxxxxxxx.
   See Determining the Delphix server ID and host name for more on the Delphix Server ID. If the Delphix engine does not exist or is unreachable, enter a temporary value (such as `delphix-sp-id`) which must later be replaced by the actual Delphix Server ID.
2. **IdP Metadata**: an XML document that must be exported from the application created in the IdP. Paste its contents into the provided field.

Kerberos

The Kerberos page allows for Kerberos authentication to communicate between hosts connected with Delphix. Enabling this option will allow Kerberos key-based authentication when adding new environments to Delphix.

1. **Realm**: the domain over which a Kerberos authentication server has the authority to authenticate a user, host, or service.
2. **Principal**: a unique identity to which Kerberos can assign tickets.
3. **Keytab**: a file containing pairs of Kerberos principals and encrypted keys (which are derived from the Kerberos password).

Registration

As described in Registration management, registration allows Delphix Support to access the engine, properly diagnose, and identify any issues during support cases. If the Delphix Engine has access to the internet, auto-register the Delphix Engine with Delphix Support credentials in the **Online Registration** section.

If external connectivity is not immediately available, perform the manual registration.

1. Copy the Delphix Engine registration code displayed.
2. Transfer the Delphix Engine's registration code to a workstation with access to the external network Internet. For example, e-mail the registration code to an externally accessible email account.

3. On a machine with access to the internet, use a browser to navigate to the **Delphix Registration Portal** at http://register.delphix.com.
4. Log in with Delphix support credentials.
5. Paste the Registration Code.
6. Click **Register**.

The Delphix engine will work without registration, but it is recommended to register each engine as part of the setup. Failing to register the Delphix Engine will impact its supportability.

Summary

The final Summary section will enable a review of the configurations for each page in the setup tutorial. Confirm that everything looks correct, and click submit to complete the setup.

After Setup

- After the configuration is complete, the Delphix engine will restart and launch the browser-based Delphix Management application.
- After the Delphix Management application launches, the admin can log in using the initial default username **admin** and the initial default password **Delphix**. On the first login, there will be a prompt to change the initial password.

Customizing the Delphix engine system settings

Overview

This article describes how to customize the initial system setup requirements for memory, number of CPUs, storage disks, and network configuration. The OVA file used to install the Delphix Engine is configured for the minimum system requirements. These can be customized to match the capabilities of a specific system.

Procedure

1. Shut down the guest operating system and power off the Delphix Engine.
2. Under **Getting Started**, select **Edit Virtual Machine Settings**.
3. The system settings can now be customized.

Setting	**Options**
Memory Size	Set to 64GB or larger based on sizing analysis. In the **Resource Allocation** panel, ensure that **Reserve all guest memory** is checked. To adjust the resource allocation of a VM without rebooting it, Delphix supports "hot-plugging" CPUs and memory to ZFS. Please note that this is only available on ESX. For more information, refer to the Enabling hot-adding section on this page.
Number of CPUs	Allocate 8 vCPUs or more based on Delphix licensing. vCPUs should be fully reserved to ensure that the Delphix Engine does not compete for CPU cycles on an overcommitted host.
Supported Controllers	Delphix supports Paravirtual SCSI (PVSCSI) controllers or LSI Parallel controllers. A mix of different types of SCSI controllers is not supported within the engine. It is recommended to use PVSCSI controllers by default. Although PVSCSI supports high throughput with minimal processing cost, the performance improvements on Delphix Engine may vary from case to case.
Disks for Data Storage	Add virtual disks to provide storage for user data such as dSources and VDBs. The underlying storage must be redundant. Add a minimum of 150GB per storage disk. All virtual disks should be the same size and have the same performance characteristics. If using VMFS, use thick provisioned, lazy zeroed disks. To alleviate IO bottlenecks at the virtual controller layer, spread the virtual disks across all four virtual SCSI controllers.

Setting	Options
Data Storage Multipathing Policy	For EMC storage, the multipathing policy should always be set to round-robin (default for 5.x). Additionally, change the IO Operation Limit from the default of 1000 to 1. This should be strongly considered for other storage platforms as well. See VMware KB article EMC VMAX and DMX symmetrix storage array recommendations for optimalperformance on VMware ESXi/ESX.
Network	The network configuration is set to have a VMXNET3 network adapter. VMXNET3 is a tuned network interface that is included with the VMtools provided in the OVA file. It will be assigned to VM Network. **JUMBO Frames** VMXNET3 supports Ethernet jumbo frames, this can be used to maximize throughput and minimize CPU utilization. **Adding Link Aggregation via VMware NIC Teaming** To increase throughput or for failover, add multiple physical NICs to the vSwitch that is connected to the Delphix Engine. To increase throughput, NIC Teaming must use the Route Based on IP Hash protocol for load balancing. **Dedicate Physical NICs to the Delphix Engine** For best performance, assign the Delphix Engine to network adapters that are used exclusively by Delphix.

Enabling hot-adding

To manually enable hot-adding in the VCenter console, use the following steps.

1. Select the Delphix Engine.
2. Right-click on the engine and select **Edit settings**.
3. Edit settings:
    a. To enable CPU hot-add, navigate to the CPU section and select the **CPU Hot Plug** checkbox.
    b. To enable memory hot-add, navigate to the memory section and select the **Memory Hot Plug** checkbox.
4. Modify the appropriate setting to the new value; note that only expansion of CPU and memory is supported, and attempts to reduce the values available to the Delphix Engine will fail.
5. Click **Ok** to confirm the changes.

Newly-added memory is not automatically reflected in system information APIs nor in the UI. Restarting the management service is needed to refresh this information.

For customers migrating from versions before 6.0, an extra step is required to enable CPU/memory hot-plug.

1. After migrating, turn off the VM.
2. Open **VM Settings**.

3. Select **VM Options**.
4. Under **General**, update the guest OS to Ubuntu (64-bit).
5. Under **Advanced**, select **Edit Configuration**.
6. Set **disk.EnableUUID** to TRUE.
7. Follow normal hot-plug enable steps.

> ⚠ For customers that are upgrading their existing engines from versions below 6.0.6.0, two vSphere properties need to be enabled manually after the upgrade from vSphere in order to use this feature. The properties are **cputHotAddEnabled** and **memoryHotAddEnabled**.
> When upgrading to 6.0 from 5.3, the guest OS and version in vSphere also needs to be updated to Ubuntu, in addition to the above two properties. Without this, it has been noticed that hot-adding memory is not allowed from vSphere, even if memoryHotAddEnabled is enabled.

Post-requisites

- After making any changes to the system settings, power on the Delphix Engine again and proceed with the initial system configuration as described in Setting up network access to the Delphix engine.

Installing an OVA or AMI

Overview

This article outlines how to install the Delphix Engine using an OVA or AMI file.

Procedure to install an OVA

Use the Delphix-supplied OVA file to install the Delphix Engine. The OVA file is configured with many of the minimum system requirements. The underlying storage for the install is assumed to be redundant SAN storage.

1. Download the OVA file from https://download.delphix.com. A support login is needed from the Delphix representative or welcome letter.
    a. Navigate to the Delphix Product Releases/<Current Version>/Appliance Images page.
2. Login using the vSphere client to the vSphere server (or vCenter Server) where the Delphix Engine will be installed.
3. In the vSphere Client, click **File**.
4. Select **Deploy OVA Template**.
5. Browse to the OVA file.
6. Click **Next**.
7. Select a **hostname** for the Delphix Engine. This hostname will also be used in configuring the Delphix Engine network.
8. Select the **data center** where the Delphix Engine will be located.
9. Select the **cluster** and the **ESX host**.
10. Select one (1) **data store** for the **Delphix OS**. This datastore can be **thin-provisioned** and must have enough free space to accommodate the 127GB comprising the Delphix operating system.
11. Select four (4) or more **data stores** for Database Storage for the Delphix Engine. The Delphix Engine will stripe all of the Database Storage across these VMDKs, so for optimal I/O performance, each VMDK must be equal in size and be configured **Thick Provisioned - Eager Zeroed**. Additionally, these VMDKs should be distributed as evenly as possible across all four SCSI I/O controllers.
12. Select the **virtual network** to be used. If using multiple physical NICs for link aggregation, vSphere NIC teaming must be used. Do not add multiple virtual NICs to the Delphix Engine itself. The Delphix Engine should use a single virtual network. For more information, see Optimal Network Architecture for the Delphix Engine.
13. Click **Finish**. The installation will begin and the Delphix Engine will be created in the location specified.
14. Once the installation has completed, power on the Delphix Engine and proceed with the initial system configuration as described in Setting Up Network Access to the Delphix Engine.

Procedure to install an AMI

Use the Delphix-supplied AMI file to install the Delphix Engine.

1. On the Delphix download site, click the AMI to be shared and accept the Delphix License agreement. Alternatively, follow a link given by the Delphix solutions architect.
2. On the **Amazon Web Services Account Details** form presented:
    a. Enter the **AWS Account Identifier**, which can be found here: https://console.aws.amazon.com/billing/home?#/account. To use the **Gov Cloud AWS Region**, be sure to enter the ID for the AWS Account which has Gov Cloud enabled.
    b. Select which **AWS Region** the AMI should be shared in. To have the AMI shared in a different region, contact the Delphix account representative for the proper arrangements.
3. Click **Share.** The Delphix Engine will appear in the list of AMIs in AWS momentarily.

Validating host deployment with host Checker

Overview

Delphix has developed a hostchecker script that contains standardized checks for source and target hosts - these checks generally fall into three buckets:

- OS and Host permissions/access
- Network Port Checks
- DB-specific functionality

OS and Host permissions/access and network port checks can (and should) be performed prior to Delphix installation to ensure a smooth deployment.

Each DB should have a specifically associated hostchecker, there is detailed documentation on the DB-specific hostchecker page.

Procedure

1. Download the appropriate **HostChecker tarball** for the engine from https://download.delphix.com/. Tarballs follow the naming convention "hostchecker_<OS>_<processor>.tar". For example, if validating a linux x86 host, download the hostchecker_linux_x86.tar tarball.
2. Create a working directory and extract the **HostChecker files** from the **HostChecker tarball**

```
mkdir dlpx-host-checker
cd dlpx-host-checker/
tar -xf hostchecker_linux_x86. tar
```

3. Run the `sh` script contained within:

```
sh hostchecker.sh
```

This will extract the JDK included in the tarball (if necessary) and invoke the HostChecker.

```
ora10205@bbdhcp: /home/ora10205/hostchecker- > sh hostchecker.sh
Extracting the JDK from the tarball jdk-6u45-linux-i586. tar .gz.
```

Do not run the HostChecker as root; this will cause misleading or incorrect results from many of the checks.

4. Select which **checks** to run. Note that it is possible to run checks without spawning the interface. Enter `--help` to get a list of arguments that can pass to the HostChecker.
5. As the checks are made, enter the requested **arguments**.
6. Read the output of the check. The general format is that severity increases farther down the output. First comes informational output, then warnings, then errors. If you see a message that starts with `Internal Error`, forward it to Delphix Support immediately. This represents a potential bug in the HostChecker, and not necessarily a problem with your environment.
7. Error or warning messages will explain any possible problems and how to address them. Resolve the issues that the HostChecker describes. Do not be surprised or undo your work if more errors appear the next time you run HostChecker, because the error you just fixed may have been masking other problems.
8. Repeat steps 3 - 7 until all the checks return no errors or warnings.

Deployment for VMware

Overview

This article outlines the requirements for deploying the Delphix Engine on VMware (including supported versions and instance configurations), as well as recommended configuration parameters for optimal performance.

The Delphix Engine is an intensive platform, both from a network and a storage perspective. If the Delphix Engine competes with other virtual machines on the same host for resources it will result in increased latency for all operations. As such, it is crucial that the ESXi host is not over-subscribed, as this eliminates the possibility of a lack of resources for the Delphix Engine. This includes allowing a percentage of CPU resources for the hypervisor itself as it can de-schedule an entire VM if the hypervisor is needed for managing IO or compute resources.

Supported ESX versions

Requirements	Notes
• VMware ESXi 8.0, 8.0 U1 • VMware ESXi 7.0, 7.0 U1, 7.0 U2, 7.0 U3 • VMware ESXi 6.7 U3 • VMware ESXi 6.5 U1, 6.5 U3	• If a minor release version is listed as supported, then all patch releases applicable to that minor release are supported. • Public cloud deployments such as VMware cloud on AWS and Azure VMware solution are supported. • While ESXi 7.0 U3 is supported, this is only for patch releases 3c and newer, due to critical issues in the earlier patch versions.

Virtual machine hardware versions

The Delphix Engine VM is distributed as an OVA for VMware, and is configured with the hardware version corresponding to the lowest ESXi version that release is qualified for. As the Engine is upgraded, the ESXi versions supported may change, but the hardware version may remain the same based on the original deployment.

Users who wish to upgrade the VM hardware version for compatibility, enhanced feature support, or other reasons may feel free to do so, with consideration for any compatibility concerns that may arise in environments where multiple ESXi versions are present, as an upgraded hardware version can affect other VMware operations (vMotion, etc). VMware support and documentation should be consulted before committing any hardware version upgrade for a guest VM, but Delphix does not maintain any version requirements in this regard.

Virtual CPUs

Requirements	Notes
8 vCPUs	• CPU resource shortfalls can occur both on an over-committed host as well as competition for host resources during high IO utilization. • CPU reservations are **strongly recommended** for the Delphix VM so that Delphix is guaranteed the full complement of vCPUs even when resources are overcommitted. • It is suggested to use a single core per socket unless there are specific requirements for other VMs on the same ESXi host.
Never allocate all available physical CPUs to virtual machines	• CPU for the ESXi Server to perform hypervisor activities must be set aside before assigning vCPUs to Delphix and other VMs. • We recommend that a minimum of 8-10% of the CPUs available are reserved for hypervisor operation. (e.g. 12 vCPUs on a 128 vCore system).

Memory

Requirements	Notes
128 GB vRAM (recommended) 64GB vRAM (minimum)	• The Delphix Engine uses its memory to cache database blocks. More memory will provide better read performance. • Memory reservations are **required** for the Delphix VM. The performance of the Delphix Engine will be significantly impacted by the over-commitment of memory resources in the ESX Server. • Reservations ensure that the Delphix Engine will not be forced to swap pages during times of memory pressure on the host. A swapped page will require orders of magnitude more time to be brought back to physical memory from the ESXi swap device.

Requirements	Notes
Never allocate all the available physical memory to virtual machines.	• The default ESX minimum free memory requirement is 6% of the total RAM. When free memory falls below 6%, ESX starts swapping out the Delphix guest OS. Delphix recommends leaving about 8-10% free to avoid swapping. • For example, when running on an ESX Host with 512GB of physical memory, no more than 470GB (92%) should be allocated to the Delphix VM (and all other VMs on that host).
Memory for the ESX Server to perform hypervisor activities must be set aside before assigning memory to Delphix and other VMs.	Failure to ensure sufficient memory for the host can result in a hard memory state for all VMs on the host which will result in a block for memory allocations.

Network

Requirements	Notes
The ova is pre-configured to use one virtual ethernet adapter of type VMXNET 3.	• Jumbo frames are highly recommended to reduce CPU utilization, decrease latency and increase network throughput. (typically 10-20% throughput improvement) • If additional virtual network adapters are desired, they should also be of type VMXNET 3.
A 10GbE NIC in the ESX Server is recommended.	• For VMs having only gigabit networks, it is possible to aggregate several physical 1GbE NICs together to increase network bandwidth (but not necessarily to reduce latency). Refer to the VMware Knowledge Base article NIC Teaming in ESXi and ESX. However, it is not recommended to aggregate NICs in the Delphix Engine VM.
If the network load in the ESX Server hosting the Delphix engine VM is high, dedicate one or more physical NICs to the Delphix Engine.	• Adding NICs only works if VMs are discovered using different interfaces. The NFS/iSCSI mounts will only use the network associated with the discovery. • See General Network and Connectivity Requirements for information about specific port configurations, and Network Performance Configuration Options for information about network performance tuning

To bootstrap a networking configuration to reach the Delphix Engine, after deploying it into your environment for the first time, you can use one of the following options:

- DHCP

- Cloud-init for public clouds
- You can login to the serial console to configure networking via the CLI
- OVF guest customizations to pass in network configuration to the VM before it has a network connection. For the customization steps:
    - Type the name of the profile and click **Next**
    - The domain will be ignored by Delphix. Click **Next**
    - The time zone setting will be ignored by Delphix, Click **Next**
    - Delphix doesn't allow scripts, click **Next**
    - Manually select custom settings, select a **NIC** and click **Edit**
        - Provide **Netmask** and **Gateway**
        - Select one of the options for IP
        - Confirm changes and click **Next**
    - Add **DNS**
    - Confirm creation of profile, click **Finish**
    - You can use this template to clone a VM

SCSI controller

Requirements	Notes
PVSCSI (default)/ LSI Logic Parallel	When adding virtual disks make sure that they are evenly distributing the load across the maximum of 4 virtual SCSI controllers. Spreading the disks across available SCSI controllers evenly will ensure optimal IO performance from the disks. For example, a VM with 4 SCSI controllers and 8 virtual disks should distribute the disks across the controllers as follows: disk0 = SCSI(0:0) - System Disk on Controller 0 Port 0 (ignore for purposes of load balancing) disk1 = SCSI(0:1) - Data Disk on Controller 0 Port 1 disk2 = SCSI(1:1) - Data Disk on Controller 1 Port 1 disk3 = SCSI(2:1) - Data Disk on Controller 2 Port 1 disk4 = SCSI(3:1) - Data Disk on Controller 3 Port 1 disk1 = SCSI(0:2) - Data Disk on Controller 0 Port 2 disk2 = SCSI(1:2) - Data Disk on Controller 1 Port 2 disk3 = SCSI(2:2) - Data Disk on Controller 2 Port 2 disk4 = SCSI(3:2) - Data Disk on Controller 3 Port 2 ⓘ For load purposes, we generally focus on the DB storage and ignore the controller placement of the system disk.

General storage

VMware offers options for disk storage, which include "Independent - Persistent" and "Independent - Non-persistent". The non-persistent setting is catastrophic to the Delphix platform when the VM reboots, thus, **Independent - Persistent** is required for installation.

Requirements	Notes
Storage used for Delphix must be provisioned from storage that provides data protection.	For example, using RAID levels with data protection features, or equivalent technology. The Delphix engine product does not protect against data loss originating at the hypervisor or SAN layers. For more information refer to, Optimal Storage Configuration Parameters for the Delphix Engine.

Delphix storage options

There are three types of data that Delphix stores on disk, which is:

1. **Delphix VM Configuration Storage:** stores data related to the configuration of the Delphix VM. VM Configuration Storage includes the VMware ESX configuration data as well as log files.
2. **Delphix Engine System Disk Storage:** stores data related to the Delphix Engine system data, such as the Delphix .ova settings.
3. **Database Storage:** stores data used by Delphix objects such as dSources and virtual databases (VDBs).

Delphix VM configuration storage

The Delphix VM configuration should be stored in a VMFS volume (often called a "datastore").

Requirements	Notes
The VMFS volume should have enough available space to hold all ESX configuration and log files associated with the Delphix Engine.	If a memory reservation is not enabled for the Delphix Engine (in violation of memory requirements stated above), then space for a paging area equal to the Delphix Engine's VM memory must be added to the VMFS volume containing the Delphix VM configuration data.

Delphix engine system disk storage

The VMFS volume must be located on shared storage in order to use vMotion and HA features.

Requirements	Notes
The Delphix Engine system disk should be stored in a VMDK.	• The VMDK for the Delphix Engine System Disk Storage is often created in the same VMFS volume as the Delphix VM definition. In that case, the datastore must have sufficient space to hold the Delphix VM Configuration, the VDMK for the system disk, and a paging area if a memory reservation was not enabled for the Delphix Engine.
The Delphix .ova file is configured for a 127GB system drive.	• The VMFS volume where the .ova is deployed should, therefore, have at least 127GB of free space prior to deploying the .ova.

Database storage

Option 1: Block Storage for database storage

Shared storage is required in order to use vMotion and HA features. In addition to making sure the latest VMware patches have been applied, check with the organizations hardware vendor for updates specific to the hardware configurations. VMDKs (Virtual Machine Disks) or RDMs (Raw Device Mappings) operating in virtual compatibility mode can be used for database storage.

Requirements	Notes
A minimum of 4 VMDKs or RDMs should be allocated for database storage.	• Allocating a minimum of 4 VMDKs or RDMs for database storage enables the Delphix File System (DxFS) to make sure that its file systems are always consistent on disk without additional serialization. This also enables the Delphix Engine to achieve higher I/O rates by queueing more I/O operations to its storage.

If using VMDKs:
• Each VMDK should be the only VMDK in its VMFS volume
• The VMFS volumes should be assigned to dedicated physical LUNs on redundant storage.
• The VMDKs should be created with the Thick Provision Lazy Zeroed option.
	• Provisioning VMDKs from isolated VMFS volumes on dedicated physical LUNs:
	  • Reduces contention for the underlying physical LUNs
	  • Eliminates contention for locks on the VMFS volumes from other VMs and/or the ESX Server Console
	  • Enables higher availability of the Delphix VM by allowing vSphere to vMotion the VM to a different ESX host in the event of a failure of the Delphix ESX host

Requirements	Notes
The quantity and size of VDMKs or RDMs assigned must be identical across all 4 controllers	• If the underlying storage array allocates physical LUNs by carving them from RAID groups, the LUNs should be allocated from different RAID groups. This eliminates contention for the underlying disks in the RAID groups as the Delphix engine distributes IO across its storage devices.
The physical LUNs used for VMFS volumes and RDMs should be of the same type in terms of performance characteristics such as latency, RPMs, and RAID level.	• The total number of disk drives that comprise the set of physical LUNs should be capable of providing the desired aggregate I/O throughput (MB/sec) and IOPS (Input/Output Operations per Second) for all virtual databases that will be hosted by the Delphix Engine.
The physical LUNs used for VMFS volumes can be thin-provisioned in the storage array.	• If the storage array allocates physical LUNs from storage pools comprising dozens of disk drives, the LUNs should be distributed evenly across the available pools.
For best performance, the LUNs used for RDMs should not be thin-provisioned in the storage array but should be thick-provisioned with a size equal to the amount of storage that will be initially allocated to the Delphix Engine. The RDM can be expanded in the future when more storage is needed.	• Using thin-provisioned LUNs in the storage array for VMFS volumes can be useful if adding storage to the Delphix engine in the future is anticipated. In this case, the LUNs should be thin-provisioned with a size larger than the amount of storage that will be initially allocated to the Delphix Engine. When appropriate, to add more storage to the Delphix engine, use vSphere to expand the size of the VMDKs. Be sure to specify that the additional storage is also thick-provisioned and eager-zeroed.

In addition to making sure the latest VMware patches have been applied, check with the organization's hardware vendor for updates specific to the hardware configurations.

Option 2: Elastic Data where Object storage is used for database storage and block storage is used for cache.

We support Elastic Data with on prem object storage for the database storage. Traditional disks as cache are used to reduce latencies for frequently read data and as temporary storage for synchronous writes before the writes are sent to object storage.

- For on prem object storage we currently support storage vendors that confirm to the following
    - s3 REST API compatibility
    - strong read-after-write consistency
    - supports s3 key id and secret access key authentication
    - perpetual key support
- On-premise object storage may require a security certificate to create secure connections between the Delphix engine and object storage. If a certificate is required, then it must be installed on the Delphix engine prior to configuring the storage. Refer to Certificate management for more information.
- Refer to Initial setup for additional details for setting up the Elastic Data Engine.

Additional VMware configuration notes

- Running Delphix inside of vSphere is supported.
- Using vMotion on a Delphix VM is supported.
- Device passthrough is not supported.

Procedure to install an OVA

Use the Delphix-supplied OVA file to install the Delphix Engine. The OVA file is configured with many of the minimum system requirements. The underlying storage for the install is assumed to be redundant SAN storage.

1. Download the OVA file from https://download.delphix.com. You will need a support login from your sales team or a welcome letter.
    a. Navigate to the Delphix Product Releases/<Current Version>/Appliance Images page.
2. Login using the vSphere client to the vSphere server (or vCenter Server) where you want to install the Delphix Engine.
3. In the vSphere Client, click **File**.
4. Select **Deploy OVA Template**.
5. Browse to the OVA file.
6. Click **Next**.
7. Select a **hostname** for the Delphix Engine.
   This hostname will also be used in configuring the Delphix Engine network.
8. Select the **data center** where the Delphix Engine will be located.
9. Select the **cluster** and the **ESX host**.
10. Select one (1) **data store** for the **Delphix OS**. This datastore can be **thin-provisioned** and must have enough free space to accommodate the 127GB comprising the Delphix operating system.
11.
    a. For traditional block storage engines
        i. Select four (4) or more **data stores** for Database Storage for the Delphix Engine. The Delphix Engine will stripe all of the Database Storage across these VMDKs, so for optimal I/O performance, each VMDK must be equal in size and be configured **Thick Provisioned - Eager Zeroed**. Additionally, these VMDKs should be distributed as evenly as possible across all four SCSI I/O controllers.
    b. For Elastic data
        i. The VMDK will be used as cache to reduce latencies for frequently read data and as temporary storage for synchronous writes before the writes are sent to object storage. For optimal I/O performance, each VMDK must be equal in size and be configured **Thick Provisioned - Eager Zeroed.** Make sure the disks satisfy the I/O needs of the engine. Eg: At 500 IOPS per 1GiB ratio, a 32 GiB volume can be configured to have the 16K IOPS limit. Two devices would be sufficient for the instance that requires 30K IOPS. You can always add disks later if the cache is insufficient. You can only reduce the cache by removing disks, so if you think you are over provisioning the cache, increase the number of disks used and reduce the size of each disk so that removal is possible at a later point in time.
12. Select the **virtual network** you want to use.
    If using multiple physical NICs for link aggregation, you must use vSphere NIC teaming. Do not add multiple virtual NICs to the Delphix Engine itself. The Delphix Engine should use a single virtual network. For more information, see Optimal network architecture for the Delphix engine.
13. Click **Finish**.
    The installation will begin and the Delphix Engine will be created in the location you specified.
14. Once the installation has completed, power on the Delphix Engine and proceed with the initial system configuration as described in Setting up network access to the Delphix engine

Deployment for KVM

Overview

This article outlines the requirements for deploying the Delphix Engine via Linux KVM, as well as recommended configuration parameters.

Contact a Delphix representative to request this capability. Delphix will assist in assuring that all KVM requirements are met to successfully run a Delphix Engine with the most appropriate configurations for the use case.

The Delphix Engine is intensive from both a network and storage perspective. If the Delphix Engine competes with other virtual machines on the same host for resources it will result in increased latency for all operations. As such, it is crucial that your KVM host is not over-subscribed, as this eliminates the possibility of a lack of resources for the Delphix Engine. This includes allowing a percentage of CPU resources for the hypervisor itself as it can de-schedule an entire VM if the hypervisor is needed for managing I/O or compute resources.

The KVM ecosystem includes many versions/variations. Delphix is announcing general support for KVM, but will forgo any formal certification for specific versions and rather will focus on declared support for specific Linux Kernels.

Pre-requisites

1. The KVM provider must explicitly state that the Ubuntu 18.04/20.04 kernel is supported by their variation of KVM.

Most, if not all, KVM hypervisor providers should have public facing documentation which include support matrices for guest OS compatibility. If the KVM hypervisor provider does not explicitly state support for the Ubuntu kernel versions used in the supported Linux Delphix versions (either 18.04 currently, or 20.04 in the future), Delphix cannot support the KVM provider.

As an example, RedHat Linux (a KVM hypervisor provider), provides a publicly accessible guest operating system support matrix. As of December 2021, this matrix does not include support for any Ubuntu guest OS, and therefore Delphix would also not support deployment on RedHat Linux KVM. Oracle Linux (a different KVM hypervisor provider) also provides a publicly accessible guest operating system support matrix, which declares support for Ubuntu 16.04, 18.04, and 20.04. In this case, since the hypervisor provider declares support for the Delphix Ubuntu kernel(s), Delphix will support deployment on Oracle Linux KVM.

2. A check of the Delphix appliance on the variation/version of KVM.

It is **required** that a Delphix representative works with the organization to check the Delphix appliance (and/or use case) on the specified KVM variation/version. This check will ensure that the software is compatible.

Virtual CPUs

Requirements	Notes
8 vCPUs	• CPU resource shortfalls can occur on an over-committed host as well as competition for host resources during high IO utilization. • CPU reservations are strongly recommended for the Delphix VM so that Delphix is guaranteed the full complement of CPUs even when resources are overcommitted. • It is suggested to use a single core per socket unless there are specific requirements for other VMs on the same KVM host.
Never allocate all available physical CPUs to virtual machines.	• A CPU for the KVM Server to perform hypervisor activities must be set aside before assigning vCPUs to Delphix and other VMs. • We recommend that a minimum of 8-10% of the CPUs available are reserved for hypervisor operation. (e.g. 12 vCPUs on a 128 vCore system).

Memory

Requirements	Notes
128 GB vRAM (recommended) 64GB vRAM (minimum)	• The Delphix Engine uses its memory to cache database blocks. More memory will provide better read performance. • Memory reservations are required for the Delphix VM. The performance of the Delphix Engine will be significantly impacted by the over-commitment of memory resources in the KVM Server. • Reservations ensure that the Delphix Engine will not be forced to swap pages during times of memory pressure on the host. A swapped page will require orders of magnitude more time to be brought back to physical memory from the KVM swap device.
Memory for the KVM Server to perform hypervisor activities must be set aside before assigning memory to Delphix and other VMs.	• Failure to ensure sufficient memory for the host can result in a hard memory state for all VMs on the host which will result in a block for memory allocations.

Network

Requirements	Notes
Virtual ethernet adapter requirements.	• Jumbo frames are highly recommended to reduce CPU utilization, decrease latency, and increase network throughput (typically 10-20% throughput improvement). • A 10GbE NIC/network is recommended for performance dSource and VDB operations.
If the network load in the KVM Server hosting the Delphix engine VM is high, dedicate one or more physical NICs to the Delphix Engine.	• Adding NICs only works if VMs are discovered using different interfaces. The NFS/iSCSI mounts will only use the network associated with the discovery. • See General Network and Connectivity Requirements for information about specific port configurations, and Network Performance Configuration Options for information about network performance tuning.

SCSI controller

Notes
When adding virtual disks, make sure that they are evenly distributing the load across the maximum of 4 virtual SCSI controllers. Spreading the disks across available SCSI controllers evenly will ensure optimal IO performance from the disks. For example, a VM with 4 SCSI controllers and 8 virtual disks should distribute the disks across the controllers as follows:
disk0 = SCSI(0:0) - System Disk on Controller 0 Port 0 (ignore for purposes of load balancing)
disk1 = SCSI(0:1) - Data Disk on Controller 0 Port 1
disk2 = SCSI(1:1) - Data Disk on Controller 1 Port 1
disk3 = SCSI(2:1) - Data Disk on Controller 2 Port 1
disk4 = SCSI(3:1) - Data Disk on Controller 3 Port 1
disk5 = SCSI(0:2) - Data Disk on Controller 0 Port 2
disk6 = SCSI(1:2) - Data Disk on Controller 1 Port 2
disk7 = SCSI(2:2) - Data Disk on Controller 2 Port 2

Notes
disk8 = SCSI(3:2) - Data Disk on Controller 3 Port 2
For load purposes, we generally focus on the DB storage and ignore the controller placement of the system disk.

General storage

Delphix recommends using a minimum of four disks to run your Delphix Engine. One disk is used for the Delphix File System (DxFS) to ensure that its file systems are always consistent on disk without additional serialization. The other three or more equally sized disks will be used for data storage. This also enables the Delphix Engine to achieve higher I/O rates by queueing more I/O operations to its storage.

Requirements	Notes
Storage used for Delphix must be provisioned from storage that provides data protection.	For example, using RAID levels with data protection features, or equivalent technology. The Delphix Engine does not protect against data loss originating at the hypervisor or SAN layers. For more information refer to, Optimal storage configuration parameters for the Delphix engine.

Deployment for Hyper-V

Overview

This article outlines the requirements for deploying the Delphix Engine on Hyper-V (including supported versions and instance configurations), as well as recommended configuration parameters for optimal performance.

Contact a Delphix representative to request this capability. Delphix will assist in assuring that all Hyper-V requirements are met to successfully run a Delphix Engine with the most appropriate configuration for the use case.

The Delphix Engine is intensive both from a network and a storage perspective. If the Delphix Engine competes with other virtual machines on the same host for resources it will result in increased latency for all operations. As such, it is crucial that your Hyper-V host is not over-subscribed, as this eliminates the possibility of a lack of resources for the Delphix Engine. This includes allowing a percentage of CPU resources for the hypervisor itself as it can de-schedule an entire VM if the hypervisor is needed for managing I/O or compute resources.

Supported versions

- Hyper-V Version: 10.0 and later
- Virtual Machine: Gen 1 only is supported

Virtual CPUs

Requirements	**Notes**
8 vCPUs	• CPU resource shortfalls can occur on an 0ver-committed host as well as competition for host resources during high IO utilization. • CPU reservations are strongly recommended for the Delphix VM so that Delphix is guaranteed the full complement of CPUs even when resources are overcommitted. • It is suggested to use a single core per socket unless there are specific requirements for other VMs on the same Hyper-V host.
Never allocate all available physical CPUs to virtual machines.	• CPU for the Hyper-V Server to perform hypervisor activities must be set aside before assigning vCPUs to Delphix and other VMs • We recommend that a minimum of 8-10% of the CPUs available are reserved for hypervisor operation. (e.g. 12 vCPUs on a 128 vCore system).

Memory

Requirements	Notes
128 GB vRAM (recommended) 64GB vRAM (minimum)	• The Delphix Engine uses its memory to cache database blocks. More memory will provide better read performance. • Memory reservations are required for the Delphix VM. The performance of the Delphix Engine will be significantly impacted by the over-commitment of memory resources in the Hyper-V Server. • Reservations ensure that the Delphix Engine will not be forced to swap pages during times of memory pressure on the host. A swapped page will require orders of magnitude more time to be brought back to physical memory from the Hyper-V swap device.
Memory for the Hyper-V Server to perform hypervisor activities must be set aside before assigning memory to Delphix and other VMs.	Failure to ensure sufficient memory for the host can result in a hard memory state for all VMs on the host which will result in a block for memory allocations.

Network

Requirements	Notes
Virtual ethernet adapter requirements.	• SR-IOV is recommended for all virtual ethernet adapters that will be used for Delphix data I/O. • Jumbo frames are recommended. • A 10GbE NIC in the Hyper-V Server is recommended.
If the network load in the Hyper-V Server hosting the Delphix engine VM is high, dedicate one or more physical NICs to the Delphix Engine.	• Adding NICs only works if VMs are discovered using different interfaces. The NFS/iSCSI mounts will only use the network associated with the discovery. • See General network and connectivity requirements for information about specific port configurations, and Network performance configuration options for information about network performance tuning.

SCSI controller

Requirements	Notes

lDE Controller for the boot drive
 	
SCSI Controller for database storage
	Per Hyper-V Storage I/O Performance Tuning Guidelines; For optimal performance, it is recommended that you attach multiple disks to a single virtual SCSI controller and create additional controllers only as they are required to scale the number of disks connected to the virtual machine.
	
	For example, a VM with 3 virtual disks should distribute the disks across the single SCSI controller as follows:
	
	• IDE Controller 1
	  • Boot Drive
	• SCSI Controller
	  • disk1
	  • disk2
	  • disk3
	
	For load purposes, we generally focus on the DB storage and ignore the controller placement of the system disk.

General storage

Delphix recommends using a minimum of four disks to run your Delphix Engine. One disk is used for the Delphix File System (DxFS) to ensure that its file systems are always consistent on disk without additional serialization. The other three or more equally sized disks will be used for data storage. This also enables the Delphix Engine to achieve higher I/O rates by queueing more I/O operations to its storage.

Requirements	Notes
Storage used for Delphix must be provisioned from storage that provides data protection.	For example, using RAID levels with data protection features, or equivalent technology. The Delphix Engine does not protect against data loss originating at the hypervisor or SAN layers. For more information refer to, Optimal storage configuration parameters for the Delphix engine.

Delphix storage options

There are two types of data that Delphix stores on disk which must be stored on NTFS volumes:

1. Delphix Engine System Disk Storage: This is the system boot drive.
2. Database Storage: stores data used by Delphix objects such as dSources and virtual databases (VDBs).

Delphix engine system disk storage

Requirements	Notes
The Delphix Engine disks must be stored on NTFS volume(s).	The volume for the Delphix Engine System Disk Storage is often created on the same volume as the Delphix VM definition. In that case, the volume must have sufficient space to hold the Delphix VM Configuration, the virtual disk for the system disk, and a paging area if a memory reservation was not enabled for the Delphix Engine.
The Delphix .vhdx file is configured for a 128GB system drive.	The volume where the .vhdx is deployed should, therefore, have at least 127GB of free space prior to deploying the .vhdx.

Database storage

In addition to making sure the latest Hyper-V patches have been applied, check with your hardware vendor for updates specific to your hardware configuration. VHDXs (virtual machine disks).

Requirements	Notes
A minimum of 3 VHDXs should be allocated for database storage.	Allocating a minimum of 3 VHDXs for database storage enables the Delphix File System (DxFS) to make sure that its file systems are always consistent on disk without additional serialization. This also enables the Delphix Engine to achieve higher I/O rates by queueing more I/O operations to its storage.
If using VHDXs: • Each VHDX should be the only VHDX on its NTFS volume • The VHDX volumes should be assigned to dedicated physical LUNs on redundant storage.	Provisioning VHDXs from isolated volumes on dedicated physical LUNs: • Reduces contention for the underlying physical LUNs • Eliminates contention for locks on the volumes from other VMs and/or the Hyper-V Server Console
	If the underlying storage array allocates physical LUNs by carving them from RAID groups, the LUNs should be allocated from different RAID groups. This eliminates contention for the underlying disks in the RAID groups as the Delphix engine distributes IO across its storage devices.

Requirements	Notes
The physical LUNs used for NTFS volumes should be of the same type in terms of performance characteristics such as latency, RPMs, and RAID levels.	The total number of disk drives that comprise the set of physical LUNs should be capable of providing the desired aggregate I/O throughput (MB/sec) and IOPs (Input/Output Operations per Second) for all virtual databases that will be hosted by the Delphix Engine
Disk types supported by the physical LUNs used for NTFS volumes are: Fixed-size and Dynamically expanding.	If the storage array allocates physical LUNs from storage pools comprising dozens of disk drives, the LUNs should be distributed evenly across the available pools.

Procedure for deploying with Hyper-V

Overview

This article outlines the procedure for deploying the Delphix Engine in Hyper-V.

Creating a Delphix engine VM

1. Download the image from https://download.delphix.com and copy it to a VM directory.
2. Start the Hyper-V Manager and specify **Name** and **Location**, then select **Next**.
3. Select **Generation 1**, configure memory and then select **Next**.
   Memory: 64 GB (minimum), 128 GB (recommended)
4. Set up Networking by selecting **vNIC**, then select **Next**.
5. Attach the downloaded image as a boot disk. Create a unique boot disk for each image. Please note that boot disks cannot be shared.
   a. Use an existing virtual hard disk.
   b. Browse to the location of VM.
   c. Select the Image.
6. Select **Finish**, the VM will appear in the inventory.

Customize the VM by selecting **settings**

1. Delphix recommends having the IDE as the first device to boot from (under BIOS setting).
2. Adjust the number of CPUs (min 8).
3. Add a minimum of four equal-sized **Hard Drives** for the database storage. Use VHDX formatted disks. Recommend Fixed Size disk type. Please note that Differencing Disk Types are not supported.
4. Connect to the console and start the VM.
5. Once the installation has been completed, power on the Delphix Engine and proceed with the initial system configuration as described in Setting up network access to the Delphix engine.

Deployment for AWS EC2

Overview

This article outlines the virtual machine requirements, including memory and data storage, for deploying the Delphix Engine on Amazon EC2 (Elastic Cloud Compute). Once the requirements listed on this page are reviewed, refer to the next articles in the AWS EC2 Deployment topic:

- Prerequisites to Deploying in AWS
- Procedure for Deploying in AWS

Instance types

The following is a list of instance types that are supported to deploy Delphix on AWS EC2. Delphix periodically certifies new instance types, which will be added to the list here.

Requirements	Notes
Memory Optimized instance families: The minimum requirements are listed below: • 8 vCPUs minimum • 64 GB RAM minimum • 10 Gbps min (25 Gbps recommended) • Enhanced Networking • Processors: Intel or AMD (No ARM/ Graviton) • Storage: SSD or NVMe (No HDD) Recommended instance families: • R4 • R5n • R5b • R6in • R6idn	• Larger instance types provide more CPU, which can prevent resource shortfalls under high I/O throughput conditions. • Larger instances also provide more memory, which Continuous Data uses to cache database blocks. More memory will provide better read performance. • Delphix Elastic Data Engines (engines backed by object storage) will need more bandwidth than traditional engines (backed by EBS) because the s3 bucket used to store data is accessed over a network pipe that is shared with other engines activity. Information on AWS instances

Network configuration

It is recommended to enable Enhanced Networking to maximize performance. For more information view Enhanced Networking on Linux. Note that the enhanced networking driver is included with Delphix software, and no steps need to be taken on the Delphix Engine to enable the feature other than rebooting.

Requirements	Notes
Virtual Private Cloud	The Delphix Engine and all of the source and target environments must be deployed in a VPC network to ensure that private IP addresses are static and do not change when restarting instances. When adding environments to the Delphix Engine, use the host's VPC (static private) IP addresses.
Static Public IP	The EC2 Delphix instance must be launched with a static IP address; however, the default behavior for VPC instances is to launch with a dynamic public IP address – which can change whenever the instance restarts. If using a public IP address for the Delphix Engine, static IP addresses can only be achieved by using assigned AWS Elastic IP Addresses
Security Group Configuration	The default security group will only open port 22 for SSH access. The security group must be modified to allow access to all of the networking ports used by the Delphix Engine and the various source and target engines. See Network Performance Configuration Options for information about network performance tuning. See General Network and Connectivity Requirements for information about specific port configurations.

EBS configuration

Deploying Delphix on AWS EC2 requires EBS volumes. Since EBS volumes are connected to EC2 instances via the network, other network activity on the instance can affect throughput to EBS volumes. Delphix recommends EBS General Purpose SSD (GP2 and GP3) or Provisioned IOPS SSD volumes to provide consistent and predictable storage performance. For more information on EBS volumes, see external documentation href="https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-volume-types.html ">EBS Volume Types.

Recommendations	Notes
EBS Provisioned IOPS SSD Volumes (Highly Recommended)	Delphix does not support the use of instance store volumes. Use EBS volumes with provisioned IOPs in order to provide consistent and predictable performance. The number of provisioned IOPs depends on the estimated IO workload on the Delphix Engine. Provisioned IOPs volumes must be configured with a volume size to provisioned IOPs per the EBS Volume Types guidelines.
EBS General Purpose SSD Volumes	Delphix supports GP2 and GP3 General Purpose SSDs. You can now migrate your Amazon EBS volumes from GP2 to GP3. For more details on migration, see this Amazon documentation on EBS Volumes Migration. Delphix Elastic Data (Engines backed by s3 storage) need gp3 or higher.

General storage configuration

Requirements	Notes
General Storage Configuration	For Delphix Continuous Cloud Engines backed by s3 object storage, read the section ***Delphix Elastic Data Engines (Engines backed by object storage)*** in Initial Setup. For Regular engines (backed by EBS): • Allocate initial storage equal to the size of the physical source database storage. For high redo rates and/or high DB change rates, allocate an additional 10-20 %. • All data storage volumes must be EBS volumes. Delphix recommends using a minimum of four equally sized storage disks to run the Delphix Engine. This allows the Delphix File System (DxFS) to assure its file systems are always consistent on disk without additional serialization. This also enables the Delphix Engine to achieve higher I/O rates by queueing more I/O operations to its storage.

Additional AWS configuration notes

- Using storage other than EBS is not supported.
- Limits on the number of volumes are dictated by the EBS instance type, and is generally advised that over 40 can be expected to cause issue on Linux VMs. More information can be found in the AWS Volume Limits and AWS Volume Constraints articles. The maximum device limit imposed by AWS can be handled by the Delphix Engine.
- Cold HDD (sc1) volumes (not supported due to poor performance)
- Using fast storage for EBS volumes is supported and recommended, including (in order of decreasing speed):
    - Provisioned IOPs SSD volumes (recommended)
    - General Purpose SSD volumes (supported)
    - Throughput Optimized HDD (supported, not recommended due to performance)
- Use of EBS volumes encrypted at creation is supported (during initial deployment from AMI, as well as storage devices added post-deployment), but can have negative performance consequences. Conversion of existing EBS volumes is possible in AWS but is not supported in Delphix at this time.

Prerequisites to deploying in AWS

Overview

This article outlines the prerequisites for deployment of the Delphix Engine on AWS. The setup user should have experience launching and configuring instances in the Amazon Web Services environment. Review and complete the tasks in the next section before deployment.

Prerequisites

1. Review Checklist of information required for installation and configuration.
2. Make sure that the Amazon account being used to deploy the Delphix Engine has an appropriate level of enablement to subscribe to the Delphix Engine for AWS subscription.
3. Determine which virtual private cloud (VPC) is being used when launching the virtualization instance. To maximize performance, deploy the Delphix Engine instance in the same VPC/subnet in which the virtual databases (VDBs) will be created.
    a. Provisioning a VDB requires a compute instance running the same database engine as the source. Please note, however, that the target instance only needs storage to accommodate the OS, database platform binaries, etc., because Delphix delivers all of the data files.
4. Make sure that the necessary ports are open.
    a. Using the Delphix Engine for AWS will require connections to source and target database servers. Such connections require various ports to be open, enabling communications. For a detailed list of the network and port requirements, click the link that corresponds with the relevant database platform:
        i. Network and connectivity requirements for oracle environments
        ii. Network access requirements for SQL server environments
        iii. Network and connectivity requirements for SAP ASE base environments
        iv. Network and connectivity requirements for Db2 environments
5. Update Security Group settings to accommodate the necessary connections.
    a. Select the same Security Group that the current (or future) non-production EC2 compute nodes utilize.
    b. Modify the Security Group to allow access to all of the networking ports used by the Delphix Engine and the various source and target platforms. See links above for information about specific port configurations.
6. Allocate storage.
    a. To properly size the initial storage capacity and determine the number and size of EBS Provisioned IOPs Volumes required, download and utilize the Delphix-dynamic-data-platform-storage-calculator. It is helpful to first create a list of the data sources intended for making dSources. A data source is typically a production database linked to the Virtualization Engine, enabling to create virtual, full, read-write copies of the source within minutes. The list should include the database name, platform (for example, Oracle or SQL Server), current size (in GB), the estimated number of virtual copies, and retention period (in days) of snapshots (backup copies).
    b. All data storage volumes must be EBS volumes. Delphix recommends using a minimum total of four disks to run the Delphix Engine. One disk is used for the boot device. The other four equally sized disks will be used for data storage. This also enables the Delphix Engine to achieve higher I/O rates by queueing more I/O operations to its storage.
    c. Provisioned IOPs EBS volumes are highly recommended.
7. During the Manual Deployment option, use the guidelines outlined in Virtual machine requirements for AWS EC2 platform.

Geographic distribution in regions and availability zones

The latency will be directly related to not just the Availability Zone configuration, but more specifically the geographies of those zones. The latencies can vary from tens to hundreds of milliseconds if the zones are geographically diverse (US West to US East would certainly be expected to perform better than US West to Europe). AWS advertises that all AZs in a given region are interconnected with high-bandwidth and low-latency networking per this AWS article.

Applications that are performance-sensitive will benefit from colocating the target servers and Engine in the same region if possible. Another possibility is building a failover strategy for those highly sensitive servers, enabling them to failover to another AZ in the instance where an Engine needs to be failed over. The architecture selected and geographies will also naturally be dependent on your redundancy requirements (your organization may require geographically diverse failover options beyond the ~60 miles advertised within a region).

Procedure for deploying in AWS

Overview

This article outlines the procedure for deploying the Delphix Engine on AWS, using the Delphix Marketplace Image or Amazon Machine Image (AMI).

Procedure to install an AMI

Use the Delphix-supplied AMI file to install the Delphix Engine.

1. On the Delphix download site, select the AMI to be shared and accept the Delphix License agreement. Alternatively, use a link provided by the organization's Delphix solutions architect.
2. On the **Amazon Web Services Account Details** form presented:
    a. Enter the AWS Account Identifier. To use the **GovCloud AWS Region**, be sure to enter the ID for the AWS Account which has GovCloud enabled.
    b. Select which **AWS Region** that the AMI will be shared in. For an AMI shared in a different region, contact the organization's Delphix account representative to make the proper arrangements.
3. Click **Share.** The Delphix Engine will appear in the list of AMIs in AWS momentarily.

Subscribe to the Delphix virtualization engine marketplace image

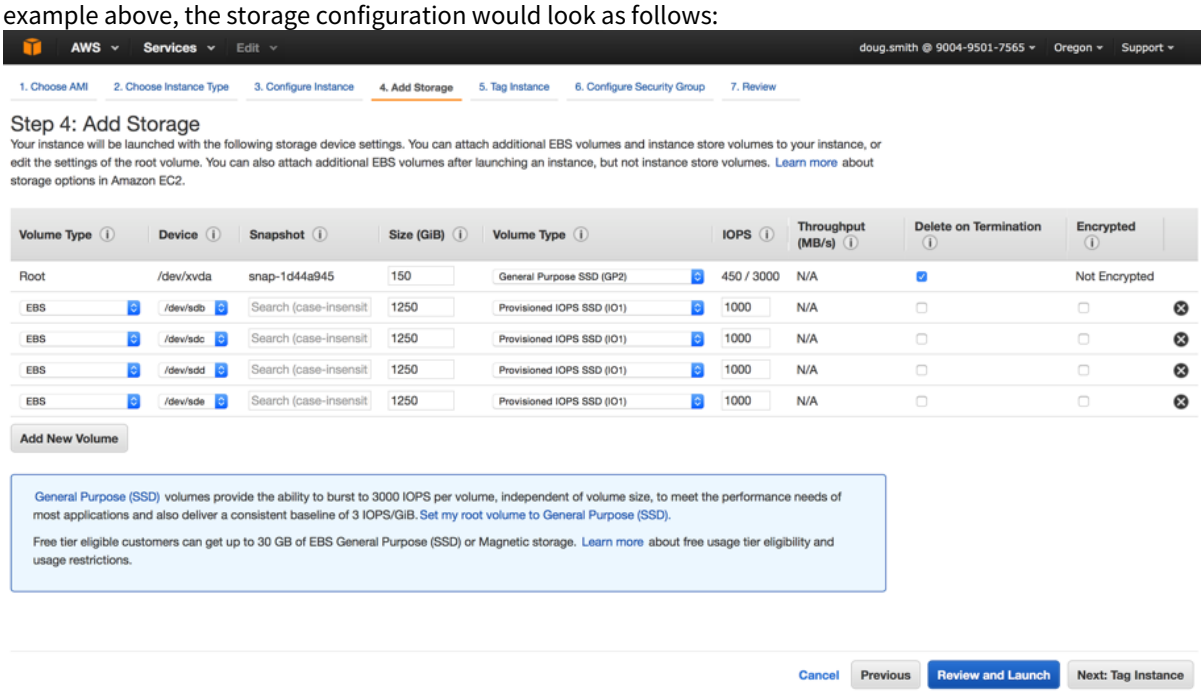
1. Login to the AWS Console at https://aws.amazon.com/console.
2. Navigate to the AWS Marketplace.
3. In the **Search** field, enter **Delphix**.
4. Select the **Delphix DevOps Data Platform for AWS**. Select the appropriate offer based on the data you will ingest into the Delphix platform.
5. Review the information on the initial Marketplace page.
6. Click **Continue to Subscribe**.
7. Review the software subscription information and select the **Manual Launch** tab.
8. Select **Accept Software Terms**. Your subscription will take a few minutes to become enabled.
9. From the **Subscription** page, click **Return to Product Page**. There you will see **Launch with EC2 Console** enabled in all of the available regions.

Launching the Delphix engine

1. In the desired region, select **Launch with EC2 Console**. Launching will redirect to the standard EC2 instance launch process.
2. Select the size of the virtual machine to deploy. For supported instance types and capabilities, see Virtual machine requirements for AWS platform.
3. Click **Next: Configure Instance Details**.
4. Configure instance details as follows:
    a. Set the **number of instances** to **1**.
    b. **Purchasing option** – Leave **Request Spot Instances** unchecked.
    c. **Network** – Select the VPC into which the instance will be deployed.
    d. **Subnet** – Select the Subnet into which the instance will be deployed.
    e. **Auto-assign Public IP** – Select **Disable**.
    f. **Placement group** – Optional. The default is **No placement group**.
    g. **IAM Role** – None
    h. **Shutdown behavior** – Stop
    i. **Enable termination protection** – Yes (select checkbox). This ensures that the Delphix Engine is not accidentally terminated.

   j. **Monitoring** – Optional
   k. **EBS-optimized instance** – Yes (select checkbox)
   l. **Tenancy** – Shared – Run a shared hardware instance
   m. **Network Interfaces**– Eth0
      Note : This option will not appear if a default subnet is chosen. In this case, the first option below will apply.
      i. Auto-assign (default if using DHCP)
      ii. Enter static IP in Primary IP field (recommended for consistency)
5. Click **Next: Add Storage**.
6. By default, the Delphix Virtualization instance includes a 150GB OS volume. Ensure that this volume is set as a **General Purpose SSD (GP2)** type.
7.
   a. For traditional Block storage based engines
      i. Using the Delphix Virtualization Engine Storage Calculator for AWS, enter information from the predetermined inventory into the worksheet. This will automatically calculate the number, size, and IOPS required for the EBS volumes.
      In the above example, there are three (3) source databases listed with their current sizes. Each is estimated to have 5 virtual copies as well as a 14-day retention period for snapshots/ backups. The calculator indicates to add four (4) EBS-provisioned IOPS volumes, each being 1250GB with 1000 IOPS.
      Note
      This calculator is only for estimating the anticipated storage size and IOPS needs. A manual adjustment may be required prior to or after launch depending on storage and performance needs.
   b. For Elastic Data Engines (Refer to Initial setup for additional details) s3 bucket acts as the storage and EBS disks as cache are used to reduce latencies for frequently read data and as temporary storage for synchronous writes before the writes are sent to object storage. Make sure the disks satisfy the I/O needs of the engine. You can always add disks later if the cache is insufficient. You can only reduce the cache by removing disks, so if you think you are over provisioning the cache, increase the number of disks used and reduce the size of each disk so that removal is possible at a later point in time.
      When selecting disks for Elastic Data Engines:
      i. gp3 disks are recommended as they offer good performance at a lower cost. At 500 IOPS per 1GiB ratio, a 32 GiB volume can be configured to have the gp3 volume 16K IOPS limit and 1000 MB/s throughput. For reference, r5n.8xlarge instance has a 30K IOPS and 850 MB/s throughput limits so two gp3 devices would be sufficient for the instance.
      ii. io2 disks have lower latency at a higher cost. However, the lower latency is not beneficial once the instance IOPS or throughput limit is reached.
8. Click **Add New Volume**. The new volume will be created using the specifications in the estimation provided. The below values are sample values set for a Volume Type. See EBS configuration for related information.
   a. **Volume Type** – EBS
   b. **Device** – Use default provided
   c. **Size (GiB)** – 1250
   d. **Volume Type** – Provisioned IOPS SSD (IO1) Delphix deploys as an EC2 instance on AWS using EBS volumes for storage. AWS provides the following options for Elastic Block Storage (EBS) storage volumes, namely
      i. Standard
      ii. General Purpose SSD (GP2)
      iii. General Purpose SSD (GP3)
      iv. Provisioned IOPS SSD (IO1)
   e. **IOPS** – 1000
   f. **Delete on Termination** – Optional. If this option is not set and the Delphix Engine is terminated, the EBS volumes will persist, and they will need to be removed manually.
   g. **Encrypted** – Optional

9. Follow the above procedure for adding as many EBS volumes as specified in the calculator. Based on the example above, the storage configuration would look as follows:



10. Verify the storage configuration matches the output from the calculator.
11. Click **Next: Tag Instance**.
12. To provide a name for the Delphix Virtualization Instance, enter it in the **Value** field corresponding to the **Key Name**.
13. Click **Next: Configure Security Group**.
14. Assign a security group. Do one of the following:
    a. Create a new security group.
    b. If an existing security group meets Delphix requirements, it can be selected. For more details on required and recommended ports for general Delphix usage, see General network and connectivity requirements.
15. Click **Review and Launch**.
16. Verify the Delphix Virtualization Instance details and click **Launch**.
17. Proceed without a key pair.
18. Select the **I acknowledge** checkbox.
19. Click **Launch Instances**.
20. Click **View Instances**.
21. To complete the initial Delphix Virtualization Engine configuration, wait for the instance to be up and running. The status is available on the **Instances** page. Note that it can take up to 15 minutes for the first launch of Delphix, and even if the status of the server is “running” it may not yet be ready.

Configuring the Delphix engine

1. Connect to the running Delphix instance with a web browser. Use the IP address or DNS name noted in the Instance Description.
2. Upon successful connection, the browser will automatically redirect to the **Delphix Setup Page**.
3. Refer to the standard product deployment instructions to complete the Delphix deployment.

Logging in for the first time

On the first time login to the Delphix Engine, follow these steps:

1. Enter the default Administration User: **sysadmin**.
2. Enter **sysadmin** for the password (when installing a new engine via AWS AMI, the initial sysadmin password is the AWS Instance ID).
   Note
   Find the <*Instance ID*> in the **Instances** section of the AWS EC2 Management Console.
3. Once redirected, enter the following as prompted:
   a. **Email address** for the admin account.
   b. A new **password**. This password will be used for the Engine Administrator with the username "admin".
   c. Confirm the new password.
      Note:
      From now on, this is the password used to log in to the Delphix Engine Administrator interface to manage Datasets.
4. Click **Continue**.

Next steps

Congratulations! The Delphix Virtualization Engine should be successfully deployed in AWS.

Use Delphix documentation to learn how to:

- configure a database source
- configure target environments
- create virtual databases (VDBs)

AWS RDS custom for Oracle and SQL server

Overview

The Amazon Relational Database Service (Amazon RDS) makes it easy to set up, operate, and scale a relational database in the cloud. It allows organizations to automate time-consuming administration tasks such as hardware provisioning, database setup, patching, and backups. It frees focus that can be utilized on applications, eliminating extra overhead. Amazon RDS is available for Oracle and SQL Server, amongst other database services.

Delphix AWS RDS custom support for Oracle and SQL server

AWS has announced a new version of their RDS databases, thus, Delphix is adding AWS RDS Custom support for Oracle. AWS RDS Custom is an RDS service that allows Delphix to access the operating system and database level functions required to provision vDBs into the AWS RDS PaaS (Product as a Service) database service.

With AWS RDS Custom, organizations using Delphix can obtain the full benefits of PaaS and Delphix automation for their target environments.

- Users ingest data from production in RDS Oracle/SQL Server, mask the data, and provision secure copies back to RDS. All Delphix automation tools work and also manual operations via the UI.
- Delphix data compression and block sharing capabilities will allow organizations to potentially reduce their AWS RDS storage consumption.

For additional questions or use case evaluations, please contact Delphix Support.

Deployment for Microsoft Azure

Overview

This article outlines the virtual machine requirements, including memory and data storage, for deploying the Delphix Engine on the Azure Public Cloud and Government Cloud. Once the requirements listed on this page are reviewed, refer to the next articles on Azure deployment:

- Prerequisites to deploying in Microsoft Azure
- Procedure for deploying in Microsoft Azure

Instance types

Requirements	Notes
Instance families: The minimum requirements are listed below: • 8vCPUs minimum • 64GB RAM minimum • Network Bandwidth: 10Gbps min (25Gbps recommended) • Azure Accelerated Networking • Processors: Intel or AMD (No ARM) • Storage • SSD or NVMe (No HDD) Recommended instance families: • Dsv2-series • Dsv3-series • Esv3-series • Esv4-series • Esv5-series	• Larger instance types provide more CPU, which can prevent resource shortfalls under high I/O throughput conditions. • Larger instances also provide more memory, which the Delphix Engine uses to cache database blocks. More memory will provide better read performance. • Delphix Elastic Data Engines (engines backed by blob storage) will need more bandwidth than traditional engines (backed by block storage) because the blob bucket used to store data is accessed over a network pipe that is shared with other engines activity. • Information on Azure instances

Network configuration

The use of Azure Accelerated Networking is supported for Delphix Engines and is recommended to maximize network performance. Accelerated Networking provides more bandwidth with more consistent and lower network latencies.

Requirements	Notes
Azure virtual network (VNet)	The Delphix Engine must have <= 1 ms of latency to the Target and Staging environments. Longer latencies are permitted between the Delphix Engine and Source environments, but may impact Snapsync timings. VNET peering may be used, as long as latency requirements are met. For very low latency requirements, consider Proximity Placement Groups for Delphix Engine, Target, and Staging.
Network security group (NSG)	The security group must be modified that allows access to all networking ports used by the Delphix Engine and the various source/target platforms. See General network and connectivity requirements for information about specific port configurations. See Network performance configuration options for information about network performance tuning.

Storage configuration

Requirements	Notes
General Storage Configuration	• Allocate initial storage equal to the size of the physical source databases. For high redo rates and/or high DB change rates, allocate an additional 10-20% storage. • Add storage when storage capacity approaches 30% free. • Delphix recommends using a minimum of four equally sized storage disks to run the Delphix Engine. This allows the Delphix File System (DxFS) to assure its file systems are always consistent on disk without additional serialization. This also enables the Delphix Engine to achieve higher I/O rates by queueing more I/O operations to its storage. • Maximize Delphix Engine RAM for a larger system cache to service reads. • See, Delphix storage migration.
Azure Premium Storage	• Premium storage utilizes solid-state drives (SSDs). Ultra disk is also supported. • Devices up to 32TB are supported. • A maximum of 256TB is supported. • Cache setting of NONE. • I/O requests of up to 256 kilobytes (KB) are counted as a single I/O operation (IOP) for provisioned IOPS volumes. • IOPS vary based on storage size with a maximum of 20,000 IOPS. • For Delphix Continuous Cloud Engines backed by object storage, read the section ***Delphix Elastic Data Engines (Engines backed by object storage)*** in Initial Setup.

Extensions

Azure VM Extensions are not currently supported.

Prerequisites to deploying in Azure

Overview

This article outlines the prerequisites for deployment of the Delphix Engine on Azure. The setup user should have experience launching and configuring instances in the Microsoft Azure environment. Review and complete the tasks in the next section before deployment.

Prerequisites

1. Review Checklist of information required for installation and configuration.
2. Determine which virtual private cloud (VPC) is being used when launching the Virtualization instance. To maximize performance, deploy the Delphix Engine instance in the same VPC/subnet where the virtual databases (VDBs) will be created.

> Provisioning a VDB requires a compute instance running the same database engine as the source. Please note, however, that the target instance only needs storage to accommodate the OS, database platform binaries, etc., because Delphix delivers all of the data files.

3. Make sure that the necessary ports are open.

> Using the Delphix Engine for Azure will require connections to source and target database servers. Such connections require various ports to be open, enabling communications. For a detailed list of the network and port requirements, click the link that corresponds with the relevant database platform:
>
> 1. Network and Connectivity Requirements for Oracle Environments
> 2. Network Access Requirements for SQL Server

4. Update Security Group settings to accommodate the necessary connections.

   a. Select the same **Network Security Group** that the current (or future) non-production Azure virtual machines utilize.

   b. Modify the Network Security Group to allow access to all of the networking ports used by the Delphix Engine and the various source and target engines. See links above for information about specific port configurations.

5. Allocate storage.

> It is helpful to first create a list of the data sources intended for making dSources. A data source is typically a production database linked to the Virtualization Engine, enabling to create virtual, full, read-write copies of the source within minutes. The list should include the database name, platform (for example, Oracle or SQL Server), current size (in GB), the estimated number of virtual copies, and retention period (in days) of snapshots (backup copies).

   a. All data storage disks must be comprised of Azure Premium Storage devices. Delphix recommends using a minimum total of five disks to run the Delphix Engine. One disk is used for the boot device. The other four equally sized disks will be used for data storage. This also enables the Delphix Engine to achieve higher I/O rates by queueing more I/O operations to its storage.

   b. Azure premium storage devices have different levels of guaranteed IOPs that vary from 120 IOPs to 20,000 IOPs. Please refer to Microsoft's documentation to ensure that the selected devices will meet the requirements for your deployment. UltraHD disk storage is supported for performance-sensitive customers.

c. It is highly recommended to use Azure-managed disks.

d. Use the guidelines outlined in Deployment for Microsoft Azure

Procedure for deploying in Azure

Overview

This article outlines the procedure for deploying the Delphix Engine in Microsoft Azure.

Deploying the Delphix engine

If you are deploying a cloud engine on Azure Blob please read the Delphix Elastic Data Engines Initial setup section before deploying the cloud engine.

1. Navigate to the Azure Marketplace at https://azuremarketplace.microsoft.com
2. In the **Search** field, enter **Delphix**.
3. Select the **Delphix Virtualization for Azure (3TB)** for an hourly subscription or **Delphix DataOps Platform for Azure** for customers with existing licenses.
4. Review the information on the initial Marketplace page.
5. Click **GET IT NOW**.
6. Click **Create**.
7. Follow the creation wizard to deploy the Delphix Engine.
8. Provide the basic information for the Delphix Engine.
    a. Select a name for the virtual machine.
    b. For the OS disk type, select **Premium SSD**.
    c. Enter a **username** and **password**.
    Info : This information is never used but is part of the built-in wizard used by the marketplace. Any information provided in these fields is discarded.
    d. Select your **Subscription**, **Resource group**, and **Location**.
    e. Click **OK**.
9. Select the **Size** of the virtual machine you want to deploy. For supported instance types and capabilities, See Virtual machine requirements for Azure platform
    Each instance type will determine the limits for:
    - Network bandwidth
    - Maximum data disk
    - Total IOPS
10. Click **Select**.
11. Configure the **Settings** for this virtual machine:
    a. Use managed disks.
    b. Select or create a **Virtual Network** and **Subnet**.
    c. Optional: select a **Public IP address** (this will make an engine accessible over the internet)
    d. Select or create a **Network Security Group**.
    e. **Extensions** and **High Availability sets** are not currently supported. Please ensure that no extensions or availability sets have been configured.
    f. Enable **Boot diagnostics**.
12. Click **OK**.
13. Review the configuration in the **Summary** screen and click **OK**.
14. Review the offer details in the **Buy** screen and click **Purchase** to deploy the Delphix Engine.

Configuring the Delphix engine virtual machine

1. From the **Azure Portal** sidebar, select **Virtual Machines**. The running virtual machines will be displayed.
2. Click the virtual machine's **name** to open its detailed information.
3. Click **Disks** to open the detailed panel.
4. Click **Add data disk**.

5. Create a new device:
    a. In the **Create disk** drop-down menu, select **Create managed disk**. This will open the managed disk wizard.
    b. In the wizard,
        i. For block engine
            1. ensure that the **Account type** indicates **Premium (SSD)** and that the **Source Type** shows **None (empty disk)**.
        ii. For Elastic Data engine
            1. Ultra disks are recommended as they can be configured to have high IOPS and throughput at relatively small sizes. At 300 IOPS per GiB ratio, a 256 GB volume can have 76,800 IOPS and 4000 MB/s throughput.
            2. Premium SSD disks can have high performance though they need to be much larger, e.g. P80 32TiB volume has 20K IOPS and 900 MB/s. The new Premium SSD v2 disks have better IOPS per GiB ratio though they're not available in all regions
            3. You can always add disks later if the cache is insufficient. You can only reduce the cache by removing disks, so if you think you are over provisioning the cache, increase the number of disks used and reduce the size of each disk so that removal is possible at a later point in time.
    c. Select the size of the device to create. Note that the device size will determine the IOPS limit for that device.
    d. Click **Create**.
6. Once the device is created, change the **Host Caching** parameter for that device to **None**.
7. Repeat the above steps to create additional disks.
8. Once adding all the disks is complete, click **Save**.
9. Optional: modify the virtual machine's **Network Security Group**.
    a. From the side panel of the virtual machine's detail screen, click **Network interfaces**.
    b. Click the name of the network interfaces to modify.
    c. From the **Network interface** details window, click the **Network security group** specified in the **Overview** pane. In the example below, the user would click **delphix-engine-nsg**:
    d. From the **Network security group** details pane, modify the **Inbound security rules** and/or the **Outbound security rules** by selecting from the side panel and clicking **Add**.
    e. Add the appropriate rule and click **OK**. The screenshot below shows adding the inbound rule for the Delphix Session Protocol.
    f. Repeat the above steps for all the rules that should be added.

Configuring the Delphix engine

1. Connect to the running Delphix instance with a web browser. Use the Public IP address or DNS name noted in the **Network Interface** Panel. Upon successful connection, the browser will automatically redirect to the **Delphix Setup Page**.
2. Refer to the standard product deployment instructions to complete the Delphix deployment.

Next steps

Congratulations! The Delphix Virtualization Engine should be successfully deployed in Azure.

Use Delphix documentation to learn how to:

- configure a database source
- configure the target environments
- create virtual databases (VDBs)

Enabling Azure accelerated networking

Overview

This article describes how to enable Azure Accelerated Networking. If a VM was created without Accelerated Networking, enabling this feature on an existing VM is possible. The VM must meet the following prerequisites:

- VM must be a supported size for Accelerated Networking
- VM must be a supported Azure Gallery image (and kernel version for Linux)
- All individual VMs or VMs in an availability set must be stopped/deallocated before enabling Accelerated Networking on any NIC

For more information, please see the Azure documentation

Individual VMs and VMs in an availability set

Stop/deallocate the VM or, if an Availability Set, all the VMs in the Set:

- If the VM was created individually (without an availability set), only the individual VM needs to stop/deallocate to enable Accelerated Networking.
- If the VM was created with an availability set, all VMs contained in the availability set will need to stop/deallocate before enabling Accelerated Networking on any Network Interface Card (NIC).

Azure CLI

```
az vm deallocate \
    --resource-group myResourceGroup \
    --name myVM
```

Once stopped, enable Accelerated Networking on the NIC of the VM:

Azure CLI

```
az network nic update \
    --name myNic \
    --resource-group myResourceGroup \
    --accelerated-networking true
```

Restart the VM (or all VMs if using an availability set) and confirm that Accelerated Networking is enabled:

Azure CLI

```
az vm start --resource-group myResourceGroup \
    --name myVM
```

After the restart, the Mellanox VF (Virtual Function) device will be exposed to the VM.

Deployment for Google cloud platform

Overview

This article covers the virtual machine requirements, including memory and data storage, for the deployment of the Delphix Engine on Google Cloud Platform (GCP). Once the requirements listed on this page are reviewed, refer to the next articles on GCP deployment:

- Prerequisites to deploying in GCP
- Procedure for deploying in GCP

Machine types

The following is a list of instance types that are supported to deploy Delphix on GCP. Delphix periodically certifies new instance types, which will be added to the list here.

Recommendations	Notes
General-purpose machine families. The minimum requirements are listed below: • 8vCPUs minimum • 64GB RAM minimum • 10Gbps min (25Gbps recommended) • Processors: Intel or AMD (No ARM) • Storage • SSD or NVMe (No HDD) Recommended instance families: • N2 standard • N2 high-mem	• The Delphix Engine most closely resembles a storage appliance and performs best when provisioned using a storage-optimized instance type. • Larger instance types provide more CPU, which can prevent resource shortfalls under high I/O throughput conditions. • Larger instances also provide more memory, which the Delphix Engine uses to cache database blocks. More memory will provide better read performance. Information on GCP general-purpose machines

Network configuration

Requirements	Notes
Virtual private cloud	The Delphix Engine and all of the source and target environments must be deployed in a VPC network to ensure that private IP addresses are static and do not change when restarting instances. When adding environments to the Delphix Engine, use the host's VPC (static private) IP addresses.

Requirements	Notes
Static public IP	The GCP Delphix instance must be launched with a static IP address; however, the default behavior for VPC instances is to launch with a dynamic public IP address – which can change whenever the instance restarts. If using a public IP address for the Delphix Engine, static IP addresses can be assigned using the Google cloud documentation
Security group configuration	The default security group will only open port 22 for SSH access. The security group must be modified to allow access to all of the networking ports used by the Delphix Engine and the various source and target engines. See Network performance configuration options for information about network performance tuning. See General network and connectivity requirements for information about specific port configurations.

Storage configuration

Requirements	Notes
General storage configuration	Allocate initial storage equal to the size of the physical source database storage. For high redo rates and/or high DB change rates, allocate an additional 10-20 %. Add storage when storage capacity approaches 30% free. Delphix recommends using a minimum of four equally sized storage disks to run the Delphix Engine. This allows the Delphix File System (DxFS) to assure its file systems are always consistent on disk without additional serialization. This also enables the Delphix Engine to achieve higher I/O rates by queueing more I/O operations to its storage.

Prerequisites to deploying in GCP

Overview

This article outlines the prerequisites for deployment of the Delphix Engine on GCP. The setup user should have experience launching and configuring instances in the Google Cloud Platform environment. Review and complete the tasks in the next section before deployment.

Prerequisites

1. Review the Checklist of information required for installation and configuration
2. Make sure that the Google account being used to deploy the Delphix Engine has an appropriate level of enablement for the **Delphix virtualization for GCP** subscription.
3. Determine which virtual private cloud (VPC) is being used when launching the virtualization instance. To maximize performance, deploy the Delphix Engine instance in the same VPC/subnet in which the virtual databases (VDBs) will be created.
   Note:
   Provisioning a VDB requires a compute instance running the same database engine as the source. Please note, however, that the target instance only needs storage to accommodate the OS, database platform binaries, etc., because Delphix delivers all of the data files.
4. Make sure that the necessary ports are open.
   Note:
   Using the Delphix Engine for GCP will require connections to source and target database servers. Such connections require various ports to be open, enabling communications. For a detailed list of the network and port requirements, click the link that corresponds with the relevant database platform.
   a. Network and Connectivity Requirements for Oracle Environments
   b. Network and Connectivity Requirements for SQL Server Environments
   c. Network and Connectivity Requirements for SAP ASE Base Environments
   d. Network and Connectivity Requirements for Db2 Environments
5. Update Security Group settings to accommodate the necessary connections.
   a. Select the same Security Group that the current (or future) non-production EC2 compute nodes utilize.
   b. Modify the Security Group to allow access to all of the networking ports used by the Delphix Engine and the various source and target platforms. See links above for information about specific port configurations.
6. Allocate storage.
   a. To properly size the initial storage capacity and determine the number and size of Provisioned IOPs Volumes required, download and utilize the Delphix-Dynamic-Data-Platform-Storage-Calculator
      Note:
      It is helpful to first create a list of the data sources intended for making dSources. A data source is typically a production database linked to the Virtualization Engine, enabling to create virtual, full, read-write copies of the source within minutes. The list should include the database name, platform (for example, Oracle or SQL Server), current size (in GB), the estimated number of virtual copies, and retention period (in days) of snapshots (backup copies).
   b. Delphix recommends using a minimum total of four disks to run the Delphix Engine. One disk is used for the boot device. The other four equally sized disks will be used for data storage. This also enables the Delphix Engine to achieve higher I/O rates by queueing more I/O operations to its storage.

Additional GCP configuration notes

- Delphix supports both Zonal and Regional SSD persistent disks.

Procedure for deploying in GCP

Overview

This article outlines the procedure for deploying the Delphix Engine using a tar.gz file in Google Cloud Console or in Google Cloud Marketplace.

Deployment in Google cloud console

1. Download the latest **Delphix Platform for GCP** tar.gz file from https://download.delphix.com/.
2. Create a **GCP storage bucket** and upload the tar.gz file from the first step.
3. Once the upload is complete, navigate to **Compute Engine > Images** and select **Create Image.**
4. In the **Create Image** dialog, provide the following information.
    a. Image Name
    b. Source: Cloud Storage File
    c. Select the bucket and newly uploaded Delphix File.
    d. Location: Multi-Regional or Regional. (To deploy Delphix in multiple regions - select **Multi-Region**)
    e. Family: Optional
    f. Description: Optional
    g. Labels: Optional
    h. Encryption: Select based on the organization's policy
    i. Click **Create**. (Creating the image could take some time)
5. Once the image creation is complete, select the newly created image and create an instance from it.
6. Configure the instance creation screen with the following items.
    a. Identify and select the region and zone to run Virtual Databases.
    b. Choose a supported machine type, commensurate with the expected workload.
    c. Change/modify the **Boot Disk** to **SSD Persistent Disk**. (Defaults to **Standard Persistent Disk**)
    d. Add **Data Disks** (minimum of four recommended) with a total size at least equal to the total source DB sizes. Be sure to utilize SSD Persistent Disks.
    e. Update Networking, commensurate with the Network and Subnet(s) on which the target non-production instances reside. Note, a “target” is an instance running the DB platform identical to the source DBs.
    f. Once all of the necessary instance information has been configured, select **Create**.
    g. Wait for the instance to be created/available and connect to the newly deployed Engine using the assigned IP/hostname via the support web browser.

Deployment in Google cloud marketplace

1. Login to the **Google Cloud Marketplace**.
2. In the **Search** field, enter **Delphix**.
3. Select the **Delphix Data Virtualization for GCP (3TB)**.
4. Review the information on the initial Marketplace page.
5. Click **Launch**.
6. Review the deployment configuration and software subscription information.
7. Accept Google Cloud’s and Delphix Corp’s Terms of Service.
8. Click **Deploy**, which will start the deployment of the instance.
9. After the deployment, add equally-sized data disks to the instance.

Configuring the Delphix engine

1. Connect to the running Delphix instance with a web browser. Use the IP address or DNS name noted in the Instance Description. Upon successful connection, the browser will automatically redirect to the **Delphix Setup Page**.
2. Refer to the standard product deployment instructions to complete the Delphix deployment.

Next Steps

Congratulations! The Delphix Virtualization Engine should be successfully deployed in Google Cloud Platform.

Use Delphix documentation to learn how to:

- configure a database source
- configure the target environments
- configure virtual databases (VDBs)

Deployment for OCI

Overview

This article outlines the virtual machine requirements, including memory and data storage, for deploying the Delphix Engine on Oracle Cloud Infrastructure (OCI). Once the requirements listed on this page are reviewed, refer to the next articles on OCI deployment:

- Prerequisites to deploying in OCI
- Procedure for deploying in OCI

Compute image types

Delphix distributes product images for OCI using the QCOW2 image type. Compute Images must be imported into OCI using the Paravirtualized launch mode; currently, images using the Emulated launch mode are not supported.

Supported shapes

The following is a list of shapes that are supported to deploy Delphix on OCI.

Requirements	Notes
Compute shapes The minimum requirements are listed below: • 8vCPUs minimum • 64GB RAM minimum • Network Bandwidth: 10 Gbps min (25 Gbps recommended) • Processors: Intel or AMD (No ARM) • Storage • SSD or NVMe (No HDD) Recommended compute shapes: • VM.Standard2 • VM.Standard3.Flex • VM.Standard.E[3,4,5].Flex	• The Delphix Engine most closely resembles a storage appliance and performs best when provisioned using a storage-optimized shape. • Larger shapes provide more CPU, which can prevent resource shortfalls under high I/O throughput conditions. • Larger shapes also provide more memory, which the Delphix Engine uses to cache database blocks. More memory will provide better read performance. • For more information on OCI shapes, refer to OCI shapes.

Network configuration

Requirements	Notes
Virtual cloud network (VCN)	• The Delphix Engine and all of the source and target environments must be deployed in a VCN to ensure that private IP addresses are static and do not change when restarting instances. • By default, OCI subnets are considered public. When defining a subnet, it is recommended to set it as private. Unless required by the environment, the VCN should not include a Public Subnet. • When adding environments to the Delphix Engine, use the host's VCN (static private) IP addresses.
Static private IP	• The Delphix instance should be launched with a static private IP address. For security reasons, it is encouraged to avoid configuring the engine with a Public IP address; however, it could be passable to use a dynamic Public IP address in addition to a static Private IP address if the environment requires such access.
Security rules configuration	• OCI allows two firewall features: **Network Security Groups** (NSGs) and **Security Lists**. Oracle recommends the use of NSGs over Security Lists because NSGs let the VCN subnet architecture be separate from the application security requirements. More information can be found in this Oracle documentation • A VCN will use a Security List to define default rules. By default, the security list will only open port 22 for SSH access. The Security List must be modified, or new NSGs created, to allow access to all of the networking ports used by the Delphix Engine and the various source and target engines. • This dual implementation of firewall or security rules may be different from other clouds, please see OCI documentation for best practices. • See Network performance configuration options for information about network performance tuning. • See General network and connectivity requirements for information about specific port configurations.

General storage configuration

Requirements	Notes
General storage configuration	• Allocate initial storage equal to the size of the physical source database storage. For high redo rates and/or high DB change rates, allocate an additional 10-20 %. • Add storage when storage capacity approaches 30% free. • Delphix recommends using a minimum of four equally sized storage disks to run the Delphix Engine. This allows the Delphix File System (DxFS) to assure its file systems are always consistent on disk without additional serialization. This also enables the Delphix Engine to achieve higher I/O rates by queueing more I/O operations to its storage. • Must use Block Volume for data storage. • Block Volumes must be attached using Paravirtualized mode.
OCI storage configuration	• Currently supported Shapes only support Block Volumes; File Storage is not supported. • Paravirtualized block devices are required; currently, iSCSI devices are not supported. • Elastic Performance Configuration Options (aka Volume Performance Policy): use Higher Performance. • For Delphix Continuous Cloud Engines backed by object storage, read the section ***Delphix Elastic Data Engines (Engines backed by object storage)*** in Initial Setup. • Block storage for caching requires a minimum of 200G Higher Performance volumes for caching.

Additional OCI configuration notes

- When running low on storage space, Delphix recommends adding additional equivalently sized block storage volumes, or devices, instead of resizing existing volumes.
- If existing storage volumes must be expanded, this must be done using the “online” resizing strategy specified in OCI documentation; “offline” storage resizing is not supported and may lead to unexpected downtime. If an existing storage volume is expanded, use the Setup or sysadmin interface to expand each storage “device” or volume. The additional storage, as a result of a resize, will not be available for use until the storage devices are explicitly instructed to make use of the additional space.
- If expanding storage volumes, it is recommended that all volumes are expanded to the same size. When storage volumes or devices are the same size, the Delphix product is able to balance I/O distribution among the disks for optimal performance.
- Hot removal of storage volumes is not supported.

Prerequisites to deploying OCI

Overview

This article outlines the prerequisites for deployment of the Delphix Engine on OCI. The setup user should have experience launching and configuring instances in the Oracle Cloud Infrastructure environment. Review and complete the tasks in the next section before deployment.

Prerequisites

1. Contact a Delphix representative to request this capability. Delphix will assist in assuring that all Oracle Cloud requirements are met to successfully run a Delphix Engine with the most appropriate configuration for the use case.
2. Review the Checklist of information required for installation and configuration
3. Review Oracle's Creating an instance documentation
4. Make sure that the Oracle account being used to deploy the Delphix Engine has an appropriate level of permissions to upload files to a Bucket in Object Storage, and use that object to produce a Custom Image.
5. Determine which virtual cloud network (VCN) is being used when launching the virtualization instance. To maximize performance, deploy the Delphix Engine instance in the same VCN/subnet in which the virtual databases (VDBs) will be created.
   Note:
   Provisioning a VDB requires a compute instance running the same database engine as the source. Please note, however, that the target instance only needs storage to accommodate the OS, database platform binaries, etc., because Delphix delivers all of the data files.
6. Make sure that the necessary ports are open.
   Using the Delphix Engine for OCI will require connections to source and target database servers. Such connections require various ports to be open, enabling communications. For a detailed list of the network and port requirements, click the link that corresponds with the relevant database platform.
   - Network and Connectivity Requirements for Oracle Environments
   - Network and Connectivity Requirements for SQL Server Environments
   - Network and Connectivity Requirements for SAP ASE Environments
   - Network and Connectivity Requirements for Db2 Environments
7. Update Security Group (or Security List) settings to accommodate the necessary connections.
   a. Select the same Security Group that your current (or future) non-production compute nodes utilize.
   b. Modify the Security Group to allow access to all of the networking ports used by the Delphix Engine and the various source and target platforms. See Oracle Network Security Groups
8. Allocate storage.
   a. To properly size the initial storage capacity and determine the number and size of Provisioned IOPs Volumes required, download and utilize the Delphix-dynamic-data-platform-storage-calculator
      Note:
      It is helpful to first create a list of the data sources intended for making dSources. A data source is typically a production database linked to the Virtualization Engine, enabling to create virtual, full, read-write copies of the source within minutes. The list should include the database name, platform (for example, Oracle or SQL Server), current size (in GB), the estimated number of virtual copies, and retention period (in days) of snapshots (backup copies).
   b. Delphix only supports using Block Volume devices for data storage. These block devices must be attached to the Delphix Engine using the Paravirtualized attach mode (iSCSI is not currently supported). Block Volumes must be provisioned in the same Availability Domain as the Delphix Engine, otherwise, they will not be discoverable. Block Volumes should be configured to use the "Higher performance" performance option. All Block Volumes attached to a single Delphix Engine

should be of the same size. A minimum of four (4) Block Volumes should be attached to a Delphix Engine.

9. During the Manual Deployment option, use the guidelines outlined in Virtual machine requirements for OCI

Procedure for deploying in OCI

Overview

This article outlines the procedure for deploying the Delphix Engine in Oracle Cloud Infrastructure.

Download and verify the Delphix engine image

1. Contact an account manager to request access to the OCI variant of the Delphix product.
2. Follow the link given by the Delphix solutions architect. Download the **Delphix_x.x.x.x_....Standard_OCI.qcow**2 file and the **SHA256SUMS** file.
3. Once both files have finished downloading and are in the same directory, run the following command to verify the download: `$ grep -i OCI.qcow2 ./SHA256SUMS | sed -E 's,Appliance_Images/(Controlled_Availability/)?,,g' | sha256sum --check`

Upload the Delphix engine image as an object

1. Authenticate with OCI and navigate to the Infrastructure console
2. Use the navigation menu to reach the Object storage buckets, core infrastructure, page (Menu > Object Storage > Object Storage).
3. Set the **List Scope Compartment**. This will depend on the organization's strategy for managing OCI resources.
4. Create a storage bucket or select an existing bucket.
5. Click the blue **Upload** button.
6. In the **Upload Objects** modal window, specify an optional prefix and choose the OCI specific QCOW2 file that was previously downloaded.
7. Click the blue **Upload** button.

Creating a custom compute image from an object

1. Authenticate with OCI and navigate to the Infrastructure console
2. Use the navigation menu to reach the Compute custom images, core infrastructure, page (Menu > Compute > Custom Images).
3. Set the **List Scope Compartment**.
4. Click the blue **Image Import** button.
5. In the **Import Image** modal window, select a suitable compartment in the **Create In Compartment** field that conforms to the organization's strategy on managing OCI resources.
6. In the **Name** field enter a unique name to identify the Custom Compute Image (using the same name as the image object from the previous step is recommended). Upload the Delphix Engine Image as an Object.
7. For **Operating System** select **Linux**.
8. Next, identify an object by specifying its Compartment, Bucket, and Object Name. Or, specify an Object Storage URL.
   Note:
   The Object Details will identify this value as **URL Path (URI)**.
9. For **Image Type** select **QCOW2**.
10. For **Launch Mode** select **Paravirtualized Mode**.
11. For organizations that have a tagging policy for cloud-based resources, expand the **Tagging Options** section, and define tags.
12. Click the blue **Import Image** button.

Launching the Delphix engine

1. Authenticate with OCI and navigate to the Infrastructure Console
2. Use the navigation menu to reach the Compute instances, core infrastructure, page (Menu > Compute > Instances).
3. Set the **List Scope Compartment**.
4. Click the blue **Create Instance** button.
5. In the **Create Compute Instance** window pane, specify a unique name for the VM.
6. For the **Create In Compartment** field, select a suitable compartment that conforms to the organization's strategy on managing OCI resources.
7. In the **Image or operating system** section, click the **Change Image** button. Switch to the **Custom Images** tab. Find the Delphix image that corresponds to the instance being deployed. Click the blue **Select Image**button.
   Note:
   If the Delphix Custom Image is not visible, look for the **Change Compartment** option near the top of the current window pane.
8. Each Availability Domain has its own quota, it is ok to use AD-1, AD-2, or AD-3 - but, be sure to make note of which Availability Domain is being used.
   Note:
   Compute Instances and attached Storage will need to be in the same Availability Domain.
9. In the **Shape** section, click the **Change Shape** button. For **Instance type**, specify **Virtual Machine**. For **Shape series**, use **Intel Skylake**. Select an OCI Shape that is supported by Delphix.
10. Continue on to the **Configure networking** section.
    a. If the network is not specified correctly, it is likely that firewall issues will occur. In this case, please consult the organization's IT or DevOps team. If the organization is using Network Security Groups (NSGs), mark the **Use Network Security Groups to Control Traffic** checkbox after consulting the appropriate teams. Last, select the **Do Not Assign a Public IP Address** radio button.
11. The **Boot Volume** section can be skipped.
12. In the **Add SSH Keys**, select the **No SSH Keys** radio option. The Delphix product is a closed appliance and manages users independently.
13. In general, all of the Advanced Options can be skipped. For organizations that have a tagging policy for cloud-based resources, expand into the Advanced Management section, and look for the **Tagging** sub-section to define tags.
14. Click the blue **Create** button - wait about 2-5 minutes for the Delphix Engine instance to boot.

Create block storage volumes

1. Authenticate with OCI and navigate to the Infrastructure console
2. Use the navigation menu to reach the Block volumes, core infrastructure, page (Menu > Block Storage > Block Volumes).
3. Set the **List Scope Compartment**.
4. Click the blue **Create Block Volume** button.
5. In the **Create Block Volume** modal window, specify a unique name for this Block Volume. It can be helpful if this name is descriptive or identifies the VM it is intended to be attached to and ends in a sequence number.
6. For the **Availability Domain,** this value MUST be the same Availability Domain used for the Delphix Engine instance, otherwise, this volume will not be available for use.
7. In the **Volume Size and Performance** section, select the **Custom** option. Set the size of the volume to be sufficiently large (with room for growth) to support the databases that will be virtualized, or masked, by this Delphix Engine. Set the **Default Volume Performance** to the **Higher Performance** setting.
8. A **Backup Policy** is not required and can be left blank or **No Backup Policy Selected**. However, depending on the organization's needs, consider selecting a Backup Policy.
9. For **Encryption,** the default option of **Encrypt Using Oracle-Managed Keys** is permissible. Optionally, for independent encryption keys, use the **Encrypt Using Customer-Managed Keys** option.

10. For organizations that have a tagging policy for cloud-based resources, expand the **Tagging Options** section and define tags.
11. Uncheck the checkbox that says **View Detail Page After This Block Volume is Created**. This will prevent navigating away from the Block Volumes page because multiple Block Volumes will need to be created at the same time.
12. Click the blue **Create Block Volume** button.
13. A Delphix Engine requires a minimum of four equally sized Block Volumes for data storage. Repeat Steps 4-12 as many times as necessary.

Attach block storage volumes

(For an Elastic Data engine, the block storage will be used for caching. See additional details in the Delphix Elastic Data Engines section in Initial setup)

1. Authenticate with OCI and navigate to the Infrastructure console
2. Use the navigation menu to reach the Block volumes, core infrastructure, page (Menu > Block Storage > Block Volumes).
3. Set the **List Scope Compartment**.
4. From the list of pre-existing Block Volumes, identify the resources being attached to the Delphix Engine and wait until the volume state becomes Available.
5. Select one of the **Block Volumes** to enter the **Block Volume Details** page.
6. On the left-hand side, locate the **Resources** menu and select **Attached Instances**.
7. If the Block Volume has not been previously attached to another VM, selecting the blue **Attach to Instance** button will be available.
8. In the **Attach to Instance** modal window, specify the **Attachment Type** as **Paravirtualized**. Currently, iSCSI is not supported.
9. For **Access Type** use the **READ/WRITE** option.
10. Next, identify a Delphix Engine by selecting an instance or by specifying an instance OCID. If the Delphix Engine instance is not shown in the **Select an Instance** drop-down menu, the **Change Compartment** option could be needed. Block Volumes can only be attached to VM instances that were created in the same Availability Domain - if these values do not match, either re-provision Block Volumes or the Delphix Engine in the correct Availability Domain.
11. Click the blue **Attach** button.
12. Repeat Steps 4-11 until all associated Block Volume resources have been attached to the Delphix Engine instance.

Configuring the Delphix engine

1. Connect to the running Delphix Engine instance with a web browser. Use the IP address or FQDN noted in the Instance Description. Upon successful connection, the browser will display a login prompt to enter the Delphix Setup Page.
2. Refer to the standard product deployment instructions to complete the Delphix deployment.

Next steps

Congratulations! The Delphix Virtualization Engine should be successfully deployed in OCI.

Use Delphix documentation to learn how to:

- configure a database source
- configure target environments
- create virtual databases (VDBs)

Deployment for IBM cloud

Overview

This article covers the virtual machine requirements, including memory and data storage, for the deployment of the Delphix Engine on IBM Cloud. Once the requirements listed on this page are reviewed, refer to the next articles on GCP deployment:

- Prerequisites for deploying in IBM Cloud
- Procedure for deploying in IBM Cloud

Supported profiles

The following is a list of profiles that are supported to deploy Delphix on IBM Cloud.

Requirements	Notes
Memory instance families. The minimum requirements are listed below: • 8vCPUs minimum • 64GB RAM minimum • 10Gbps min (25Gbps recommended) • Processors: Intel or AMD (No ARM) • Storage • SSD or NVMe (No HDD) Recommended instance families: • mx2	• The Delphix Engine most closely resembles a storage appliance and performs best when provisioned using a storage-optimized profile. • Larger profiles provide more CPU, which can prevent resource shortfalls under high I/O throughput conditions. • Larger profiles also provide more memory, which the Delphix Engine uses to cache database blocks. More memory will provide better read performance. Information on IBM Cloud instances

Network configuration

Requirements	Notes
Virtual server instances	• Deploy the Delphix Engine and all of the source and target environments in the same VPC network. • When adding environments to the Delphix Engine, use the host's VPC IP addresses.

Requirements	Notes
Security configuration	• The default security group will only open port 22 for SSH access. The security group must be modified to allow access to all of the networking ports used by the Delphix Engine and the various source and target engines. • See Network performance configuration options for information about network performance tuning. • See General network and connectivity requirements for information about specific port configurations. • Reference: IBM cloud security and compliance documentation

Port Requirements

Using the Delphix Engine for OCI will require connections to source and target database servers. Such connections require various ports to be open, enabling communications. For a detailed list of the network and port requirements, click the link that corresponds with the relevant database platform.

- Network and connectivity requirements for Oracle environments
- Network and connectivity requirements for SQL Server environments
- Network and connectivity requirements for SAP ASE environments
- Network and connectivity requirements for Db2 environments

General storage configuration

Requirements	Notes
General storage configuration	• Allocate initial storage equal to the size of the physical source database storage. For high redo rates and/or high DB change rates, allocate an additional 10-20 %. • Add storage when storage capacity approaches 30% free. • Delphix recommends using a minimum of four equally sized storage disks to run the Delphix Engine. This allows the Delphix File System (DxFS) to assure its file systems are always consistent on disk without additional serialization. This also enables the Delphix Engine to achieve higher I/O rates by queueing more I/O operations to its storage. • Reference: IBM block storage documentation

Additional IBM configuration notes

- Expandable volume is a beta feature that is available for evaluation and testing purposes. This feature is available in the US South, US East, London, and France regions. Contact an IBM Sales representative if interested in getting access at https://cloud.ibm.com/docs/vpc?topic=vpc-expanding-block-storage-volumes .
- After performing an “online” resize/expansion of a storage volume using IBM Cloud tools, then use the Delphix sysadmin interface to “Expand” the storage device; otherwise, the newly allocated storage space, from the resize/expansion, will not be used.
- Resize/expansion of a storage volume using IBM Cloud is not supported while the Delphix engine is in a stopped state.

- Removing a storage volume should be done while the machine is running. First use the Delphix sysadmin CLI interface to “Unconfigure” the storage device, then remove it from IBM Cloud.

Prerequisites for deploying in IBM Cloud

Overview

This article outlines the prerequisites for deployment of the Delphix Engine on IBM Cloud. The setup user should have experience launching and configuring instances in the IBM Cloud environment. Review and complete the tasks in the next section before deployment.

Prerequisites

1. A Delphix software license is required. New users should contact Delphix to get started.
2. Review the Checklist of information required for installation and configuration
3. Review IBM's cloud documentation for IBM Cloud-specific information.
4. Determine which virtual private cloud (VPC) is being used when launching the virtualization instance. To maximize performance, deploy the Delphix Engine instance in the same VCN/subnet in which the virtual databases (VDBs) will be created. **Note:** Provisioning a VDB requires a compute instance running the same database engine as the source. Please note, however, that the target instance only needs storage to accommodate the OS, database platform binaries, etc., because Delphix delivers all of the data files.
5. Make sure that the necessary ports are open. **Note:** Using the Delphix Engine for GCP will require connections to source and target database servers. Such connections require various ports to be open, enabling communications. For a detailed list of the network and port requirements, click the link that corresponds with the relevant database platform.
    - Network and connectivity requirements for Oracle environments
    - Network access requirements for SQL server
    - Network and connectivity requirements for SAP ASE environments
    - Network and connectivity requirements for Db2 environments
6. Update Security Group settings to accommodate the necessary connections.
7. Select the same Network Security Group that your current (or future) non-production Azure virtual machines utilize.
8. Modify the Security Group to allow access to all of the networking ports used by the Delphix Engine and the various source and target platforms. See the links above for information about specific port configurations.
9. Allocate storage.

> It is helpful to first create a list of the data sources from which you intend to make dSources. A data source is typically a production database that you link to the Delphix Engine, enabling you to create virtual, full, read-write copies of the source within minutes. The list should include the database's name, platform (for example, Oracle or SQL Server), current size (in GB), the estimated number of virtual copies, and retention period (in days) of snapshots (backup copies).

a. All data storage disks must be comprised of Premium Storage devices.

b. Delphix recommends using a minimum total of four disks to run the Delphix Engine. One disk is used for the boot device. The other four equally sized disks will be used for data storage. This also enables the Delphix Engine to achieve higher I/O rates by queueing more I/O operations to its storage.

c. Premium storage devices have different levels of guaranteed IOPs that vary from 120 IOPs to 7500 IOPs. Please refer to IBM Cloud documentation to ensure that the selected devices will meet the requirements for your deployment.

d. Use the guidelines outlined in Procedure for Deploying in IBM Cloud.

Procedure for deploying in the IBM Cloud

Overview

This article outlines the procedure for deploying the Delphix Engine in IBM Cloud. There are two methods for deploying a Delphix Engine in the IBM Cloud using the Software Catalog or Manually Uploading the Delphix Image

Deploying from the IBM software catalog

1. Navigate to the IBM Software Catalog and search for Delphix.
2. Select the Delphix Data Virtualization Tile or Masking Tile for the Masking product.
3. Scroll down to the Deployment Values section and input specifics for the environment.

Required parameters	**Description**
hostname	The name of the VSI used to deploy Delphix.
profile	Compute profile to be used for deploying Delphix (see recommended profiles).
ssh_key	Public SSH key is to be used when provisioning the VSI.
subnet_id	The id of the subnet where the VSI will be provisioned.
volumecount	Number of block storage volumes
volumeprofile	Block storage profile to use (recommended is >= 10 IOPS/GB
volumesize	Block storage volume size.
vpcname	Name of the VPC where the VSI is provisioned.
zone	VPC zone to provision the environment.

Downloading the Delphix image

Contact an account manager to request access to the IBM variant of the Delphix product.

1. Follow the link given by the Delphix solutions architect. Download the **Delphix_Verson...._Standard_IBM.qcow2** file and the **SHA256SUMS** file.
2. Once both files have finished downloading and are in the same directory, run the following command to verify the download:

```
$ grep -i IBM.qcow2 ./SHA256SUMS | sed -E 's,Appliance_Images/
(Controlled_Availability/)?,,g' | sha256sum --check
```

Uploading the Delphix engine image as an object

1. Authenticate with the IBM Cloud and navigate to the Dashboard
2. Use the navigation menu to reach the Resource List page. The Resource List page can be navigated from the Dashboard by clicking on **Storage** within the Resource Summary pane.
3. Expand Storage from the menu and select the appropriate resource group. Depending on the organization's strategy for managing IBM resources, creating a resource group may be required.
4. Create a storage bucket or select an existing bucket.
5. Click the blue **Upload** button and select **Files**.
6. When the pop-up menu appears, select the **Transfer Type**. Aspera High-Speed Transfer is required for large files, the plugin will need to be installed. It will automatically navigate through the steps to install the plugin.
7. In the **Upload Files** (objects) window, click on the **Select Files** (objects) button and choose the **IBM-specific QCOW2** file that was previously downloaded.
8. Click the **Upload** button.

Creating a custom image

1. Authenticate with IBM Cloud and navigate to the Dashboard
2. Use the navigation menu to reach the Custom images page for VPC within the VPC infrastructure (IBM Cloud pull-down menu > VPC Infrastructure > Custom images).
3. Click the blue **Create** button.
4. In the **Import Custom Image** page, specify a unique name for the image.
5. From the **Resource Group** drop-down, select the organization's resource group.
6. Optional: In the **Tags** section, provide appropriate tags to organize resources.
7. Select the appropriate **Region**.
8. Select the **Cloud Object Storage** bucket containing the uploaded image by navigating to **Instances** > **Location** > **Bucket** from the drop-down menus. The downloaded **QCOW2** image should appear in the pane below the three drop-down menus.
9. Within the **Operating System** section, click on the **Ubuntu Linux** tile and select **ubuntu-18-04-amd64** from the drop-down menu.
10. Once all the parameters are entered, in the right pane click on the blue button to import custom image.

Launching the Delphix engine

1. Authenticate with IBM Cloud and navigate to the Dashboard
2. Use the navigation menu to reach the Virtual server Instances page within the VPC Infrastructure (IBM Cloud pull-down menu > VPC Infrastructure > Virtual Server Instances). Note: To maximize performance, deploy the Delphix Engine instance in the same VPC/subnet in which you will create your virtual databases (VDBs).
3. Click the blue **Create** button.
4. In the **New Virtual Server** for VPC page, specify a unique name for the VM.
5. From the **Virtual Private Cloud** drop-down, select the organization's VPC.
6. From the **Resource Group** drop-down, select the organization's resource group.
7. Optional: In the **Tags** section, provide appropriate tags to organize resources.
8. Select the **Location** of the IBM Cloud resources.
9. In the **Operating System** section, click on the **Select Custom Image** link within the Custom Image block.
10. In the pop menu, select the IBM-specific image previously uploaded.
11. Within the **Profile** section, click on **View** all profiles. Select one of the supported instance types and click **Save**.
12. The **User Data** section can be skipped.

13. The Boot Volume section can be skipped since it would already have the default values.
14. Creating block storage volumes can be done at a later time and will be discussed in the next section.
15. Continue on to the **Network Interfaces** section. If a subnet is already configured in the zone and VPC, then this section will already have a default network interface. Otherwise, create a subnet with the appropriate security groups. If the network is not specified correctly, it is likely that firewall issues will occur. In this case, please consult the organization's IT or DevOps team. If the organization is using Network Security Groups (NSGs), mark the **Use Network Security Groups to Control Traffic** checkbox after consulting the appropriate teams. Last, select the **Do Not Assign a Public IP Address** radio button.
16. Click the **Create virtual server instance** button on the right panel. This will take a couple of minutes.

Creating block storage volumes

1. Authenticate with IBM Cloud and navigate to the Dashboard
2. Use the navigation menu to reach the Block Storage Volumes within VPC Infrastructure (IBM Cloud pull-down menu > VPC Infrastructure > Block Storage Volumes).
3. Click the blue **Create** button.
4. In the **Block Storage Volume for VPC** modal window, specify a unique name for this Block Volume. It can be helpful if this name is descriptive or identifies the VM it is intended to be attached to and ends in a sequence number.
5. From the **Resource Group** drop-down, select the organization's resource group.
6. Optional: In the **Tags** section, provide appropriate tags to organize resources.
7. Select the **Location** of the IBM Cloud resources.
8. Enter the required **IOPS**. The recommended supported IOPS is 10/GB.
9. Enter the storage size in GB. Set the size of the volume to be sufficiently large, with room for growth, to support the databases that will be virtualized, or masked, by this Delphix Engine.
10. Keep the **Encryption** settings at default, e.g. Provider Managed.
11. Click the blue **Create Volume** button.
12. Delphix recommends using a minimum total of four disks to run the Delphix Engine. One disk is used for the boot device. The other four equally sized disks will be used for data storage. This also enables the Delphix Engine to achieve higher I/O rates by queueing more I/O operations to its storage. Repeat Steps 3-11 as many times as necessary.

Attaching block storage volumes

1. Authenticate with IBM Cloud and navigate to the Dashboard
2. Use the navigation menu to reach the Block storage volumes within VPC Infrastructure (IBM Cloud pull-down menu > VPC Infrastructure > Block Storage Volumes).
3. From the list of pre-existing **Block Volumes**, identify the volumes attaching to a Delphix Engine and wait until the volume state becomes Available.
4. Note that the volumes being attached have **Attachment Type** set as a hyphen.
5. Select the right side of the volume row menu, then select **Attach to Instance**.
6. In the **Attach Virtual Server Instance** modal window, select the virtual server instance (Delphix Engine) from the drop-down menu.
7. Click on the blue **Attach Volume** button.
8. Repeat Steps 3-7 until all associated **Block Volume** resources have been attached to the Delphix Engine instance.

Configuring the Delphix engine

1. Connect to the running Delphix Engine instance with a web browser. Use the IP address or DNS name noted in the Instance Description. Upon successful connection, the browser will display a login prompt to enter the **Delphix Setup Page**.
2. Refer to the standard product deployment instructions to complete the Delphix deployment.

Next steps

Congratulations! The Delphix Virtualization Engine should be successfully deployed in IBM Cloud.

Use Delphix documentation to learn how to:

- configure the database source
- configure target environments
- create virtual databases (VDBs)

Hotfix information

Overview

If there are any hotfixes installed for the Delphix Engine, the list can be accessed via the system administrator CLI with the following commands.

```
engine> cd system
engine system> ls
....
hotfixes: [HF-111]
....
```

Configuration

Configuration

Once you have deployed Delphix in the infrastructure of your choice, you will need to manage the settings of each Delphix Engine. The configuration section covers everything you need to do and know about Delphix settings and system administration. Each Delphix engine has its own settings, user profiles, policies, and many other configuration parameters. The majority of these are managed per engine, but increasingly will be managed via the Central Managed service. There are also many administrative functions you may need to manage, such as storage and capacity utilization, system monitoring integrations, and Support bundle upload. You may also want to change your engine's configuration, such as adding new storage disks, authentication mechanisms, or network properties.

Registration management

Each Delphix Engine can be registered via our internal system, typically for association with the Delphix Support team. Learn how to ensure up-to-date registration of your Delphix Engines.

User and authentication management

There are various user types and configurations to consider when using Delphix. Learn how to set up users, and manage authentication mechanisms such as Single Sign-on and Kerberos.

Network and DNS management

You may want to manage and configure certain network services, such as DNS, for Delphix. Here we specify general network and connectivity requirements and detail how to test network performance.

Capacity and resource management

Delphix will be responsible for managing many different data sources and data types. Learn storage and quota best practices, as well as the different options to optimally manage capacity.

Monitoring and log management

Using both Delphix and its associated datasets will generate many types of logs. Here, we explain the types of logs that Delphix creates as well as the monitoring tool integrations, such as SNMP and Splunk.

Performance analytics management

Delphix offers various performance analytics tools to help users monitor throughput, latency, and other key metrics. Learn how to leverage these tools and architect your Delphix deployment for optimal performance.

Starting, stopping, and restarting your engine

Occasionally, you may need to reboot or stop your Delphix engine, here, you will find steps to securely and safely restart your engines.

Usage data management

The Delphix User-click Analytics feature is a lightweight method to capture how users interact with Delphix product user interfaces. User-click Analytics may also be disabled via the UI.

Registration management

Registering your Delphix Engines is a prerequisite for serviceability from the Delphix Support team. Registration allows Delphix Support to access the engine and properly diagnose and identify any issues during support cases. It is important to manage your registration regularly in order to ensure the security of Support access, as described in the section below.

Registering a Delphix Engine with support is different from registering it with Data Control Tower, *formerly* Central Management. For more information refer to Data Control Tower

> The registration code contains an encrypted key that only Delphix can decrypt, which is unique for each engine. Delphix uses this key to generate one-time authentication codes that authorized Support personnel can use to log into the engine during support sessions.

Retrieving the Delphix engine registration code

To enable Delphix Support, you may perform registration either during the initial setup of the Delphix Engine. If you want to register later, you may retrieve the registration code from the system setup portal.

1. You can retrieve the Delphix Engine Registration Code during the Initial Setup or later through the Delphix Setup application after logging in as a system administrator.
2. In the Registration panel, click View.
3. The Registration Code is displayed in the bottom half of the Registration window.
4. If your local workstation is connected to the external Internet, you can auto-register the Delphix Engine:
    a. Enter your Delphix Support Username and Support Password.
    b. Click Register.
5. If external connectivity is not immediately available, you must register manually.
    a. Copy the Delphix Engine registration code
    b. Log in with your support credentials at https://register.delphix.com
    c. Paste the Registration Code and click Register.

Following registration, you will receive an email confirming the registration of your Delphix Engine

Regenerating the Delphix engine registration code

Delphix recommends that you regenerate the registration code every six months to rotate the secret key in order to maximize the Support Security of the Delphix Engine.

Procedure

1. Log into the CLI (command-line interface) of the Delphix Engine with the sysadmin credentials.
2. Type /registration/regenerate and hit enter.
3. Type commit and hit enter. After a few seconds, the new code will be displayed.
4. Re-register the engine with this new code.

Failing to re-register the Delphix Engine after regenerating the registration code may prevent Support personnel from accessing the engine. In such a case, a support session cannot begin until the engine has been re-registered with the new registration code.

User and authentication management

There are two main topics in this section:

- Managing users and groups who will access and manage Delphix.
- Configuring various authentication mechanisms to access both Delphix and connected environments and datasets.

This section covers the following topics:

Users and groups

User types and user management

There are three user types in the Delphix user model: the system administrator, the Delphix user, and the Self-service user.

> Usernames must start with a letter and contain only alphanumeric characters, hyphens, underscores, and/or periods.

System administrators

System administrator users are responsible for managing the Delphix Engine itself, but not the objects (Environments, dSources, VDBs) within the server. For example, a system administrator is responsible for setting the time on the Delphix Engine and its network address, restarting it, creating new system administrator users (but not Delphix users), and other similar tasks.

A user called sysadmin is the default system, administrator user. While this user can be suspended, it may not be deleted. When the Delphix Management application first launches, this user can log in using the username sysadmin and password sysadmin.

To create or modify system administrators, first, log in to Delphix Setup and navigate to the Users section of the homepage. Here, you can:

- **Add new** system administrators with the plus sign
- **Change** system administrator passwords with the pencil icon
- **Delete** system administrators with the trashcan icon
- **Suspend** system administrators with the pause button
- **Reinstate** system administrators with the play button

Delphix users

Delphix users are responsible for managing the environments and datasets within Delphix, such as dSources, virtual databases (VDBs), users, groups, and related policies and resources.

A Delphix user can be marked as a Engine Administrators. Engine Administratorss have three special privileges:

- They can manage other Delphix users
- They implicitly have Owner privileges for all Delphix objects
- They can create new groups and new environments

The default Delphix user provided with a Delphix Engine is a Engine Administrators and is called admin. Like the sysadmin user, the admin cannot be deleted. When the Delphix Management application launches, the admin user can log in using the password specified during the initial setup when Delphix was first launched.

Only these two users require password-based authentication. Also, other users may use other mechanisms such as LDAP or Kerberos, as described in Configuring and managing kerberos and Configuring and using LDAP with the Delphix Engine.

Self-service users

Delphix Self-Service has two types of users: the admin user and the data user.

Admin users have full access to all report data and can configure Delphix Self-Service, additionally, they can:

- Use the Delphix Engine to add/delete users

- Change tunable settings
- Add/delete tags
- Create and assign data templates and containers

Data users have access to production data provided in a data container. The data container provides these users with a playground in which to work with data using the Self-Service Toolbar.

For more information on Self-service users, visit our Self-service documentation.

User privileges for Delphix objects

The user roles on Delphix objects consist of four types, which the Engine Admin user assigns: Provisioner, Owner, Data Operator, and Reader. These privileges apply both to objects, such as dSources and Virtual Databases (VDBs), and to groups, which are containers that hold those objects.

The Engine Administrators user can assign privileges to groups, dSources, and VDBs. Privileges are inherited, meaning that privileges assigned to a group are effective for the dSources and VDBs contained in that group.

If a user does not have a privilege in relation to an object or group, then he or she has no visibility into that object or group.

Roles and Privileges for Delphix Objects

Role	Object privileges	Group privileges
Owner	• Can provision VDBs from owned dSources and VDBs • Can perform Virtual to Physical (V2P) from owned dSources and VDBs • Can access statistics on the dSource, VDB, or snapshot such as usage, history, and space consumption • Can refresh or rollback VDBs • Can snapshot dSources and VDBs • Can start, stop, or re-start VDBs	• Can provision VDBs from all dSources and VDBs in the group • Can refresh or rollback all VDBs in the group • Can snapshot all dSources and VDBs in the group • Can perform Virtual to Physical (V2P) from owned dSources and VDBs • Can view Templates for policies. • Can not create, edit, or delete a policy template from the policy page. • Can assign Owner privileges for dSources and VDBs • Can access the same statistics as a Provisioner, Data Operator, or Reader • Can start, stop, or re-start VDBs
Provisioner	• Can access statistics on the dSource, VDB, or snapshot such as usage, history, and space consumption • Can provision VDBs from dSources and VDBs	• Can access statistics on all dSources, VDBs, or snapshots in the group such as usage, history, and space consumption • Can provision VDBs from all dSources and VDBs in the group

Role	Object privileges	Group privileges
Data Operator	• Can access statistics on the dSource, VDB, or snapshot such as usage, history, and space consumption • Can refresh or rollback VDBs • Can snapshot dSources and VDBs	• Can access statistics on all dSources, VDBs, or snapshots in the group such as usage, history, and space consumption • Can refresh or rollback all VDBs in the group • Can snapshot all dSources and VDBs in the group
Reader	• Can access statistics on the dSource, VDB, or snapshot such as usage, history, and space consumption	• Can access statistics on all dSources, VDBs, or snapshots in the group such as usage, history, and space consumption

Self-Service Only	• In the Delphix Self Service UI this user can:
	  • Refresh
	  • Restore
	  • Bookmark
	  • Reset
	  • Branch
	  • Stop/Activate
	  • Share	

Managing groups

Creating groups helps you manage policies and privileges over objects within that group. When privileges are created for users at the group level, those privileges apply to all objects of that type within the group. When new objects are created or added to the group, the policies and privileges you have created at the group level will be applied to them.

- Assigning group and object ownership
- Adding and deleting groups

Authentication mechanisms

Delphix supports a variety of authentication mechanisms to connect to several different interfaces and systems. For example, you can connect via the UI using the default users described above, or you can connect to the CLI using an API token.

There are three categories of authentication related to Delphix: the Delphix UI, the Delphix CLI/API, and external systems such as Kerberos access to connected source and target hosts. Below are detailed pages related to each of these three sections:

- UI authentication:
  - Data Control Tower, formerly Central Management
  - Username and password
  - LDAP: Directory-based authentication to Delphix engines rather than the default local access
  - Single Sign-on: Integration and support for identity providers to authenticate users on a per engine basis using SAML2-SSO.
- CLI authentication:

    - Username and password
    - Auto-authentication via SSH keys: to automatically sign in to the Delphix CLI without requiring user-input credentials
- API authentication
    - Username and password
    - API Tokens (for Delphix Engines registered with Data control tower)
    - OAuth2 JSON Web Tokens
- External systems:
    - Username and password
    - SSH keys
    - Kerberos: Authentication for environments and data sources using Kerberos

Keberos support

Kerberos support is for access to connected environments, rather than the Delphix engine itself. This is an advanced topic and will require a solid understanding of Delphix concepts and architecture.

Assigning group and object ownership

This topic describes how to assign group and object ownership to users in the Delphix Domain.

1. Log into the **Delphix Management** application.
2. Select **Manage > Users**.
3. For an existing user, select a user then select the edit icon.
4. Click **Next**.
5. In the **Privileges** tab assign **Owner** or **Provisioner** rights for groups or objects within groups. You do not have to assign a specific owner or auditor right for each object.
6. Click Next when finished.
7. For new users, refer to Users and Groups. When you click **Submit**, the User Profile manager will reload, and then you can follow steps 4 - 6 to assign privileges.

Adding and deleting groups

This topic describes how to add and delete groups within the Delphix Domain.

> **User terminology for Delphix admin has been changed to engine admin.**
> The default domain user created on Delphix Engines is now **admin** instead of delphix_admin. When engines created before 5.3.1 are upgraded to 5.3.1 or later they will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then Disabling delphix_admin.

Adding a group

1. Log into the **Delphix Management** application.
2. From the **Manage** menu, select **Datasets**.
3. Select the **plus** icon and then select **Add Dataset Group**.
4. Enter a **Group Name** and an optional description.
5. Click **Add**.

> **At least one group must exist**
> At least one group must always exist on the Delphix Engine in order to link a dSource. If you delete the last group, you will need to create a new group in order to create a dSource.

Deleting a group

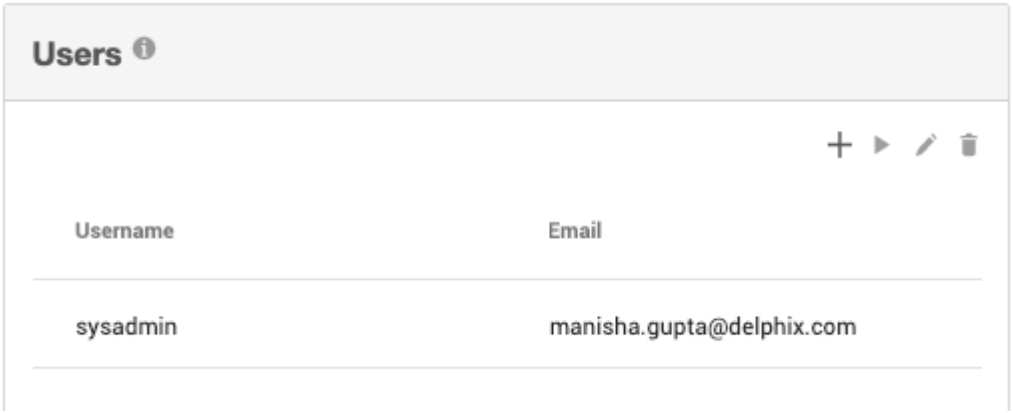
1. Log into the **Delphix Management** application as a user with **Engine Admin** privileges or group **OWNER** privileges for the target group.
2. From the **Datasets** panel, select the target group.
3. Click the **Trash Can** icon.
4. Click **Delete**.

> **Deleting groups containing objects**
>
> A group cannot be deleted if it contains VDBs or dSources. All databases within a group must be deleted prior to deleting the group.

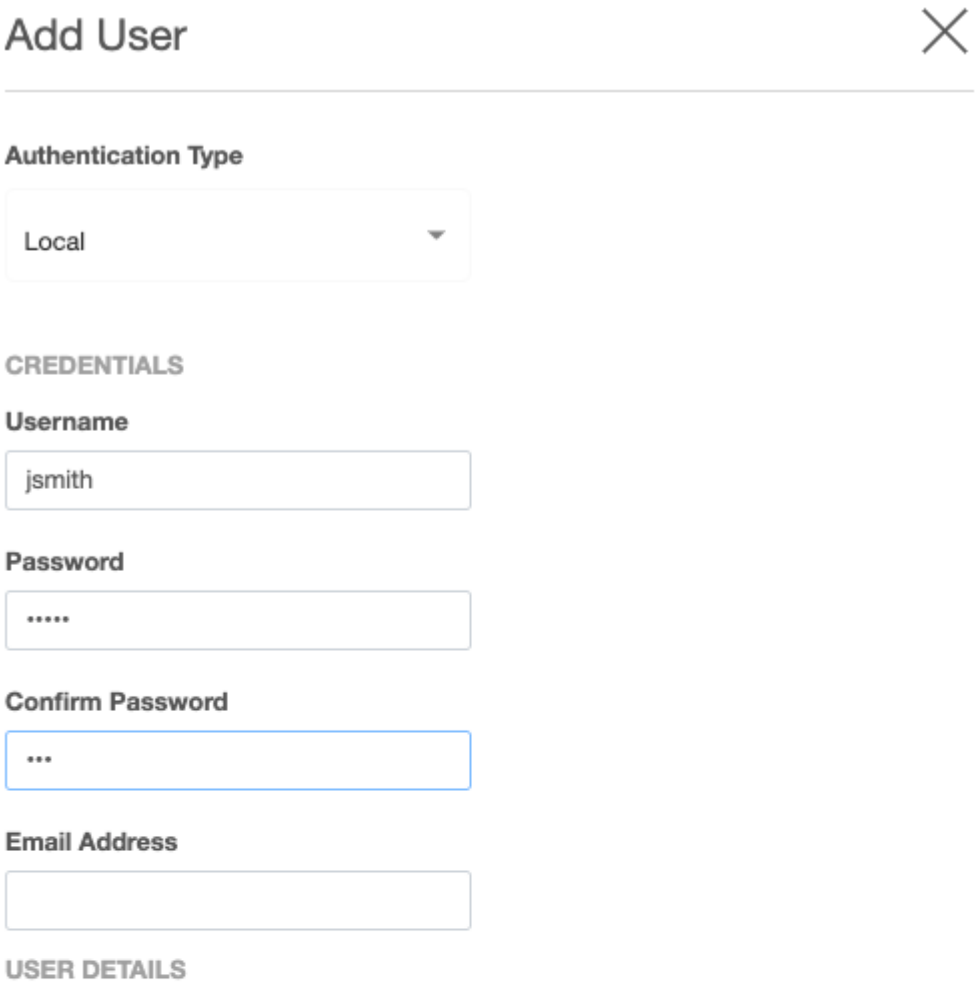
Managing system administrators

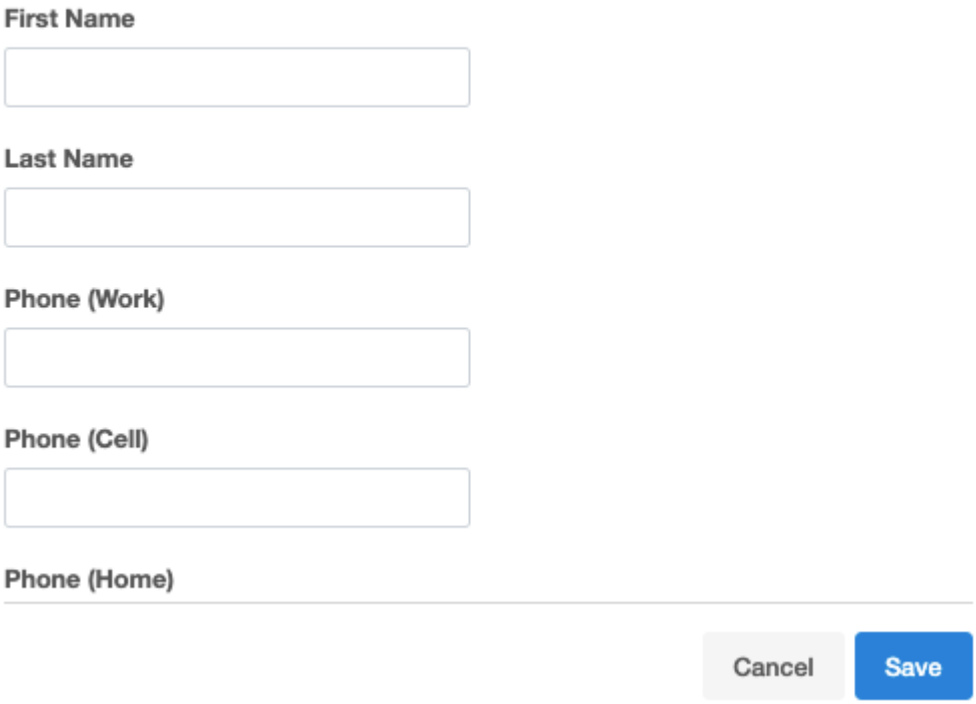
Adding a new system administrator

1. Launch the **Setup** application.
2. From the **Dashboard** select the **+** icon located in the **Users** card next to the filter field.



3. Enter the required information.

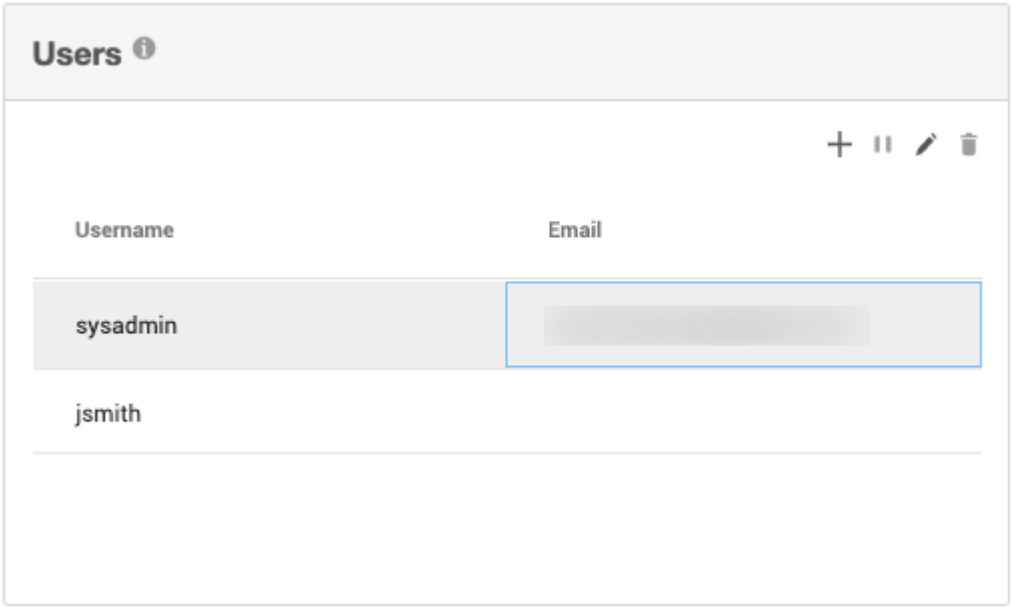




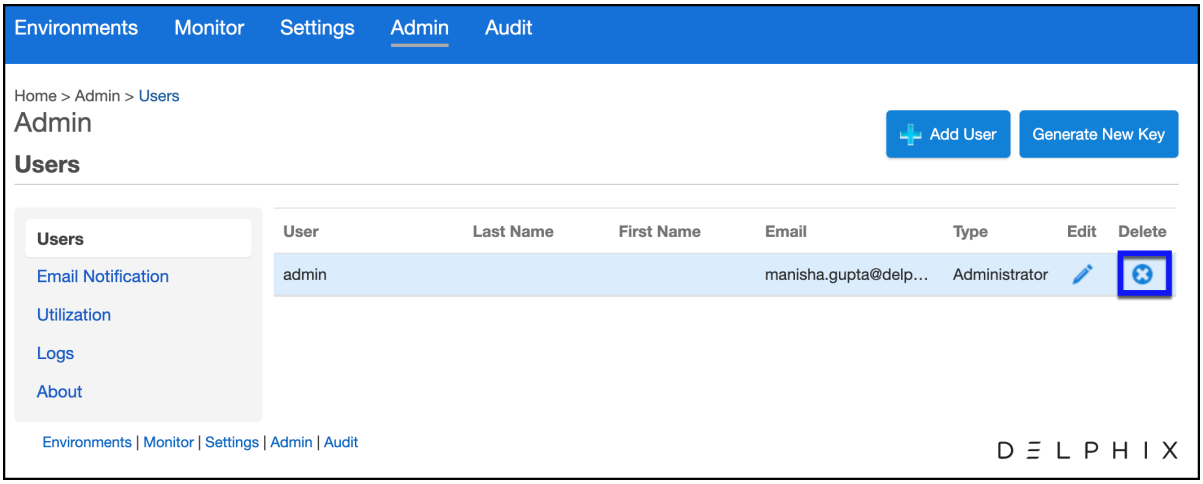
4. Click **Save**.

Deleting and suspending system administrators

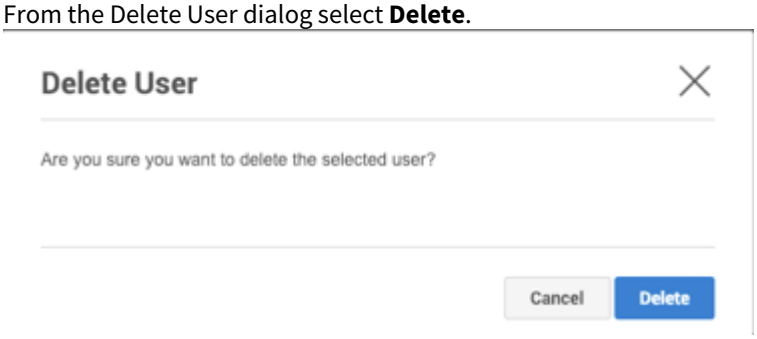
1. Launch the **Delphix Setup** application and log in using the **sysadmin** (or another system administrator) credentials.
2. In the **User** panel, click the user you want to suspend or delete.
3. Suspend the user by clicking the **pause** icon ( ).



4. Once the user has been paused you can delete the user by clicking the **trash can**.



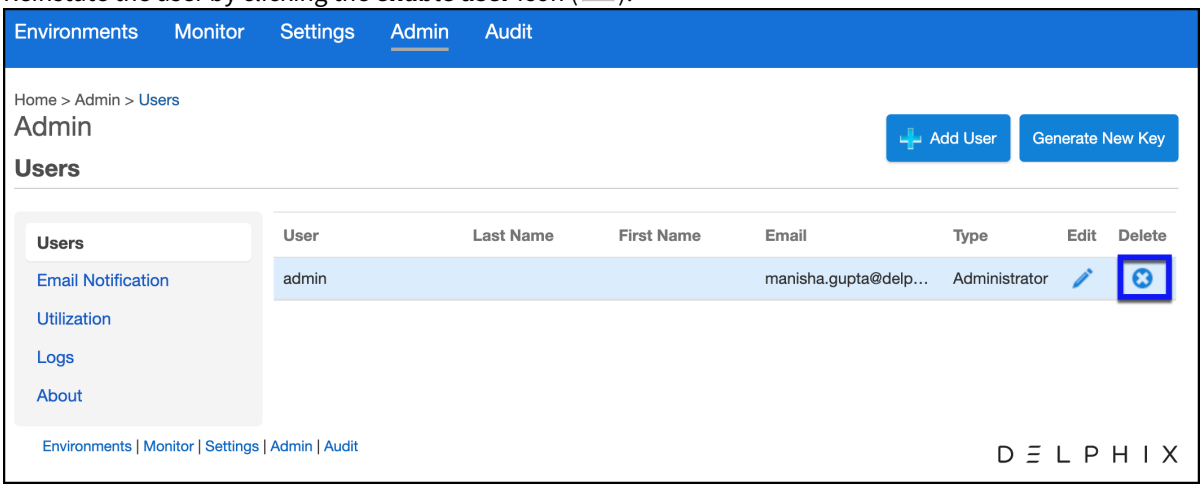
5. From the Delete User dialog select **Delete**.



> **Suspending the sysadmin user**
> The sysadmin user is a required user for the Delphix Engine. This user cannot be deleted but can be suspended. Suspending the sysadmin user prevents that user from being able to log into **Delphix Setup** or to the console via **ssh**.

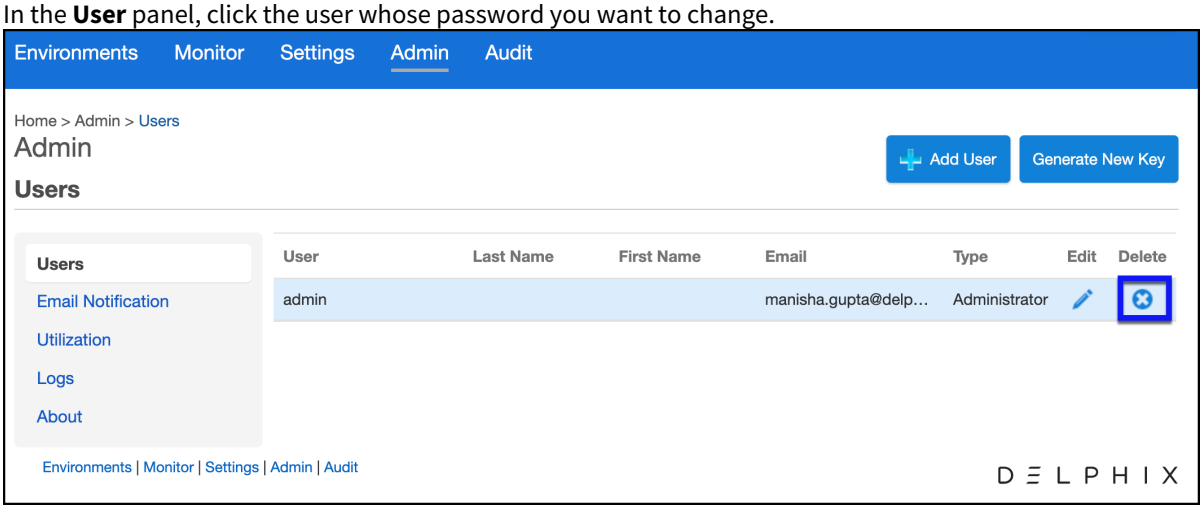
Reinstating system administrators

1. Launch the **Delphix Setup** application and login using system administrator credentials.
2. In the **User** panel, click on the name of the user you want to reinstate.
3. Reinstate the user by clicking the **enable user** icon ( ).

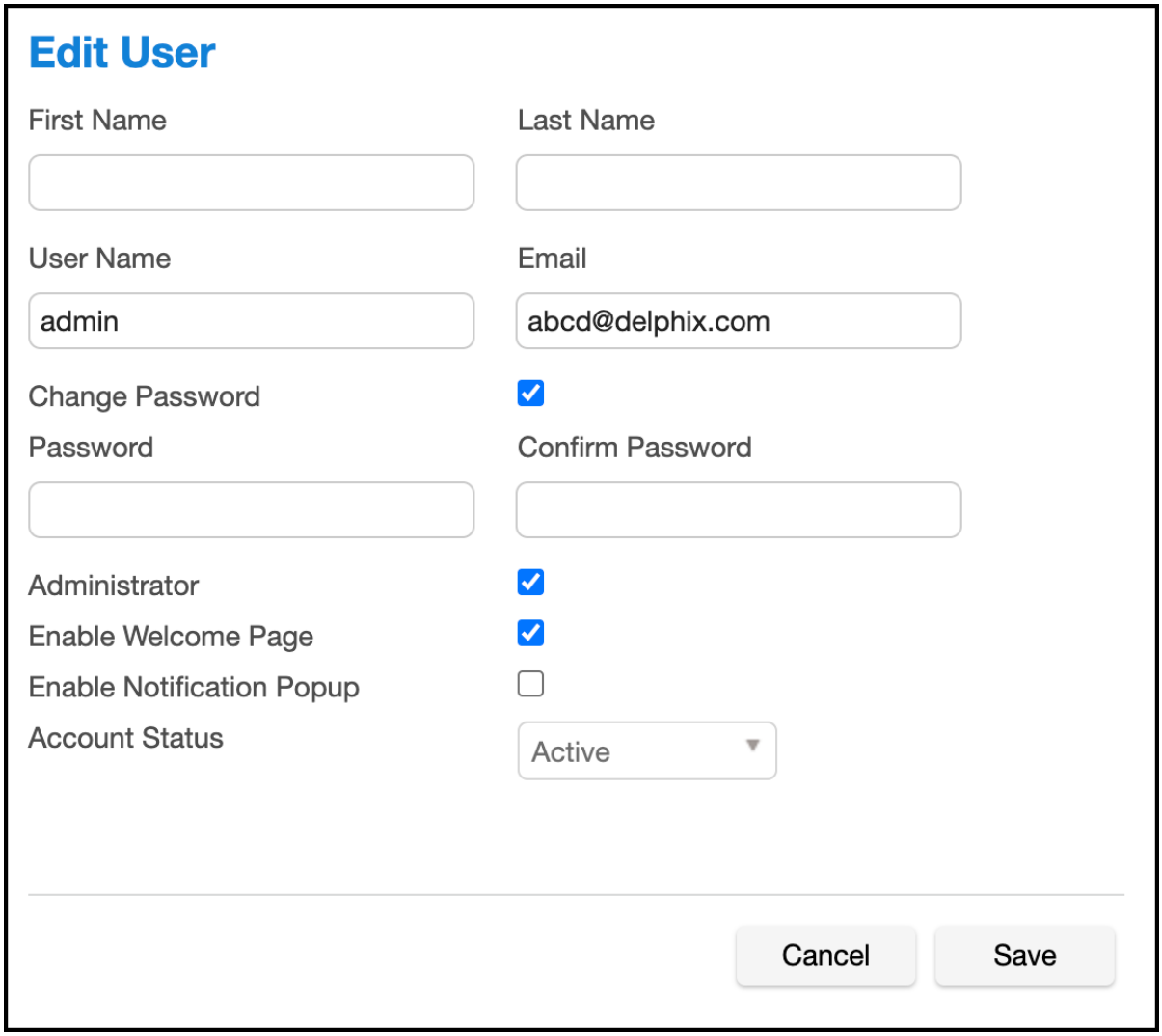


Changing system administrator passwords

1. Launch the **Delphix Setup** application and log in using **sysadmin** level credentials.
2. In the **User** panel, click the user whose password you want to change.



3. Select the **Edit** icon.
4. Enter the new password fields.



5. Click **Save**.

Adjust session timeout (command-line only)

Configuring session timeout for System Setup users is not available in the GUI. As such, the only method is via CLI using the commands below.

```
Delphix> user
Delphix user> select sysadmin
Delphix user 'sysadmin'> ls
Properties
    type: User
    name: sysadmin
    apiUser: true
    authenticationType: NATIVE
    emailAddress: user.name@domain.com
    enabled: true
    firstName: (unset)
    homePhoneNumber: (unset)
```

```
    isDefault: true
    lastName: (unset)
    locale: en-US
    mobilePhoneNumber: (unset)
    passwordUpdateRequest: NONE
    principal: sysadmin
    publicKey: (empty)
    reference: USER-1
    sessionTimeout: 30min
    userType: SYSTEM
    workPhoneNumber: (unset)

Operations
delete
update
disable
enable
updateCredential
Delphix user 'sysadmin'> update
Delphix user 'sysadmin' update *> set sessionTimeout=180
Delphix user 'sysadmin' update *> commit
```

Managing Delphix users

This section describes how to manage users. Here, you can learn how to:

- Add Users
- Edit, Delete and Suspend Users
- Manage Profile Information
- Delphix User Account Lockouts

Adding users

Prerequisites

If you intend to validate user logins using LDAP authentication, make sure a system administrator has configured LDAP.

Procedure

1. Launch the **Delphix Management** application.
2. Click **Manage**.
3. Select **Users**.
4. Click plus icon to **Add User**.
5. Enter the mandatory fields **Username**, **Email Address**, and **New Password**for the new user.
   Note:
   Rules for creating a username
   Your username:
   - must be between 1 to 256 characters
   - can be just letters, just numbers, or just any of the following special characters (_, -, ., @) or a combination of all of these. For example, a username could be just "@".
   - can start with any of the above-listed characters and is case-sensitive.

   Your password has no restrictions..
6. Select the **User Type.**
7. Click **Next.**
8. In the **Privileges** tab enter the privileges for the user.
9. Click **Next** and review the summary.
10. Click **Submit.**

> **Assigning owner and provisioner privileges**
> Assigning owner privileges at the group level conveys ownership privileges over all objects in that group. Click the **expand** icon next to each group name to see all objects in that group.
> You can also assign ownership privileges only for specific objects in a group. You do not have to assign owner or auditor privileges for all Delphix objects, only those for which you want to grant the user-specific access.

Editing, deleting, and suspending users

> **The delphix_admin user**
> The user named **delphix_admin** cannot be deleted since this is a user created by the Delphix Engine. However, you can suspend it.

⚠ When engines created before 5.3.1 are upgraded to 5.3.1 or later they will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then Disabling delphix_admin.

1. Launch the **Delphix Management** application.
2. Select **Manage > Users**.
3. Click the user's name to open the user's profile panel.
4. Click the **disable** icon to disable the user.
5. Click the **trash can** icon to delete the user.

⚠ Deleting a user cannot be undone.

Managing individual profile information

1. After logging in, click your name in the menu bar.
2. Click **Profile**.
3. Edit profile information as necessary.
4. Select options for the event level that will trigger a notification email.
5. Select a time period for **Session Timeout**.
6. Click **Password** to edit your password.
7. Click **OK** when finished.
8. Click **Privileges** to see your privileges (Auditor or Owner) for Delphix objects.

Delphix user account lockouts

User account lockouts

This feature applies to all kinds of users -- Delphix and LDAP. It also applies to usernames that do not correspond to any user in the system. A user who enters a wrong password three times in a row is "locked out" (i.e., unable to continue attempting to log in) for an initial period of 30 seconds. After three more bad login attempts, the user must wait 60 seconds, then 90 seconds, and so on.

Troubleshooting a user account lockout

The initial wait time for any future lockouts is reset to 30 seconds when the user successfully logs in or when an administrator resets the user's password. When an administrator resets a locked-out user's password, the user can immediately attempt to log in.

Managing individual profile information

This topic describes how individual users can manage personal settings such as personal information, passwords, event notifications, and session timeouts. It also describes how users can view their privileges for Delphix objects.

Procedure

1. After logging in, click your name in the menu bar.
2. Click **Profile**.
3. Edit profile information as necessary.
4. Select options for the event level that will trigger a notification email.
5. Select a time period for **Session Timeout**.
6. Click **Password** to edit your password.
7. Click **OK** when finished.
8. Click **Privileges** to see your privileges (Auditor or Owner) for Delphix objects.

Authentication mechanisms

There are various authentication mechanisms that Delphix supports besides signing in via username and password to engines via the GUI, CLI or API. Below, you'll find documentation for the following capabilities:

- Auto-authentication via SSH keys: to automatically sign in to the Delphix CLI without requiring user-input credentials
- LDAP: Directory-based authentication to Delphix engines rather than the default local access.
- Single Sign-on: Integration and support for identity providers to authenticate users.
- Kerberos: Authentication for environments and data sources using Kerberos.
- OAuth2: Offers an alternative, password-less authentication method for API access to the Delphix Engine.

Kerberos support

Kerberos support is for access to connected environments, rather than the Delphix Engine itself. This is an advanced topic and will require a solid understanding of Delphix concepts and architecture.

How to setup auto-authentication

Generally, users need to enter a username and password when logging into the Delphix CLI. There are situations in which users may find entering a password cumbersome, or manual password entry may not be possible. These situations can be alleviated by setting up auto-authentication for the Delphix CLI.

There are two basic steps:

1. Generate a public and Private RSA key pair.
2. Register the public key with the specific Delphix Engine user.

There are two methods available:

- PuTTY
- OpenSSH with OpenSSL

If the examples provided do not work for you, you may need to consult your SSH documentation, we can only provide support for the Delphix Engine side of the connection. In both examples we grant password less login to the **sysadmin** user to host **Delphix5010**.

Using PuTTY

You will need both **putty.exe** and **puttygen.exe** for this.

Launch **puttygen.exe**

Set the **Type of key to generate** to *SSH-2 RSA* and the **Number of bites in generated key** to a suitable value such as *2048*. Click **Generate**

Once it has generated the key pair, leave the password fields blank and save the public and private keys to file.

Add the full contents of the public key to each Delphix Engine user you want to allow automatic login for.

```
Delphix5010> user Delphix5010 user> select sysadmin Delphix5010 user 'sysadmin'> update Delphix5010 user 'sysadmin' update> set publicKey="ssh-rsa AAAAB3NzaC1yc2EAAAABJQAAAQEAjdQYr1WU6UPr6FZqyt3eKNJEkAe8IdKQ8hcuBWa3HvRVmUuv0Lykm5AYQlIW0B33aWusr0o+2FVTzt3/6G1lLCf7wfhCShlJsYgwgmHeEGjixK5tacFCD8r+8dALaXlv8uOlddK0A2LPXbCCCIRL7IyVEnlSbUFY8s+E/2R3owy5XSbLJLE1el5m1lQPOyUuQddAh25ruWR+1HHSAWG3p+wofOh6l7czkEcq7fPjtAZvivX90e8Ggt6JQ8bv6td7aJWObU2Y9YY0HLLHot7NQ4AT/0tXSRKAG8sIdL7tY9hbHMNHRftCLzfn7mL+Qk8TjUYni3JGB4Vyi0bmkj6nHQ== rsa-key-20160309" Delphix5010 user 'sysadmin' update> commit
```

1. In Putty, create a profile that uses the private key. In the PuTTY Connection settings set **SSH** > **Auth** >*Private key file for authentication* to the private key file you just generated
2. Next, still in the PuTTY Connection settings set **Data** > *Auto-login username* to `sysadmin@SYSTEM`.
3. Test the connection by setting the connection hostname as you normally would for PuTTY and click **Open**.

```
Connection > SSH . Auth
```

Using OpenSSH with open SSL

Generally, OpenSSH will already have default public and private keys that can be used, if not (or the default keys are password locked) you can create them this way. OpenSSL is required but OpenSSH will take care of the background OpenSSL stuff for you.

1. Create your RSA key pair to a bit length suitable for your security needs (2048 is commonly required for recent security audits)

```
$ ssh-keygen -b 2048 -t rsa -P ' ' -f /etc/ssh/ssh_host_rsa_key
```

Results in a matching public file called `/etc/ssh/ssh_host_rsa_key.pub` If you want to create different key pairs, just specify a different file path.

2. Add the full contents of the public key (`/etc/ssh/ssh_host_rsa_key.pub` in this example) to each Delphix Engine user you want to allow automatic login for.

```
Delphix5010> user Delphix5010 user> select sysadmin Delphix5010 user 'sysadmin'> update Delphix5010 user 'sysadmin' update> set publicKey="ssh-rsa AAAAB3NzaC1yc2EAAAABJQAAAQEAjdQYr1WU6UPr6FZqyt3eKNJEkAe8IdKQ8hcuBWa3HvRVmUuv0Lykm5AYQlIW0B33aWusr0o+2FVTzt3/6G1lLCf7wfhCShlJsYgwgmHeEGjixK5tacFCD8r+8dALaXlv8uOlddK0A2LPXbCCCIRL7IyVEnlSbUFY8s+E/2R3owy5XSbLJLE1el5m1lQPOyUuQddAh25ruWR+1HHSAWG3p+wofOh6l7czkEcq7fPjtAZvivX90e8Ggt6JQ8bv6td7aJWObU2Y9YY0HLLHot7NQ4AT/0tXSRKAG8sIdL7tY9hbHMNHRftCLzfn7mL+Qk8TjUYni3JGB4Vyi0bmkj6nHQ== rsa-key-20160309" Delphix5010 user 'sysadmin' update> commit
```

3. Test no password login on the command line from your client.

```
$ ssh -i /etc/ssh/ssh_host_rsa_key sysadmin@Delphix5010
```

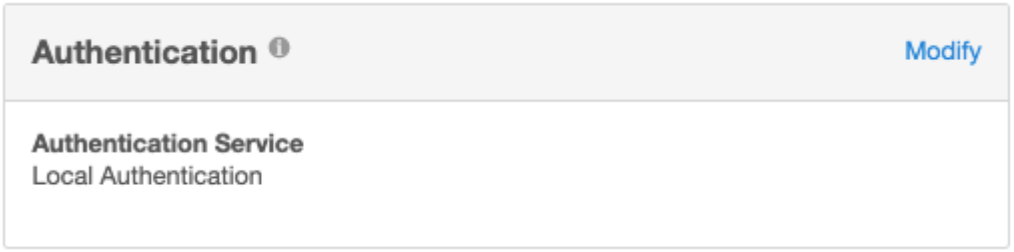
Configuring and using LDAP with the Delphix engine

Using LDAP with the Delphix Engine requires the following:

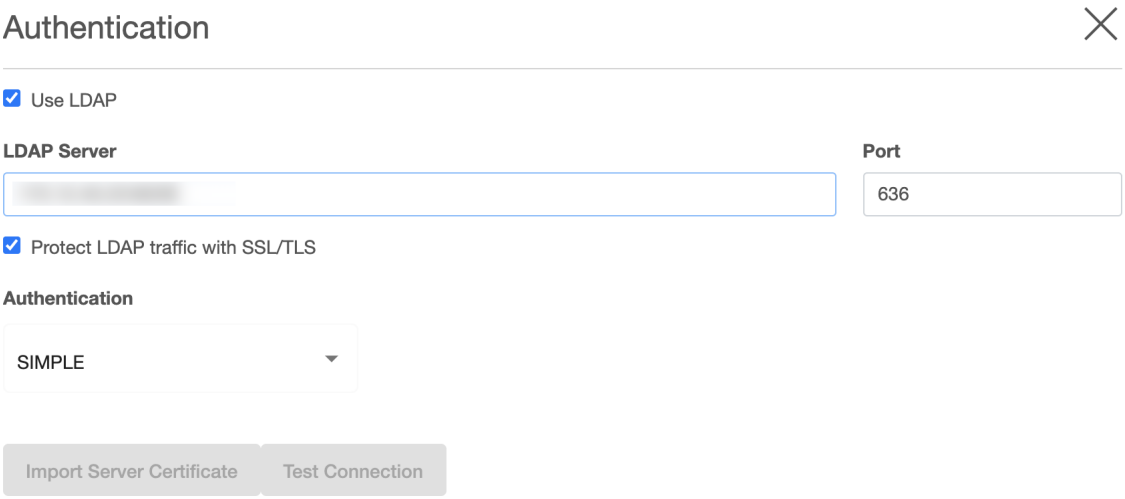
- configure the Delphix Engine to use LDAP
- add LDAP users in the Delphix Management application

Configuring LDAP on the Delphix engine

1. From the Delphix Setup application configure the LDAP server with the Delphix Engine by selecting Modify in the Authentication section.



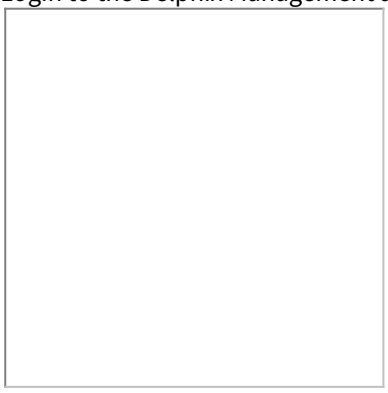
2. Enter the information about the LDAP Authentication Server. This must be an LDAP server that is configured for authentication. This information should come from the LDAP admin who runs the server. As a general rule only use simple auth. If using SSL/TLS typically use port 636 and import a certificate. If not using SSL/TLS, use port 389 and you will not need a certificate. If the remote LDAP server has disabled anonymous access and the user is trying to use SSL/TLS, the user will be unable to import the certificate. If this occurs file a support case so that Delphix Support can help manually upload the certificate.
Note:
Import Server certificate option may import more than one certificate. It is recommended to import the CA certificate in the TrustStore and then click the Test Connection option to validate. For more information on adding a certificate in the TrustStore, refer to TrustStore Settings.
Test Connect will issue an anonymous login request to the LDAP server. If the LDAP server has disabled anonymous access the test will fail. Test the server by adding a valid LDAP user and try logging in.



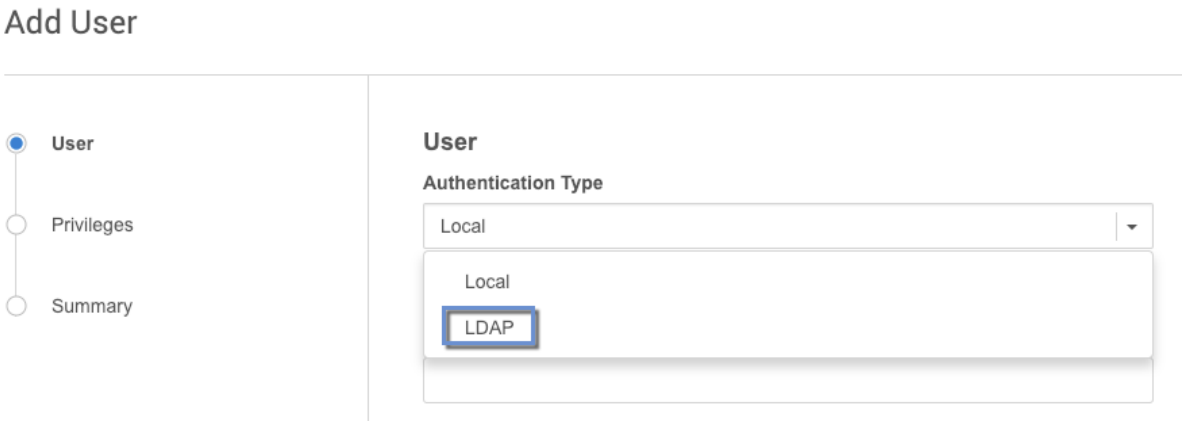
3. After updating the information and clicking the **Save** button, the *Authentication Service* section should reflect the proper information.

Create a new LDAP user account

1. Login to the Delphix Management application and go to **Manage > Users** to add a new user.



2. In the **Users** screen, click the **Add User** icon ( ) and choose **LDAP** as the *Authentication Type*.



3. Fill out the data fields and decide if the user will be a *Delphix Admin*. For more info on the *Delphix Admin* setting please see Managing System Administrators. When adding the principal, it is mandatory to specify the entire DN of the user to be added.

Each entry in an LDAP tree has a unique identifier; its Distinguished Name (DN). This consists of its Relative Distinguished Name (RDN), constructed from some attribute(s) in the entry, follow the parent entry's DN. Think of the DN as the full file path and the RDN as its relative filename in its parent folder (e.g. if /foo/bar/myfile.txt is the DN then myfile.txt would be the RDN). Some users prefer to use the term fully qualified DN to emphasize that a proper DN should include all of the components.

Example of LDAP tree in which the base is:

dc=example,dc=com

and people are stored in a People subtree with RDN:

ou=people

and each individual is keyed by the cn (common name) attribute.

An example DN in this case would be: cn=Tony,ou=people,dc=example,dc=com

When adding an LDAP user you will be asked for the following information:

- Principal - which is the DN from above
- email address
- username - used to login into Delphix

A password is no longer required because it will authenticate against the password already stored in the LDAP entry, which is presumably known to the individual already. It is probably best if someone familiar with the LDAP tree and using it for authentication were involved at least initially to help understand how to describe the fully qualified DN for users.

Using Microsoft AD as an LDAP server

Using Microsoft AD as a modified LDAP server is also possible. Microsoft AD allows some shortcuts in the specification of the DN when binding.

Examples:

- <domain>\<user logon name>
- <user logon name>@<domain>.com

As with generic LDAP, it is probably best if someone familiar with using the AD LDAP instance for authentication was involved.

When users log in, they will enter the username as chosen above, and the password that matches the principal entered above.

Configuring single sign-on

Overview

This article provides instructions on how to set up Single Sign-on (SSO) on the 5.3.3+ Delphix Engine. Delphix Engines (Masking and Virtualization) versions 5.3.3+ support authentication via the SAML 2.0 standard (SP initiated and Idp Initiated). SLO (Single log-out) is not supported. This means that logging out of a Delphix Engine will not terminate sessions on other Delphix Engines, nor will it terminate the IDP session.

Identity provider configuration

The steps to configure an Identity Provider (IdP) are specific to each IdP product (e.g. Okta, OneLogin, PingFederate). The terminology may vary, but one SAML 2.0 application (or SP connection) will need to be created for each Delphix Engine. The engine does not expose a metadata document. The following attributes must be entered:

- ACS URL (Assertion Consumer Service URL): http(s)://<delphix-engine>/sso/response
    - Delphix strongly recommends that HTTPS is used instead of HTTP for all UI and API communication with the Delphix Engine. For HTTPS or the automatic HTTP to HTTPS redirect, use the **https://** scheme in the ACS URL, otherwise use the **http://** scheme. Refer to the CLI Cookbook: Changing HTTP and HTTPS Web Connections article on how to set up HTTPS.
- SAML Bindings: Delphix Engines support the POST and Redirect bindings.
- Audience Restriction (SP entity ID, Partner's Entity ID): The audience restriction must be set to the entity id configured in the Delphix Server via the Delphix Setup (see below). The default value is **https://<Delphix Server ID>** where <Delphix Server ID> is a 36-character hexadecimal string of the form xxxxxxxx-xxxx-xxxx-xxxxxxxxxxxx. See See Determining the Delphix Server ID and Host Name for more on the Delphix Server ID.
    - If the Delphix Engine does not exist or is unreachable, enter a temporary value (such as delphix-sp-id) to later be replaced by the actual Delphix Server ID.
- Signature policy: The Delphix Engine does not sign authentication requests; it requires that either the responses, assertions, or both are signed.
- Name ID: The SAML NameID attribute must be set to the email address of the user. See the *User Management When SSO is Enabled* section below. for more information.

The SP initiated flow must be enabled in the IDP.

New engine configuration

Follow this procedure when installing a new Delphix Engine.

1. Connect to the Delphix Engine at `http://<DelphixEngine>/login/index.html#serverSetup`. The **Delphix Setup** application will launch when connecting to the server.
2. Enter the **sysadmin** login credentials; this account has a default username of **sysadmin** and password of **sysadmin**.
3. In the Authentication step of the Delphix Setup wizard, check the **use SAML/SSO** box and enter the required information:
    a. The entity id is a unique identifier of the Delphix Engine for Single Sign-On providers. The default value is **https://<Delphix Server ID>**, where <Delphix Server ID> is a 36-character hexadecimal string of the form xxxxxxxx-xxxx-xxxx-xxxxxxxxxxxx. This is just an identifier; there is no resource at that URL. This value can be changed to any string, but note that some identify providers require this to have a URL format.
    b. The IdP metadata is an XML document that must be exported from the application created in the IdP (see Identity Provider Configuration).
    c. Optional **Advanced** settings include the **response skew time** which is the maximum time difference allowed between a SAML response and the engine's current time, in seconds. If not set, it defaults to

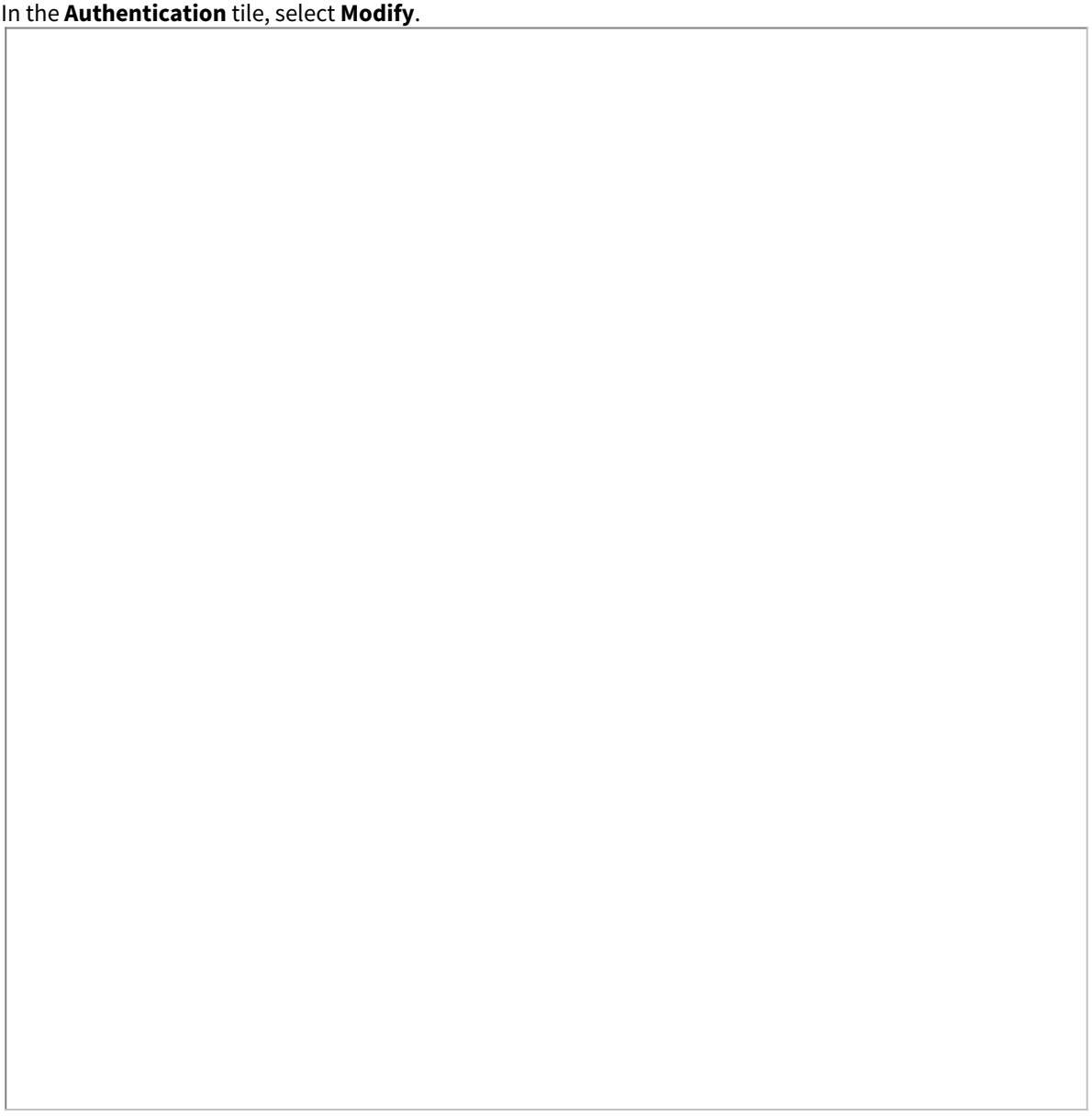
120 seconds. The **maximum age of IdP authentication** indicates how far in the past to accept authentications to the identity provider, in seconds. If not set, it defaults to 86,400 seconds (one day).

4. Complete the remaining setup steps as usual.

Existing engine configuration

Follow this procedure to enable SSO on an already configured engine.

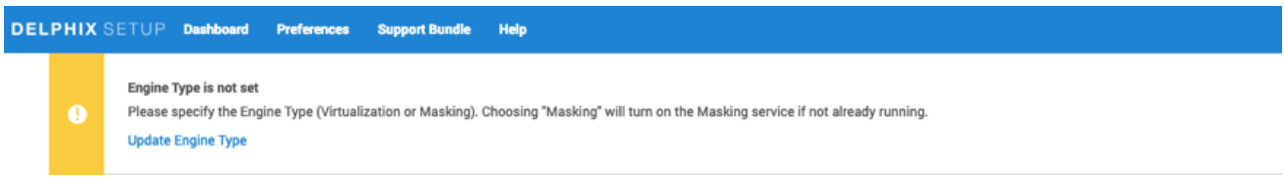
1. Connect to the Delphix Engine at `http://>>`. The **Delphix Setup** application will launch once connected to the server.
2. Enter the **sysadmin** login credentials.
3. In the **Authentication** tile, select **Modify**.



4. In the Authentication dialog, check the use SAML/SSO box, then enter the entity id (if not using the default), IdP metadata, and the optional advanced time settings (described in New Engine Configuration).

Engine type

If this is an upgraded engine, make sure the engine type is set from the banner of the ServerSetup application dashboard.



User management when SSO is enabled

Access to the Delphix Setup application is not affected by the use of SSO, it only affects access to the Delphix Management Application and Masking Application. When SSO is enabled, authentication to the Delphix Management Application or Masking Application UIs are performed via SAML/SSO instead of a combination of username and password. Non-administrators can no longer change their email address.

An administrator must create a Delphix user for each user to whom access via SSO must be granted. The Delphix user can be used to assign roles and permissions. The email address of the Delphix user is used to match users authenticated via SAML/SSO and must be set to the exact value defined in the IdP. This same value is used in the NameID attribute of the SAML response. Supported NameID formats are `urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified` and`urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress` .

If multiple Delphix users share an email address, all Delphix users will have access to the SAML/SSO session.

Once SSO is enabled, usernames and passwords (LDAP or locally stored) still need to be used for API access, and newly created user will have API access disabled by default.

API access

Neither the API access for use in scripts and integrations nor the Virtualization CLI access requires SSO. Instead, username and password (optionally with LDAP integration) authentication must be used for API or CLI access. When SSO is enabled on a Delphix Engine, new users by default will have no API access and no password. Administrators can enable API access for any user through the User Management or Masking UIs. The user's password or LDAP credentials are used for API authentication.

Users created before enabling SSO will maintain their API access enabled. Users with API access but no email address are useful for scripts or integration via the API - they cannot be used to log in via the UI. When SSO is enabled, only administrators can change or set email addresses.

A user with API access may also log in via SAML into an SSO-enabled engine through the UI when they have an email set.

Troubleshooting

If authentication via SSO fails with a message stating that the issue time is either too old or in the future, the error is due to the time on the Delphix Engine not being in sync with the time of the Identity provider server. If the time on the Delphix Engine is not correct, correct the time settings manually or configure NTP. Alternatively, update the response skew time parameter (see Existing Engine Configuration).

If authentication via SSO fails with a message stating that the authentication to the identity provider is too old, re-authenticate with the identity provider or adjust the maximum age parameter (see Existing Engine Configuration).

Configuring and managing kerberos

> **Version 6.0.7.0 or later recommended for kerberos**
> Any Delphix Engine intending to leverage Kerberos credentials should be running version 6.0.7.0 or later. Versions 6.0.0.0-6.0.6.1 may encounter issues in authentication ticket renewal, causing Environment and Dataset job failures. More information can be found in this Delphix Knowledge base article.

This section covers the following topics :

- Delphix kerberos implementation
- SSH implementation
- Kerberos requirements
- Configuring kerberos

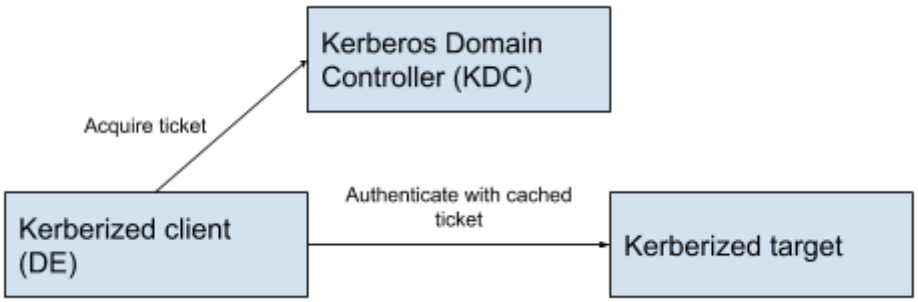
Delphix kerberos implementation

> **Version 6.0.7.0 or later recommended for Kerberos**
> Any Delphix Engine intending to leverage Kerberos credentials should be running version 6.0.7.0 or later. Versions 6.0.0.0-6.0.6.1 may encounter issues in authentication ticket renewal, causing Environment and Dataset job failures. More information can be found in this Delphix Knowledge Base article

Shared infrastructure/ticket management

The Delphix Engine (DE) has a single Kerberos principal shared between all connections to the host (SSH, ASE JDBC, etc).

Overview of the authentication process



1. The client acquires a ticket from the Kerberos Domain Controller (KDC) (e.g kinit <principal>) which it stores locally.
2. The client uses a ticket from KDC to authenticate with the target (e.g., ssh- or JDBC authentication using gssapi to pass the cached ticket acquired in step 1).

Kerberos master/replica KDCs

Kerberos supports a master/replica system with multiple KDCs running on different hosts. This is used for High Availability (HA) or to provide faster service via a local node in dispersed network environments. Delphix supports a list of KDCs for the Kerberos realm to which it has been joined.

Delphix infrastructure to support the authentication process

Kerberized environment user

Delphix has introduced a KerberosCredential type that indicates the global Kerberos principal to be used for authentication, rather than user-specific credentials.

Keytab based authentication

It is possible to use kinit with a keytab file instead of password-based authentication to acquire tickets. This is similar in principle to passwordless SSH authentication and allows Delphix to function in the customer's environment without storing any passwords on the Delphix Engine. It does, however, put us at the mercy of the customer's keytab expiration policy.

The Delphix Engine creates a background thread that periodically checks the expiration of the cached Kerberos credentials. If the credentials have expired, it calls kinit using the keytab that was provided.

Keytab file storage

Keytab file data is sent via a web service API as a Base64 encoded string. This is then decoded back to the binary file and persisted on local storage on the Delphix Engine with root user ownership permissions.

Default behavior

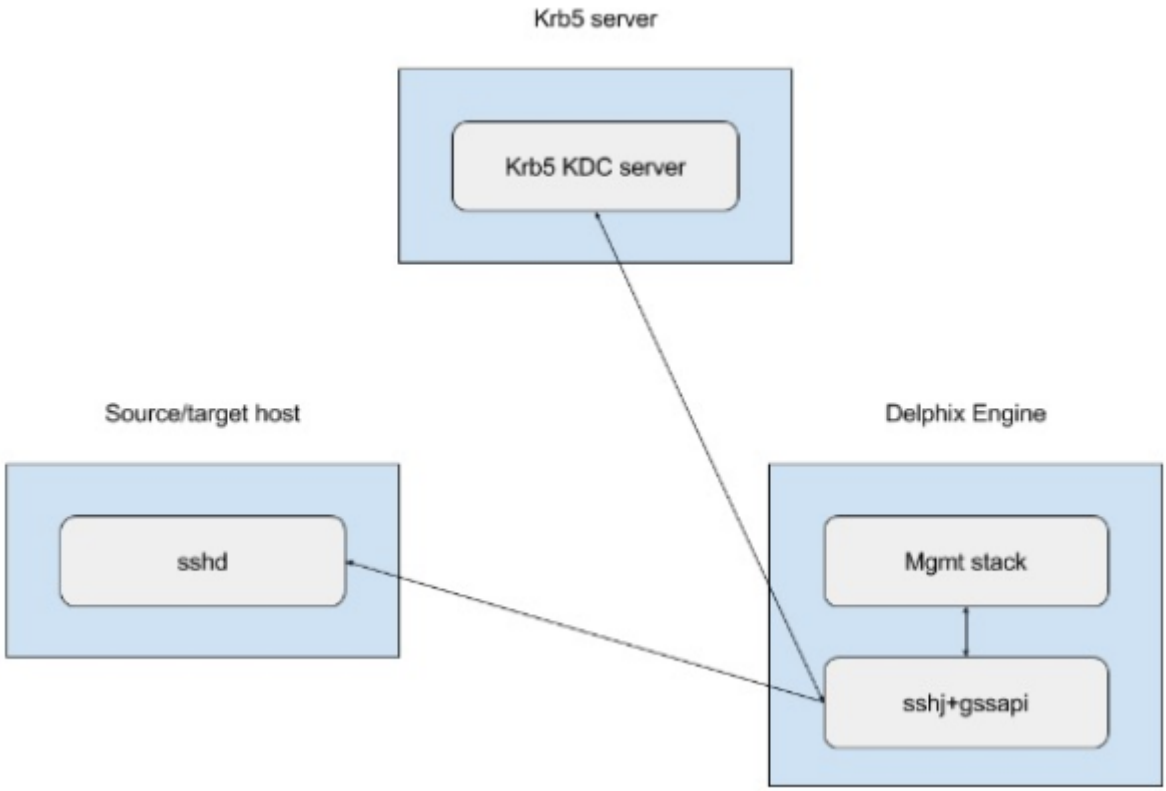
Default Kerberos ticket refresh configuration:

- Delphix checks if the TGT-cached Kerberos ticket should be refreshed every hour.
- The TGT-cached ticket for the global Delphix principal will be refreshed if it expires in less than two hours. The default values can be changed by Delphix Support.

SSH implementation

The management stack uses sshj+gssapi to pass already-generated Kerberos tickets to the Kerberized sshd on the source/target side if prompted to do so by the end-user passing a Kerberized environment user to existing wrapper functions.

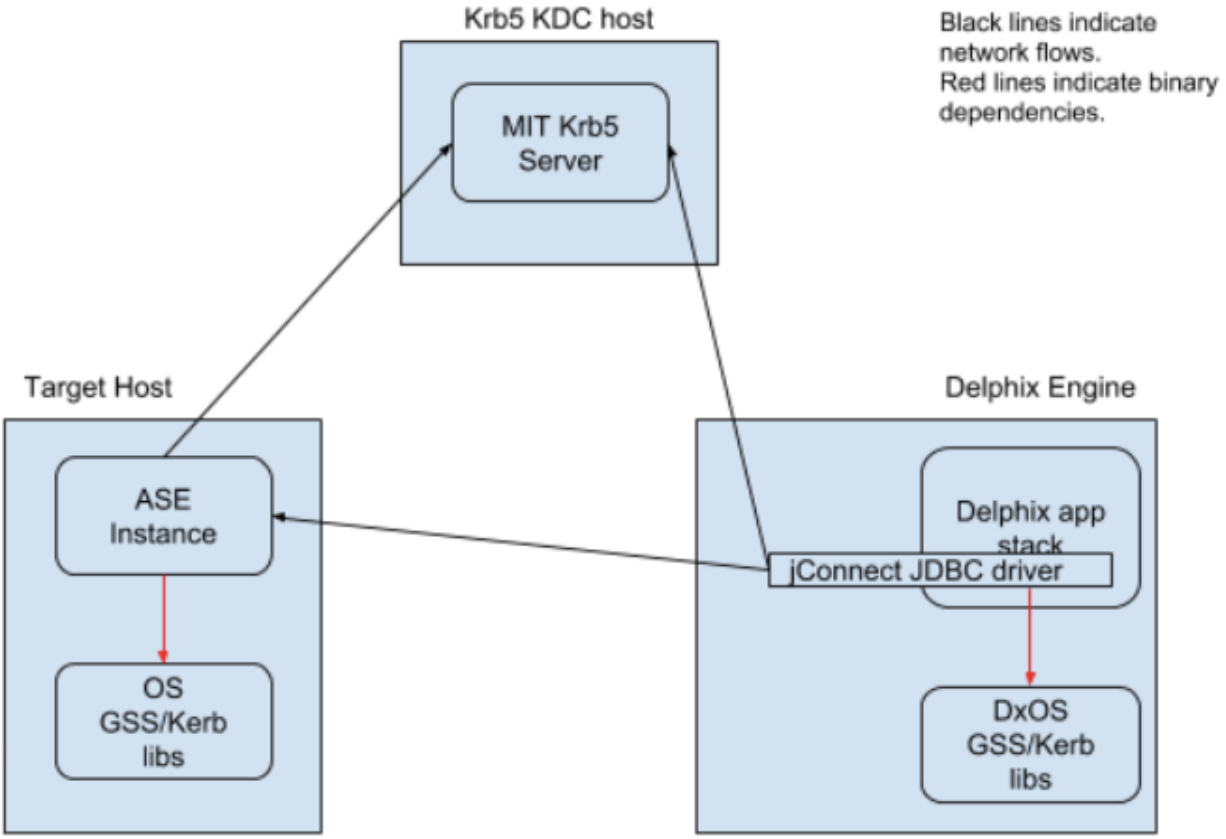
The only thing changing from password-based or regular passwordless SSH authentication is the authentication step. Command execution remains unchanged.



SAP ASE, Oracle, and DB2 connections

Delphix connects to SAP ASE, Oracle, and DB2 instances using the two listed mechanisms below. This example configuration uses an SAP ASE instance.

- via isql process
- via the jConnect JDBC driver



When connecting via isql Delphix uses the "-V" parameter rather than specifying a username/password. The "-V" option uses the Kerberos principal in the current user's cached credentials file. Delphix relies on the end customer to configure this appropriately for their environment (for example, the cached credentials could be populated by a PAM module during login). Delphix also expects that the `KRB5CCNAME` is set appropriately or the credential cache is in the host default location.

When connecting via JDBC, Delphix uses additional connection options: `REQUEST_KERBEROS_SESSION=true&SERVICE_PRINCIPAL_NAME=`. By default, the instance Service Principal Name (SPN) is identical to the instance name for authentication. Delphix allows the instance SPN to be manually set on a per-repository basis to allow for non-default values. The jConnect JDBC driver connects using the cached credentials that were obtained as described in the Shared infrastructure/Ticket Management section.

For example, if the instance name is `ASE_INSTANCE_1` and has been configured to use `REALM.COM`, then the instance will attempt to authenticate with the KDC using `ASE_INSTANCE_1@REALM.COM`. However, this is configurable and can be specified either via an environment variable or a command-line option to the data server process. If an environment variable is used to configure the SPN, the instance must be manually discovered via web service APIs or the Delphix CLI.

Kerberos requirements

Prerequisites

Basic requirements prior to the configuration:

- MIT Kerberos 1.4.4 KDC
- Kerberos `REALM` name
- Global Kerberos principal name (specified without trailing `@REALM` name)
- Global Kerberos principal `keytab` data encoded as a base 64 string
- KDC hostnames and port numbers (one or more in priority list order)

Environment requirements

> **Version 6.0.7.0 or later recommended for Kerberos**
> Any Delphix Engine intending to leverage Kerberos credentials should be running version 6.0.7.0 or later. Versions 6.0.0.0-6.0.6.1 may encounter issues in authentication ticket renewal, causing Environment and Dataset job failures. More information can be found in this Delphix Knowledge Base article.

The following hosts and software versions are required:

- A source host with the following configuration:
  - A running SAP ASE, Oracle, or DB2 instance.
  - A database to link from and its corresponding full database dump.
  - The Delphix principal is able to access the SAP ASE instance and SSH onto the host as per our product documentation (see Requirements for SAP ASE).
  - The credential cache for the Delphix principal is populated and kept current. The environment variable `KRB5CCNAME` is set to the location for a credential cache. Login to the Adaptive Server via "`isql_r64 -V -R>><>`" or "`isql_r -V -R>><>`" or otherwise make sure that "`isql`" points to either "`isql_r64 or isql_r`" so that "`"isql -V -R>><>` works.
- A staging host with the following configuration:
  - A running SAP ASE, Oracle, or DB2 instance with the same version as the source instance.
  - The Delphix principal is able to access the SAP ASE instance and SSH onto the host as per our product documentation (see Requirements for SAP ASE).
  - The credential cache for the Delphix principal is populated and kept current. The environment variable `KRB5CCNAME` is set to the location for a credential cache. Login to the Adaptive Server via "`isql_r64 -V -R>><>`" or "`isql_r -V -R>><>`" or otherwise make sure that "`isql`" points to either "`isql_r64 or isql_r`" so that "`"isql -V -R>><>` works.
- A target host to create a VDB on. Configuration details:
  - A running SAP ASE, Oracle, or DB2 instance with the same version as the source instance.
  - The Delphix principal is able to access the SAP ASE instance and SSH onto the host as per our product documentation (see Requirements for SAP ASE).
  - The credential cache for the Delphix principal is populated and kept current. The environment variable `KRB5CCNAME` is set to the location for a credential cache. Login to the Adaptive Server via "`isql_r64 -V -R>><>`" or "`isql_r -V -R>><>`" or otherwise make sure that "`isql`" points to either "`isql_r64 or isql_r`" so that "`"isql -V -R>><>` works.

Supported databases and Kerberos configurations

For detailed Kerberos support please refer to Kerberos Support Matrix

Delphix Engine 5.3.2.0 is the first generally available customer release to support Kerberos on a subset of supported OS' with SAP ASE, Oracle, and DB2 databases.

With the Delphix Engine 5.3.2.0 release, many of the previously unsupported GUI functions now function when Kerberos is enabled. See Configuring Kerberos via the UI below for details, including limitations.

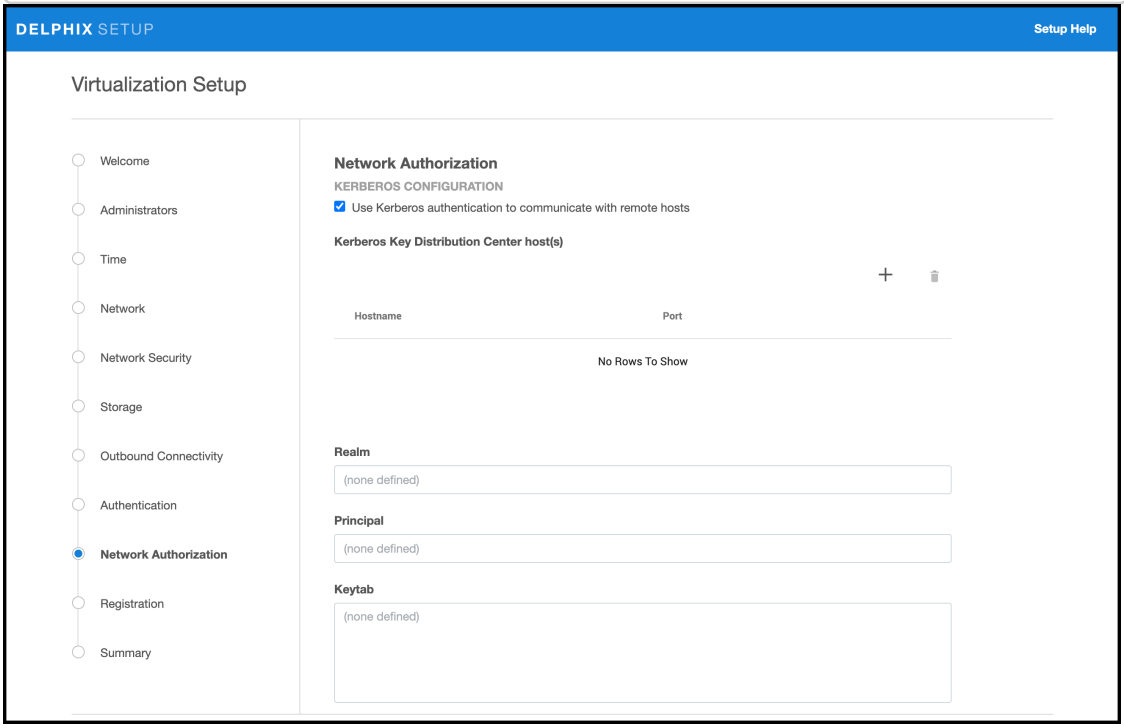
Configuring Kerberos during engine setup

> **Version 6.0.7.0 or later recommended for Kerberos**
> Any Delphix Engine intending to leverage Kerberos credentials should be running version 6.0.7.0 or later. Versions 6.0.0.0-6.0.6.1 may encounter issues in authentication ticket renewal, causing Environment and Dataset job failures. More information can be found in this Delphix Knowledge Base article.

For using the kerberos authentication along with another authentication,

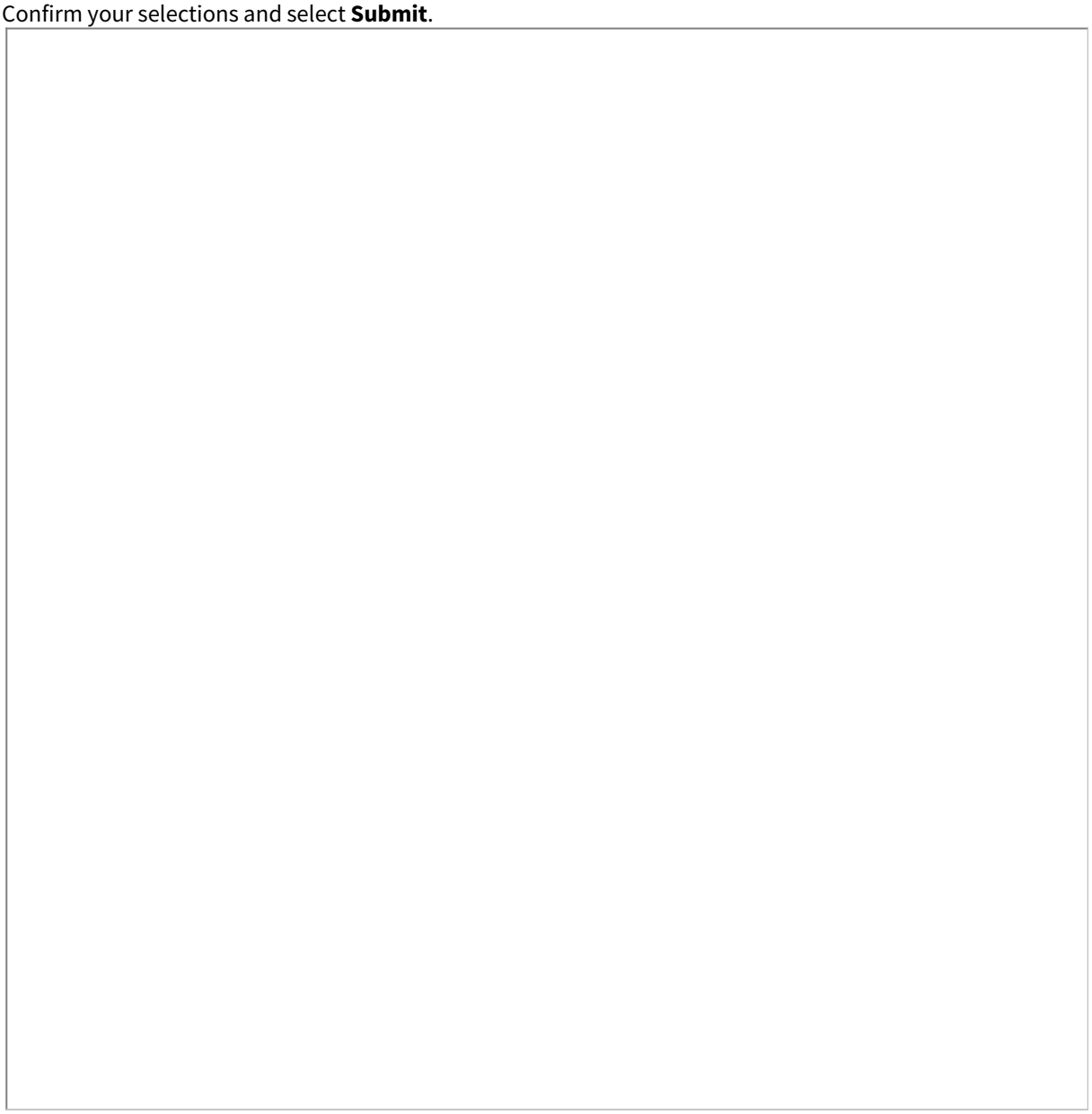
1. In the **Virtualization Setup** wizard, Click **Next** on the **Welcome** screen.
2. Complete the following tabs as you normally would.
3. In the **Network Authorization** tab, select the **Use Kerberos authentication to communicate with the remote hosts** checkbox.
4. Provide values in the following fields:
   - **Realm**
   - **Principal**
   - In the **Keytab** field, enter the key in a base64 encoded format.
     Note:
     You must provide the key must in a single line. For example, you can use the following command to view the base64 representation of the Keytab key.

```
echo "$(base64 --wrap=0 <keytab>)"
```



5. Then select **Next**.

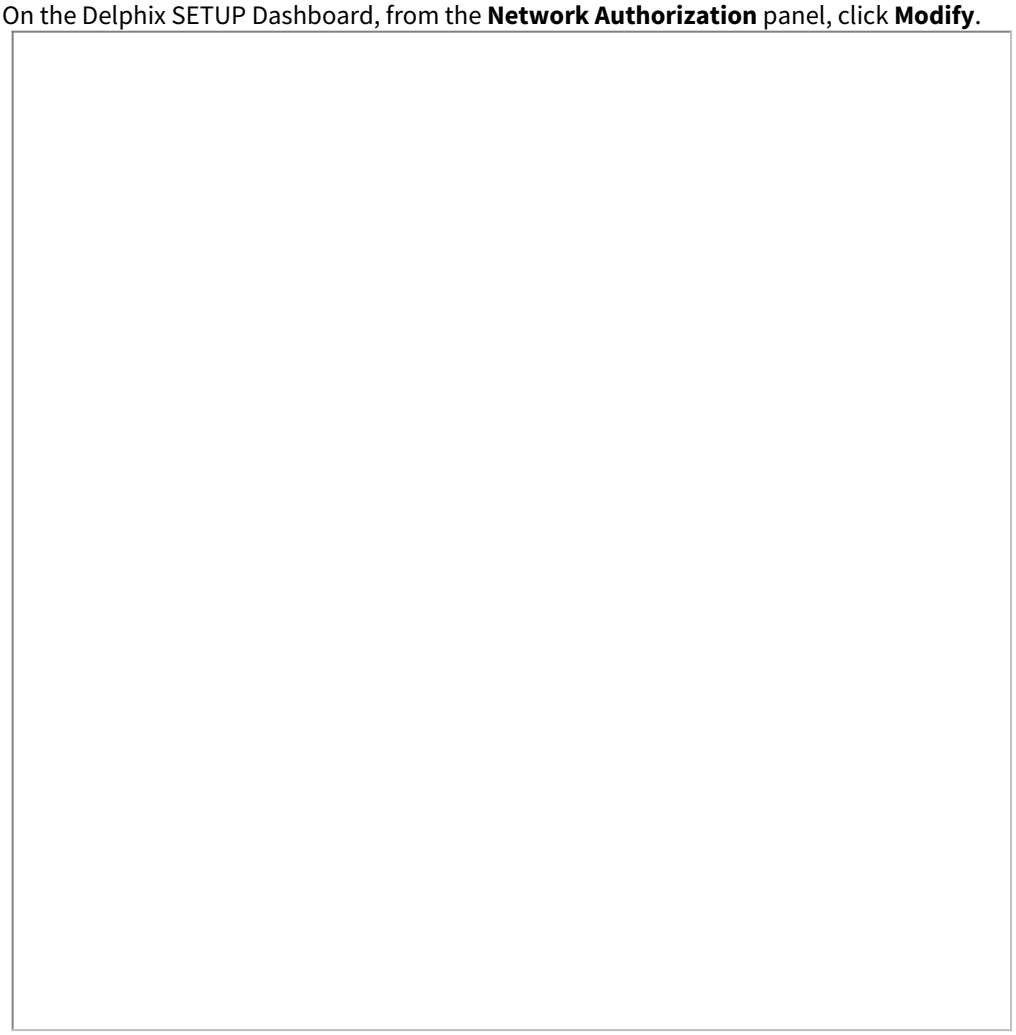
6. Confirm your selections and select **Submit**.



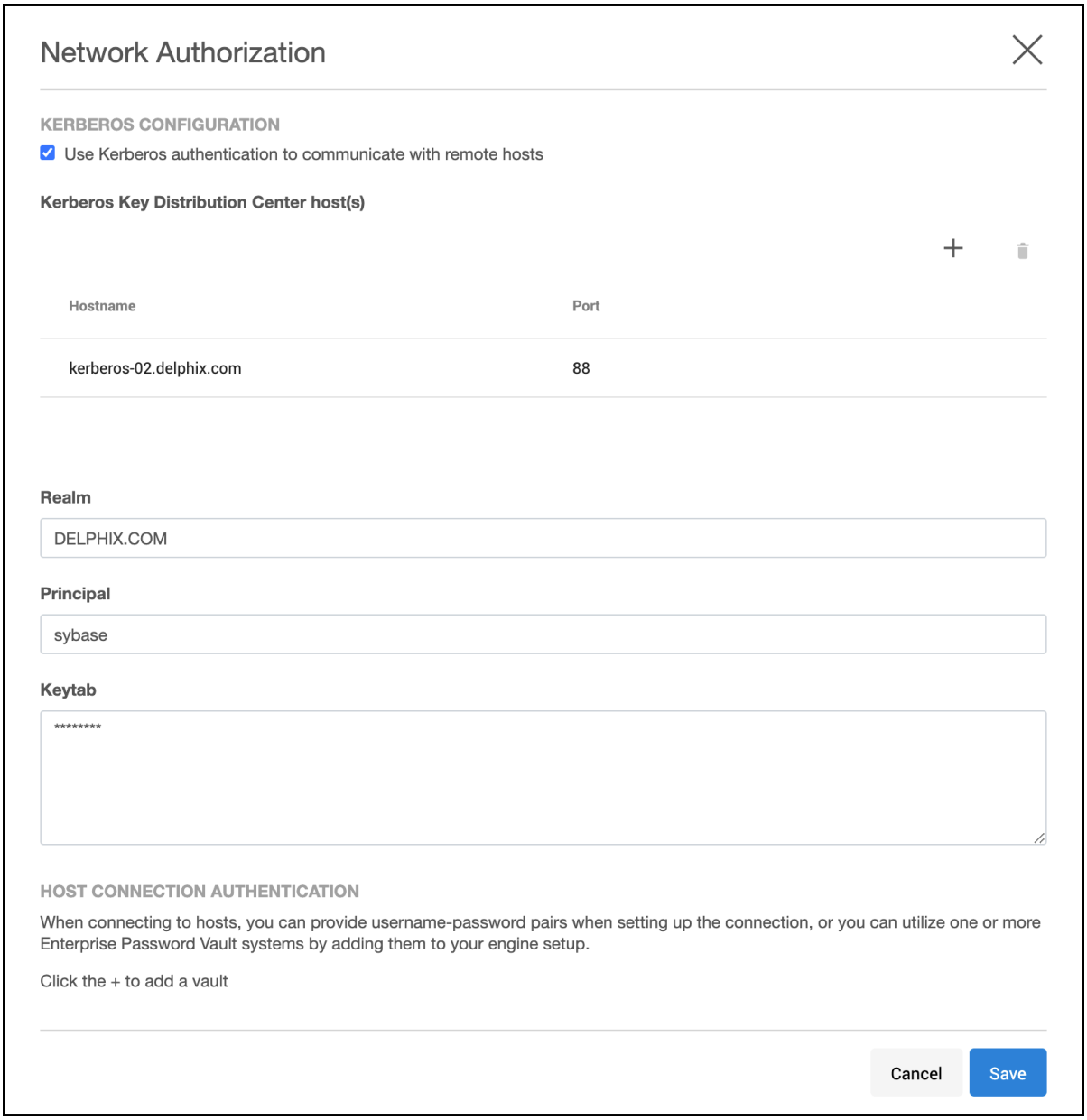
Editing Kerberos configuration settings

To edit your configuration settings after setup, login as a sysadmin user and complete the following steps:

1. On the Delphix SETUP Dashboard, from the **Network Authorization** panel, click **Modify**.



2. Select the check box under KERBEROS CONFIGURATION, select the hostname and edit the configuration.


Network Authorization
KERBEROS CONFIGURATION
Use Kerberos authentication to communicate with remote hosts
Kerberos Key Distribution Center host(s)
Hostname
Port
kerberos-02.delphix.com
88
Realm
DELPHIX.COM
Principal
sybase
Keytab
****
HOST CONNECTION AUTHENTICATION
When connecting to hosts, you can provide username-password pairs when setting up the connection, or you can utilize one or more Enterprise Password Vault systems by adding them to your engine setup.
Click the + to add a vault
Cancel
Save

3. Click **Save**.

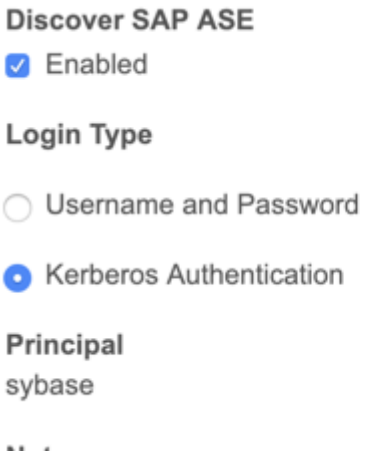
Add a Kerberos environment

.

> **Version 6.0.7.0 or later recommended for Kerberos**
> Any Delphix Engine intending to leverage Kerberos credentials should be running version 6.0.7.0 or later. Versions 6.0.0.0-6.0.6.1 may encounter issues in authentication ticket renewal, causing Environment and Dataset job failures. More information can be found in this Delphix Knowledge Base article

It is possible to create a Unix/Linux Standalone environment with Kerberos authentication.

1. Login to the **Delphix Management** application as an **admin**.
2. From the **Manage** menu, select **Environments**.
3. Then click on the plus icon to open a wizard to create a new environment.
4. Select your **Host OS** and **Server Type,** then select **Next.**
5. Under **Login Type**, select **Kerberos Authentication**.
6. If **Discover SAP ASE** is enabled, ASE DB Kerberos authentication will be available, select **Kerberos Authentication**.



7. Select **Submit**.

Changing the environment user

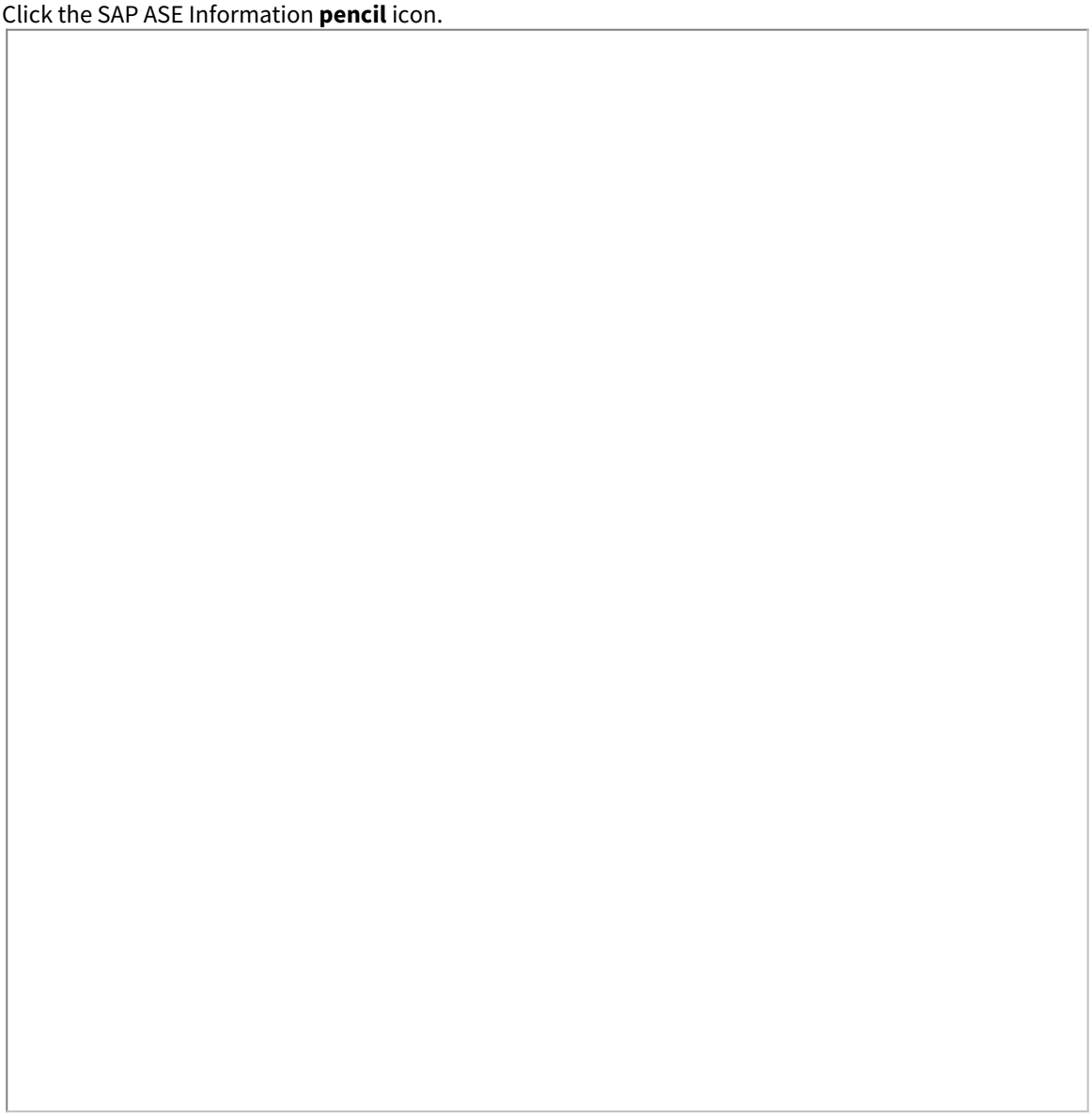
1. Login to the **Delphix Management** application as an **admin**.
2. From the **Manage** menu, select **Environments**.
3. Select an Environment, the Details tab allows you to see Environment information.
4. On the grid of Environment Users, you can see existing users. Click the **plus icon**, to add a new user.
5. It is now possible to create a user with Kerberos Authentication. There can only be one Kerberos user per environment. The Principal is taken to the Kerberos Configuration.

It is possible to set the Kerberos user as Primary.

Changing SAP ASE DB user and DB password

1. Login to the **Delphix Management** application as an **admin**.
2. From the **Manage** menu, select **Environments**.
3. Select an Environment, the Details tab allows you to see Environment information.

4. Click the SAP ASE Information **pencil** icon.



5. Click the edit icon and edit ASE DB User and ASE DB Password.
6. Click the **checkmark** to save.

Updating environment notes

1. Login to the **Delphix Management** application as an **admin**.
2. From the **Manage** menu select **Environments**.
3. Select an Environment, the Details tab allows you to see Environment information.
4. Under **SAP ASE Information**, you can see **Notes**. To edit click the **pencil** icon.
5. Click the **checkmark** to save.

Changing host address, SSH port number, and toolkit path

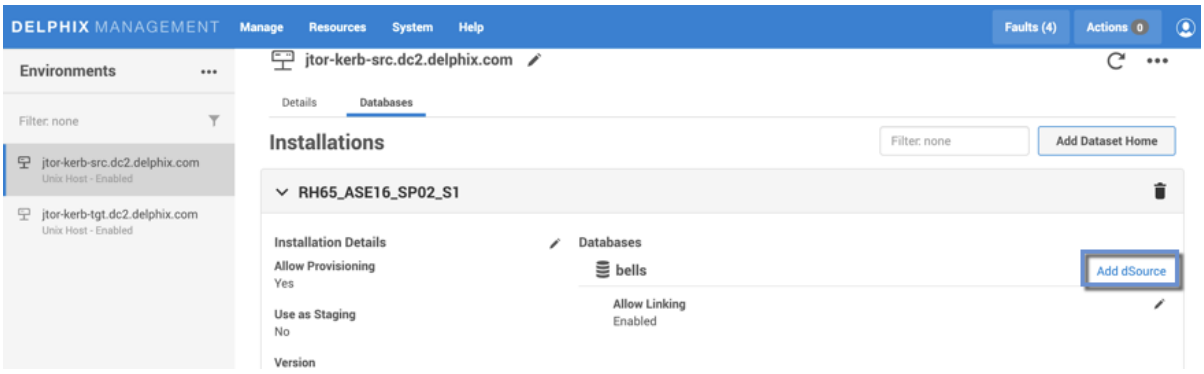
1. Login to the **Delphix Management** application as an **admin**.
2. From the **Manage** menu select **Environments**.
3. Select an Environment, the Details tab allows you to see Environment information.

4. Click the **pencil** icon located next to **Attributes** to edit the Host Address or SSH Port number or Toolkit Path.
5. Select the **checkmark** to save.

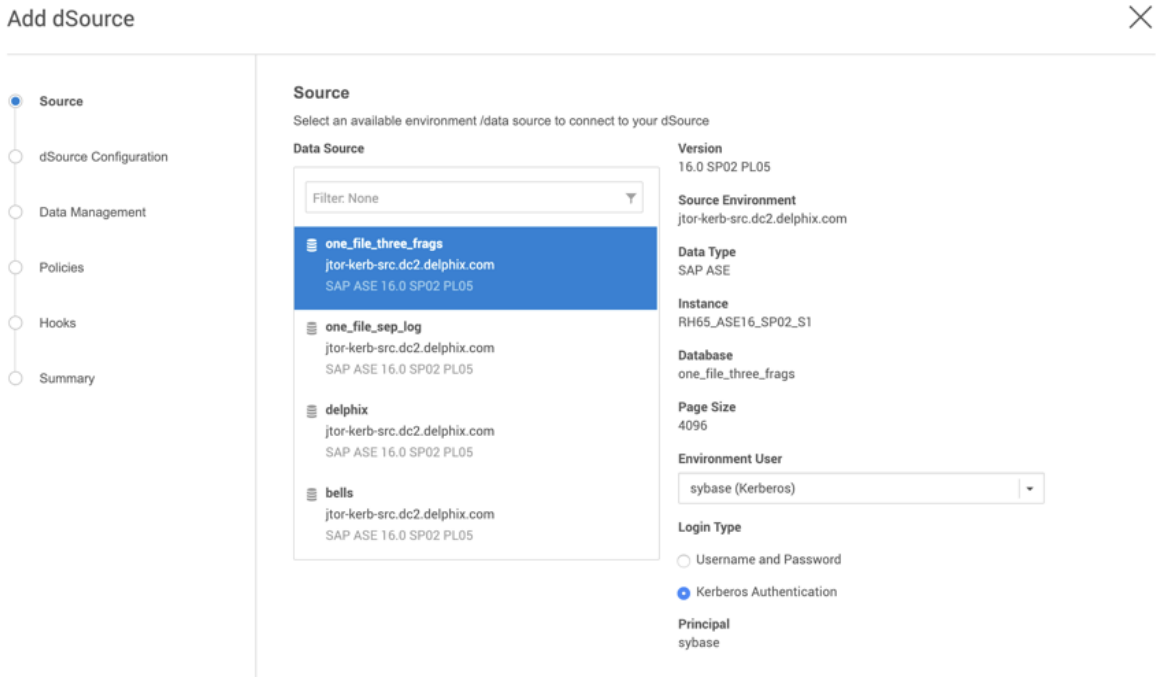
Adding and linking a dSource

Adding a dSource

1. Login to the **Delphix Management** application as an **admin**.
2. From the **Manage** menu select **Environments**.
3. Select an Environment and click the **Databases** tab. This tab provides details on all the available repositories.
4. Click on the **Add dSource** link.



5. Click on a **Data Source**, then select **Kerberos Authentication**.



6. Complete all remaining fields, and then click **Submit**.

Linking a dSource

1. Login to the **Delphix Management** application as an **admin**.
2. From the **Manage** menu select **Datasets**.
3. Select a **Dataset**.
4. From the Actions menu located on the top-right select **Link dSource**.
   All information in the Link dSource screen is automatically filled in with the data you previously selected.

Configuring OAuth2 authentication for API access

Overview

This article provides instructions on how to set up Open Authorization 2.0 (OAuth2) on the 6.0.11 Delphix Engine. OAuth2 offers an alternative, password-less authentication method for API access to the Delphix Engine.

Delphix Engine (Masking and Virtualization) version 6.0.11.0 supports authentication using JSON Web Tokens (JWTs) issued by a known authorization server or identity provider (IdP). It is necessary for JWTs to contain a claim that can be used to associate an authentication request with a user that exists in the Delphix Engine. This article describes how to configure the Delphix Engine to validate tokens and associate token claims with Delphix Engine users.

> ⚠ The authentication feature described in this section differs from the API token authentication feature supported by Delphix Engines registered with Data Control Tower (*formerly* Central Management). For more information on API token authentication, refer to Data Control Tower

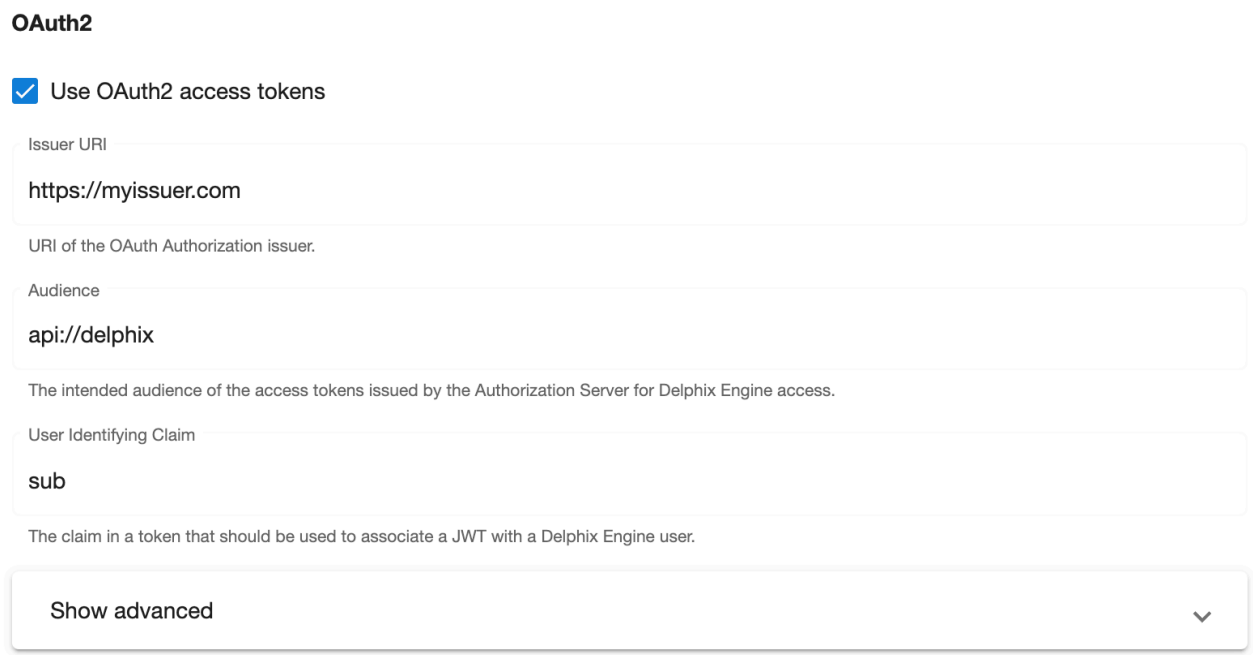
Configuration options

The following options for configuring OAuth2 for the Delphix Engine are available in the Delphix CLI (as the sysadmin user, under `service; oauth2`) as well as via the API endpoint `/resources/json/delphix/service/oauth2`.

Option	Description	Default (if applicable)
`audience`	Specifies the expected value of the audience claim (`aud`) of JWTs indicating that the tokens are intended for this particular Delphix engine.	`api://delphix`
`enabled`	Specifies whether the OAuth2 feature should be enabled.	`false`
`issuerURI`	(**Required**) Specifies the base location or identifier of the authorization server (IdP) which the Delphix Engine will use to validate incoming JWTs.	
`jwkSetURI`	Specifies the URI used to retrieve the JSON Web Key (JWK) set, if supported by the authorization server (IdP).	
`tokenSkewTime`	Specifies the maximum time difference (in seconds) allowed between the validity period of a JWT and the engine's current time.	`60`

Option	Description	Default (if applicable)
`userIdClaim`	Specifies which claim in a JWT should be used by the Delphix Engine (in conjunction with the `userMatchingFieldType` setting) to associate a token with a user configured on the Delphix Engine. The default (`sub`) corresponds to the subject claim of the token.	`sub`
`userMatchingFieldType`	Specifies which property of a Delphix Engine user will be used to match with the claim (specified in `userIdClaim`) of a JWT. The Delphix Engine can be configured to match users based on a user's `name` , `emailAddress` , or `principal` properties.	`PRINCIPAL`

These options may also be set in the Delphix Setup application. Please refer to the following procedures for configuring OAuth2 parameters for new or existing Delphix Engines.



New engine configuration

Follow this procedure to configure OAuth2 on a new Delphix Engine.

1. Connect to the Delphix Engine at `http://>>` . The **Delphix Setup** application will launch when connecting to the server.
2. Enter the **sysadmin** login credentials; this account has a default username of **sysadmin** and password of **sysadmin**.
3. In the Authentication step of the Delphix Setup wizard, check the **Use OAuth2 access tokens** box and configure the OAuth2 settings as desired. (Refer to the table above for more information on these settings.)

4. Complete the remaining setup steps as usual.

Existing engine configuration

Follow this procedure to configure OAuth2 on an existing Delphix Engine.

1. Connect to the Delphix Engine at `http://>>`. The **Delphix Setup** application will launch when connecting to the server.
2. Enter the **sysadmin** login credentials.
3. In the **Authentication** tile, select **Modify**.

4. In the Authentication dialog, check the **Use OAuth2 access tokens** box and configure the OAuth2 settings as desired. (Refer to the table above for more information on these settings.)

Example of API access using OAuth2 token

1. Obtain a JWT from the Authorization Server (IdP). (The details for this process will vary depending on the IdP vendor.) For the purposes of this example, the contents of the token are stored in the environment variable `t`.
2. Access the oauth2-login API endpoint of the Delphix Engine, providing the OAuth2 token. In this example the session information is stored in the file `cookies.txt` in the working directory. For Virtualization Engines the oauth2-login API endpoint is `/virtualization/api/oauth2-login`. For Masking Engines the oauth2-login API endpoint is `/masking/api/oauth2-login`.
Virtualization API Endpoint Example

```
h=<engine address>; curl -i -X POST $h/virtualization/api/oauth2-login -H
"Authorization: Bearer $t" -b cookies.txt -c cookies.txt -H 'Content-Type:
application/json' -d '{"type": "APISession", "version": {"type":
"APIVersion","major": 1,"minor": 11,"micro": 11}}'
```

Masking API Endpoint Example

```
h=<engine address>; curl -i -X POST $h/masking/api/oauth2-login -H
"Authorization: Bearer $t" -H 'Content-Type: application/json' -b cookies.txt
-c cookies.txt
```

3. Refer to the saved cookies.txt file in subsequent `curl` invocations. The example below can be used to list the users configured on the Delphix Engine.
Virtualization API Example

```
curl -X GET -b cookies.txt -c cookies.txt -H 'Content-Type: application/json'
 $h/resources/json/delphix/user
```

Masking API Example

```
# Store authorization code returned by /masking/api/oauth2-login in $m
curl -i -X GET -b cookies.txt -c cookies.txt -H 'Content-Type: application/
json' -H "Authorization: $m" $h/masking/api/v5.1.11/users
```

CLI access using OAuth2 token

When OAuth2 is enabled, CLI logins will prompt the user to supply either an OAuth2 access token or a password to authenticate:

CLI Access Prompt with OAuth2 Enabled

```
$ ssh myengine -l admin
Enter access token or press enter to provide a password:
```

To authenticate using an OAuth2 token, paste its contents when this prompt is shown. As with API authentication, the OAuth2 token must be current (not expired) and must contain a claim that can be associated with a valid Delphix Engine user, based on the **userIdClaim** and **userMatchingFieldType** values set on the Delphix Engine.

To bypass OAuth2 authentication and use password authentication, press **Enter** when this prompt is shown and a conventional password prompt will be displayed.

User matching policy

If the `userIdClaim` component of a JWT matches more than one Delphix Engine user (for example, if `userMatchingFieldType` is set to `EMAIL_ADDRESS`, and the same email address is associated with multiple Delphix Engine users), the oldest user account (by time of creation) will be authenticated.

> Suggestion
> To ensure all users can be authenticated using OAuth2, make sure that the property specified in `userMatchingFieldType` is populated and unique for all Delphix Engine users.

HTTPS Proxy Configuration

If the OAuth2 identity provider cannot be directly reached by a Delphix Engine, a HTTPS proxy may be used. The sysadmin user can create or modify HTTPS proxy settings via the CLI/API endpoint `service; proxy`, or in the **Delphix Setup** application by selecting to modify the **Outbound Connectivity** tile, checking the **Configure web proxy** box, and entering the host, port, and (optionally) username and password of the proxy server.

Network and DNS management

You may want to manage and configure certain network services, such as DNS, for Delphix. Here we specify general network and connectivity requirements, as well as detail how you can test network performance. General Network and Connectivity Requirements.

This section covers the following topics:

General network and connectivity requirements

Overview

This topic covers the general network and connectivity requirements for the Delphix Engine, including connection requirements, port allocation, and firewall and Intrusion Detection System (IDS) considerations. For platform-specific network and connectivity requirements, see the relevant topics under the **Requirements** section for each platform.

General outbound from the Delphix engine port allocation

Protocol	Port numbers	Use
TCP	25	Connection to a local SMTP server for sending email
TCP/UDP	53	Connections to local DNS servers
UDP	123	Connection to an NTP server
UDP	162	Sending SNMP TRAP messages to an SNMP Manager
TCP	443	HTTPS connections from the Delphix Engine to the Delphix Support upload server
TCP/UDP	636	Secure connections to an LDAP server
TCP	8415	Connections to a Delphix replication target. See Configuring Replication
TCP	50001	Connections to source and target environments for network performance tests.

General inbound to the Delphix engine port allocation

Protocol	Port number	Use
TCP	22	SSH connections to the Delphix Engine
TCP	80	HTTP connections to the Delphix GUI
UDP	161	Messages from an SNMP Manager to the Delphix Engine
TCP	443	HTTPS connections to the Delphix Management Application

Protocol	Port number	Use
TCP	8415	Delphix Session Protocol connections from all DSP-based network services including Replication, SnapSync for Oracle, V2P, and the Delphix Connector.
TCP	50001	Connections from source and target environments for network performance tests via the Delphix CLI.

Firewalls and intrusion detection systems (IDS)

Production databases on source environments (for dSources) are often separated from the non-production environment by firewalls. Firewalls can add milliseconds to the latency between servers. Accordingly, for best performance, there should be no firewalls between the Delphix Engine and the virtual database (VDB) target environments. If the Delphix Engine is separated from a source environment by a firewall, the firewall must be configured to permit network connections between the Delphix Engine and the source environments for the application protocols (ports) listed above.

Intrusion detection systems (IDSs) should also be made permissive to the Delphix Engine deployment. IDSs should be made aware of the anticipated high volumes of data transfer between dSources and the Delphix Engine.

Network performance configuration options

This section covers the following topics:

Optimal network architecture for the Delphix engine

This topic describes basic network performance considerations for the Delphix Engine.

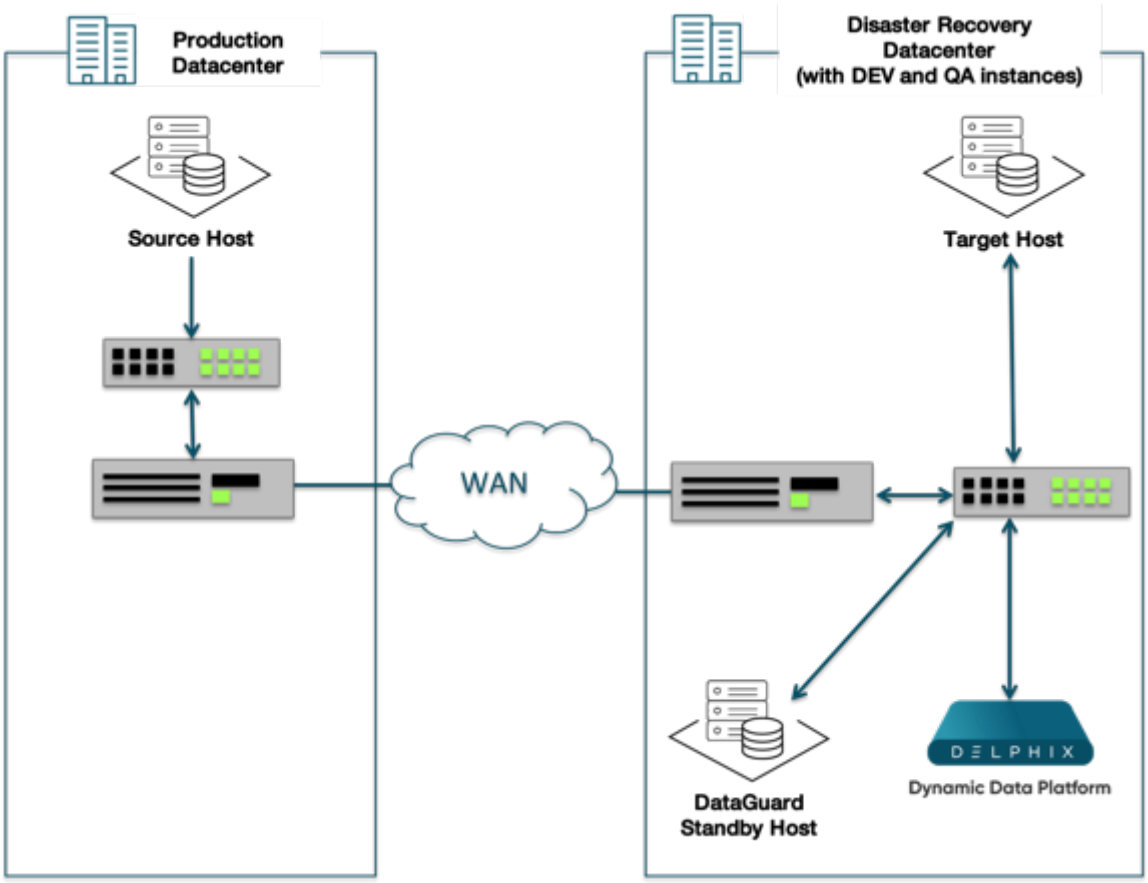
Network architecture and latency

All VDB I/O operations are serviced over the network. Delphix uses NFS as the primary transport for Oracle VDBs, and iSCSI for MS SQL VDBs. The network architecture, latency, and capacity between the Delphix Engine and the target environment are key network components for improving the performance of a Delphix deployment. The latency between the Delphix Engine and the source environment is not relevant for the best performance of VDBs.

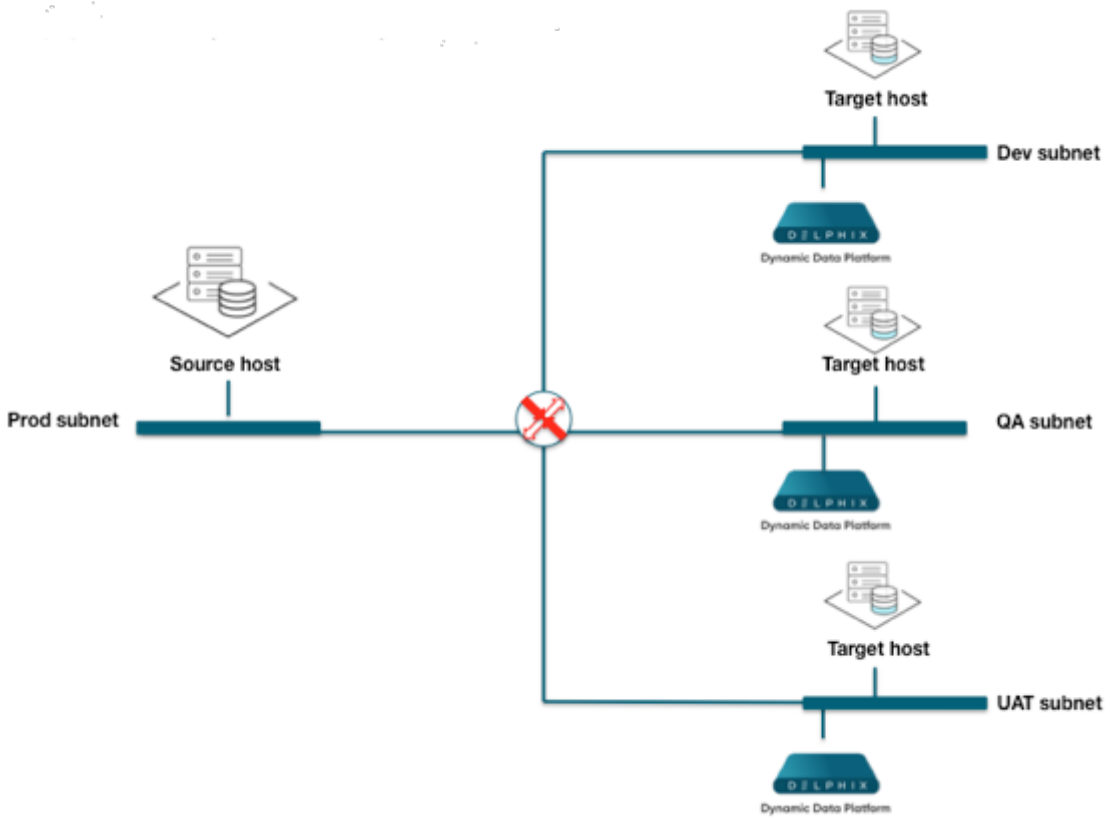
For optimal performance of VDBs, round-trip latency between the Delphix Engine and the target environment should be kept under 1 millisecond, and preferably in the range of 300 microseconds. If network latency exceeds 500 microseconds, the VDBs will not perform as well as a database connected to physical storage.

Latency can be introduced by having to route the network packets across multiple networks, or by the presence of routers, switches, and firewalls between the Delphix Engine and the target environment. Best practices to reduce network latency include:

- Keep the Delphix Engine on the same subnet as the target environment
- Reduce the number of hops between the Delphix Engine and the target environment
    - Reduce the number of switches in the network. Each switch can add 50 - 100 microseconds of latency to the network.
    - Reduce the number of routers in the network. Each router can add 500 - 1000 microseconds of latency in a network, and the round trip for an I/O operation could increase by as much as 1 - 2 milliseconds.
- There should be no firewalls between the Delphix Engine and the target environment.
- When linking the Delphix Engine to a source database across a WAN, consider the time needed for the initial link and load. It may be necessary to schedule the load operation as multiple steps across multiple days.



A common WAN deployment architecture



Deployment of the Delphix engine on Separate sub-nets

Network throughput and bandwidth

Network throughput measures the rate at which data can be sent continuously between two servers on a network. Network throughput is affected by network latency, but the dominant factor affecting throughput is the bandwidth of the network. As a point of comparison, consider the bandwidth available for three types of Ethernet networks:

Ethernet type	Network Bandwidth
100Mb Ethernet (100Base-T)	~=10MB/sec
Gigabit Ethernet (GbE)	~=100MB/sec
10 Gigabit Ethernet (10GbE)	~=1GB/sec

Low network throughput can impact the Delphix Engine in a number of ways:

- Increasing the amount of time it takes to perform a SnapSync operation, both for initial load and subsequent regular snapshots
- Managing LogSync operations in a high change environment
- Poor VDB performance when an application is performing large sequential I/O operations, such as sequential table scans for reporting or business intelligence, or RMAN backups of the VDB.

Delphix Engine throughput must exceed the sum of the peak I/O loads of all VDBs. Delphix incorporates an I/O-Collector toolkit to collect I/O data from each production source database and pre-production server.

Best practices to improve network throughput include:

- Use 10 Gigabit Ethernet (10GbE)
- Use a dedicated storage network

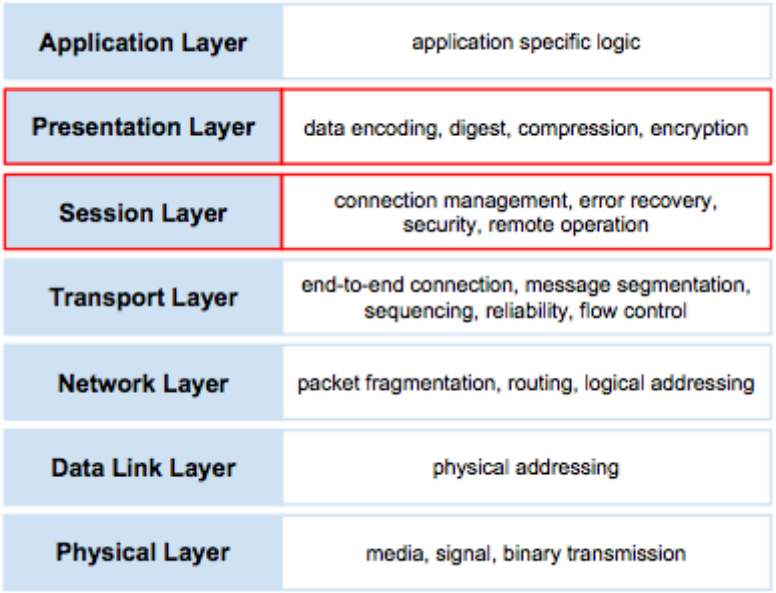
If you are concerned about your network throughput, you can test it with the built-in CLI tool for network testing.

Network operations using the Delphix session protocol

This topic describes how the Delphix Engine uses the Delphix Service Protocol (DSP) for network operations, and how this affects features such as replication, V2P, and SnapSync.

Overview

Delphix Session Protocol, or DSP, is a communication protocol that operates at the session and presentation layer in the Open Systems Interconnection (OSI) model.



DSP supports the request-reply pattern for communication between two networked peers. In contrast to the traditional remote procedure call (RPC) models, which focus exclusively on low-level details such as data encoding and wireframing, DSP implements a generic session layer that supports a number of advanced functionalities desired for network communication, including:

- Full-duplex remote operation execution and end-to-end cancellation support
- Advanced connectivity model with connection trunking and ordered delivery
- Fault resilience with automatic connection and session recovery, exactly-once semantics, and optional data digest
- High performance with concurrent execution, session flow control, optional data compression, and bandwidth throttling
- Built-in security support with pluggable SASL authentication mechanisms and optional TLS encryption
- Asynchronous model for session management and remote operation

Most of the features above are essential to the proper operation of a distributed application and yet non-trivial to implement. By offering them in the framework, we can significantly simplify the development of enterprise quality distributed applications.

DSP is officially registered with the Internet Assigned Numbers Authority under the service name dlpx-sp and port number 8415.

Currently, DSP supports Java language binding and provides a java based service framework for distributed applications.

Key concepts

The foundation of DSP is built on top of a few key abstractions, namely, **exchange**, **task**, **nexus**, and **service**. For an overview of how DSP works and the features it provides, let's start with these abstractions.

An **exchange** refers to an application-defined protocol data unit which may be a request or a response. DSP supports the request-response pattern for communication. For each request sent, there is a corresponding response that describes the result of the execution. An application protocol is made up of a set of exchanges.

A **nexus** (a.k.a., session) refers to a logical conduit between the client and server application. In contrast, a transport connection (a.k.a., connection) refers to a "physical" link. A nexus has a separate naming scheme from the connection, which allows it to be uniquely and persistently identified independently of the physical infrastructure. A nexus has a different lifecycle than the connection. It is first established over a leading connection. After it comes into existence, new connections may be added and existing ones removed. It must have at least one connection to remain operational but may live on even after all connections are lost. Nexus lifecycle management actions, such as create, recover, and destroy, are always initiated by the client with the server remaining passive.

A nexus has dual channels, namely, the fore channel and the backchannel. The fore channel is used for requests initiated from the client to the server, and the backchannel from the server to the client. From a request execution perspective, the nexus is full-duplex and the channels are functionally identical, modulo the operational parameters that may be negotiated independently for each channel. A channel supports a number of features for request processing, such as ordered delivery, concurrent execution, remote cancellation, exactly-once semantics, and throughput throttling.

A **service** refers to a contract that consists of all exchanges (both the requests and the corresponding responses) defined in an application protocol. Given the full-duplex nature of request execution in DSP, part of the service is fulfilled by the server and the remaining by the client, where the client and server are from the nexus management perspective.

A **task** implements a workflow that typically involves multiple requests executed in either or both directions over the nexus. A task is a self-contained building block, available in the form of a sharable module including both the protocol exchanges and implementation, that can be easily integrated into other application protocols. A library of tasks may significantly simplify distributed application development by making it more of an assembly experience.

The following is a diagram that illustrates the key abstractions and how they are related to each other.

Security

As a network protocol, DSP is designed with security in mind from the onset. It supports strong authentication as well as data encryption. It follows a session-based authentication model which requires each connection to authenticate before it is allowed to join the session. Authentication is performed using the Simple Authentication and Security Layer (SASL) framework, a standard-based pluggable security framework. The currently supported SASL mechanisms include DIGEST-MD5, PLAIN with TLS, CRAM, and ANONYMOUS. Optionally, TLS encryption may be negotiated between the client and the server for data privacy.

Performance

DSP offers a number of features to enable the support for high-performance network applications. For example, it allows multiple requests to be exchanged in both directions simultaneously, which provides effective pipelining of data transfer to minimize the impact of network latency while ensuring the total ordering at the same time. It supports trunking that can effectively aggregate the throughput across multiple connections, which is crucial for a long fat network (LFN) and 10GigE. It also provides optional compression support which boosts performance over a bandwidth-limited network. We have observed, through both internal benchmarking and in customer environments, DSP-based applications delivering multi GigE in an ideal environment and getting a performance boost of as much as x10 in bandwidth-limited settings.

Resiliency

DSP automatically recovers from transient connection loss without any application involvement. It may also detect random data corruption on the wire and automatically recovers from it. In both cases, outstanding requests are retried once the fault condition is resolved.

DSP offers control over a remotely executing request. Once a request is initiated, the application may cancel it at any time before completion. In the rare event of a session loss, a new session creation request will be held until the old session has been reinstated. It ensures that we never leave any unknown or unwanted activities on the remote side and provides better predictability and consistency guarantees over an otherwise unreliable network.

Diagnosability

Application exceptions encountered during remote execution of a request are communicated back to the initiator through DSP. A standard Java API is used to facilitate the handling of remote exceptions that is in many ways identical to local ones.

DSP provides detailed information and statistics at the session level. The information may be used to examine the state of the session as well as diagnose performance problems. It is currently exposed via an internal support tool called the JMX tool.

Supported applications

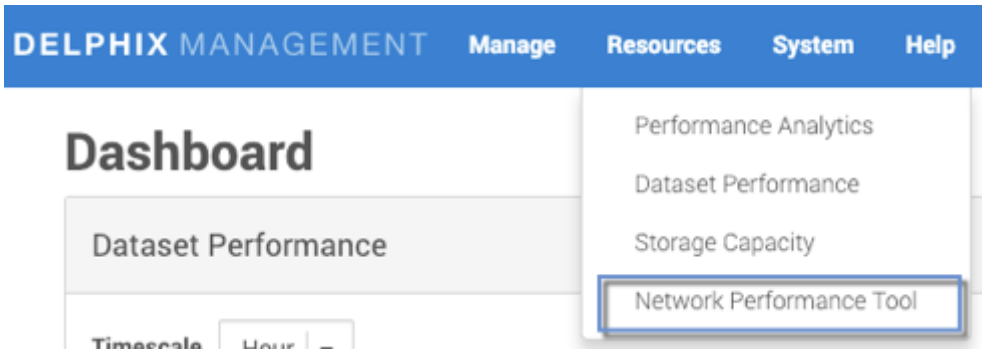
Replication is the first feature to take advantage of DSP. It has been rebuilt on top of DSP and shipping in the field since 3.1. In the latest release, a number of host-based applications, such as SnapSync, V2P, and Delphix connector, use DSP as well.

Network performance test tool interface

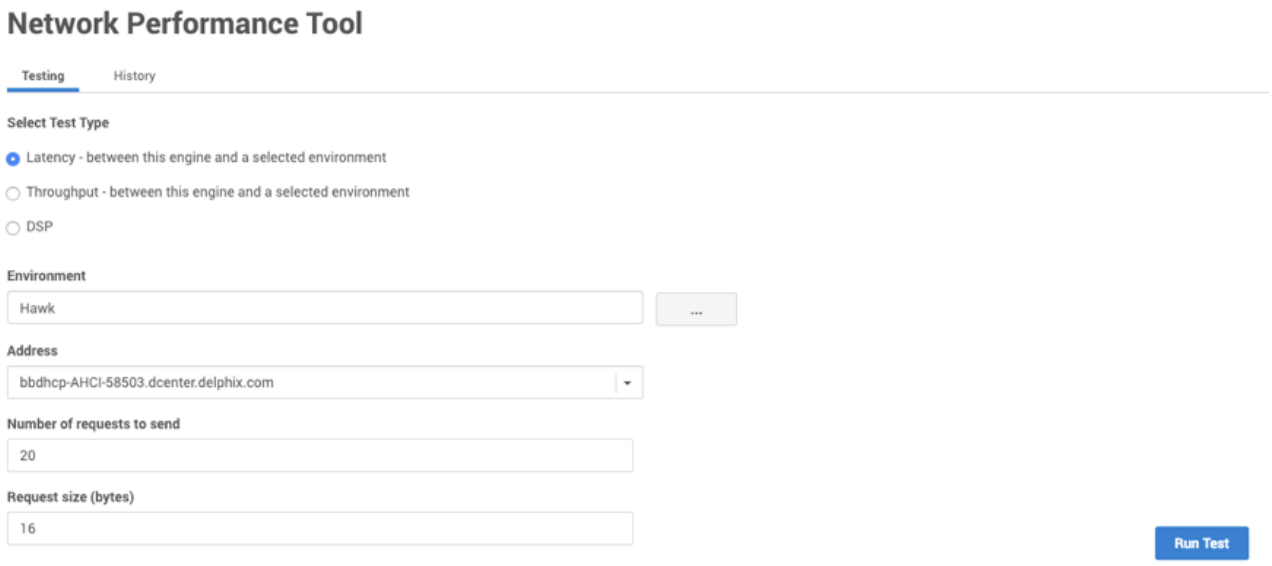
Accessing the network performance test tool

To access the Network Performance Test Tool:

1. In the top navigation bar, click **Resources**.
2. Select **Network Performance Tool**.



The **Network Performance Tool** page will appear. There are two tabs: **Testing** and **History**.



The testing tab

On the **Testing** tab, you can select one of three test types:

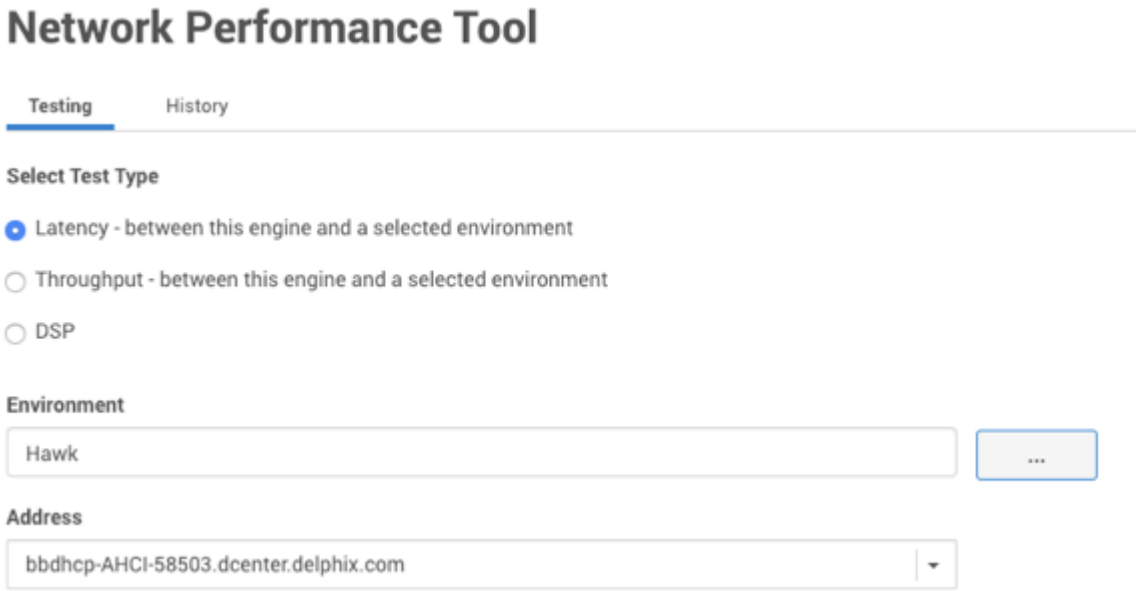
Test type	Parameters required
Latency	• Environment – If you selected an environment in the **Datasets** panel, this field will auto-populate. See below for instructions on changing the selected environment. • Number of requests to send • Request size (bytes)

Test type	Parameters required
Throughput	• Environment • Duration – in seconds • Number of connections • Direction – Select either **Transmit** or **Receive** • Block size (bytes)
Delphix session protocol	• Target Engine • Username • Password • Duration – in seconds • Block size (bytes) • Number of connections • Queue depth • Direction – Select either **Transmit** or **Receive** • Traffic Options

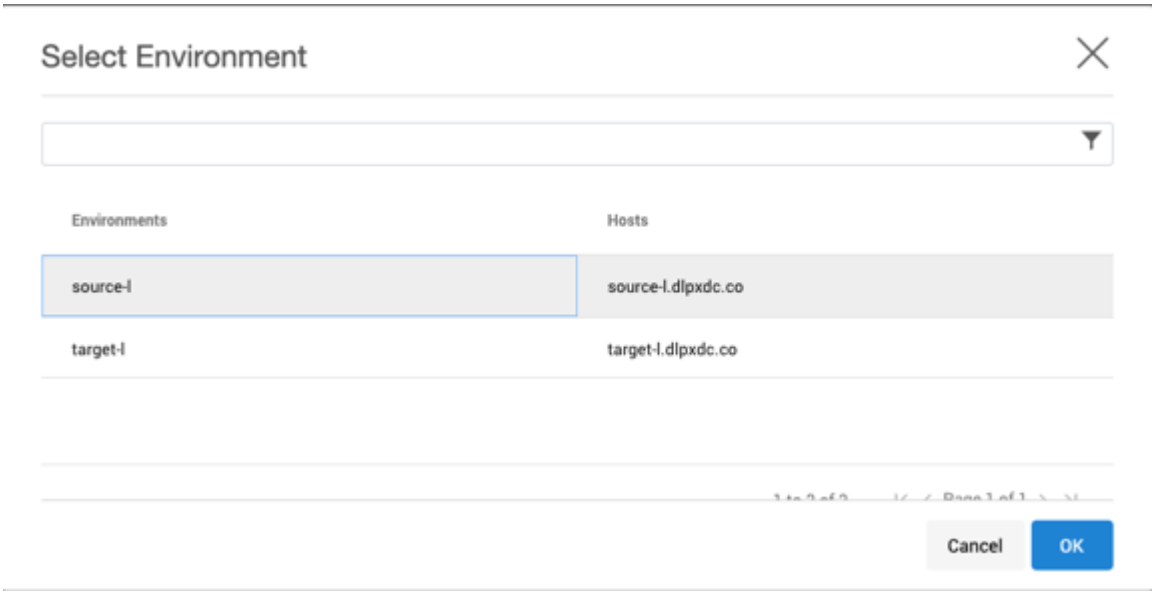
Selecting an environment

For either a Latency or Throughput test, you must select an environment. If you selected an environment in the **Datasets** panel, this field will auto-populate. If you need to select a different environment:

1. Next to the **Environment** field, click the button with three dots.
   The **Select Environment** screen will appear.



2. Select the environment you want to use in the test.



3. Click **OK**.

Running and canceling a network performance test

To run a test:

1. Enter all the required parameters.
2. Click **Run Test**.
3. To cancel the test click **Cancel**.

The history tab

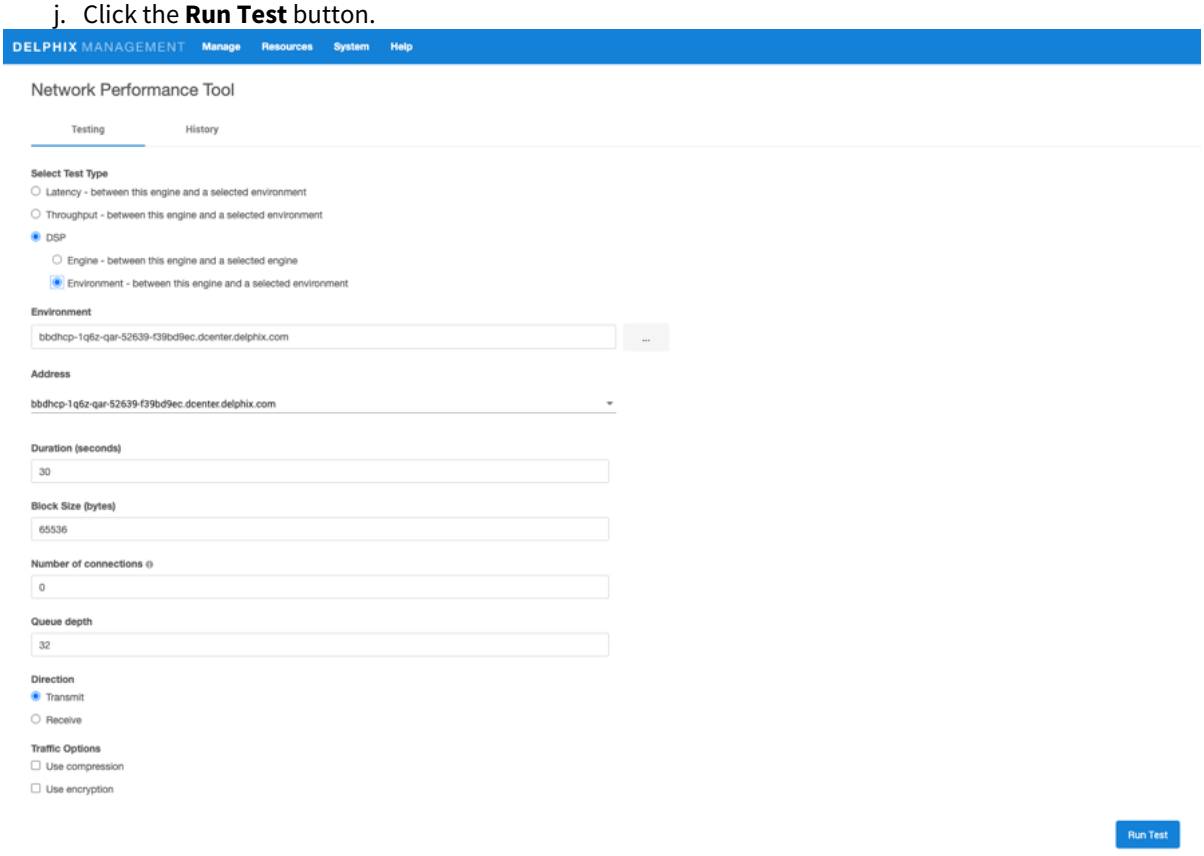
In the **History** tab, you can view the results of all previous tests you have run.

1. Select the radio button for the type of test for which you want to see the results:
    a. Latency, or
    b. Throughput, or
    c. Delphix Session Protocol
2. Optional: sort the tests by clicking one of the column headings.
3. Click the particular test for which you want to see the results.
   The details of that test will appear.
4. Click **OK** to return to the **History** tab.

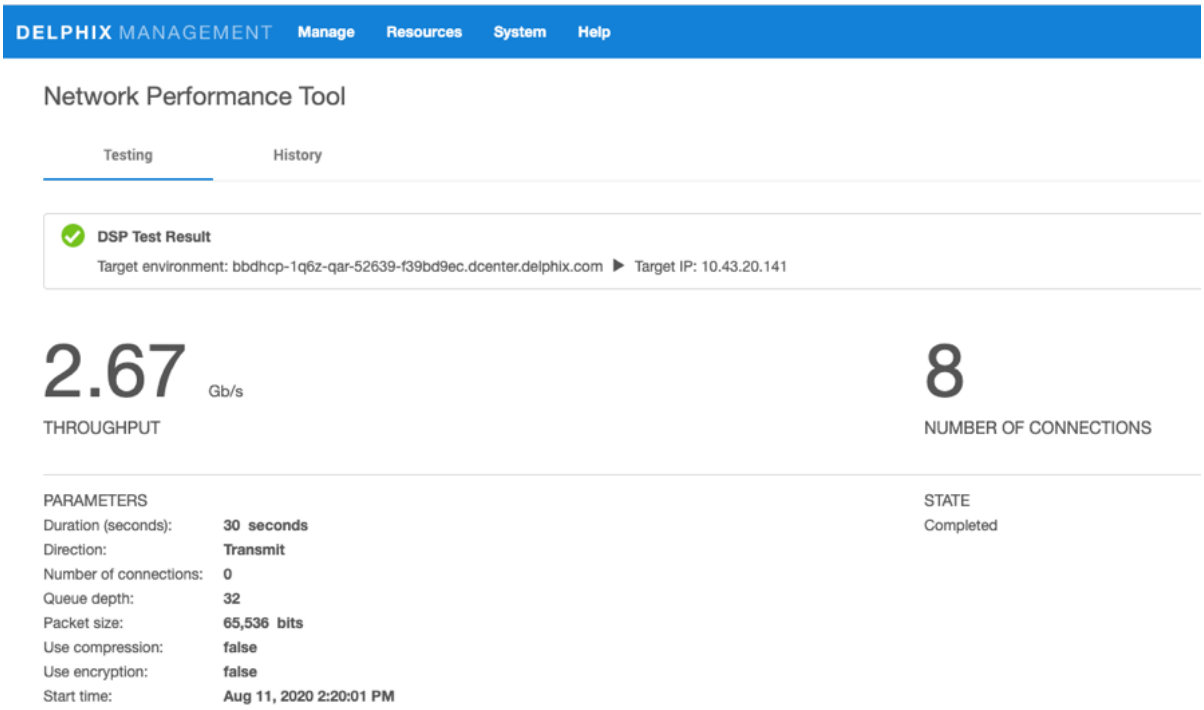
Delphix session protocol test from primary engine to replication engine

Delphix uses the DSP protocol to communicate between primary and replication engines.

1. Login to Delphix Management application using an Engine administrator account.
2. Click the **Resource** menu and select **Network Performance Tool.**
3. Select the **Delphix Session Protocol**option.
    a. Select Engine option to run the DSP test between a primary engine and a replication engine.
    b. Provide hostname or IP address of the replication engine.
    c. Provide admin user credentials for the replication engine. The user name must not include @DOMAIN or other qualifiers.
    d. Enter a **Duration** in seconds (30 default).
    e. Enter **Block Size** in bytes.
    f. Enter the **Number of Connections** (optional).
    g. Provide **Queue Depth**.
    h. Select Direction - **Transmit** or **Receive.**
    i. **Enter Traffic Options by selecting either Use compression or Use encryption.**
    j. Click the **Run Test** button.



4. View the test results.

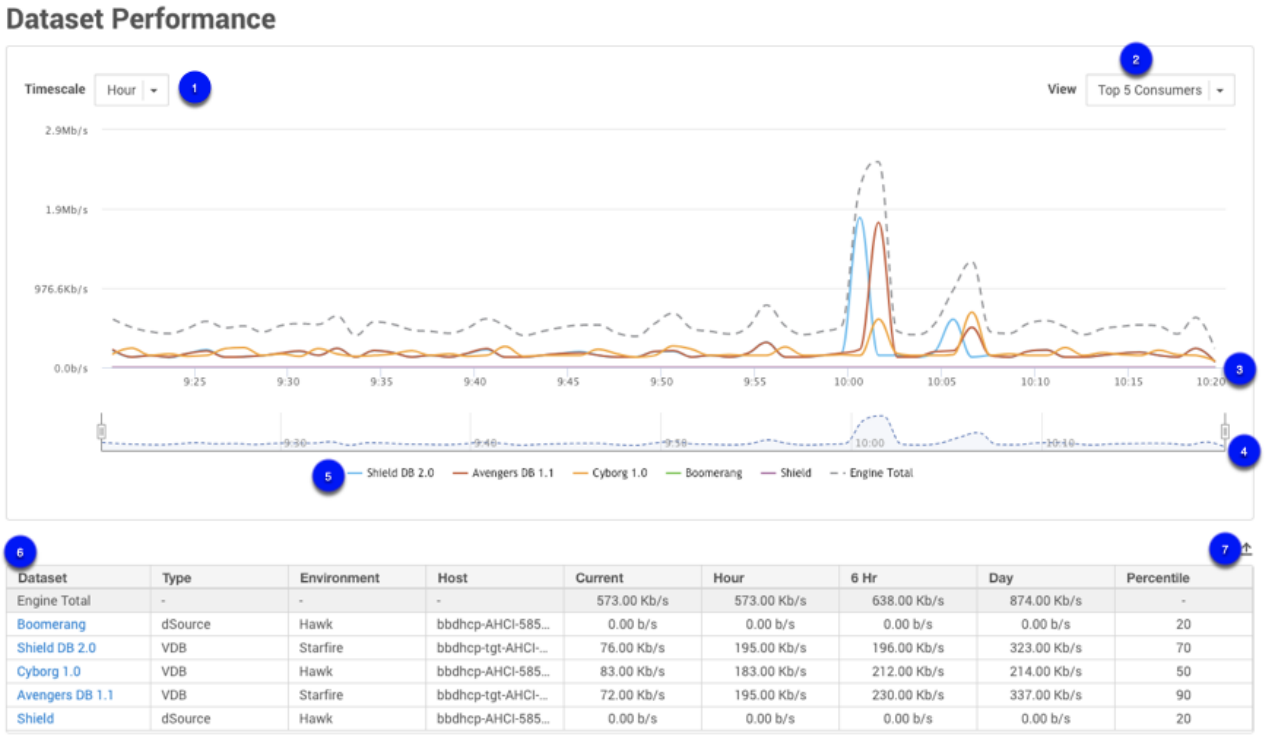

DELPHIX MANAGEMENT
Manage
Resources
System
Help
Network Performance Tool
Testing
History
DSP Test Result
Target environment: bbdhcp-1q6z-qar-52639-f39bd9ec.dcenter.delphix.com ▶ Target IP: 10.43.20.141
2.67
Gb/s
THROUGHPUT
8
NUMBER OF CONNECTIONS
PARAMETERS
Duration (seconds): 30 seconds
Direction: Transmit
Number of connections: 0
Queue depth: 32
Packet size: 65,536 bits
Use compression: false
Use encryption: false
Start time: Aug 11, 2020 2:20:01 PM
STATE
Completed

Working with dataset performance

Accessing the dataset performance graph

1. Log into the Delphix Management application.
2. In the **Resources** menu, select **Dataset Performance**.
3. Use the controls described below to view statistics and their related graphs.

General graph display and controls



Dataset	Type	Environment	Host	Current	Hour	6 Hr	Day	Percentile
Engine Total	-	-	-	573.00 Kb/s	573.00 Kb/s	638.00 Kb/s	874.00 Kb/s	-
Boomerang	dSource	Hawk	bbdhcp-AHCI-585...	0.00 b/s	0.00 b/s	0.00 b/s	0.00 b/s	20
Shield DB 2.0	VDB	Starfire	bbdhcp-tgt-AHCI-...	76.00 Kb/s	195.00 Kb/s	196.00 Kb/s	323.00 Kb/s	70
Cyborg 1.0	VDB	Hawk	bbdhcp-AHCI-585...	83.00 Kb/s	183.00 Kb/s	212.00 Kb/s	214.00 Kb/s	50
Avengers DB 1.1	VDB	Starfire	bbdhcp-tgt-AHCI-...	72.00 Kb/s	195.00 Kb/s	230.00 Kb/s	337.00 Kb/s	90
Shield	dSource	Hawk	bbdhcp-AHCI-585...	0.00 b/s	0.00 b/s	0.00 b/s	0.00 b/s	20

Timescale: Select from hour, 6 hours, day

View: View data by top 5 consumers or by the 75% percentile

Dateline: Displays timestamps of data points in the graph.

Timeline selector: Specifies the start and end time for the currently displayed data. The range displayed is controlled by selecting the slider. Drag the slider to view statistics for the specific time.

Graph legend: If more than one set of information is presented on the graph, Graph Legend displays a description and color for each set.

Dataset table: Displays the information for each Dataset in a table format. Selecting a Dataset link takes you to the Dataset Status page.

 **Export**: Exports the displayed table data.

Network performance expectations and troubleshooting

Overview

Once you have run your network performance tests to each source and target environment, you should confirm that they meet expectations. Corporate networks commonly leverage 10Gbps "line speeds" between servers. Although networks are shared environments, switching infrastructure is typically used to isolate traffic between different hosts, with the goal of allowing each host to reach its potential. While many environments perform within the 70-90% range of line speed, in well-maintained environments we can see 90%+ line speed.

In some circumstances, you can exceed typical expectations by implementing more best practice recommendations – for example, putting two VMs on the same hypervisor (e.g. ESX) host, in the same chassis or blade enclosure, or when teamed (bonded) NIC cards are used for extra bandwidth. Furthermore, when Jumbo Frames are implemented correctly, they provide a substantial improvement.

If your network is not meeting expectations, you will almost always need help from someone whose role is focused on networking to help arrive at a root cause. You may also need to obtain temporary access to other VMs and/or physical hosts to isolate some issue(s). Additionally, you may need the help of Delphix Support or Professional Services to perform some tests from the Delphix Engine to systems that are not already connected environments within the Delphix product. (The CLI test will only work for environments connected to the Delphix Engine).

With assistance or not, you can do a number of things to narrow down the potential causes of poor performance between two systems by a process of elimination. Below are some high-level steps to consider. Keep in mind that network throughput will always represent the least performant component, so many of the steps below are intended to help isolate which component may be performing poorly.

Troubleshooting and information gathering questions in rough order of priority - record and share your answers

1. Have all source/target tuning settings been applied?
    a. Is AIX in scope? LSO / LRO can have a significant impact, as can Jumbo Frames (see below)
2. What is the link speed on the hosts in question? Is NIC teaming / bonding / LACP in use?
    a. Linux: ethtool <device>
    b. Solaris: dladm show-phys
    c. Windows: wmic NIC where "NetEnabled='true'" get "Name,Speed"
3. What are the test results with greater or fewer connections in parallel?
4. Can we test throughput to alternate servers? (See below)
5. What is the overall latency? What is the latency to each (OSI layer 3) hop? Is there one hop that consistency has a higher cost? (check the latency with ping and hops with traceroute)
6. How many devices (OSI layer 2) are in the path? Your network team will need to help you identify these devices.
    a. Note: Only Layer 3 devices will show up when reviewing a traceroute however each layer two devices can impact traffic and each needs to be configured when implementing jumbo frames
    b. Example devices in path to physical server: 1. virtual NIC -> 2. Virtual switch (ESX) -> 3. Chassis NIC -> 4. Rack switch -> 5. Core Switch -> 6. Rack Switch -> 7. Physical NIC
    c. Example devices in path to virtual server: 1. virtual NIC -> 2. Virtual switch (ESX) -> 3. Chassis NIC -> 4. Rack switch -> 5. Chassis NIC -> 6. Virtual Switch (ESX) -> 7. Virtual NIC
7. What is the average network utilization on each hop? Is there congestion on any hop? (Network team will need to review)
8. Is QoS / VirtualConnect / 802.1p enabled? At what threshold will it engage? (Network team will need to review)
9. Is there a firewall or any deep packet inspection in the route? (Network team will need to review)
10. Are Jumbo Frames enabled on any or all hops? E.g. Delphix Engine, Virtual NIC, Virtual switch, and all hops down to the destination. (Network team will need to review)

a. We have seen Delphix installations often benefit 10-20% from Jumbo frames, but certain platforms (such as AIX) can benefit much more dramatically
b. Note that JF enablement on two hosts without confirming all the network pieces are properly enabled will result in VERY poor performance
c. Test Jumbo Frames via with “Do Not Fragment” flag from the remote host to the Delphix Engine.
d. Note that typical MTU Jumbo Frame setting is 9000 bytes, although some vendors recommend a little above or below this.
e. The test below is at 8000, but you can test larger from there. Our goal is primarily to ensure that a number substantially larger than 1500 and somewhat close to the 9000 "de facto" standard is working.
f. Whenever two hosts connect, they perform a handshake called Path MTU negotiation, where they agree on the highest MTU they both support. This is how we avoid impact when communicating to hosts with differing MTUs.
g. `Linux$ ping` -M `do` -s 8000 [Delphix_Engine_IP]
h. `Windows> ping -f -l 8000 [Delphix_Engine_IP]`
i. `Solaris v10-# traceroute -F [Delphix_Engine_IP] 8000 (` "Do Not Fragment" not supported by ping on Solaris until v11)
j. `Solaris v11+# ping -s -D [Delphix_Engine_IP] 8972`

11. Depending on the results above, a dedicated network or VLAN may help. Consider if that is an option for you. (Your network team will need to review)

Testing throughput testing to alternate servers

This will help isolate where a problem may be.

1. Delphix to Server A – already known
2. Delphix to Server B – physical; helps us see if there is a problem with the original server NIC or physical network settings
3. Server B to Server C – physical; helps us see if there is a problem with the Delphix server NIC or physical network settings
4. Delphix to Server D – virtual; helps us see if there is a problem with the virtual network or Delphix settings
5. Server D to Server E – virtual; same host; helps us see if there is a problem with the virtual network on the host
6. Server D to Server F – virtual; different host; helps us see if there is a problem with the virtual network

Conclusion

If you need further help, please contact Delphix Support or Professional Services to assist in getting the best performance possible from your environment.

Determining the Delphix server ID and host name

On occasion, it may be necessary to locate the **Delphix Server ID** and **Hostname**.

The Delphix Engine ID and Delphix Server ID are synonymous. The GUI currently uses "Server," and that is the terminology that will be used in this document.

The **Delphix Server ID** uniquely identifies each Delphix Engine. It is a 36-character hexadecimal string of the form **xxxxxxxx-xxxx-xxxx-xxxxxxxxxxxx**. You can view the Delphix Server ID in the Server Setup application, the Delphix Management application, or by using the Command Line Interface (CLI) method.

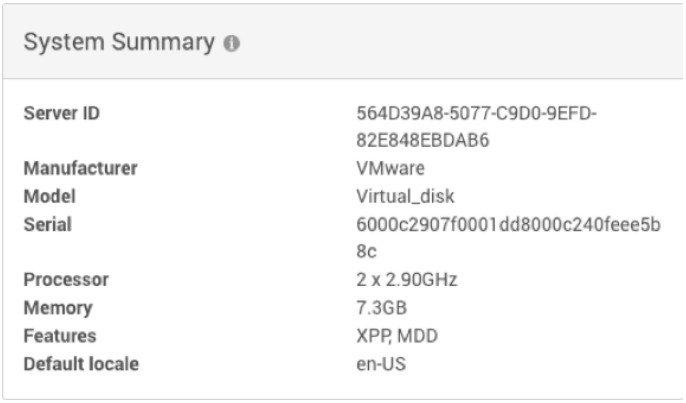
The **Hostname** is a name you assign. It typically matches the IP (DNS) name assigned to the Delphix Engine. The hostname can only be viewed by using the System Setup application or the CLI method.

Server setup application method

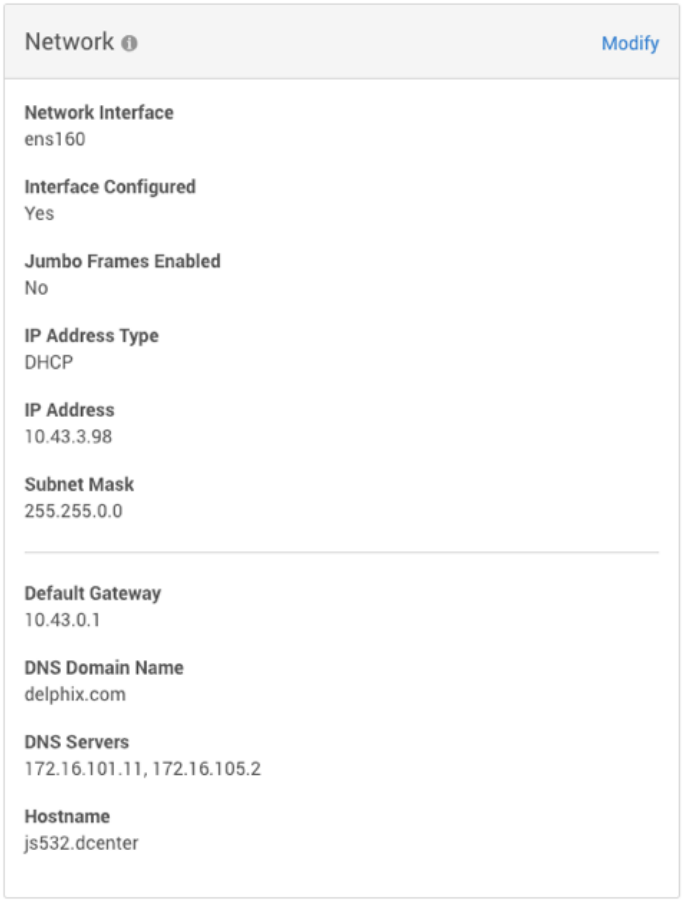
Login to the Delphix Setup application with sysadmin-level credentials:

1. Access the Delphix Engine through the URL: `http://<Delphix Engine>/ServerSetup.html` where `<DelphixEngine>` is the DNS name or IP address of the Delphix Engine for which you wish to find the Delphix Server ID and hostname.
2. Enter a valid **Username**.
3. Enter a valid **Password**.
4. Click **Log In**.

On the **Dashboard** screen, there is a **System Summary** panel in the lower-left portion of the screen**.** The **Server ID** field displays the Delphix Server ID, in this example **564D39A8-5077-C9D0-9EFD-82E848EBDAB6.**



The Delphix Engine Hostname is located on the same screen, in the **Network** panel. The **Delphix Engine Hostname** field displays the hostname information.



Delphix admin application

The Delphix Server Hostname is not available from this view but typically matches the IP (DNS) name assigned to the Delphix Engine.

Login to the Delphix Management application with delphix_admin level credentials:

1. Access the Delphix Engine through the URL: `http://<Delphix Engine>/Server.html` where `<DelphixEngine>` is the DNS name of the IP address of the Delphix Engine for which you wish to find the Delphix Server ID.
2. Enter a valid **Username**.
3. Enter a valid **Password**.
4. Click **Log In**.
5. Under **System Summary**, the **Server ID** field displays the Delphix Server ID.

CLI method

1. Use SSH to access your Delphix Engine: `ssh <userid>@<delphix_engine>` where < `userid>` is a user ID with either delphix_admin- or sysadmin-level credentials.
2. Enter a valid password if prompted for one.
3. Enter `system ls` .
   You will see an output similar to this example:

```
Properties
   type: SystemInfo
   apiVersion:
   type: APIVersion
      major: 1
      micro: 0
      minor: 5
   buildTimestamp: 2015-02-24T09:24:58.000Z
   buildTitle: Delphix Engine 4.2.0.1
   buildVersion:
      type: VersionInfo
      major: 4
      micro: 0
      minor: 2
      patch: 1
   configured: true
   currentLocale: en-US
   enabledFeatures: XPP
   hostname: delphix42.dcenter
   installationTime: 2015-02-24T19:53:32.000Z
   locales: en-US
   memorySize: 3.99GB
   platform: VMware with BIOS date 04/14/2014
   processors:
      0:
         type: CPUInfo
         cores: 1
         speed: 2.40GHz
      1:
         type: CPUInfo
         cores: 1
         speed: 2.40GHz
   productName: Delphix Engine
   productType: standard
   sshPublicKey: ssh-rsa
 AAAAB3NzaC1yc2EAAAABIwAAAQEAszNCFnfziuK8dBdv6DNB+LrhVP1wRWc/
vXVrxrDlgYQTrqvEx4BKgHDZ2hnbAmqq2xXHR5Ah6WDsEfo6u5B45JZc8qHpx8VZSZa053IdMK9LEg
```

```
oKPepmo7JV3kVY9oHK9PngLm9tFK+hN7AUHcGTt68IHq54GWYQNBtx0kgSR5HtkkFhVfX2amFsHIs
q1K96bgRkL0I5f3SjF4NnyElgBU9grGDajm9RXv+sz+Fn7h79AtFm0+W2Ymr5gQrdgh2vPyeFtG8G7
rxnQx3qiRBY6lNqepBhitXnMYSduGfW+fMJpV8TOOJ9ZLCfE7rjAgH7RxPybTfb4u70sm2krS8SgQ=
= root@delphix
    storageTotal: 23.07GB
    storageUsed: 2.00GB
    uuid: 564d2f7c-b84f-8bd1-6f45-2060ac9b9a65
```

The **Delphix server ID** is shown as the `uuid` property at the bottom of the output, and the **Hostname** is displayed in the `hostname` property.

4. Enter `exit` to leave the command-line interface.

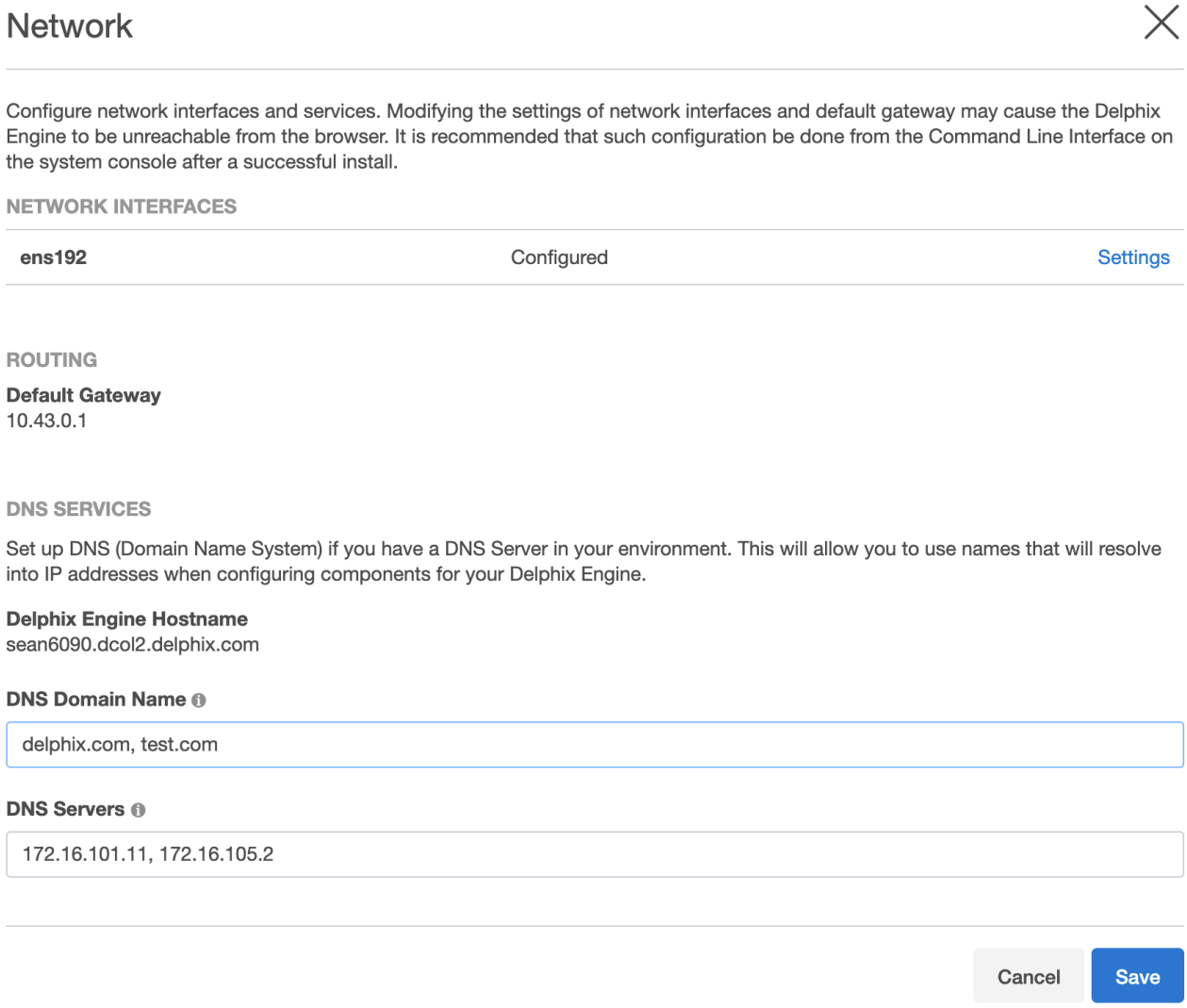
Configuring multiple DNS domain names in DNS search list

This topic describes the steps to configure multiple DNS domain names in the DNS search list.

Procedure

Perform the following steps to configure multiple DNS domain names in the DNS search list.

1. Launch the Delphix Engine Setup interface using the sysadmin credentials.
2. Navigate to the **Network** widget and click **Modify**.
3. Under **DNS SERVICES**, use the **DNS Domain Name** and **DNS Servers** boxes for adding multiple DNS domain name configurations.



Each domain name needs to be separated by a comma.

If .local DNS domains are in use, then you must add these explicitly to the list of DNS domains configured in order for name resolution to be successful. Multiple .local subdomains can be added as desired (for example, dev.company.local), or "local" can simply be added to the DNS domain configuration to enable all .local domains to be successfully looked up in DNS. Multicast DNS is not currently supported by the Delphix Engine

In order to understand if there is more than one domain name in the search list, check for "DNS Suffix Search List" from the output of `ipconfig /all` in the Windows Server:

or check for "`search`" in `/etc/resolv.conf` on a Linux server:

```
[root@rhel62 ~]# cat /etc/resolv.conf search delphix.com nameserver
192.168.0.1
```

To update DNS using the CLI:

1. Log into the CLI as sysadmin and navigate to **service > dns**.

   ```
   ssh sysadmin@yourengine
   > service
   > dns
   ```

2. List the current DNS configuration and **update** to add new configurations.

   ```
   > ls
   > update
   > set domain=xxx.xxx, xxx.xxx
   ```

3. **Commit** the action and verify the new list.

   ```
   > commit
   > ls
   ```

   For example:

   ```
   delphix> /service dns
   delphix service dns> ls

   Properties
   type: DNSConfig
   domain: delphix.com
   node: (unset)
   servers: 192.168.0.1

   Operations
   update
   ```

```
delphix service dns> update
delphix service dns update *> set
domain=delphix.com,one.delphix.com,two.delphix.com
delphix service dns update *> commit
delphix service dns>
delphix service dns> ls

Properties
type: DNSConfig
domain: delphix.com,one.delphix.com,two.delphix.com
node: (unset)
servers: 192.168.0.1

Operations
update

delphix service dns>
```

How to change the IP address of the Delphix engine

Changing the IP address

1. Stop all running VDBs by clicking the **Stop** button on the VDB card.
2. Disable all dSources.
3. You can use either the command-line interface or the Delphix Setup application to change the IP address of the Delphix Engine.
    a. To use the command-line interface, follow the instructions described in Setting Up Network Access to the Delphix Engine
    b. To use the Delphix Setup application, go to **System > Server Setup** in the Delphix Management interface, or click **Server Setup** in the Delphix Engine login screen.
        i. In the **Network** panel, click **Modify**.
        ii. Under **DNS Services**, enter the new IP address.
        iii. Click **OK**.
4. Refresh all Environments by clicking the **Refresh** Symbol on the Environments screen.
5. Enable all dSources.
6. Start all VDBs by clicking the **Start** button on the VDB card.

Changing the IP address via CLI

1. Stop all running VDBs by clicking the **Stop** button on the VDB card.
2. Disable all dSources.
3. Log into the Delphix CLI using your *sysadmin* account. You can find instructions on how to do this in the Connecting to the CLI article.

```
delphix> network
delphix network> setup
delphix network interface> list
NAME
vmxnet3s0
delphix network interface> select vmxnet3s0
delphix network interface 'vmxnet3s0'> get
    type: NetworkInterface
    name: vmxnet3s0
    addresses:
        0:
            type: InterfaceAddress
            address: 10.1.2.3/24
            addressType: STATIC
            enableSSH: true
            state: OK
    dataNode: DATA_NODE-34
    device: vmxnet3s0
    macAddress: 0:c:29:32:96:a3
    mtu: 1500
    mtuRange: 60-9000
    reference: NETWORK_INTERFACE-vmxnet3s0-DATA_NODE-34
    state: OK
```

4. Run the update command and update the address to the new IP address for the Delphix Engine.

```
delphix network interface 'vmxnet3s0'> update
delphix network interface 'vmxnet3s0' update *> edit addresses.0
delphix network interface 'vmxnet3s0' update addresses.0 *> get
Properties
    type: InterfaceAddress
    address: 172.16.151.154/24
    addressType: STATIC
    enableSSH: true

delphix network interface 'vmxnet3s0' update addresses.0 *> set address=10.1.2.
4/24
delphix network interface 'vmxnet3s0' update addresses.0 *> get
    type: InterfaceAddress (*)
    address: 10.1.2.4/24 (*)
    addressType: STATIC (*)
    enableSSH: true (*)
```

5. Commit the operation.

```
delphix network interface 'vmxnet3s0' update addresses.0 *> commit
delphix network interface 'vmxnet3s0'> get
    type: NetworkInterface
    name: vmxnet3s0
    addresses:
        0:
            type: InterfaceAddress
            address: 10.1.2.4/24
            addressType: STATIC
            enableSSH: true
            state: OK
    dataNode: DATA_NODE-34
    device: vmxnet3s0
    macAddress: 0:c:29:32:96:a3
    mtu: 1500
    mtuRange: 60-9000
    reference: NETWORK_INTERFACE-vmxnet3s0-DATA_NODE-34
    state: OK
```

6. Re-enable the VDBs and dSources running from the engine.

How to change the hostname of the Delphix engine

Changing the hostname

Perform the following steps to change the hostname of a Delphix engine.

> Currently, it is only possible to change the hostname via the Command Line Interface (CLI).

1. Stop all running VDBs by clicking the **Stop** button on the VDB card.
2. Disable all dSources.
3. Log into the Delphix CLI using your ***sysadmin*** username and password. If you are using the same interface that you connected to the CLI on, it will interrupt the connection. Therefore, it is recommended to log in to the CLI on the console.

```
ssh sysadmin@yourdelphixengine
```

4. Run the following commands.

```
dlpx01> network
dlpx01 network> setup
dlpx01 network setup> ls
Properties
    defaultRoute: 192.168.0.1
    dhcp: false
    dnsDomain: plb.internal
    dnsServers: 192.168.0.111,198.142.152.164,198.142.152.165
    hostname: dlpx01
    primaryAddress: 192.168.0.109/24

Operations
update
dlpx01 network setup> update
dlpx01 network setup update *> set hostname=newhostname
dlpx01 network setup update *> commit
```

5. If DHCP is being used, the Delphix Engine will expect the Hostname to be provided by the DHCP server. As such, there will be no property 'hostname' to update. This process requires changing the IP addressing configuration from DHCP to static address configuration. To proceed you will need to take the following action (making sure that there is no conflict between your DHCP server and the changes you are implementing).

```
dlpx01 network setup> update
dlpx01 network setup update *> set dhcp=false
dlpx01 network setup update *> set hostname=newhostname
dlpx01 network setup update *> commit
newhostname network setup> update
newhostname network setup update *> set dhcp=true
newhostname network setup update *> commit
```

```
newhostname network setup> ls
Properties
    defaultRoute: 192.168.0.1
    dhcp: true
    dnsDomain: plb.internal
    dnsServers: 192.168.0.111,198.142.152.164,198.142.152.165
    hostname: newhostname
    primaryAddress: 192.168.0.109/24
```

6. Re-enable the VDBs and dSources running from the engine.

How to change the DNS server of the Delphix engine

When DHCP is enabled on the Delphix Engine, the DNS server cannot be changed in the GUI. In this case, the DNS server can be changed in the command-line interface (CLI). If the IP address, net mask, or any other properties of the network interface itself are not changed, then VDBs do not need to shutdown. Login via Hypervisor Console for the Delphix Engine virtual machine is optional.

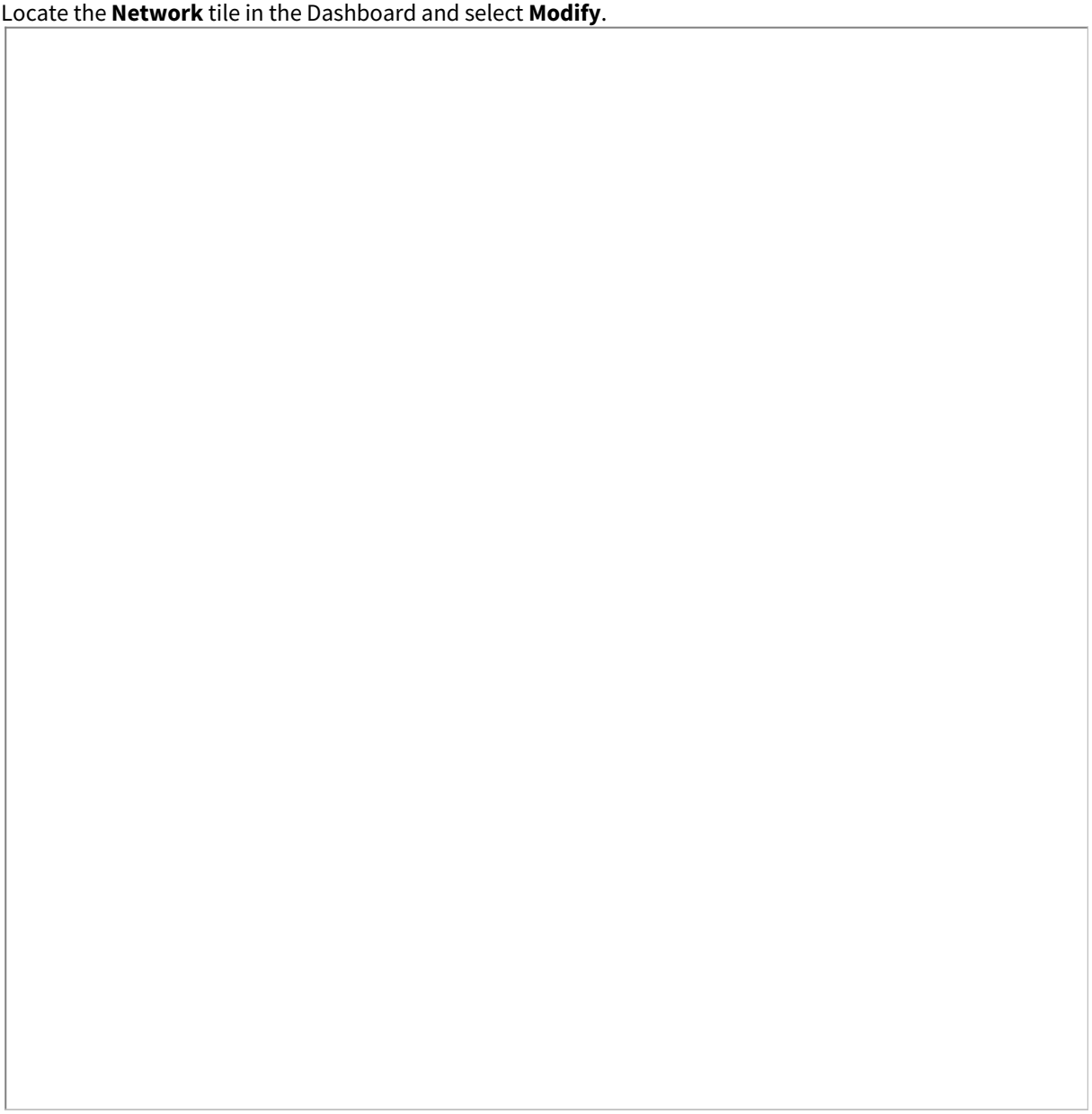
Use the commands below and enter the address of a server or a comma-separated list of addresses in replace of the example servers.

```
ssh -l sysadmin DelphixEngineName
Password:
DelphixEngineName> service dns
DelphixEngineName service dns> update
DelphixEngineName service dns update *> set servers=1.1.1.1,2.2.2.2
DelphixEngineName service dns update *> commit
DelphixEngineName service dns> exit
```

If DHCP is disabled and the address is static, the DNS server can be updated via the GUI.

1. Login to **Delphix Setup** with the appropriate credentials.

2. Locate the **Network** tile in the Dashboard and select **Modify**.



3. Scroll down in the **Network** window.
4. Use the **DNS Domain Name** and **DNS Servers** boxes for new configurations.

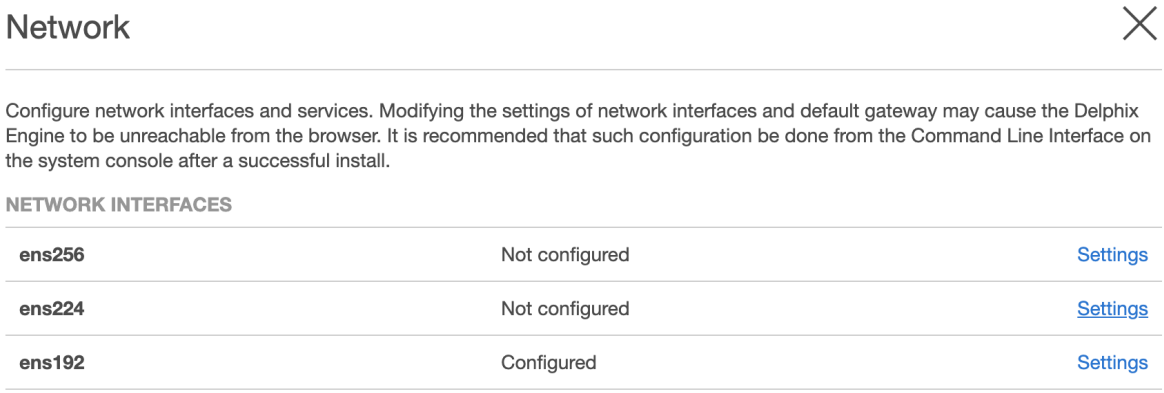
Configuring a second network interface

This topic describes how to configure a static IP address on a second network interface.

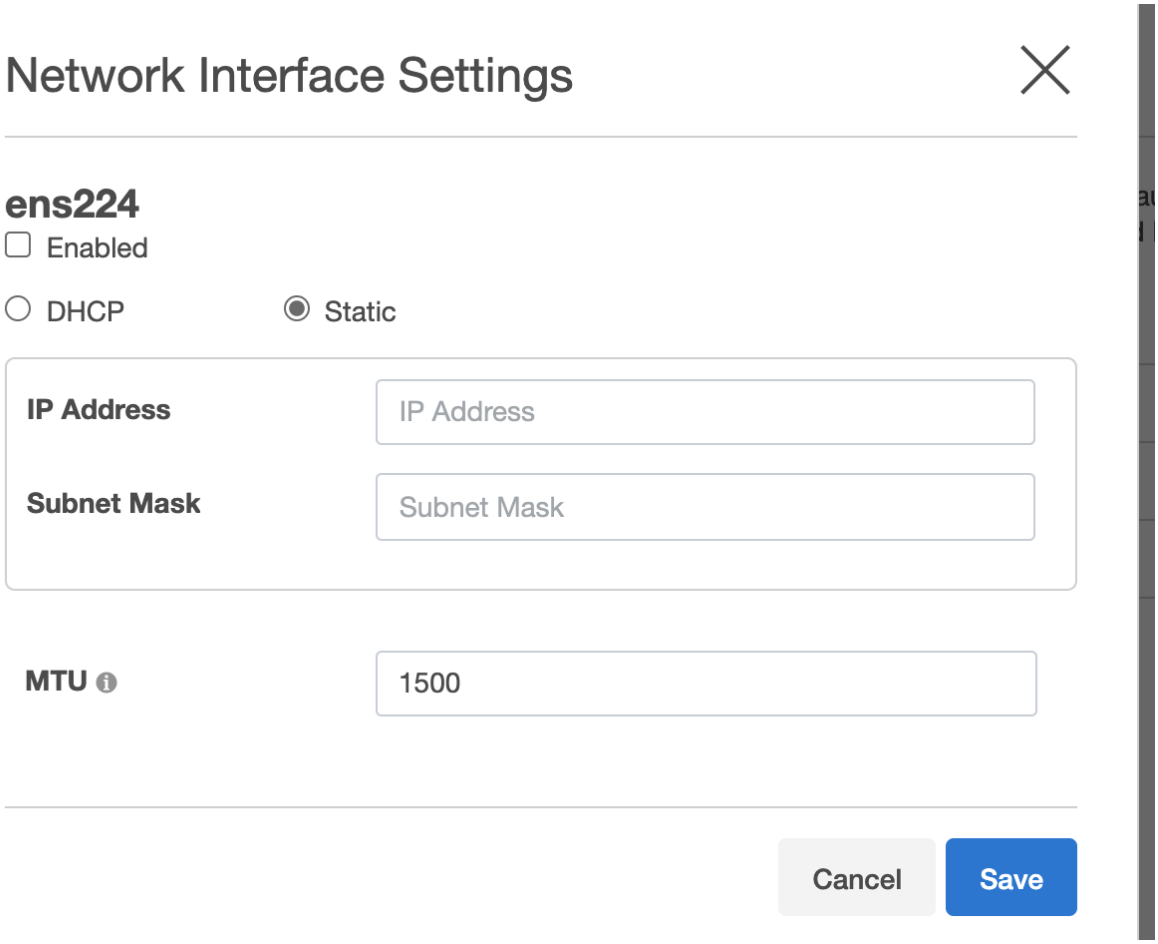
Procedure

Perform the following steps to configure a second network interface.

1. Launch the Delphix Engine Setup interface using the sysadmin credentials.
2. Navigate to the **Network** widget and click **Modify** to view the available network interfaces, and select the new interface to be configured.
3. Click the **Settings** button next to the network interface that you want to configure.



4. The **Network Interface Settings** screen appears. Select the checkbox before **Enabled** to enable the network.



5. Select one of the following: **DHCP** or **Static** andenter the **IP address** and **Subnet Mask** address in the respective fields.
6. **MTU**: Enter a value for the MTU field. This is the maximum size in bytes of a packet that can be transmitted on this interface.
7. Click **Save** to save the settings.

NFSv4 configuration

Overview

This article shows which OS versions support NFSv4. NFSv4 is enabled by default, but the target/staging hosts require additional configuration changes before it can be used.

Redhat/SLES: "NFSv4 Only - Enabling Recover Lost Locks" section under Linux/Redhat/CentOs

AIX: IBM AIX article.

OS version	NSFv4 support
RHEL 6.4 or later	Supported
SLES 11.4 or later	Supported
AIX 7.1, 7.2	Supported
Solaris 11	Supported

If these target host configurations are not set, the engine will choose NFSv3. The provided reason (e.g., "OS not supported") that is shown for choosing NFSv3 over NFSv4 can be observed under the Status tab of the selected VDB in the Delphix Engine GUI. Additional information is provided in this tab for datasets and current values.

Dataset status tab NFS reasons

NFS Reason	Notes
Default	NFSv4 is used by default for client mounts.
Old RedHat	NFSv4 is used with RedHat versions 6.4 and newer.
Unsupported OS	NFSv4 is not supported for HP-UX clients.
DNFS	Oracle Direct NFS (dNFS) is in use by the database. NFSv4 is supported by dNFS with Oracle versions 12c and newer. Additionally, due to Oracle bug 33596056, dNFS does not work with NFSv4 in Oracle versions 19.12-19.15 and 21.1-21.6.
Tunable Override	The nfs.version tunable is set to force NFSv3 mounts.
Configuration Override	Appears with default operation, implies that a configuration parameter has changed.

NFS Reason	Notes
No Recover Lost Locks	On Linux hosts, if the required recover_lost_locks setting is not enabled for the NFS client, then NFSv4 cannot be used.
Incomplete v4 Config	`On AIX hosts, if the nfsrgyd` daemon is not running or reverse DNS lookup is missing, then NFSv4 cannot be used.

NFSv4-Only mode

A Delphix Engine can be configured only to use NFSv4 and disable NFSv3 services.

Behavioral Differences Compared Automatic (default):

- If NFSv4 is not supported on the mount, then the Delphix Engine will fail to mount the VDBs and return a user exception with the reason "NFSv4 is not supported".
- If dNFS is enabled, the Delphix Engine will attempt to use NFSv4 in Oracle versions 19.12-19.15 and 21.1-21.6 despite Oracle bug 33596056. Additional details are in this KB.

If NFSv3 is enabled, it will be turned off when the option is set:

- If existing VDBs on a continuous data engine are using NFSv3, they will go down since the mounts will no longer serve data once the option is set.
- If the continuous compliance engine is using NFSv3 via remote mounts, they can no longer access data once the option is set.

Capacity and resource management

Delphix will be responsible for managing many different data sources and data types. As such, it is critical to understand how to manage your capacity and storage resources within each Delphix engine. Learn storage and quota best practices, as well as the different options to optimally manage capacity.

This section covers the following topics:

An overview of capacity and performance information

This topic describes the Delphix Engine performance reservoir and capacity threshold warnings and various ways to obtain information about capacity and resource usage for the Delphix Server.

The performance reservoir and capacity threshold warnings

In order to obtain the best performance and continued operations, the Delphix Engine requires a certain amount (minimum 512GB) of free space of the total quota for storage space. As storage capacity approaches this threshold, the following system faults occur:

- When **85%** of the total storage quota is reached or **1536GB** of free space is remaining (whichever is less), a **Warning** fault is triggered. You can resolve this fault by deleting objects in the Delphix Engine, adding storage, or changing policies, as described in the topics Adding, expanding, and removing storage devices, Deleting objects to increase capacity. Additionally, refreshing target databases will clear the space the engine uses to track changes over time for each DB.
  Note:
  When a **Warning** fault is first raised, it will not impact the functioning of the system. However, there may be an impact to Oracle LogSync and SQL Server Validated sync when a **Warning** fault is seen as storage usage decreases following a **Critical** storage usage fault.
- When **90%** of the total storage quota is reached or **1024GB** of free space is remaining (whichever is less), a **Critical** fault is triggered. You cannot ignore or resolve this fault. This fault will have a significant impact on the system's behavior such as:
  - All pending link, sync, refresh, and provisioning processes will be canceled, and no new operations can be initiated
  - Scheduled replication processes will first check for capacity on the target engine and hold data (replication currently in progress to the engine will not be halted)
  - Policy operations such as SnapSync, snapshot, and refresh are suspended for **all** databases
  - dSources stop pulling in new changes. LogSync is suspended for all **Oracle dSources**. Validated sync is disabled for **SQL Server** dSources.
  - No virtual database (VDB) snapshots can be taken.
- When **95%** of the total storage quota is reached or **512GB** of free space is remaining (whichever is less), a second **Critical** fault is triggered. This fault will have a significant impact on the system's behavior and certain dSources and VDBs will stop in order to maintain data integrity. For example, SQL Server VDBs will shut down.

As the free space of the system improves, the following functionalities can be automatically or manually resumed.

- When the system falls below **95%** of the total storage quota, you can manually start SQL Server VDBs that had stopped
- When the system falls below **90%** of the total storage quota:
  - SQL Server VDBs that had stopped will automatically start
  - New link, sync, refresh and provisioning operations are allowed
  - Policy operations such as SnapSync, Snapshot, and Refresh resume for all databases
- When the system falls below **85%** of the total storage quota
  - dSources start pulling in new changes from their corresponding data sources. LogSync is resumed for **Oracle dSources**. Validated sync is enabled for **SQL Server** dSources.

For more information, see Setting quotas

Ways to view capacity usage

You can access capacity and performance information for the Delphix Engine through several different means, including the **Dashboard** view, and the **Storage Capacity** screen.

The dashboard view

You can access the **Dashboard** view in the **Delphix Management** application by clicking **Dashboard** in the Manage menu. Note that the Dashboard view provides only summary information about capacity and performance. You must access the **Storage Capacity** and **Dataset Performance** screens in the **Resources** menu to manage storage space and database objects.

The Dashboard view provides more detailed information about the overall performance of the Delphix Engine:

- **Storage Capacity** - the amount of physical storage available and what is currently used
- **TimeFlow Ratio** - see above
- **VDB Ratio** - a measure of the amount of physical space that would be occupied by the database content against the amount of storage occupied by that same database content as VDBs.
- **Dataset Performance** - the amount of network bandwidth available and the amount that VDBs are currently utilizing, as well as information about specific VDB network usage

The storage capacity screen

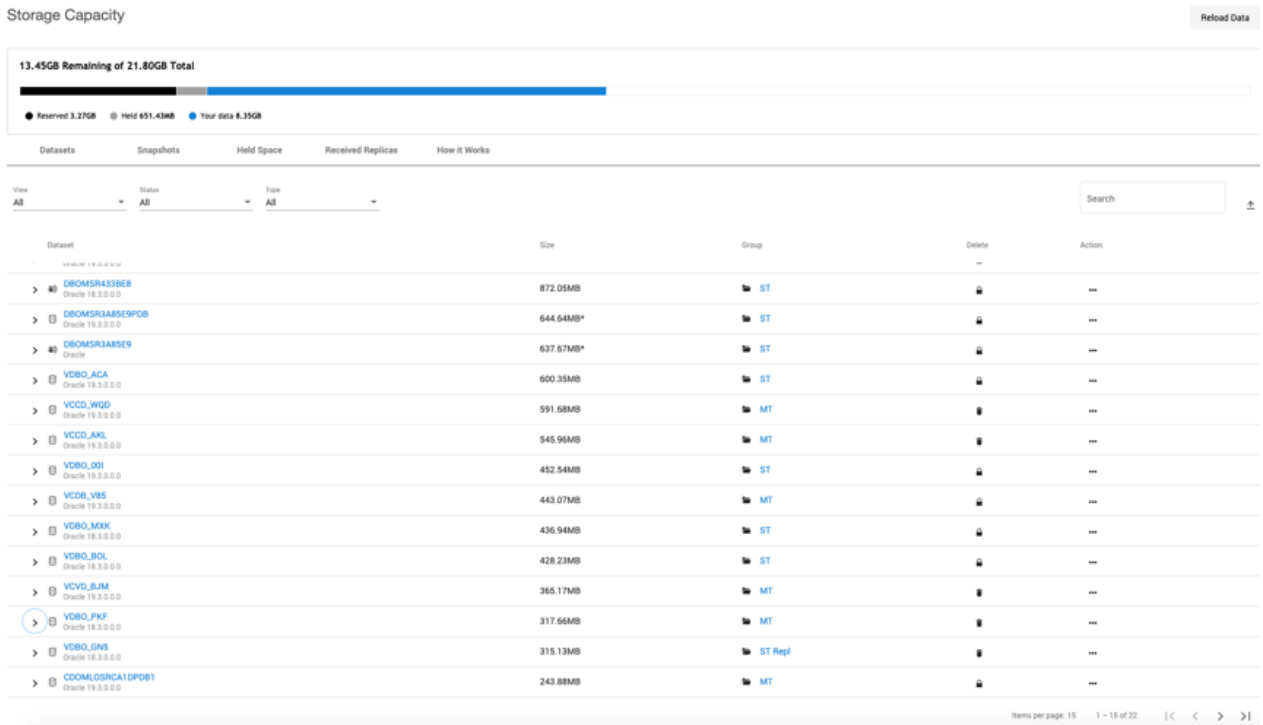
You can access the **Storage Capacity** screen through the **Resources** menu in the **Delphix Management** application. The **Storage Capacity** screen now has an improved format the interface makes it easier to identify which datasets are consuming the most space on your Delphix Engine.

Major additions in 6.0.6:

- Users can now see what's pinning a held space and batch delete the dependencies to free it up.

What's changed 6.0.6:

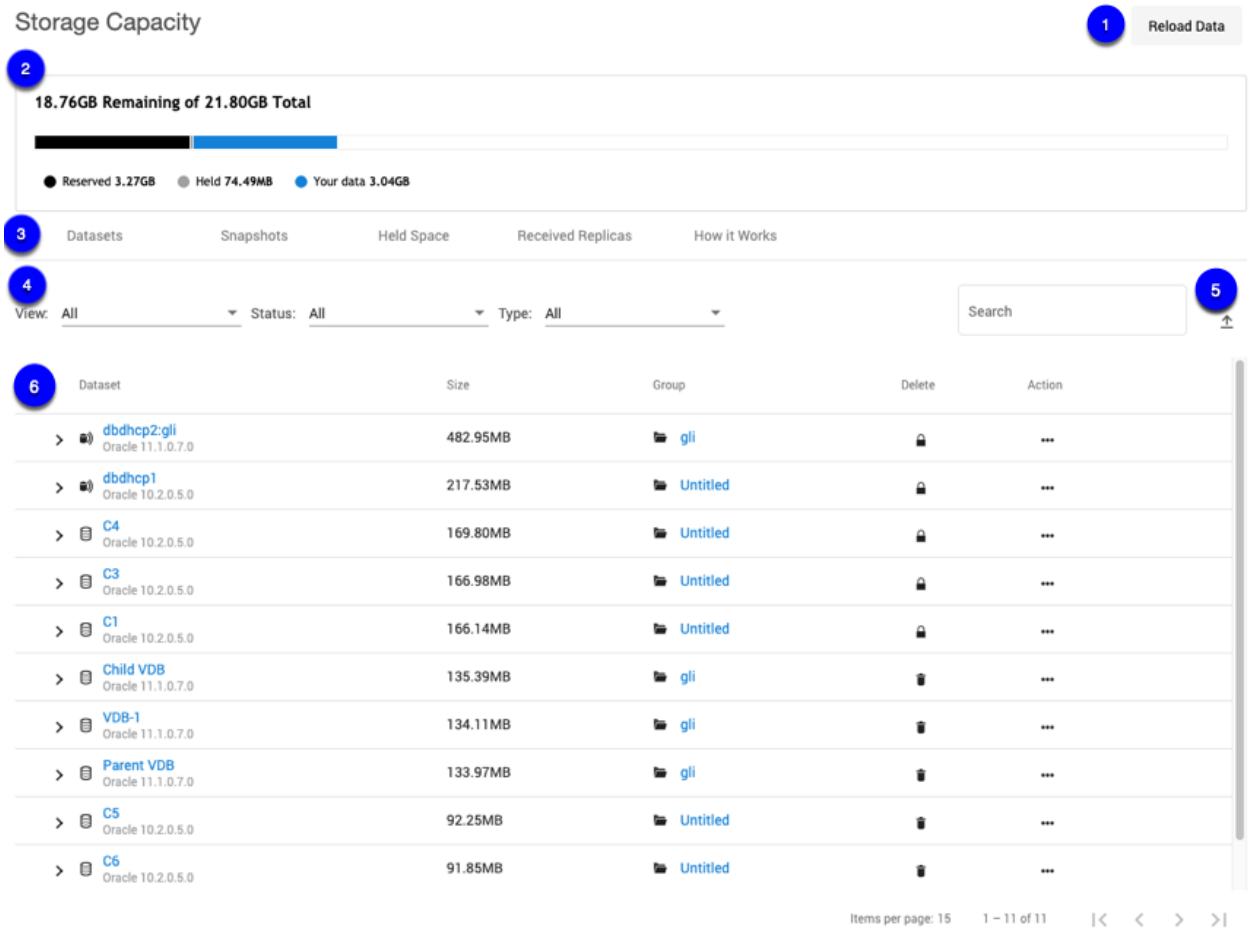
- Snapshot Delete dialog now shows a tree of "prerequisites" and "dependencies" that must have operations performed on them / be deleted in order to enable the deletion of the locked root snapshot.
- The Snapshot Delete dialog now automates the steps.
- Held Space shown in the table now represents deadbeat Timeflows rather than Containers



For more information please refer to Using and understanding the storage capacity screen

Using and understanding the storage capacity screen

The **Storage Capacity** screen can be reached via the **Capacity** item in the **Resources** menu. This screen shows how storage capacity is allocated for dSources, VDBs, and Snapshots, and permits storage space to be reclaimed through the deletion of objects and Snapshots.



Storage Capacity Screen

☐ Select the Reload Data button to clear your capacity cache and refresh the page. If you are clearing the cache of a large Delphix Engine, this operation could take several minutes. During this operation, the capacity page is not available.

☐ The usage Graph at the top of the page provides a visual representation of the distribution of storage consumption across your engine.

- **Reserved** - The amount of reserved space is hard-coded to be 10% (maximum up to 1TB) of the system capacity and cannot be changed. The system has a 90% consumption cap.
- **Held** - Total held space size
- **Your data** - User data (snapshot, log, etc.); Includes shared space

⚠
- Currently, this only includes dataset space. It doesn't include space of the MDS.

☐ Click a tab to view one of the following tables:

- **Datasets -** this tab shows a list of datasets on the engine ordered by the size of the dataset. You can filter the list by whether the dataset is VDB or dSource, whether the dataset is locked or unlocked, and the type of dataset. The drop-down drawer of each dataset shows some metadata about the dataset. If the dataset has descendants, the drop-down drawer will also show the list of **first-level** descendants. You can use the Action menu (...) to delete or refresh a dataset. If a dataset has descendants, the delete column will show a lock icon which opens a dialog explaining why this dataset is locked. Replicated datasets are excluded.
- **Snapshots -** rows are always sorted by descending size with a link to the dataset page for the Snapshot's parent dataset. Snapshots created from Replicas are excluded. The drop-down drawer for each snapshot shows the latest snapshot date of the dataset that this snapshot is from. If a snapshot has any descendants or can't be deleted due to descendants, the delete column will show a locked icon indicating this snapshot is locked. Clicking on the lock icon will display the snapshot why-locked dialog. Freshly created Snapshots are not displayed as this table is not updated automatically.
- **Held space -** provides a list of Held Space on the engine. You can search the held spaces by their names (in this case their references). As shown below users can now see what's pinning a held space and batch delete the dependencies to free it up.
  
- **Received replicas -** provides a list of Namespaces and by default is sorted by descending size. This tab is updated automatically (except the size). The drop-down drawer lists groups inside of each replicas sorted by size.
- **How it works** - provides general information about Storage, Retention Policy and Manual deletion.

 Filters available are table-specific and do not apply to the content inside a drawer

- **View** - Dataset type (VDB, dSource, vFiles, etc.)
- **Status** - locked/unlocked for deletion
- **Type** - database type (Oracle, SQL Server, AppData, etc.)
- **Search** - search by name (case insensitive)

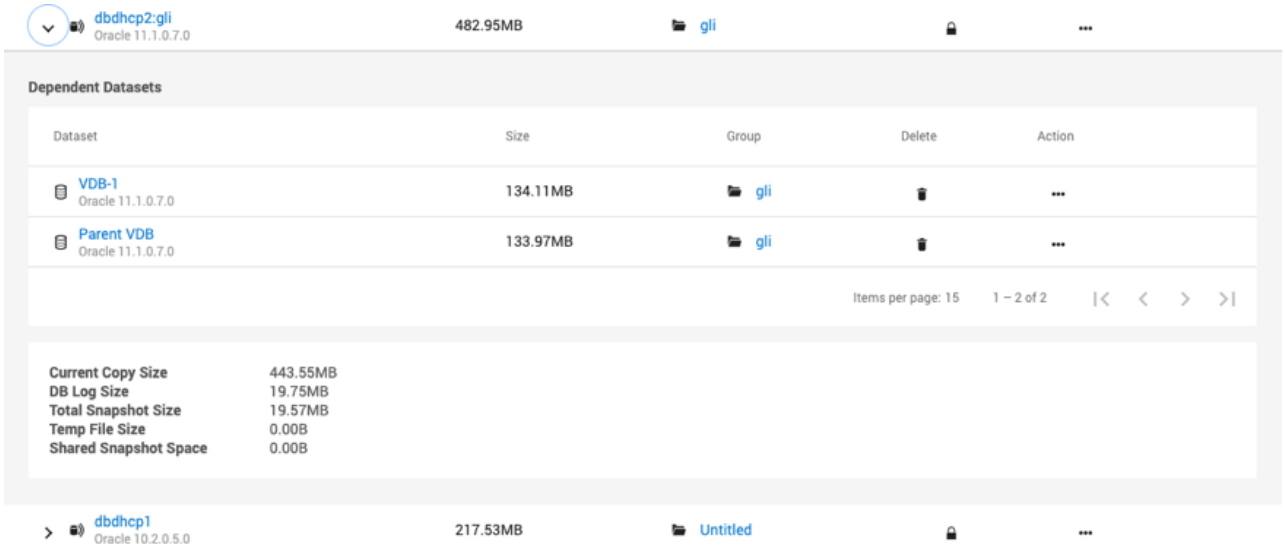
 Select  to export the information provided in the grid to a.csv file. Note: Only the page you are viewing will be exported.

 The **table** section displays a set of information about objects tracked by the Delphix Engine. The information displayed varies depending on the selected tab.

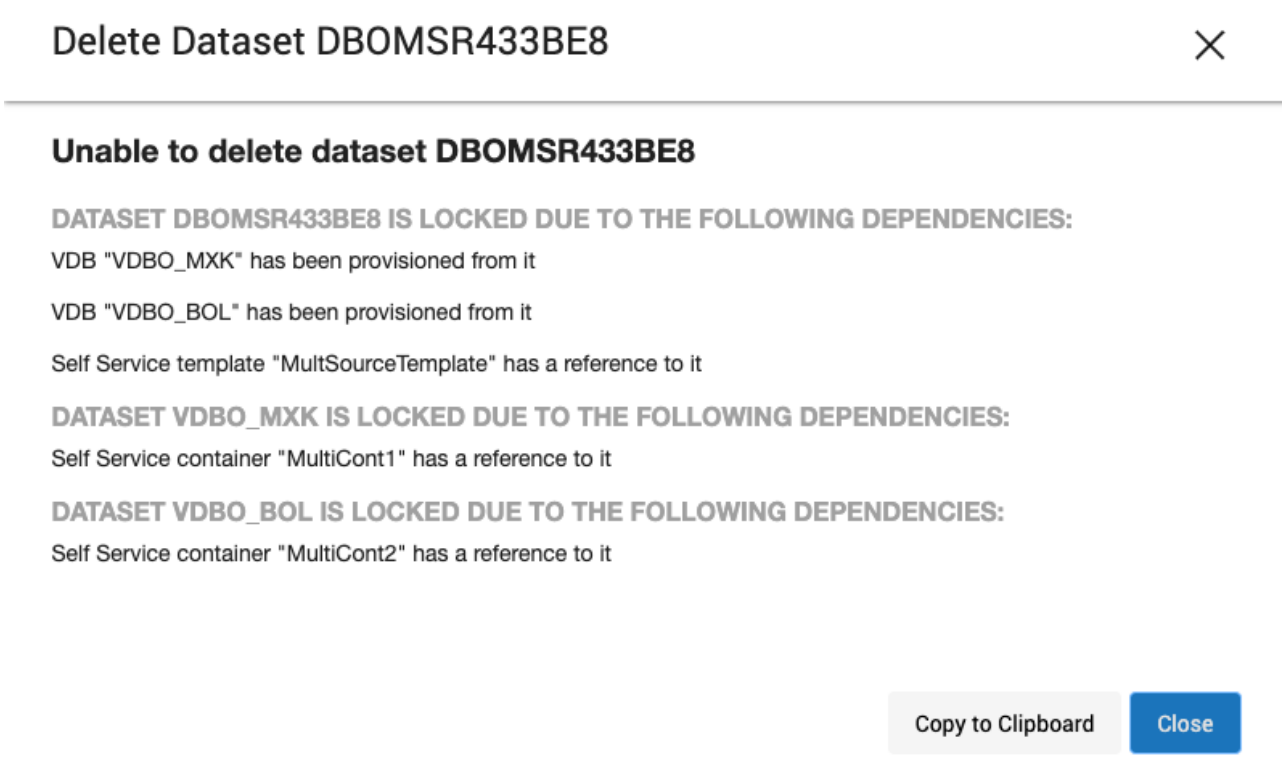
Datasets	
Dataset	Dataset's type as an icon and name. Datasets created from replicas are excluded and deleted Datasets have a strike-through their name.
Size	Space actually used by the object.
Group	The name of the Dataset Group.
Delete	Either a Trashcan or a locked icon will be displayed. The Trashcan opens a confirm delete dialog, for Oracle users can select to provide credentials. A Lock icon as shown below opens a dialog explaining why the selected object cannot be deleted.

Action	Refresh actions that can be taken against the Dataset. • Refresh N/A • Refresh Dataset
Snapshots	
Timestamp	The timestamp reflecting when the snapshot was taken.
Size	Space actually used by the snapshot.
Dataset	The name of the Dataset.
Group	The name of the Dataset Group.
Delete	Either a Trashcan or a locked icon will be displayed. The Trashcan opens a confirm delete dialog. A Lock icon as shown below opens a dialog explaining why the selected object cannot be deleted.
Held space	
Object	The name of the object.
Size	Space actually used by the object.
Delete	For Held Space every row has a lock icon for opening up the Why-locked dialog.
Received replicas	
Namespace	Once replication is complete, the target Engine will create a received replica, also known as a namespace.
Size	The amount of space used by the Namespace.

 Dataset Table or Snapshot drawer provides information on dependent or descendant. The inner dependency table is direct children only and has the same behavior as the outer table. Filter or sorting from the outer table does not apply to the inner table.



The **Delete dataset dialog** - this dialog provides information on why a dataset cannot be deleted. It displays **all** descendants of a dataset, as well as any self-service objects on any of the descendant datasets. You can trace the chain of descendants from the current dataset all the way to leaf datasets. Click **Copy to Clipboard** to copy the list of descendants to clipboard.



The **Delete snapshot dialog** - The new snapshot dialog provides the complete steps on how to unlock the locked snapshot for deletion. It also allows users to select descendants and batch delete them. The dialog shows a tree of **prerequisites** and **dependencies** that must have an operation performed or be deleted in order to enable the deletion of the locked root snapshot. **Prerequisites** refer to operations that must be performed before the object it's associated with can be deleted. Examples of prerequisite operations include: removing a keep-forever retention policy from a snapshot, deleting a Self-Service bookmark, or refreshing a dataset. **Dependencies** refer to objects that must be deleted before the parent object can be deleted. There is also a special **All (other) Snapshots**

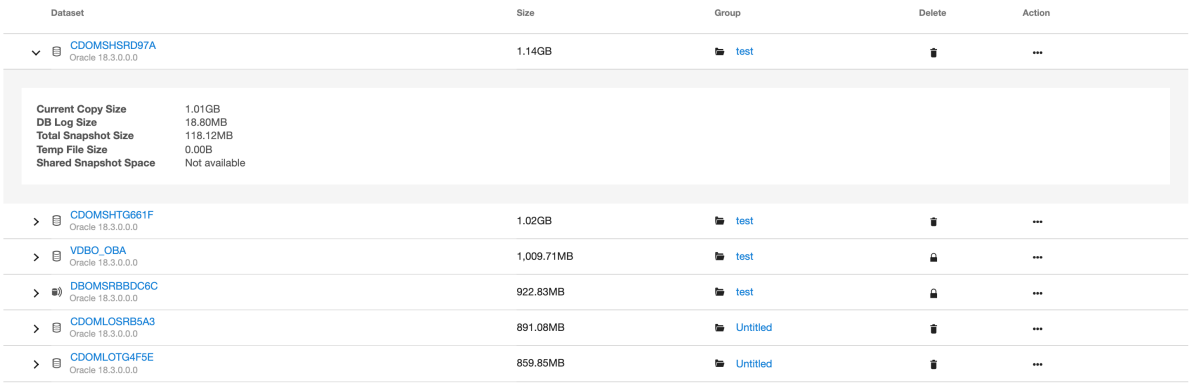
dependency item. This item represents all children snapshots of the parent Timeflow, which will also need to be removed, minus those with a dedicated row in the table.

Understanding Delphix disk usage

Within the Delphix Timeflow, incremental restore points are called Snapshots. Delphix Snapshots use a shared block architecture to provide all the functionality of daily full backups using only a tiny percentage of the storage for full backups. Because shared block architectures are not common, a "common sense" understanding of how storage is associated with a Snapshot can be misleading.

- The size of a Snapshot is defined to be the size of changes that are unique to that Snapshot. In other words, changed blocks that are only associated with that one Snapshot.
- The latest Snapshot will always have 0 sizes initially, as there are no changes associated with it, nothing has changed since the Snapshot was taken, so there is no space used by unique changes.
- A block can be shared by multiple Snapshots if the block has not changed between the creation of those Snapshots. Any blocks that are shared amongst multiple Snapshots are accounted for in the shared Snapshot Space total.

The following screenshot provides an example of disk usage.



- **Current Copy Size:** The current copy size is the amount of space used on the Delphix Engine by the VDBs data but across all the Time flows.
- **DB Log Size:** The DB log size is the size for archive log files.
- **Temp File Size:** This field is currently not active (for future use).
- **Container Size:** The container size is the size of the whole dataset.
- **Total Snapshot Size:** The total snapshot size is the size of all snapshots combined, including anything shared.
- **Shared snapshot space:** The shared snapshot space is the space shared by snapshots.

Reviewing historical capacity from the CLI

You can retrieve historical capacity data through the Delphix Command Line Interface (CLI).

If your Delphix Engine is a new engine you would log in as Engine Admin and ssh admin@your engine. Engines created before 5.3.1 and upgraded to 5.3.1 or later will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then disabling delphix_admin.

Process

1. Log in as Delphix Admin (or an account with admin privileges)
   `ssh delphix_admin@yourengine`
2. Navigate to capacity system historical
   `delphix > capacity system historical`
3. Then you can list a start and end date for the utilization
   `delphix capacity system historical > list startDate=<time> endDate=<time>`

For example, looking at the system space utilization, we can show 8 hours starting Sep 1 with the following:

We can also set the output detail level to different granularity, which is based on **seconds**.

```
delphix> capacity
delphix capacity> system
delphix capacity system> historical
delphix capacity system historical> list startDate=2016-09-01T00:00:00.000Z
endDate=2016-09-01T08:00:00.000Z
TIMESTAMP SOURCE.ACTUALSPACE VIRTUAL.ACTUALSPACE
2016-09-01T00:25:11.359Z 5.05TB 9.58TB
2016-09-01T00:55:28.869Z 5.06TB 9.44TB
2016-09-01T01:25:42.940Z 5.06TB 9.46TB
2016-09-01T01:56:14.585Z 5.06TB 9.38TB
2016-09-01T02:26:50.893Z 5.06TB 9.38TB
2016-09-01T02:57:15.987Z 5.06TB 9.32TB
2016-09-01T03:27:35.381Z 5.06TB 9.34TB
2016-09-01T03:57:54.657Z 5.06TB 9.35TB
2016-09-01T04:28:12.099Z 5.06TB 9.35TB
2016-09-01T04:58:22.028Z 5.06TB 9.35TB
2016-09-01T05:28:33.680Z 5.06TB 9.34TB
2016-09-01T05:58:46.666Z 5.06TB 9.27TB
2016-09-01T06:28:57.181Z 5.06TB 9.21TB
2016-09-01T06:59:39.567Z 5.06TB 9.21TB
2016-09-01T07:30:03.527Z 5.06TB 9.20TB
2016-09-01T07:50:18.548Z 5.06TB 9.20TB
```

The following example will show the data for **each day** (86400 seconds) from Sep 1 to Oct 1:

```
delphix capacity system historical> list startDate=2016-09-01T00:00:00.000Z
endDate=2016-10-01T08:00:00.000Z resolution=86400
TIMESTAMP SOURCE.ACTUALSPACE VIRTUAL.ACTUALSPACE
2016-09-01T00:25:11.359Z 5.05TB 9.58TB
2016-09-02T00:34:16.844Z 5.03TB 8.45TB
2016-09-03T00:37:50.372Z 5.10TB 8.30TB
2016-09-04T00:32:22.877Z 4.68TB 8.79TB
2016-09-05T00:34:37.715Z 4.64TB 8.56TB
2016-09-06T00:37:27.480Z 4.65TB 8.44TB
2016-09-07T00:31:17.808Z 4.70TB 8.60TB
2016-09-08T00:33:34.219Z 4.74TB 8.70TB
2016-09-09T00:43:40.760Z 4.81TB 8.57TB
2016-09-10T00:48:27.222Z 5.14TB 8.95TB
2016-09-11T00:50:41.843Z 4.90TB 8.72TB
2016-09-12T00:50:33.215Z 4.92TB 8.82TB
2016-09-13T00:48:06.350Z 4.93TB 8.72TB
2016-09-14T00:42:36.904Z 5.34TB 8.79TB
2016-09-15T00:48:58.580Z 0B 0B
2016-09-16T00:43:12.565Z 5.64TB 9.23TB
2016-09-16T21:46:06.333Z 5.74TB 9.29TB
2016-09-18T19:41:29.692Z 5.74TB 9.29TB
2016-09-19T19:38:35.268Z 5.47TB 7.94TB
2016-09-20T19:39:07.809Z 5.57TB 8.50TB
2016-09-21T19:42:32.602Z 5.64TB 9.04TB
2016-09-22T19:37:53.507Z 5.82TB 8.77TB
2016-09-23T19:37:43.373Z 5.88TB 8.96TB
2016-09-24T19:29:40.625Z 5.78TB 9.03TB
2016-09-25T19:32:30.592Z 5.80TB 8.91TB
2016-09-26T19:26:43.971Z 5.81TB 8.86TB
2016-09-27T19:33:26.939Z 5.87TB 9.44TB
2016-09-28T19:33:01.483Z 5.94TB 9.44TB
2016-09-29T19:43:55.451Z 6.15TB 9.08TB
2016-09-30T19:37:38.462Z 6.19TB 9.72TB
2016-10-01T07:35:53.644Z 5.87TB 8.98TB
```

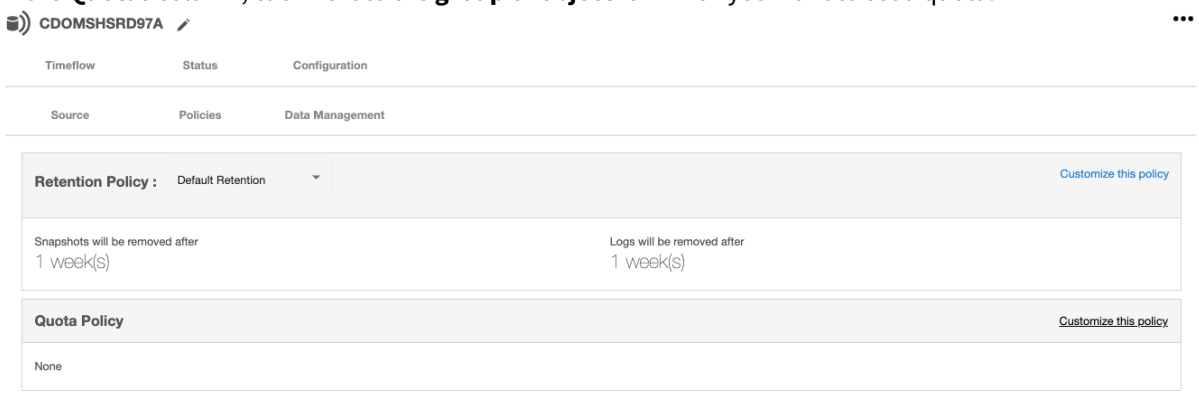
The following query will show the data for each week (604800 seconds), for a few months:

```
delphix capacity system historical> list startDate=2016-06-01T00:00:00.000Z
endDate=2016-10-01T08:00:00.000Z resolution=604800
TIMESTAMP SOURCE.ACTUALSPACE VIRTUAL.ACTUALSPACE
2016-06-01T00:16:19.391Z 3.36TB 5.96TB
2016-06-08T00:16:04.620Z 2.79TB 6.11TB
2016-06-15T00:16:35.471Z 2.95TB 4.84TB
2016-06-22T00:22:57.619Z 0B 0B
2016-06-29T00:28:48.350Z 3.38TB 6.53TB
2016-07-06T00:38:20.643Z 3.38TB 7.33TB
2016-07-13T00:36:38.876Z 3.30TB 8.05TB
2016-07-20T00:50:24.318Z 3.33TB 8.95TB
2016-07-27T00:49:06.488Z 3.36TB 9.28TB
2016-08-03T01:03:59.577Z 3.59TB 9.25TB
2016-08-10T01:12:21.949Z 3.81TB 8.80TB
2016-08-17T01:20:44.878Z 4.08TB 9.89TB
2016-08-24T01:21:01.080Z 4.23TB 9.00TB
2016-08-31T01:29:51.907Z 4.45TB 9.48TB
2016-09-07T01:31:56.383Z 4.70TB 8.67TB
2016-09-14T01:43:21.456Z 5.34TB 8.87TB
2016-09-21T01:42:34.318Z 5.58TB 8.84TB
2016-09-28T01:28:10.313Z 5.88TB 9.47TB
2016-10-01T07:35:53.644Z 5.87TB 8.98TB
```

Setting quotas

This topic describes how to set quotas for database objects.

1. Log into the **Delphix Management** application.
2. Select **Manage > Datasets**.
3. Select the **Configuration** tab for the object you want to set a quota for.
4. Select the **Policies** tab and from the Quota tile select **Customize this policy**.
5. In the **Quotas** column, click next to the **group** or **object** for which you want to set a quota.



6. In the **Quota Policy** window enter the amount of storage space you want to allocate for a quota.
7. Click **Save** to set the amount.

> Quotas and Low Space Errors
>
> Be very careful setting quotas. As a group or virtual database (VDB) approaches the quota level, snapshots may fail and logs may not be captured, causing LogSync to fail.

Quota thresholds

The following is a table of the thresholds and a description of the actions taken at each threshold. This behavior is generic across all engines.

Thresholds	Default ranges	Actions taken
Critical	Quota ≥ 95%	• Disable VDBs and dSources under quota (for a group quota, everything inside gets disabled). • Disallow jobs detailed in the DB actions table below. • Cancel all in-flight jobs listed in the DB actions table except for enabling. • Poll every minute.

Thresholds	Default ranges	Actions taken
Resume	95% > Quota ≥ 90%	• VDBs and dSources disabled by **critical** quotas will not be automatically re-enabled (DBs can still be manually re-enabled). • **Resume** fault is triggered if the quota has fallen from the **critical** threshold and if a DB was disabled from hitting the **critical** threshold. • **Warning** fault is triggered. • Poll every 3 minutes.
Warning	90% > Quota ≥ 80%	• **Warning** fault is triggered. • VDBs and dSources disabled from hitting **critical** quota will be re-enabled. • Poll every 3 minutes.
Safe	80% > Quota	• VDBs and dSources disabled from hitting **critical** quota will be re-enabled. • Poll every 5 minutes.

If a DB was disabled prior to reaching the **critical** quota, it will not be automatically re-enabled when falling to an acceptable range or when the quota is removed.

Disallowed database actions when in quota critical threshold

Here is a table of the actions that are not allowed when the quota is in the critical threshold.

DB actions table

Target is group	Target is not group
• Refresh target in the group. • Sync target in the group. • Link into the group. • Provision into the group. • Enable target in the group.	• Refresh target in the group. • Sync target in the group. • Enable target in the group.

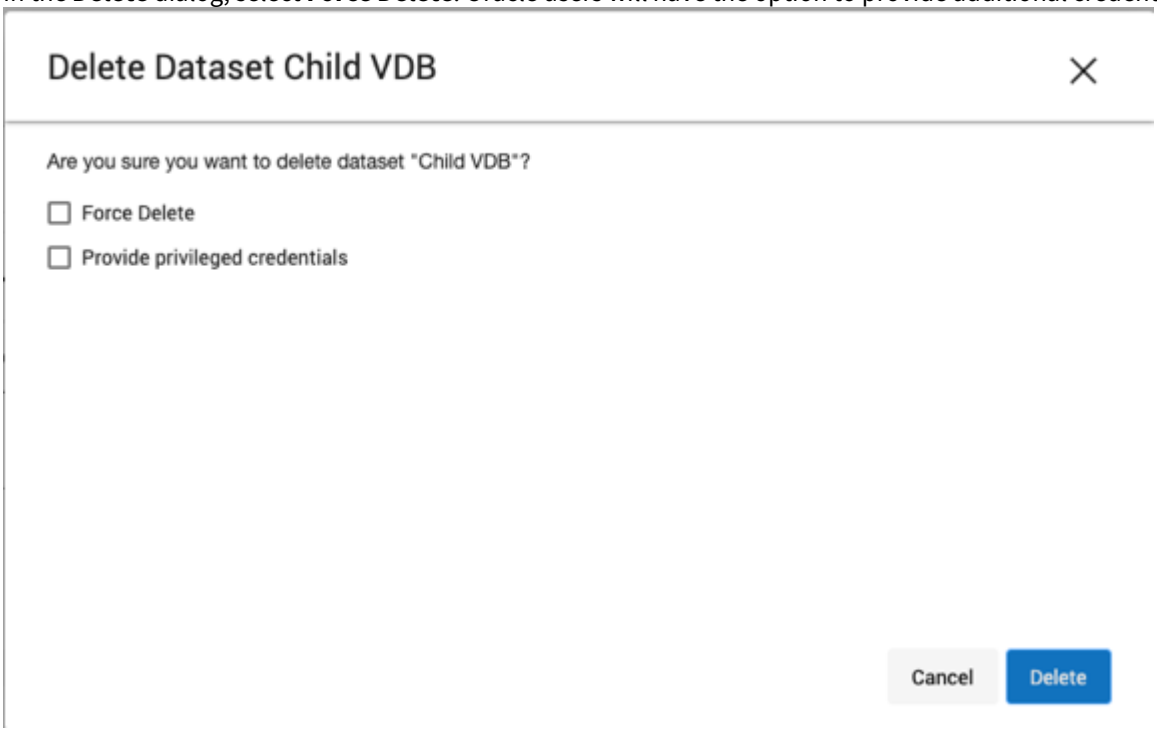
If a **critical** quota is placed on a target, dSources, and VDBs can still provision to groups that are quota below the **critical** threshold.

Deleting objects to increase capacity

This topic describes how to delete database objects to create additional capacity.

Deleting unused or outdated objects should be a regular part of Delphix Engine administration. This is especially important to prevent low space errors, which can cause the Delphix Engine to stop. The Delphix Engine holds a maximum of 400 objects.

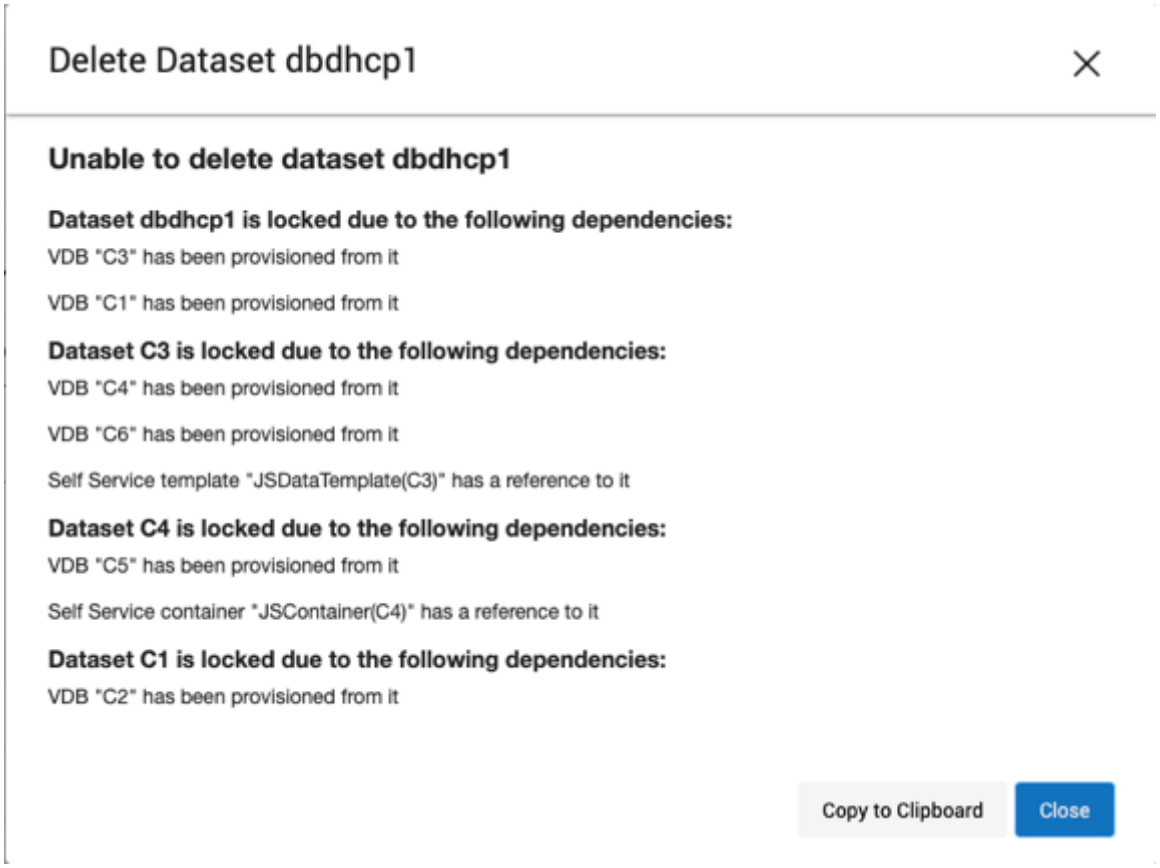
1. Log into the **Delphix Management** application.
2. Select **Resources > Storage Capacity**.
3. Next to the object you want to delete select the **Trashcan**.
4. In the **Delete** dialog, select **Force Delete**. Oracle users will have the option to provide additional credentials.



5. Click Delete.

Dependencies

If there are dependencies on the snapshot, you will not be able to delete the snapshot free space; the dependencies rely on the data associated with the snapshot. These items are displayed with a lock icon next to the name.


Delete Dataset dbdhcp1
Unable to delete dataset dbdhcp1
Dataset dbdhcp1 is locked due to the following dependencies:
VDB "C3" has been provisioned from it
VDB "C1" has been provisioned from it
Dataset C3 is locked due to the following dependencies:
VDB "C4" has been provisioned from it
VDB "C6" has been provisioned from it
Self Service template "JSDataTemplate(C3)" has a reference to it
Dataset C4 is locked due to the following dependencies:
VDB "C5" has been provisioned from it
Self Service container "JSContainer(C4)" has a reference to it
Dataset C1 is locked due to the following dependencies:
VDB "C2" has been provisioned from it
Copy to Clipboard
Close

Adding, expanding, and removing storage devices

Multiple Device Removal in the Delphix Engine 6.0.12.0 and higher version introduces a breaking kernel module change that requires a reboot to load the new module. Therefore, a deferred reboot engine upgrade operation will be unable to remove devices until a reboot is performed.

Prerequisites

For expanding a storage device after initial configuration, first make sure to add capacity to it using the storage management tools available through the device's operating system. For example, capacity can be added in vSphere using the **Edit system settings**.

VMware Hypervisor (vSphere)

Rebooting the Delphix Engine to add/expand storage is not typically necessary when using vSphere.

The guidelines for adding initial storage using all 4 virtual SCSI controllers should cause the Delphix Engine to see new storage without a reboot. However, if the new storage is leveraging a new virtual SCSI controller, the Delphix Engine will need a reboot to detect the new storage. See Deployment for VMware for more information.

Adding or increasing storage and/or cache

If it is available you can add more storage devices to the Delphix Engine.

1. Launch the **Delphix Setup** application and log in using the **sysadmin** credentials.
2. In the **Storage** section of the **Server Setup Summary** screen, click **Modify**.
3. For engines backed by disks
    a. Under the **Block Storage** tab, the Delphix engine should automatically detect any new storage devices. If a newly added storage device does not appear in the **Storage** section of the **Server Setup Summary** screen, click **Rediscover**.
    b. Select the **Enable** check box before the device name to add the device to the storage pool and click **Save**.
4. For Delphix Elastic Data Engines
    a. You can modify the maximum amount of data that can be stored by the s3 storage under the **Object Storage** tab.
    b. The **Block Storage** tab lets you modify the number of EBS volumes used for cache. Please see step 3 above.
5. Click **Save**.

Expanding a storage/cache device

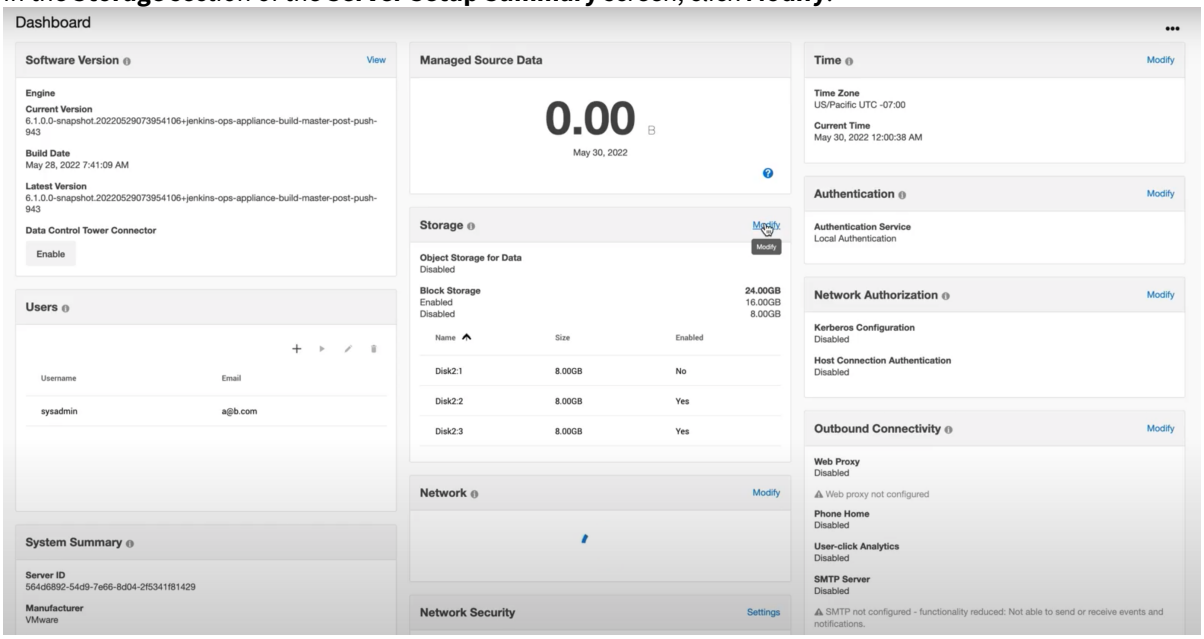
1. Launch the **Delphix Setup** application and log in using the **sysadmin** credentials.
2. In the **Storage** section of the **Server Setup Summary** screen, click **Modify**.
3. Under the **Block Storage** tab, select **Expand** for each device that you want to expand. The **Expand** checkbox appears next to the name of devices that have added capacity (in other words, the underlying LUN has been expanded), and the **Unused** column indicates how much capacity is available for each device.
   Note:

It is imperative that the VMDKs created and assigned to the Delphix Engine do not exceed the capacity of the backing datastore. Any 'overcommitment' of storage resources can be expected to cause an unplanned outage, and may not be recoverable.

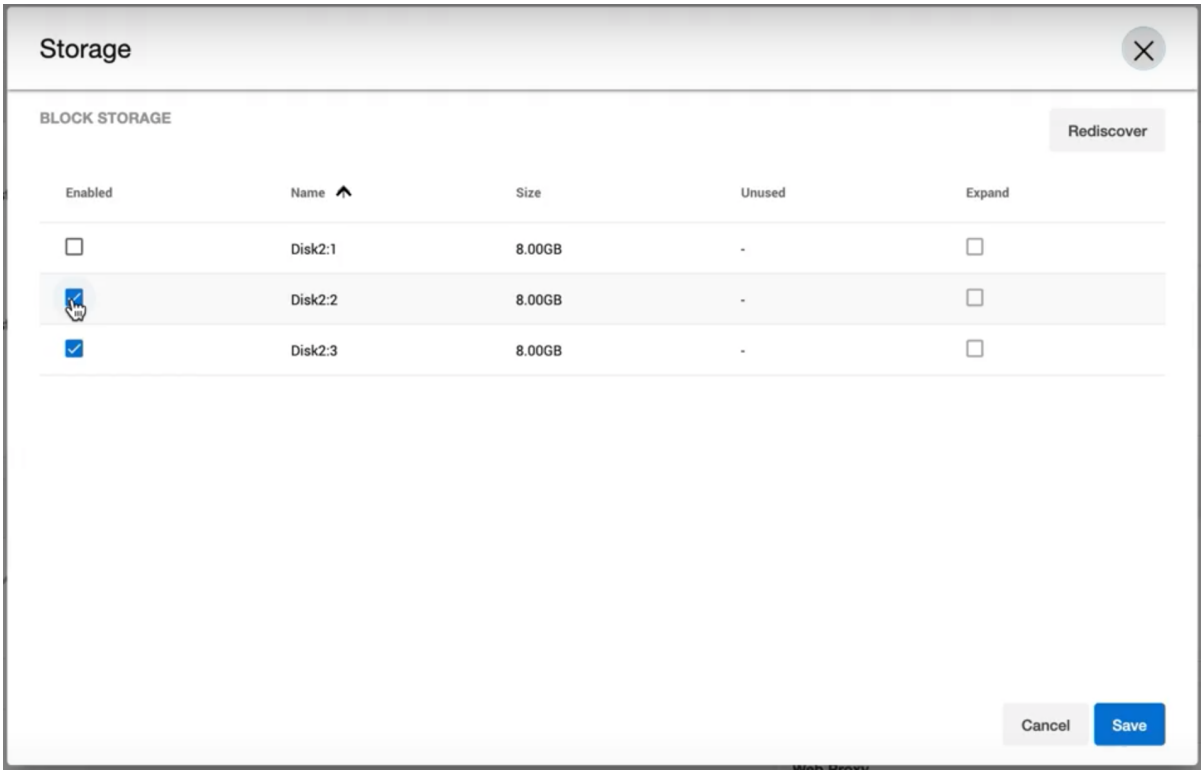
4. Click **Save**.

Removing device storage

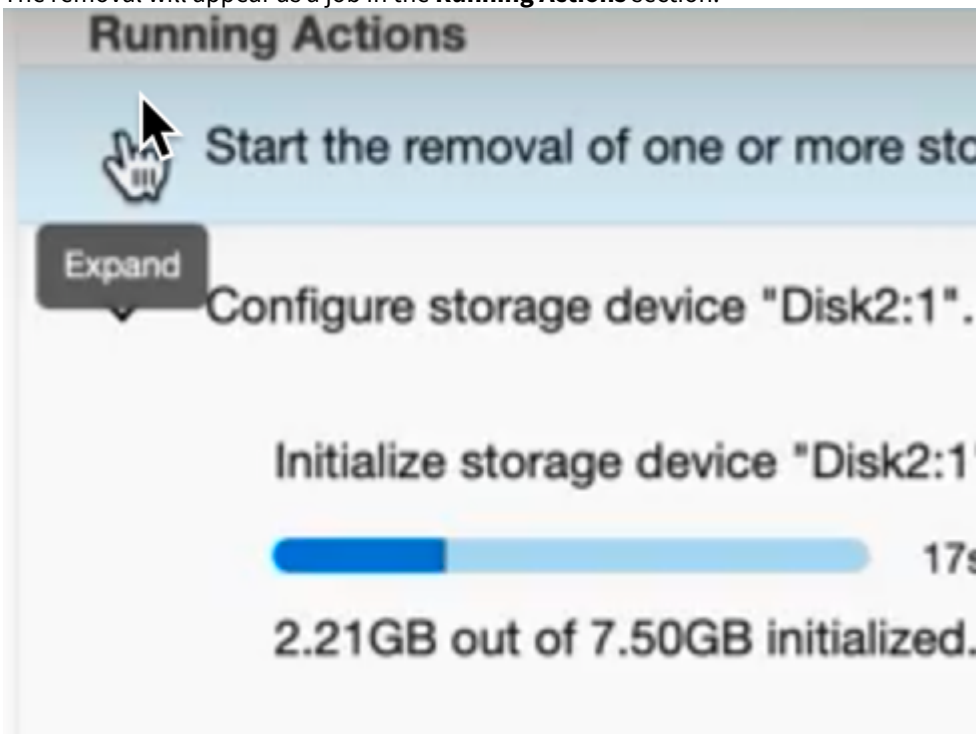
1. Launch the **Delphix Setup** application and log in using the **sysadmin** credentials.
2. In the **Storage** section of the **Server Setup Summary** screen, click **Modify**.



3. In the **Block Storage** tab, uncheck the **Enable box** for any storage device that will be removed.
    a. A minimum of 10GB of object storage is required, thus, a device or devices must be enabled to accommodate for at least 10GB. This will give an error otherwise.



4. Once the desired storage devices are disabled, click **Save**.
5. The removal will appear as a job in the **Running Actions** section.



Delphix storage migration

Getting started

Delphix storage migration allows you to remove storage devices from your Delphix Engine, provided there is sufficient unused space, thus allowing you to repurpose the removed storage device to a different server. You can also use this feature to migrate your Delphix Engine to different storage by adding the new storage devices and then removing the old storage devices.

> **Feature compatibility**
> This feature is only compatible with Delphix Engine Releases 5.0.4 and later. This feature is not supported on Delphix Elastic Data Engines (engines backed by blob/object storage).

Possible Migration Methods

Method	Pros	Cons
Delphix Storage Migration	• Good for migrating storage that was accidentally added to the engine or added to the engine improperly (wrong size). • May reduce fragmentation if new storage is added to replace old disks. • Good for migrating a small amount of storage (e.g.: < 10 TB).	• With Delphix versions prior to 5.3, this mapping table can consume 2-3GB of RAM for every 1TB of allocated data that is migrated, if the disk being removed has a high level of fragmentation. From version 5.3, DxFS will migrate the data in larger blocks, comprising both allocated and unallocated space. This allows for significantly fewer mapping entries, with memory usage typically reduced to 50-100MB per TB of allocated data that is migrated. • May increase fragmentation on remaining disks if no new disks are added. • Depending on the size of the disk and storage performance this method could be less performant than other methods. • Each device removed could take longer than the previous one as data is remapped across the remaining disks. • The maximum number of 20 devices can be removed in releases prior to Delphix version 5.1.
vMotion	• Fast	• If there is high fragmentation on the existing disks, this is copied to the new disks.

Method	Pros	Cons
Delphix Replication	• Data is completely rewritten from one Delphix Engine to another which significantly reduces fragmentation on the new Delphix Engine. • Replication can be configured to limit the impact on the network (compression and bandwidth). • Replication is resumable from network disruptions, on in the event of replication source or target stack/ host restarts. It is currently NOT possible to manually suspend/ resume a replication job.	• Depending on the number of objects to replicate as well as network and storage performance, this method could be considered slow. • This does require an outage to "migrate" the objects from the replication source Delphix Engine to the replication target Delphix Engine - outage time depends on several factors like the number of objects, incremental replication time, time to enable/disable objects, etc. • Only migrates storage objects like VDBs/dSources, and dependent environment information. Other items like users/policies/events/job history/config templates are NOT replicated.

Understanding Delphix storage migration

Delphix storage migration is a multi-step process:

1. Identify a storage device for removal. The device you choose will depend on your use case.
    a. To remove extra storage that is unused, you can select any device for removal. For best performance, select the device with the least allocated space; typically, this is the device that you added most recently. The allocated space is defined by the `usedSize` property of the storage device:

```
test-env@delphix 'Disk10:2'> ls
Properties
    type: ConfiguredStorageDevice
    name: Disk10:2
    bootDevice: false
    configured: true
    expandableSize: 0B
    model: Virtual disk
    reference:  STORAGE_DEVICE-6000c293733774b7bb0e4aea83513b36
```

```
    serial: 6000c293733774b7bb0e4aea83513b36
      size: 8GB
      usedSize: 7.56MB
    vendor: VMware
```

   b. To migrate the Delphix Engine to new storage, add storage devices backed by the new storage to the Delphix Engine. Then remove all the devices on the old storage.
2. Use the Delphix command-line interface (CLI) to initiate the removal of your selected device.
3. Data will be migrated from the selected storage device to the other configured storage devices. This process will take longer the more data there is to move; for very large disks, it could potentially take hours. You can cancel this step if necessary.
4. The status of the device changes from **configured** to **unconfigured** and an alert is generated to inform you that you can safely detach the storage device from the Delphix Engine. After this point, it is not possible to undo the removal, although it is possible to add the storage device back to the Delphix Engine.
5. Use the hypervisor to detach the storage device from the Delphix Engine. After this point, the Delphix Engine is no longer using the storage device, and you can safely re-use or destroy it.

Limitations of Delphix storage migration

After removal, the Delphix Engine uses memory to track the removed data. In the worst-case scenario, this could be as much as 2-3 GB of memory per TB of used storage. Note that this is used storage; the overhead of removing a 1TB device with only 500MB of data on it will be much lower than the overhead of removing a 10GB device with 5GB of data on it.

User interface

Delphix storage migration is currently available exclusively via the CLI. There are two entry points.

- `verifyStorageDeviceRemoval` - Verifies available pool space and available memory.
- `startStorageDeviceRemoval` - Kicks off job, detailed discussion of job concurrency/recovery.

Device removal for storage migration

Do not remove a configured storage device

Do not remove a configured storage device or reduce its capacity. Removing or reducing a configured storage device will cause a fault with the Delphix Engine, and will require the assistance of Delphix Support for recovery.

1. Identify which device you want to remove.
   a. If you are using a VMware **RDM** disk, note the UUID of the device by looking at its name in the vSphere GUI. For more information, see the Getting the UUID of a RDM Disk from VMware, via the vSphere GUI article.
   b. If you are using a VMware **virtual** disk, note the UUID of the device via the vSphere API. See the section of this VMware KB article on how to get the UUID of your virtual disk.
   c. In EC2, note the attachment point – e.g., `/dev/sdf`.
   d. In KVM, note the UUID.
2. Login to the Delphix CLI as a **sysadmin** user.
3. Navigate to the storage/device directory with `cd storage/device`.
4. Select one or more devices to remove and make note of the name (e.g., select "Disk10:2" and "Disk10:3").

```
test-env@delphix storage/device> ls
Objects
NAME      CONFIGURED  SIZE  EXPANDABLESIZE
Disk10:2  true        8GB   0B
Disk10:0  true        24GB  0B
Disk10:1  true        8GB   0B
Disk10:3  true        8GB   0B
```

5. (For VMware only) Confirm the disk selection is correct by validating that the serial matches your UUID:

```
test-env@delphix storage device 'Disk10:2'> ls
Properties
   type: ConfiguredStorageDevice
   name: Disk10:2
   bootDevice: false
   configured: true
   expandableSize: 0B
   model: Virtual disk
   reference: STORAGE_DEVICE-6000c2909ccd9d3e4b5d62d733c5112f
   serial: 6000c2909ccd9d3e4b5d62d733c5112f
   size: 8GB
   usedSize: 8.02MB
   vendor: VMware
```

6. Navigate to `storage/remove`. Run `verify` and set the devices to be removed, as shown below.

```
test-env@delphix storage remove > verify
test-env@delphix storage remove verify *> set devices="Disk10:2,Disk10:3"
test-env@delphix storage remove verify *> commit
   type: StorageDeviceRemovalVerifyResult
   newFreeBytes: 15.85GB
   newMappingMemory: 3.14KB
   oldFreeBytes: 23.79GB
   oldMappingMemory: 0B
```

7. Navigate to `storage/remove`. Run `start` and set the devices to be removed, as shown below.

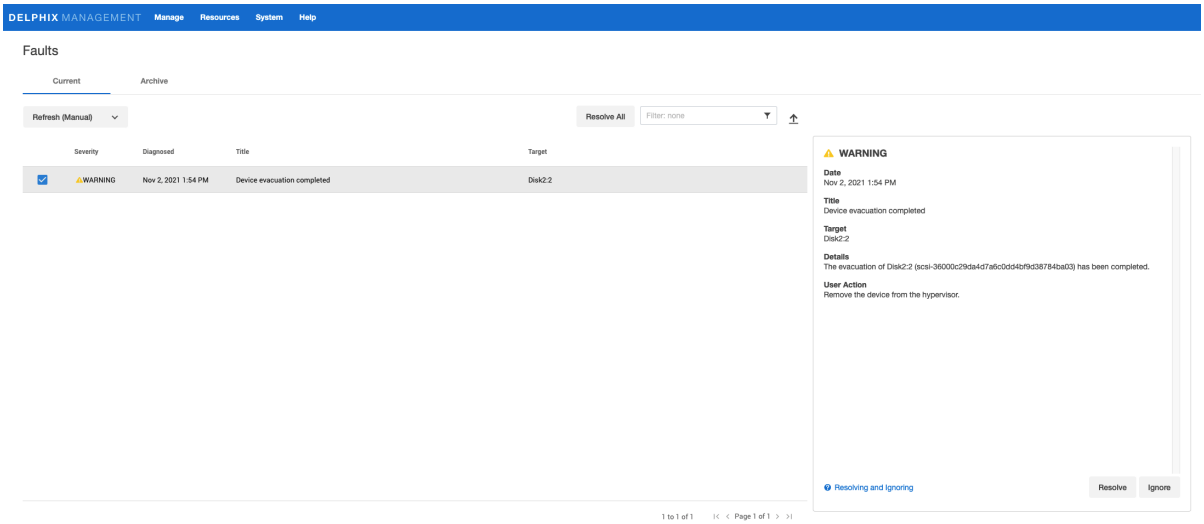
```
test-env@delphix storage remove > start
test-env@delphix storage remove start *> set devices="Disk10:2,Disk10:3"
test-env@delphix storage remove start *> commit
    Dispatched job JOB-4
    STORAGE_DEVICES_START_REMOVAL job started.
    STORAGE_DEVICES_START_REMOVAL job completed successfully.
```

Note : This does not signify that the device migration has been completed. A `STORAGE_DEVICE_REMOVAL` job will start for each device to be removed, which handles the data migration from that disk.

8. Wait for device evacuation to complete. Alternatively, you can cancel the evacuation.

Note:
Do not detach the device from the Delphix Engine in your hypervisor until the data evacuation is completed. You can monitor the progress of the STORAGE_DEVICE_REMOVAL job in the Management GUI under **System >Jobs**.

9. Once the device evacuation has been completed, the job will finish and a fault will be generated. Detach the disks from your hypervisor and the fault will clear on its own. An example of the fault created is shown below.



Using VMDKs

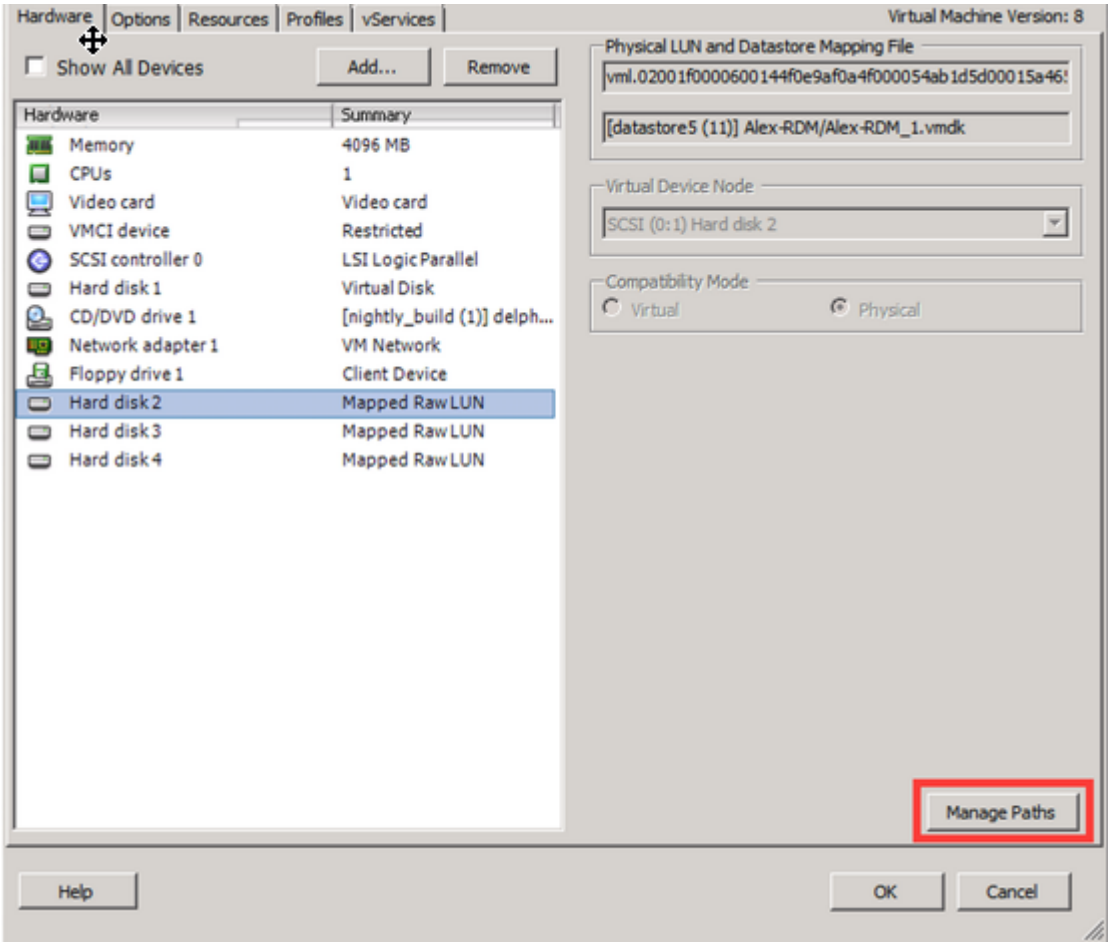
When using VMDKs, deleting the wrong VMDK could cause data loss. Therefore, it is highly advisable to detach the device, then verify that the Delphix Engine continues to operate correctly, and lastly delete the VMDK.

Getting the UUID of an RDM disk from VMware via the vSphere GUI.

In the event that the disk serial number displayed in Delphix does not match the UUID in VMware, the Delphix Engine must be powered off and back on in order to make VMware provide the correct values to the guest operating system (Delphix). This has been necessary when using **vmkfstools** with **setuuid**. When forcing the guest OS to re-read the SCSI sense data for the device, VMWare still provides the original values. Even after a simple reboot VMWare still provides the previous UUID values. It was not until the VM was explicitly powered off and back on did VMWare present the new UUID values to the guest. After which the UUIDs matched between the **vmkfstools getuuid** command and the CLI output.

In the ESX graphical user interface (GUI), select your **VM**.

1. Click **Edit settings**.
2. If not already displayed, select the **Hardware** tab.
3. Select the **device** you want to remove.
4. Click **Manage Paths**.



The UUID of the device appears in the title bar, as seen below.

Getting the UUID of a VMDK from VMware, via ssh to the ESX server

1. ssh onto the ESX server as the root user.
2. Navigate to the directory containing the .vmdk files for the Delphix VM.

```
Use the 'vmkfstools -J getuuid <.vmdk filename>' command to obtain the UUID,
for example:
```

```
/vmfs/volumes/25894daa-f7b2b044/delphix01-2356 # vmkfstools -J getuuid
delphix01-2356_1.vmdk
UUID is 60 00 C2 91 01 bc 8e 72-31 a4 cd b0 b3 f6 e5 74
```

Getting the UUID of a VMDK from VMware, via VMware PowerCLI

```
PS C:\> Connect-VIServer -Server durban -Protocol https -Username root -Password
root_password

Name                           Port  User
----                           ----  ----
durban                         443   root


PS C:\> Get-VM delphixVM | Get-HardDisk | select name,filename,@{name="UUID";expr={$_
.extensiondata.backing.uuid}}

Name                                       Filename
UUID
----                                       --------
----
Hard disk 1                                [zfs_delphixVM] dlpx-5.1.9.0-432-61155cf.
.. 6000C294-a115-0327-e417-02560d86e944
Hard disk 2                                [zfs_delphixVM] dlpx-5.1.9.0-432-61155cf.
.. 6000C299-38fe-5050-1eb2-1ee6db62b257
Hard disk 3                                [zfs_delphixVM] dlpx-5.1.9.0-432-61155cf.
.. 6000C294-662d-c674-8957-03e0514b7006
Hard disk 4                                [zfs_delphixVM] dlpx-5.1.9.0-432-61155cf.
.. 6000C29d-0719-1072-0f85-96da2efef4a3

PS C:\> Disconnect-VIServer

Confirm
Are you sure you want to perform this action?
Performing operation "Disconnect VIServer" on Target "User: root, Server: durban,
Port: 443".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "Y")
: Y
```

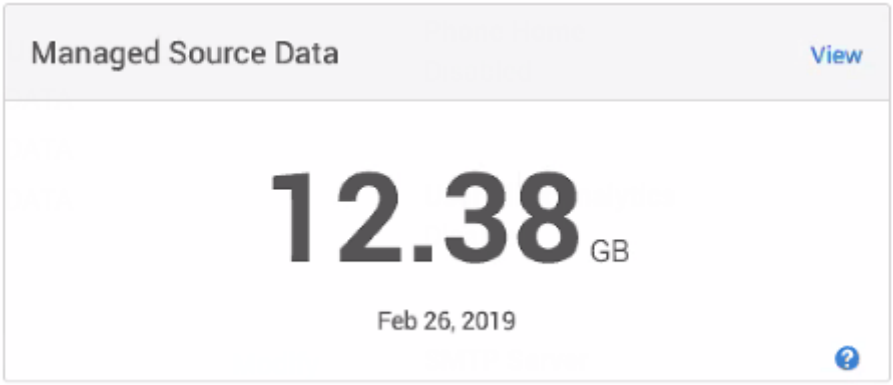
Managing source data

The **Managed source Data** dashboard tile provides an aggregate sum of the size of ingested source databases. This helps to understand the total amount of data per-engine that is managed by Delphix.

Managed Source Data refers to the size of the source data that is ingested and managed by the Delphix Engine. It is the physical, allocated size of the source database.

Today, the value displayed represents the source data size for Oracle, SQL Server, and SAP ASE databases. Since this value is a measure of source size, it is not affected by Delphix constructs or operations, e.g. the sizes of dSources, VDBs, snapshots. Since the intent is to measure the size of the sources, data from replication or Selective Data Distribution are not included in the sum for the engine. The total usage is displayed in gigabytes.

When an admin (Delphix Management Application) or sysadmin (Delphix Setup) login to the Delphix Engine they will have access to the Managed Source Data tile (shown below) from their dashboard.

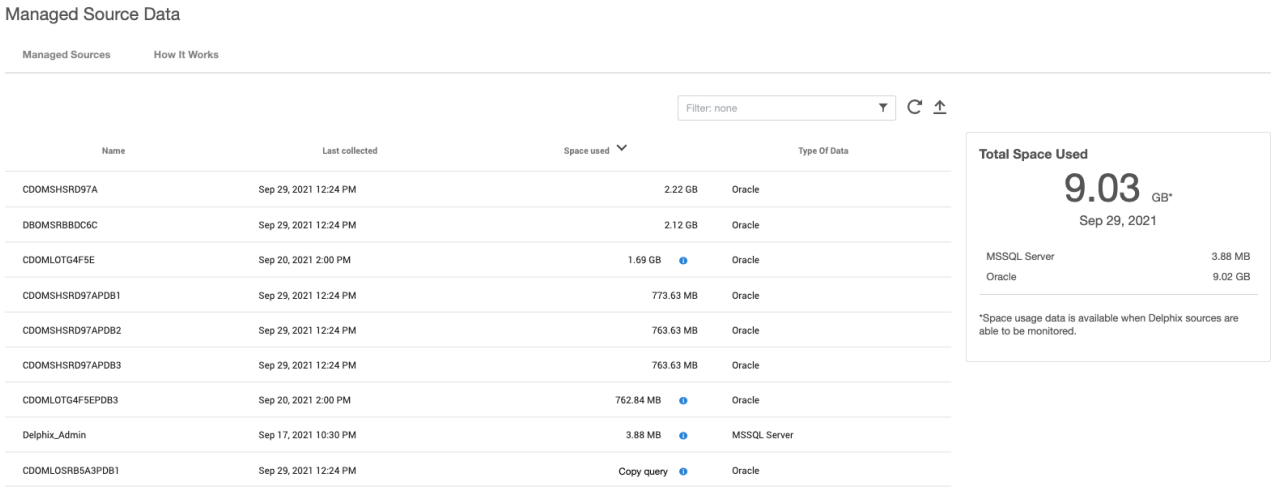


Selecting the **View** link located on the top-right takes users to the Managed Source Data page. This page provides a detailed breakdown of the total usage and can also be reached via **Resources** > **Managed Source Data** from the navigation bar.

Blue question marks indicate:

- A “fallback” value is being used in the sum: A fallback value is used when sources are unavailable at the time when the usage query is made (e.g., the source is disabled), a “fallback” value is used in the sum. The fallback value refers to the last usage value Delphix collected while connected to the source.
- Sources that are not included in the sum: These are source types for which Delphix does not currently collect data (for example, sources like EBS, DB2, HANA, PostgreSQL).

Hovering over the **blue question marks**, a tooltip appears to describe the exact situation (whether or not Delphix used historical data/fallback values or if the source for collection is unsupported).



To search the **Managed source Data** table, enter the name of the data source you are looking for. The grid will refresh to display the selected data source. Select to refresh data to get updated space usage information. To export the information provided in the grid to a .csv file select.

Backend data refresh

The backend data refreshes at a regular interval but does not trigger a UI refresh.

An overview of held space

Scenario description

The existence of a Held Space is not indicative of an issue but rather a representation of an underlying storage dependency.

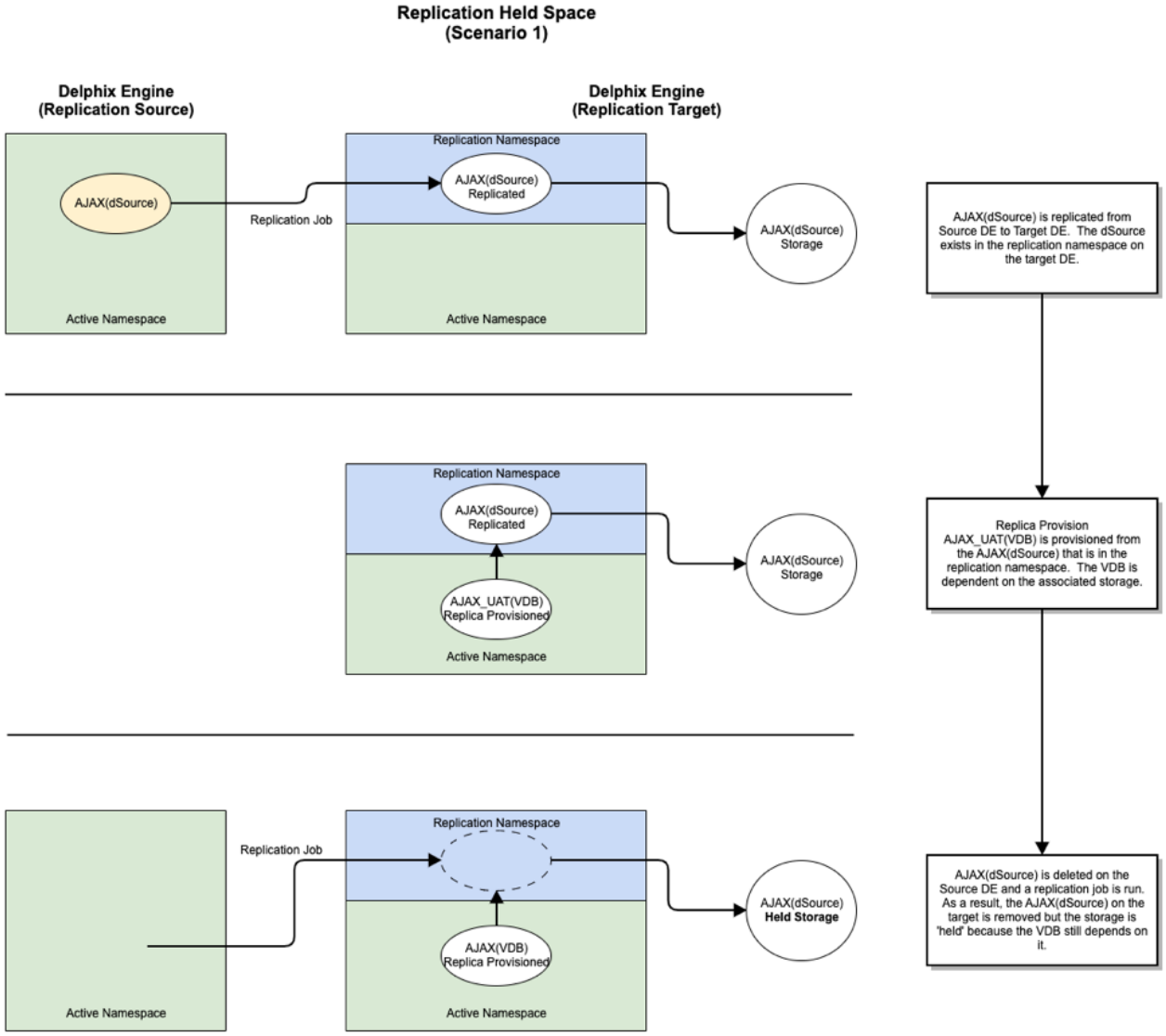
There are three scenarios that can create Held Space.

Scenario 1:

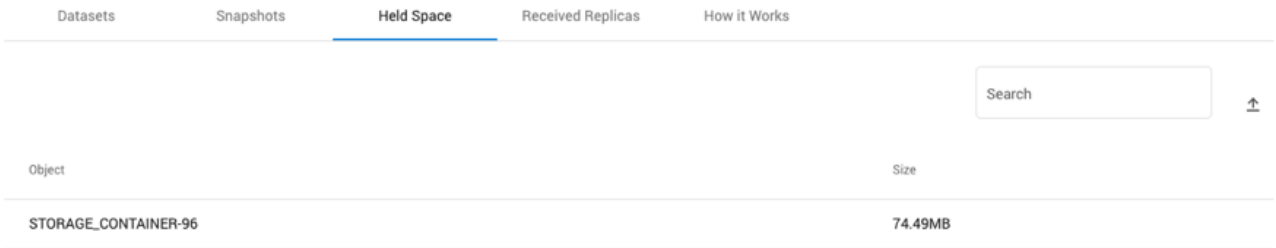
The first scenario can occur when you replicate a dSource to a second engine, which we'll call a Replication Target.

If you then provision a VDB from the Replication Target, and the dSource is subsequently deleted from the Replication Source, you will create a Held Space when that delete replicates to the Replication Target engine.

When this happens, the deleted dSource is removed from the target, but its storage remains held because it is needed by the replica provisioned VDB.



Held Space can be viewed in the **Storage Capacity** screen by selecting the Held Space tab.



Scenario 2:

The second scenario can occur in the context of a snapshot that is in use by a Delphix Self-Service branch or bookmark.

The snapshot may be removed via policy, but until the branches and/or bookmarks that leverage that snapshot are removed, its space will be held.

Scenario 3:

The third scenario can occur during Selective Data Distribution where a masked VDB that is provisioned from a dSource is replicated to a Replication Target.

The metadata objects relating to the dSource could contain sensitive information like hostnames, user names, and passwords. Hence they are not replicated to the Replication Target. The data belonging to the dSource is replicated after redacting any sensitive information. This redacted data shows up as held space.

Scenario 4:

The first scenario is caused by a failed or interrupted Replication job and results in held space on the Replication source engine. During replication, all data blocks to be replicated are serialized for transfer. Once the transfer is complete, the serialized data is released.

However, if a Replication job fails, the serialized data is retained until the next successful Replication job execution, or the Replication Profile is deleted - whichever happens first.

In a scenario where dSource and VDB snapshots in the serialized data reach the Retention Policy limit before the Replication job successfully executes, that snapshot data will become held space. This is most likely to happen if either of the following is true:

- A failed Replication job goes a long time before being successfully re-run.
- Replication schedules are configured with a period that is longer than one or more Retention Policies.

Monitoring and log management

Using both Delphix and its associated datasets will generate many types of logs. Monitoring these logs are a key component of good diagnosability. Here, we explain the types of logs that Delphix creates as well as the monitoring tool integrations we support, such as SNMP and Splunk.

This section covers the following topics:

Configuring SNMP

This topic describes how to configure SNMP.

Starting with version 6.0.6.0 the Delphix Engine adds support for SNMP version 3 by implementing the User Security Model (USM) of the SNMP version 3 (SNMPv3) specification. Version 3 of SNMP adds stronger security compared to version 2.

> ⚠ When SNMP v2 is configured and enabled, SNMP v1 is also configured and enabled. Similarly, v2 is configured and enabled with v3. This is a known behavior of the net-snmp package used to implement SNMP on the engine.

USM replaces the community string as in SNMP version 2 with user records. The Delphix engine can both receive and send SNMP version 3 messages only by using the USM security model. The USM model provides more security over version 2 by hashing passwords and encrypting the payload.

The Delphix Engine includes an SNMPv3 agent that is only capable of responding to read-only operations, such as GET messages using the User Security Model. USM users can be configured via the GUI, CLI, or API. Each user is composed of a username, authentication password, authentication protocol, privacy password, and a privacy protocol.

The Delphix Engine is also capable of sending alerts and faults over SNMP with the User Security Model. Messages are sent identically as with SNMP version 2 except that the USM model is used to authenticate and encrypt the alert message against a customer's manager. Users will need to properly configure their managers to receive messages. The use of the test function can be used to test configuration.

Prerequisites

There are no prerequisites for enabling SNMP to provide system status. The following are prerequisites for sending alerts to an external SNMP manager.

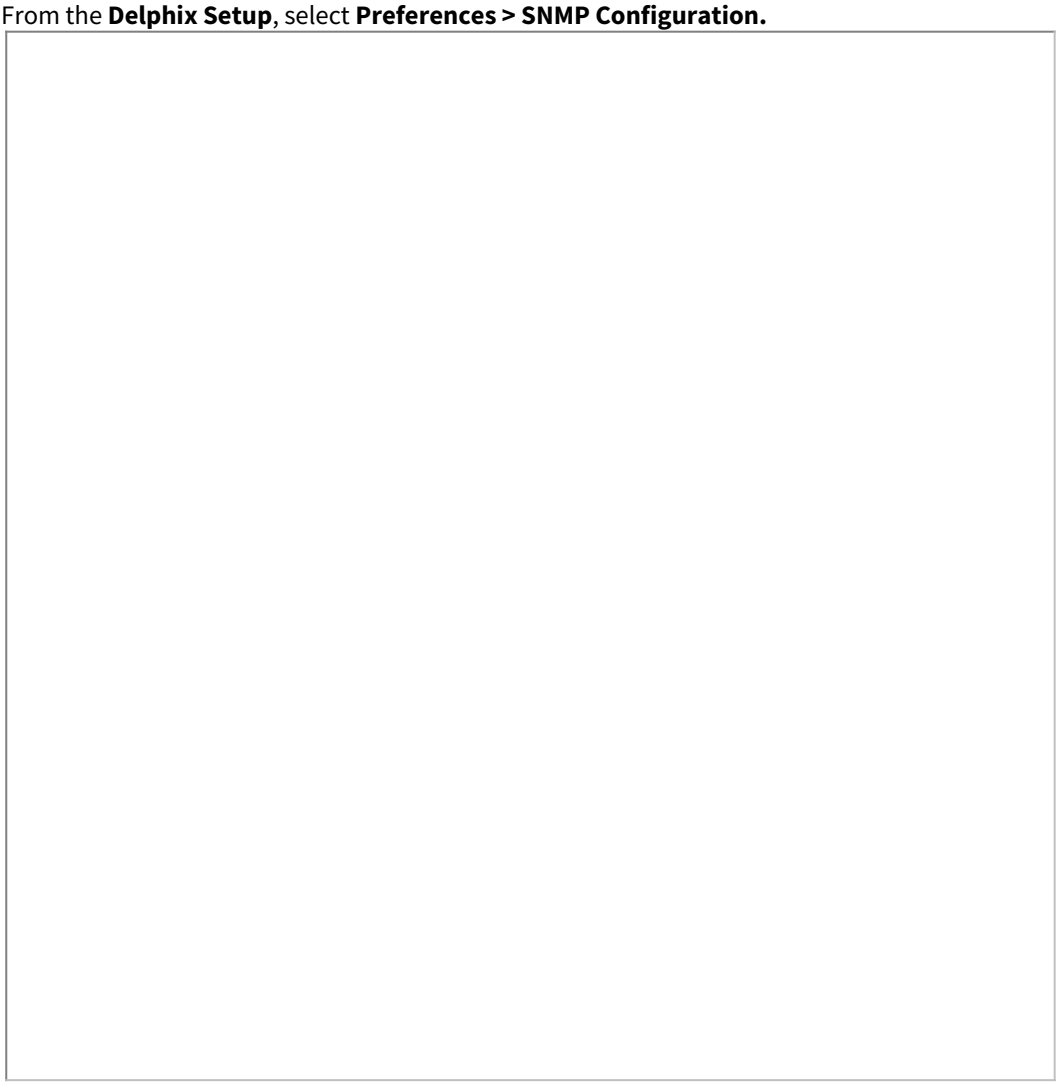
- At least one SNMP manager must be available and must be configured to accept SNMPv2 InformRequest notifications.
- Delphix's MIB (Management Information Base) files must be installed on the SNMP manager or managers. These MIB files describe the information that the Delphix Engine will send out. They are attached to this topic.

File name	Content-type	File Size
DELPHIX-ALERT-MIB.txt	text/plain	5 kB
DELPHIX-MIB.txt	text/plain	0.8 kB

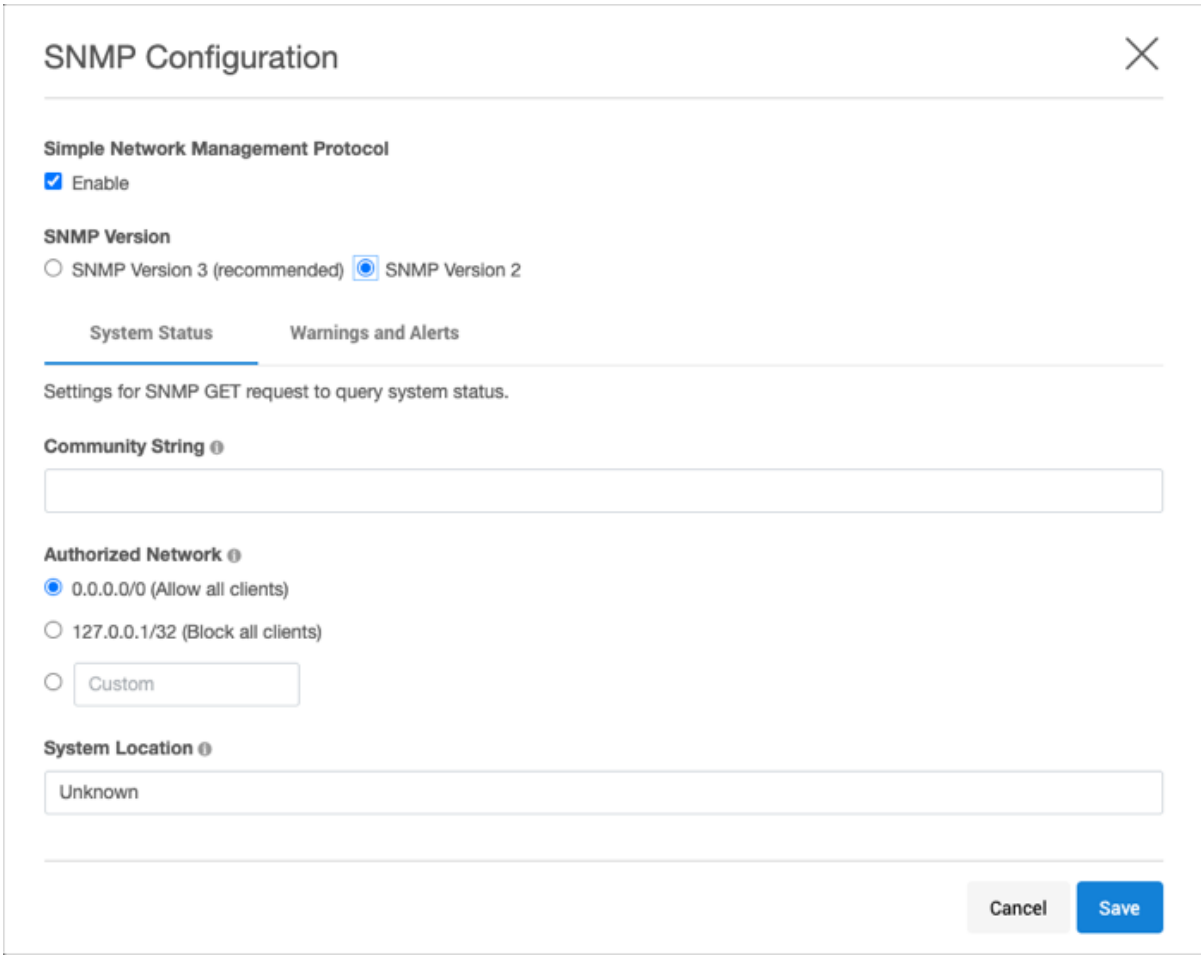
Configuring SNMP for v2

1. On the Delphix Engine login screen, select **Delphix Setup**.
2. In the Delphix Setup login screen, enter the **sysadmin** username and password.
3. Click **Log In**.

4. From the **Delphix Setup**, select **Preferences > SNMP Configuration.**



5. Select **Enable**.
6. Select **SNMP Version 2**.

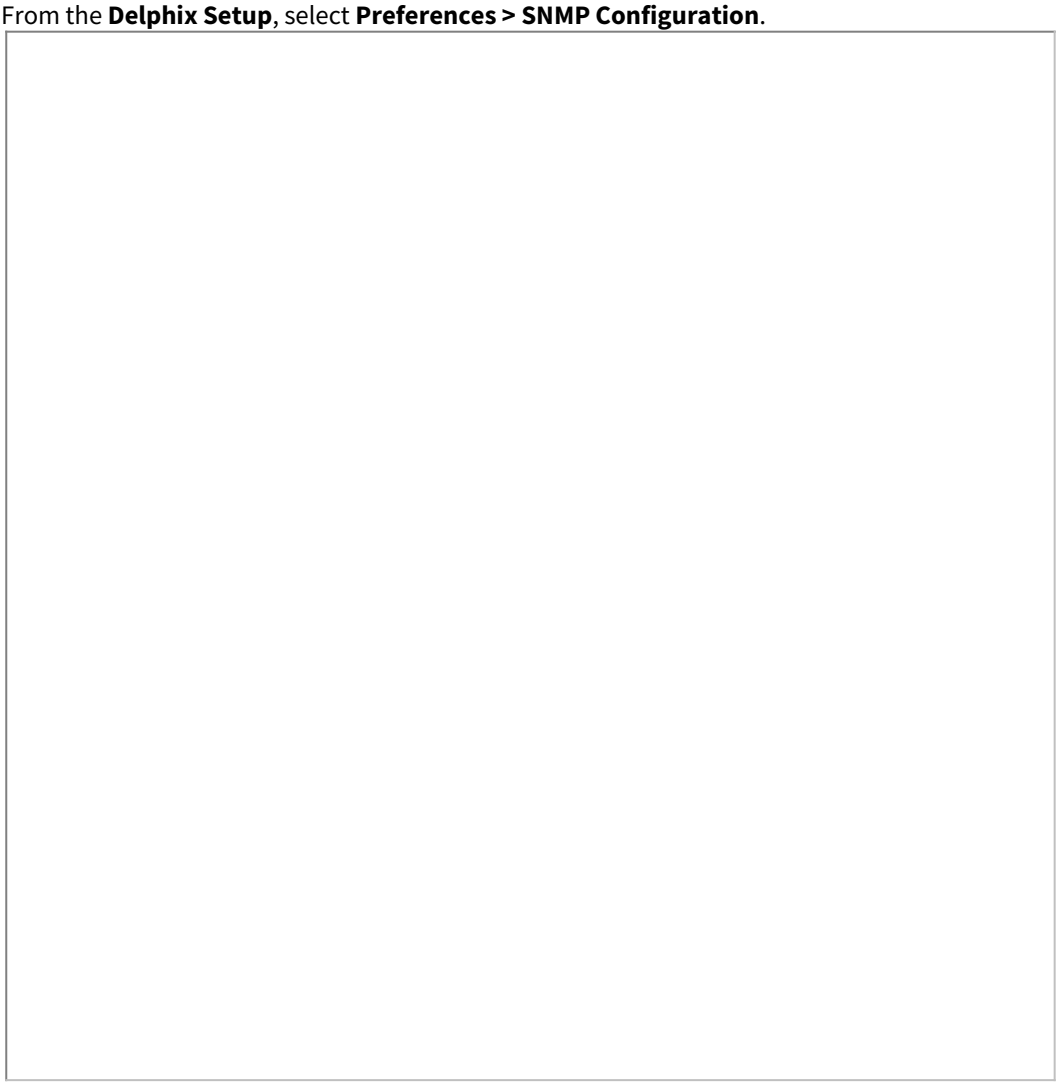


7. In **Community string**, enter the community string. It is the string that SNMP clients must provide in order to be authorized to retrieve SNMP information from the Delphix Engine.
8. OPTIONAL: change the following settings:
    a. **Authorized network** – The set of client IP addresses (in CIDR notation) authorized to retrieve SNMP information from the Delphix Engine. To allow all clients, set this to 0.0.0.0/0 (the default). To prevent all clients from connecting, set this to the loopback address, 127.0.0.1/32.
    b. **System location** – A free-form text description of the Delphix Engine's physical location. This is provided as the value for MIB-II OID.
9. Click **Save** to commit the SNMP configuration.

Configuring SNMP for v3

1. On the Delphix Engine login screen, select **Delphix Setup**.
2. In the Delphix Setup login screen, enter the **sysadmin** username and password.
3. Click **Log In**.

4. From the **Delphix Setup**, select **Preferences > SNMP Configuration**.



5. Select **Enable**.
6. Select **SNMP Version 3** (recommended).
7. Click **Save** to commit the SNMP configuration.

Configure SNMP version 3 agent

After enabling SNMP and choosing version 3, the SNMP agent can be configured with the User Security Model (USM) users.

1. Click on **System Status**.
2. Click on the **+** to show the **New SNMP USM User Information** dialog.

3. Provide the following information for the user:
    a. **Username -** A string representing the name of the user.
    b. **Authentication method -** An indication of whether messages sent on behalf of this user can be authenticated, and if so, the type of authentication protocol that is used. Users can select from:
    MD5 -the HMAC-MD5-96 authentication protocol.
    SHA -the HMAC-SHA-96 authentication protocol.

c. **Authentication passphrase -** If messages sent on behalf of this user can be authenticated, this key will be used with the authentication key for use with the authentication. This field requires a minimum of 8 characters.
d. **Encryption method -** Provides support for data confidentiality. The designated portion of an SNMP message is encrypted and included as part of the message sent to the recipient. Users can select from
AES - Advanced Encryption Standard
DES - Data Encryption Standard
e. **Encryption passphrase** - If messages sent on behalf of this user can be encrypted, this key will be used with the encryption method for use with the authentication. This field requires a minimum of 8 characters.

4. Click **Save** to add your user.

Configuring SNMP version 3 managers

1. Click on the **Warnings and Alerts** tab.
2. Choose the **Severity** of the alerts that are sent via SNMP. The severity is ranked in order from most restrictive to least restrictive: critical, warning, and informational. Setting the severity critical will only include critical alerts while setting to informational will cause an informational, warning and critical alerts to be sent over SNMP to all managers.
3. Click **+** to add a manager to show the **New SNMP manager information** dialog**.**

4. Provide the following information:
   a. **Username -** A string representing the name of the user.
   b. **Host address** - Host or IP address.
   c. **Port** - Port used by the SNMP Engine.
   d. **Authentication method -** An indication of whether messages sent on behalf of this user can be authenticated, and if so, the type of authentication protocol that is used. Users can select from:
   MD5 -the HMAC-MD5-96 authentication protocol.
   SHA -the HMAC-SHA-96 authentication protocol.
   e. **Authentication passphrase -** If messages sent on behalf of this user can be authenticated, this key will be used with the authentication key for use with the authentication. This field requires a minimum of 8 characters.
   f. **Encryption method -** Provides support for data confidentiality. The designated portion of an SNMP message is encrypted and included as part of the message sent to the recipient. Users can select from
   AES - Advanced Encryption Standard
   DES - Data Encryption Standard
   g. **Encryption passphrase** - If messages sent on behalf of this user can be encrypted, this key will be used with the encryption method for use with the authentication. This field requires a minimum of 8 characters.
5. By default, TRAP messages are sent and require the agent's authoritative engine id for the user to be configured with the SNMP engine ID from the Delphix Engine, which can be obtained via the CLI. Click Use **Inform instead of Trap** to use INFORM messages, which are more reliable and require less configuration at the expense of higher network traffic.
6. Click **Save**.

CLI: viewing SNMP engine ID

Viewing SNMP Engine ID

```
> service snmp v3
```

```
service snmp v3> ls
Properties
    type: SNMPV3Config
    enabled: true
    engineId: 0x80001f88801ad80b5e62a7f85f00000000
    location: Unknown
    securityModel: USM
    severity: WARNING

Children
manager
usm

Operations
update
```

Supported MIBs

The Delphix software can be configured to send SNMP traps when Delphix alerts are generated as described in the procedure above. In order to process these traps in your SNMP manager software, you will need the base Delphix MIB and the Delphix Alert MIB.

In addition to generating traps, the Delphix Engine supports read-only access to the following MIBs for basic system monitoring purposes.

- The following MIB-II object hierarchies are defined in RFC 1213:
  - system (OID .1.3.6.1.2.1.1): Provides basic system identity information
  - interfaces (OID .1.3.6.1.2.1.2): Provides network interface information including I/O statistics
  - IP (OID .1.3.6.1.2.1.4): IP protocol information including IP addresses configured, routes, and IP statistics
- The following UCD objects defined in http://www.net-snmp.org/docs/mibs/ucdavis.html
  - memory usage (OID .1.3.6.1.4.1.2021.4)
  - CPU usage (OID .1.3.6.1.4.1.2021.11)
  - Disk I/O statistics (OID .1.3.6.1.4.1.2021.13.15)

Examples

The following examples assume that you have enabled SNMP on a Delphix Engine named example.company.com, and have set the community string to "public".

1. Walking the MIB-II objects using the net-snmp snmpwalk tool:

```
$ snmpwalk -v 2c -c public example.company.com
SNMPv2-MIB::sysDescr.0 = STRING: Delphix Engine 5.1.6.0 DelphixOS 5.1.2017.03.2
4
SNMPv2-MIB::sysObjectID.0 = OID: SNMPv2-SMI::enterprises.41028
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (476432) 1:19:24.32
SNMPv2-MIB::sysContact.0 = STRING: administrator@company.com
SNMPv2-MIB::sysName.0 = STRING: example.company.com
SNMPv2-MIB::sysLocation.0 = STRING: VM Host
...
```

2. Walking Disk read and write I/O statistics:

```
$ snmpwalk  -v 2c -c public example.company.com .1.3.6.1.4.1.2021.13.15.1.1.12
UCD-DISKIO-MIB::diskIONReadX.1 = Counter64: 11310593921
UCD-DISKIO-MIB::diskIONReadX.2 = Counter64: 334
UCD-DISKIO-MIB::diskIONReadX.3 = Counter64: 334
UCD-DISKIO-MIB::diskIONReadX.4 = Counter64: 865912605
UCD-DISKIO-MIB::diskIONReadX.5 = Counter64: 867599133
UCD-DISKIO-MIB::diskIONReadX.6 = Counter64: 865339677
UCD-DISKIO-MIB::diskIONReadX.7 = Counter64: 11309258752
UCD-DISKIO-MIB::diskIONReadX.8 = Counter64: 0
UCD-DISKIO-MIB::diskIONReadX.9 = Counter64: 1822880256
seb-laptop:~$ snmpwalk  -v 2c -c public example.company.com .1.3.6.1.4.1.2021.1
3.15.1.1.13
UCD-DISKIO-MIB::diskIONWrittenX.1 = Counter64: 22337830400
UCD-DISKIO-MIB::diskIONWrittenX.2 = Counter64: 0
UCD-DISKIO-MIB::diskIONWrittenX.3 = Counter64: 0
UCD-DISKIO-MIB::diskIONWrittenX.4 = Counter64: 45118203392
UCD-DISKIO-MIB::diskIONWrittenX.5 = Counter64: 45137660928
UCD-DISKIO-MIB::diskIONWrittenX.6 = Counter64: 45139064320
UCD-DISKIO-MIB::diskIONWrittenX.7 = Counter64: 22337830400
UCD-DISKIO-MIB::diskIONWrittenX.8 = Counter64: 0
UCD-DISKIO-MIB::diskIONWrittenX.9 = Counter64: 33023515648
```

3. Retrieving the system uptime:

```
$ snmpget -v 2c -c public example.company.com .1.3.6.1.2.1.1.3.0
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (1453172) 4:02:11.72
```

Viewing action status

This topic describes how to view the status of actions for the Delphix Engine.

To view the status of actions that are currently running on the Delphix Engine, open the **Action sidebar**. To view details of currently-running and completed jobs, open the **Dashboard**.

Action sidebar procedure

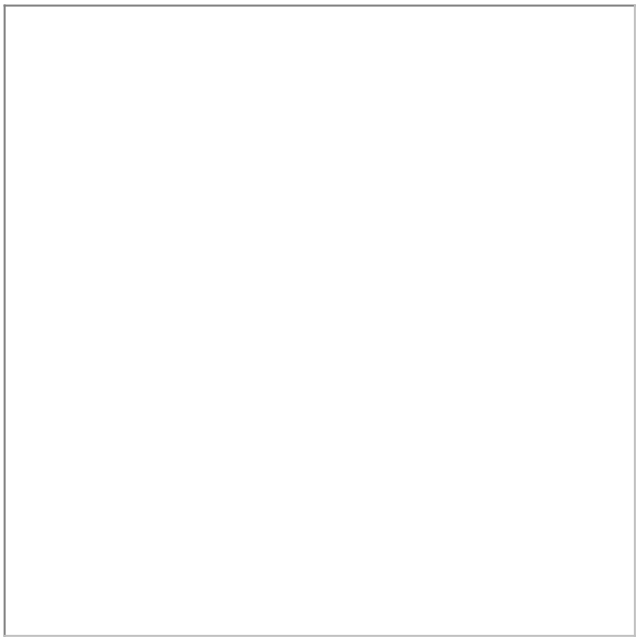
1. Login to the **Delphix Management** or **Delphix Setup** application. Depending on the width of the window, the **Action sidebar** may be automatically displayed on the right of the screen.
2. To see the **Action sidebar**, click **Actions** on the top navigation bar.

Description

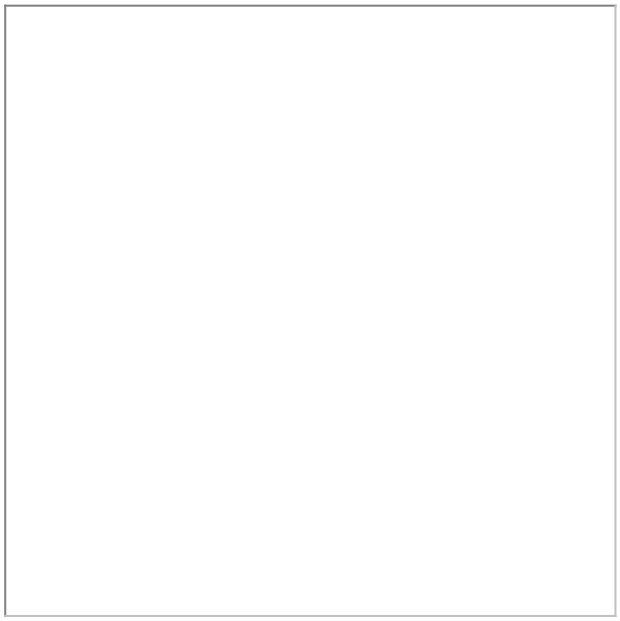
The **Action sidebar** consists of two sections. The top section lists actions that are currently running on the Delphix Engine. The bottom section, labeled **Recently completed**, contains actions that have recently been completed.

Each action is initially collapsed and only presents the title of the action. Click an action to expand it and see more details such as progress, elapsed time, and a description of the operation in progress.

The following is an example of the **Action sidebar** when an Enable action is running.



When an action has been completed it will move down to the bottom of the panel under Recently completed. The following is an example of the **Recently completed** section when a Refresh action is has completed.

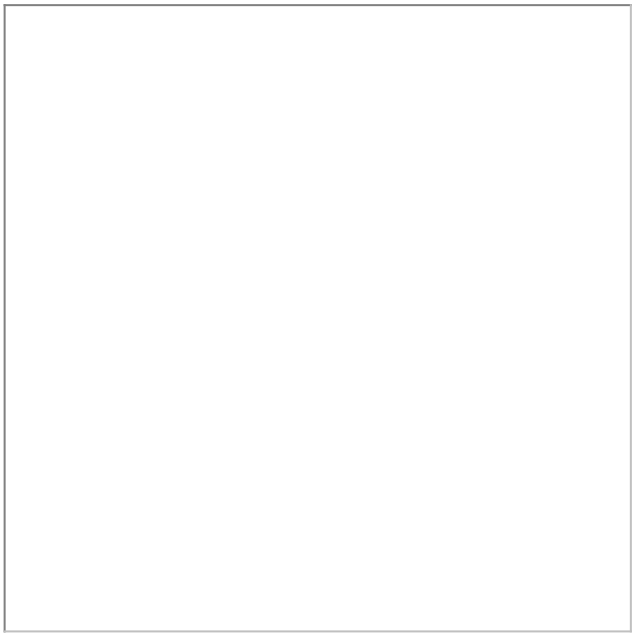


If you are a Delphix Admin or a System Admin, you will be able to see all the actions of your respective application. If you are not an admin user, you will only see actions you have permission to see.

Sub-action

Each action may contain one or several sub-actions which represent the execution of a subset of the action itself. Click an action to see its sub-actions and their respective details. Note that the list of sub-actions is created dynamically during the execution of the action.

The following is an example of an Environment Refresh action and its sub-actions.



Errors

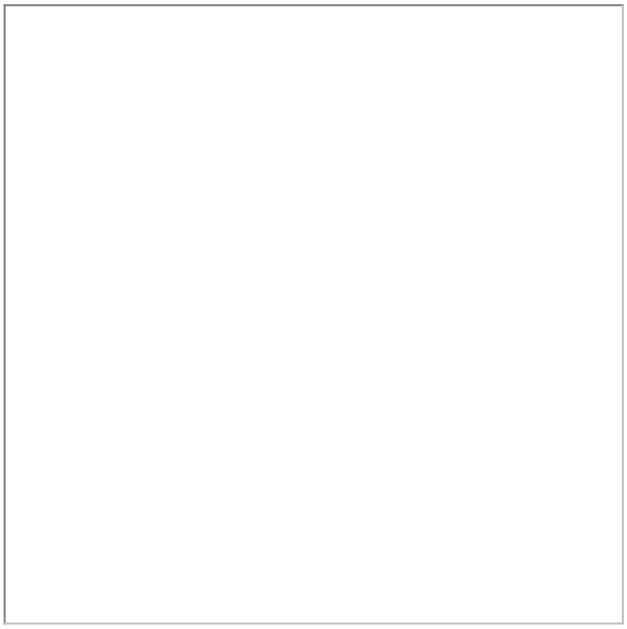
When an error condition occurs during the execution of an action, the background color of the action's box becomes red, and the action remains in the top section until you dismiss it.

- Click the **action** title to expand it. The action will expand to display a description of the error, suggestions to resolve it, and sometimes the raw output of command execution.

To dismiss the action:

- Click the **X** next to the action displaying an error.

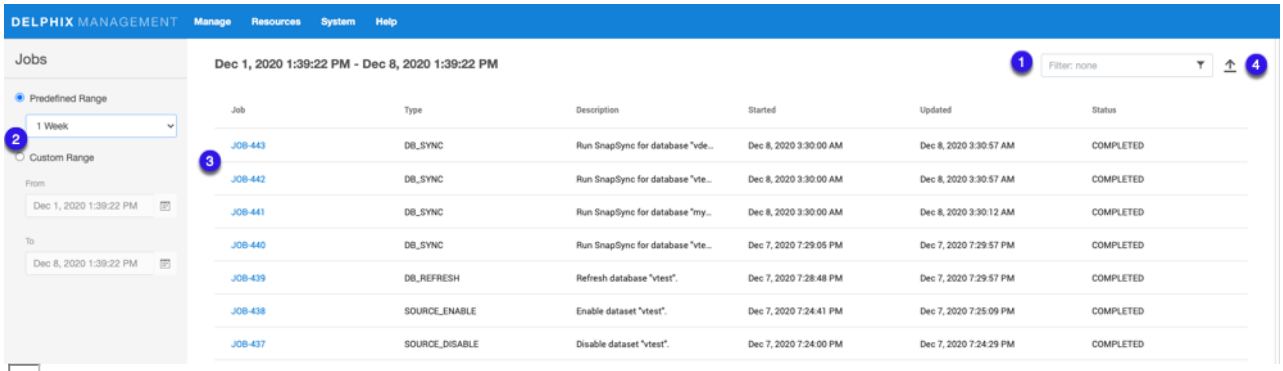
The following is an example of an action failure displayed in the **Action sidebar**.



Viewing jobs

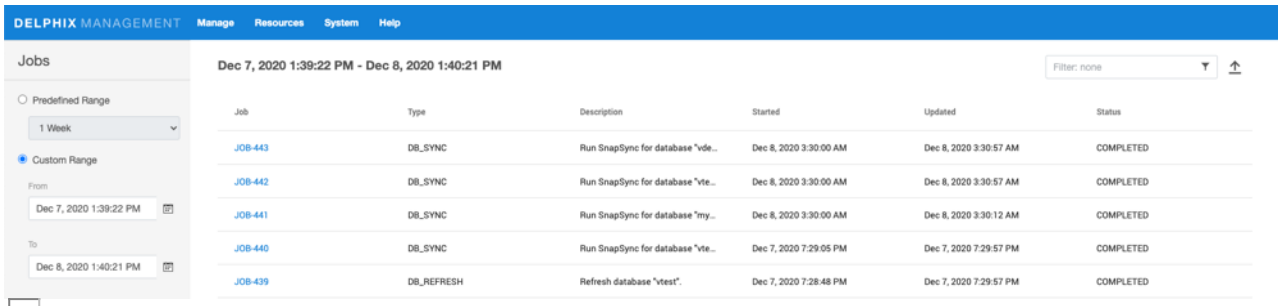
Login to the **Delphix management** application using **admin** credentials. From the **System** menu select **Jobs**.

The **Jobs** panel displays all jobs that have been initiated by the Delphix Engine, and their status.

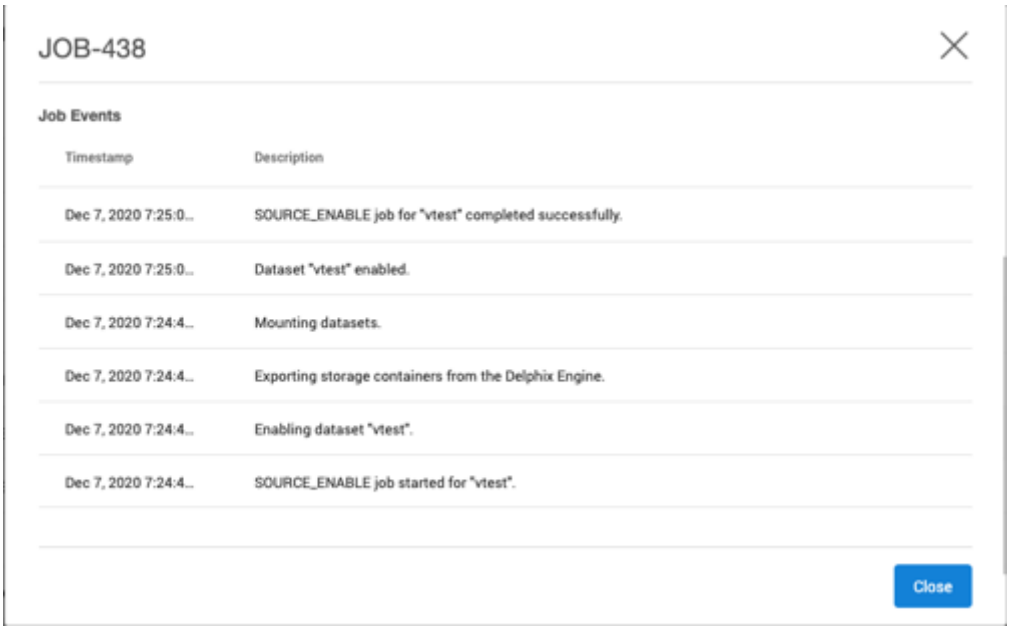


☐ Enter filter text to reduce the results to only those rows matching the text entered.

☐ Customize results by selecting from a Predefined Range or from a Custom Range.



☐ Select a job to view more details.



☐ Click the **Export** button to download the current page of results to a file of comma-separated values (CSV).

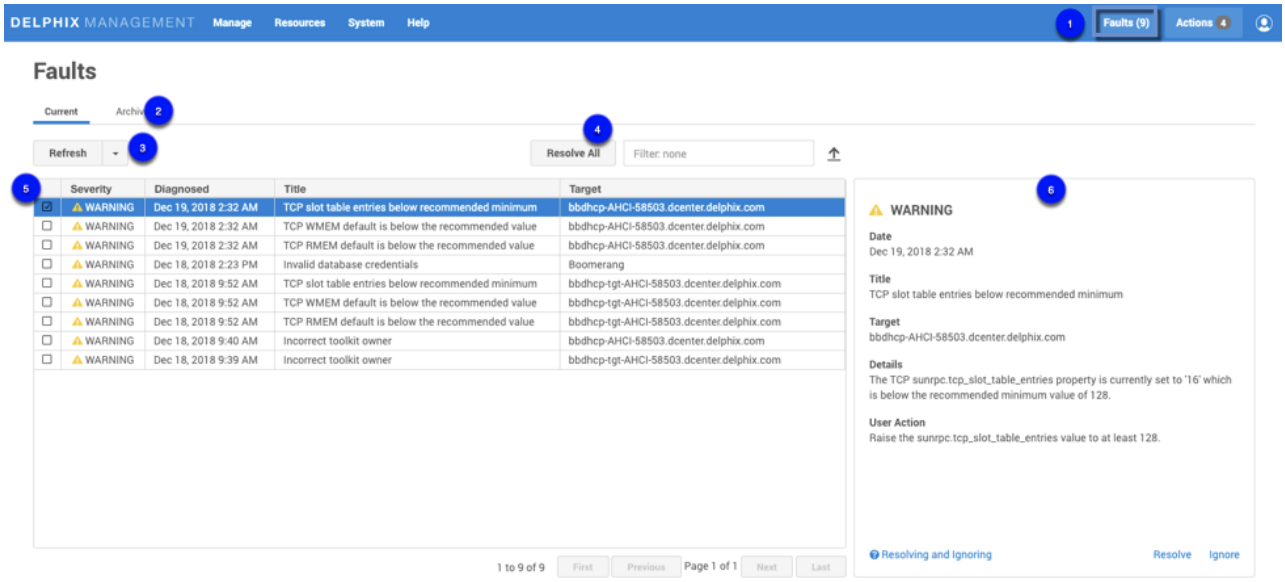
Job Retention
The Jobs panel retains records of the most recent 10,000 jobs that have been initiated by the Delphix Engine. Delphix administrators may use the **job/retention** CLI to retain a larger number of jobs.

System faults

Overview

The Faults screen provides information on states and configurations that may negatively impact the functionality of the Delphix Engine and which can only be resolved through active user intervention. When you login to the **Delphix management** application as admin, the number of outstanding system faults appears on the right-hand side of the navigation bar at the top of the screen. Faults serve as a record of all issues impacting the Delphix Engine and can never be deleted. However, ignored and resolved faults are not displayed in the faults list.

The **Faults** screen as shown below has two tabs, **Current** and **Archive**.



System Faults screen

1. The number of system faults.

2. The **Faults** screen has two tabs, **Current** and **Archive**. Details of the selected fault are displayed on a card located to the right of the fault list. In the **Archive** tab, you can switch between **Resolved** or **Ignored** faults and reset all ignored faults.

3. Selecting **Refresh (Manual),** will refresh the faults table manually, or you can select one of the other available options from the drop-down menu, available options include; Manual, 1 Second, 1 Minute, 5 Minutes.

(Note: as there is no longer a Refresh button on the screen, you must select Refresh (Manual) to refresh the screen.

4. To search the **Faults** table, enter the name of the object you are looking for. The grid will refresh to display the selected object. You can also sort using the column headings.

Resolve All will resolve all the faults in your system.

Select to expand or close the objects in the grid.

Select to export the information provided in the grid to a.csv file.

5. You can select and resolve multiple faults; the card panel will display how many of each type are selected. For example:

There are 2 WARNING items selected

There are 0 CRITICAL items selected

6. Details for the selected fault are displayed on a card. You can resolve or ignore faults by selecting the appropriate link at the bottom of the card.

Resolving and Ignoring Faults

Ignoring a fault will also ignore future faults of that exact type against the same object, so that future fault conditions will not be re-diagnosed even if the fault condition persists or recurs. No further notifications will be received for that specific fault condition. It is advisable to only ignore faults when the following criteria are met:

- The fault is caused by a well-understood issue that cannot be changed.
- Its impact on the Delphix Engine is well understood and does not require action.

For example, if you think that knowing about this error in the future will be important, use "Resolve" rather than "Ignore". If you reset ignored items, this clears all ignored faults, but it leaves them as resolved and does not restore the actual fault. For reset ignored faults, new faults against the same object will no longer be ignored and you will again receive notifications. Examples: If you ignore a fault "Unable to ping host" for target "192.168.1.1", Delphix ignores "Unable to ping host" errors against target 192.168.1.1. You will never see the "Unable to ping host" fault again for that target 192.168.1.1 unless you reset ignored items. Similarly, some faults are raised against snapshots which are part of a dSource. Ignoring those errors only ignores similar errors for that exact snapshot. Tomorrow's snapshot could produce the fault again.

Delphix object-based environment monitor faults

Delphix now has a self-contained Java-based discovery infrastructure that consolidates with environmental monitoring, communicates via a common framework, and is able to provide feedback.

The environment monitor previously only created faults for "hosts" and "sources." There are several faults that more logically apply to other Delphix objects, such as repositories, which are DB install files. Posting them against sources results in fault duplication. The environment monitor now posts faults against -- and re-associates the offending faults with -- the correct objects. Consequently, users see fewer errors that are easier to diagnose.

Viewing faults

To view the list of active system faults:

1. In the top navigation bar, click **System** then **Faults**.
2. In the Faults screen, click any fault in the list to expand it and see its details.
   The details for the selected Fault will be displayed in the details card located on the right.

Each fault comprises six parts:

- **Severity** – How much of an impact the fault will have on the system. A fault can have a severity of either **Warning** or **Critical**.
  - A **Warning Fault** implies that the system can continue despite the fault but may not perform optimally in all scenarios.
  - A **Critical Fault** describes an issue that breaks certain functionality and must be resolved before some or all functions of the Delphix Engine can be performed.
- **Date** – The date that the Delphix Engine diagnosed the fault.
- **Title** – A short descriptive summary of the fault

- **Target** – The object against which the fault was posted. Faults will be posted against the host for incorrect environment configurations, sources for problems with the database, and repositories for issues with the installation.
- **Details** – A detailed summary of the cause of the fault
- **User action** – The action you can take to resolve the fault

Addressing faults

After viewing a fault and deciding on the appropriate course of action, you can address the fault through the user interface (UI). You can mark a fault as **Ignored** or **Resolved**. If you have fixed the underlying cause of the fault, mark it as **Resolved**. Note that if the fault condition persists, it will be detected in the future and re-diagnosed. You can mark the fault as **Ignored** if it meets the following criteria:

- The fault is caused by a well-understood issue that cannot be changed
- Its impact on the Delphix Engine is well understood and acceptable

In this case, the fault will not be re-diagnosed even if the fault condition persists. You will receive no further notifications.

To address a fault follow the steps below.

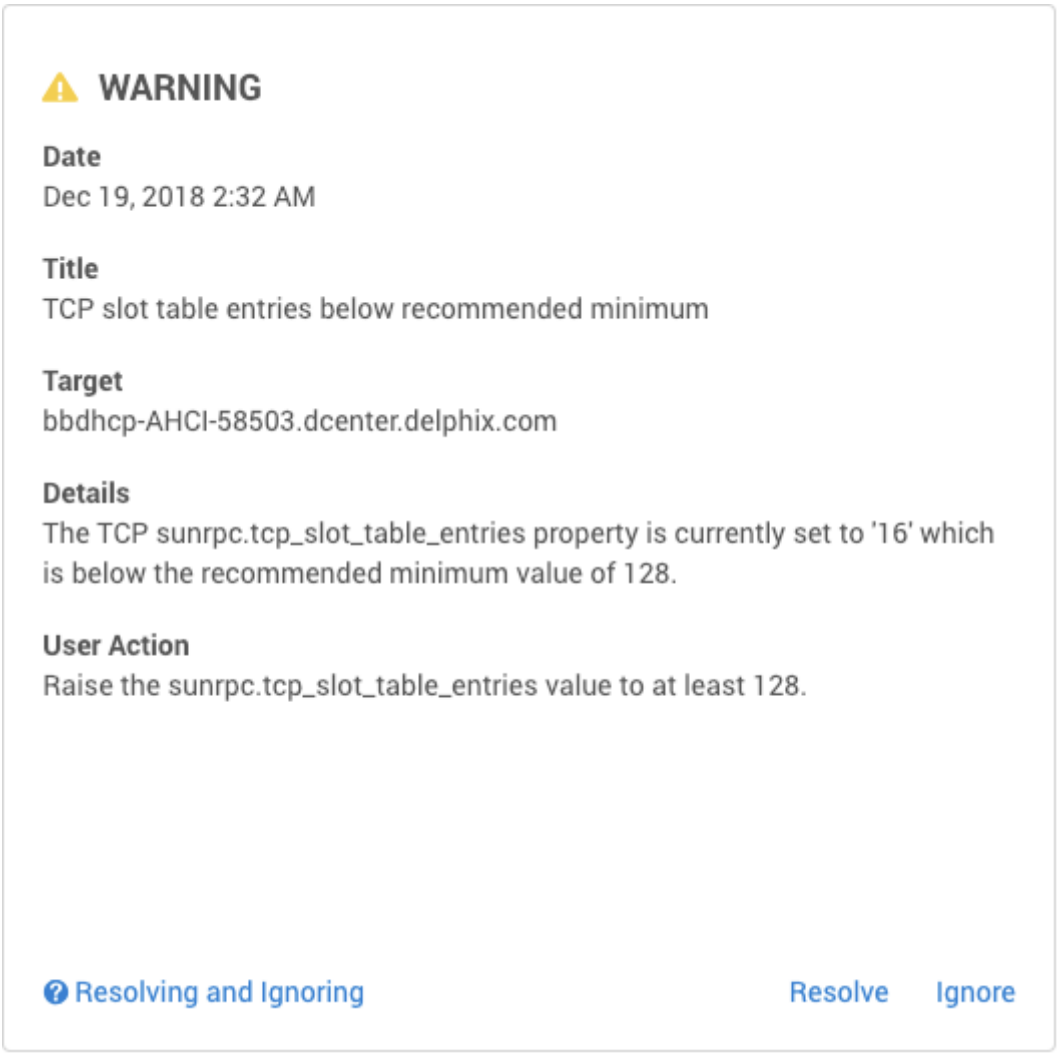
1. In the top menu bar, click **Faults**.
2. In the list of faults, click a **fault date/name** to view the fault details.
3. If the fault condition has been resolved, click **Resolve**.
   Note that if the fault condition persists it will be detected in the future and re-diagnosed.
4. If the fault condition describes a configuration with a well-understood impact on the Delphix Engine that cannot be changed, you can ignore the fault by clicking **Ignore**.
   Note that an ignored fault will not be diagnosed again even if the underlying condition persists.

By default, when a **critical or warning fault** occurs, the Delphix Engine immediately sends an email to the Engine Administrator (**admin**). Make sure you have configured an SMTP server and defined an appropriate email address for Engine Administrator (**admin**). See Initial Setup for more information.

> **Critical or warning alert emails**
> By default, emails will also be sent for **critical or warning alerts** (aka events). You can modify the default behavior by changing the alert profile with the CLI. See the CLI Cookbook Creating Alert Profiles for more information.

Fault lifecycle example

Below is an image of the fault card for the fault "TCP slot table entries below the recommended minimum."

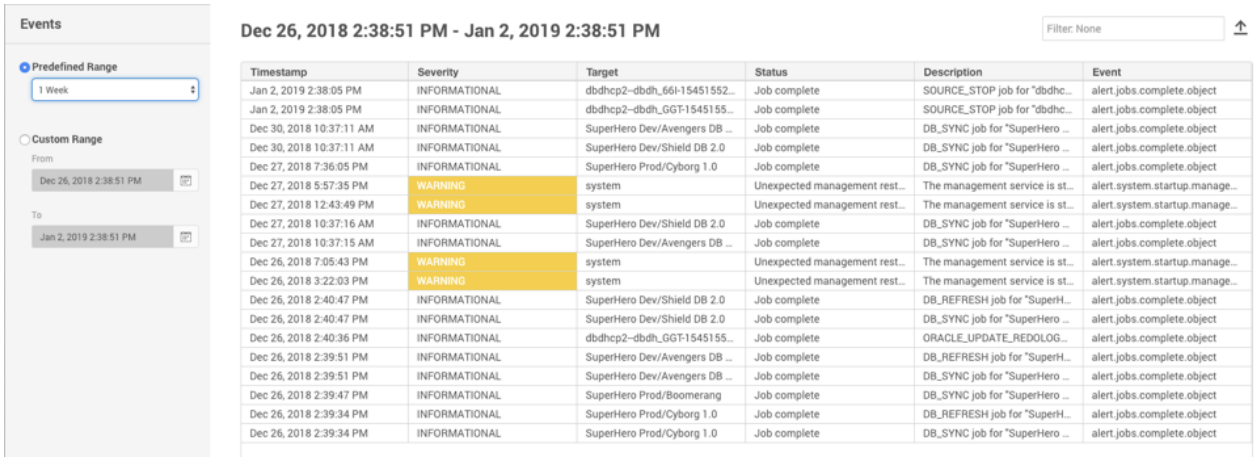


The **Details** section of the fault explains that the sunrpc.tcp_slot_table_entries property is set to a value that is below the recommended minimum of 128. The **User action** section instructs you to adjust the value of the sunrpc.tcp_slot_table_entries property upward to the recommended minimum. The process for adjusting this property differs between operating systems. To resolve the underlying issue, search "how to adjust sunrpc.tcp_slot_table_entries" using a search engine and find that the second result is a link to the Delphix community forum describing how to resolve this issue. After following the instructions applicable to your operating system, return to the Delphix UI and mark the fault **resolved**.

Viewing system faults

System events overview

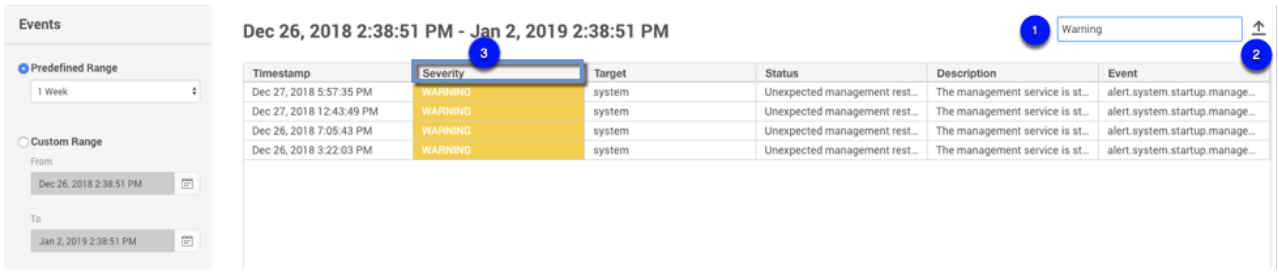
The event log interface has been improved to provide filtering, sorting, and exporting.



Event Viewer screen

As shown above, the **Event Viewer** window provides information about all the events that occurred for the selected time period. Text matching is limited to the following columns:

- Action
- Description



Event Viewer screen filtered for warning events

In the **Event Viewer** window, you can:

1. Enter filter text to reduce the results to only those rows matching the text entered. In the example above, we are filtering for “warning.”
2. Click the **Export** button to export your results to a .csv file.
3. Click a column header to sort rows by the values found in that column.

The first time you click a header, rows will sort in ascending order. Clicking the same header a second time will sort the rows in descending order. Clicking the same header a third time will restore the results to their default sort order.

Procedure

1. Launch the **Delphix Management** application.
2. Click **System**.
3. Select **Events**.
4. Select a time range.

Sorting and filtering

Optional: You can enter filter text to reduce the results to only those rows matching the text entered.

Text matching is limited to the following columns:

- Severity
- Status
- Description

You can click on a table column header to sort rows by the values found in that column.

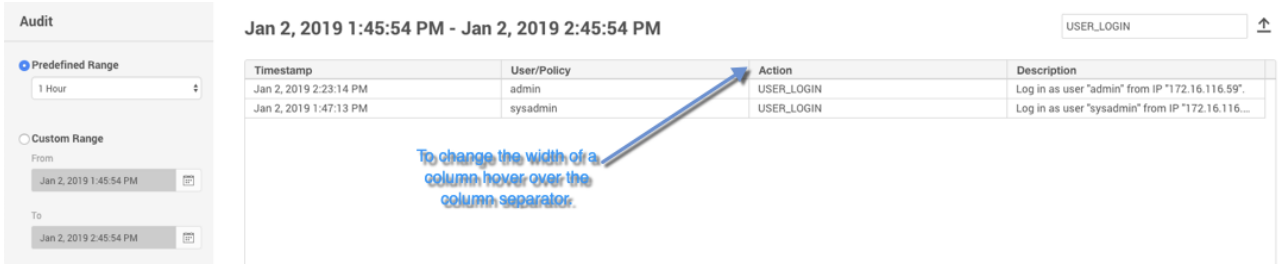
The first time you click a header, rows will sort in ascending order. Clicking the same header a second time will sort the rows in descending order. Clicking the same header a third time will restore the results to their default sort order.

Click the page navigation buttons to advance through large result sets.

Column resizing and tooltips

If you wish, you can resize column widths to better fit the data to the available screen space. To resize a column:

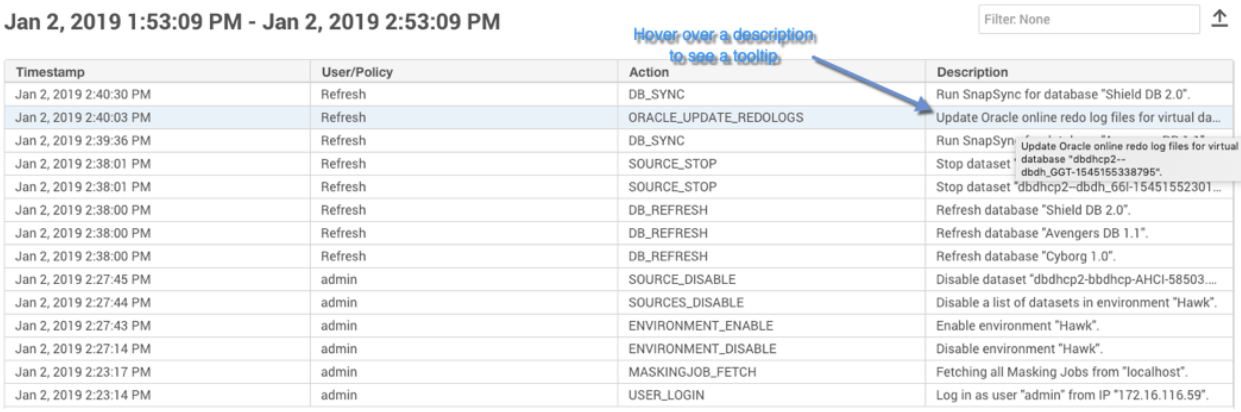
1. Hover the mouse over a column separator found in the header. This will cause the mouse pointer to change shape.
2. Click and drag and the column separator to the desired position. Dragging to the left will reduce the column width. Dragging to the right will increase the width.
3. Release the mouse button.



Alternatively, you can auto-size a column to fit the widest value of the current page:

1. Hover the mouse over a column separator found in the header. This will cause the mouse pointer to change shape.
2. Double click the column separator.

Values that do not fit within their column will be truncated with an ellipses (...). Hover the mouse over any value to see a tooltip rendering the complete, non-truncated value.



Exporting results

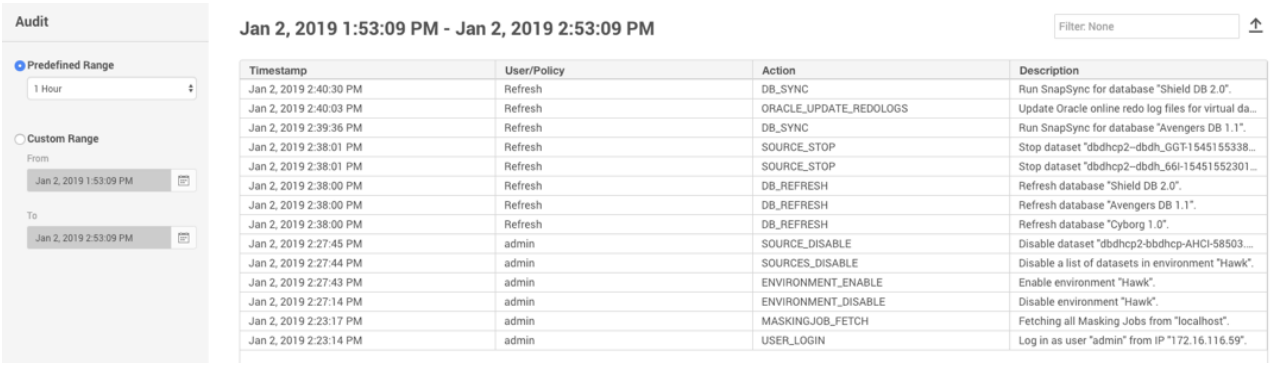
Click the **Export** button to download the current page of results to a file of comma-separated values (CSV).

Accessing audit logs

This topic describes how to access audit logs. The audit log provides a record of all actions that were initiated by a policy or user, regardless of whether that action was successful.

Overview

Audit logs provide a record of all actions that were initiated by a policy or user, regardless of whether that action was successful. The audit log interface has been improved to provide filtering, sorting, and exporting.

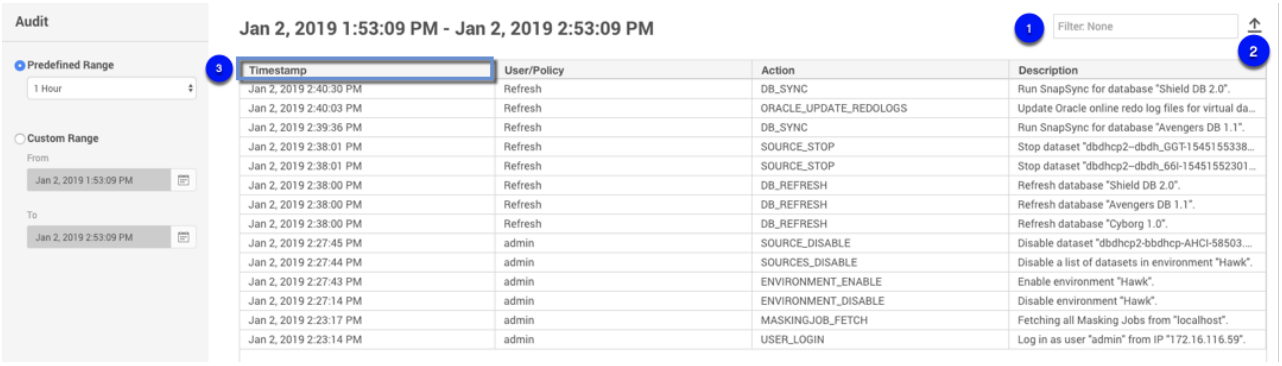


Audit window

As shown above, the **Audit** window displays all actions that were initiated for the selected period of time. You can enter filter text to reduce the results to only those rows matching the text entered. In the figure below, we are filtering for “user.”

Text matching is limited to the following columns:

- Action
- Description



Audit window filtered for User events

Enter filter text to reduce the results to only those rows matching the text entered. In the example above, we are filtering for “user.”

Click the **Export** icon to export your results to a .csv file.

You can click a column header to sort rows by the values found in that column.

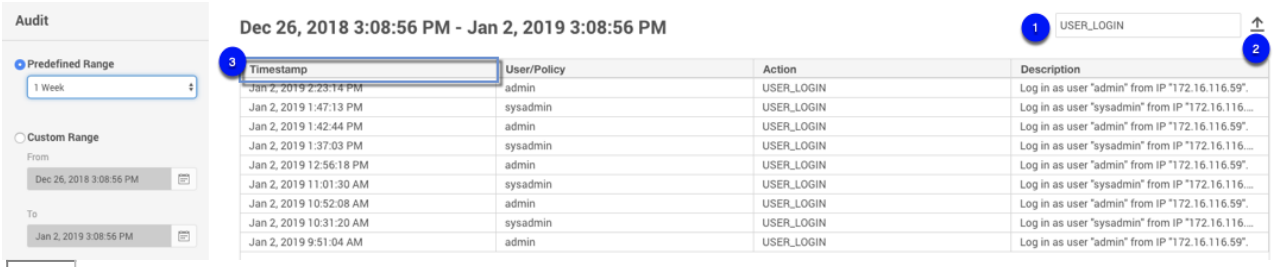
The first time you click a header, rows will sort in ascending order. Clicking the same header a second time will sort the rows in descending order. Clicking the same header a third time will restore the results to their default sort order.

> ⚠ Actions displayed in the Actions panel or on the Audit page are kept forever.

Procedure

1. Login to the **Delphix Management** application using **admin** credentials.
2. Click **System**.
3. Select **Audit**.
4. Select an audit log time range.

Sorting and filtering



Enter filter text (e.g. USER_LOGIN) to reduce the results to only those rows matching the text entered.

Click the **Export** icon to export your results to a .csv file.

You can click on a table column header to sort rows by the values found in that column. The first time you click a header, rows will sort in ascending order. Clicking the same header a second time will sort the rows in descending order. Clicking the same header a third time will restore the results to their default sort order. Text matching is limited to the following columns:

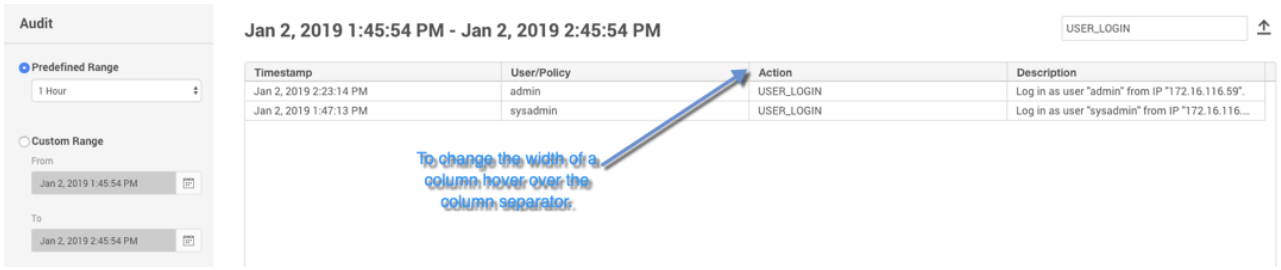
- Action
- Description

Click the page navigation buttons to advance through large result sets.

Column resizing and tooltips

If you wish, you can resize column widths to better fit the data to the available screen space. To resize a column:

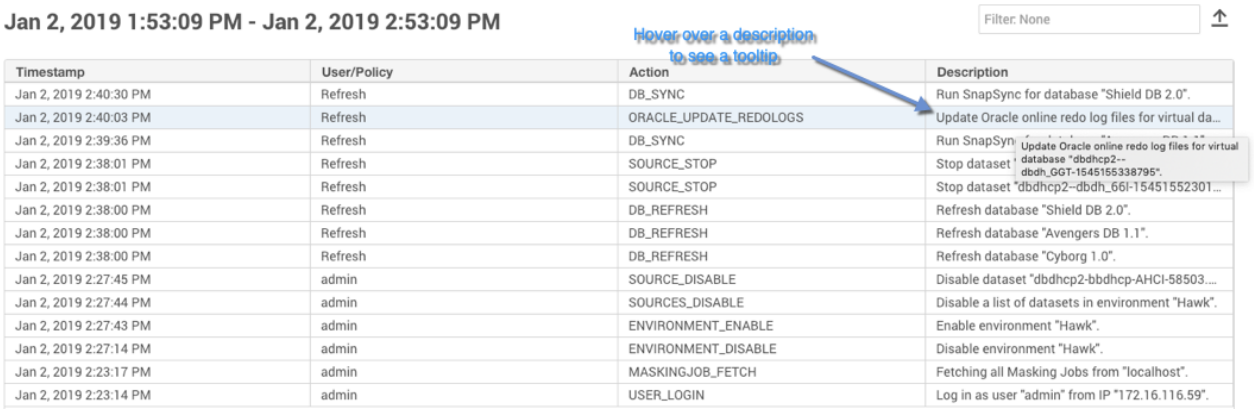
1. Hover the mouse over a column separator found in the header. This will cause the mouse pointer to change shape.
2. Click and drag and the column separator to the desired position. Dragging to the left will reduce the column width. Dragging to the right will increase the width.
3. Release the mouse button.



Alternatively, you can auto-size a column to fit the widest value of the current page:

1. Hover the mouse over a column separator found in the header. This will cause the mouse pointer to change shape.
2. Double click the column separator.

Values that do not fit within their column will be truncated with ellipses (...). Hover the mouse over any value to see a tooltip rendering the complete, non-truncated value.



Exporting results

Click the icon to download the current page of results to a file of comma-separated values (CSV).

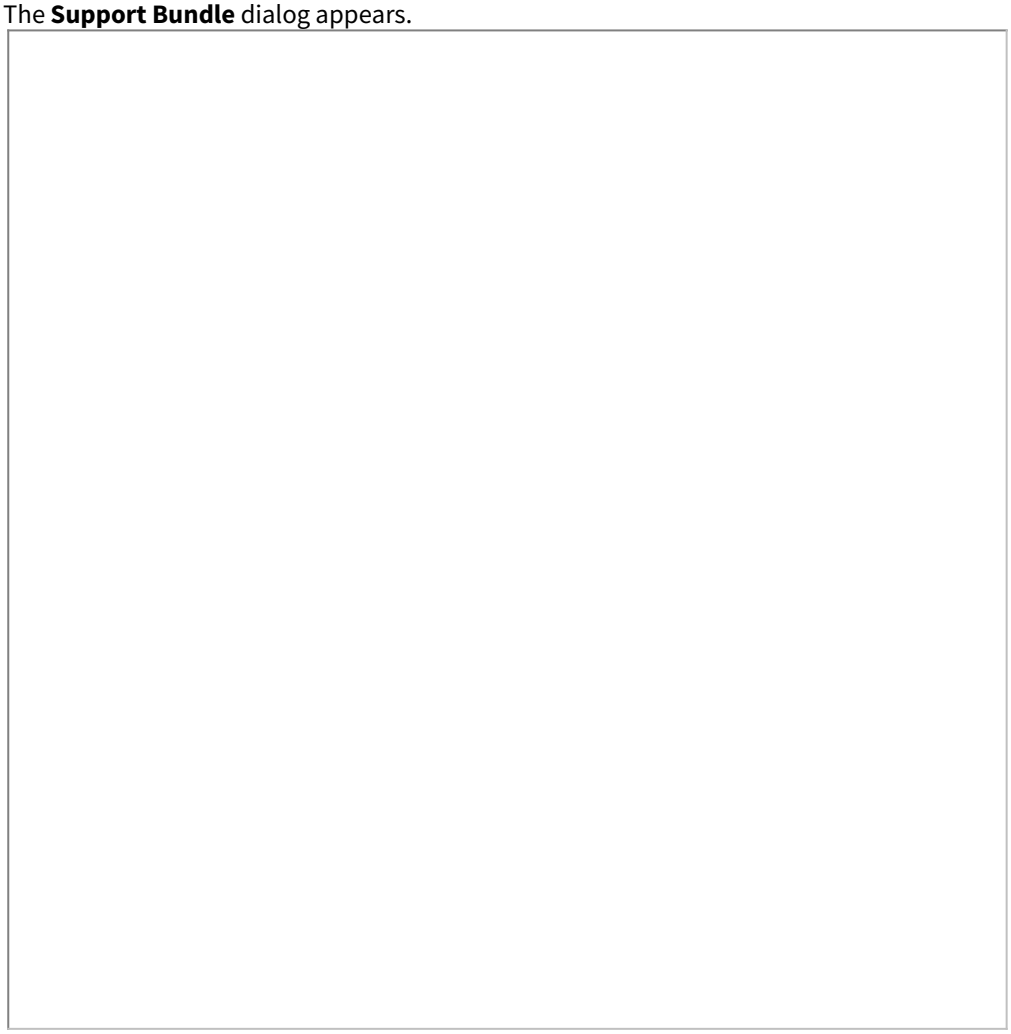
Creating support logs

This topic describes how to create support bundles and manage server access control for Delphix Support.

Support bundles are used by Delphix as diagnostic tools for resolving Delphix Engine issues and routine maintenance. Support bundles can be transferred directly to Delphix Support or downloaded. No customer-specific data is included in the support bundle information. All passwords and personal data are either encrypted or omitted. This is an outbound-only connection from the Delphix Engine.

Using the GUI

1. Log into the **Delphix Management** appliance as an Engine administrator.
2. Click **Help**.
3. Select **Support Logs**.
4. The **Support Bundle** dialog appears.



5. Select **Download** or **Transfer**.
    a. If you select **Download**, then the support logs will be downloaded as a compressed `".tar"` file into a folder on your workstation.
    b. If you select **Transfer**, then the support logs will be uploaded over HTTPS to Delphix Support. If you have configured an HTTP Proxy, it will be used to send the support logs.

   c. If there is a support case involved, then please enter the case number to associate the logs to the case.
6. Click on **Show advanced** and select **Analytics**. This will include all the analytics data (default, up to 10MB) in the Support Bundle.
7. Click **OK**.
   a. If you selected **Download** and have the compressed " `.tar` " file in a folder on your workstation, please **upload** that file to Delphix Support via the website at http://upload.delphix.com.
   b. If there is a support case involved, then please enter the case number (again) to associate the logs to the case.

You can also access support log functionality in the **ServerSetup** application using **sysadmin** credentials.
Click **Support Bundle** in the top menu bar.

Using the CLI

1. ssh into your Delphix Engine.

```
ssh <sysadmin_user>@<delphixengine>
```

2. Run the upload operation.

```
delphix > service
delphix service > support
delphix service support > bundle
delphix service support bundle > upload
```

3. Commit the operation.

```
delphix service support bundle upload *> commit
```

Setting support access control

This topic describes how to set the Support Access Control for Delphix Support. Support access control enables Delphix Support to access your instance of the Delphix Engine for a defined period of time using an access token.

1. Log into the **Delphix Setup** application using **sysadmin** credentials.
2. Click **Server Preferences**.
3. Select **Support Access**.
4. Click **Enable**.
5. Set the time period during which you want to allow Delphix Support to have access to your instance of the Delphix Engine.
6. Click **Generate Token**.
   Provide the token to Delphix Support to enable access to your server.

Setting syslog preferences

Syslog is a widely used standard for message logging. It permits the separation of the software that generates messages, the system that stores them, and the software that reports and analyzes them. Delphix makes use of Syslog as one of the standard mechanisms, along with SNMP and email, to distribute important user and system events, such as alerts, faults, and audits. In the case of Delphix, each Delphix Engine acts as a Syslog client which propagates the events to a centralized Syslog server.

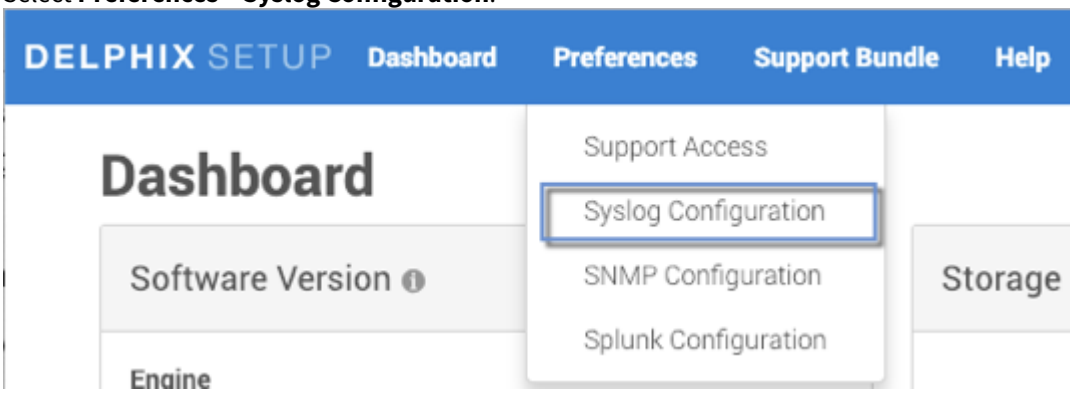
The network protocol over which the Delphix Engine communicates with the Syslog server is standardized in RFC 5424. As a protocol, it supports using either UDP (RFC 5426) or TCP (RFC 6587) as the underlying transport and optional TLS mapping has been introduced to encrypt the messages over the wire for security purposes (RFC 5425). However, as of this release, we only support Syslog over UDP with no encryption, which implies that Syslog messages are always sent in the clear and may be lost during transmission and delivered out of order due to the limitations of UDP.

To configure for Syslog support, you must specify the communication endpoint to which the Syslog server listens, which includes the hostname or IP address of the Syslog server and an optional port number. The latter defaults to 514 according to the Syslog standard but it can be changed if necessary.

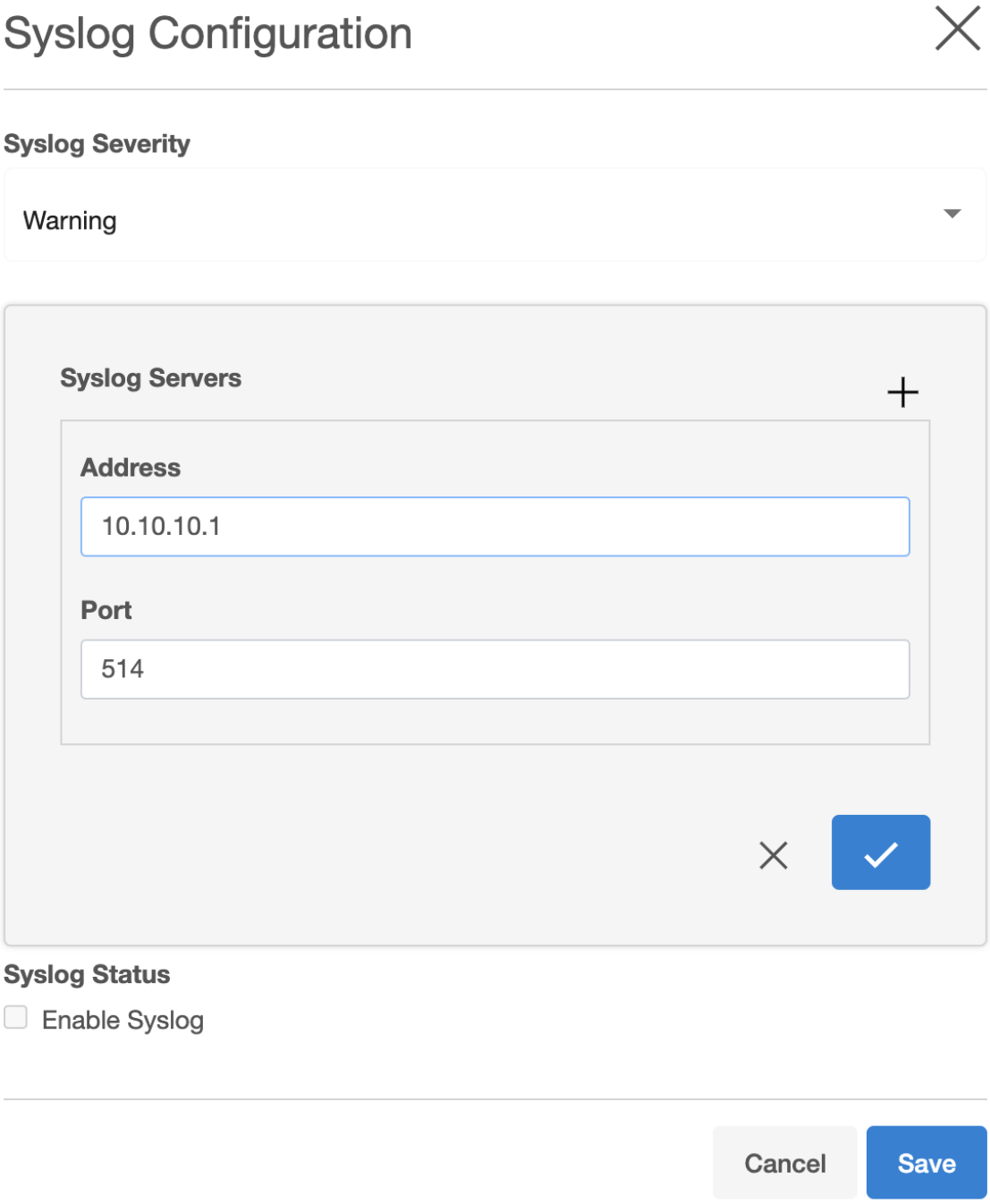
System and user events generated by Delphix are always forwarded immediately to the Syslog server, which ensures the timely delivery of important events that may require immediate action.

A couple of different output formats are supported for messages delivered over Syslog, namely, TEXT and JSON. The TEXT format is the default. To change the message format, as of this release, you must do so via the CLI.

1. Log into the **Delphix Setup** application using **sysadmin** credentials.
2. Select **Preferences > Syslog Configuration**.



3. Select the severity level of the messages you want to be sent to the Syslog server.
4. Click the **pencil** icon next to **Syslog Servers** and then in the **Syslog Configuration** window select the plus icon.
5. Enter the Syslog server hostname/IP address and port number.



6. Select **Enable Syslog**.
7. Click **Save**.

Severity levels for syslog messages

This topic discusses the Syslog reporting feature of the Delphix Engine, along with severity levels.

Syslog is a widely used standard for message logging. It permits the separation of the software that generates messages, the system that stores them, and the software that reports and analyzes them. Delphix makes use of Syslog as one of the standard mechanisms, along with SNMP and email, to distribute important user and system events, such as alerts, faults, and audits. In the case of Delphix, each Delphix Engine acts as a Syslog client which propagates the events to a centralized Syslog server.

Every Syslog message is attached to a severity level. As the name suggests, the severity level describes the severity of the event in question.

Audit Logs

Audit records are Informational Syslog messages. If you would like to forward Audit records, choose Severity Level Informational.

Severity levels

Every Syslog message is attached to a severity level number. Delphix defines the severity of Syslog messages in accordance with RFC 3164. There are eight severity levels available, as follows:

```
Numerical         Severity
Code

           0       Emergency: system is unusable
           1       Alert: action must be taken immediately
           2       Critical: critical conditions
           3       Error: error conditions
           4       Warning: warning conditions
           5       Notice: normal but significant condition
           6       Informational: informational messages
           7       Debug: debug-level messages
```

When setting up the Syslog settings for your Delphix Engine, you have the ability to choose what alerts to report. The severity levels above are available for users to select. Once you select a severity level, the Delphix Engine will send messages of the same or higher severity (i.e., the same or lower number) to your Syslog server. Therefore, there is no reason to select more than one severity. For example, if the "Notice" severity level is selected, all events less severe than Notice (Informational and Debug) will not be reported. If you want all events to be reported via Syslog, the Debug severity level should be chosen.

Support access audit logs

This topic describes how terminal session audit logging works within the Delphix OS. These logs contain keystroke by keystroke recordings of all terminal activity during a given shell session initiated by a super user (Delphix support).

Overview

Super user activity by Delphix support is recorded to an individual log file for each shell session. Each log file is named using the format **session_<shell user ip>_<epoch timestamp>**. The contents of the logs include commands entered into the shell and the output of those commands. Timestamps are additionally prepended to each line of the log to facilitate assessing the timeline of events.

> Session logs created during a super user shell session are kept forever unless deleted by a Delphix engine SYSTEM user.

Listing the session audit logs

Super user session logs can be reviewed/deleted through the CLI or API and downloaded through the API only. Any DOMAIN or SYSTEM user can list the current logs or download a given log file but only SYSTEM users can delete them. To review a list of the session logs currently present on a Delphix engine:

1. Login to the **Delphix CLI** using **admin** or **sysadmin** credentials.
2. Navigate to **superuser session** and press enter.
3. Use **list** or **ls** to view the files.

```
ip-12-345-678-90 superuser session> ls
Objects
NAME                                  IPADDRESS      STARTTIMEUTC
DURATION
session_123.45.678.90_1686923517171  123.45.678.90  2023-06-16T13:51:57.171Z  20sec
session_123.45.678.90_1686923559856  123.45.678.90  2023-06-16T13:52:39.856Z  439sec
session_123.45.678.90_1686924008788  123.45.678.90  2023-06-16T14:00:08.788Z  87sec
```

Here is an example of calling the list API directly using curl:

```
curl -b ~/cookies.txt -X GET "http://mydelphixengine.myorg.com/resources/json/
delphix/superuser/session"
```

Downloading a session audit log

Any DOMAIN or SYSTEM user can download a super user session log file via the Delphix API. Here is an example of calling the download API using curl:

```
curl -v -O -J "http://mydelphixengine.myorg.com/resources/json/delphix/superuser/
session/download?sessionLogName=session_123.45.678.90_1686923517171" -b ~/cookies.txt
```

Reviewing a session audit log

It is recommended that session logs be viewed through a program such as **cat**, which is capable of interpreting control characters. This is because the logs not only include key strokes and terminal output, but also the control characters that dictate how that output was formatted and displayed, ensuring that the logs reflect what was actually seen during the shell session as accurately as possible. It is also possible to view the logs using any text editor, but in most cases this will be more difficult to read because the control characters themselves will be visible. Here is a snippet from a brief session log as it might be displayed by **cat**:

```
[2023-06-16T13:52:28.061Z] delphix:~$ echo testing 123
[2023-06-16T13:52:28.062Z] testing 123
[2023-06-16T13:52:30.599Z] delphix:~$ exit
```

Note that each line of the log includes a timestamp. This timestamp is prepended to each line as the log is written. It is not from the session terminal output, but rather is provided to more conveniently assess the timeline of a given session. The timestamp is generated in the instant before a given command is executed rather than when the prompt was first printed to the terminal to maximize its accuracy.

Limitations

Shell activity that involves opening a pager or buffer (e.g. **less, more, vi,** etc.) may not be fully reflected in the session log, though the command that initiates the pager/buffer will be present. For example, if a super user opens a file in **vi** for editing, the line to open the file would be present, followed by the next command run after **vi** was closed.

Deleting a session audit log

These audit logs are meant to live as long as they are needed, and thus are not governed by a retention policy. Should you wish to delete a log this can be done by SYSTEM users only through the Delphix CLI or API. To use the CLI:

1. Login to the **Delphix CLI** using **sysadmin** credentials.
2. Navigate to **superuser session** and press enter.
3. Review the current log files using **list** or **ls**
4. Select the log file that you wish to delete.
5. Use **list** or **ls** to review the log details and confirm this is the log you want to delete.
6. Type **delete** and press enter.
7. Type **commit** and press enter to delete the log.

```
ip-12-345-678-90 superuser session> select "session_123.45.678.90_1686923517171"
ip-12-345-678-90 superuser session 'session_123.45.678.90_1686923517171'> ls
Properties
    type: SuperuserSession
    name: session_123.45.678.90_1686923517171
    duration: 20sec
    ipAddress: 123.45.678.90
    reference: SUPERUSER_SESSION-session_123.45.678.90_1686923517171
    startTimeUTC: 2023-06-16T13:51:57.171Z

Operations
delete
```

```
ip-12-345-678-90 superuser session 'session_123.45.678.90_1686923517171'> delete
ip-12-345-678-90 superuser session 'session_123.45.678.90_1686923517171' delete *>
commit
ip-12-345-678-90 superuser session> ls
Objects
NAME                                  IPADDRESS      STARTTIMEUTC
DURATION
session_123.45.678.90_1686923559856  123.45.678.90  2023-06-16T13:52:39.856Z  439sec
session_123.45.678.90_1686924008788  123.45.678.90  2023-06-16T14:00:08.788Z  87sec
```

The delete API can also be called directly. Unlike the download API, delete requires the session log reference, which is always SUPERUSER_SESSION-<log name>.Here is an example of calling the delete API using curl:

```
curl -X POST "http://mydelphixengine.myorg.com/resources/json/delphix/superuser/
session/SUPERUSER_SESSION-session_123.45.678.90_1686923517171/delete" -b ~/cookies.tx
t
```

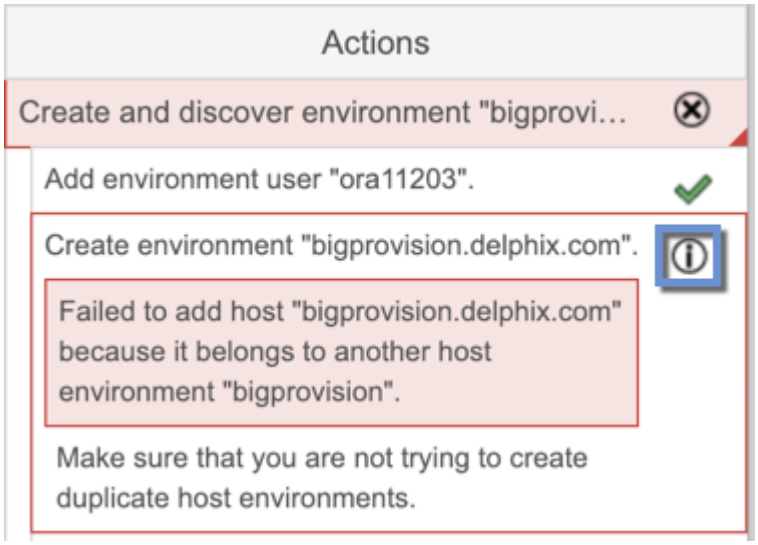
Diagnosing connectivity errors

Prior to the 5.1 release, when the Delphix Engine ran into an error operating on an external database or environment, it reported the immediate error that it had encountered; there was no mechanism for automatic analysis of the root causes of failures. The 5.1 release included infrastructure for automatic diagnosis of errors. When one of there errors occurs, the Delphix Engine now launches a set of tests to locate the root cause of the problem and present the result of the diagnosis. This will help you easily identify the true sources of errors such as closed ports or misconfigured router.

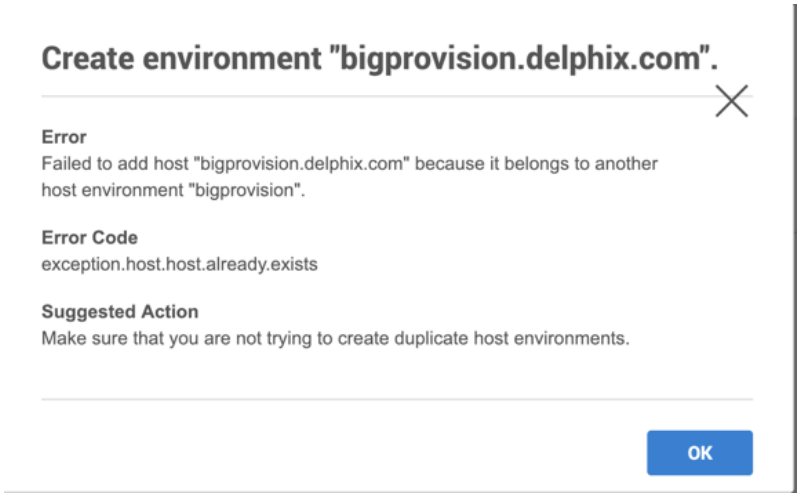
Failed actions

The Delphix Engine communicates failures in two different manners: actions that fail to complete, and faults. To view failed actions:

1. In the top right-hand corner of the Delphix Management application, click **Actions**.
2. For more information about why the action failed, click the (**i**) icon to show the error dialog.



The following shows a popup message with more information about the problem and what actions to take to resolve it. For some errors, the Delphix Engine will be able to diagnose the problem further and display this extra information under **Diagnosing Information**. In the screenshot above, the job failed because the Delphix Engine was unable to lookup the host address.



Viewing active faults

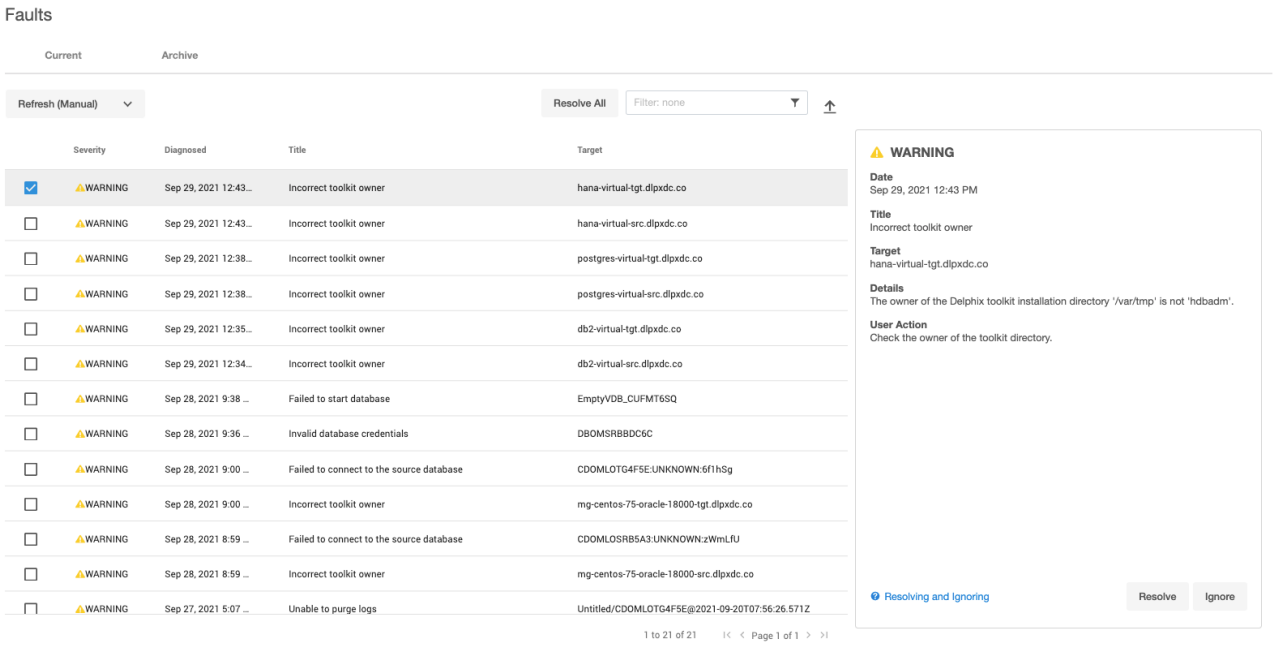
A fault symbolizes a condition that can affect the performance or functionality of the Delphix Engine and must be addressed. Faults can be either warnings or critical failures that prevent the Delphix Engine from functioning normally. For example, a problem with a source or target environment can cause SnapSync or LogSync policy jobs to fail. Faults will show up as active as long as:

- The error is still occurring, or
- You have chosen to manually resolve it or ignore it

For example, if a background job fails, it will create a fault that describes the problem. To view any active faults:

- In the top right-hand corner of the Delphix Management interface, click **Faults**.

This brings the Faults screen listing all active faults.



The screenshot above illustrates a fault with regard to a failure to a TCP slot table entry. The Delphix Engine will mark an object with a warning triangle to indicate that it is affected by an external problem. You can view more details of the fault by looking at the active faults and their fault effects.

Email (SMTP) alert notifications

Overview

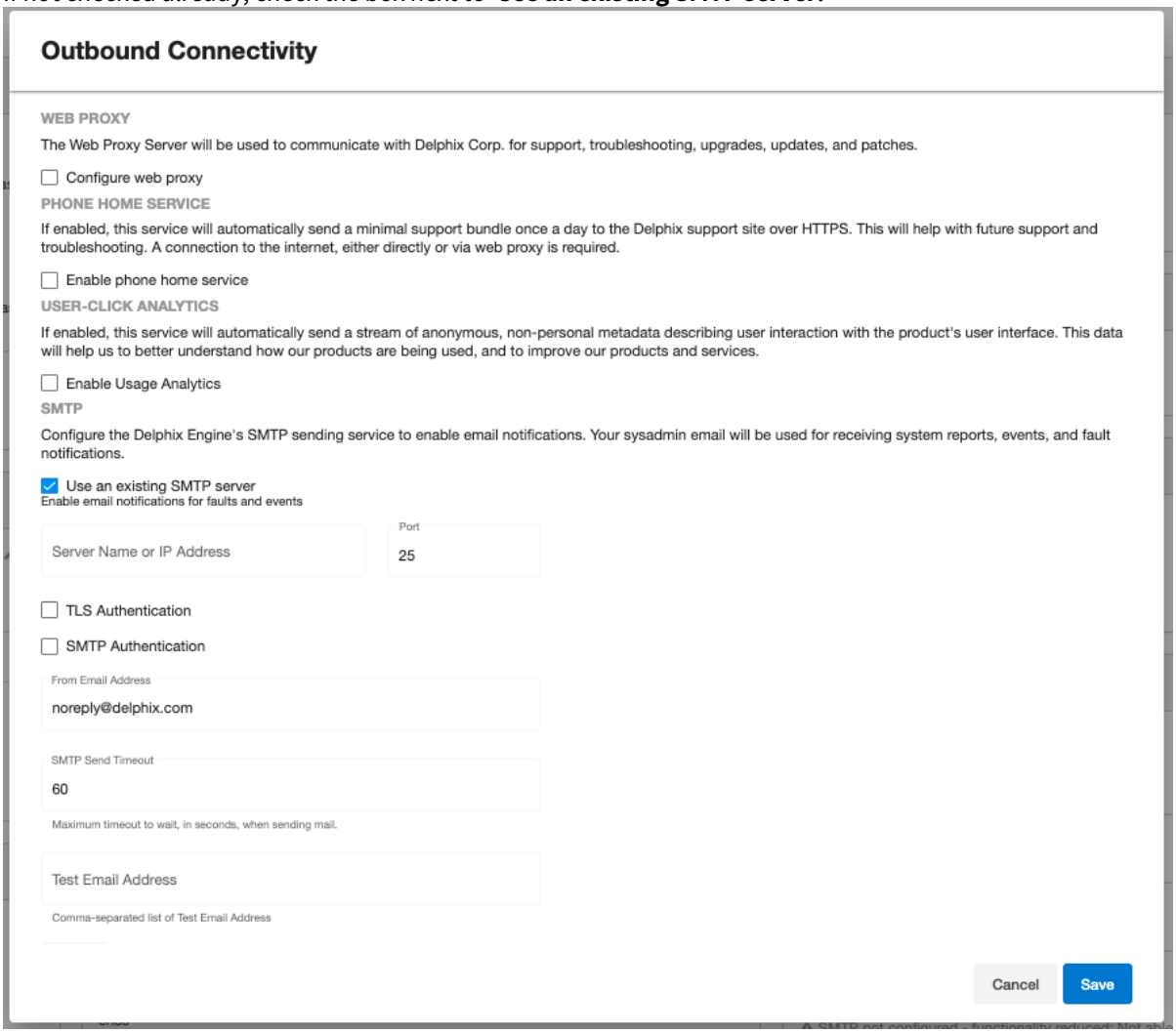
The configuration for SMTP-based alert notifications has two components:

- The configuration of an SMTP gateway by the Delphix system administrator
- The configuration of one or more alert profiles (if needed)

Configuring the SMTP gateway

Before email-based alerts can function properly, many organizations require that an SMTP gateway is configured, through which all outbound email is sent.

1. Contact the appropriate administrator for your site in order to determine the SMTP gateway settings.
2. Login to the **Delphix setup** application as **sysadmin** or another user with system administrator privileges.
3. On the Dashboard screen, locate the **Outbound connectivity** section, and click **modify**. 
Outbound Connectivity for SMTP Gateway
4. If not checked already, check the box next to **Use an existing SMTP server.**



5. At a minimum, enter the required information:
    a. SMTP Server Name of IP Address
    b. SMTP port
    c. From Email Address This will be the email address from which all alert emails will be sent.
6. In **Test Email Address**, enter the same email address to verify that you are able to receive email properly.
7. Click **Save** to save changes.

For further information, see the "Outbound Connectivity" section of Initial Setup

Alert profiles

The Delphix Engine can send out email notifications when alerts happen. Alert profiles control this functionality.

An alert profile is composed of two things:

- **Filter Specification:** A filter, or combination of filters, that specifies which alerts are of interest.
- **Alert Action:** This specifies the email addresses to which the Delphix Engine will send an email when an alert matches the filter specification.

By default, the Delphix Engine has a single alert profile configured with the following parameters:

- Filter Specification: Match any alert with a severity level of **CRITICAL** or **WARNING**.
- Alert Actions: Send an email to the address defined for user **admin**.
  Default domain user
  The default domain user created on Delphix Engines is now **admin** instead of delphix_admin. When engines created before 5.3.1 are upgraded to 5.3.1 or later they will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then Disabling delphix_admin.

Using the CLI, it is possible to:

- Modify the system default alert profile
- Create additional profiles in addition to the default one
- Set multiple actions for a single profile, such as "email dephix_admin" *and* "email user1@mycompany.com".

Simple filters

- Filtered by Owner of alerts target – for example, objects owned by user 1

Complex filters

Complex filters combine/modify other sub-filters:

- "And" filter – Used when all conditions defined must be met for the filter to notify the user with an email
- "Or" filter – Used when either one or the other of the conditions defined in the filters must be met for the filter to notify the user with an email
- "Not" filter – Used to exclude items

Limitations

- This is a CLI feature.
- Alert Profiles do not override permission settings. If you do not have Read permission on an object then your alert profile will never get triggered for that object's alerts, regardless of your filter settings.

The following CLI examples will run through how to create these three filters. Each example provides three different methods of setting up a profile. These include the following:

- A simple profile
- A profile with two filters

- A complicated profile

For more information, see CLI Cookbook: Creating Alert Profiles

A simple profile

A simple profile approach matches the Delphix out-of-the-box default alert profiles. To create a simple alert profile using the CLI as seen in the figure below, go into the alert profile section of the command-line interface (CLI) and create a new profile. Line four prompts the engine to send an email when the filters are triggered. The following three command lines refer to the filter specifications. Follow two severity levels: warning and critical. This will trigger an email alert when any warning or critical events occur.

```
delphix > cd alert
delphix alert > cd profile
delphix alert profile > create
delphix alert profile create > set actions.0.type=AlertActionEmailUser
delphix alert profile create > set filterSpec.type=SeverityFilter
delphix alert profile create > set filterSpec.severityLevels.0=CRITICAL
delphix alert profile create > set filterSpec.severityLevels.1=WARNING
delphix alert profile create > commit
```

Simple Alert Profile example in the CLI

A compound alert profile

Creating a compound alert profile in the CLI will combine two filters together. This profile triggers an email about any alert on objects owned by the delphix_admin plus any other alert that is critical. The compound alert profile creates two filters. The first one will be the target owner filter, which in this case is **admin**. The second filter is the severity filter, allowing users to match anything that is critical. Combine these two filters using the OR logic so that if any of the sub-filters match, the whole filter matches. An example of this can be seen in the figure below.

Alert Profile using OR logic

While working in the CLI, the first four lines describe a simple alert profile. The distinction between simple and compound alert profiles is that in a compound profile, the top-level filter uses an OR filter with sub-filters for target owner and severity level, as seen in line five of the figure below.

```
delphix > cd alert
delphix alert > cd profile
delphix alert profile > create
```

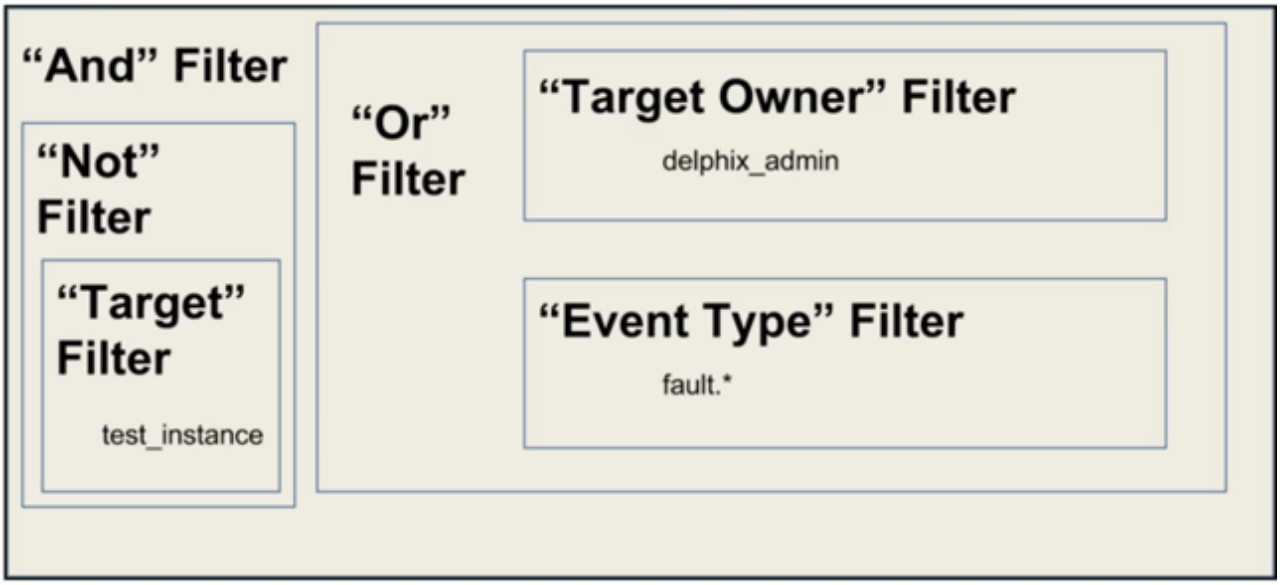
```
delphix alert profile create > set actions.0.type=AlertActionEmailUser
delphix alert profile create > set filterSpec.type=OrFilter
delphix alert profile create > set filterSpec.subFilters.0.type=TargetOwnerFilter
delphix alert profile create > set filterSpec.subFilters.0.owners.0=delphix_admin
delphix alert profile create > set filterSpec.subFilters.1.type=SeverityFilter
delphix alert profile create > set filterSpec.subFilters.1.severityLevels.0=CRITICAL
delphix alert profile create > commit
```

A Compound Alert Profile

Complex alert profile

A complex alert profile uses the profile filter created in the compound alert profile and modifies it. For the example shown in the figure below, you have a VDB named test_instance that you do not need any emails about. The following commands will create an effective filter.

1. Create an OR filter with two sub filters: target owner and event type.
2. Create a NOT filter that will exclude the VDB (test_instance) from which you do not want to receive notifications.
3. Use the AND logic to combine all these filters together, as seen below.



Complex Alert Profile

Below is an example of the command lines used to set up this complex profile.

```
delphix > cd alert
delphix alert > cd profile
delphix alert profile > create
delphix alert profile create > set actions.0.type=AlertActionEmailUser
delphix alert profile create > set filterSpec.type=AndFilter
delphix alert profile create > set filterSpec.subFilters.0.type=NotFilter
delphix alert profile create > edit filterSpec.subFilters.0.subFilter
delphix alert profile create filterSpec.subFilters.0.subFilter > set
type=TargetFilter
delphix alert profile create filterSpec.subFilters.0.subFilter > set targets.0=test_i
nstance
```

```
delphix alert profile create filterSpec.subFilters.0.subFilter > back
delphix alert profile create > set filterSpec.subFilters.1.type=OrFilter
delphix alert profile create > set filterSpec.subFilters.1.subFilters.0.type=TargetOw
nerFilter
delphix alert profile create > set filterSpec.subFilters.1.subFilters.0.owners.0=delp
hix_admin
delphix alert profile create > set filterSpec.subFilters.1.subFilters.1.type=Severity
Filter
delphix alert profile create > set filterSpec.subFilters.1.subFilters.1.severityLevel
s=CRITICAL
delphix alert profile create > commit
```

Complex Alert Profile CLI

Creating alert profiles

1. SSH into your engine's CLI using your delphix_admin username and password

   ```
   ssh delphix_admin@yourdelphixengine
   ```

2. Start creating your new profile. After logging in to get to the alert section enter alert.

   ```
   delphix > alert
   delphix alert > profile
   delphix alert profile > create
   delphix alert profile create > ls
   ```

3. Set Action(s) Use **AlertActionEmailList**if you want to specify a list of email addresses for this profile.

   ```
   delphix alert profile create > set actions.0.type=AlertActionEmailList
   delphix alert profile create > set actions.0.addresses.0=<email address to send
   to>
   delphix alert profile create > set actions.0.addresses.1=<additional email
   address>
   delphix alert profile create > set actions.0.addresses.2=<additional email
   address>
   ```

   Or, use **AlertActionEmailUser** if you just want the emails to go to the email address associated with this Delphix user.

   ```
   delphix alert profile create > set actions.0.type=AlertActionEmailUser
   ```

   It is possible to add more than one action here, so you may use both **AlertActionEmailList** and **AlertActionEmailUser** if desired.

4. Set filter Here is an example of setting a simple severity filter. With this filter, emails will be sent for any **CRITICAL** or **WARNING** alerts.

   ```
   delphix alert profile create > set filterSpec.type=SeverityFilter
   delphix alert profile create > set filterSpec.severityLevels.0=CRITICAL
   ```

```
delphix alert profile create > set filterSpec.severityLevels.1=WARNING
```

Here is an example of setting a simple target-owner filter. With this filter, emails will be sent for any alert whose target is owned by delphix_admin.

```
delphix alert profile create > set filterSpec.type=TargetOwnerFilter
delphix alert profile create > set filterSpec.owners.0=delphix_admin
```

Here is an example of a compound filter. With this filter, we combine the above two filters – an email is sent when an alert is **CRITICAL** or **WARNING**, and the alert's target is owned by delphix_admin.

```
delphix alert profile create > set filterSpec.type=AndFilter
delphix alert profile create > set filterSpec.subFilters.0.type=SeverityFilter
delphix alert profile create > set filterSpec.subFilters.0.severityLevels.0=CRI
TICAL
delphix alert profile create > set filterSpec.subFilters.0.severityLevels.1=WAR
NING
delphix alert profile create > set filterSpec.subFilters.1.type=TargetOwnerFilt
er
delphix alert profile create > set filterSpec.subFilters.1.owners.0=delphix_adm
in
```

5. Commit your changes

```
delphix alert profile create > commit
```

Profile filters

As seen above, you can use different filter types to customize which alerts the Delphix Engine will send emails about. The various filter types are listed below.

Simple filters

Filter type	Purpose	Example	Allowed Values
SeverityFilter	Match based on the alert's severity level (critical, warning, informational)	severityLevels.0=CRITICAL severityLevels.1=WARNING This would match any alert whose severity level is CRITICAL or WARNING.	1 or more of: • CRITICAL • WARNING • INFORMATIONAL
EventFilter	Match based on the alert's event type.	eventTypes.0=fault.* This would match any alert that is generated due to a newly-raised fault on the engine.	One or more text entries, optionally using the * wildcard.

Filter type	Purpose	Example	Allowed Values
TargetFilter	Match based on the alert's target.	targets.0=”Group/DB” This would match any alert whose target is the database “DB” located in the group “Group”.	Any object in the system. 1 or more objects may be specified.
TargetOwnerFilter	Match based on the owner of the alert's target.	owners.0=delphix_admin This would match any alert whose target's owner is the **delphix_admin** user.	Any user in the system. 1 or more users may be specified.

Compound filters

These filters combine/modify the behavior of other filters, called "subfilters". The subfilters may be of any type (simple or complex).

Filter Type	Purpose	Number of subfilters required
AndFilter	This filter matches if **all** subfilters match	2 or more
OrFilters	This filter matches if **any** subfilter matches.	2 or more
NotFilter	This filter matches if the subfilter does **not** match.	1

Action types

With the AlertActionEmailUser type, notification emails will be sent to the email address of the user who owns the alert profile.

With the AlertActionEmailList type, a list of email addresses must be specified in the "addresses" array. Notification emails will be sent to these addresses.

Email format options

You can send plain text as well as structured JSON. JSON can be useful for constructing solutions that will parse the email and perform further actions (notify, escalate, log).

To change the format, while updating an alert profile:

```
delphix alert profile ALERT_PROFILE-X update > set actions.0.format=<JSON|TEXT>
```

Fluentd plugin service for API modules

Overview

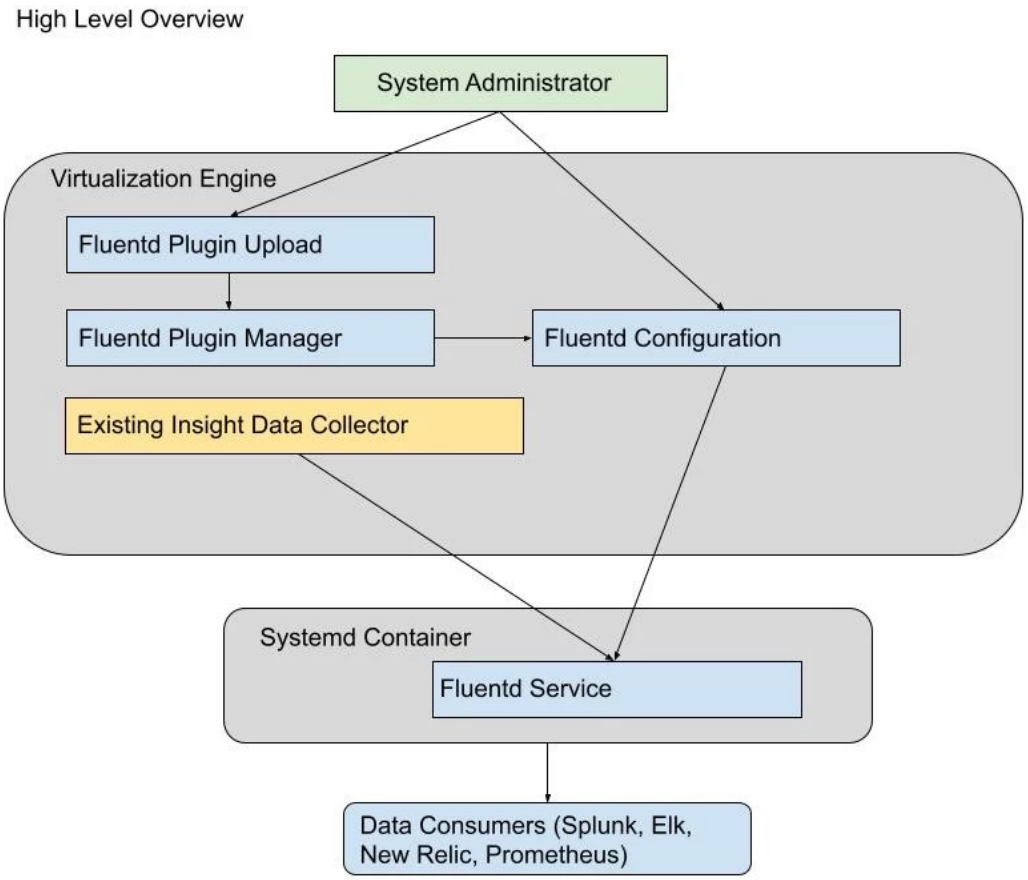
The Delphix Fluentd plugin service assisted in a feature that provided options for configuring a Delphix engine that forwards events and metrics to a Splunk host. Delphix has developed methods to extend this capability for use with other monitoring packages (ELK Stack, Datadog, etc.). Fluentd plugins provide a mechanism to customize the Delphix engine's forwarding capabilities for output to other data consumers.

The Fluentd plugin is a tar file with a configuration template, including any gem files that are required to execute the directives in the configuration file. Once the plugin is uploaded, the configuration occurs by collecting user input for the variables required by the template. The interfaces will have changed since the set of variables needed is now mutable from plugin to plugin. Splunk configuration will be available out of the box with no additional plugins. This feature uses one UI for all Fluentd configurations.

> The current implementation is limited to one data consumer at a time. Only one plugin, in addition to the default SplunkHec plugin, can be uploaded. This is an intended simplification for the initial Fluentd release version.

Technical details

Architectural diagram



GUI, API or CLI changes (if any)

APIs with a corresponding CLI for uploading plugins and Fluentd configuration have been added. The service/insight/plugins API displays the plugins that are available.

```
service insight plugins> ls

Objects   REFERENCE         PLUGIN       GEMS   ATTRIBUTEDEFINITIONS
SCHEMADEFINITION

FLUENTD_PLUGIN-1           splunkHec    …          …
…

Operations

requestUploadToken
```

The splunkHec plugin appears by default and cannot be deleted. Additionally, the requestUploadToken operation provides a token needed to upload a plugin via the data/upload API. If a second plugin is uploaded, it will appear in the list. Before a third plugin could be uploaded, the second must be selected and deleted.

The following curl commands illustrate the use of the API to upload a plugin:

```
curl -s -X POST -k --data @- http://<engine-name>/resources/json/delphix/session -c
~/cookies.txt -H "Content-Type: application/json" <<EOF

{

   "type": "APISession",

   "version": {

       "type": "APIVersion",

       "major": 1,

       "minor": 4,

       "micro": 3

   }

}

EOF

curl -s -X POST -k --data @- http://<engine-name>/resources/json/delphix/login -b ~/
cookies.txt -c ~/cookies2.txt -H "Content-Type: application/json" <<EOF

{

"type": "LoginRequest",

"username": "sysadmin",

"password": "sysadmin",

"target": "SYSTEM"

}

EOF

curl -s -X POST -k --data @- http://<engine-name>/resources/json/delphix/service/
insight/plugins/requestUploadToken -b ~/cookies2.txt -H "Content-Type: application/
json" <<EOF

Returns the token e.g:  {"type":"OKResult","status":"OK","result":{"type":"FileUpload
Result","url":"/resources/json/delphix/data/upload","token":"59346df5-3ecd-4ced-
afbf-97acefa156dc"},"job":null,"action":null}
```

```
curl  -b ~/cookies2.txt -X POST -F "file=@/Users/blewis/ws/far-dev/
splunk_host_port.far" -F "token=<token>" http:/<engine-name>/resources/json/delphix/
data/upload
```

The service/insight/plugins API displays the tokens that are available.

An example for creating a configuration with the default splunkHec plugin:

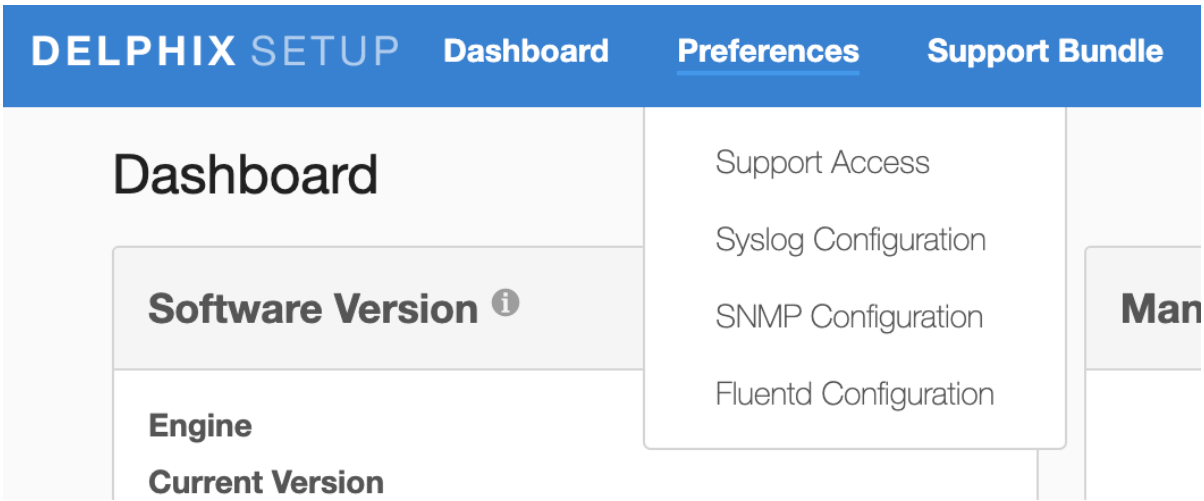
```
service insight configuration> create

service insight configuration create *> set plugin=splunkHec; set enabled=true; edit
attributes

service insight configuration create attributes *> add; set name=hec_host; set
value=http://vmname-splunkhost.delphix.com; back

service insight configuration create attributes *> add; set name=hec_port; set
value=8088; back;

service insight configuration create attributes *> add; set
type=FluentdSecretAttribute;

service insight configuration create attributes 2 *> set name=hec_token; set
secretValue=bb75c032-bdea-4c19-b152-d158cf13e019; back

service insight configuration create attributes *> add; set name=eventsIndex; set
value=delphix_events; back;

service insight configuration create attributes *> add; set name=protocol; set
value=https; back

service insight configuration create attributes *> add; set
name=metricsPushFrequency; set value=60; back

service insight configuration create attributes *> add; set name=metricsIndex; set
value=delphix_metrics; back

service insight configuration create attributes *> add; set name=eventsPushFrequency;
set value=60; back;

service insight configuration create attributes *> ls

Properties

    0:

        type: FluentdRegularAttribute (*)

        name: hec_host (*)

        value: http://vmname-splunkhost.delphix.com (*)
```

```
1:
    type: FluentdRegularAttribute (*)
    name: hec_port (*)
    value: 8088 (*)
2:
    type: FluentdSecretAttribute (*)
    name: hec_token (*)
    secretValue: ******** (*)
3:
    type: FluentdRegularAttribute (*)
    name: eventsIndex (*)
    value: delphix_events (*)
4:
    type: FluentdRegularAttribute (*)
    name: protocol (*)
    value: https (*)
5:
    type: FluentdRegularAttribute (*)
    name: metricsPushFrequency (*)
    value: 60 (*)
6:
    type: FluentdRegularAttribute (*)
    name: metricsIndex (*)
    value: delphix_metrics (*)
7:
    type: FluentdRegularAttribute (*)
    name: eventsPushFrequency (*)
```

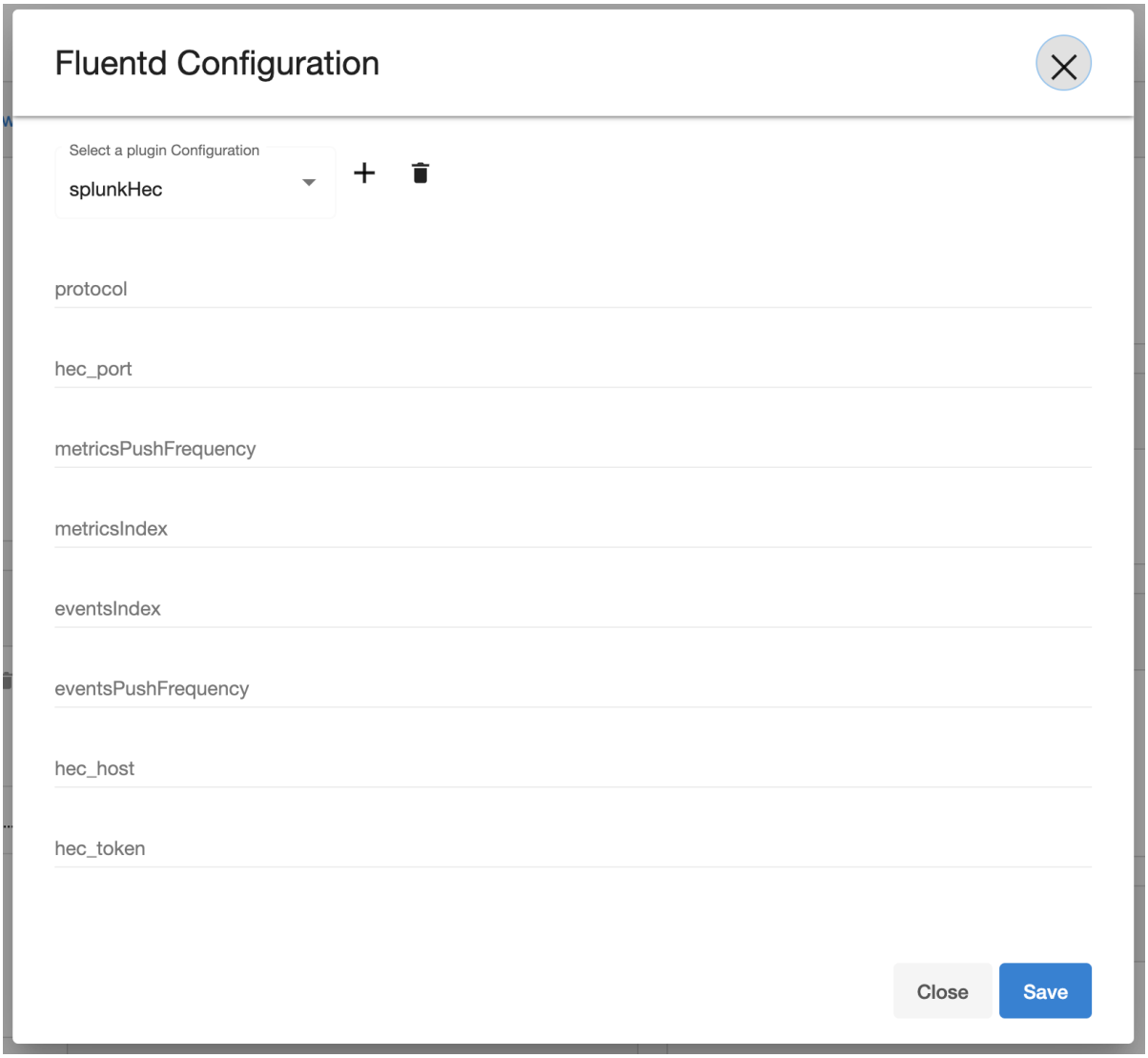
```
        value: 60 (*)
## Use the "add" command to add an element to this array.
service insight configuration create attributes *> back
service insight configuration create *> ls
Properties
    type: FluentdConfig
    name: (unset)
    attributes: [ ... ] (*)
    enabled: true (*)
    plugin: splunkHec (*)
service insight configuration create *> commit
```

Creating default splunkHec configuration from GUI

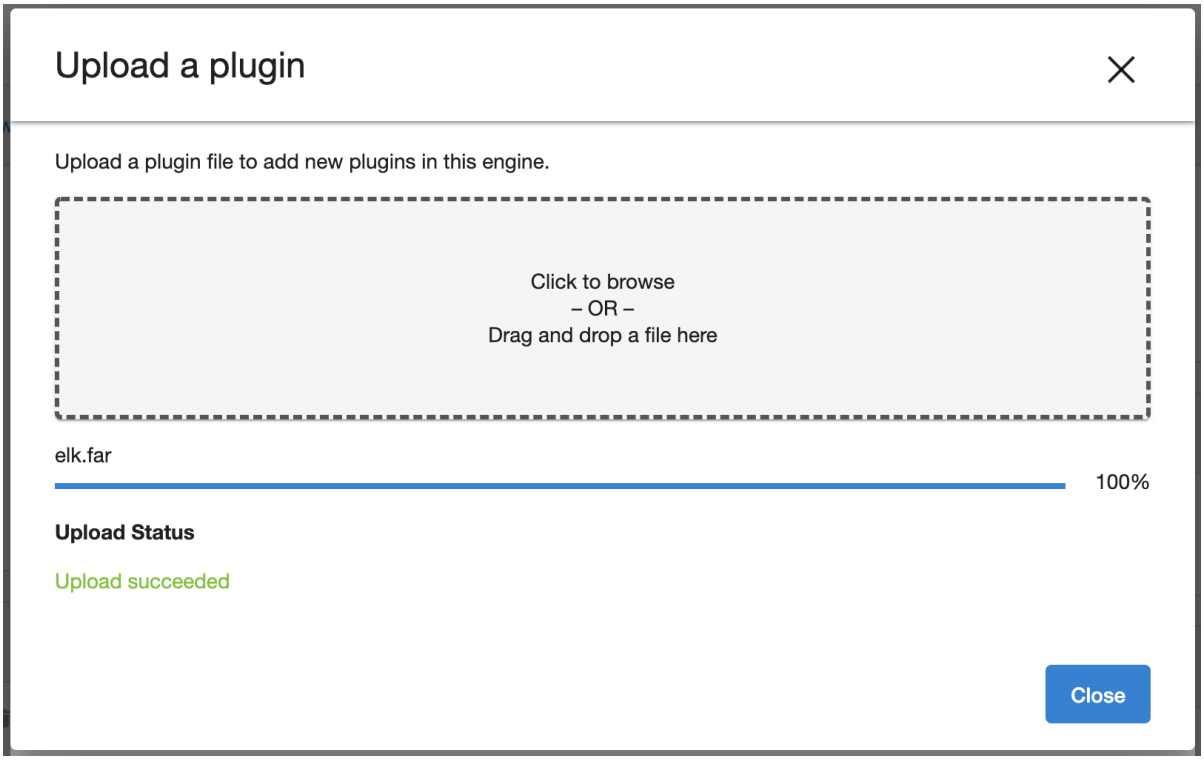
1. Login as a sysadmin user and select **Fluentd Configuration** under the Preferences menu.



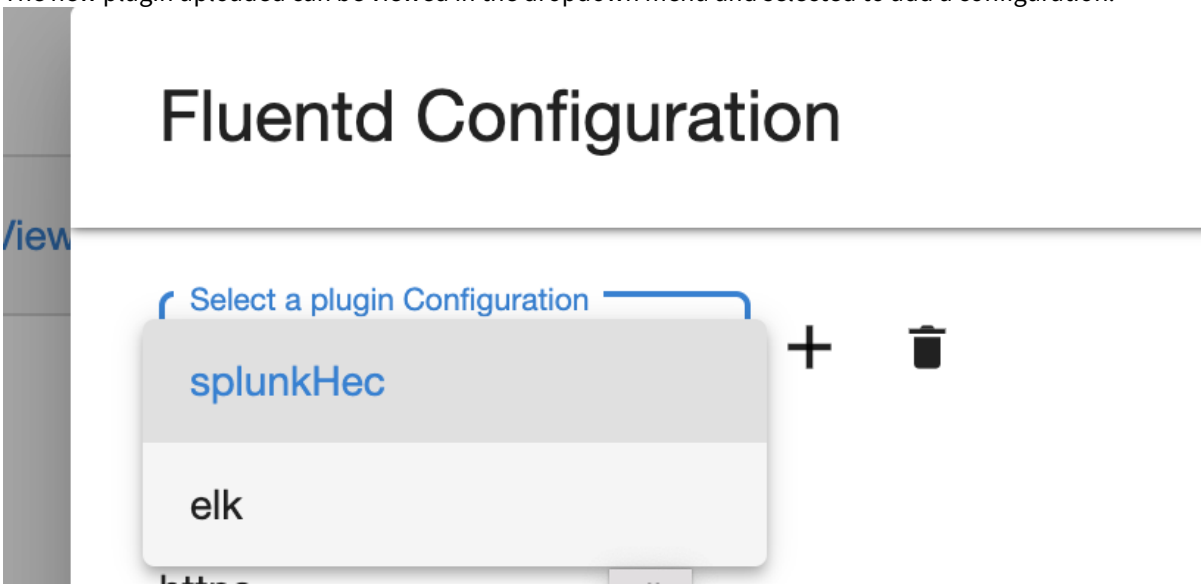
2. The splunkHec plugin appears in the dropdown by default.



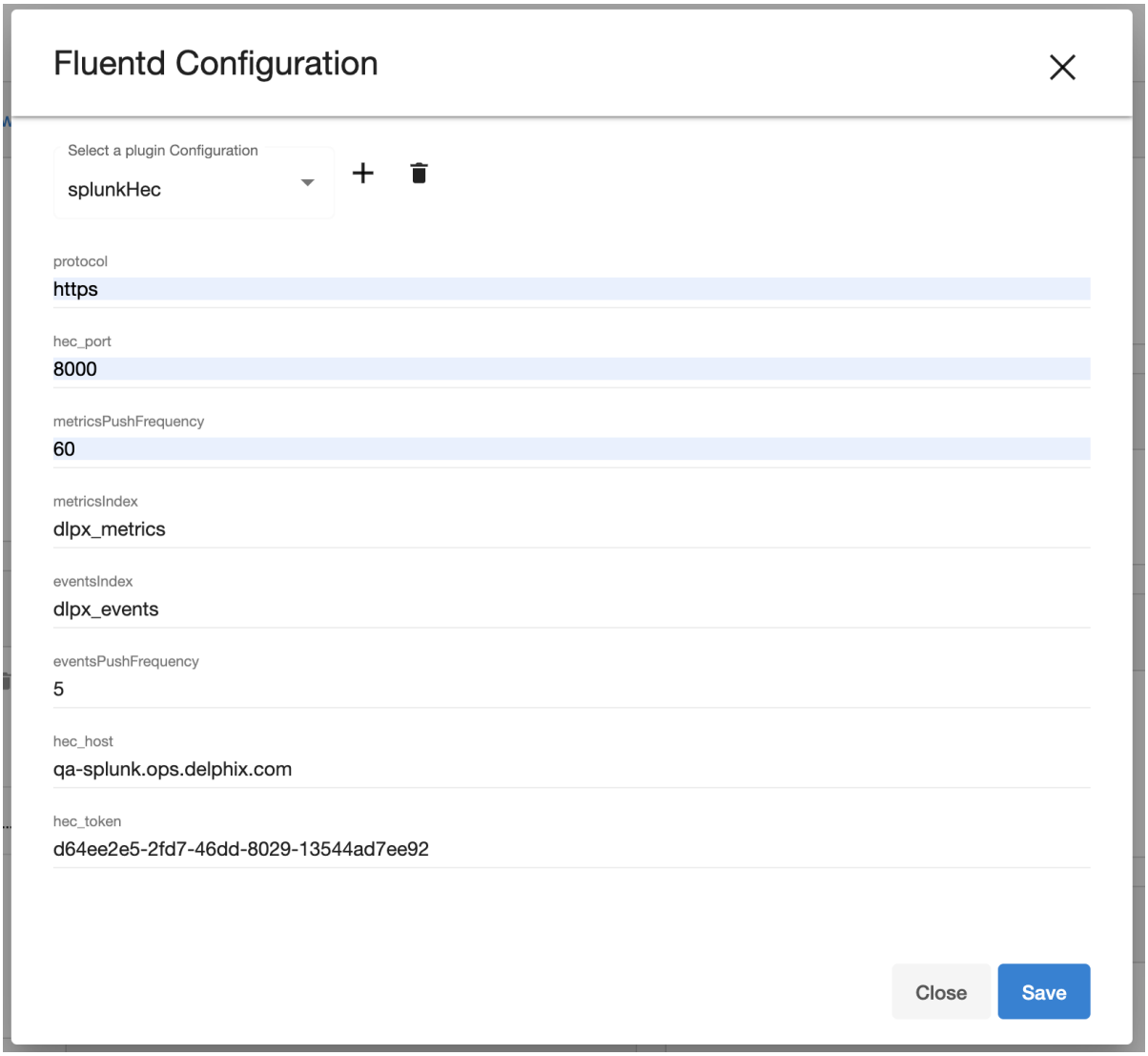
3. Additional plugins can be added by using the **+** button on the right.



4. The new plugin uploaded can be viewed in the dropdown menu and selected to add a configuration.



5. The delete button on the right can be used to delete a selected plugin.
6. The name of fields for configuration would be the same as those provided in the uploaded plugin file.



7. The configuration can be saved by clicking the save button on the bottom.
8. When loading an existing configuration on UI, the secret fields are masked.

Implementation

Secret attributes have been introduced in order to protect private data such as passwords and tokens. In the CLI and API, they are identified by the **FluentdSecretAttribute** type. In plugins, attribute names in the config file template beginning with **_secret** indicate secret data.

Known issues

Fluentd runs in a systemd container to limit negative side-effects from a bad plugin. More security can be added to this container by running as a non-root user, in addition to limiting the size of the log and buffer files it can write.

Splunk integration

Delphix enables self-monitoring/diagnosability of Delphix Engines by providing native integration with Splunk Enterprise. By providing details about your Splunk instance, you can allow Delphix to send structured JSON logs to Splunk that capture activity on the Delphix engine(s). These logs include Delphix events (Actions, Job Events, Faults, and Alerts) as well as performance metrics (CPU, disk, network, TCP, dataset, NFS, iSCSI) and capacity metrics. Delphix Insight enables extensible search and visualization of actionable information and provides a centralized, comprehensive view of Delphix activity (including the ability to cross-reference information from multiple Delphix engines) on a platform that allows building your own operational intelligence for your Delphix installation.

This section covers the following topics:

Configuring splunk

Prerequisites

Before you configure the Delphix Engine you will need to configure and make a note of the following in Splunk:

⚠ Please refer to the Splunk documentation for detailed steps on how to configure your values.

ⓘ **Supported Splunk versions**
Delphix only supports Splunk Enterprise 6.3.0 or higher.

1. In the Splunk web **UI Enable SSL** (this is optional but best practice for security) in your global HTTP Event Collector (HEC) settings.
2. The **Splunk hostname** or **IP Address**.
3. The **HEC Port** number for your Splunk instance (default 8088).
4. Enable the HTTP Event Collector on Splunk, and create a new **HEC Token** with a new Splunk index set as an allowed index for the token. Make sure **Enable Indexer Acknowledgement** is **unchecked** for the token. **Warning** : If you wish, you can use a separate Splunk index for performance and capacity metrics (otherwise, the same index will be used for both events and metrics). If you are using Splunk 7.0+, it is recommended that you create this second index as a special “Metrics” type index that is optimized for indexing and searching metrics data.
5. Note the **HEC Token Value** and the **Allowed Indexes** for the token.

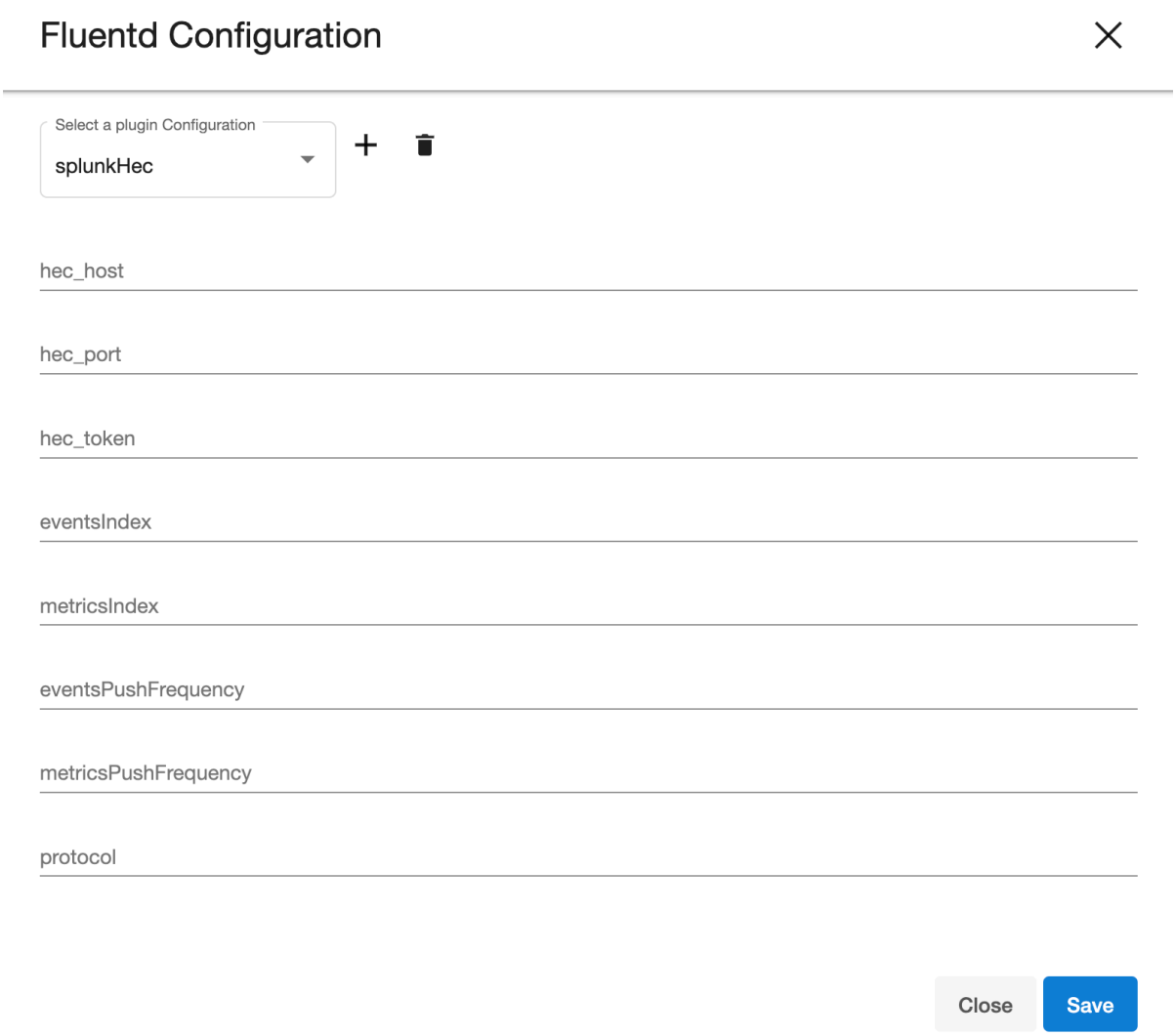
The following table provides an example of the data you will need.

Attribute	**Sample data**
Splunk Server IP address	192.168.8.8
Splunk Server HEC Port Number	8088
Splunk HEC Token	12345678-1234-1234-1234-1234567890AB
Index Name for Events	delphix_events
Index Name for Metrics	delphix_metrics

Configuring Delphix for Splunk

1. Log in to the **Delphix Server Setup UI** as the sysadmin.
2. From the **Preferences** menu select **Fluentd Configuration**.

3. In the Fluentd Configuration window, enter your Splunk values, using the default **splunkHec** plugin configuration.



Host	Splunk hostname or IP address
HEC Port	The TCP port number for the Splunk HTTP Event Collector (HEC)
HEC Token	The token for the Splunk HTTP Event Collector (HEC)
Events Index	The Splunk Index events will be sent to. It must be set as an allowed index for the HEC token.
Metrics Index	The Splunk Index metrics will be sent to. If none is specified then the Main Index will be used for metrics as well. It must be set as an allowed index for the HEC token.

Host	Splunk hostname or IP address
Events Push Frequency	The frequency at which the Events will be pushed to Splunk. Specified in seconds.
Metrics Push Frequency	The frequency at which the Performance Metrics will be pushed to Splunk. Specified in seconds
Protocol	What protocol to use (HTTP or HTTPS) when connecting to Splunk. Must match your HTTP Event Collector settings in Splunk.

4. Click **Save** to enable the Splunk configuration and begin sending all new Actions, Job Events, Faults, Alerts, and Metrics to your Splunk instance.

Using search

Use the search to analyze your data and enumerate items in a metrics index. For more about searching a metrics index, refer to the Splunk documentation.

Search examples - Metrics

The following examples provide information on viewing Metrics on Splunk 7.x

To get a list of all Metrics:

```
| mcatalog values(metric_name)
```

To get a list of all dimensions of a given metric - say CPU utilization percentage:

```
| mcatalog values(_dims) where metric_name="system.cpu.util.pct"
```

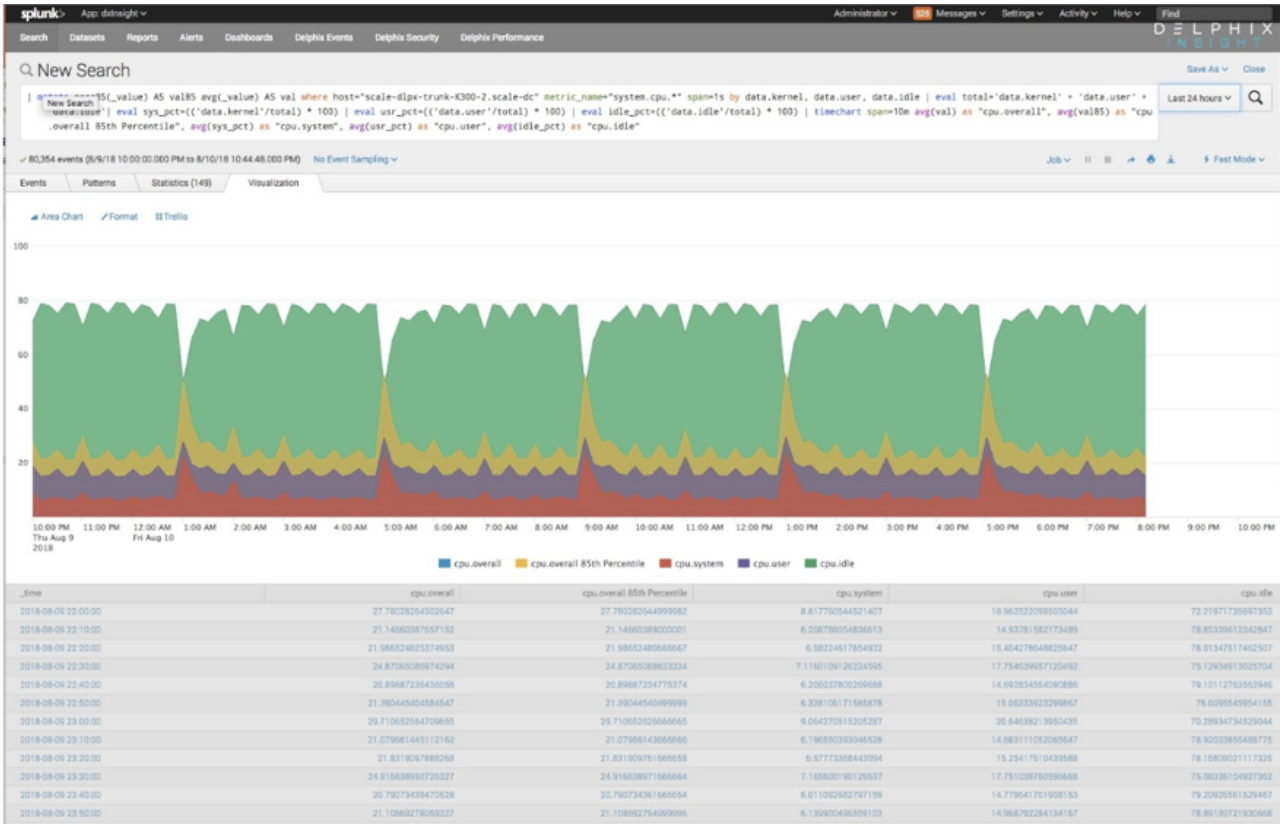
To view the average values of overall CPU utilization percentage across all hosts with a span of 30 seconds:

```
| mstats avg(_value) WHERE index=delphix_metrics AND metric_name=system.cpu.util.pct
span=30s
```

You can also display results in a chart with CPU wildcard:

```
| mstats perc85(_value) AS val85 avg(_value) AS val where metric_name="system.cpu.*"
 span=1s by data.kernel, data.user, data.idle
| eval total='data.kernel' + 'data.user' + 'data.idle'
| eval sys_pct=(('data.kernel'/total) * 100)
| eval usr_pct=(('data.user'/total) * 100)
| eval idle_pct=(('data.idle'/total) * 100)
| timechart span=10m avg(val) as "cpu.overall", avg(val85) as "cpu.overall 85th
Percentile", avg(sys_pct) as "cpu.system", avg(usr_pct) as "cpu.user", avg(idle_pct)
as "cpu.idle"
```

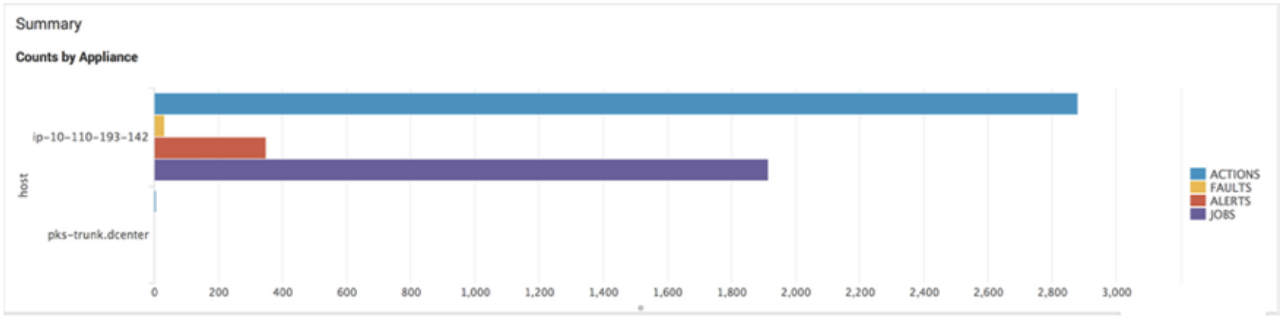
This type of search can be used to stack different CPU metrics that add up to 100%. Here is a sample screenshot of the above “stack different CPU metrics” from the Delphix Engines.



Search examples - events

The following queries demonstrate some basic visualizations of various Delphix events. The **delphix _index** value should be replaced with the name of the **Main Index** provided in during Delphix Setup. These examples serve as useful starting points that can be expanded to include other relevant data. See Events Format for a full description of the structure of each type of event.

Display event statistics per host

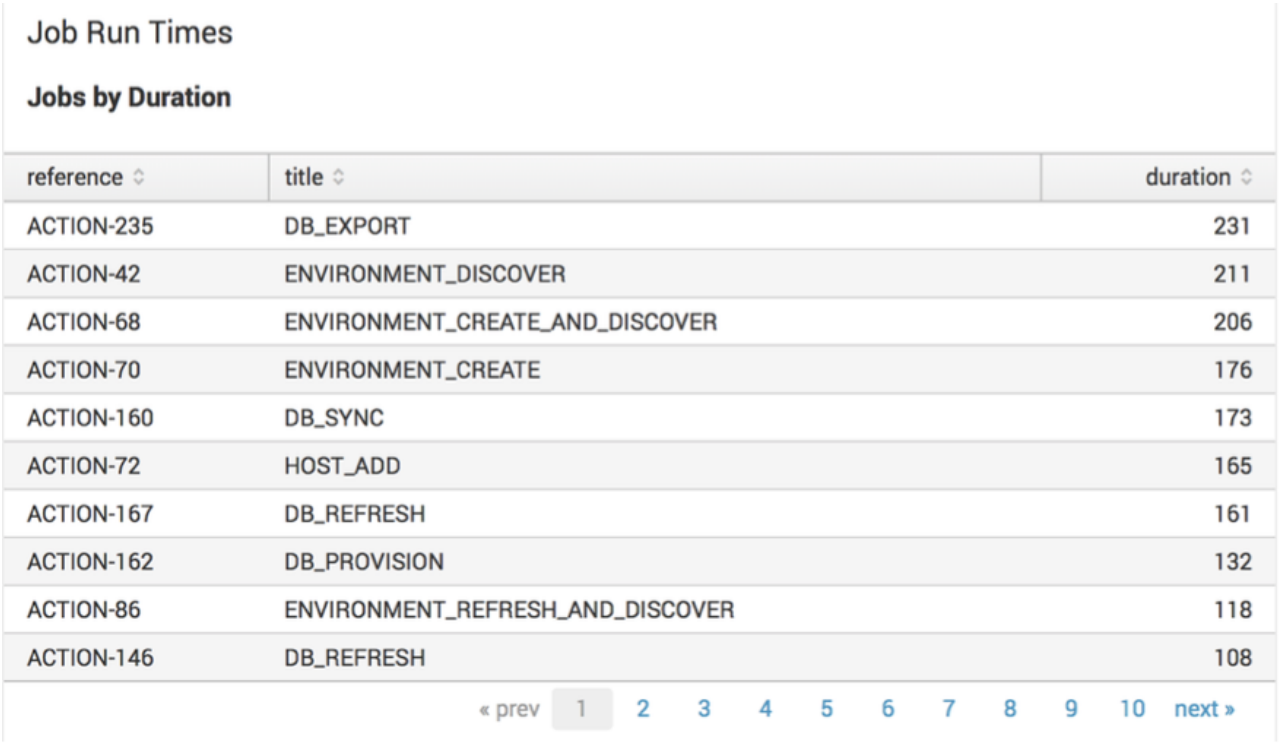


Search

```
index="delphix_index" | stats
count(eval(source="delphix.events.action.completed" OR
source="delphix.events.action.started" OR
source="delphix.events.action.waiting" )) AS ACTIONS
count(eval(source="delphix.events.fault.posted")) AS FAULTS
```

```
count(eval(source="delphix.events.alert" )) AS ALERTS
count(eval(source="delphix.events.job.event" )) AS JOBS BY host
```

List actions in descending order by the duration



Job Run Times

Jobs by Duration

reference	title	duration
ACTION-235	DB_EXPORT	231
ACTION-42	ENVIRONMENT_DISCOVER	211
ACTION-68	ENVIRONMENT_CREATE_AND_DISCOVER	206
ACTION-70	ENVIRONMENT_CREATE	176
ACTION-160	DB_SYNC	173
ACTION-72	HOST_ADD	165
ACTION-167	DB_REFRESH	161
ACTION-162	DB_PROVISION	132
ACTION-86	ENVIRONMENT_REFRESH_AND_DISCOVER	118
ACTION-146	DB_REFRESH	108

« prev 1 2 3 4 5 6 7 8 9 10 next »

Search

```
index="delphix_index" source="delphix.events.action.*" | transaction reference |
table reference title duration | sort duration
```

Lists faults

Faults

host	dateDiagnosed	details	reference
ip-10-110-193-142	2018-07-05T23:34:37.968Z	The TCP sunrpc.tcp_slot_table_entries property is currently set to '16' which is below the recommended minimum value of 128.	FAULT-35
ip-10-110-193-142	2018-07-05T23:34:37.902Z	The default send buffer that can be allocated for a TCP socket dictated by the second value in the net.ipv4.tcp_wmem property is currently set to '16384' which is below the recommended default of 4194304.	FAULT-34
ip-10-110-193-142	2018-07-05T23:34:37.612Z	The default receive buffer that can be allocated for a TCP socket dictated by the second value in the net.ipv4.tcp_rmem property is currently set to '87380' which is below the recommended default of 16777216.	FAULT-33
ip-10-110-193-142	2018-07-05T23:20:45.107Z	The owner of the Delphix toolkit installation directory is not 'sybase'.	FAULT-32
ip-10-110-193-142	2018-07-05T23:19:19.827Z	The owner of the Delphix toolkit installation directory is not 'sybase'.	FAULT-31

« prev 1 2 3 4 5 6 7 next »

Search

```
index="delphix_index"   source="delphix.events.fault.posted" | table host
dateDiagnosed details reference
```

Completed jobs

Events with "Job Complete" Status

host	timestamp	messageDetails	parentAction
ip-10-110-193-142	2018-07-05T23:36:50.018Z	SUPPORT_BUNDLE_DOWNLOAD job for "delphix_admin" completed successfully.	ACTION-1585
ip-10-110-193-142	2018-07-05T23:27:43.114Z	DB_UNDO job for "Untitled/dbdhcp3--dbdh_WSH-1530832953092" completed successfully.	ACTION-1386
ip-10-110-193-142	2018-07-05T23:27:14.739Z	SOURCE_STOP job for "dbdhcp3--dbdh_WSH-1530832953092" completed successfully.	ACTION-1387
ip-10-110-193-142	2018-07-05T23:27:02.994Z	DB_REFRESH job for "Untitled/dbdhcp3--dbdh_WSH-1530832953092" completed successfully.	ACTION-1383
ip-10-110-193-142	2018-07-05T23:27:02.977Z	DB_SYNC job for "Untitled/dbdhcp3--dbdh_WSH-1530832953092" completed successfully.	ACTION-1385

« prev 1 2 3 4 5 6 7 8 9 10 next »

Search

```
index="delphix_index"   source="delphix.events.job.event"| spath jobState | search
jobState=COMPLETED | table host timestamp messageDetails parentAction
```

Lists faults

Faults

host	dateDiagnosed	details	reference
ip-10-110-193-142	2018-07-05T23:34:37.968Z	The TCP sunrpc.tcp_slot_table_entries property is currently set to '16' which is below the recommended minimum value of 128.	FAULT-35
ip-10-110-193-142	2018-07-05T23:34:37.902Z	The default send buffer that can be allocated for a TCP socket dictated by the second value in the net.ipv4.tcp_wmem property is currently set to '16384' which is below the recommended default of 4194304.	FAULT-34
ip-10-110-193-142	2018-07-05T23:34:37.612Z	The default receive buffer that can be allocated for a TCP socket dictated by the second value in the net.ipv4.tcp_rmem property is currently set to '87380' which is below the recommended default of 16777216.	FAULT-33
ip-10-110-193-142	2018-07-05T23:20:45.107Z	The owner of the Delphix toolkit installation directory is not 'sybase'.	FAULT-32
ip-10-110-193-142	2018-07-05T23:19:19.827Z	The owner of the Delphix toolkit installation directory is not 'sybase'.	FAULT-31

« prev 1 2 3 4 5 6 7 next »

Search

```
index="delphix_index"   source="delphix.events.fault.posted" | table host
dateDiagnosed details reference
```

Completed jobs

Events with "Job Complete" Status

host	timestamp	messageDetails	parentAction
ip-10-110-193-142	2018-07-05T23:36:50.018Z	SUPPORT_BUNDLE_DOWNLOAD job for "delphix_admin" completed successfully.	ACTION-1585
ip-10-110-193-142	2018-07-05T23:27:43.114Z	DB_UNDO job for "Untitled/dbdhcp3--dbdh_WSH-1530832953092" completed successfully.	ACTION-1386
ip-10-110-193-142	2018-07-05T23:27:14.739Z	SOURCE_STOP job for "dbdhcp3--dbdh_WSH-1530832953092" completed successfully.	ACTION-1387
ip-10-110-193-142	2018-07-05T23:27:02.994Z	DB_REFRESH job for "Untitled/dbdhcp3--dbdh_WSH-1530832953092" completed successfully.	ACTION-1383
ip-10-110-193-142	2018-07-05T23:27:02.977Z	DB_SYNC job for "Untitled/dbdhcp3--dbdh_WSH-1530832953092" completed successfully.	ACTION-1385

« prev 1 2 3 4 5 6 7 8 9 10 next »

Search

```
index="delphix_index"   source="delphix.events.job.event"| spath jobState | search
jobState=COMPLETED | table host timestamp messageDetails parentAction
```

Search examples - event formats

The Actions, Job Events, Faults, and Alerts that Delphix sends to Splunk are structured according to predefined JSON schemas

JSON schemas for events

The following set of JSON schemas define the shape of each Splunk event, including which properties are expected to exist for each event type and what those properties mean. Some of these .json files are used as shared building blocks to define the other schemas; the top-level schemas which define each distinct event type are **Action.json**, **Alert.json**, **JobEvent.json**, **Fault.json**, and **FaultEffect.json.**

Below key-values are Splunk event metadata - This follows GeneralSplunkHeader.json - schema and GeneralSplunkEvent.json
GeneralSplunkHeader.json

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "type": "object",
    "properties": {
        "time": {
            "description": "The time the event was logged. The default time format is
epoch time format, in the format <sec>.<ms>.",
            "type": "number"
        },
        "host": {
            "description": "The system's hostname. Will be the host value assigned to
the event data in Splunk.",
            "type": "string"
        },
```

```
        "source": {
            "description": "For example mgmt.event.action. Will be the source value
to assigned to the event data in Splunk.",
            "type": "string"
        },
        "sourcetype": {
            "description": "The sourcetype value to assign to the event data.",
            "type": "string"
        },
        "index": {
            "description": "The name of the index by which the event data is to be
indexed.",
            "type": "string"
        }
    },
    "required": ["time", "host", "source", "sourcetype", "index"]
}
```

GeneralSplunkEvent.json

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "type": "object",
    "properties": {
        "systemUniqueId": {
            "description": "The UUID of the system.",
            "type": "string"
        },
        "systemVersion": {
            "description": "The release version of the system.",
            "type": "string"
        }
    },
    "required": ["systemUniqueId", "systemVersion"]
}
```

Action.json

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "definitions": {
        "ActionEvent": {
            "type": "object",
            "properties": {
                "reference": {
                    "description": "The object reference of the action.",
                    "type": "string"
                },
                "title": {
                    "description": "Action title.",
                    "type": "string"
                },
```

```
                "details": {
                    "description": "Plain text description of the action.",
                    "type": "string"
                },
                "startTime": {
                    "description": "The time the action occurred. For a long running
process, this represents the starting time.",
                    "type": "string"
                },
                "endTime": {
                    "description": "The time the action completed.",
                    "type": "string"
                },
                "user": {
                    "description": "The user who initiated the action.",
                    "type": "string"
                },
                "userAgent": {
                    "description": "Name of the client software used to initiate the
action.",
                    "type": "string"
                },
                "originIp": {
                    "description": "Network address used to initiate the action",
                    "type": "string"
                },
                "parentAction": {
                    "description": "The parent action of this action.",
                    "type": "string"
                },
                "state": {
                    "description": "State of the action",
                    "type": "string"
                },
                "workSource": {
                    "description": "Origin of the work that caused the action.",
                    "type": "string"
                },
                "workSourceName": {
                    "description": "Name of the user or policy that initiated the
action.",
                    "type": "string"
                },
                "failureDescription": {
                    "description": "Details of the action failure.",
                    "type": "string"
                },
                "failureAction": {
                    "description": "Action to be taken to resolve the failure",
                    "type": "string"
                },
                "failureMessageCode": {
                    "description": "Message ID associated with the event.",
```

```
                    "type": "string"
                }
            },
            "required": ["reference", "title", "details", "state"]
        }
    },
    "type": "object",
    "title": "Action",
    "allOf": [{ "$ref": "GeneralSplunkHeader.json#" }],
    "properties": {
        "event": {
            "type": "object",
            "allOf": [{ "$ref": "#/definitions/ActionEvent" }, { "$ref":
"GeneralSplunkEvent.json#" }]
        }
    }
}
```

Alert.json

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "definitions": {
        "AlertEvent": {
            "type": "object",
            "properties": {
                "reference": {
                    "description": "The object reference of the alert.",
                    "type": "string"
                },
                "title": {
                    "description": "Title of the event which triggered the alert.",
                    "type": "string"
                },
                "code": {
                    "description": "Dotted descriptor of the type of event which
triggered the alert.",
                    "type": "string"
                },
                "eventSeverity": {
                    "description": "The severity of the event.",
                    "type": "string"
                },
                "details": {
                    "description": "Plain text description of the event which
triggered the alert.",
                    "type": "string"
                },
                "response": {
                    "description": "Automated response, if any, taken by the system.",
                    "type": "string"
                },
```

```
                "timestamp": {
                    "description": "The time the alert occurred.",
                    "type": "string"
                },
                "target": {
                    "description": "Reference to the target object.",
                    "type": "string"
                },
                "targetName": {
                    "description": "Name of the target object.",
                    "type": "string"
                }
            },
            "required": ["reference", "title", "code", "details", "timestamp"]
        }
    },

    "type": "object",
    "title": "Alert",
    "allOf": [{ "$ref": "GeneralSplunkHeader.json#" }],
    "properties": {
        "event": {
            "type": "object",
            "allOf": [{ "$ref": "#/definitions/AlertEvent" }, { "$ref":
"GeneralSplunkEvent.json#" }]
        }
    }
}
```

JobEvent.json

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "definitions": {
        "JobEvent": {
            "type": "object",
            "properties": {
                "job": {
                    "description": "The object reference of job associated with this
event.",
                    "type": "string"
                },
                "parentAction": {
                    "description": "The object reference of the parent action of the
associated job.",
                    "type": "string"
                },
                "jobState": {
                    "description": "The new state of the job.",
                    "type": "string"
                },
                "timestamp": {
```

```
                    "description": "The time the event occurred.",
                    "type": "string"
                },
                "percentComplete": {
                    "description": "Completion percentage of the job associated with
this event",
                    "type": "number"
                },
                "diagnoses": {
                    "description": "If job failed, a set of diagnoses of things that
may have caused the failure.",
                    "type": "array",
                    "items": { "$ref": "#/definitions/DiagnosisResult" }
                },
                "eventType": {
                    "description": "The type of this event (info, warning, or
error).",
                    "type": "string"
                },
                "messageCode": {
                    "description": "A message code describing this event.",
                    "type": "string"
                },
                "messageDetails": {
                    "description": "A message describing the details of this event.",
                    "type": "string"
                },
                "messageAction": {
                    "description": "Action to be taken by the user to repair or
remedy the situation.",
                    "type": "string"
                },
                "messageCommandOutput": {
                    "description": "Any command output generated by this event",
                    "type": "string"
                }
            },
            "required": ["job", "parentAction", "timestamp", "percentComplete",
"messageCode", "messageDetails", "eventType"]
        },
        "DiagnosisResult": {
            "type": "object",
            "properties": {
                "diagnosisCode": {
                    "description": "Message code associated with this diagnosis
check.",
                    "type": "string"
                },
                "diagnosisMessage": {
                    "description": "Description of this diagnosis check.",
                    "type": "string"
                },
                "failed": {
```

```
                    "description": "True if this diagnosis check did not pass.",
                    "type": "boolean"
                },
                "targetReference": {
                    "description": "Reference of the target object of this diagnosis
check, if applicable.",
                    "type": "string"
                }
            },
            "required": ["diagnosisCode", "diagnosisMessage", "failed"]
        }
    },
    "type": "object",
    "title": "JobEvent",
    "allOf": [{ "$ref": "GeneralSplunkHeader.json#" }],
    "properties": {
        "event": {
            "type": "object",
            "allOf": [{ "$ref": "#/definitions/JobEvent" }, { "$ref":
"GeneralSplunkEvent.json#" }]
        }
    }
}
```

AbstractFault.json

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "type": "object",
    "title": "AbstractFault",
    "properties": {
        "reference": {
            "description": "The object reference of the fault.",
            "type": "string"
        },
        "title": {
            "description": "Title of the event which triggered the fault.",
            "type": "string"
        },
        "code": {
            "description": "Dotted descriptor of the type of event which triggered
the fault.",
            "type": "string"
        },
        "details": {
            "description": "Plain text description of the event which triggered the
fault.",
            "type": "string"
        },
        "response": {
            "description": "Automated response, if any, taken by the system.",
            "type": "string"
```

```
        },
        "dateDiagnosed": {
            "description": "The date when the fault was diagnosed.",
            "type": "string"
        },
        "target": {
            "description": "Reference to the target object.",
            "type": "string"
        },
        "targetName": {
            "description": "Name of the target object.",
            "type": "string"
        },
        "state": {
            "description": "The state of the fault."
        },
        "eventSeverity": {
            "description": "The severity of the event.",
            "type": "string"
        }
    },

    "required": ["reference", "title", "details", "state", "target", "dateDiagnosed"]
}
```

Fault.json

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "definitions": {
        "FaultEvent": {
            "type": "object",
            "properties": {
                "dateResolved": {
                    "description": "The date when the fault was resolved.",
                    "type": "string"
                },
                "resolutionComments": {
                    "description": "Comments regarding the resolution of the fault.",
                    "type": "string"
                }
            }
        }
    },

    "type": "object",
    "title": "Fault",
    "allOf": [{ "$ref": "GeneralSplunkHeader.json#" }],
    "properties": {
        "event": {
            "type": "object",
```

```
            "allOf": [{ "$ref": "AbstractFault.json#" }, { "$ref": "#/definitions/
FaultEvent" }, { "$ref": "GeneralSplunkEvent.json#" }]
        }
    }
}
```

FaultEffect.json

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "definitions": {
        "AlertEvent": {
            "type": "object",
            "properties": {
                "reference": {
                    "description": "The object reference of the alert.",
                    "type": "string"
                },
                "title": {
                    "description": "Title of the event which triggered the alert.",
                    "type": "string"
                },
                "code": {
                    "description": "Dotted descriptor of the type of event which
triggered the alert.",
                    "type": "string"
                },
                "eventSeverity": {
                    "description": "The severity of the event.",
                    "type": "string"
                },
                "details": {
                    "description": "Plain text description of the event which
triggered the alert.",
                    "type": "string"
                },
                "response": {
                    "description": "Automated response, if any, taken by the system.",
                    "type": "string"
                },
                "timestamp": {
                    "description": "The time the alert occurred.",
                    "type": "string"
                },
                "target": {
                    "description": "Reference to the target object.",
                    "type": "string"
                },
                "targetName": {
                    "description": "Name of the target object.",
                    "type": "string"
                }
```

```
            },
            "required": ["reference", "title", "code", "details", "timestamp"]
        }
    },

    "type": "object",
    "title": "Alert",
    "allOf": [{ "$ref": "GeneralSplunkHeader.json#" }],
    "properties": {
        "event": {
            "type": "object",
            "allOf": [{ "$ref": "#/definitions/AlertEvent" }, { "$ref":
"GeneralSplunkEvent.json#" }]
        }
    }
}
```

Types of events

Delphix uses the **source** field in Splunk to designate the type of each event. Here is a full list of the possible values of the **source** field for events, along with an explanation of when each event is generated and the name of the corresponding JSON schema that describes the event structure.

source	Explanation	Schema
delphix.events.action.started	Action has started running.	Action.json
delphix.events.action.waiting	Action has moved to the WAITING state.	Action.json
delphix.events.action.completed	Action has completed successfully.	Action.json
delphix.events.action.failed	Action has failed.	Action.json
delphix.events.action.canceled	Action has been canceled.	Action.json
delphix.events.job.event	Job Event has been generated in response to a Job progress update.	JobEvent.json
delphix.events.fault.posted	Fault has been posted.	Fault.json
delphix.events.fault.resolved	Fault has been resolved.	Fault.json
delphix.events.fault.ignored	User has chosen to ignore a fault.	Fault.json
delphix.events.fault.unignored	User has chosen to “unignore” a previously ignored fault.	Fault.json

source	Explanation	Schema
delphix.events.fault.effect.posted	Fault Effect has been posted as a downstream effect of some Fault.	FaultEffect.json
delphix.events.fault.effect.resolved	Fault Effect has been resolved as a result of its cause being resolved.	FaultEffect.json
delphix.events.fault.effect.ignored	Fault Effect has been ignored when a user chose to ignore its cause.	FaultEffect.json
delphix.events.fault.effect.unignored	Fault Effect has been “unignored” when a user chose to unignore its previously ignored cause.	FaultEffect.json
delphix.events.alert	Alert has been posted.	Alert.json

Search examples - metric formats

JSON schemas for metrics

GeneralMetricEvent.json

```
{
   "$schema": "http://json-schema.org/draft-04/schema#",
   "type": "object",
   "properties": {
       "systemUniqueId": {
           "description": "The UUID of the system.",
           "type": "string"
       },
       "systemVersion": {
           "description": "The release version of the system.",
           "type": "string"
       },
       "event": {
          "description": "A tag that indicates this Event is a metric.",
          "type": "string",
          "enum": ["metric"]
       }
   },
   "required": ["systemUniqueId", "systemVersion", "event"]
}
```

A schema that is common for all metrics, the “data” nested JSON object contents varies depending on the metric:

CommonMetric.json

```
{
   "$schema": "http://json-schema.org/draft-04/schema#",
   "definitions": {
       "CommonMetric": {
           "type": "object",
           "properties": {
               "name": {
                   "description": "The name of the metric.",
                   "type": "string"
               },
               "time": {
                   "description": "The timestamp of the metric when it was collected.
The default time format is epoch time format, in the format <sec>.<ms>.",
                   "type": "number"
               },
               "value": {
                   "description": "The numeric value of the metric.",
                   "type": "number"
               },
               "type": {
                   "description": "The type of measurement.",
                   "type": "string",
                   "enum": ["counter", "g", "value", "summary"]
               },
               "data": {
                   "description": "JSON object having further details on the metric.
Contents depends on the actual metric.",
                   "type": "object"
               }
           }
       }
   },

   "type": "object",
   "title": "CommonMetric",
   "allOf": [{ "$ref": "GeneralSplunkHeader.json#" }],
   "properties": {
       "event": {
           "type": "object",
           "allOf": [{ "$ref": "#/definitions/CommonMetric" }, { "$ref":
"GeneralMetricEvent.json#" }]
       }
   }
}
```

Metrics format

Metrics Format is the combination of the two tables below (metadata + metric specific key-values).

JSON key	Value type	Description	Example	Comments
source	String	Dotted name hierarchy for insight source.	delphix.metrics.xyz	This is the “source” value assigned to an event data in Splunk.
index	String	Splunk metrics index name	insight_metrics	Splunk Index name
host	String	Hostname/IP	pks-insight.dc2.delphix.com	Could also serve as a tag in other time-series data (like opentsdb)
event	***"metric"***	Describes what kind of event this is.	n/a	Signifies this Splunk event is a “Metric”.
sourcetype	***“_json”***	Data format	n/a	Used for Splunk Indexed field extractions
systemUniqueId	String	The UUID of the system	"423f22db-4ee9-6ebe-ff0f-884ffdc351f7"	
systemVersion	String	The release version of the system	5.2.5.0	

And the general key-values specific to an Insight metric are:

JSON key	Value type	Description	Example	Comments
name	String	Pseudo hierarchical dotted format of the metric name.	system.cpu.util.pct system.disk.ops.count system.net.total.bytes	A metric name: • is alphanumeric (underscore & dot allowed) • has *prefix* that points to the source of the data (like *system.cpu*) • has *suffix* that describes the unit (when it can) • an aggregate/summary metric would have *total* as part of suffix for example.
time	UNIX time notation (epoch)	Timestamp	1525399950	UNIX epoch time format.
value	Numeric value of the metric	The actual measurement	85.17 (system.cpu.util.pct)	Numeric only
type	String (Enum): • counter, • gauge, • summary	Type of the measurement	counter | gauge | summary	This field could be used to identify what kind of value a metric is presenting (like a “gauge” for cpu metric implies the values will fall into a range of 0-100% - similarly a “summary” would imply the value is an “accumulated value” like network utilization)
data	JSON Object	Dimensions of the metric (if any).	{ "read_latency": 0.98, "ops_write": 20, "ops_read": 30, "write_latency": 0.55, }	Has additional info about a metric (called “Dimensions” in Splunk) Has different key-vals depending on a given metric Like for CPU: it can have key-vals to like “kernel”, “user” For disk: can have latency/ throughput, operation name

Summary of all the current metrics

Metric name	Metric value	Metric dimensions (data.xxx below)
system.cpu.util.pct	cpu utilization percentage aggregated ***across all cpus***	user, kernel, idle
system.disk.ops.count	disk read/write operations aggregated ***across all disk instances***	count, op (read/write), latency, avgLatency, throughput
***system.net*.total.bytes**	network utilization - total bytes (in + out) of a given network interface	networkInterface, inBytes, outBytes, inPackets, outPackets
system.nfs.ops.count	nfs read/write operations aggregated ***across all instances***	count, op (read/write), latency, avgLatency, throughput
system.iscsi.ops.count	iscsi read/write operations aggregated ***across all instances***	count, op (read/write), latency, avgLatency, throughput
system.tcp.total.bytes	tcp connection statistics aggregated by service (nfs/iscsi/dsp etc)	congestionWindowSize, inBytes, inUnorderedBytes, localAddress, outBytes, receiveWindowSize, remoteAddress, retransmittedBytes, roundTripTime, sendWindowSize, service unacknowledgedBytes, unsentBytes
system.dataset.total.bytes	dataset performance - total number of bytes read + written per dataset (dsource/vdb etc)	dataset, nread, nwritten, type (virtual, dsource, staging)
system.capacity.source.size	overall system capacity - actual used space in bytes	actualSpace, totalSpace, activeSpace, actualSpace, descendantSpace, logSpace, manualSpace, policySpace, syncSpace, timeflowUnvirtualizedSpace, unownedSnapshotSpace, unvirtualizedSpace

Metric name	Metric value	Metric dimensions (data.xxx below)
system.capacity.consumer.size	consumer capacity - actual used space in bytes per consumer (vdb/pdb etc)	syncSpace, descendantSpace, activeSpace, timeflowUnvirtualizedSpace, objectType, dSource, actualSpace, groupName, manualSpace, unownedSnapshotSpace, unvirtualizedSpace, logSpace, policySpace, consumerName

Creating Fluentd plugins

The Delphix Fluentd service includes built-in support for sending data to Splunk, but the feature can be extended to other data consumers through the use of plugins that can be uploaded to a Delphix engine for use by Fluentd. These plugins can be adopted from various widely available options, such as those found at http://rubygems.org, but only require a few specific gem files and a configuration file to tell Fluentd where and how to send metrics to your platform of choice.

Delphix Fluentd plugin structure

The expected file structure of a Delphix Fluentd plugin is simple: A root folder containing a properly configured `fluent.conf.stg` file, and a /gems subfolder with the necessary .gem files. Delphix Fluentd plugins are simpler than a typical Fluentd plugin because of Delphix's native Fluentd integration, which comes with a variety of Ruby gems that are used by many standard plugins. Additionally, the Delphix Fluentd service requires only the dependency gem files (i.e. the files ending with a .gem extension). Additional files that come with a downloaded plugin should be removed.

Pre-installed Fluentd ruby gems

To minimize the surface area of uploads to a given Delphix engine, all .gem files that are already present on the Delphix OS should also be removed from your plugin before upload. Here is a list of the provided gems that should be removed from your /gems folder, if present:

```
addressable (2.8.1)
async (1.30.3)
async-http (0.59.2)
async-io (1.34.0)
async-pool (0.3.12)
aws-eventstream (1.2.0)
aws-partitions (1.650.0)
aws-sdk-core (3.164.0)
aws-sdk-kms (1.58.0)
aws-sdk-s3 (1.116.0)
aws-sdk-sqs (1.51.1)
aws-sigv4 (1.5.2)
benchmark (default: 0.1.0)
bigdecimal (default: 2.0.0)
bindata (2.4.14)
bundler (2.3.18, default: 2.1.4)
cgi (default: 0.1.0.1)
cmetrics (0.3.3)
concurrent-ruby (1.1.10)
console (1.16.2)
cool.io (1.7.1)
csv (default: 3.1.2)
date (default: 3.0.3)
delegate (default: 0.1.0)
did_you_mean (default: 1.4.0)
digest-crc (0.6.4)
digest-murmurhash (1.1.1)
elastic-transport (8.1.0)
elasticsearch (8.4.0)
```

```
elasticsearch-api (8.4.0)
etc (default: 1.1.0)
excon (0.93.1)
faraday (1.10.2)
faraday-em_http (1.0.0)
faraday-em_synchrony (1.0.0)
faraday-excon (1.1.0)
faraday-httpclient (1.0.1)
faraday-multipart (1.0.4)
faraday-net_http (1.0.1)
faraday-net_http_persistent (1.2.0)
faraday-patron (1.0.0)
faraday-rack (1.0.0)
faraday-retry (1.0.3)
faraday_middleware-aws-sigv4 (0.6.1)
fcntl (default: 1.0.0)
ffi (1.15.5)
fiber-local (1.0.0)
fiddle (default: 1.0.0)
fileutils (1.6.0, default: 1.4.1)
fluent-config-regexp-type (1.0.0)
fluent-diagtool (1.0.1)
fluent-logger (0.9.0)
fluent-plugin-calyptia-monitoring (0.1.3)
fluent-plugin-elasticsearch (5.2.4)
fluent-plugin-flowcounter-simple (0.1.0)
fluent-plugin-kafka (0.18.1)
fluent-plugin-metrics-cmetrics (0.1.2)
fluent-plugin-opensearch (1.0.8)
fluent-plugin-prometheus (2.0.3)
fluent-plugin-prometheus_pushgateway (0.1.0)
fluent-plugin-record-modifier (2.1.1)
fluent-plugin-rewrite-tag-filter (2.4.0)
fluent-plugin-s3 (1.7.2)
fluent-plugin-sd-dns (0.1.0)
fluent-plugin-systemd (1.0.5)
fluent-plugin-td (1.2.0)
fluent-plugin-utmpx (0.5.0)
fluent-plugin-webhdfs (1.5.0)
fluentd (1.15.3)
forwardable (default: 1.3.1)
getoptlong (default: 0.1.0)
hirb (0.7.3)
http_parser.rb (0.8.0)
httpclient (2.8.3)
io-console (default: 0.5.6)
ipaddr (default: 1.2.2)
irb (default: 1.2.6)
jmespath (1.6.1)
json (2.6.2, default: 2.3.0)
linux-utmpx (0.3.0)
logger (default: 1.4.2)
ltsv (0.1.2)
```

```
matrix (default: 0.2.0)
mini_portile2 (2.8.0)
minitest (5.13.0)
msgpack (1.6.0)
multi_json (1.15.0)
multipart-post (2.2.3)
mutex_m (default: 0.1.0)
net-pop (default: 0.1.0)
net-smtp (default: 0.1.0)
net-telnet (0.2.0)
nio4r (2.5.8)
observer (default: 0.1.0)
oj (3.13.17)
open3 (default: 0.1.0)
opensearch-api (2.0.2)
opensearch-ruby (2.0.3)
opensearch-transport (2.0.1)
openssl (default: 2.1.3)
ostruct (default: 0.2.0)
parallel (1.22.1)
power_assert (1.1.7)
prime (default: 0.1.1)
prometheus-client (2.1.0)
protocol-hpack (1.4.2)
protocol-http (0.23.12)
protocol-http1 (0.14.6)
protocol-http2 (0.14.2)
pstore (default: 0.1.0)
psych (default: 3.1.0)
public_suffix (5.0.0)
racc (default: 1.4.16)
rake (13.0.6, 13.0.1)
rdkafka (0.11.1)
rdoc (default: 6.2.1.1)
readline (default: 0.0.2)
reline (default: 0.1.5)
rexml (default: 3.2.3.1)
rss (default: 0.2.8)
ruby-kafka (1.5.0)
ruby-progressbar (1.11.0)
ruby2_keywords (0.0.5)
rubyzip (1.3.0)
sdbm (default: 1.0.0)
serverengine (2.3.0)
sigdump (0.2.4)
singleton (default: 0.1.0)
stringio (default: 0.1.0)
strptime (0.2.5)
strscan (default: 1.0.3)
systemd-journal (1.4.2)
td (0.16.9)
td-client (1.0.8)
td-logger (0.3.28)
```

```
test-unit (3.3.4)
timeout (default: 0.1.0)
timers (4.3.5)
tracer (default: 0.1.0)
traces (0.7.0)
tzinfo (2.0.5)
tzinfo-data (1.2022.5)
uri (default: 0.10.0)
webhdfs (0.10.2)
webrick (1.7.0, default: 1.6.1)
xmlrpc (0.3.0)
yajl-ruby (1.4.3)
yaml (default: 0.1.0)
zip-zip (0.3)
zlib (default: 1.1.0)
```

Note that many gems have similar names but may still require your plugin to work. For example, `elasticsearch.gem` from the list above is distinct from `elasticsearch-7.17.1.gem`, which is required for the Elasticsearch 7 Fluentd plugin. When in doubt, there is no harm in including additional gem files, but there is a 256-character limit on the list of file names that the engine will accept, so it is necessary to eliminate as many non-essential gems as possible before uploading.

Setting up a Fluentd configuration file

Besides requiring the necessary gems to connect to a given data consumer, Fluentd needs a configuration file to know where and how to send the data it has received from Delphix. The Fluentd Configuration GUI in Delphix also uses the configuration file to determine what parameters should be displayed to the system administrator during setup. The configuration file should be named `fluent.conf.stg` and must be placed in your plugin's root folder. Each data consumer requires specific syntax to connect with Fluentd. Consult the documentation of your data consumer of choice when creating your `fluent.conf.stg` file. Here is an example configuration file template, which must be customized depending on the requirements of your chosen data provider (`Splunk`, `elasticsearch`, etc.):

```
/*
 * Copyright (c) 2023 Delphix. All rights reserved.
 */

delimiters "^", "^"

/*
 * This template is used by the Delphix management stack to auto-generate the final
configuration used
 * by the internal fluent service. User-editable fields are filled in with
information provided through
 * the GUI or API. Additional params specific to your needs can be substituted for
my_param.
 * buffer_flush_interval is a fluentd parameter that is dynamically populated based
on user input and
 * is provided here as an example.
 */
```

```
fluentConfig(my_param, buffer_flush_interval) ::= <<

<system>
  # equal to -v command line option
  log_level info
  <log>
    format json
    time_format %Y-%m-%dT%H:%M:%S.%NZ
  </log>
</system>

<source>
  @type forward
  port 24224
  bind 127.0.0.1 # only accept connections from localhost
</source>

<match delphix.events.**>
  ^commonFields(tagPrefix="delphix.events", index={^event_index^},
                flushInterval={^buffer_flush_interval^},
                retryTimeout="72h",
                totalLimitSize="10g",
                chunkLimitSize="1m", ...)^
  data_type event
</match>

<match delphix.metrics.**>
  ^commonFields(tagPrefix="delphix.metrics", index={^metrics_index^},
                flushInterval={^buffer_flush_interval^},
                retryTimeout="72h",
                totalLimitSize="10g",
                chunkLimitSize="1m", ...)^
</match>
>>

commonFields(tagPrefix, index, flushInterval, retryTimeout, totalLimitSize,
chunkLimitSize) ::= <<

/* Replace with syntax specific to your data consumer of choice */
@type myDataConsumer

/* Replace with parameters specific to your data consumer */
my_param ^my_param^

source ${tag} # Filled in at runtime by fluentd

<buffer>
  @type file
  path /var/lib/fluent/^tagPrefix^.*.buffer ^! buffer path must be unique for each
match section !^
  chunk_limit_size ^chunkLimitSize^
  total_limit_size ^totalLimitSize^
  flush_interval ^flushInterval^
```

```
  retry_timeout ^retryTimeout^
</buffer>
>>
```

Fluentd defaults to using JSON formatting for its output plugins. Refer to Fluentd documentation for including a formatter plugin in your config should you require an alternative.

Readying the plugin for upload

Here is an example of what a plugin for `elasticsearch 7` might look like after taking the above steps:

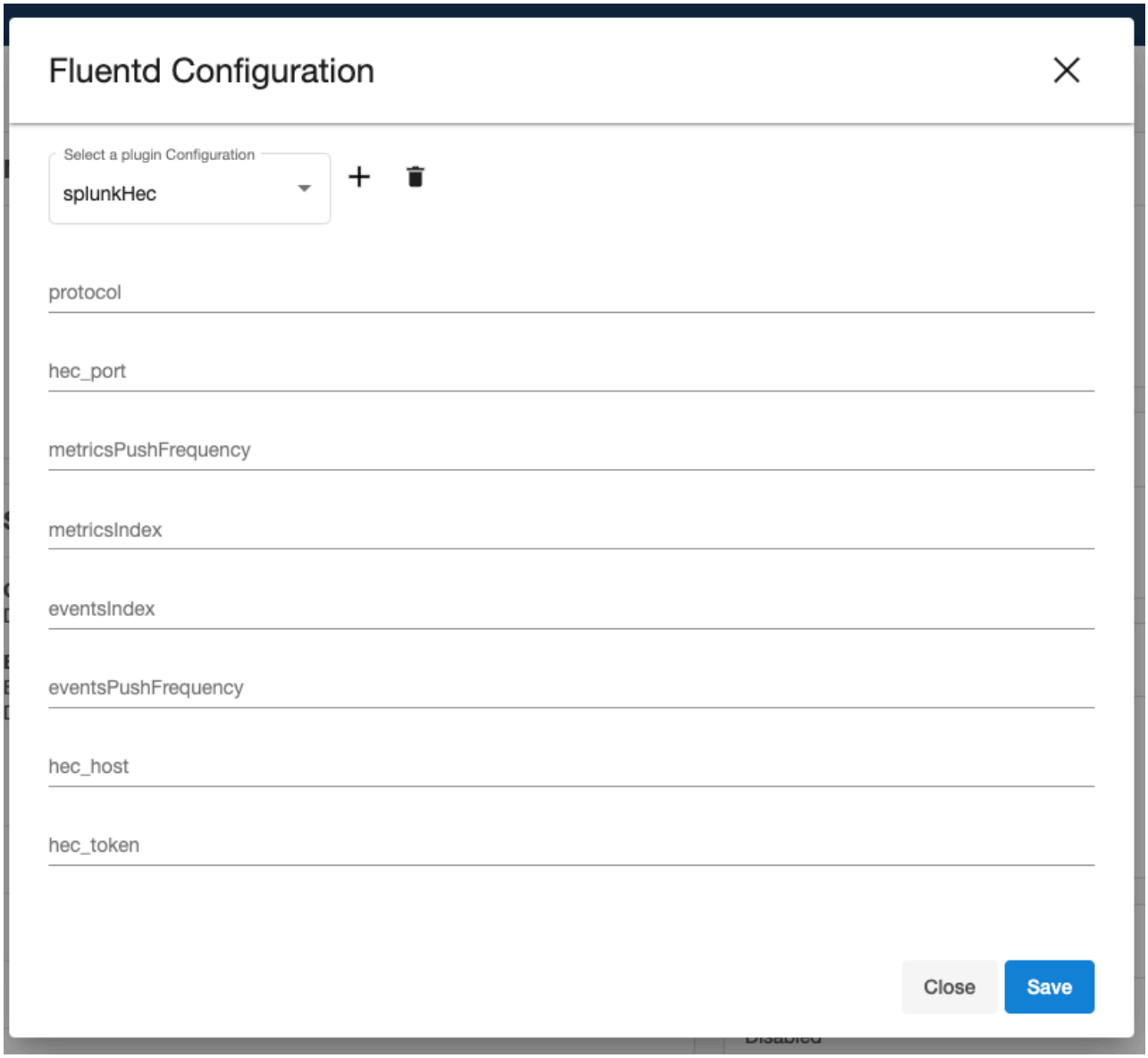
```
$ ls ./elasticsearch-7

fluent.conf.stg gems

$ ls ./elasticsearch-7/gems/

elasticsearch-7.17.1.gem elasticsearch-api-7.17.1.gem elasticsearch-transport-7.17.1.
gem fluent-plugin-elasticsearch-5.1.4.gem
```

Delphix Fluentd plugins must be stored in a .far archive file before uploading them to your engine. You can create a .far archive of your plugin files using a utility such as *tar*. Here is an example *tar* syntax for creating a plugin archive file:

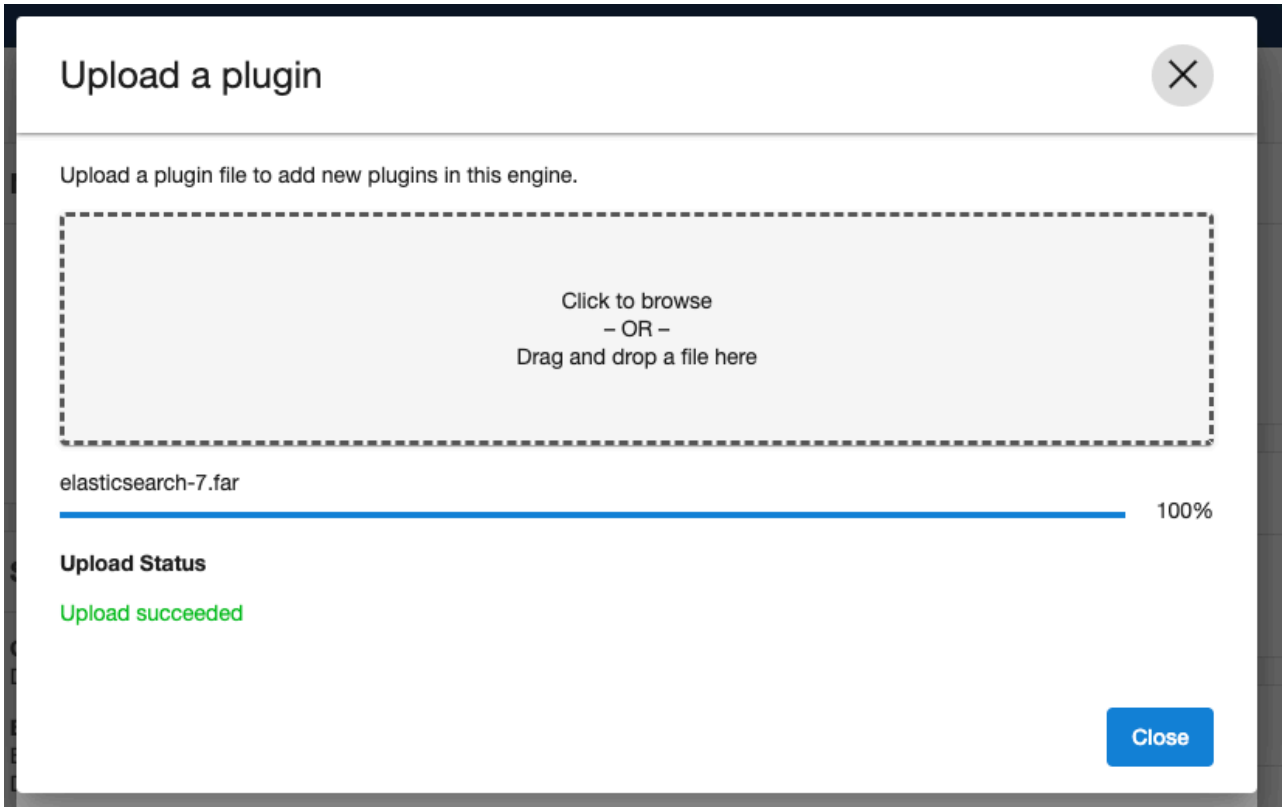
```
tar --create --verbose --owner=0 --group=0 --exclude-backups --exclude-vcs --one-
file-system --format=gnu --file=dlpx-example-plugin.far dlpx-example-plugin/
```

Uploading plugins to Delphix

Your new Fluentd plugin .far file can be uploaded through the API or the GUI in the Fluentd configuration wizard, by clicking the plus icon next to the plugin selection dropdown menu:

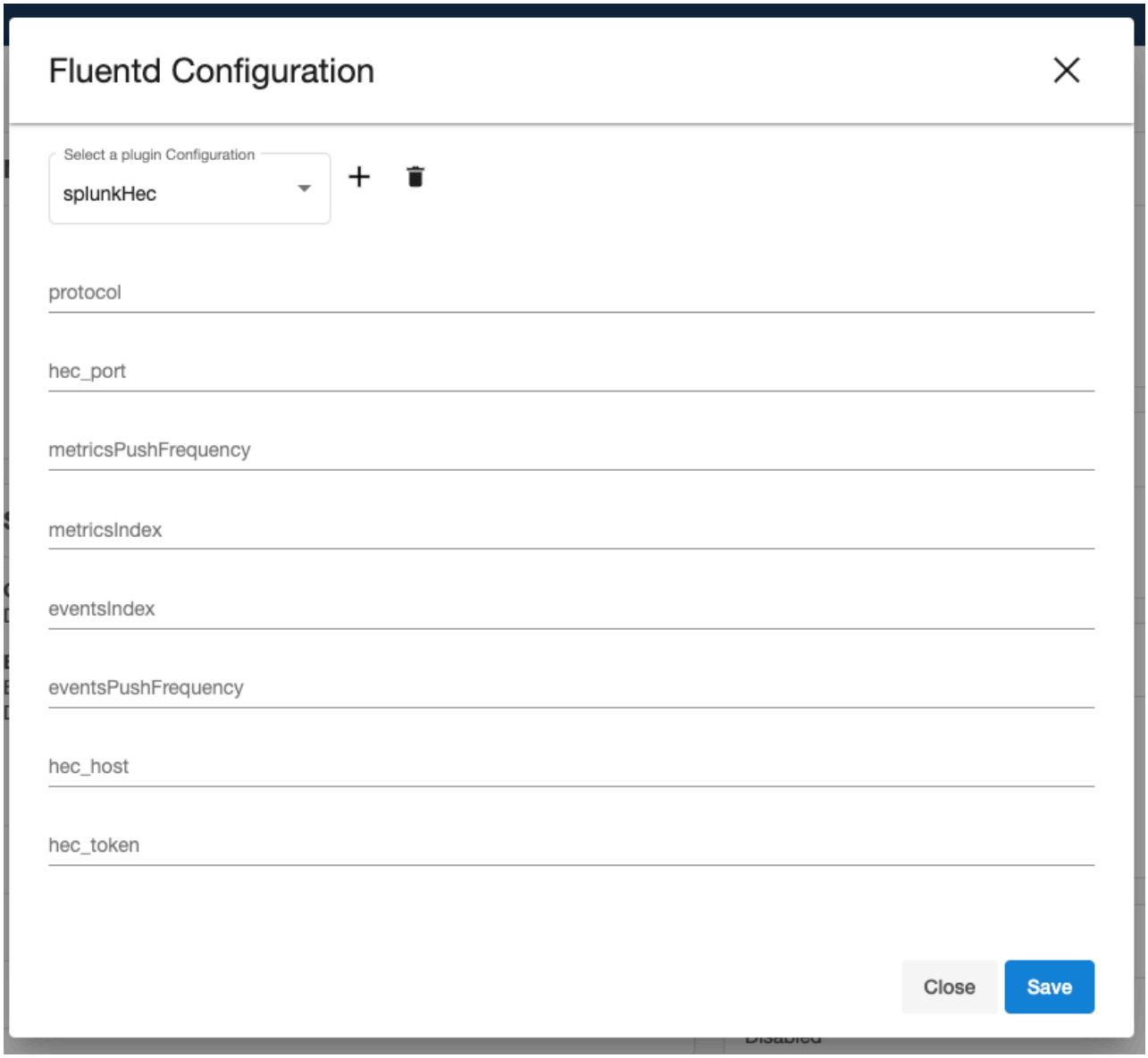


Click the gray box shown in the next window to locate your plugin .far file or drag it to the box to upload it to your Delphix engine:



Limitations

The Delphix Fluentd service supports sending logs to Splunk instances by default through use of the built-in splunkHec plugin. Only one additional plugin can be uploaded to a given Delphix engine at a time. If logs need to be sent to another service requiring a new plugin, the existing one needs to be removed from the engine, and a new plugin uploaded for the new data consumer. Removing an existing plugin can be done from the same window for uploading the plugin by selecting your plugin from the “Select a plugin Configuration” drop-down menu and clicking the trash can icon next to it:



SSL Verification is not currently supported for data consumers other than Splunk and must be disabled.

These limitations may be relaxed in a future Delphix release.

Performance analytics management

Delphix offers various performance analytics tools to help users monitor throughput, latency, and other key metrics. Learn more about how to leverage these tools and how to architect your Delphix deployment for optimal performance.

This section covers the following topics:

Performance analytics

This section covers the following topics:

Performance analytics tool overview

This topic describes the Performance Analytics tool and illustrates some basic uses of it.

Introduction

The performance analytics tool allows introspection into how the Delphix Engine is performing. The introspection techniques it provides are tuned to allow an iterative investigation process, helping to narrow down the cause associated with the performance being measured. Performance analytics information can be accessed through the Delphix Management application, as described in Working with Performance Analytics Graphs in the Graphical User Interface, as well as the CLI and the web services API, as described in other topics in this section. The default statistics that are being collected on the Delphix Engine include CPU utilization, network utilization, and disk, NFS, and iSCSI IO operations (see Performance Analytics Statistics Reference for details).

The performance tool operates with two central concepts: **statistics** and **statistic slices**.

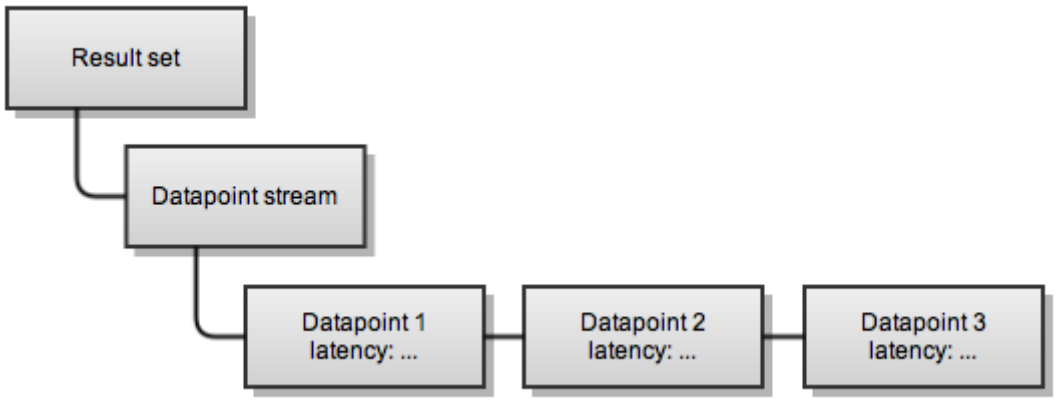
Statistics

Each statistic describes some data that can be collected from the Delphix Engine. The first piece of information a statistic provides is its **type**, which you will use as a handle when creating a statistic slice. It also gives the **minimum collection interval**, which puts an upper bound on the frequency of data collection. The actual data a statistic can collect is described through a set of **axes**, each of which describes one "dimension" of that statistic. For example, the statistic associated with Network File System (NFS) operations has a **latency axis**, as well as an **operation type axis** (among many others), which allows users to see NFS latencies split by whether they were reads or writes.

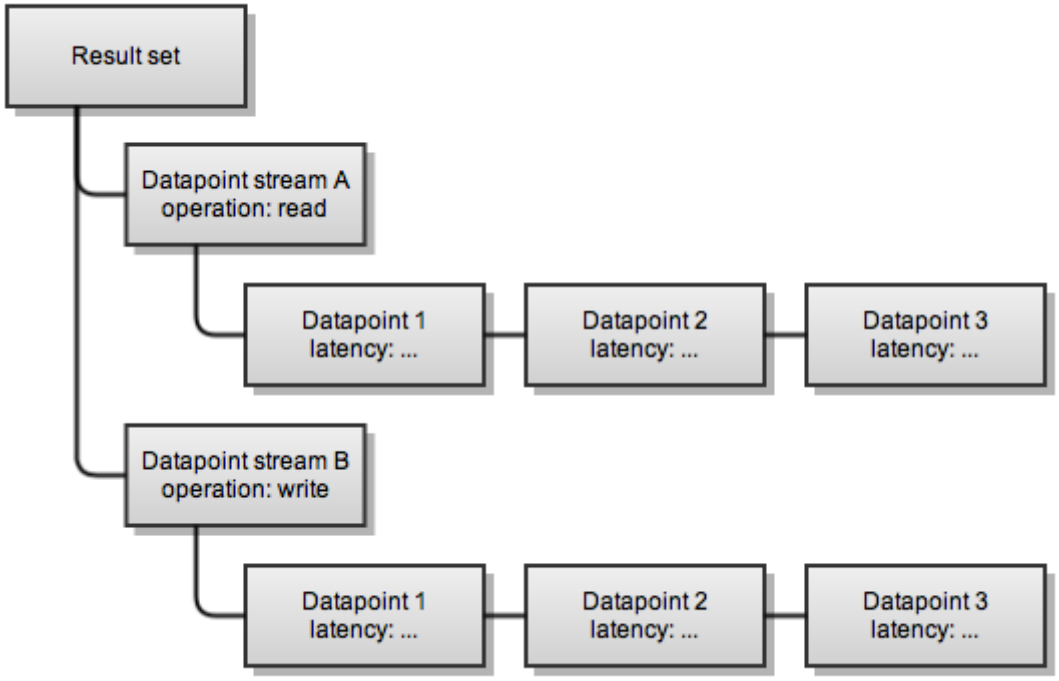
Each axis has some important information embedded in it.

- The **name** of the axis provides a short description of what the axis collects and is used when creating a statistic slice
- A **value** type, which tells you what kind of data will be collected for this axis. The different value types are **integer**, **boolean**, **string**, and **histogram**. The first three are straightforward, but statistic axes with a histogram type can collect a distribution of all the values encountered during each collection interval. This means that instead of seeing an average NFS operation latency every collection interval, you can see a full distribution of operation latencies during that interval. This allows you to see outliers as well as the average, and observe the effects of caching on the performance of your system more easily.
- A **constraint** type, which is only relevant while creating a statistic slice, and will be described in more detail below

One last bit of information that an axis provides makes the most sense after seeing how data points are queried. In the most basic situation, you would only collect one axis of a statistic, such as the latency axis from the NFS operations statistic. When you ask for data, you would get back a data point for every collection interval in the time range you requested. These data points would be grouped into a single stream.



However, if you had collected the operation type axis as well as the latency axis, you would get two streams of datapoints: one for reading operations, and one for write operations.



Because the operation axis applies to many data points, the data points returned are split into two streams, and the operation axis is stored with the top-level stream instead of with each data point in the streams. However, the latency axis will be different for each data point in a stream, so it is not an attribute of the stream, but instead an attribute of the datapoint.

Statistic slices

Statistics describe what data can be collected and are auto-populated by the system, but statistic slices are responsible for actually collecting the data, and you must create them manually when you want to collect some performance data. Each slice is an instantiation of exactly one statistic, and can only gather data that is described by that statistic. "Slices" are so named because each one provides a subset of the information available from the parent statistic it is associated with. A statistic can be thought of as describing the axes of a multidimensional

space, whereas you typically will only want to collect a simpler slice of that space due to the large number of axes available.

When you specify a slice, there are several fields that you must supply:

- The statistic type this slice is associated with. This must be the same type as the statistic of which this is an instantiation.
- The collection interval, which must be greater than the minimum collection interval the parent statistic gives
- The axes of the parent statistic this slice will collect

Finally, a slice can place constraints on axes of its parent statistic, allowing you to limit the data you get back. For instance, if you're trying to narrow down the cause of some high NFS latency outliers, it may be useful to filter out any NFS latencies which are shorter than one second. To do this, you would place a constraint on the latency axis of an NFS operation slice that states that the values must be higher than one second. You can constrain any axis in the same fashion, and each axis' description in the parent statistic gives a constraint type that can be applied to it. This allows you to place different types of constraints on the latency axis (which is a number measured in nanoseconds) than the operation type axis (which is an enum that can take the values "read" or "write").

Persisting analytics data

Data collected by slices is persisted temporarily on the Delphix Engine. Performance data is guaranteed to be available at the finest resolution for six hours, then is compressed to per-minute data and held for seven days, and finally compressed to per-hour data and held for 30 days. If data of a certain resolution will be needed longer than these limits, you should instruct the slice to remember the data permanently until you are done using it. The commands to manage this are listed in the Performance Analytics Tool API Reference.

Working with performance analytics graphs in the graphical user interface

This topic describes the performance analytics graphs that are available in the Delphix Management application, and the controls for changing the views of those graphs.

Accessing the performance analytics graphs

1. Log into the Delphix Management application.
2. In the **Resources** menu, select **Performance Analytics**.
3. Use the controls described below to view statistics and their related graphs.

General graph display and controls

Control Name	Control description	Usage
Chart Tabs	Specifies which graphs/charts are displayed.	Chart tabs have a multiple select drop-down which has the same three options checked by default (CPU, Network, Disk IO). Users can show/hide graphs by checking/ unchecking drop-down options.
Timescale	Controls the time range of data displayed in the graph. Available values are 1 minute, 1 hour, and 1 day. By default, 1 minute is selected.	Timescale options are presented as a drop-down, select from **Minute**, **Hour**, or **Day** to change the Zoom Level.
Pause/Resume	Pause/Resume icons	All charts display live data so users can pause it by clicking the Pause button in the upper right corner of each graph. Once the chart is paused, the Pause button is replaced by the Resume button so a user can click it to see live data again.
Op. Type	Operation Type (used in Disk IO, NFS and iSCSI charts)	Options are presented as a drop-down.
Shown Data Timeline	Displays timestamps of data points in the graph.	
Available Data Timeline	Displays navigable time ranges for historical data.	

Control Name	Control description	Usage
Timeline Selector	Specifies the start and end time for the currently displayed data. The range displayed is controlled by the **Zoom Level**.	Drag the **Timeline Selector** to view statistics for a specific time in the past, or click the scroll bar arrows to view the desired time period. You can also use the slider controls within the Timeline Selector to change the length of time for which data is displayed. When the Timeline Selector is aligned to the right of the timeline, it represents live data that is updated every second. If the Timeline Selector is moved from the right alignment with the timeline, the data displayed is historical and no live updates are displayed. To resume live data updates, move the Timeline Selector back to the right-aligned position representing the current time. The data will be refreshed to the latest data, and live updates will resume every second.
Graph Legend	If more than one set of information is presented on the graph, the Graph Legend displays a description and color for each set and allows a user to toggle that set-off and on. For example, in the network graph there can be multiple network interfaces, and for each network interface the graph displays four statistics (bytes sent, bytes received, packets sent, packets received). When a user toggles off a specific network interface, all four statistics corresponding to that interface are hidden from the screen. The color for lines representing bytes sent and packets sent is the same. Similarly, the color for lines representing bytes received and packets received is the same. This makes it easier to correlate the number of bytes and the number of packets sent/received for a given network interface.	To hide a set of information, click on the set name within the Graph Legend. Data representing that set is removed from the graph, and the set's name is greyed out. To show a set that has been hidden, click on the set name.

Latency, timeline page, and tooltip graph display and controls

Control Name	Control Description	Usage
Timeline Page Left/Right Button	Scrolls Available Data Timeline by a specified time range depending on the current **Zoom Level**.	When the **Zoom Level** is set to **Minute**, click **Timeline Page Left**. The **Available Data Timeline** is changed to show the time period for the previous hour prior.
Graph Value Tooltip	Shows a value, along with the time stamp, for a specific data point.	Mouse over a data point on the graph to view the tooltip.
Latency Range Selector (shown on latency heatmaps only)	Controls the lower and upper limits for displayed latency buckets.	Drag the lower and upper controls to drill down into a specific range of latency buckets. Latency buckets that fall outside of the selected range are summarized, the lower row representing latency buckets that are below the lower limit, and the upper row representing latency buckets that are below the upper limit of the latency range selector. Use Latency Range Selector to view a more detailed distribution of latencies for a specific range.
Latency Outlier Selector (shown on latency heatmaps only)	Hides infrequent latencies (outliers) based on a percentage threshold. Its range is 0%-10%, with a default of 0%. The percentage establishes a threshold below which buckets are considered "outliers" and are hidden from the graph. Each bucket is assigned a percentage based on the ratio of its count vs the maximum count of any bucket in the graph.	Drag the control to the desired percentage threshold.

Performance analytics statistics reference

This topic describes the various performance statistics that are available for the Delphix Engine and how they can be used to analyze and improve performance.

The Delphix Engine is shipped with a default set of statistics that are collected on Delphix Engine virtual appliance, as listed below. The statistics are stored for up to 30 days for historical analysis.

Statistic	Description
CPU Utilization	Total CPU utilization for all CPUs. This statistic includes both kernel and user time.
Network Throughput	Measures throughput in bytes and packets, broken down by sent vs. received data and by the network interface. Each network interface shows four graphed lines: bytes sent, bytes received, packets sent, and packets received. To help easily correlate bytes and packets, the same color is used for both bytes and packet values.
Disk IO	Measures a number of IO operations, and the latencies and throughput of the underlying storage layer. The statistic is represented by the graphs - a column chart for IO operations, a heat map for latency distribution, and a line chart for throughput. IO operations are grouped by reads and write. A shaded rectangle on a latency heat map represents an IO operation (read or write) which falls within a particular time range (bucket). The shading of rectangles depends on the number of IO operations that fall within a particular bucket - the higher the count the darker the shading.
NFS	Measures a number of IO operations and the latencies and throughput of the NFS server layer in the Delphix Engine. Its graphical representation is similar to the Disk IO graph. It is useful to diagnose the performance of dSources and VDBs that use NFS mounts (Oracle, PostgreSQL).
iSCSI	Measures the number of IO operations, and the latencies and throughput, of the iSCSI server layer in the Delphix Engine. Its graphical representation is similar to the Disk IO graph. It is useful to diagnose the performance of Microsoft SQL Server dSources and VDBs.

Performance analytics tool API reference

This topic describes basic commands and command syntax for using the Performance Analytics tool.

Statistic types

More detailed information can be found about each statistic type through the command-line interface (CLI) and webservices API, but the following table provides more information about how similar I/O stack statistic types relate to each other.

Statistic type	Description	Axis name	Axis description	Axis value type
NFS_OPS	Provides information about Network File System operations. This is the entry point to the Delphix Engine for all Oracle database file accesses.	op	I/O operation type	STRING
		path	Path of the affected file	STRING
		size	I/O sizes in bytes	HISTOGRAM
		avgLatency	Average I/O latency in nanoseconds	INTEGER
		cached	Whether reads were cached	BOOLEAN
		latency	I/O latencies in nanoseconds	HISTOGRAM
		count	Number of I/O operations	INTEGER
		client	Address of the client	STRING
		throughput	I/O throughput in bytes	INTEGER
		sync	Whether writes were synchronous	BOOLEAN

Statistic type	Description	Axis name	Axis description	Axis value type
iSCSI_OPS	Provides information about iSCSI operations. This is the entry point to the Delphix Engine for all SQL Server file accesses.	Same axes as NFS_OPS, except for path, cached, and sync.		
VFS_OPS	This layer sits immediately below NFS_OPS and iSCSI_OPS. It should give almost exactly the same latencies, assuming no unexpected behavior is occurring.	Same axes as NFS_OPS, except for client.		
DxFS_OPS	This layer sits immediately below VFS_OPS, and the two of them should give almost exactly the same latencies.	Same axes as VFS_OPS.		
DxFS_IO_QUEUE_OPS	This layer sits below DxFS_OPS, but the latencies will differ from that layer because this layer batches together operations to increase throughput.	op	I/O operation type	STRING
		count	Number of I/O operations	INTEGER
		size	I/O sizes in bytes	HISTOGRAM
		avgLatency	Average I/O latency in nanoseconds	INTEGER
		latency	I/O latencies in nanoseconds	HISTOGRAM
		throughput	I/O throughput in bytes	INTEGER
		priority	Priority of the I/O	STRING

Statistic type	Description	Axis name	Axis description	Axis value type
DISK_OPS	This layer sits below DxFS_IO_QUEUE_OPS at the bottom of the I/O stack and measures interactions between the Delphix Engine and disks.	op	I/O operation type	STRING
		count	Number of I/O operations	INTEGER
		size	I/O sizes in bytes	HISTOGRAM
		avgLatency	Average I/O latency in nanoseconds	INTEGER
		latency	I/O latencies in nanoseconds	HISTOGRAM
		throughput	I/O throughput in bytes	INTEGER
		error	Whether the I/O resulted in an error	BOOLEAN
		device	Device the I/O was issued to	STRING
CPU_UTIL	This is unrelated to the layers of the I/O stack. It measures CPU utilization on the Delphix Engine.	idle	Idle time in milliseconds (showAxes command may incorrectly state nanoseconds)	INTEGER
		user	User time in milliseconds (showAxes command may incorrectly state nanoseconds)	INTEGER
		kernel	Kernel time in milliseconds (showAxes command may incorrectly state nanoseconds)	INTEGER

Statistic type	Description	Axis name	Axis description	Axis value type
		dtrace	DTrace time in milliseconds (showAxes command may incorrectly state nanoseconds) Subset of time in kernel	INTEGER
		cpu	Which CPU was utilized	INTEGER
NETWORK_INTERFACE_UTIL	Network interface utilization on the Delphix Engine.	inBytes	Number of bytes received	INTEGER
		inPackets	Number of packets received	INTEGER
		outBytes	Number of bytes transmitted	INTEGER
		outPackets	Number of packets transmitted	INTEGER
		networkInterface	Which network interface was utilized	STRING
TCP_STATS	Statistics for all established TCP connections on the Delphix Engine.	localAddress	Local address for the TCP connection	STRING
		localPort	Local port for the TCP connection	INTEGER
		remoteAddress	Remote address for the TCP connection	STRING
		remotePort	Remote port for the TCP connection	INTEGER
		inBytes	Data bytes received	INTEGER

Statistic type	Description	Axis name	Axis description	Axis value type
		outBytes	Data bytes transmitted	INTEGER
		receiveWindowSize	The size of the local receive window	INTEGER
		sendWindowSize	The size of the peer's receive window	INTEGER
		congestionWindowSize	The size of the local congestion window	INTEGER
		retransmittedBytes	Bytes retransmitted	INTEGER
		inUnorderedBytes	Number of bytes received out of order. This is a subset of the "inBytes" value	INTEGER
		unacknowledgedBytes	Number of bytes sent but unacknowledged	INTEGER
		roundTripTime	Smoothed average round-trip time in microseconds	INTEGER

Statistic axis value types

Values are returned when a slice's data is queried. Each axis has a value type, which specifies how the data will be returned.

Value type	Description
INTEGER	The value is returned as an integer. For information about what units the integer is measured in, read the documentation for the related datapoint or datapoint stream type.
BOOLEAN	The value is returned as a boolean.

Value type	Description
STRING	The value is returned as a string. This is used for enum values as well, although the set of strings that can be returned is limited.

HISTOGRAM	The value is returned as a log-scale histogram. The histogram has size buckets whose minimum and maximum value get doubled. Histograms are returned as JSON maps, where the keys are the minimum value in a bucket and the values are the height of each bucket.
	Here is an example histogram. Notice that buckets with a height of zero are not included in the JSON object and that keys and values are represented as strings.
	`{`
	`"32768": "10",`
	`"65536": "102",`
	`"262144": "15",`
	`"524288": "2"`
	`}`

Axis constraints are used to limit the data which a slice can collect. Each axis specifies a constraint type which can be used to limit that axis' values.

Constraint type	Description
`BooleanConstraint`	A superclass in which constraints on boolean values must extend. Currently, the only subclass is `BooleanEqualConstraint`, which requires that a boolean axis equal either true or false (depending on user input).
`EnumConstraint`	A superclass in which constraints on enum values must extend. Currently, the only subclass is `EnumEqualConstraint`, which requires that an enum axis be equal to a user-specified value.
`IntegerConstraint`	A superclass in which constraints on integer values must extend. Subclasses include `IntegerLessThanConstraint`, `IntegerGreaterThanConstraint`, and `IntegerEqualConstraint`, which map to the obvious comparators for integers.
`NullConstraint`	This class signifies that an axis cannot be constrained. This makes the most sense for axes that provide an average value - placing a constraint on an average doesn't make sense because you are not able to include or discard a particular operation based on what its effects would be on the average of all operations.

Constraint type	Description
`PathConstraint`	A superclass in which constraints on file path values must extend. Currently, the only subclass is `PathDescendantConstraint`, which requires that a path value must be a descendant of the specified path (it must be contained within it). This only applies to paths on the Delphix Engine itself, and all paths used must be canonical Unix paths starting from the root of the filesystem.
`StringConstraint`	A superclass in which constraints on string values must extend. Currently, the only subclass is `StringEqualsConstraint`, which requires that a string value must equal a user-specified string.

Statistic slice commands

Command	Description and Usage Examples
`getData`	This is used to fetch data from a statistic slice which has been collecting data for a while. It returns a datapoint set, which is composed of datapoint streams, which contain datapoints. For a full description, see the Performance Analytics Tool Overview.
`rememberRange`	This is used to ensure that data collected during an ongoing investigation doesn't get deleted unexpectedly. If this is not used, data is only guaranteed to be persisted for 24 hours. If it is used, data will be remembered until a corresponding call to `stopRememberingRange` is made.
`stopRememberingRange`	This is used to allow previously-remembered data to be forgotten. The data will be forgotten on the same schedule as brand new data, so you will have at least 24 hours before data which you have stopped remembering is deleted. This undoes the `rememberRange` operation.
`pause`	This command pauses the collection of a statistic slice, causing no data to be collected until `resume` is called.
`resume`	This command resumes the collection of a statistic slice, undoing a `pause` operation.

Performance analytics case study: using a single statistic

Overview

The Delphix Engine uses the Network File System (NFS) as the transport for Oracle installations. An increase in the NFS latency could be causing sluggishness in your applications running on top of Virtual Databases. This case study illustrates how this pathology can be root caused using the analytics infrastructure. This performance investigation uses one statistic to debug the issue, and utilizes the many axes of that statistic to filter down the probable cause of the issue. This technique uses an approach of iteratively drilling down by inspecting new axes of a single statistic and filtering the data to only include information about the operations that appear slow. This technique is valuable for determining which use patterns of a resource might be causing the system to be sluggish. If you isolate a performance issue using this approach but aren't sure what is causing it or how to fix it, Delphix Support can provide assistance for your investigation.

The following example inspects the statistic which provides information about NFS I/O operations on the Delphix Engine. This statistic can be collected a maximum of once every second, and the axes it can collect, among others, are:

- **latency**, a histogram of wait times between NFS requests and NFS responses
- **size**, a histogram of the NFS I/O sizes requested
- **op**, whether the NFS requests were reads or writes
- **client**, the network address of the NFS client which was making requests

> Roughly the same performance information can be obtained from the iSCSI interface as well.

Investigation

Because the NFS layer sits above the disk layer, all NFS operations that use the disk synchronously (synchronous writes and uncached reads) will have latencies that are slightly higher than those of their corresponding disk operations. Usually, because disks have very high seek times compared to the time the NFS server spends on CPU, disk operations are responsible for almost all of the latency of these NFS operations. In the graphical representation, you can see this by looking at how the slower cluster of NFS latencies (around 2ms-8ms) have similar latencies to the median of the disk I/O (around 2ms-4ms). Another discrepancy between the two plots is that the number of disk operations is much lower than the corresponding number of NFS operations. This is because the Delphix filesystem batches together write operations to improve performance.

If database performance is not satisfactory and almost all of the NFS operation time is spent waiting for the disks, it suggests that the disk is the slowest piece of the I/O stack. In this case, disk resources (the number of IOPS to the disks, the free space on the disks, and the disk throughput) should be investigated more thoroughly to determine if adding more capacity or a faster disk would improve performance. However, care must be taken when arriving at these conclusions, as a shortage of memory or a recently-rebooted machine can also cause the disk to be used more heavily due to fewer cache hits.

Sometimes, disk operations will not make up all of the latency, which suggests that something between the NFS server and the disk (namely, something in the Delphix Engine) is taking a long time to complete its work. If this is the case, it is valuable to check whether the Delphix Engine is resource-constrained, and the most common areas of constraint internal to the Delphix Engine are CPU and memory. If either of those is too limited, you should investigate whether expanding the resource would improve performance. If no resources appear to be constrained or more investigation is necessary to convince you that adding resources would help the issue, Delphix Support is available to help debug these issues.

While using this technique, you should take care to recognize the limitations that caching places on how performance data can be interpreted. In this example, the Delphix Engine uses a caching layer for the data it stores, so asynchronous NFS writes will not go to disk quickly because they are being queued into larger batches, and

cached NFS reads won't use the disk at all. This causes these types of NFS operations to return much more quickly than any disk operations are able to, resulting in a very large number of low-latency NFS operations in the graph above. For this reason, caching typically creates a bimodal distribution in the NFS latency histograms, where the first cluster of latencies is associated with operations that only hit the cache, and the second cluster of latencies is associated with fully or partially uncached operations. In this case, cached NFS operations should not be compared to the disk latencies because they are unrelated. It is possible to use techniques described in the first example to filter out some of the unrelated operations to allow a more accurate mapping between disk and NFS latencies.

1. Begin the performance investigation by examining some high-level statistics such as **latency**.
    a. Create a slice with statistic type **NFS_OPS.**
    b. Set the slice to collect the **latency** axis.
    c. Do not add any constraints.
    d. Set the collection interval. Anything over one second will work, but ten seconds gives good data resolution and will not use a lot of storage to persist the data that is collected. The rest of this example will assume a collection period of ten seconds for all other slices, but any value could be used.

```
/analytics

create
set name=step1
set statisticType=NFS_OPS
set collectionInterval=10
set collectionAxes=latency
commit
```

This will collect a time-series of histograms describing NFS latencies as measured from inside the Delphix Engine, where each histogram shows how many NFS I/O operations fell into each latency bucket during every ten-second interval. After a short period of time, read the data from the statistic slice:

```
select step1
getData
setopt format=json
commit
setopt format=text
```

The `setopt` steps are optional but allow you to see the output better via the CLI. The output looks like this:

```
{
    "type": "DatapointSet",
    "collectionEvents": [],
    "datapointStreams": [{
        "type": "NfsOpsDatapointStream",
        "datapoints": [{
            "type": "IoOpsDatapoint",
            "latency": {
                "32768": "16",
                "65536": "10"
```

```
            },
            "timestamp": "2013-05-14T15:51:40.000Z"
        }, ...]
    }],
    "resolution": 10
}
```

The data is returned as a set of datapoint streams. Streams hold the fields which are shared by all the datapoints they contain. Later on, in this example, the **opt** and **client** fields will be added to the streams, and multiple streams will be returned. Streams are described in more detail in Performance Analytics Tool Overview. The `resolution` field indicates the number of seconds that corresponds to each datapoint, which in our case matches the requested `collectionInterval`. The `collectionEvents` field is not used in this example, but lists when the slice was paused and resumed, to distinguish between moments when no data was collected because the slice was paused, and moments when there was no data to collect.

2. If the latency distributions show some slow NFS operations, the next step would be to determine whether the slow operations are reads or writes.
    a. Specify a new **NFS_OPS** slice to collect this by collecting the **op** and **latency** axes.
    b. To limit output to the long-running operations, create a constraint on the **latency**axis that prohibits the collection of data on operations with latency less than 100ms.

```
/analytics

create
set name=step2
set statisticType=NFS_OPS
set collectionInterval=10
set collectionAxes=op,latency

edit axisConstraints.0
set axisName=latency
set type=IntegerGreaterThanConstraint
set greaterThan=100000000
back

commit
```

The `greaterThan` field is 100ms converted into nanoseconds.

Reading the data proceeds in the same way as the first step, but there will be two streams of datapoints, one where `op=write,` and one where `op=read`.

info : Because we constrained output to operations with latencies higher than 100ms, none of the latency histograms will all have any buckets for latencies lower than 100ms.

3. After inspecting the two data streams, you might find that almost all slow operations are writes, so it could be valuable to determine which clients are requesting the slow writes, and how large each of the writes is.
    a. To collect this data, create a new **NFS_OPS** slice which collects the **size** and **client** axes.
    b. Add constraints ensuring that the **op** axis should be constrained to only collect data for **write** operations, and the **latency** axis should be constrained to filter operations taking less than 100ms.
    info : Because the constraint on the op axis dictates that it will always have the value write, it is not necessary to collect the op axis anymore.

```
/analytics

create
set name=step3
set statisticType=NFS_OPS
set collectionInterval=10
set collectionAxes=size,client

edit axisConstraints.0
set type=IntegerGreaterThanConstraint
set axisName=latency
set greaterThan=100000000
back

edit EnumEqualConstraint
set type=StringEqualConstraint
set axisName=op
set equals=write
back

commit
```

Reading the data proceeds in the same way as the first two steps, but there will be one stream for every NFS client. The dataset collected by this will consist of a set of streams, one corresponding to each NFS client, and each stream will be a time-series of histograms showing write sizes that occurred during each ten-second interval.
Continuing to use this approach will allow you to narrow down the slow writes to a particular NFS client, and you may be able to tune that client in some way to speed it up.

Performance analytics case study: using multiple statistics

Overview

This case study illustrates an investigation involving more than one metric. In typical performance investigations, you will need to peel out multiple layers of the stack in order to observe the component causing the actual performance pathology. This case study specifically examines sluggish application performance caused due to slow IO responses from the disk subsystem. This example will demonstrate a technique of looking at the performance of each layer in the I/O stack to find which layer is responsible for the most latency, then looking for constrained resources that the layer might need to access. This technique is valuable for finding the most-constrained resource in the system, potentially giving actionable information about resources that can be expanded to increase performance.

For the following example, we will inspect latency at two layers: the Network File System (NFS) layer on the Delphix Engine, and the disk layer below it. Both of these layers provide the **latency** axis, which gives a histogram of wait times for the clients of each layer.

Investigation

The analytics infrastructure enables users to observe the latency of multiple layers of the software stack. This investigation will examine the latency of both layers, and then draw conclusions about the differences between the two.

Setup

To measure this data, create two slices. When attempting to correlate data between two different statistics, it can be easier to determine causation when collecting data at a relatively high frequency. The fastest that each of these statistics will collect data is once per second, so that is the value used.

1. A slice collecting the **latency** axis for the statistic type **NFS_OPS.**

```
/analytics

create
set name=slice1
set statisticType=NFS_OPS
set collectionInterval=1
set collectionAxes=latency
commit
```

2. A slice collecting the **latency** axis for the statistic type **DISK_OPS.**

```
/analytics

create
set name=slice2
set statisticType=DISK_OPS
set collectionInterval=1
set collectionAxes=latency
commit
```

After a short period of time, read the data from the first statistic slice.

```
select slice2
getData
setopt format=json
commit
setopt format=text
```

The same process works for the second slice. The `setopt` steps allow you to see the output better via the CLI. The output for the first slice might look like this:


```
{
    "type": "DatapointSet",
    "collectionEvents": [],
    "datapointStreams": [{
        "type": "NfsOpsDatapointStream",
        "datapoints": [{
            "type": "IoOpsDatapoint",
            "latency": {
                "512": "100",
                "1024": "308",
                "2048": "901",
                "4096": "10159",
                "8192": "2720",
                "16384": "642",
                "32768": "270",
                "65536": "50",
                "131072": "11",
                "524288": "64",
                "1048576": "102",
                "2097152": "197",
                "4194304": "415",
                "8388608": "320",
                "16777216": "50",
                "33554432": "20",
                "67108864": "9",
                "268435456": "2"
            },
            "timestamp": "2013-05-14T15:51:40.000Z"
        }, {
            "type": "IoOpsDatapoint",
            "latency": {
                "512": "55",
                "1024": "130",
                "2048": "720",
                "4096": "6500",
                "8192": "1598",
                "16384": "331",
                "32768": "327",
                "65536": "40",
                "131072": "14",
                "262144": "87",
                "524288": "42",
```

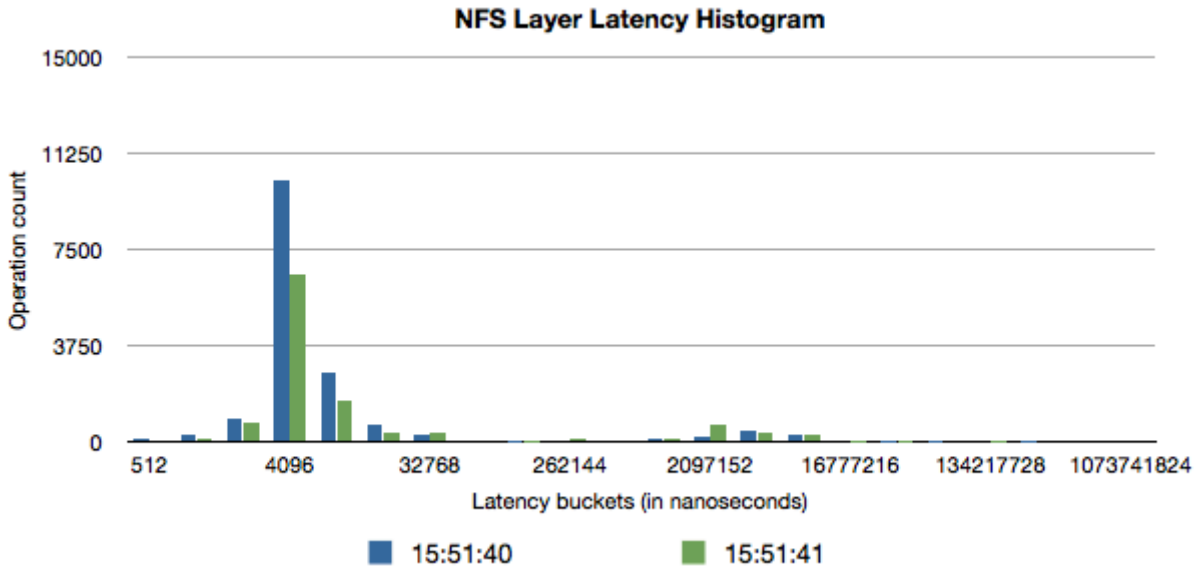
```
                "1048576": "97",
                "2097152": "662",
                "4194304": "345",
                "8388608": "280",
                "16777216": "22",
                "33554432": "15",
                "134217728": "1"
            },
            "timestamp": "2013-05-14T15:51:41.000Z"
        }, ...]
    }],
    "resolution": 1
}
```

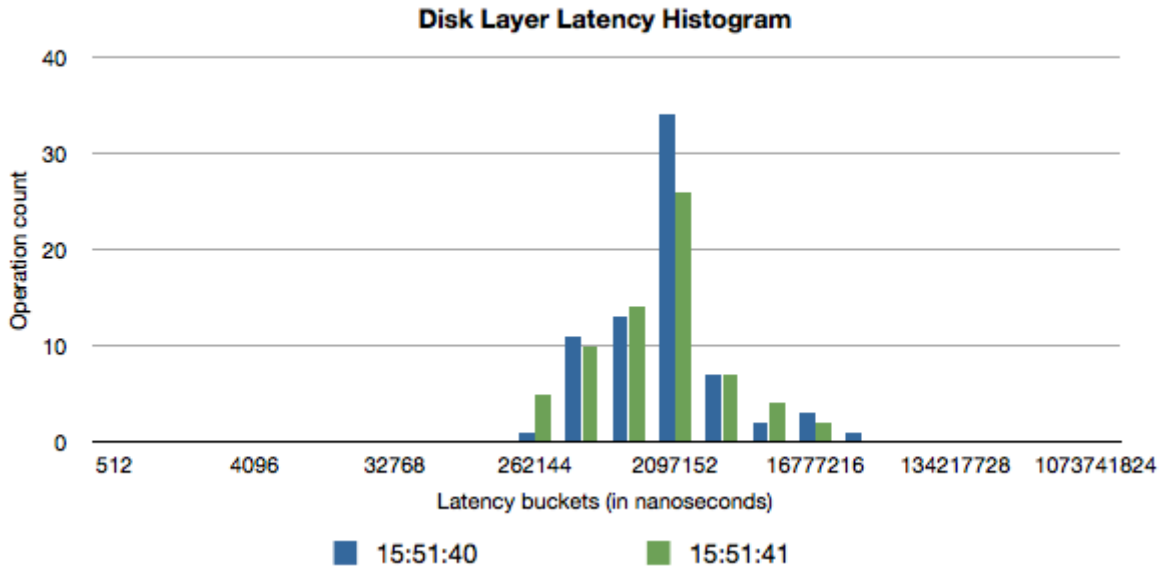
For the second slice, it might look like this:

```
{
    "type": "DatapointSet",
    "collectionEvents": [],
    "datapointStreams": [{
        "type": "DiskOpsDatapointStream",
        "datapoints": [{
            "type": "IoOpsDatapoint",
            "latency": {
                "262144": "1",
                "524288": "11",
                "1048576": "13",
                "2097152": "34",
                "4194304": "7",
                "8388608": "2",
                "16777216": "3",
                "33554432": "1"
            },
            "timestamp": "2013-05-14T15:51:40.000Z"
        }, {
            "type": "IoOpsDatapoint",
            "latency": {
                "262144", "5",
                "524288", "10",
                "1048576", "14",
                "2097152", "26",
                "4194304", "7",
                "8388608", "4",
                "16777216", "2"
            },
            "timestamp": "2013-05-14T15:51:41.000Z"
        }, ...]
    }],
    "resolution": 1
}
```

The data is returned as a set of datapoint streams. Streams hold the fields that would otherwise be shared by all the datapoints they contain, but only one is used in this example because there are no such fields. Streams are discussed in more detail in the Performance Analytics Tool Overview. The `resolution` field indicates how many seconds each datapoint corresponds to, which in our case matches the requested `collectionInterval`. The `collectionEvents` field is not used in this example, but lists when the slice was paused and resumed to distinguish between moments when no data was collected because the slice was paused, and moments when there was no data to collect.

Graphically, these four histograms across two seconds look like this:





Analysis

Because the NFS layer sits above the disk layer, all NFS operations that use the disk synchronously (synchronous writes and uncached reads) will have latencies that are slightly higher than those of their corresponding disk operations. Usually, because disks have very high seek times compared to the time the NFS server spends on CPU, disk operations are responsible for almost all of the latency of these NFS operations. In the graphical representation, you can see this by looking at how the slower cluster of NFS latencies (around 2ms-8ms) have similar latencies to the median of the disk I/O (around 2ms-4ms). Another discrepancy between the two plots is that the number of disk operations is much lower than the corresponding number of NFS operations. This is because the Delphix filesystem batches together write operations to improve performance.

If database performance is not satisfactory and almost all of the NFS operation time is spent waiting for the disks, it suggests that the disk is the slowest piece of the I/O stack. In this case, disk resources (the number of IOPS to the disks, the free space on the disks, and the disk throughput) should be investigated more thoroughly to determine if adding more capacity or a faster disk would improve performance. However, care must be taken when arriving at these conclusions, as a shortage of memory or a recently-rebooted machine can also cause the disk to be used more heavily due to fewer cache hits.

Sometimes, disk operations will not make up all of the latency, which suggests that something between the NFS server and the disk (namely, something in the Delphix Engine) is taking a long time to complete its work. If this is the case, it is valuable to check whether the Delphix Engine is resource-constrained, and the most common areas of constraint internal to the Delphix Engine are CPU and memory. If either of those is too limited, you should investigate whether expanding the resource would improve performance. If no resources appear to be constrained or more investigation is necessary to convince you that adding resources would help the issue, Delphix Support is available to help debug these issues.

While using this technique, you should take care to recognize the limitations that caching places on how performance data can be interpreted. In this example, the Delphix Engine uses a caching layer for the data it stores, so asynchronous NFS writes will not go to disk quickly because they are being queued into larger batches, and cached NFS reads won't use the disk at all. This causes these types of NFS operations to return much more quickly than any disk operations are able to, resulting in a very large number of low-latency NFS operations in the graph above. For this reason, caching typically creates a bimodal distribution in the NFS latency histograms, where the first cluster of latencies is associated with operations that only hit the cache, and the second cluster of latencies is associated with fully or partially uncached operations. In this case, cached NFS operations should not be compared to the disk latencies because they are unrelated. It is possible to use techniques described in the first example to filter out some of the unrelated operations to allow a more accurate mapping between disk and NFS latencies.

1. Begin the performance investigation by examining some high-level statistics such as **latency**.
    a. Create a slice with statistic type **NFS_OPS.**
    b. Set the slice to collect the **latency** axis.
    c. Do not add any constraints.
    d. Set the collection interval. Anything over one second will work, but ten seconds gives good data resolution and will not use a lot of storage to persist the data that is collected. The rest of this example will assume a collection period of ten seconds for all other slices, but any value could be used.

```
/analytics

create
set name=step1
set statisticType=NFS_OPS
set collectionInterval=10
set collectionAxes=latency
commit
```

This will collect a time series of histograms describing NFS latencies as measured from inside the Delphix Engine, where each histogram shows how many NFS I/O operations fell into each latency bucket during every ten-second interval. After a short period of time, read the data from the statistic slice:

```
select step1
getData
setopt format=json
commit
setopt format=text
```

The `setopt` steps are optional but allow you to see the output better via the CLI. The output looks like this:

```
{
    "type": "DatapointSet",
    "collectionEvents": [],
    "datapointStreams": [{
        "type": "NfsOpsDatapointStream",
        "datapoints": [{
            "type": "IoOpsDatapoint",
            "latency": {
                "32768": "16",
                "65536": "10"
            },
            "timestamp": "2013-05-14T15:51:40.000Z"
        }, ...]
    }],
    "resolution": 10
}
```

The data is returned as a set of datapoint streams. Streams hold the fields which are shared by all the data points they contain. Later on, in this example, the **opt** and **client** fields will be added to the streams, and multiple streams will be returned. Streams are described in more detail in Performance Analytics Tool Overview. The `resolution` field indicates the number of seconds that corresponds to each datapoint, which in our case matches the requested `collectionInterval`. The `collectionEvents` the field is not used in this example, but lists when the slice was paused and resumed, to distinguish between moments when no data was collected because the slice was paused, and moments when there was no data to collect.

2. If the latency distributions show some slow NFS operations, the next step would be to determine whether the slow operations are reads or write.
    a. Specify a new **NFS_OPS** slice to collect this by collecting the **op** and **latency** axes.
    b. To limit output to the long-running operations, create a constraint on the **latency**axis that prohibits the collection of data on operations with latency less than 100ms.

```
/analytics

create
```

```
set name=step2
set statisticType=NFS_OPS
set collectionInterval=10
set collectionAxes=op,latency

edit axisConstraints.0
set axisName=latency
set type=IntegerGreaterThanConstraint
set greaterThan=100000000
back

commit
```

The `greaterThan` the field is 100ms converted into nanoseconds.

Reading the data proceeds in the same way as the first step, but there will be two streams of data points, one where `op=write,` and one where `op=read` .

Info : Because we constrained output to operations with latencies higher than 100ms, none of the latency histograms will all have any buckets for latencies lower than 100ms.

3. After inspecting the two data streams, you might find that almost all slow operations are writes, so it could be valuable to determine which clients are requesting the slow writes, and how large each of the writes is.
    a. To collect this data, create a new **NFS_OPS** slice which collects the **size** and **client** axes.
    b. Add constraints ensuring that the **op** axis should be constrained to only collect data for **write** operations, and the **latency**axis should be constrained to filter operations taking less than 100ms.
    Info: Because the constraint on the op axis dictates that it will always have the value write, it is not necessary to collect the op axis anymore.

```
/analytics

create
set name=step3
set statisticType=NFS_OPS
set collectionInterval=10
set collectionAxes=size,client

edit axisConstraints.0
set type=IntegerGreaterThanConstraint
set axisName=latency
set greaterThan=100000000
back

edit EnumEqualConstraint
set type=StringEqualConstraint
set axisName=op
set equals=write
back

commit
```

Reading the data proceeds in the same way as the first two steps, but there will be one stream for every NFS client. The dataset collected by this will consist of a set of streams, one corresponding to each NFS client, and each stream will be a time series of histograms showing write sizes that occurred during each ten-second interval.

Continuing to use this approach will allow you to narrow down the slow writes to a particular NFS client, and you may be able to tune that client in some way to speed it up.

Storage performance configuration options

This section covers the following topics:

- Optimal storage configuration parameters for the Delphix engine
- Storage performance test tool (fio)
- Storage performance expectations and troubleshooting
- Storage performance test tool notes - restricted

Optimal storage configuration parameters for the Delphix engine

This topic describes minimum capacity and throughput requirements for storage devices used with the Delphix Engine.

Storage for the Delphix Engine must be able to sustain the aggregated Input/Output Operations Per Second (IOPS) and throughput (MBPS) requirements of all its Virtual Databases. Throughput required for data source synchronization (SnapSync and LogSync) must also be supported.

The Delphix Engine requires storage for:

Item	Description
A copy of each Source Database	The copies are compressed.
Unique Block Changes in VDBs	When changes are made to a VDB, the Delphix Engine stores the changes in new blocks associated with only that VDB. The new blocks are compressed.
Timeflow for dSources and VDBs	The TimeFlow kept for each dSource and VDB comprises snapshots of the database (blocks changed since the previous snapshot) and archive logs. The retention period for this history of changes is determined by policies established for each dSource and VDB. The TimeFlow is compressed.

In addition to the storage for these items, the Delphix Engine requires 30% free space in its storage for the best performance. See An Overview of Capacity and Performance Information and related topics for more details on managing capacity for the Delphix Engine.

Best practices for storage performance include:

- Initial storage is equal to the size of the physical source databases. For high redo rates and/or high DB change rates, allocate an additional 10-20% storage.
- Add storage when storage capacity approaches 30% free
- Use physical LUNS allocated from storage pools or RAID groups that are configured for availability
- Never share physical LUNs between the Delphix Engine and other storage clients.
- Keep all physical LUNs the same size.
- Provision storage using VMDKs or RDMs operating in virtual compatibility mode.
- VMDKs should be **Thick Provisioned, Lazy Zeroed.** The underlying physical LUNs can be thin provisioned.
- Physical LUNs used for RDMs should be thick provisioned.
- Measure or estimate the required IOPS and manage the storage disks to provide this capacity. It is common to use larger numbers of spindles to provide the IOPS required.
- Physical LUNs carved from RAID 1+0 groups or pools with dedicated spindles provide higher IOPS performance than other configurations
- Maximize Delphix Engine vRAM for a larger system cache to service reads

Example

There are two production dSources, totaling 5 TB in size. 5 VDBs will be created for each. The Sum of read and write rates on the production source database is moderate (1000 IOPS), the sum of VDB read rate is moderate (950 IOPS), and the VDB update rate is low (50 IOPS).

- Initial storage equal to 5TB, provisioned as 5 x 1 TB physical LUNs, Thin Provisioned. Allow for expansion of the LUNs to 2TB.

- Provision as 5 x 950 GB Virtual Disks. VMDKs must be Thick Provisioned, Lazy Zeroed. Using 1 TB LUNs allows expansion to 2 TB (ESX 5.1 limit).
- The storage provisioned to the Delphix Engine storage must be able to sustain 1000 IOPs (950 + 50). For this reason, each physical LUN provisioned to the Delphix Engine must be capable of sustaining 200 IOPs. IOPs on the source databases are not relevant to the Delphix Engine.
- 64GB Delphix Engine vRAM for a large system cache

Storage performance test tool (fio)

Overview

This fio-based Storage Performance Tool executes a synthetic workload to evaluate the performance characteristics of the storage assigned to the Delphix Engine. The Storage Performance Tool is a feature that is only available from the command-line

Prerequisites

Prior to setting up the Delphix Engine, the admin can login to the Delphix CLI using a **sysadmin** account to launch the Storage Performance Tool. Because the test is destructive, it will only run against storage which has not been allocated to Delphix for use by the engine.

Running the storage test via CLI

1. Login as the sysadmin user to the Delphix Engine CLI.
    a. If the Delphix Engine has not been setup yet, the **network setup** prompt appears. Discard the command.

    ```
    delphix network setup update *> discard delphix>
    ```

2. Create a storage test.

    ```
    delphix> storage test
    delphix storage test> create
    delphix storage test create *>
    ```

3. Use 'get' to see other optional arguments. Modify the test parameters as needed and **commit** to start the test.

    ```
    delphix storage test create *> get
       type: StorageTestParameters
       devices: (unset)
       duration: 120
       initializeDevices: true
       initializeEntireDevice: false
       testRegion: 128GB
       tests: ALL
    delphix storage test create *> commit
        STORAGE_TEST-1
        Dispatched job JOB-1
        STORAGE_TEST_EXECUTE job started for "SYSTEM".
        Initializing storage test.
        Starting storage device initialization.
        ETA: 1:28:44.
        Storage device initialization complete.
        Starting storage benchmarking.
        Starting random read workload with 4 KB block size and 8 jobs.
        Starting random read workload with 4 KB block size and 16 jobs.
    ```

```
    Starting random read workload with 4 KB block size and 32 jobs.
    Starting random read workload with 4 KB block size and 64 jobs.
    Starting random read workload with 8 KB block size and 8 jobs.
    Starting random read workload with 8 KB block size and 16 jobs.
    Starting random read workload with 8 KB block size and 32 jobs.
    Starting random read workload with 8 KB block size and 64 jobs.
    Starting sequential write workload with 1 KB block size and 4 jobs.
    Starting sequential write workload with 4 KB block size and 4 jobs.
    Starting sequential write workload with 8 KB block size and 4 jobs.
    Starting sequential write workload with 16 KB block size and 4 jobs.
    Starting sequential write workload with 32 KB block size and 4 jobs.
    Starting sequential write workload with 64 KB block size and 4 jobs.
    Starting sequential write workload with 128 KB block size and 4 jobs.
    Starting sequential write workload with 1024 KB block size and 4 jobs.
    Starting sequential write workload with 1 KB block size and 16 jobs.
    Starting sequential write workload with 4 KB block size and 16 jobs.
    Starting sequential write workload with 8 KB block size and 16 jobs.
    Starting sequential write workload with 16 KB block size and 16 jobs.
    Starting sequential write workload with 32 KB block size and 16 jobs.
    Starting sequential write workload with 64 KB block size and 16 jobs.
    Starting sequential write workload with 128 KB block size and 16 jobs.
    Starting sequential write workload with 1024 KB block size and 16 jobs.
    Starting sequential read workload with 64 KB block size and 4 jobs.
    Starting sequential read workload with 64 KB block size and 8 jobs.
    Starting sequential read workload with 64 KB block size and 16 jobs.
    Starting sequential read workload with 64 KB block size and 32 jobs.
    Starting sequential read workload with 64 KB block size and 64 jobs.
    Starting sequential read workload with 128 KB block size and 4 jobs.
    Starting sequential read workload with 128 KB block size and 8 jobs.
    Starting sequential read workload with 128 KB block size and 16 jobs.
    Starting sequential read workload with 128 KB block size and 32 jobs.
    Starting sequential read workload with 128 KB block size and 64 jobs.
    Starting sequential read workload with 1024 KB block size and 4 jobs.
    Starting sequential read workload with 1024 KB block size and 8 jobs.
    Starting sequential read workload with 1024 KB block size and 16 jobs.
    Starting sequential read workload with 1024 KB block size and 32 jobs.
    Starting sequential read workload with 1024 KB block size and 64 jobs.
    Storage benchmarking complete.
    Generating results.
    Storage test completed successfully.
    STORAGE_TEST_EXECUTE job for "SYSTEM" completed successfully.
delphix storage test>
```

4. The job will be submitted and visible in the Delphix Management application.
5. Retrieve the test results

```
delphix storage test> select STORAGE_TEST-1
delphix storage test 'STORAGE_TEST-1'> result
delphix storage test 'STORAGE_TEST-1' result *> commit
Test Results
------------
Test ID:          1
```

```
Test System UUID: 564dc710-7bb1-c064-12c2-2659032acf1b
Start Time:       03-Feb-2015 10:52:31 -0800
End Time:         03-Feb-2015 12:20:25 -0800

Test Grades:

Test Name                                  Latency            Load Scaling
-------------------------------     --------------------------  ---------------
                                    Average  95th %ile   Grade  Scaling   Grade
                                    -------  ---------   -----  -------   -----
Random 8K Reads w/ 16 jobs             2.16       4.77     A-       0.89    poor
Random 4K Reads w/ 16 jobs             1.62       3.73     A        0.54    fair
Sequential 1M Reads w/ 4 jobs         62.60     182.00     D        1.40     bad
Sequential 1K Writes w/ 4 jobs         1.30       2.61     C        0.07    good
Sequential 128K Writes w/ 4 jobs      10.19      26.00     D        1.35     bad


Grading Key:

Test Name          Grade:  A+    A     A-    B     B-    C     C-     D
-----------------          ----  ----  ----  ----  ----  ----  ----  -----
Small Random Reads          2.0   4.0   6.0   8.0  10.0  12.0  14.0 > 14.0
Large Seq Reads            12.0  14.0  16.0  18.0  20.0  22.0  24.0 > 24.0
Small Seq Writes            0.5   1.0   1.5   2.0   2.5   3.0   3.5 >  3.5
Large Seq Writes            2.0   4.0   6.0   8.0  10.0  12.0  14.0 > 14.0

IO Summary:

Test Name                         IOPS  Throughput (MBps)          Latency (msec)
-------------------------------  -------  -----------------
-----------------------------------
                                                                 Average      Min
Max   StdDev
                                                                 -------  -------
-------  -------
Rand 4K Reads w/ 8 Jobs          15703              61.34     0.50     0.05
754.74     1.72
Rand 4K Reads w/ 16 Jobs         15631              61.06     1.00     0.05
1347.10     5.12
Rand 4K Reads w/ 32 Jobs         15972              62.39     1.95     0.05
1231.40    17.56
Rand 4K Reads w/ 64 Jobs         17341              67.74     3.62     0.05
1750.10    30.09
Rand 8K Reads w/ 8 Jobs          15151             118.37     0.52     0.05
45.18     0.27
Rand 8K Reads w/ 16 Jobs         16457             128.58     0.95     0.05
501.90     3.57
Rand 8K Reads w/ 32 Jobs         16908             132.10     1.84     0.05
1336.10    16.93
Rand 8K Reads w/ 64 Jobs         16865             131.76     3.71     0.05
1505.50    30.03
```

```
Seq 1K Writes w/ 4 Jobs        22053              21.54     0.18     0.04
168.14     0.26
Seq 4K Writes w/ 4 Jobs        24937              97.41     0.16     0.04
152.17     0.27
Seq 8K Writes w/ 4 Jobs        22946             179.27     0.17     0.04
120.19     0.28
Seq 16K Writes w/ 4 Jobs       18003             281.31     0.22     0.05
81.24     0.26
Seq 32K Writes w/ 4 Jobs       12993             406.05     0.30     0.05
40.33     0.38
Seq 64K Writes w/ 4 Jobs        6429             401.83     0.62     0.06
116.19     2.26
Seq 128K Writes w/ 4 Jobs       3614             451.86     1.10     0.08
200.12     4.75
Seq 1M Writes w/ 4 Jobs          388             388.83    10.28     0.27
832.57    42.24
Seq 1K Writes w/ 16 Jobs       25965              25.36     0.60     0.04
814.84     6.86
Seq 4K Writes w/ 16 Jobs       25610             100.04     0.61     0.04
1022.50     7.76
Seq 8K Writes w/ 16 Jobs       25183             196.75     0.62     0.04
910.55     7.91
Seq 16K Writes w/ 16 Jobs      23433             366.14     0.66     0.04
948.57     8.05
Seq 32K Writes w/ 16 Jobs      19327             604.00     0.81     0.05
1180.50     8.10
Seq 64K Writes w/ 16 Jobs       9313             582.08     1.71     0.06
711.96     4.40
Seq 128K Writes w/ 16 Jobs      3369             421.14     4.75     0.08
69.12     4.32
Seq 1M Writes w/ 16 Jobs         481             481.06    33.22     0.27
269.88    32.05
Seq 64K Reads w/ 4 Jobs        16912            1057.20     0.23     0.05
40.36     0.15
Seq 64K Reads w/ 8 Jobs        18862            1178.10     0.42     0.05
78.57     0.41
Seq 64K Reads w/ 16 Jobs       20352            1272.50     0.77     0.06
900.81     7.41
Seq 64K Reads w/ 32 Jobs       20750            1296.10     1.50     0.06
1231.60    20.02
Seq 64K Reads w/ 64 Jobs       21146            1321.70     2.95     0.06
2440.30    34.37
Seq 128K Reads w/ 4 Jobs       11649            1456.30     0.34     0.06
53.66     0.25
Seq 128K Reads w/ 8 Jobs       15995            1999.50     0.49     0.06
32.21     0.42
Seq 128K Reads w/ 16 Jobs      17413            2176.80     0.90     0.07
1057.60     6.46
Seq 128K Reads w/ 32 Jobs      17874            2234.30     1.76     0.07
1355.40    19.87
Seq 128K Reads w/ 64 Jobs      17523            2190.50     3.58     0.07
1926.20    36.79
```

```
Seq 1M Reads w/ 4 Jobs          1404            1404.20     2.84      0.31
64.38     0.75
Seq 1M Reads w/ 8 Jobs          2360            2360.70     3.38      0.32
17.60     0.46
Seq 1M Reads w/ 16 Jobs         3876            3876.50     4.10      0.33
429.44     3.20
Seq 1M Reads w/ 32 Jobs         4732            4732.60     6.69      0.29
1305.70    34.64
Seq 1M Reads w/ 64 Jobs         4730            4730.10    13.33      0.32
1847.90    54.39



IO Histogram:

Test Name                      us50  us100  us250  us500    ms1    ms2    ms4
ms10   ms20   ms50  ms100  ms250  ms500     s1     s2     s5
---------------------------  -----  -----  -----  -----  -----  -----  -----
-----  -----  -----  -----  -----  -----  -----  -----  -----
Rand 4K Reads w/ 8 Jobs          0      0      0      2     46     41      7
3      1      0      0      0      0      0      0      0
Rand 4K Reads w/ 16 Jobs         0      0      0      0     39     47     10
3      1      0      0      0      0      0      0      0
Rand 4K Reads w/ 32 Jobs         0      0      0      0      6     64     22
6      2      0      0      0      0      0      0      0
Rand 4K Reads w/ 64 Jobs         0      0      0      0      0      4     75
16      3      1      0      0      0      0      0      0
Rand 8K Reads w/ 8 Jobs          0      0      0      0     41     49      7
2      1      0      0      0      0      0      0      0
Rand 8K Reads w/ 16 Jobs         0      0      0      0      8     66     20
4      2      1      0      0      0      0      0      0
Rand 8K Reads w/ 32 Jobs         0      0      0      0      0      5     72
18      3      2      0      0      0      0      0      0
Rand 8K Reads w/ 64 Jobs         0      0      0      0      0      0      3
85      8      4      1      0      0      0      0      0
Seq 1K Writes w/ 4 Jobs          0      0      0      4     53     36      4
2      1      0      0      0      0      0      0      0
Seq 4K Writes w/ 4 Jobs          0      0      0      2     44     44      6
3      1      0      0      0      0      0      0      0
Seq 8K Writes w/ 4 Jobs          0      0      0      1     41     47      7
3      1      0      0      0      0      0      0      0
Seq 16K Writes w/ 4 Jobs         0      0      0      0     27     57     10
3      1      0      0      0      0      0      0      0
Seq 32K Writes w/ 4 Jobs         0      0      0      0      4     55     30
8      2      1      0      0      0      0      0      0
Seq 64K Writes w/ 4 Jobs         0      0      0      0      0      1     56
33      7      3      0      0      0      0      0      0
Seq 128K Writes w/ 4 Jobs        0      0      0      0      0      0      0
76     16      5      2      0      0      0      0      0
Seq 1M Writes w/ 4 Jobs          0      0      0      0      0      0      0
0      0     24     57     14      4      0      0      0
```

```
Seq 1K Writes w/ 16 Jobs          0       0       0       1      55      34       6
2       1       0       0       0       0       0       0       0
Seq 4K Writes w/ 16 Jobs          0       0       0       1      46      42       7
3       1       0       0       0       0       0       0       0
Seq 8K Writes w/ 16 Jobs          0       0       0       0       3      43      38
12      2       1       0       0       0       0       0       0
Seq 16K Writes w/ 16 Jobs         0       0       0       0       0       4      67
23      4       2       0       0       0       0       0       0
Seq 32K Writes w/ 16 Jobs         0       0       0       0       0       0       1
74      16      8       2       0       0       0       0       0
Seq 64K Writes w/ 16 Jobs         0       0       0       0       0       0       0
2       81      12      3       1       0       0       0       0
Seq 128K Writes w/ 16 Jobs        0       0       0       0       0       0       0
1       2       85      7       4       0       0       0       0
Seq 1M Writes w/ 16 Jobs          0       0       0       0       0       0       0
0       0       1       4       45      38      9       3       0
Seq 64K Reads w/ 4 Jobs           0       0       0       0       0      29      59
9       2       1       0       0       0       0       0       0
Seq 64K Reads w/ 8 Jobs           0       0       0       0       0       0      15
74      8       3       1       0       0       0       0       0
Seq 64K Reads w/ 16 Jobs          0       0       0       0       0       0       0
14      53      27      5       1       0       0       0       0
Seq 64K Reads w/ 32 Jobs          0       0       0       0       0       0       0
1       27      59      8       4       0       0       0       0
Seq 64K Reads w/ 64 Jobs          0       0       0       0       0       0       0
0       1       29      42      25      2       0       0       0
Seq 128K Reads w/ 4 Jobs          0       0       0       0       0       0      10
75      9       5       1       0       0       0       0       0
Seq 128K Reads w/ 8 Jobs          0       0       0       0       0       0       0
64      29      5       2       0       0       0       0       0
Seq 128K Reads w/ 16 Jobs         0       0       0       0       0       0       0
1       45      45      6       2       0       0       0       0
Seq 128K Reads w/ 32 Jobs         0       0       0       0       0       0       0
0       1       65      24      8       1       0       0       0
Seq 128K Reads w/ 64 Jobs         0       0       0       0       0       0       0
0       1       8       54      29      5       1       0       0
Seq 1M Reads w/ 4 Jobs            0       0       0       0       0       0       0
0       0       66      23      8       2       0       0       0
Seq 1M Reads w/ 8 Jobs            0       0       0       0       0       0       0
0       0       1       33      52      11      3       0       0
Seq 1M Reads w/ 16 Jobs           0       0       0       0       0       0       0
0       0       1       5       70      15      8       1       0
Seq 1M Reads w/ 32 Jobs           0       0       0       0       0       0       0
0       0       1       4       19      58      11      6       1
Seq 1M Reads w/ 64 Jobs           0       0       0       0       0       0       0
0       0       1       2       10      40      32      11      2

delphix storage test 'STORAGE_TEST-1'>
```

Storage performance expectations and troubleshooting

Overview

When you initially set up a Delphix Engine, there is a one-time opportunity (and requirement) to run your storage performance tests on unconfigured storage. Because of the test method we use, we cannot run it again in the same way after the engine is set up. When run from a Windows target host, DiskSpd can help establish a performance baseline that combines the network and storage performance over **iSCSI**. This can be a significant help in determining whether your current or future performance is within normal expectations. Similarly, you can use fio on the *nix side from a target system to establish a performance baseline that combines the network and storage performance over **NFS**.

Troubleshooting and information gathering questions

These questions are presented roughly in order of priority. The answers to these questions should help narrow down possible causes of poor performance.

- All storage traffic for virtualization Target hosts goes over the network. Have you reviewed Network Performance Expectations and Troubleshooting?
- Were all of the storage best practices applied?
    - In particular, check the IO Operation limit and round-robin settings.
- Was a baseline ever created for this host using DiskSpd or fio? (How does our current performance compare?)
- Gather storage specs from VM/storage administrator:
    - Vendor, Model (EMC VMAX 40k, HP 3PAR StoreServ 7000)
    - IO latency SLO (e.g. 5ms 99.999%)
    - IOPS/GB SLO (e.g. 0.68 IOPS/GB for EMC Gold-1)
    - Cache type and size (e.g. FAST cache 768GB)
    - Tier, #Pools; if auto – tiering; relocation schedule (e.g. Gold/Silver/Auto/3 pools/etc)
    - Pool detail: (#) drives, RPM, Type (e.g. Pool1: (20) EFD, (30) 15k SAS, Pool 2: (40) 10k SATA)
    - Connection (XXGb Fibre Channel)
    - Dedicated or Shared pool (how many applications/servers)

Conclusion

If you need further help, please contact Delphix Support or Professional Services to assist in getting the best performance possible from your environment.

Storage performance test tool notes - Restricted

In addition to displaying the storage results on screen via CLI or retrieving results from a locally downloaded support bundle, Delphix employees have additional options to retrieve the information via ssh or via a support bundle uploaded to http://upload.delphix.com.

Download via SSH

Note the destination folder increments based on the job number. storage-test/1, storage-test/2, storage-test/3. etc.

```
delphix$ ls -l /var/delphix/server/storage-test/1/fio_summary*
rw-r---- 1 delphix staff 2310 Apr 29 17:51 /var/delphix/server/storage-test/1/
fio_summary_full.out
rw-r---- 1 delphix staff 1318 Apr 29 17:51 /var/delphix/server/storage-test/1/
fio_summary_grades.out
rw-r---- 1 delphix staff 793 Apr 29 17:51 /var/delphix/server/storage-test/1/
fio_summary.out
```

Versions older than 4.2 default to /opt/delphix/monitor/htdocs

Upload to portal; download from portal

1. Upload to portal:
    a. Login to the Delphix Management application.
    b. Select System -> Support Logs
    c. Choose to transfer the logs to Delphix
2. Download steps:
    a. Go to http://upload.delphix.com, login and select "browse uploaded files" to find the file

Extraction

1. Extract the file to locate "storage_tests.tar" in the root of the compressed file.

Architecture for performance - hypervisors and host

The Delphix Engine is a "virtual appliance": a virtual machine guest running within a hypervisor on a physical host. There are a few key best practices we need to keep in mind as we consider this architecture.

Architecture best practices hypervisor host ESX

Hypervisor

1. ESXi 6.x or 7.0 is recommended. ESXi 5.5 or earlier is no longer supported.
2. HyperThreading (HT) for Intel®-based servers (no HT on AMD CPUs).
    a. Disable HT in BIOS, on the ESXi Host, **and** disable HT Sharing on the Delphix VM for consistency. This is our best practice, disable at all levels.
    Any other combination may result in a non-deterministic performance.
    b. When HT cannot be turned off for both the Host and Delphix VM, it should be turned on at all levels, not run in a "mixed-mode".
        i. HT disablement at a guest level only can result in non-deterministic performance.
        ii. A dedicated ESXi host, cluster, or DRA is recommended where consistent VDB performance is paramount.
    c. VM migration to a new host (e.g. VMware HA or vMotion) can create mismatched HT settings.
3. ESXi overhead (resources required for hypervisor cannot be reserved, they must be left unallocated).
    a. Memory Overhead - 10% of available RAM must not be allocated to guest VMs.
        i. *Example: 256GB RAM, allocate 230GB to Delphix VM, leave 26GB for ESX*
    b. CPU Overhead - At least 2 cores (ideally 4) must not be allocated to guest VMs.
        i. *Example: If 16 physical cores are available, allocate 12 to the virtual machines, leaving 4 for the hypervisor*
        ii. Why 4? Certain hypervisor functions require precedence over any virtualized system. If a hypervisor needs more CPU than the amount currently available, it can de-schedule all other virtual processes to ensure adequate CPU resources for the hypervisor. Ensuring the hypervisor will not have to de-schedule any running virtual processes (worlds) by setting aside and not over-subscribing CPUs for virtual functions will leave them available for hypervisor use.
    c. Even if the Delphix VM is the only VM on a host, the hypervisor is still active and essential; and still needs resources.
4. BIOS Power Management should be set to High Performance where ESXi controls power management.
    a. It can be impacted by VMware KB 1018206 - poor VM application performance caused by power management settings.
    b. Ensure that all BIOS-managed C-States other than C0 are disabled if power management is hardware controlled.
    c. Ensure that all ACPI sleep states above S0 are disabled in the BIOS.
    d. Examples of popular server lines from Cisco, HP, Dell are below. *Specific models will vary, use the appropriate spec sheet.*
        i. UCS: disable the Processor Power States, disable Power Technology, set Energy Performance to "Performance"
        ii. HP Proliant: set HP Power Regulator to HP "Static High Performance" mode
        iii. Dell: set BIOS System Profile to "Performance Optimized" mode
5. VMware HA can be enabled; VMware DRS is generally disabled.
6. Blade/Rack Server Firmware and ESXi Drivers should be updated to the latest versions.
7. For Intel®-based servers with E5-2600 v2 processors.
8. Two typical server configurations:
    a. Blade Farm

   b. Rack Server
      Hyper-Converged configurations are possible for high performance.

Virtual machine guest

1. For VM machine settings, see Virtual Machine Requirements for VMware Platform.
2. VMWare Guest Specifications:
   a. Minimum: 8 vCPU x 64 GB
      Small: 8 vCPU x 128GB
      Medium: 16 vCPU x 256 GB
      Large: 24 vCPU x 512 GB
   b. Reserve 100% of RAM and CPU:
      i. If the ESX host is dedicated to Delphix, CPU and RAM reservations are advised but not necessary, however, swap space will be required on the hypervisor to compensate for the lack of reserved RAM.
   c. Hyperthreading - See the ESX host section at the top. Disable HT Sharing on VM, disable HT on ESX Host.
3. Assign single-core sockets for vCPUs in all cases. If there is a compelling reason to use multi-core vCPUs, reference the following article from VMware which describes matching virtual multi-core sockets to the hardware ESX is running on.
   VMware Article on CoresPerSocket
   Example: *ESXi Host has 2 socket x 18 core Intel Xeon, Delphix Engine wants 16 vCPU.*
   Configure Delphix VM with 2 Virtual Sockets, 8 Cores Per Socket to utilize hardware architecture.
4. Avoid placing other extremely active VMs on the same ESX host.
5. Monitoring - vSphere Threshold Alerts for CPU, Network, Memory, Capacity.
6. To set the number of vCPUs per virtual machine via the vSphere client, please see "*Virtual CPU Configuration"* in the Administration guide:
   ESXi 6.0
   a. Delphix VM CPU Utilization - Delphix KB article on what makes Delphix VMs similar to other resource-intensive applications
   b. Exchange on VMware Best Practices - VMworld 2013 session
   c. ESXTOP Reference, Blog
7. Ensure that the latest available VMware drivers and firmware versions are installed for HBAs, NICs, and any other hardware components configured on the Delphix virtual machine. This is a critical step that can have a massive impact on the performance and robustness of our solution.

Target host OS and database configuration options

This topic describes configuration options to maximize the performance of a target host in a Delphix Engine deployment. These network-tuning changes should improve performance for any data source

OS-specific tuning recommendations

Solaris

When exclusively using Oracle's Direct NFS Feature (dNFS), it is unnecessary to tune the native NFS client. However, tuning network parameters is still relevant and may improve performance.

Tuning the Kernel NFS client

On systems using Oracle Solaris Zones, the kernel NFS client can only be tuned from the global zone.

On Solaris, by default the maximum I/O size used for NFS read or write requests is 32K. For I/O requests larger than 32K, the I/O is broken down into smaller requests that are serialized. This may result in poor I/O performance. To increase the maximum I/O size:

1. As superuser, add to the `/etc/system` file:

   ```
   * For Delphix: change the maximum NFS block size to 1M
   set nfs:nfs3_bsize=0x100000
   ```

2. Run this command:

   ```
   # echo "nfs3_bsize/W 100000" | mdb -kw
   ```

   It is critical that the above command be executed exactly as shown, with quotations and space. Errors in the command may cause a system panic and reboot.

Tuning TCP buffer sizes

On systems using Oracle Solaris Zones, TCP parameters, including buffer sizes, can only be tuned from the global zone or in exclusive-IP non-global zones. Shared-IP non-global zones always inherit TCP parameters from the global zone.

Solaris 10

It is necessary to install a new Service Management Facility (SMF) service that will tune TCP parameters after every boot. These are samples of the files needed to create the service:

File	Installation location
dlpx-tcptune	`/lib/svc/method/dlpx-tcptune`
dlpx-tune.xml	`/var/svc/manifest/site/dlpx-tune.xml`

1. As superuser, download the files and install them in the path listed in the **Installation location** in the table.
2. Run the commands:

```
# chmod 755 /lib/svc/method/dlpx-tcptune
# /usr/sbin/svccfg validate /var/svc/manifest/site/dlpx-tune.xml
# /usr/sbin/svccfg import /var/svc/manifest/site/dlpx-tune.xml
# /usr/sbin/svcadm enable site/tcptune
```

Verify that the SMF service ran after being enabled by running the command:

```
# cat `svcprop -p restarter/logfile tcptune`
```

You should see output similar to this:

```
[ May 14 20:02:02 Executing start method ("/lib/svc/method/dlpx-tcptune
start"). ]
Tuning TCP Network Parameters
tcp_max_buf adjusted from 1048576 to 16777216
tcp_cwnd_max adjusted from 1048576 to 4194304
tcp_xmit_hiwat adjusted from 49152 to 4194304
tcp_recv_hiwat adjusted from 128000 to 16777216
[ May 14 20:02:02 Method "start" exited with status 0. ]
```

Solaris 11

As superuser

Run the following commands:

```
# ipadm set-prop -p max_buf=16777216 tcp
# ipadm set-prop -p _cwnd_max=4194304 tcp
# ipadm set-prop -p send_buf=4194304 tcp
# ipadm set-prop -p recv_buf=16777216 tcp
```

Linux/Redhat/CentOs

Tuning the Kernel NFS client

In Linux, the number of simultaneous NFS requests is limited by the Remote Procedure Call (RPC) subsystem. The maximum number of simultaneous requests defaults to 16. Maximize the number of simultaneous requests by changing the kernel tunable `sunrpc.tcp_slot_table_entries` value to 128.

To ensure that the interface does not drop packets because the driver is configured with one receive queue, use the following commands to view the adaptor policy/increase Rx queue length.

```
<LinuxHost> $ ifconfig -a
```

```
eth0      Link encap:Ethernet  HWaddr 00:22:BB:CC:DD:22
          inet addr:www.xxx.yyy.zzz  Bcast:www.xxx.yyy.zzz  Mask:255.255.255.0
          inet6 addr: feee::222:bbff:fffc:ddd/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:9000  Metric:1
          RX packets:760729910 errors:0 dropped:700 overruns:0 frame:0
          TX packets:309094054 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1023150307866 (952.8 GiB)  TX bytes:190673864056 (177.5 GiB)
```

RHEL4 through RHEL5.6

1. As superuser, run the following command to change the instantaneous value of simultaneous RPC commands:

   ```
   # sysctl -w sunrpc.tcp_slot_table_entries=128
   ```

2. Edit the file `/etc/modprobe.d/modprobe.conf.dist` and change the line:

   ```
   install sunrpc /sbin/modprobe --first-time --ignore-install sunrpc &&
   { /bin/mount -t rpc_pipefs sunrpc / var /lib/nfs/rpc_pipefs > /dev/ null
    2>&1 || :;
   ```

   to

   ```
   install sunrpc /sbin/modprobe --first-time --ignore-install sunrpc &&
   { /bin/mount -t rpc_pipefs sunrpc / var /lib/nfs/rpc_pipefs > /dev/ null
    2>&1 ; /sbin/sysctl -w sunrpc.tcp_slot_table_entries=128; }
   ```

   Improper changes to the modprobe.conf.dist file may disrupt the use of NFS on the system. Check with your system administrator or operating system vendor for assistance. Save a copy of the modprobe.conf.dist in a directory other than /etc/modprobe.d before starting.

RHEL 5.7 through RHEL 6.2

1. As superuser, run the following command to change the instantaneous value of simultaneous RPC commands:

   ```
   # sysctl -w sunrpc.tcp_slot_table_entries=128
   ```

2. If it doesn't already exist, create the file `/etc/modprobe.d/rpcinfo` with the following contents:

   ```
   options sunrpc tcp_slot_table_entries=128
   ```

RHEL 6.3 onwards

Beginning with RHEL 6.3, the number of RPC slots is dynamically managed by the system and does not need to be tuned. Although the `sunrpc.tcp_slot_table_entries` tuneable still exists, it has a default value of 2, instead of 16 as in prior releases. The maximum number of simultaneous requests is determined by the new tuneable, `sunrpc.tcp_max_slot_table_entries` which has a default value of 65535.

Tuning TCP buffer sizes

Packages should install their configuration files in /usr/lib/(distribution packages) or /usr/local/lib/(local installs). Files in /etc/ are reserved for the local administrator, who may use this logic to override the configuration files installed by vendor packages. It is recommended to prefix all filenames with a two-digit number and a dash, to simplify the ordering of the files. The following is an example approach and should be tested beforehand.

```
echo "Target Kernel Parameter Tunings.  This is optional but highly recommended."
echo
echo "Tuning TCP Buffer Sizes - Parameters should be as below"
echo
echo "This script takes the recommended vendor approach of creating a file in /usr/
lib/sysctl.d and running \"sysctl -p\""
echo "This script will comment out identical settings in /etc/sysctl.conf, which
would otherwise override the Delphix settings"
echo "Admins can at their discretion set larger values, either in sysctl.conf or /
usr/lib/sysctl.d/60-sysctl.conf"
echo "----"
echo "net.ipv4.tcp_timestamps = 1"
echo "net.ipv4.tcp_sack = 1"
echo "net.ipv4.tcp_window_scaling = 1"
echo "net.ipv4.tcp_rmem = 4096 16777216 16777216"
echo "net.ipv4.tcp_wmem = 4096 4194304 16777216"
cat /dev/null > /usr/lib/sysctl.d/60-sysctl.conf
#Set NOW
NOW=$(date +"%m%d%Y%H%m%s")
echo "net.ipv4.tcp_timestamps = 1" >> /usr/lib/sysctl.d/60-sysctl.conf
echo "net.ipv4.tcp_sack = 1" >> /usr/lib/sysctl.d/60-sysctl.conf
echo "net.ipv4.tcp_window_scaling = 1" >> /usr/lib/sysctl.d/60-sysctl.conf
echo "net.ipv4.tcp_rmem = 4096 16777216 16777216" >> /usr/lib/sysctl.d/60-sysctl.conf
echo "net.ipv4.tcp_wmem = 4096 4194304 16777216" >> /usr/lib/sysctl.d/60-sysctl.conf
echo
echo "Running /sbin/sysctl -p /usr/lib/sysctl.d/60-sysctl.conf"
/sbin/sysctl -p /usr/lib/sysctl.d/60-sysctl.conf
if [ $? -ne 0 ]; then
    echo "Command \"sysctl -p /usr/lib/sysctl.d/60-sysctl.conf\" failed; aborting..."
    exit 1
fi
# Make a backup and Comment out similar lines in /etc/sysctl.conf if they exist
# sed will search for lines which are NOT comments ( ^[^#]* means starts with
anything other than a "#" ) but contain the parameters either with "." or "/"
 notation, and add a Comment.
cp /etc/sysctl.conf /tmp/sysctl.conf.$NOW
sed -i "/^[^#]*net[\./]ipv4[\./]tcp_timestamps/s/^/#Commented out by Delphix /" /etc/
sysctl.conf
sed -i "/^[^#]*net[\./]ipv4[\./]tcp_sack/s/^/#Commented out by Delphix /" /etc/
sysctl.conf
sed -i "/^[^#]*net[\./]ipv4[\./]tcp_window_scaling/s/^/#Commented out by Delphix /" /
etc/sysctl.conf
sed -i "/^[^#]*net[\./]ipv4[\./]tcp_rmem/s/^/#Commented out by Delphix /" /etc/
sysctl.conf
```

```
sed -i "/^[^#]*net[\./]ipv4[\./]tcp_wmem/s/^/#Commented out by Delphix /" /etc/
sysctl.conf
echo
echo "Running /sbin/sysctl -p /etc/sysctl.conf"
/sbin/sysctl -p /etc/sysctl.conf
if [ $? -ne 0 ]; then
    echo "Command \"sysctl -p /etc/sysctl.conf\" failed; aborting..."
    echo "We put a copy of the original at /tmp/sysctl.conf.$NOW"
    exit 1
fi
```

NFSv4 only - enabling recover lost locks

RHEL 6.6 onwards

By default, the Redhat NFSv4 client does not attempt to reclaim locks that were lost due to a lease expiration event. This can cause an application to encounter unexpected EIO errors on system calls such as write. The Delphix use case requires that the NFSv4 client attempt to reclaim lost locks that were due to lease expiration. An NFS client module parameter, '`recover_lost_locks`', is used to change the default behavior. Use the following command to check if the "recover_lost_locks" option is set to 1:

```
grep recover_lost_locks /etc/modprobe.d/*.conf
```

If the option is currently set to 0, change it to 1. If it is missing, proceed to the next paragraph where instruction is provided on how to add the option.

As superuser, run the following two commands to enable the NFS client to recover the lost locks feature:

```
# cat > /etc/modprobe.d/nfs4-locks.conf <<EOF
options nfs recover_lost_locks=1
EOF

# [ -d "/sys/module/nfs" ] && echo Y > /sys/module/nfs/parameters/recover_lost_locks
```

IBM AIX®

AIX NFSv4 configuration requirements (7.1 and 7.2)

1. An NFS Domain must be configured.
2. The nfsrgyd service must be running.
3. The NFS server IP address from the Delphix Engine must be mappable to an FQDN.

Configure the nfsv4 domain on the AIX target host

bash-3.2# chnfsdom test.com

Start the nfsrgyd service and confirm it is active

```
bash-3.2# startsrc -s nfsrgyd

bash-3.2# lssrc -s nfsrgyd
```

```
Subsystem         Group            PID          Status
 nfsrgyd          nfs              7536760      active
```

Confirm that IP address can be resolved

```
bash-3.2$ host 172.16.105.81
81.105.16.172.in-addr.arpa is  dcol1.delphix.com
```

Reference: IBM AIX: HOW TO SETUP NFSV4 MOUNT IN CLIENT AND SERVER

Tuning the Kernel NFS Client

On AIX, by default the maximum I/O size used for NFS read or write requests is 64K. When Oracle does I/O larger than 64K, the I/O is broken down into smaller requests that are serialized. This may result in poor I/O performance. IBM can provide an Authorized Program Analysis Report (APAR) that allows the I/O size to be configured to a larger value.

1. Determine the appropriate APAR for the version of AIX you are using:

AIX Version	APAR Name
6.1	IV24594
7.1	IV24688

2. Check if the required APAR is already installed by running this command:If the APAR is installed, you will see a message similar to this:

```
# /usr/sbin/instfix -ik IV24594
```

3. If the APAR is not yet installed, you will see a message similar to this:

```
All filesets for IV24594 were found.
```

```
There was no data for IV24594 in the fix database.
```

4. Download and install the APAR, as necessary. To find the APARs, use the main search function at http://www.ibm.com/us/en/, specifying the name of the APAR you are looking for from step 1. A system reboot is necessary after installing the APAR.
5. Configure the maximum read and write sizes using the commands below:

```
# nfso -p -o nfs_max_read_size=524288
# nfso -p -o nfs_max_write_size=524288
```

6. Confirm the correct settings using the command:
You should see an output similar to this:

```
# nfso -L nfs_max_read_size -L nfs_max_write_size
NAME CUR DEF BOOT MIN MAX UNIT TYPE
   DEPENDENCIES
------------------------------------------------------------------------------
```

```
------
nfs_max_read_size 512K 64K 512K 512 512K Bytes D
------------------------------------------------------------------------
------
nfs_max_write_size 512K 64K 512K 512 512K Bytes D
------------------------------------------------------------------------
------
```

Tuning delayed TCP acknowledgements

By default, AIX implements a 200ms delay for TCP acknowledgments. However, it has been found that disabling this behavior can provide significant performance improvements.

To disable delayed ACKs on AIX the following command can be used:

```
# /usr/sbin/no -o tcp_nodelayack=1
```

To make the change permanent use:

```
# /usr/sbin/no -p -o tcp_nodelayack=1
```

HP-UX

Tuning the Kernel NFS client

On HP-UX, by default the maximum I/O size used for NFS read or write requests is 32K. For I/O requests larger than 32K, the I/O is broken down into smaller requests that are serialized. This may result in poor I/O performance.

1. As superuser, run the following command:

   ```
   # /usr/sbin/kctune nfs3_bsize=1048576
   ```

2. Confirm the changes have occurred and are persistent by running the following command and checking the output:

   ```
   # grep nfs3 /stand/system
   tunable nfs3_bsize 1048576
   ```

Tuning TCP buffer sizes

1. As superuser, edit the `/etc/rc.config.d/nddconf` file, adding or replacing the following entries:

```
TRANSPORT_NAME[0]=tcp
NDD_NAME[0]=tcp_recv_hiwater_def
NDD_VALUE[0]=16777216
#
TRANSPORT_NAME[1]=tcp
NDD_NAME[1]=tcp_xmit_hiwater_def
NDD_VALUE[1]=4194304
```

In this example, the array indices are shown as `0` and `1.` In the actual configuration file, each index used must be strictly increasing, with no missing entries. See the comments at the beginning of `/etc/rc.config.d/nddconf` for more information.

1. Run the command:

   ```
   /usr/bin/ndd -c
   ```

2. Confirm the settings:

   ```
   # ndd -get /dev/tcp tcp_recv_hiwater_def
   16777216
   # ndd -get /dev/tcp tcp_xmit_hiwater_def
   4194304
   ```

OS-specific tuning recommendations for windows

iSCSI tuning

These are our recommendations for the Windows iSCSI initiator configuration. Please note that the parameters below will affect **all** applications running on the Windows target host, so you should make sure that the following recommendations do not contradict best practices for other applications running on the host.

For details about the Windows iSCSI configuration, see Requirements for Windows iSCSI Configuration

For targets running Windows Server, the iSCSI initiator driver timers can be found at: `HKLM\SYSTEM\CurrentControlSet\Control\Class\{4D36E97B-E325-11CE-BFC1-08002BE10318}\<>` . Please see How to Modify the Windows Registry on the Microsoft Support site for details about configuring registry settings.

Registry Value	Type	Default	Recommended	Comments
`MaxTransferLength`	REG_DWORD	262144	131072	This controls the maximum data size of an I/O request. A value of 128K is optimal for the Delphix Engine as it reduces the segmentation of the packets as they go through the stack.
`MaxBurstLength`	REG_DWORD	262144	131072	This is the negotiated maximum burst length. 128K is the optimal size for the Delphix Engine.
`MaxPendingRequests`	REG_DWORD	255	512	This setting controls the maximum number of outstanding requests allowed by the initiator. At most this many requests will be sent to the target before receiving a response for any of the requests.
`MaxRecvDataSegmentLength`	REG_DWORD	65536	131072	This is the negotiated MaxRecvDataSegmentLength.

Receive side scaling

Follow the instructions here to enable RSS: Enable Receive Side Scaling (RSS) on Staging/Target Network Interfaces

hosted on the target, such as Oracle, SQL Server, Sybase, or vFiles. They should be applied to all Delphix targets.

Usage data management

The Delphix User-click Analytics feature is a lightweight method to capture how users interact with Delphix product user interfaces. The goal of capturing this data is to get a better understanding of product usage, engagement, and user behavior, and to use this data to improve Delphix products and customer experience. This feature is enabled by default for customers deploying on or upgrading to this version. User-click Analytics may also be disabled via the UI.

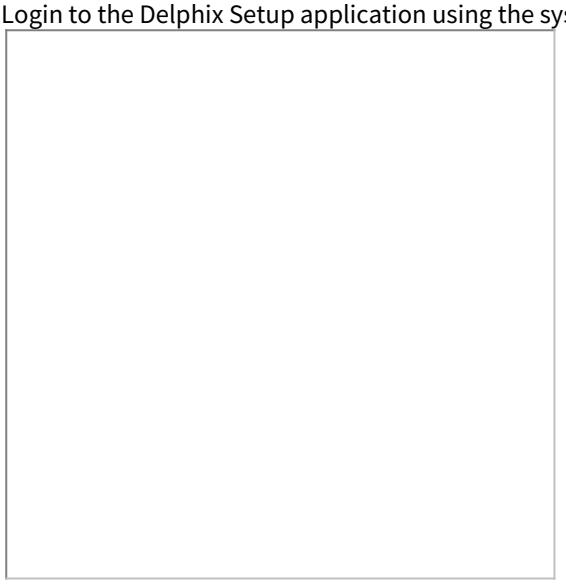
Disabling user-click analytics

ⓘ This procedure will disable user-click analytics on both the Delphix Engine and Delphix Self-Service.

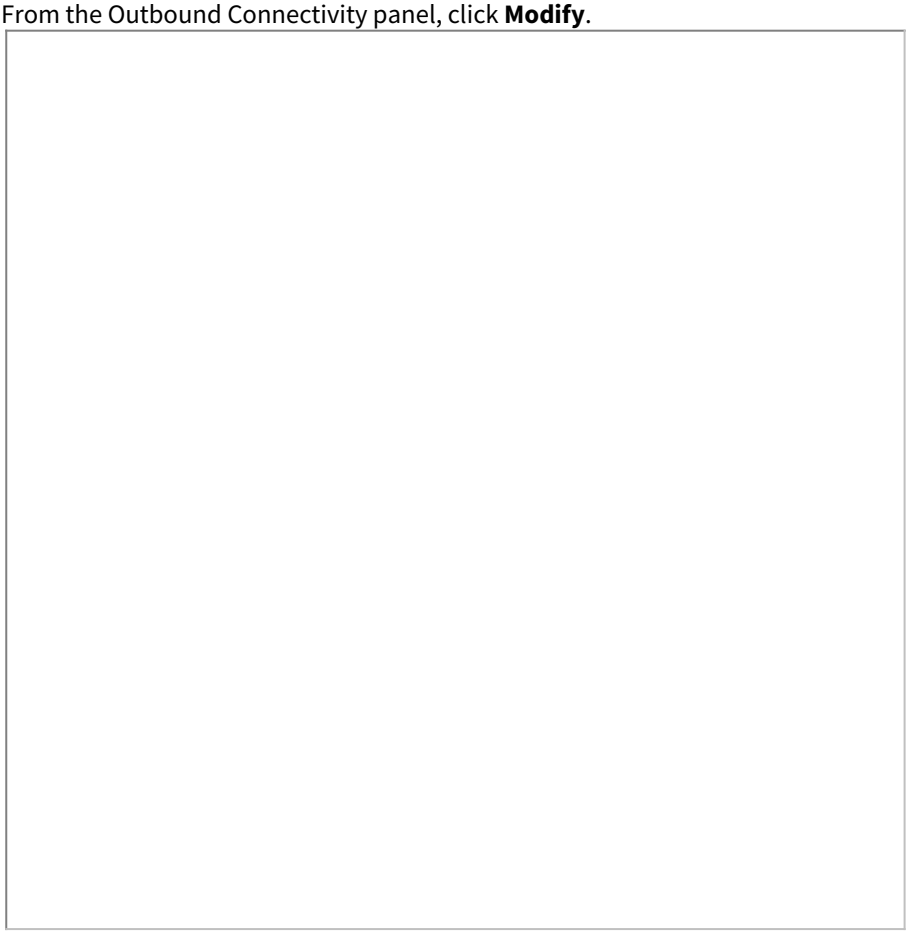
Procedure to disable user-click analytics via the GUI

Complete the following steps to disable user-click analytics via the GUI.

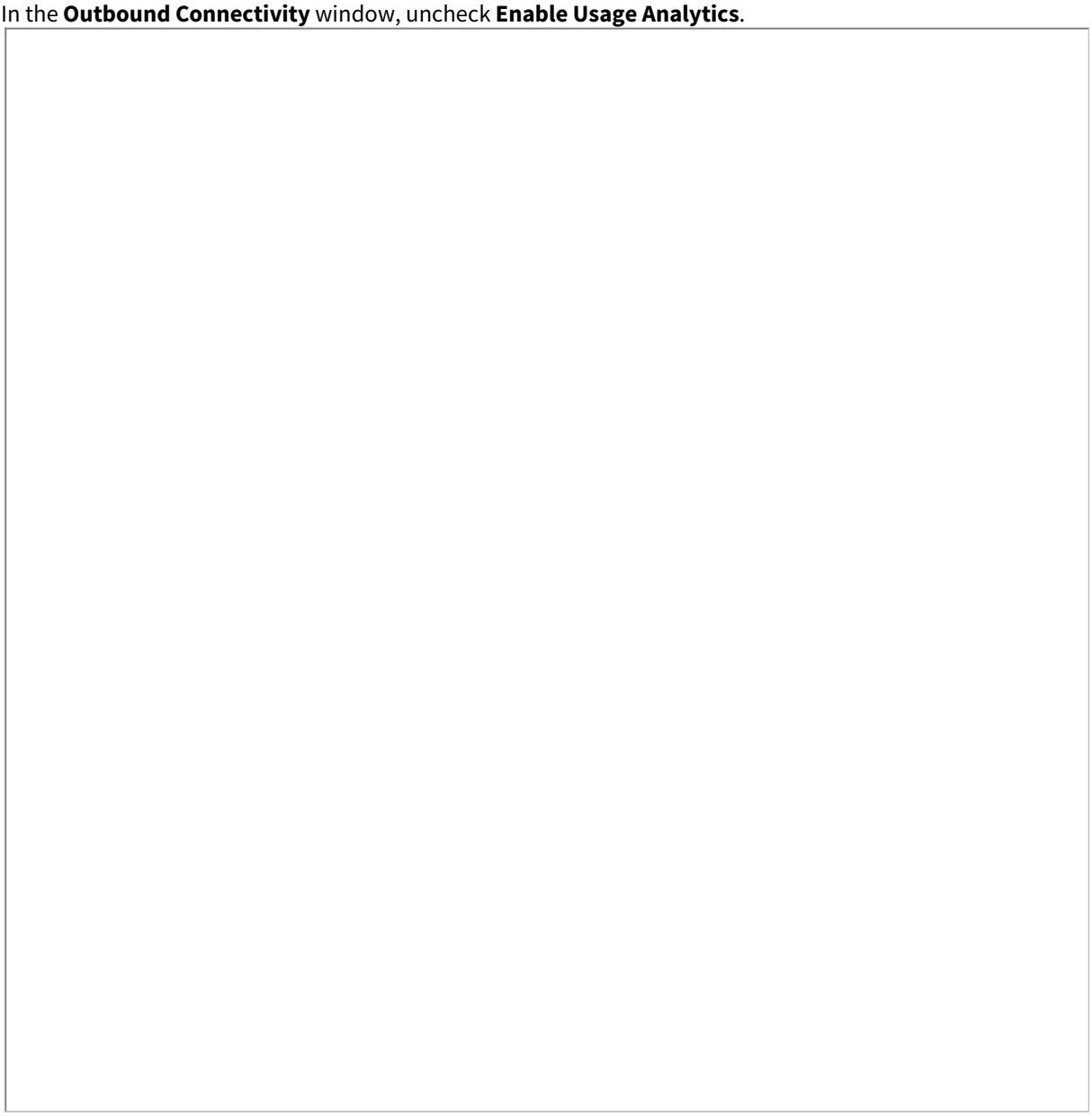
1. Login to the Delphix Setup application using the sysadmin **username** and **password**.



2. From the Outbound Connectivity panel, click **Modify**.



3. In the **Outbound Connectivity** window, uncheck **Enable Usage Analytics**.



4. Click **Save**.

Starting, stopping, and restarting your engine

Several options are available to start, stop, restart, or reset the Delphix engine for maintenance or debugging purposes.

Factory reset

This option returns the Delphix Engine to the "factory default" state. This removes all the data and configurations except for the network configuration and NTP settings. The network configuration and NTP settings are retained in order to make it is easy for a user to start the Delphix Engine otherwise you will need to perform the initial steps to configure the network via the VMware Hypervisor console.

It is recommended to shut down and remove all VDBs before resetting the Delphix Engine. Failure to do so could lead to stale data mounts in target environments. For the same reason, disable all dSources that use validated sync.

Use Factory Reset only when a complete reset and reconfiguration of the Delphix Engine is necessary, as all Delphix Engine objects will be de-allocated.

Complete the following steps to reset the Delphix Engine via GUI:

1. Connect to the Delphix Setup application and log in as a system administrator.
2. From the **Dashboard** panel, click the Actions menu (**...**) on the top right and select **Factory Reset**.

3. Click **OK**.

Restart

If the Delphix Java process is in a bad state, you may need to restart it to get it operational again.

You can restart the management process safely without shutting down VDBs or disabling dSources because restarting this process has no impact on running VDBs or dSources.

Complete the following steps to restart the Delphix Management Process via GUI:

1. Login to the Delphix Setup application as a system administrator.
2. From the **Dashboard** panel, click the Actions menu (**...**) on the top right and select **Restart**.
3. Click **OK**.

Complete the following steps to restart the Delphix Management Process via CLI:

1. Log in to the CLI as a system administrator.
2. Go to system > restart
3. delphix system restart *> commit

Reboot

If the management stack for your Delphix Engine hangs, you will need to perform a reboot.

Before performing a reboot, all your VDBs must be shut down and dSources disabled to maintain data integrity.

Complete the following steps to reboot the Delphix Engine via GUI:

1. Login to the Delphix Setup application as a system administrator.
2. From the **Dashboard** panel, click the Actions menu (**...**) on the top right and select **Reboot**.

3. Click **OK**.

Complete the following steps to reboot the Delphix Engine via CLI:

1. Log in to the CLI as a system administrator.
2. Go to system > reboot
3. delphix system reboot *> commit

> **Reboot vs restart**
> The only difference between the Restart and Reboot is that the Restart functions without shutting down the VDBs and disabling the dSources.

Power off

Occasionally, it is necessary to shut down the Delphix Engine for maintenance purposes. Before performing a shutdown, all your VDBs must be shut down and dSources disabled to maintain data integrity.

Once the Delphix Engine is powered off, you can power it back on through your hypervisor console.

Use the following steps to shut down the Delphix Engine via GUI:

1. Login to the Delphix Setup application as a system administrator.
2. From the **Dashboard** panel, click the Actions menu (**...**) on the top right and select **Power Off**.

3. Click **OK**.

Use the following steps to shut down the Delphix Engine via CLI:

1. ssh sysadmin@yourengine
2. delphix > system
3. delphix system > shutdown
4. delphix system shutdown *> commit

Introduction to privilege elevation profiles

This topic introduces the concept of Privilege Elevation Profiles, how they are managed, and how they are supported. Privilege Elevation Profiles exist to provide the Delphix Engine with a mechanism for running privileged commands in a secure way to achieve the following:

- Mount and Unmount NFS filesystems
- Create and Remove directories in paths not owned by the Delphix OS user
- Examine the running process list
- Run commands as root

Privilege Elevation Profiles is an advanced CLI topic and are not documented as part of the general Delphix Engine User Guide. Changes to the default sudo-based profile scripts, or the creation of new profiles that do not work as expected, can cause serious problems and render the Delphix Engine unusable. This article is aimed at advanced end-users and Delphix Professional Services consultants.

Support for privilege elevation profiles

> ⓘ Writing and troubleshooting scripts, such as those required for Privilege Elevation Profiles, is out of scope and not covered by Delphix Support.

Privilege Elevation Profiles need to be tailor-made to work with non-standard environments that may use third-party or proprietary privilege elevation mechanisms other than sudo. Customers are strongly encouraged to work with Delphix Professional Services to formulate reliable profile scripts. There is nothing that prevents customers from creating their own profile scripts. However, customers bear full responsibility for supporting and troubleshooting their own profile scripts. Support for profile scripts created by our Professional Services consultants is still supported by Professional Services.

How do privilege elevation profiles work?

Privilege Elevation Profiles exist within a two-tier cascading hierarchy. This means there is one default profile for the entire Delphix Engine that should contain scripts for all the operations that require privilege elevation. Additional profiles may contain a subset of the scripts. When a non-default profile is used, the Delphix Engine uses that profile's scripts where they exist and reverts to the scripts in the default profile if no script for the operation exists. By default, the Delphix Engine ships with simple scripts that pass commands to the standard UNIX **sudo** command.

All Environments added to the Delphix Engine get added with the default Privilege Elevation Profile. The profile can be assigned on a per-host basis. Below shows how a host using a non-standard profile will use scripts in the cascading model.

default profile (sudo)	custom profile (myProfile)	host profile	script used
dlpx_mount	my_mount	myProfile	my_mount
dlpx_umount	my_umount		my_umount
dlpx_rmdir			dlpx_rmdir

default profile (sudo)	custom profile (myProfile)	host profile	script used
dlpx_mkdir			dlpx_mkdir
dlpx_ps			dlpx_ps
dlpx_pfexec			dlpx_pfexec

Upgrade

Version compatibility and support pre-checks

Please refer to Upgrade matrix before upgrading to a newer version.

Customers running version 5.3.9 and earlier that are requesting an upgrade to 6.0.0.0 and above, please contact Delphix Support to help coordinate this upgrade.

Upgrading from 6.0.x to 6.0.x includes pre-checks packaged in the upgrade image, contacting Delphix Support for this upgrade is **not required** (e.g. 6.0.0.0 → 6.0.9.0).

Upgrade

These topics describe processes for upgrading the Delphix Engine.

- Upgrading the Delphix Engine: Overview
- Upgrade prerequisites
- Downloading the upgrade image
- Uploading the upgrade image
- Applying the upgrade
- Post upgrade

Upgrading the Delphix Engine: Overview

Upgrading the Delphix Engine is a multi-step process that requires some preparation. The engine upgrade process will affect the availability of the Delphix Engine administrative interface and virtual datasets during the operation based on the type of upgrade chosen. For large configurations and Full/Apply Now upgrades, it can take one to two hours; Delay the Reboot upgrades are typically complete within 15 to 30 minutes. Please refer to Upgrade Matrix and the **version compatibility and support pre-checks** callout below before proceeding with an upgrade.

The following sections explain the steps involved in the upgrade process with links to detailed instructions for proceeding through each of them.

Types of upgrade

There are two types of upgrades, which are characterized by the impact they have on VDBs during the operation:

ⓘ There is no need to reboot unless instructed to receive a specific fix from Delphix.

Upgrade type	Description
Delay the reboot	The user interface, API, and CLI (Command Line Interface) will only be available to the user performing the upgrade. dSources will stop refreshing from production. Policies execution will be delayed until after the upgrade has been completed. Jobs will be canceled (and resumed after upgrade if supported). Access to VDB data will not be affected by this upgrade and can be used normally.
Apply now	In addition to performing an application upgrade, DelphixOS, the operating system that runs Delphix, will be upgraded and the machine will reboot to the new OS as part of the upgrade process. The Delphix Engine will automatically disable all VDBs and dSources during the upgrade process in order to safely reboot to the new version, and thus you should schedule downtime for your VDB applications.

Outline of the upgrade process

Version compatibility and support pre-checks

Please refer to Upgrade Matrix before upgrading to a newer version.

Customers running version 5.3.9 and earlier that are requesting an upgrade to 6.0.0.0 and above, please contact Delphix Support to help coordinate this upgrade.

Upgrading from 6.0.x to 6.0.x includes pre-checks packaged in the upgrade image, contacting Delphix Support for this upgrade is **not required** (e.g. 6.0.0.0 → 6.0.9.0).

The following is an outline of the steps for upgrading the Delphix Engine:

1. Upload the upgrade image to the Delphix Engine.
2. Verify and resolve the system requirements and known defects before starting the upgrade.
3. Schedule the appropriate downtime.
4. Start the upgrade and choose the upgrade type.
5. Address any runtime failures that happen as part of the upgrade.
6. Verify that the upgrade was completed successfully.

Upgrade prerequisites

> **Version compatibility and support pre-checks**
> Please refer to Upgrade matrix before upgrading to a newer version.
> Customers running version 5.3.9 and earlier that are requesting an upgrade to 6.0.0.0 and above, please contact Delphix Support to help coordinate this upgrade.
> Upgrading from 6.0.x to 6.0.x includes pre-checks packaged in the upgrade image, contacting Delphix Support for this upgrade is **not required** (e.g. 6.0.0.0 → 6.0.9.0).

Scheduling downtime

To determine if an upgrade will require a reboot and VDB downtime refer to Upgrade matrix.

If the OS will not be updated as part of the upgrade, then the upgrade process will have no impact on the availability of VDBs, and you do not need to schedule any downtime for your VDB applications.

If the OS will be updated as part of the upgrade, then you should schedule appropriate downtime for your VDB applications. The Delphix Engine will automatically disable VDBs and dSources during the upgrade. The length of the downtime will be proportional to the number of VDBs.

Long-running jobs including replication and SnapSync will fail during any upgrade.

The upgrade file for the version to which you want to upgrade should be downloaded from the Delphix download site. From 6.0.2.0 onwards, the upgrade images are packaged with the latest version of upgrade pre-checks. Please use the latest upgrade image from the download site since the checks will be updated on a need-by basis.

Delphix Upgrade images are approximately 5GB in size; it is recommended to have both a fast internet connection to the Delphix download site as well as to the Delphix Engine.

The upgrade image should be downloaded or moved to a location accessible to the computer used for navigating the Delphix Management application.

> **Upgrading replication source**
> Delphix Engines can only perform replication to engines on the same or higher version. Upgrading a replication source without upgrading its replication targets could cause replication between those peers to fail.

Verifying the integrity of the downloaded upgrade image

SHA256 checksums can be used to verify the integrity of files downloaded via download.delphix.com.
The `sha256sum` checksum listed next to each download will help ensure that a file has not changed as a result of a faulty file transfer, a disk error, or non-malicious meddling.

To calculate the checksum of downloaded files and compare them to the checksum listed on the download portal, the following utilities can be used:

- On Windows, `certutil -HashFile <upgrade.tar.gz> SHA256`
- On macOS or Unix servers with Perl installed, `shasum -a 256 <upgrade.tar.gz>`
- On Unix servers with `coreutils` installed, `sha256sum <upgrade.tar.gz>.`

Verifying connectivity to datasets and environments

For Upgrades from versions > 5.3.6.0 and <6.0.0.0, your Delphix Engine will automatically quiesce all VDBs and dSources during the upgrade process in order to safely reboot to the new version.

For Upgrades from versions >= 6.0.0.0, based on the Upgrade matrix you can choose to postpone the VDB downtime and update just the running stack and related packages by choosing "Delay the Restart" option. You can update the OS at a later date by completing the upgrade.

At the end of the upgrade process, the Delphix Engine will also update the Delphix platform toolkit on each connected environment.

To perform these tasks, the Delphix Engine must be able to connect to the environments in which datasets exist and must have credentials to connect to datasets or applications. Environments involved must be properly configured to enable script execution.

When an environment, dataset, or application is unreachable or misconfigured, the upgrade may encounter access errors, and may not be able to re-start it after the upgrade has been applied.

If you know that a dataset is unavailable for some reason (e.g. host not reachable on the network, or the hosting server is down), it is recommended that you **disable** it (if necessary utilize **Force Disable**).

The **Impact** of **Upgrade** section in the **Details** tab contains the list of datasets that have been identified as unreachable, misconfigured or not behaving correctly and are at risk of not functioning after the upgrade.

Review the warnings in this section (if any) and take appropriate actions before applying the upgrade. 

Downloading the upgrade image

The upgrade file for the version to which you want to upgrade should be downloaded from the Delphix download site. From 6.0.2.0 onwards, the upgrade images are packaged with the latest version of upgrade pre-checks. Please use the latest upgrade image from the download site since the checks will be updated on a need-by basis.

Delphix Upgrade images are approximately 5GB in size; it is recommended to have both a fast internet connection to the Delphix download site as well as to the Delphix Engine.

The upgrade image should be downloaded or moved to a location accessible to the computer used for navigating the Delphix Management application.

> ⚠ **Version compatibility and support pre-checks**
> Please refer to Upgrade matrix before upgrading to a newer version.
> Customers running version 5.3.9 and earlier that are requesting an upgrade to 6.0.0.0 and above, please contact Delphix Support to help coordinate this upgrade.
> Upgrading from 6.0.x to 6.0.x includes pre-checks packaged in the upgrade image, contacting Delphix Support for this upgrade is **not required** (e.g. 6.0.0.0 → 6.0.9.0).

For upgrades from versions > 5.3.6.0 and < 6.0.0.0, you will download a migration image that is specific to the virtualization platform that your Delphix Engine is running on. The platforms for which images are provided include:

- Amazon AWS
- Microsoft Azure
- VMware ESX

For example, an ESX migration image will have a filename like `Delphix_6.0.0.0_2019-12-06-09-07_Standard_ESX_Migration.tar.gz.`

For upgrades from versions >=6.0.0.0, you will download an upgrade image that is generic to all the virtualization platforms that your Delphix Engine is running on. The Upgrade image will have a filename like `Delphix_6.0.1.0_2020-03-17-00-39_Standard_Upgrade.tar`.

SHA256 checksums, which can be used to verify the integrity of any downloaded files, are available via download.delphix.com. Each download link should list a `sha256sum` checksum underneath. Additionally, newer releases may also contain a downloadable SHA256SUMS file, containing the checksums of all files in that directory.

To calculate the checksum of downloaded files, and confirm that they match those in the SHA256SUMS file, the following utilities can be used:

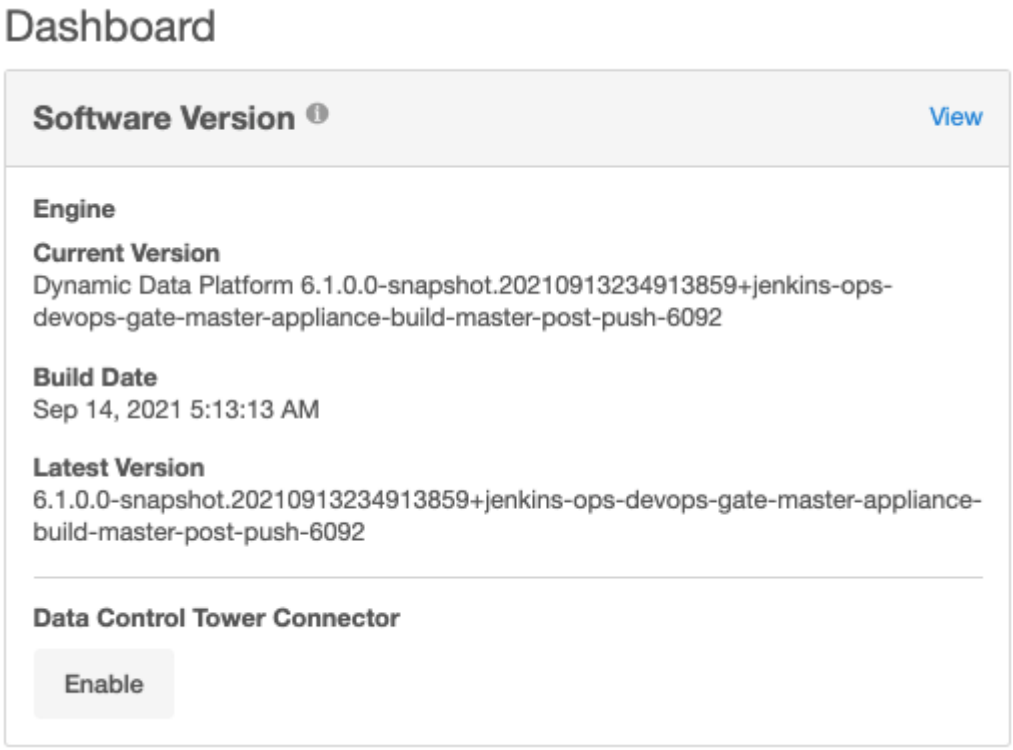
- On Windows, `certutil -HashFile <upgrade.tar.gz> SHA256`
- On macOS or Unix servers with Perl installed, `shasum -a 256 <upgrade.tar.gz>`
- On Unix servers with `coreutils` installed, `sha256sum <upgrade.tar.gz>.`

Uploading the upgrade image

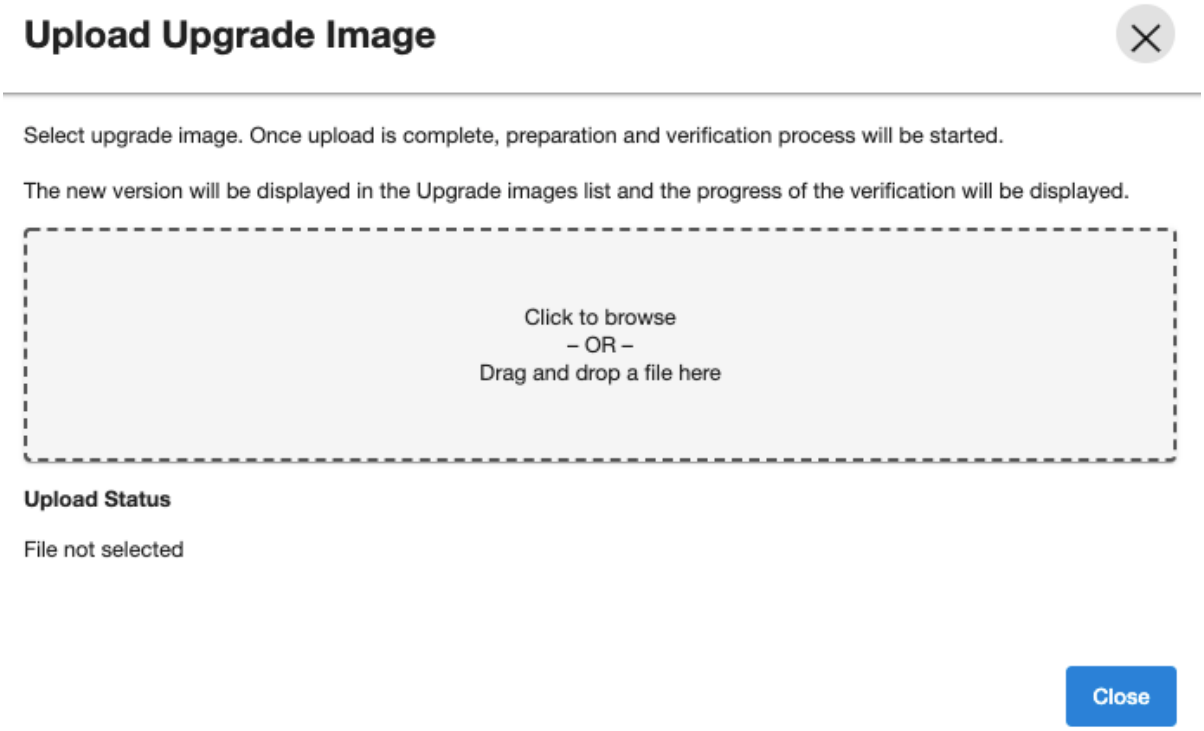
> **Version compatibility and support pre-checks**
> Please refer to Upgrade matrix before upgrading to a newer version.
> Customers running version 5.3.9 and earlier that are requesting an upgrade to 6.0.0.0 and above, please contact Delphix Support to help coordinate this upgrade.
> Upgrading from 6.0.x to 6.0.x includes pre-checks packaged in the upgrade image, contacting Delphix Support for this upgrade is **not required** (e.g. 6.0.0.0 → 6.0.9.0).

The procedure for uploading an upgrade version to the Delphix Engine is:

1. Login to the **Delphix Setup**application.
2. In the Software Version panel, click View.

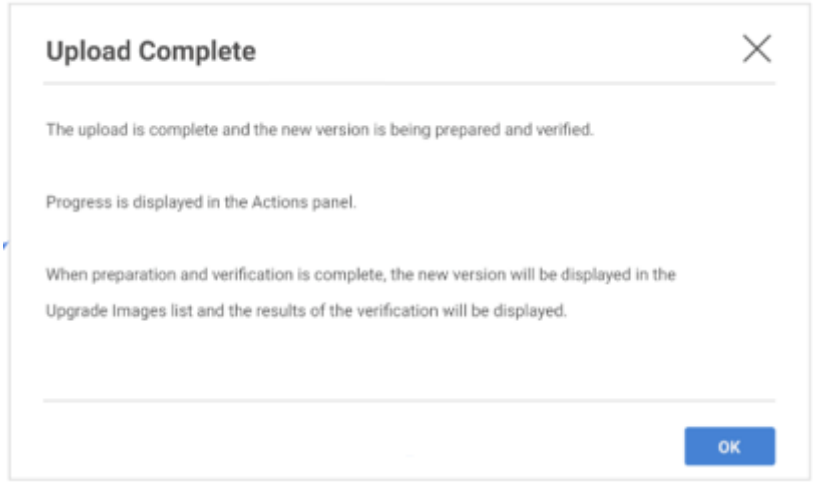


3. Click the **plus** icon to upload a new version.
4. Select the upgrade version you downloaded from the download site.



The upload begins on the drop or selection of a file (upgrade image) into the Upload Upgrade Image window. You will be able to view the progress in the Upload Status area.

5. In the Upload Complete dialog, click OK. You will see the new version appear on the page.



Once the upgrade image upload is complete, upgrade verification will start automatically

This verification process will notify you of any potential problems that require intervention from either you or Delphix support. The new version of the Delphix Engine cannot be installed before verification has completed. You can track the progress of verification from the **Actions** sidebar. If issues are found during the verification process, they will be listed on the version page. An upgrade will not be possible until the issues are resolved.

For more details, see Resolving upgrade checks

Upgrade verification

> **Version compatibility and support pre-checks**
> Please refer to Upgrade matrix before upgrading to a newer version.
> Customers running version 5.3.9 and earlier that are requesting an upgrade to 6.0.0.0 and above, please contact Delphix Support to help coordinate this upgrade.
> Upgrading from 6.0.x to 6.0.x includes pre-checks packaged in the upgrade image, contacting Delphix Support for this upgrade is **not required** (e.g. 6.0.0.0 → 6.0.9.0).

The Delphix Engine provides a feature that allows you to verify an upgrade before applying it. Each new version of the Delphix Engine can introduce new requirements for networking, hypervisor usage, the configuration of Delphix Engine objects and more. Verification can determine if the current Delphix Engine does not meet these requirements. It can also detect if the Delphix Engine is in an unexpected state before an upgrade, in which case the best solution might be to contact support. It is recommended that customers read the release notes for any version they are upgrading to and look for new Delphix Engine requirements.

As soon as the upgrade image is uploaded, a verification job is started. This verification job will use information stored within the uploaded upgrade image to examine the state of the Delphix Engine and make a best effort to validate that applying the upgrade image will succeed. It also will notify the user of any potential problems that require either customer or support intervention.

After resolving problems noted by the verification process, verification can be run again. It is expected that customers will continue to fix problems and re-verify until no more problems remain, or until none of the remaining problems are critical.

The verification does a “dry run” of some of the upgrade procedures in order to alert the administrator of potential problems before continuing with the upgrade. It is strongly recommended that you perform this verification a day or two in advance, before your upgrade downtime begins, in order to give yourself time to address any problems flagged by the verification. Perform a re-verify closer to the upgrade, when issues have been resolved.

The procedure for verifying an upgrade is:

1. Login to the **Delphix setup** application.
2. In the **System upgrade management** panel, click **View**.
3. On the left-hand side, select the version to which you will be upgrading. Details on the version will be displayed on the right.
4. In the upper right-hand section of the Details tab, click **Verify upgrade**. Verification will run in the background. You can view the different stages with the progress during the upgrade process in the **Action** sidebar.
5. Click the **Verification results** option to view problems during verification.
6. Click the **Verification steps** option to view verification steps in detail.

Understanding the verification page

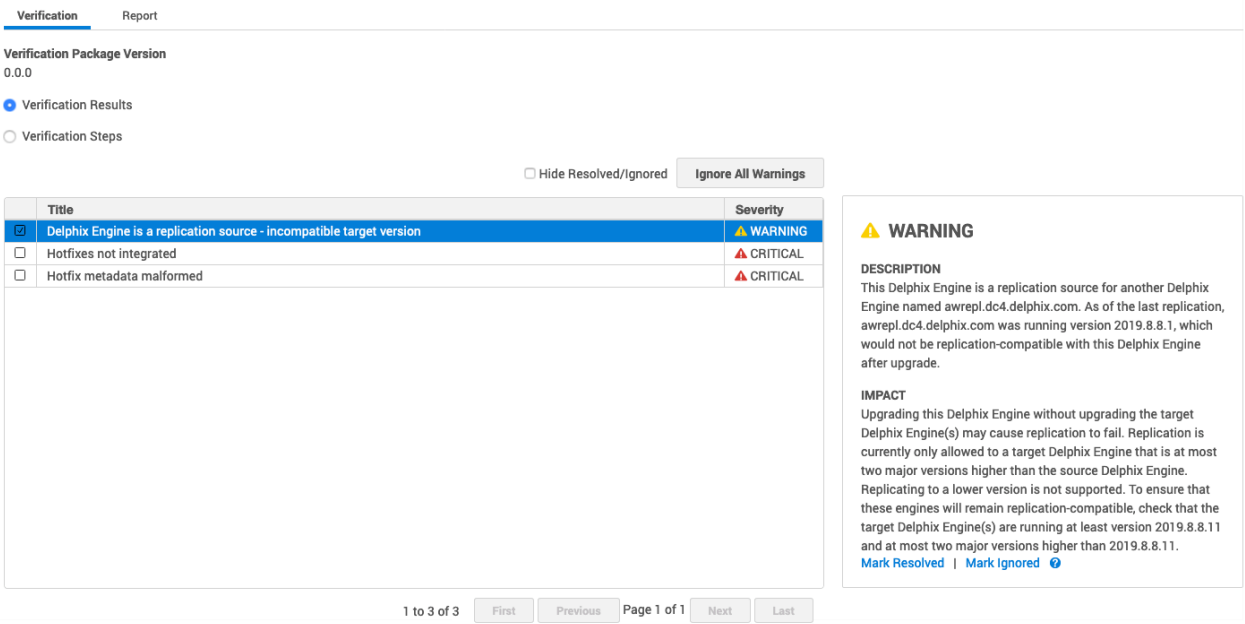
This section provides details about various components on the **Verification** page.

Verification package version displays the version of verification checks that are bundled with an upgrade version.

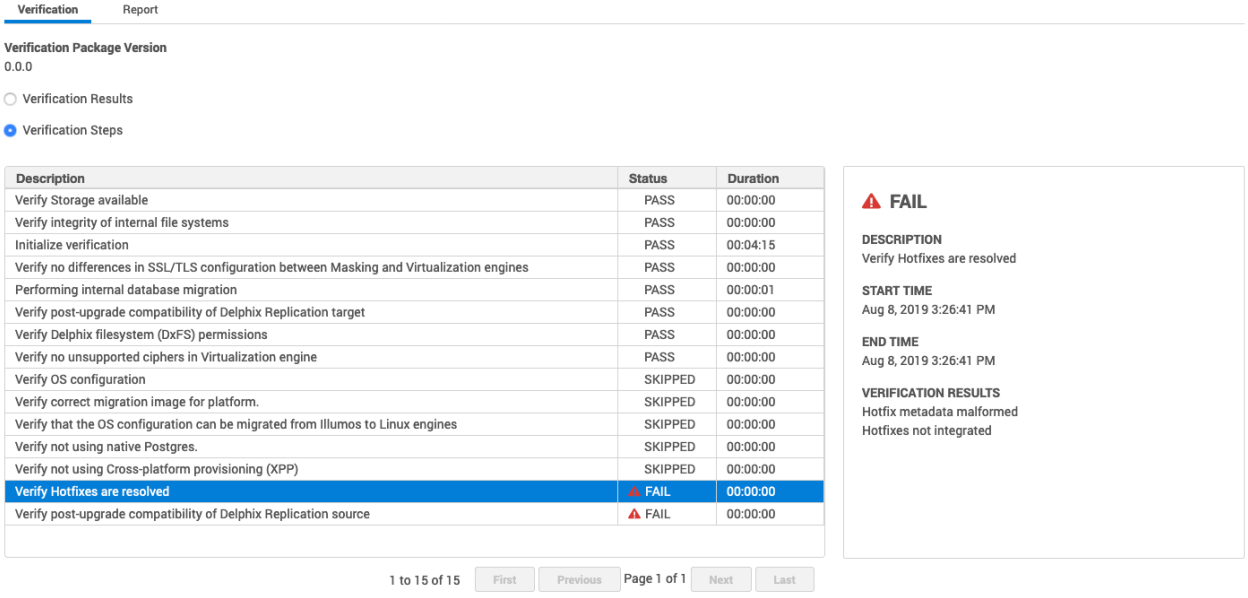
Verification results provides result details about the verification outcomes. This option is selected by default. The verification result detail includes upgrade severity, duration, and reason

Hide resolved/ignored enables hiding of all verification items that are Resolved or Ignored.

Ignore all warnings allows you to ignore all warning items during the upgrade process.



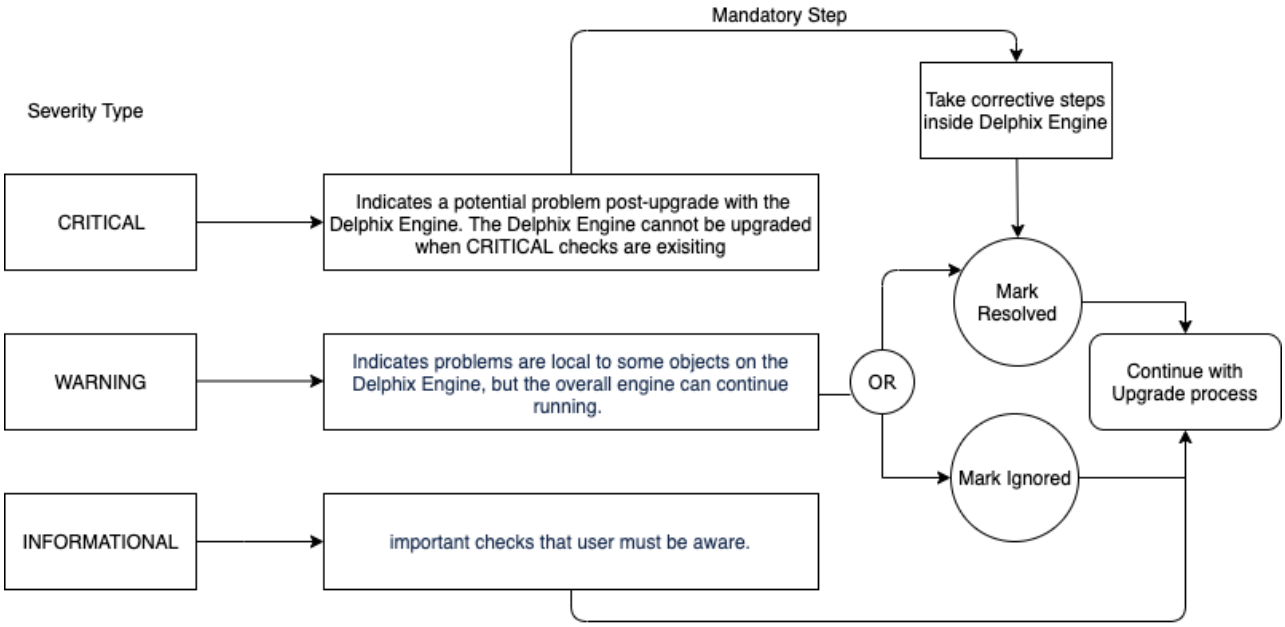
Verification steps allow you to view a complete list of verification actions, status, and durations.



Resolving upgrade checks

Overview

When verifying an upgrade, the verification job can sometimes detect particular action items called upgrade checks. These checks are items that the Delphix Engine is not capable of fixing on its own, and require customer and/or Delphix support action. Every upgrade check that appears must be resolved or ignored before the upgrade can proceed, with the exception of INFORMATIONAL checks which require no action by users. Below is a workflow for various severity types and upgrade check action.



Workflow for Severity Types and Upgrade Check Action

Upgrade severities checks

Severity	Description
CRITICAL	These checks indicate a potential problem post-upgrade with the Delphix Engine. The Delphix Engine cannot be upgraded while CRITICAL checks are present; it is likely that the engine will fail to upgrade or break catastrophically post-upgrade.

Severity	Description
WARNING	These checks indicate problems that are local to some objects on the Delphix Engine, but the overall engine can continue running. For example, WARNING checks may indicate a problem with the configuration of a particular Windows environment and indicate that if an upgrade occurs, that environment will not function properly.
INFORMATIONAL	These checks provide important information about your upgrade but do not require immediate action.

The following is a list of checks that have been added to the Delphix Engine:

1. **OS tunable settings** (CRITICAL) - Upgrade verification checks that only a certain set of operating system tunables have been adjusted. If tunable settings not on this acceptable list have been changed, please contact Delphix support.
2. **Hotfix** (CRITICAL) - Upgrade verification checks if any hotfixes have been applied to the Delphix Engine, and if any exist that are not resolved in the Upgrade version, this check will generate a Critical result. In this scenario, the Engine may still be upgraded, but we ask that you contact Delphix support to assist in addressing the check.
3. **Snapshot directory visible** (CRITICAL) - This is an internal Delphix Engine problem that Delphix support can resolve for our customers.
4. **Replication** (WARNING) - It indicates that your Delphix Engine has replication set up as either a target or source. Delphix Engines can only perform replication to engines on the same version or higher, so it is recommended to upgrade all Engines in a Replication configuration at the same time, depending on the requirements. For more information on forward-compatible replication, refer to Replication overview.
5. **Storage available** (WARNING) - Upgrade verification checks that there is sufficient space to perform data operations.
6. **SSL/TLS configuration** (WARNING) - Upgrade verification checks that the security configuration of the engine is up to date and properly configured.

An Upgrade check can result in multiple check results. Check results are essentially a to-do list of action items required before performing an upgrade. For example, the OS Tunables will create one line item in the UI for each present tunable that is not on our acceptable list. There are two actions that can be performed on each check result.

Upgrade check actions

Action	Description
resolve	You have taken the action necessary to solve the problem and complete the action item. Marking a result as resolved means that the customer believes the check result indicated will no longer be a problem. The next verification run will check the result again, but if it is fixed will not create any new check results.

Action	Description
ignore	You have no intention of fixing the problem indicated by the check result. Future verification runs will not generate that check result again if it is present. CRITICAL severity checks cannot be ignored.

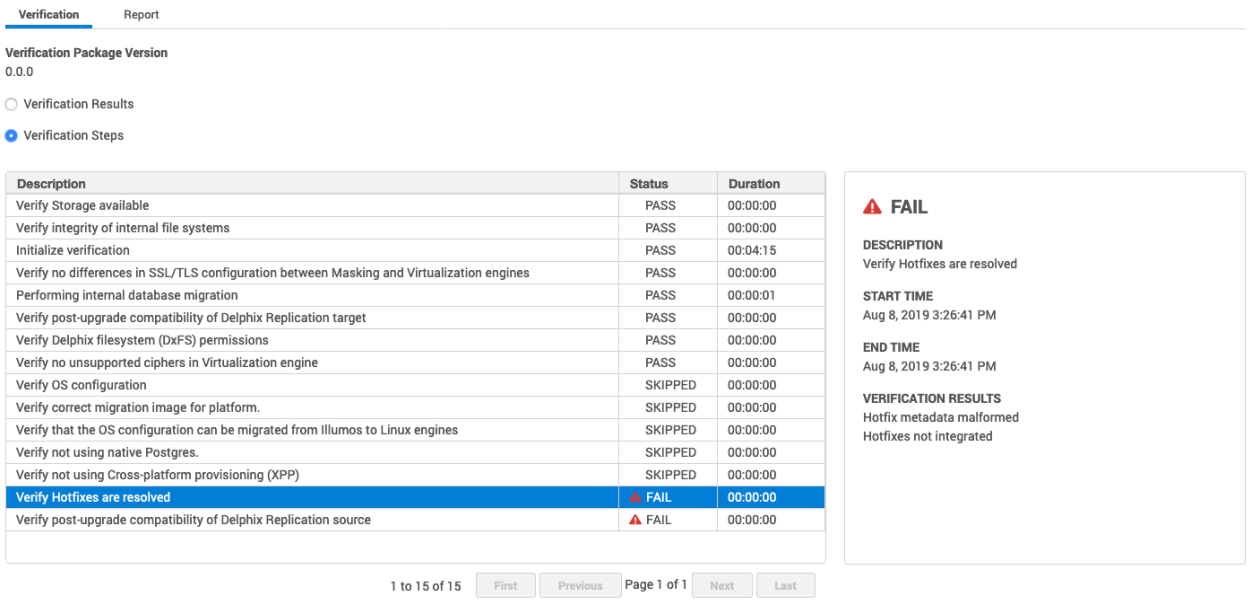
Examples of upgrade check actions

Case A: Verification results from critical severity check

Consider a case where Delphix Engine previously had a few hotfixes installed by Delphix Support to resolve an issue.

During the upgrade process, a critical severity message is displayed indicating the hotfixes that was used to resolve issue might be lost or reappear after the upgrade.

Hence new upgrade version is missing the root cause fix for that issue. In this example, the action is either to contact Delphix Support or upgrade to a new version that no longer needs these hotfixes. After taking the necessary action, you must mark resolved for the verification steps to complete for a successful upgrade.



☐ It is mandatory to resolve all critical severity errors

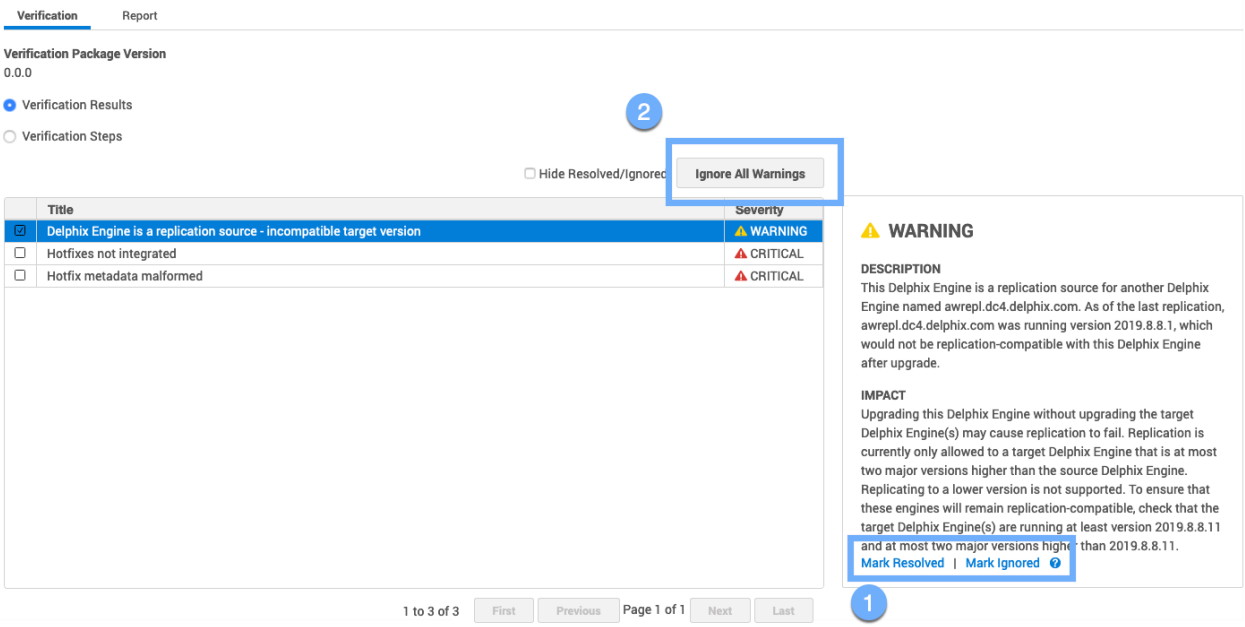
Case B: Verification results warning severity check with ignore all warnings

Consider a case where Delphix Engine is configured as a Delphix Replication source for another Delphix Engine.

During the upgrade process, a warning message appears because the target engine is on a version that is incompatible with the source engine version.

Hence replication is interrupted until the target engine is upgraded to a newer version.

In this example, the action steps are either to mark the issue as **Mark resolved** or **Mark ignored** because a replication-compatibility is acceptable with a few limitations (as indicated in below warning message).



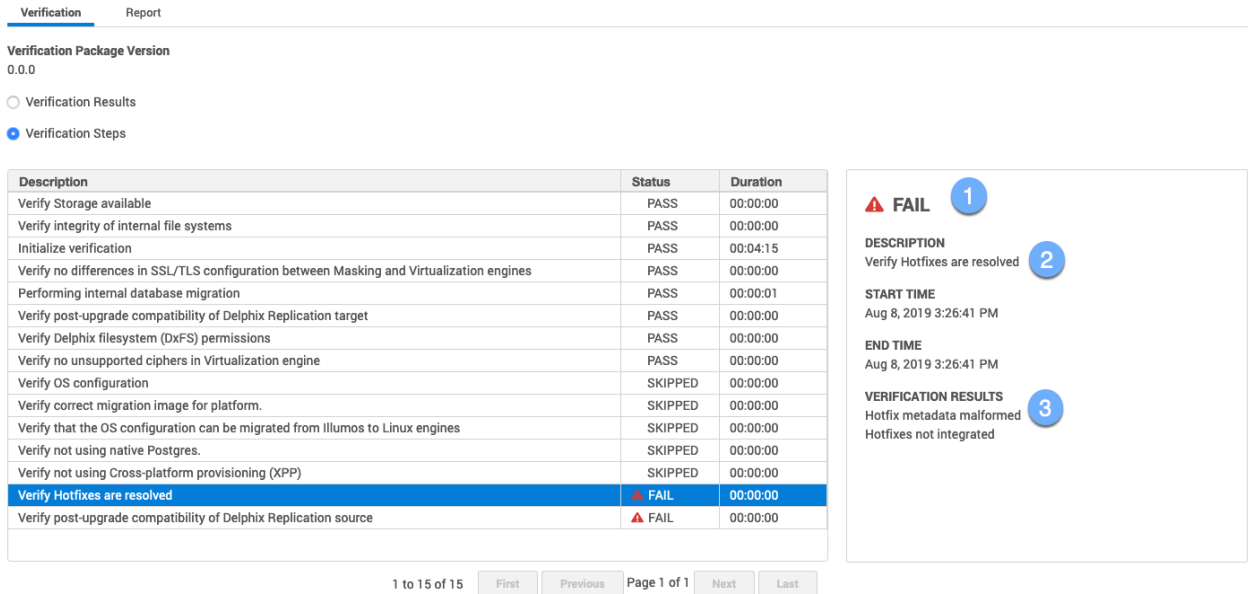
☐ All Warnings must either be **Mark resolved** or **Mark ignored**

☐ You also can use **Ignore all warnings** option to disregard all warning messages at once.

Case C: Verification steps upgrade failure

When you want to see more details about the verification process use the **Verification steps** radio button. Verification Steps is an alternate way to view the verification status. It includes all the steps that were performed by the verification process, the status, description and the duration of the step.

In this example below verification status, **FAIL** indicates that a verification result was generated. In the **Verification results** view, a **FAIL** status on this screen may correspond to a CRITICAL, WARNING, or INFORMATIONAL severity.



- [ ] Failure severity
- [ ] The description indicates why upgrade failure occurred
- [ ] Verification results

Applying the upgrade

> **Version compatibility and support pre-checks**
> Please refer to Upgrade matrix before upgrading to a newer version.
> Customers running version 5.3.9 and earlier that are requesting an upgrade to 6.0.0.0 and above, please contact Delphix Support to help coordinate this upgrade.
> Upgrading from 6.0.x to 6.0.x includes pre-checks packaged in the upgrade image, contacting Delphix Support for this upgrade is **not required** (e.g. 6.0.0.0 → 6.0.9.0).

Once you have uploaded an upgrade version, verified the upgrade, optionally reviewed the warnings in the Impact of Upgrade section, scheduled downtime pertaining to the type of upgrade you are performing, you can apply the upgrade.

1. Login to the **Delphix setup** application.
2. In the **Upgrade images** panel, click **View**.
3. On the left-hand side, select the version to which you will be upgrading.
4. Click **Apply upgrade** to initiate the upgrade process.
    a. For upgrades on engines already running 6.0.0.0 and greater, the user can choose the type of upgrade they want to perform.

The upgrade will run in the background. You can view the progress of the upgrade in the Action sidebar. Only the current system admin user can view the progress.

The status of the upgrade will be visible on the screen - if the upgrade is successful, the page will be redirected to the login view.

If an **Apply now** upgrade fails, the appliance will automatically roll back to the version running prior to the upgrade.

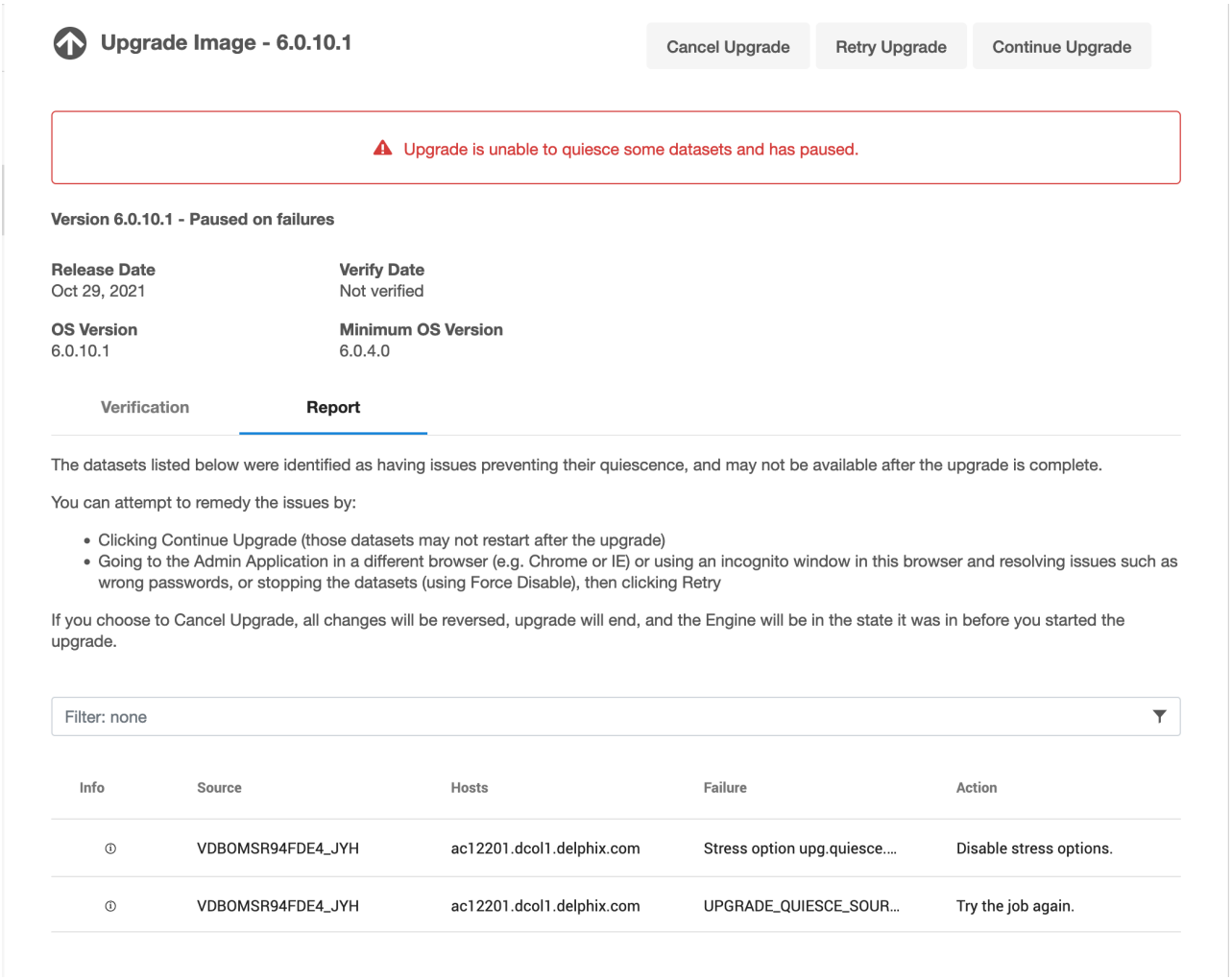
The version page will show the new version in an UPLOADED state and the Action sidebar will show that a rollback was performed. If automatic rollback was disabled through the CLI (not advised), you will have to contact support to proceed further, since you may not even be able to log in to the Delphix Engine.

Failure to quiesce a dataset

If Upgrade has failed to quiesce a dataset, it will pause and you will see the banner at the top of the **Upgrade** page as shown below:

While the upgrade is paused, datasets that have been quiesced are unavailable until you either roll back or continue the upgrade.

To review the list of failures, open the **Report** tab:



The datasets listed in the report were identified as having issues that prevented them from being quiesced, and may not be available after the upgrade is complete. Review the messages in the report and take the suggested corrective actions.

If you think that the errors may be the result of transient failures, you can click **Retry upgrade** to try again. Otherwise, it is recommended that you manually quiesce datasets that are still running. To do so:

1. Use a different browser or use an incognito window to go to the **Delphix management** application.

> ⚠ Failure to do so might result in losing the current session and you may not be able to return to the Upgrade pause screen.

2. Either resolve issues such as a wrong password or stop the dataset using **Force disable**.

3. In the original browser or window, click **Retry upgrade** to try applying the upgrade again.

If you want to ignore the failures to quiesce datasets and proceed with the upgrade:

1. Click **Continue upgrade**. This will attempt to quiesce all datasets which have not yet been quiesced, but will not pause on failures.
   Note : This may result in datasets remaining unavailable after the upgrade is complete and the Delphix Engine restarts, since the underlying storage that backs the datasets will be unreachable during the

upgrade. This may cause the databases or applications to failover or transition to a failure state, thus requiring administrator intervention to recover.

2. Review the messages in the report and take the suggested corrective actions.
3. If any of the listed datasets are critical, and you are unable to resolve the configuration errors in the report, you can **Rollback** the upgrade. If you choose **Rollback**, all changes will be reversed, the upgrade will end, and the Delphix Engine will be in the state it was in before you started the upgrade.

Post upgrade

After the upgrade is done, you will be redirected back to the login page.

Login to **Delphix setup** to make sure that the upgrade succeeded and that the new version is running.

A **post-upgrade cleanup** job is run automatically after the upgrade is done. This job refreshes environments and re-enables sources to bring objects back into a working state. VDB access will not be available at all until the environments have been refreshed and the objects are re-enabled by the job **Perform cleanup tasks following a Delphix engine upgrade**.

Failures

If you used **Continue upgrade** to force the upgrade as is described in Failure to quiesce a dataset, the **Report** tab will contain the list of pre-upgrade quiesce failures. Upgrade will make a best effort to restore functionality to these datasets, but it may still hit the same errors that prevented the datasets from being quiesced successfully before the upgrade. You may need to manually bring up these datasets on the target hosts.

Security

Given that the Delphix software is meant to interact with some of your organization's most important application data, security is very important. The security section covers everything you need to know about how you can manage the Delphix software securely (user management, etc) as well as how Delphix works to ensure that the software is not susceptible to vulnerabilities. This section outlines standard hardening techniques that you should apply to every Delphix Engine in your environment. The Delphix Engine delivers its functionality through a web-based graphical user interface (GUI), web-based RESTful APIs, and a command-line interface (CLI) accessed over SSH. All three interfaces leverage the same accounts and privileges scheme. Access to the Delphix Operating System (DxOS) is restricted to Delphix Support and Delphix Professional Services and can be controlled through the Support Access Control mechanism, described below. Customer-specific software installations and modifications to the DxOS are not required or supported. Communications to/from connected systems ("sources" and "targets") should be limited to required ports, and encryption should be used wherever possible.

This section covers the following topics:

Security principles

The Delphix approach is based on:

Embrace separation of duties: Isolate and compartmentalize capabilities and privileges and never give or concentrate access to a single role.

Apply the principle of least privilege: Users should obtain only those privileges needed to do their jobs and only for as long as they are needed.

Use an open, simple design: Make security mechanisms simple and easy to use, and rely on proven, peer-reviewed solutions.

Use a layered defense: Provide no single point of failure; if one layer fails to catch an error, catch it in another layer.

Use complete mediation and authentication: Control and check every access point every time.

Use fail-safes: Deny access when not explicitly authorized. Prevent faults from causing an opportunity to compromise.

Protect data at rest and data in motion: Utilize common security protocols as well as features of the source database and database software to protect data at all times.

Minimize the attack surface: Present the minimum sockets, services, webpages, and accounts necessary to operate.

Don’t rely on obscurity: Be secure even if everything but the key is known.

Audit and monitor everything: Provide a tamper-proof trail of evidence.

Leverage the environment: Design the Delphix Engine to leverage the security features offered by databases, operating systems, storage devices, and networks.

Anticipate external attack vectors: Combat attacks sourced from connected systems.

Enforce strong credentials: Define and enforce password policies.

Product security

This section covers the following topics:

Delphix product security

Overview

Delphix takes an active approach towards the discovery of vulnerabilities by employing third-party VAPT (vulnerability and penetration testing) Teams on an annual basis, as well as actively reviewing additional tools to assess our products. We also actively monitor bugs and vulnerabilities against the list of open-source software which is integrated into our products.

Delphix investigates all known vulnerabilities, either found by customers or internally, or discovered by the VAPT Teams. During the course of this investigative process, Delphix works with the reporter of the vulnerability to gather the technical information and determine the appropriate remedial action.

Customers can perform their own security scans and audits using their access through application administrative accounts. Because operating system administrative access is not provided with the closed software virtual appliance, scans executed with this privilege level are consistent with the level of access available in the environment.

As needed, we provide compensating controls and mitigations to reduce the impact of security issues more quickly and deliver fixes as part of the standard software release process described above.

Software delivery security

We deliver our virtual software appliance and upgrade images to customers on a secure server with access control. Additionally, we provide cryptographic signatures for each image, and customers can use these signatures to ensure the software images have not been tampered with.

Ancillary components

The virtual appliance communicates with target servers running virtual databases (VDBs). The software is pushed to target servers to facilitate communication for Oracle and SQL Server databases. (Oracle toolkit and Delphix connector, respectively). This software lives outside of the DDDP product appliance, and new versions of the DDDP product will provide updated versions of this software as well.

Software updates

Keeping Software up to date is an important part of any hardening plan. Delphix software releases are cumulative and include bug fixes, new features, and security improvements. In addition, Delphix releases hotfixes, procedures, and workarounds for critical vulnerabilities.

Patch annually

To stay up to date, patch your Delphix Engine at least once per year. If you do not, you might have to upgrade twice to get the latest releases, and your old installed version will not be able to receive vulnerability fixes.

If possible, patch more frequently. Depending on the version you are upgrading from/to, you may be able to avoid or defer the reboot sequence, which defers downtime for your virtual databases (VDBs). This allows you to patch outside of downtime windows.

Subscribe to Delphix notifications

Delphix issues email notifications when critical vulnerabilities are discovered. Registered support accounts will automatically receive these notifications. To ensure that you receive these notifications:

- Register at least two Engine Admins with Delphix Support
- Add Delphix Support accounts when Engine Admins leave the company

Network security

Review the official documentation for the full list of required ports, *which depends on your database vendor*. Open only those ports that are required. The following table only lists generic requirements; you will need additional ports to integrate with databases.

Ports

Open only required ports.

General port allocation

The Delphix Engine makes use of the following network ports irrespective of the type of database objects on it:

General outbound port allocations

Protocol	Port numbers	Use
TCP	21	Passive FTP connections from the Delphix Engine to the Delphix FTP server. Used for sending logs to Delphix Support.
TCP	25	Connection to a local SMTP server for sending email.
TCP/UDP	53	Connections to local DNS servers
UDP	123	Connection to an NTP server
TCP/UDP	389	Standard access to an LDAP server
TCP/UDP	636	Secure access to an LDAP server
TCP	1023	Used to complete the passive FTP connection from the Delphix Engine to Delphix Support. These ports are not used if Delphix uses a proxy server to connect to the Internet.
TCP	8415	A Delphix replication source will connect to the replication target using this destination port

General inbound port allocations

Protocol	Port numbers	Use
TCP/UDP	22	SSH connections to the Delphix Engine

Protocol	Port numbers	Use
TCP	80	HTTP connections to the Delphix GUI
TCP	443	HTTPS connections to the Delphix GUI
TCP	8415	A Delphix replication target will accept incoming connections to this port from the replication

Password vault support

Overview

More and more organizations use Enterprise Password Vaults (EPV) such as CyberArk and HashiCorp Vault to store securely and centrally manage identities and credentials. Delphix has added CyberArk, HashiCorp and Azure Key Vault support to the Delphix Virtualization Engine as a new authentication option for environments and databases. This minimizes the number of places where credentials need to be stored and, therefore the risk of insecure storage.

The Delphix Engine uses various authentication methods such as username/password, username/ssh key, and Kerberos credentials when connecting to hosts and databases from the Engine. These credentials are stored on the Delphix Engine in an encrypted format and can be retrieved later to perform various operations. Delphix provides an additional authentication method by integrating Virtualization with the most common vault types (CyberArk, HashiCorp. Azure Key Vault). At runtime, Delphix retrieves the credentials (passwords, ssh keys) from the customer's vault servers via API calls and avoids managing customer passwords.

Configuring password vaults

In the Setup app, system administrators can manage (add, delete, modify, and validate) vault configurations during and after the initial setup. Each engine can have multiple vaults configured of any type.

The authentication method supported for CyberArk is Certificate-based. Configuring a CyberArk vault requires providing a host address, port number, application ID, and a client certificate (certificate chain and private key).

The authentication methods supported for HashiCorp vaults are Token-based, AppRole-based, and Certificate-based. Configuring a HashiCorp vault requires providing a token ID or a role ID and secret ID or a client certificate along with the host address and port number. For HashiCorp Enterprise vaults, a namespace can also be provided.

The authentication method supported for Azure Key Vault is Client-Secret-based. Configuring an Azure Key Vault requires providing the Azure tenant ID, client ID, and client secret.

Using password vaults

The Virtualization engine retrieves credentials at runtime from a vault using a unique identifier that locates a set of credentials in a configured vault. This occurs for any activity that requires Environment access (SnapSync, Validated Sync, LogSync, as well as Environment monitoring). This may result in a significant number of requests, so any existing connection rate limits should be evaluated and adjusted accordingly. For CyberArk, the unique identifier consists of a query string. For HashiCorp Vault, it consists of four parameters: engine, path, and a pair of keys that locate the username and secret (password or SSH key) in the key-value store at that engine and path. For Azure Key Vault, it consists of the Azure Vault Name (also known as the Resource Name within Azure Vault, not to be confused with the vault name that identifies the vault in the engine), username key, and secret key.

To set up an environment or database user to use a vault, use the credential type **VaultCredential** when adding/modifying such users and specify the vault and the unique identifier of the credentials.

Roles and privileges for CyberArk and HashiCorp users

Role	Privileges
System Administrator	Can add, modify, delete, and list vault configurations.

Role	Privileges
Delphix Administrator	Can list existing vault configurations and link environment and database users to vault credentials.

Supported environments and databases

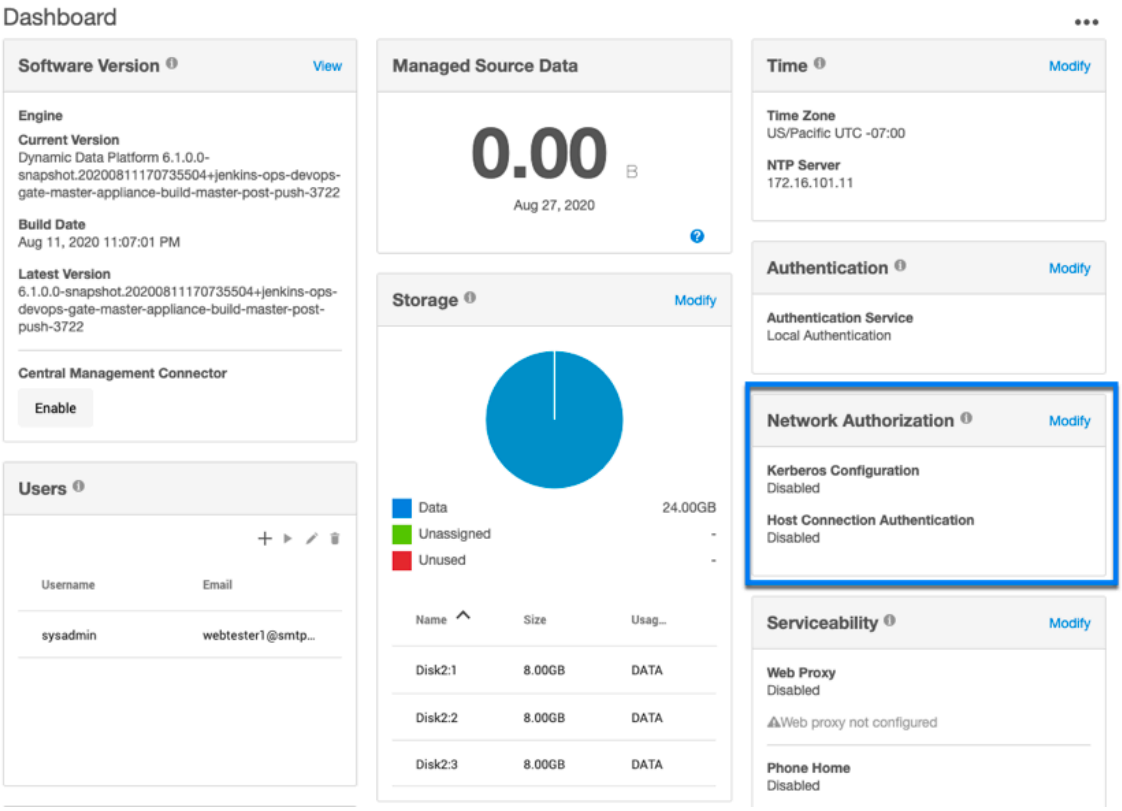
All environment users can use vault credentials. For Windows, the initial link via the Delphix Connector does not support vaults, but environment users can be subsequently updated to use vaults.

Vault integration is currently supported for SAP ASE database users, Oracle database users, and MSSQL domain users using 6.0.4 and later.

Setting up a vault via GUI

Complete setup via the GUI is available for CyberArk as of 6.0.3.0, for HashiCorp as of 6.0.4.0 and for Azure Key Vault as of 10.0.0.0.

1. Connect to the Delphix Engine http://
2. Add a CyberArk or HashiCorp CA certificate to the TrustStore as part of the initial configuration. Refer TrustStore Settings for steps to add a CA certificate.
3. Click on the **Modify** link in the top right of the **Network Authorization** panel.



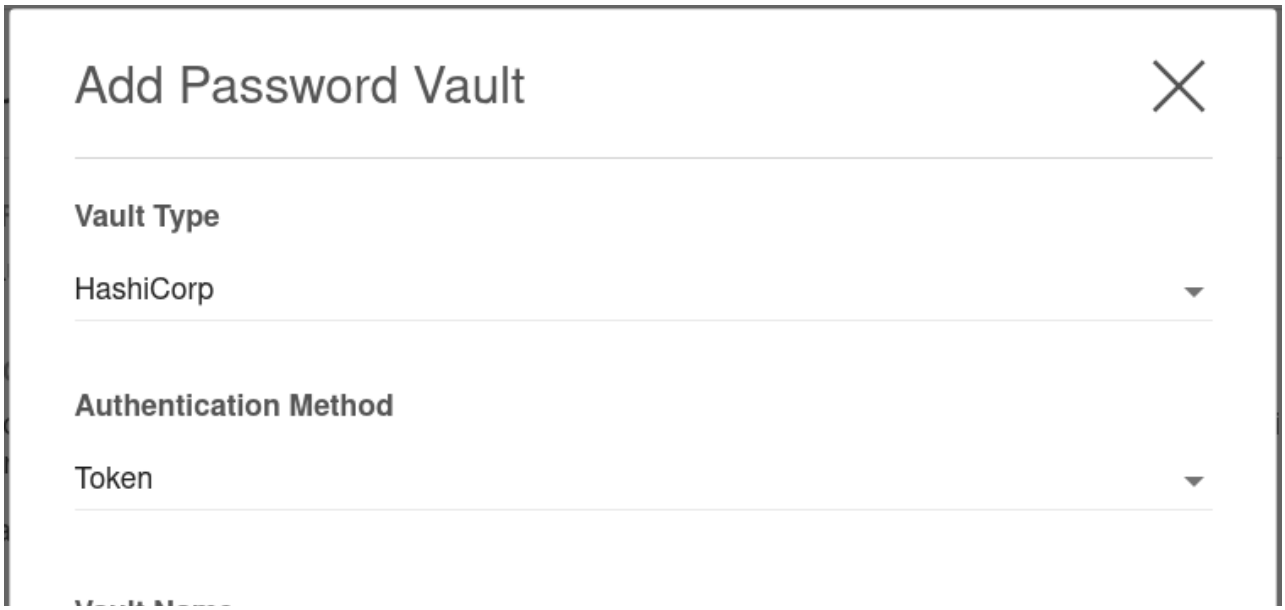
4. On the Network Authorization window, click "**+**" to add a new vault.
5. Enter the following information: Depending upon your requirements, you can set the configurations for CyberArk, HashiCorp, or Azure Key vault.

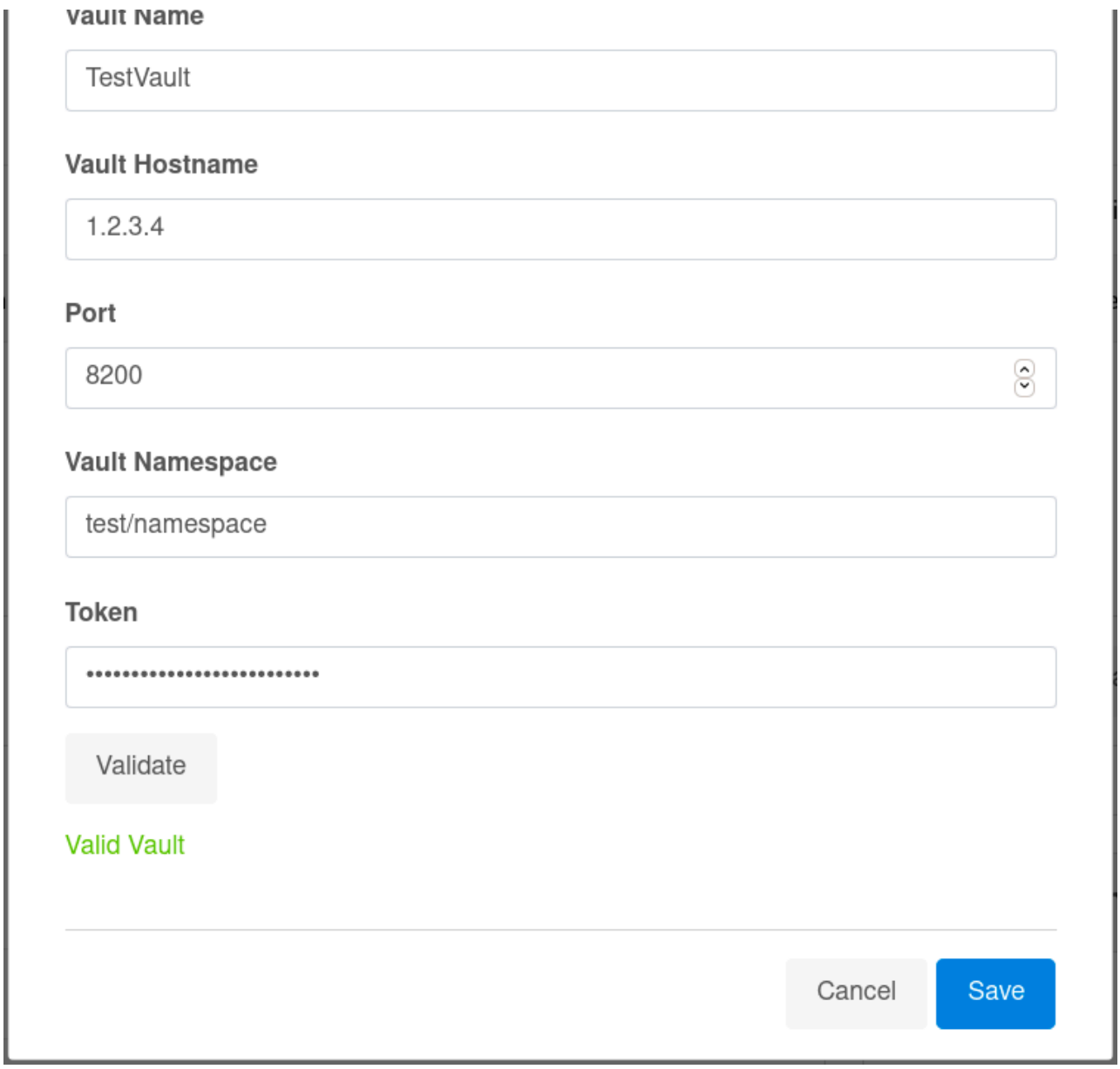
Field	Possible value and Data Type	Description
Vault Type - CyberArk		
Vault Name	*<user-specified>* Accepts a string value	Specifies the user-specified vault name
Vault Hostname	mycyberark.myorg.com Accepts a URL string value	Specifies the location of the user's vault server
Port	443 Accepts an integer value	Specifies the port number through which the communication will happen
App ID	MyAppID Accepts a string value	Specifies an application ID registered with and provided by CyberArk
Authentication Certificate	——-BEGIN CERTIFICATE—-— *<certificate>* ——-END CERTIFICATE—-— Accepts a string value	Specifies the authentication certificate provided by CyberArk
Private Key	*<CyberArk-provided>* Accepts a string value	Specifies the private key provided by CyberArk for TLS based authentication
Vault-Type - HashiCorp		
Authentication method	**Token**	
Vault Name	*<user-specified>* Accepts a string value	Specifies the user-specified vault name
Vault Hostname	12.345.678.90 Accepts a URL string value	Specifies the location of the customer vault server
Port	8100 Accepts an integer value	Specifies the port number through which the communication will happen

Field	Possible value and Data Type	Description
Vault Namespace	purple Accepts a string value	Specifies the namespace configuration specific to the user environment that is provided by the HashiCorp Enterprise Platform
Token	s.abcdefghijklmnopqrstuvwxyz123.waR7a Accepts a string value	Specifies the token specific to the user environment that is provided by the HashiCorp Enterprise Platform
Authentication method	**AppRole**	
Vault Name	*<user-specified>* Accepts a string value	Specifies the user-specified vault name
Vault Hostname	12.345.678.90 Accepts a URL string value	Specifies the location of the customer vault server
Port	8100 Accepts an integer value	Specifies the port number through which the communication will happen
Vault Namespace	purple Accepts a string value	Specifies the namespace configuration specific to the user environment. This feature is provided with the HashiCorp Enterprise Platform
RoleID	abcdefg123-4a56-7890-a2bc-34567def8901 Accepts a string value	Specifies the RoleID specific to the user environment
SecretID	ab1cde0f-123g-4h56-i789-1234jk567890 Accepts a string value	Specifies the SecretID specific to the user environment
Authentication method	**Certificate**	
Vault Name	*<user-specified>* Accepts a string value	Specifies the user-specified vault name

Field	Possible value and Data Type	Description
Vault Hostname	12.345.678.90 Accepts a URL string value	Specifies the location of the customer vault server
Port	8100 Accepts an integer value	Specifies the port number through which the communication will happen
Vault Namespace	purple Accepts a string value	Specifies the namespace configuration specific to the user environment. This feature is provided with the HashiCorp Enterprise Platform
Authentication Certificate	—–-BEGIN CERTIFICATE—-– *<certificate>* —–-END CERTIFICATE—-– Accepts a string value	Specifies the authentication certificate provided by HashiCorp for TLS based authentication
Private Key	abcdefg123-4a56-7890-a2bc-34567def8901 Accepts a string value	Specifies the private key specific to the user environment that is provided by HashiCorp for TLS based authentication
Role Name (Optional)	purple-admin-role Accepts a string value	Specifies the certificate role name for TLS based authentication

6. Click **Validate** to check the configurations before saving the vault details. The below screenshot shows an example of the HashiCorp Vault configuration.





7. Click **Save**.
The added configurations can be viewed in the Network Authorization window.

Editing a vault via the GUI

1. Connect to the Delphix Engine http:// <Delphix Engine>/login/index.html#serverSetup.
2. Click on the **Modify** link in the top right of the **Network Authorization** panel.
3. In the **Network Authorization** window, select a vault, then the **pencil** icon.
4. Edit your configuration.
5. Select **Edit**.

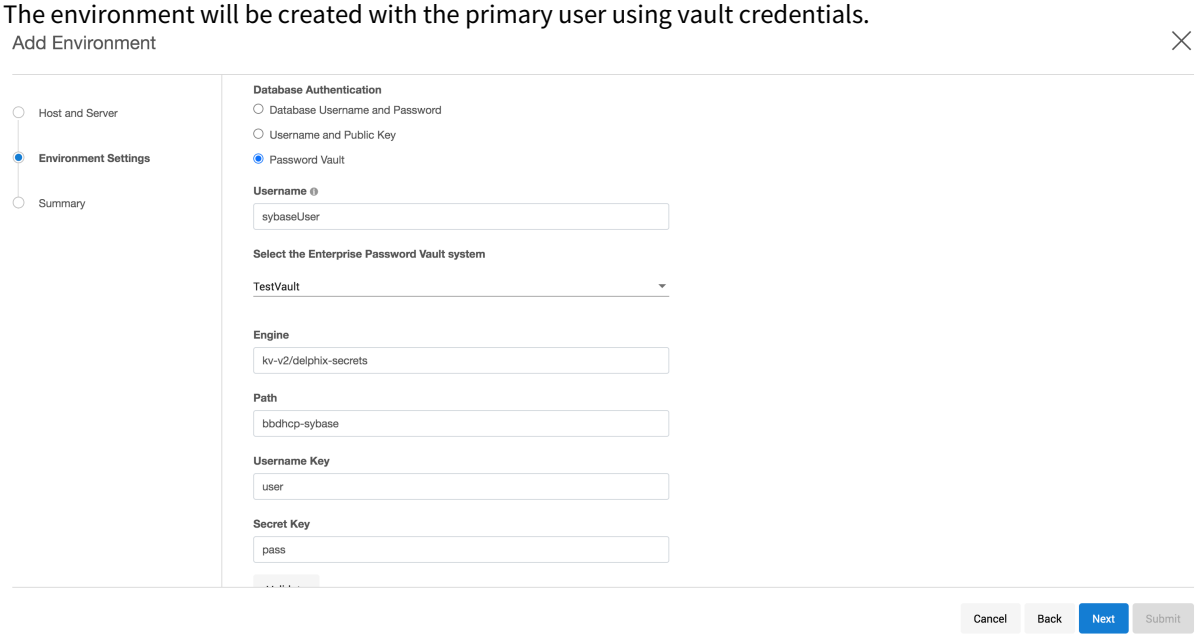
Deleting a vault via the GUI

1. Connect to the Delphix Engine http:// <Delphix Engine>/login/index.html#serverSetup.
2. Click on the **Modify** link in the top right of the **Network Authorization** panel.
3. In the **Network Authorization** window, select a vault, then select the **trashcan** icon.

4. Select **Yes** to delete the vault.

Adding a host user for HashiCorp

1. Login to the **Delphix Management Application** and select **Manage > Environments**.
2. Select Add **Environment**.
3. In the **Environment Setting** tab, select **Password Vault** as the Login Type.
4. Select the vault configuration and provide the secret engine name, path, and keys for the username secret and complete your environment configuration.
5. The environment will be created with the primary user using vault credentials.

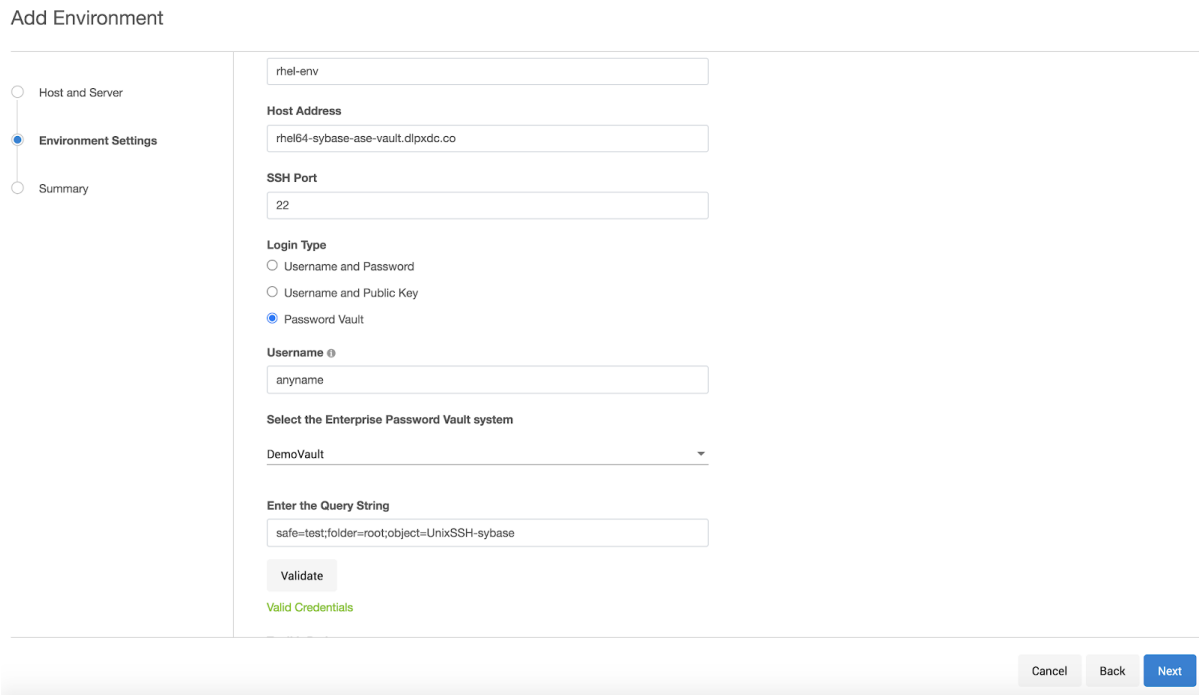


Adding a database user for HashiCorp

1. Login to the **Delphix Management Application**.
2. Add dSource using database credentials from HashiCorp vault by selecting **Password Vault** as the **Login Type**.
3. Provide the appropriate secret engine name, path, and keys for the username secret and complete configuration.

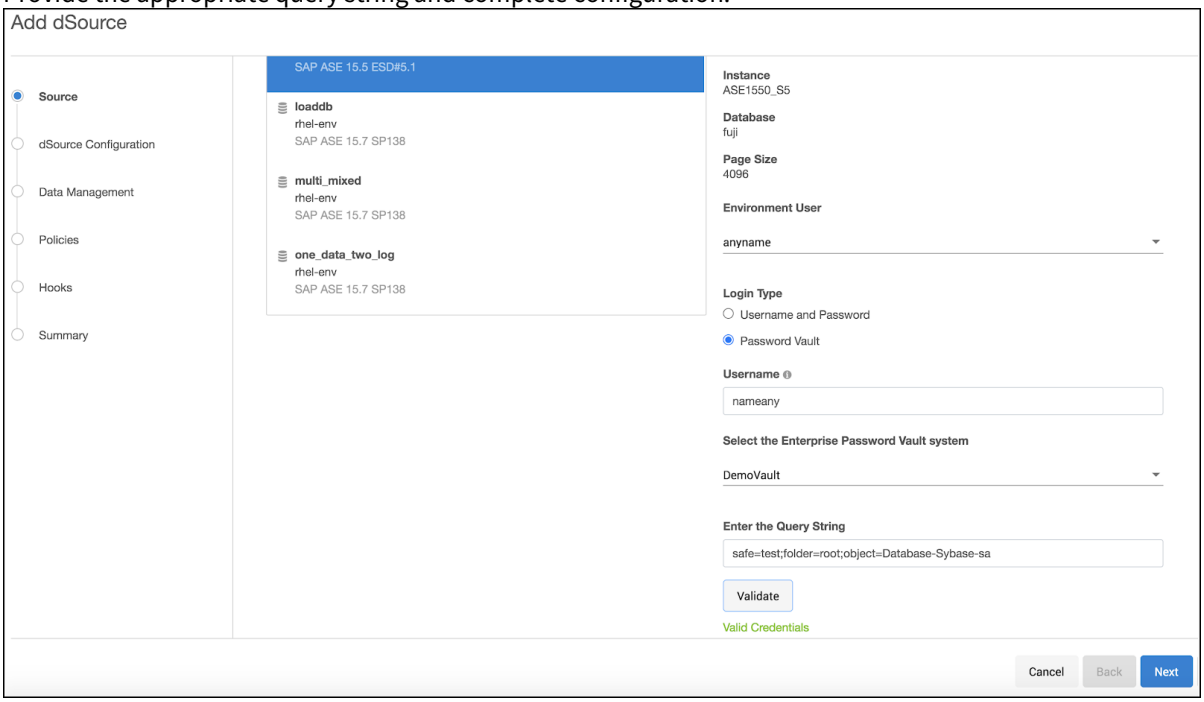
Adding a host user for CyberArk

1. Login to the **Delphix Management Application** and select **Manage > Environments**.
2. Select Add **Environment**.
3. In the **Environment Setting** tab, select **Password Vault** as the Login Type.
4. Select the vault configuration and provide the username, select the enterprise password vault system, and enter a query string that is a unique identifier pointing to the credentials to be retrieved and complete your environment configuration.
5. The environment will be created with the primary user using vault credentials.



Adding a database user for CyberArk

1. Login to the **Delphix Management Application**.
2. Add dSource using database credentials from CyberArk vault by selecting **Password Vault** as the **Login Type**.
3. Provide the appropriate query string and complete configuration.



Setting up a vault via CLI

1. Login as a system administrator and add a CyberArk or HashiCorp CA certificate to the TrustStore as part of the initial configuration.

2. SSH to <Delphix Engine IP>service passwordVault and enter **create**.

```
ip-10-110-230-197 service passwordVault>create
```

3. Add a new vault configuration by entering a name, host, port, applicationId, client certificate, and private key.

```
ip-10-110-230-197 service passwordVault create
Properties
type: CyberArkPasswordVault
   name: DemoVault (*)
   applications: Delphix (*)
   clientCertificate:
      type: PemClientCertificate (*)
      clientCertificateChain: (required)
      privateKey:(required)
host: services-uscentral.sktyop.com (*)
port: 17993 (*)
```

4. Add a new HashiCorp vault configuration by entering a name, host, port, and other authentication information based on the authentication method (Token/AppRole/Certificate) selected.
HashiCorp - Token Based Authentication

```
ip-10-110-230-197 service passwordVault create
Properties
    type: HashiCorpVault (*)
    name: HashiCorpDemoVault (*)
    authentication:
        type: HashiCorpTokenAuthentication (*)
        token: ******** (*)
    host: 10.119.132.40 (*)
    port: 8200 (*)
```

HashiCorp - AppRole Based Authentication

```
ip-10-110-230-197 service passwordVault create
Properties
    type: HashiCorpVault (*)
    name: HashiCorpDemoVault (*)
    authentication:
        type: HashiCorpAppRoleAuthentication (*)
        roleId: 20d19a46-6fd9-c78b-b7e3-e43be4c8d5c2 (*)
        secretId: ******** (*)
    host: 10.119.132.40 (*)
    port: 8200 (*)
```

HashiCorp - Certificate Based Authentication

```
ip-10-110-230-197 service passwordVault create
Properties
    type: HashiCorpVault (*)
    name: HashiCorpDemoVault (*)
    authentication:
        type: HashiCorpCertificateAuthentication (*)
        clientCertificate:
            type: PemClientCertificate (*)
            clientCertificateChain: (required)
            privateKey: (required)
        roleName: (unset)
    host: 10.119.132.40 (*)
    port: 8200 (*)
```

Updating an existing vault configuration

```
ip-10-110-230-197 service passwordVault> select DemoVault
ip-10-110-230-197 service passwordVault 'DemoVault'>update
ip-10-110-230-197 service passwordVault 'DemoVault'update *> set name=TestVault
ip-10-110-230-197 service passwordVault 'DemoVault'update *> commit
ip-10-110-230-197 service passwordVault 'TestVault'>
```

Deleting an existing vault configuration

```
ip-10-110-230-197 service passwordVault 'TestVault'> delete
ip-10-110-230-197 service passwordVault 'TestVault' delete *> commit
ip-10-110-230-197 service passwordVault>
```

Adding/Modifying host users

Add an environment with user credentials from CyberArk vault. When adding a host/database user with a vault credential, the name field would be a user identifier and not the actual username. In case this field is empty, a unique identifier is generated with a hash of vault credentials.

```
ip-10-110-230-197 environment create *> set hostEnvironment.name=bbh-env
ip-10-110-230-197 environment create *> set hostParameters.host.address=bbdhcp-vault-
demo.dlpx.co
ip-10-110-230-197 environment create *> set hostParameters.host.toolkitPath="/work"
ip-10-110-230-197 environment create *> set primaryUser.name=oracleUser
ip-10-110-230-197 environment create *> set
primaryUser.credential.type=VaultCredential
ip-10-110-230-197 environment create *> set primaryUser.credential.vault=DemoVault
ip-10-110-230-197 environment create *> set
primaryUser.credential.vaultCredentialId="safe-test;folder=root;object=UnixSSH-
sybase"
```

```
ip-10-110-230-197 environment create *> commit
    `UNIX_HOST_ENVIRONMENT -6
    Dispatched job JOB-33
    ENVIRONMENT_CREATE_AND_DISCOVER job started for "bbh-env".
    ENVIRONMENT_CREATE_AND_DISCOVER job for "bbh-env" completed successfully.
```

Adding/Modifying **database users**

Add dSource using database credentials from CyberArk vault.

```
ip-10-110-230-197 database link *> set name=fuji
ip-10-110-230-197 database link *> set group=Untitled
ip-10-110-230-197 database link *> set linkData.config=ASE_SI_CONF-70
ip-10-110-230-197 database link *> set linkData.dbUser=sybaseUser
ip-10-110-230-197 database link *> set linkData.dbCredentials.type=VaultCredential
ip-10-110-230-197 database link *> set linkData.dbCredentials.vault=DemoVault
ip-10-110-230-197 database link *> set
linkData.dbCredentials.vaultCredentialOd="safe-test;folder=root;object=Database-
Sybase-sa"
ip-10-110-230-197 database link *> set linkData.loadBackupPath='/opt/sybase/dumps"
ip-10-110-230-197 database link *> set linkData.sourceHostUser=HOST_USER-7
ip-10-110-230-197 database link *> set linkData.stagingHostUser=HOST_USER-7
ip-10-110-230-197 database link *> set linkData.stagingRepository=ASE_INSTANCE-6
ip-10-110-230-197 database link *> set
linkData.syncParameters.type=ASENewBackupSyncParameters
ip-10-110-230-197 database link *> commit
    `ASE_DB_CONTAINER-1
    Dispatched job JOB-39
    DB_LINK job started for "Untitled/fuji".
    DB_LINK job for "Untitled/fuji" completed successfully.
```

Update Existing Database Users

Convert an existing database to use vault credentials for the existing database user.

```
ip-10-110-230-197 > sourceconfig
ip-10-110-230-197 sourceconfig > select MyOraDB
ip-10-110-230-197 sourceconfig "MyOraDB" > update
ip-10-110-230-197 sourceconfig "MyOraDB" *> set
credentials.type=CyberarkVaultCredential
ip-10-110-230-197 sourceconfig "MyOraDB" *> set credentials.vault=MyVault
ip-10-110-230-197 sourceconfig "MyOraDB" *> set credentials.queryString="safe-
test;folder=root;object=UnixSSH-delphix_db"
ip-10-110-230-197 sourceconfig "MyOraDB" *> set db_user="Vault-User"
ip-10-110-230-197 sourceconfig "MyOraDB" *> commit
```

The set db_user="Vault-User" is an optional step. If the db_user field is not changed, then it will continue to hold the old value. This value may no longer be correct, or the change to Vault credentials may represent an increase in the customer's security stance, and they may not want their Delphix Admins to know the username.

Setting up Vault via API

The vault API allows users to add, modify, delete, and list vault configurations and retrieving user credentials on a Delphix Engine.

Endpoint - *https://<Delphix Engine IP>/resources/json/delphix/service/passwordVault*

Sample API Request

```
{
    "type": "CyberArkPasswordVault",
    "name": "DemoVault",
    "host": "services-uscentral.skytap.com",
    "port": 17993,
    "applicationId": "Delphix",
    "clientCertificate": {
        "type": "PemClientCertificate",
        "privateKey": "-----BEGIN PRIVATE KEY-----<>-----END PRIVATE KEY-----",
        "clientCertificateChain": {
            "type": "PemCertificateChain",
            "chain": [
                {
                    "type": "PemCertificate",
                    "contents": "-----BEGIN CERTIFICATE-----<>-----END
CERTIFICATE-----"
                }
            ]
        }
    }
}
```

Deleting an existing vault configuration

```
{
    "type": "CyberArkPasswordVault",
    "name": "DemoVault",
    "host": "services-uscentral.skytap.com",
    "port": 17993,
    "applicationId": "Delphix",
    "clientCertificate": {
        "type": "PemClientCertificate",
        "privateKey": "-----BEGIN PRIVATE KEY-----<>-----END PRIVATE KEY-----",
        "clientCertificateChain": {
```

```
            "type": "PemCertificateChain",
            "chain": [
                {
                    "type": "PemCertificate",
                    "contents": "-----BEGIN CERTIFICATE-----<>-----END
CERTIFICATE-----"
                }
            ]
        }
    }
}
```

Adding/Modifying host users

Add an environment with user credentials from CyberArk vault. When adding a host/database user with a vault credential, the name field would be a user identifier and not the actual username. In case this field is empty, a unique identifier is generated with a hash of vault credentials.

```
{
"type": "HostEnvironmentCreateParameters",
"primaryUser": {
"type": "EnvironmentUser",
"credential": {
"type": "VaultCredential",
"vault": "CYBERARK_PASSWORD_VAULT-1",
"vaultCredentialId": "safe=test;folder=root;object=UnixSSH-sybase"
}
},
"hostEnvironment": {
"type": "UnixHostEnvironment",
"name": "bbh-env"
},
"hostParameters": {
"type": "UnixHostCreateParameters",
"host": {
"type": "UnixHost",
"address": "bbdhcp-vault-demo.dlpxdc.co",
"toolkitPath": "/work"
}
}
}
```

Adding/Modifying **database users**

Add dSource using database credentials from CyberArk vault.

The following a sample API link request for MSSQL Domain User.

```
{

    "type": "LinkParameters",
```

```
"name": "ReportServer",
"group": "GROUP-1",
"linkData": {
    "type": "MSSqlLinkData",
    "config": "MSSQL_SINGLE_CONFIG-5",
    "sharedBackupLocations": [],
    "encryptionKey": "",
    "sourceHostUser": "HOST_USER-3",
    "mssqlUser":{
     "password": {
     "type": "VaultCredential",
     "vault": "CYBERARK_PASSWORD_VAULT-2",
     "vaultCredentialId": "safe=test;folder=root;object=Database-MSSql-addtully"
     },
     "type": "MSSqlDomainUser"
    },
    "pptRepository": "MSSQL_INSTANCE-4",
    "pptHostUser": "HOST_USER-3",
    "ingestionStrategy": {
     "validatedSyncMode": "TRANSACTION_LOG",
     "type": "ExternalBackupIngestionStrategy"
    },
    "sourcingPolicy": {
     "logsyncEnabled": false,
     "type": "SourcingPolicy"
    },
```

```
        "syncParameters": {
         "compressionEnabled": false,
         "backupPolicy": "PRIMARY",
            "type": "MSSqlNewCopyOnlyFullBackupSyncParameters"
        }
    }
}
```

Certificate management

The Delphix Dynamic Data Platform uses SSL certificates to secure a number of different communications. Delphix now provides users with the ability to manage their own certificates for HTTPS and DSP (Delphix Session Protocol) connections to and from the Delphix Engine. Users are able to see multiple certificates, their attributes, and deployed status, they can also:

- Replace certificates
- Delete certificates
- Create CSRs (certificate signing requests)

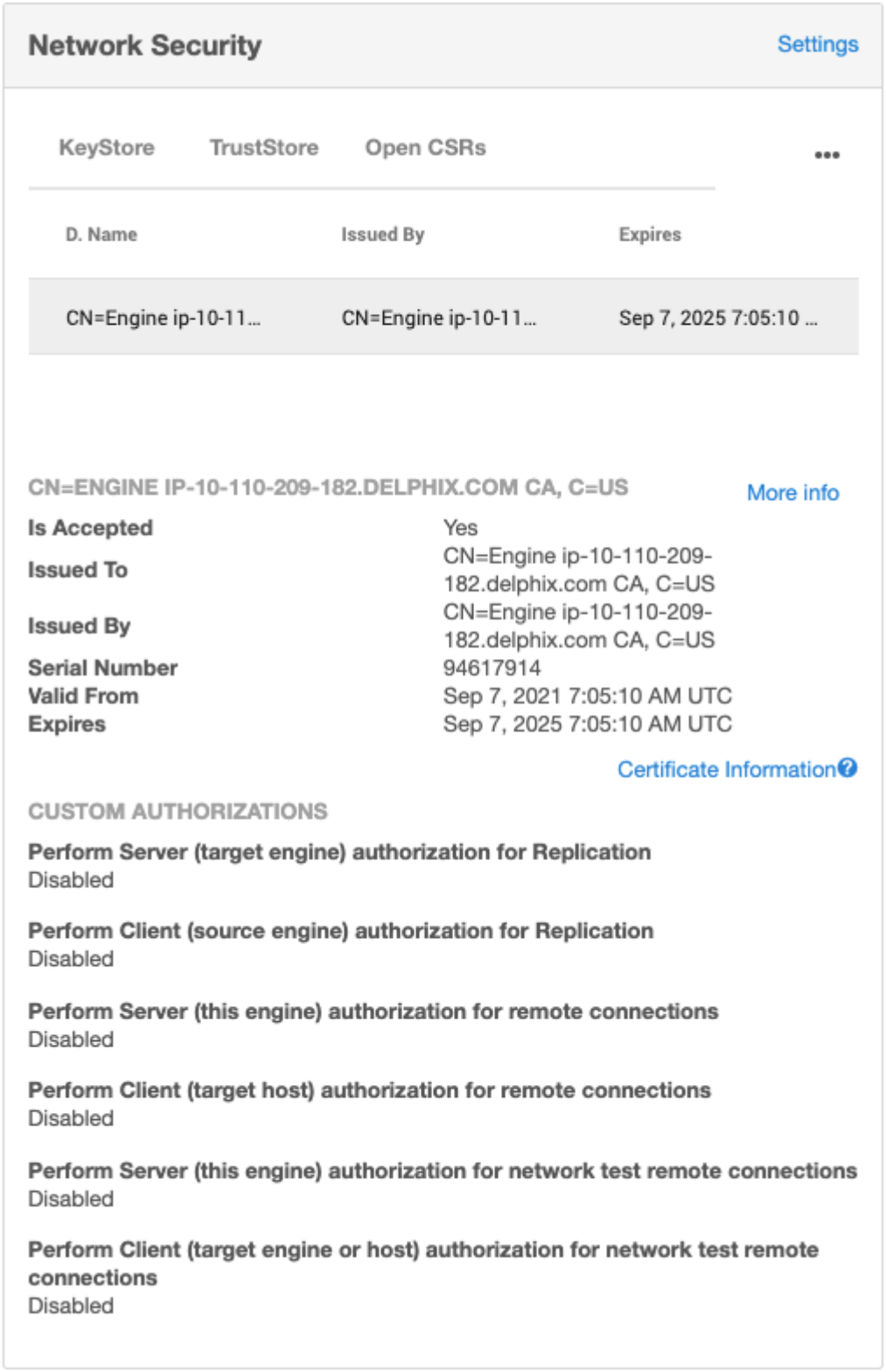
Configuring network security settings

To enable and modify the extent of security settings for HTTPS and DSP (Delphix Session Protocol), click on the “Settings” button in the top right of the “Network Security” panel. This will open a new wizard where you can modify the settings.

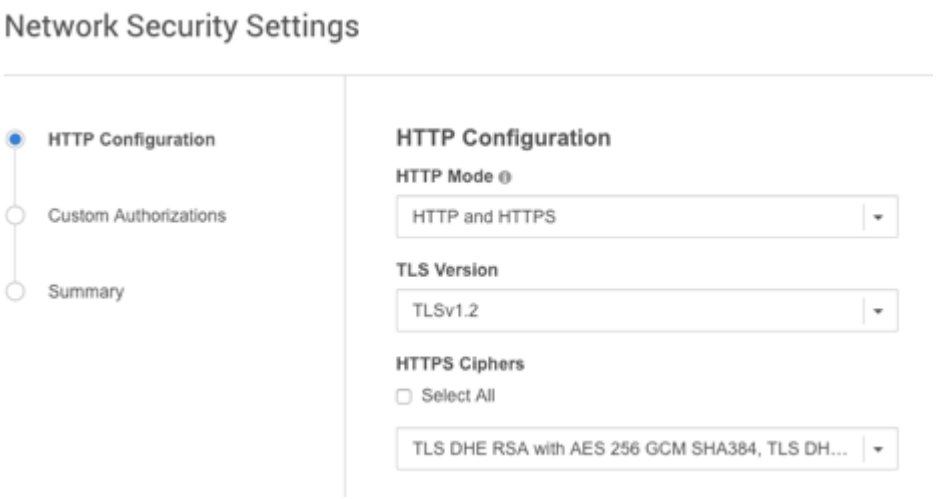
Configuring settings for HTTPS and DSP.

The following procedure guides you through the process of enabling or modifying security settings for HTTPS and DSP.

1. Connect to the Delphix Engine `http://<Delphix Engine>/login/index.html#serverSetup`
2. Click on the **Settings** link in the top right of the **Network Security** panel. This will open a new wizard where you can modify the settings.



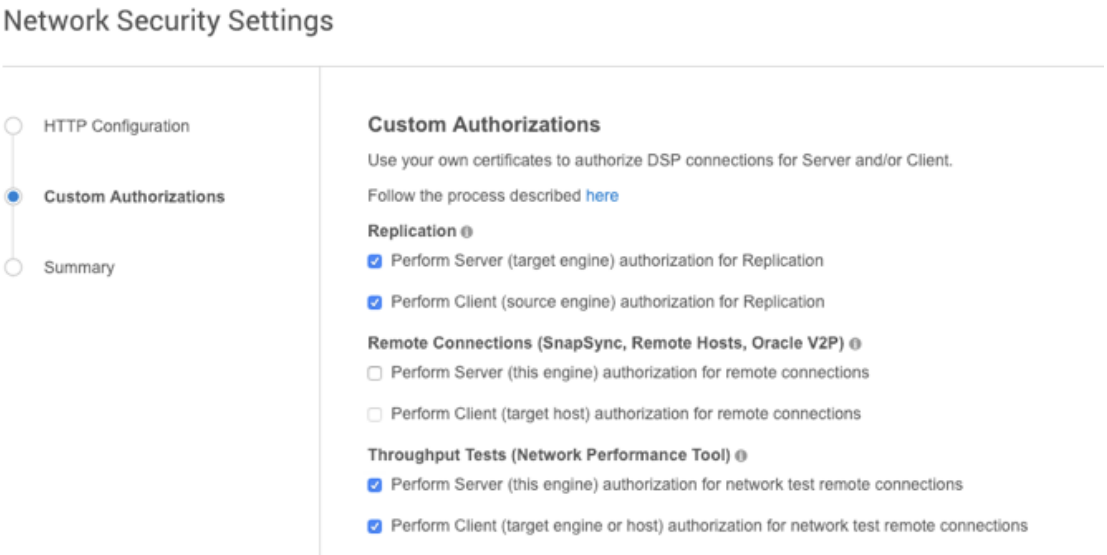
3. In the **Network Settings** wizard, you can modify the **HTTP mode**, **TLS Version**, and **TLS Cipher Suites** to be used.



4. Click **Next**.
5. By default, self-signed certificates are used for server authentication and username/password login acts as client authorization. You can enable the use of custom certificates and modify the extent of how they are used for the following features:
   - Replication
   - Remote host connections (SnapSync, Remote hosts for Target or Server environments, and Oracle V2P)
   - Throughput tests using the Network Performance Tool and DSP

   Info: For all three features, Server authentication must be enabled for Client authentication to take effect.

   Note: Any configuration change requires a stack restart to take effect. Submitting the Network Security Settings in the GUI will automatically trigger a stack restart. All jobs will be stopped, but VDBs will continue to run.



6. Once the HTTP and DSP configurations have been set as desired, click **Next**.
7. You will be presented with a Summary tab. Clicking **Submit** to accept your changes will trigger a stack restart as this is necessary for the configuration changes to take effect. Note: all jobs will be stopped, but VDBs will continue to run.

Certificate management and remote connections

Overview

The server is the Delphix engine and the client is the remote host. This can be used for SnapSync, Oracle V2P (Virtual to Physical), and remote host connections. Once either of these options is enabled, the steps for adding certificate must be done for all environments in the engine.

Enabling server authentication

To enable server authentication, follow the below steps:

1. Replace the desired certificate for DSP (Delphix Session Protocol) in the engine KeyStore. For more details, refer to KeyStore Settings
2. Create a JKS or PKCS#12 keystore on the remote host with the full CA chain of the replaced certificate. Make sure the created keystore has permissions such that it is readable by all environment users configured in Delphix, and enter the keystore details into the host's truststore configuration on the engine. For more details, refer to Host DSP Configuration
3. Select **Perform server (this engine) authorization for remote connections**.

> Altering the authentication settings will require DSP keystore and truststore parameters to be configured for all existing environments, if not the refreshing of existing host environments will fail.

Enabling client authentication

1. DSP connector (for both Windows and Unix hosts)

To enable client authentication using DSP connector, first enable server authentication (refer to the above steps), then follow the below steps:

1. Create a JKS or PKCS#12 keystore on the remote host with the desired key pair. Make sure the created keystore has permissions such that it is readable by all environment users configured in Delphix, then enter the keystore details into the host's keystore configuration on the engine. For more details, refer to Host DSP Configuration
2. Add the full CA chain of the remote host's key pair to the TrustStore on the engine. For more details, refer to TrustStore Settings
3. Select **Perform Client (the target host) authorization for remote connections**.
4. Once the configurations have been set as desired, you will be presented with a summary page. Clicking **Submit** will trigger a stack restart, which is necessary for the configuration changes to take effect. Note: all jobs will be stopped, but VDBs will continue to run.

2. Connector installer connector (specific for Windows hosts)

There are two ways to generate self signed certificates :

a) By Installing the Delphix Connector, which will by default create certificates.

b) By using Self-signed Certificates

To enable client authentication using connector installer, you must perform the below steps for all Windows hosts, which are being added to the Delphix Engine:

1. Execute the below command to generate the PEM file for the Delphix Connector (provided or self-signed)Java KeyStore file. Also, input the store password from the `DelphixConnector.properties` when prompted.

```
keytool -exportcert -alias DelphixConnector-
{UUID_From_DelphixConnector.properties} -keystore "{Installation_Dir}
\connector\DelphixConnector.jks" -rfc -file {Custom_PEM_File_Name}
```

2. Copy the PEM's file content and paste it while adding the certificate into the Delphix Engine.
3. Add the certificate to Delphix engine using the **sysadmin** login and select **Network Security**.
4. Select **Add Certificate** and upload the certificate.
5. Once the certificate is added, enable `validateWindowsConnectorCertificate` from the Delphix engine CLI. This will restart the Delphix engine.

Certificate management and replication

In Replication, the Server is the Target engine and the Client is the Source engine.

Enabling server authentication

To enable Server Authentication, do the following:

1. Replace the desired certificate for DSP (Delphix Session Protocol) in the Target engine KeyStore. For more details, refer to KeyStore Settings
2. Add the full CA chain of the replaced certificate from the Target engine to the TrustStore on the Source engine. The CA chain must match on both engines. For more details, refer to TrustStore Settings
3. Select the option **Perform Server (target engine) authorization for Replication** for both Target and Source engines.

Enabling client authentication

To enable Client Authentication, enable Server Authentication (refer to above steps), then do the following:

1. Replace the desired certificate for DSP in the Source engine KeyStore. For more details, refer to KeyStore Settings
2. Add the full CA chain of the replaced certificate from the Source engine to the TrustStore on the Target engine. The CA chain must match on both engines. For more details, refer to TrustStore Settings
3. Select the option Perform Client (source engine) authorization for Replication for both Target and Source engines.
4. Once the configurations have been set as desired, you will be presented with a summary page.
   Clicking **Submit** will trigger a stack restart as that is necessary for the configuration changes to take effect.
   Note: all jobs will be stopped, but VDBs will continue to run.

Certificate management for throughput tests

Enabling server authentication

To enable Server Authentication, do the following:

1. If using the engine to engine tests, follow the steps for Replication Server Authentication
2. If using the engine to host tests, follow the steps for Remote Connections Server Authentication
3. Select the option **Perform Server (this engine) authorization for remote connections**.

Enabling client authentication

To enable Client Authentication, enable Server Authentication (refer to above steps), then do the following:

1. If using the engine to engine tests, follow the steps for Replication Client Authentication
2. If using the engine to host tests, follow the steps for Remote Connections Client Authentication
3. Select the option **Perform Client (target engine or host) authorization for remote connections.**
4. Once the configurations have been set as desired, you will be presented with a summary page. Clicking **Submit** will trigger a stack restart as that is necessary for the configuration changes to take effect. Note: all jobs will be stopped, but VDBs will continue to run.

Delphix Engine generates alerts when certificates in the Keystore or truststore are expired or about to expire:

- A warning level alert is generated if certificates are expiring in 60 days.
- Critical level alerts are generated if certificates expire in 14 days or have already expired.

Configuring HTTP settings for the Delphix engine

Use the following steps to configure HTTP settings for the Delphix Engine.

1. Login to the Delphix Engine Setup UI as a sysadmin.
2. From the Dashboard under **Network Security** select **Settings**. This will open a new wizard where you can modify the settings.

3. In the **Network Security Settings** screen, you can modify the **HTTP mode**, **TLS Version**, and **HTTPS Ciphers** to be used.
4. In **HTTP mode**, select one of the following:**HTTP Only**: accepts only HTTP connections.
    a. **HTTPS Only**: accepts only HTTPS connections.
    b. **HTTP Redirect**: Redirect all requests made over HTTP to HTTPS.
    c. **HTTP and HTTPS**: accepts both HTTP and HTTPS connections.
    d. **HTTP Redirect with HSTS**: redirect all requests made over HTTP to HTTPS and add Strict-Transport-Security Header to all responses.
5. In **TLS Version**, select the required TLS versions.
6. In **HTTPS Ciphers**, select the required option from the drop-down list. Select the check box next to **Select All** if you want to select all the options.
7. Click **Next** and then click **Submit** to allow your engine to restart for the changes to take effect.
8. When the Delphix Engine homepage is loaded after the restart, all the HTTP requests will be redirected to the selected HTTP mode after the first load.

The above steps might not work if:

1. The certificates are not trusted - ‘not secure’ warning on the search bar of the browser. Certificate setup should be completed before setting up HSTS headers for them to be useful - otherwise, the redirect will still happen over 302 status.
2. Your site is not prepared for HTTPS setup - i.e., each of the subdomains accessed should be supporting HTTPS - parts of the application might be inaccessible otherwise.
3. Every first request whenever the cache is cleared would redirect over 302. Any further requests after the first would redirect over 307.

Don’t want to use HSTS?

1. By default engine is set up to ‘HTTP and HTTPS’. You can leave the engine as it is and not make any changes.
2. If you set up HSTS and want to revert it, the same steps for setup can be followed from above, and select your previous option on step 3.

Regenerating self-signed end-entity and CA certificates

> In many environments, the replacement of HTTPS and/or DSP may be unnecessary.

DSP certificate is only relevant if Custom Authorizations have been configured in Network Security settings, as discussed in the Configuring Network Security Settings article. If these checkboxes are not applied, this means the DSP certificate is not being used.

HTTPS certificate replacement is only necessary if HTTPS connections are used for web browser access.

The following process will leverage Java *keytool* utility. This is commonly available in most Java JDK installations, including those installed in the Delphix Toolkit for Unix, Linux, and Windows Environments under `<toolkit directory> /*host/java/jdk/bin/`. In the following example, `/work` is the toolkit directory. The subdirectory naming conventions from 5.3.x and 6.0.x are illustrated as:

```
$ find /work -name keytool
/work/Delphix_COMMON_f126df603015_33e2f61712c3_2_host/java/jdk/jre/bin/keytool
/work/Delphix_COMMON_f126df603015_33e2f61712c3_2_host/java/jdk/bin/keytool
/work/Delphix_COMMON_564d56b0_26ad_e6ac_f782_d15213207664_oracle_host/java/jdk/bin/
keytool
/work/Delphix_COMMON_564d56b0_26ad_e6ac_f782_d15213207664_oracle_host/java/jdk/jre/
bin/keytool
```

By the end of this process, a PKCS#12 keystore file is generated containing the CA certificate, DSP, and HTTPS certificate. This file will be used for upload twice in the System Setup interface.

Other notes:

- The recommended keystore password 'changeit' is used.
- For `<domain>`, replace this string with the Engine FQDN in every command. This is used as the CN (Common Name). For instance, "**-dname 'CN=Engine <domain> ca,="" c="">>**" would be replaced with `-dname 'CN=Engine example.delphix.com CA, C=US'`.
- The certificate aliases to be used are 'tomcat' for HTTPS and 'dsp' for DSP.
- The existing Delphix CA certificate in the truststore cannot be removed. Faults related to this certificate should be ignored.

Linux Version

1. Generate a new Delphix CA Certificate.

```
export PASSWORD_ENV='changeit'
keytool -genkeypair -noprompt  -alias delphixca  -keyalg RSA -keysize 2048
 -validity 397
-ext 1.3.6.1.5.5.7.3.1 -ext bc=ca:true -ext ku=kCS,cRLS -sigalg SHA256withRSA
-storepass:env
 PASSWORD_ENV  -storetype pkcs12 -startdate -10000M  -dname 'CN=Engine <domain>
CA, C=US' -keypass:env
PASSWORD_ENV -storetype pkcs12 -keystore keystore
```

2. Generate the HTTPS/TLS certificate

```
keytool -genkeypair -alias tomcat -keyalg RSA -keysize 2048 -validity 397 -ext
1.3.6.1.5.5.7.3.1
-ext san=dns:<domain> -ext bc=ca:false  -sigalg SHA256withRSA -storetype pkcs12
-storepass:env
PASSWORD_ENV -startdate -10080M -dname 'CN=<domain>, C=US' -keypass:env
PASSWORD_ENV -keystore keystore
```

3. Generate the DSP certificate

```
keytool -genkeypair -alias dsp -keyalg RSA -keysize 2048 -validity 397 -ext 1.3.6
.1.5.5.7.3.1
-ext san=dns:<domain> -ext bc=ca:false  -sigalg SHA256withRSA -storetype pkcs12
-storepass:env
PASSWORD_ENV -startdate -10080M -dname 'CN=<domain>, C=US' -keypass:env
PASSWORD_ENV -keystore keystore
```

4. Sign the HTTP/TLS certificate

```
keytool -certreq -alias tomcat -keyalg RSA -sigalg SHA256withRSA -storetype
pkcs12 -keypass:env PASSWORD_ENV
-storepass:env PASSWORD_ENV -keystore keystore -file tomcat.csr
keytool -gencert -alias delphixca -ext 'san=dns:'
-validity 397 -sigalg SHA256withRSA -storetype
pkcs12 -storepass:env PASSWORD_ENV -keystore keystore -startdate -10080M
-infile tomcat.csr -outfile tomcat.p12
keytool -importcert -alias tomcat -storetype
pkcs12 -storepass:env PASSWORD_ENV -keystore keystore -file tomcat.p12
```

5. Sign the DSP Certificate

```
keytool -certreq -alias dsp -keyalg RSA -sigalg SHA256withRSA -storetype pkcs12
-keypass:env PASSWORD_ENV
-storepass:env PASSWORD_ENV -keystore keystore -file dsp.csr
keytool -gencert -alias delphixca -ext 'san=dns:'
-validity 397 -sigalg SHA256withRSA -storetype pkcs12 -storepass:env
PASSWORD_ENV
-keystore keystore -startdate -10080M -infile dsp.csr -outfile dsp.p12
keytool -importcert -alias dsp -storetype pkcs12
-storepass:env PASSWORD_ENV -keystore keystore -file dsp.p12
```

Windows version

1. Generate a new Delphix CA Certificate.

```
$ENV:PASSWORD_ENV='changeit'
.\keytool -genkeypair -noprompt  -alias delphixca
-keyalg RSA -keysize 2048 -validity 397  -ext 1.3.6.1.5.5.7.3.1
```

```
-ext bc=ca:true -ext ku=kCS,cRLS -sigalg SHA256withRSA -storepass:env
PASSWORD_ENV  -storetype pkcs12
-startdate -10000M  -dname 'CN=Engine <domain> CA,
C=US' -keypass:env PASSWORD_ENV -storetype pkcs12 -keystore keystore
```

2. Generate the HTTPS/TLS certificate

```
.\keytool -genkeypair -alias tomcat -keyalg RSA -keysize 2048 -validity 397
 -ext 1.3.6.1.5.5.7.3.1 -ext san=dns:<domain>
 -ext bc=ca:false  -sigalg SHA256withRSA -storetype
 pkcs12 -storepass:env PASSWORD_ENV -startdate -10080M
-dname 'CN=<domain>, C=US' -keypass:env PASSWORD_ENV -keystore keystore
```

3. Generate the DSP certificate

```
.\keytool -genkeypair -alias dsp -keyalg RSA -keysize 2048
 -validity 397 -ext 1.3.6.1.5.5.7.3.1 -ext san=dns:<domain>
 -ext bc=ca:false  -sigalg SHA256withRSA -storetype
 pkcs12 -storepass:env PASSWORD_ENV -startdate -10080M -dname 'CN=<domain>,
C=US' -keypass:env PASSWORD_ENV -keystore keystore
```

4. Sign the HTTP/TLS certificate

```
.\keytool -certreq -alias tomcat -keyalg RSA -sigalg SHA256withRSA -storetype
pkcs12 -keypass:env PASSWORD_ENV
-storepass:env PASSWORD_ENV -keystore keystore -file tomcat.csr
.\keytool -gencert -alias delphixca -ext 'san=dns:<domain>'
-validity 397 -sigalg SHA256withRSA -storetype pkcs12 -storepass:env
PASSWORD_ENV -keystore keystore
-startdate -10080M -infile tomcat.csr -outfile tomcat.p12
.\keytool -importcert -alias tomcat -storetype pkcs12
-storepass:env PASSWORD_ENV -keystore keystore -file tomcat.p12
```

5. Sign the DSP Certificate

```
.\keytool -certreq -alias dsp -keyalg RSA -sigalg SHA256withRSA -storetype
pkcs12 -keypass:env PASSWORD_ENV
-storepass:env PASSWORD_ENV -keystore keystore -file dsp.csr
.\keytool -gencert -alias delphixca -ext 'san=dns:<domain>'
-validity 397 -sigalg SHA256withRSA -storetype pkcs12 -storepass:env
PASSWORD_ENV -keystore keystore
-startdate -10080M -infile dsp.csr -outfile dsp.p12
.\keytool -importcert -alias dsp -storetype pkcs12
-storepass:env PASSWORD_ENV -keystore keystore -file dsp.p12
```

At this point, the certificates can be installed by following the Customer Provided Key Pair method described in the Customer Provided Key Pair Configuration article.

The same keystore generated will be uploaded twice, once using alias 'dsp' and once using alias 'tomcat', to replace DSP and HTTPS certificates, respectively.

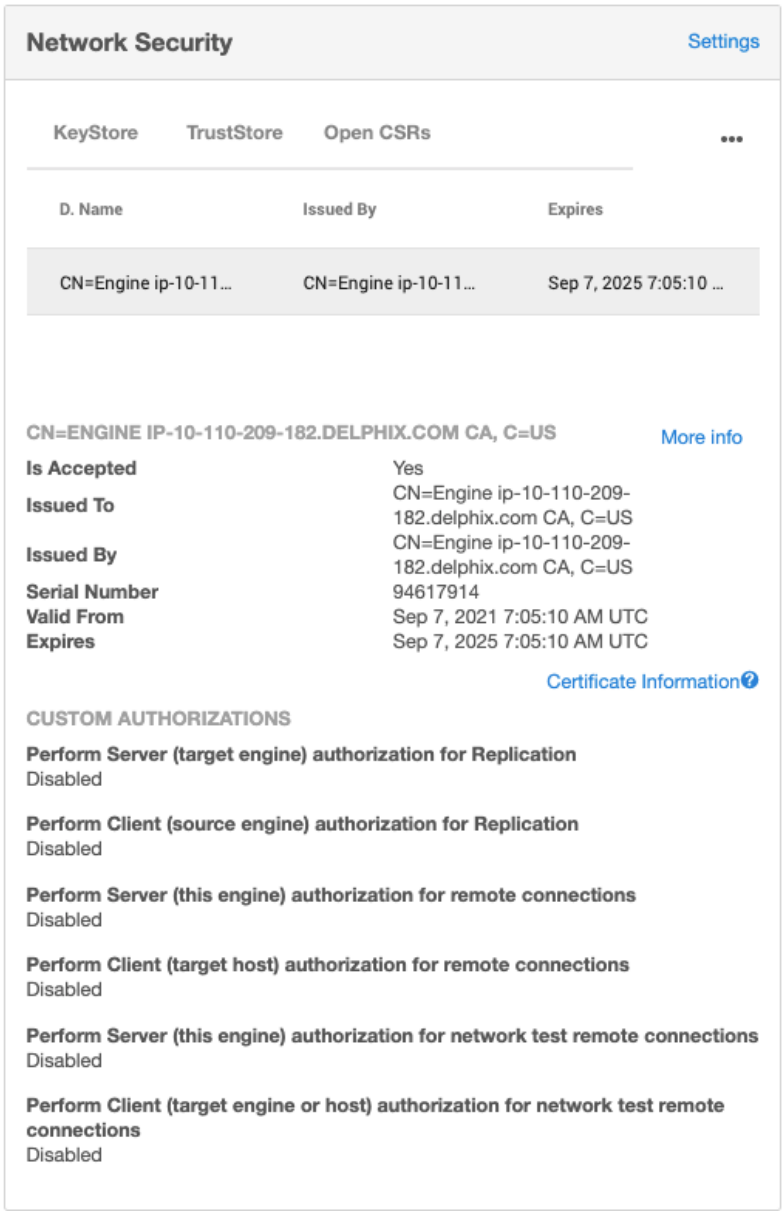
If this error occurs, **Failed to read file with error "Invalid keystore format”**, ensure the **Upload certificate from a PKCS#12 keystore** radio button is selected.

Windows keytool distinctions

- Given the Delphix Connector installation directory **C:\Program Files\Delphix\DelphixConnector**, the keytool.exe executable can be found at **C:\Program Files\Delphix\DelphixConnector\jre\bin\keytool.exe**.
- Powershell set environment variable with: `$ENV:PASSWORD_ENV='changeit'`
- Similar to the comment above with path assumption, on Windows, change ./keytool to .\keytool.exe if located in the jre\bin subdirectory.

Configuring network settings

This topic explains how to replace the HTTPS (HTTP Secure) certificate and the DSP (Delphix Session Protocol) certificate used by the Continuous Data Engine. The KeyStore tab in the Network Security panel displays the keys that the Delphix Engine uses for its identity in HTTPS and DSP communication. For a selected service, the tab will show the full certificate chain starting from the end-entity certificate and ending with the root CA.



There are two methods of replacing a certificate. The key difference between the two is whether Delphix or the user is providing the key pair (public and private key).

For more information on how to configure these see:

- Delphix provided key pair configuration
- User-provided key pair configuration

Customer provided key pair configuration

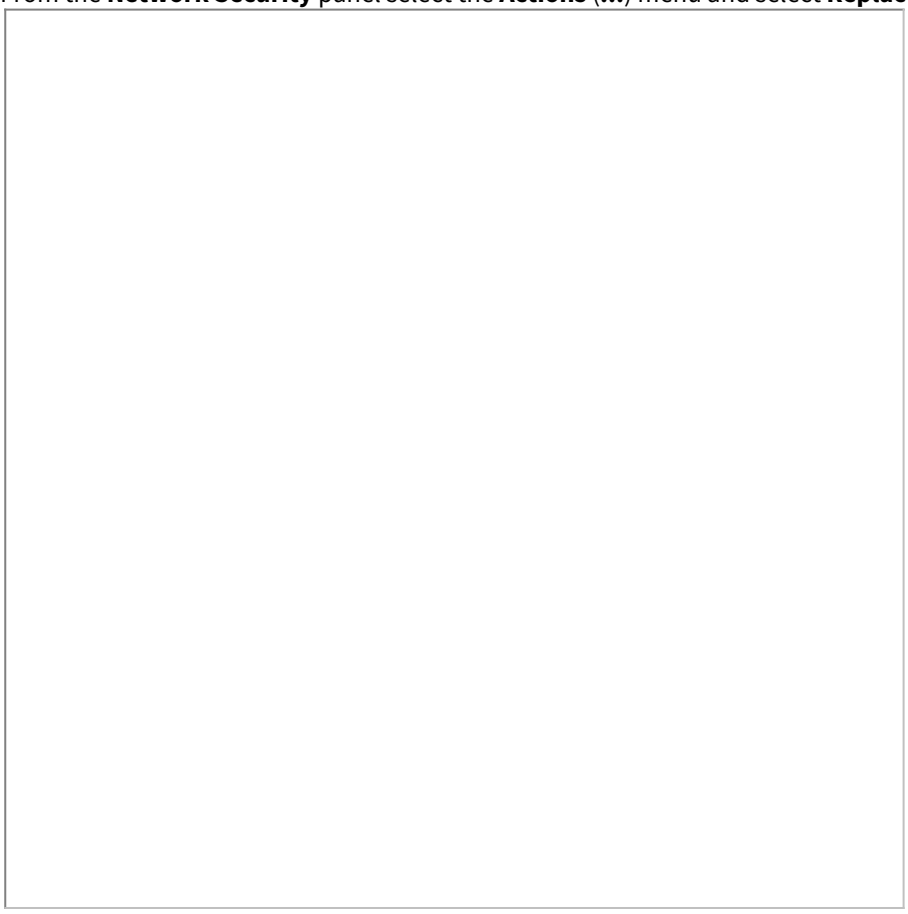
This section describes the steps to take if you are replacing the HTTPS or DSP (Delphix Session Protocol) with your own key pair and certificate. To start, you need to add the key pair and full certificate chain as an entry in a file in JKS or PKCS #12 format.

If replacing the key pair for the DSP application, note that when the DSP Server or Client authenticates each incoming DSP connection (if enabled), it will validate that each certificate of the incoming connection's identity chain has a "Valid From (or Not Before)" date that is after its own time.

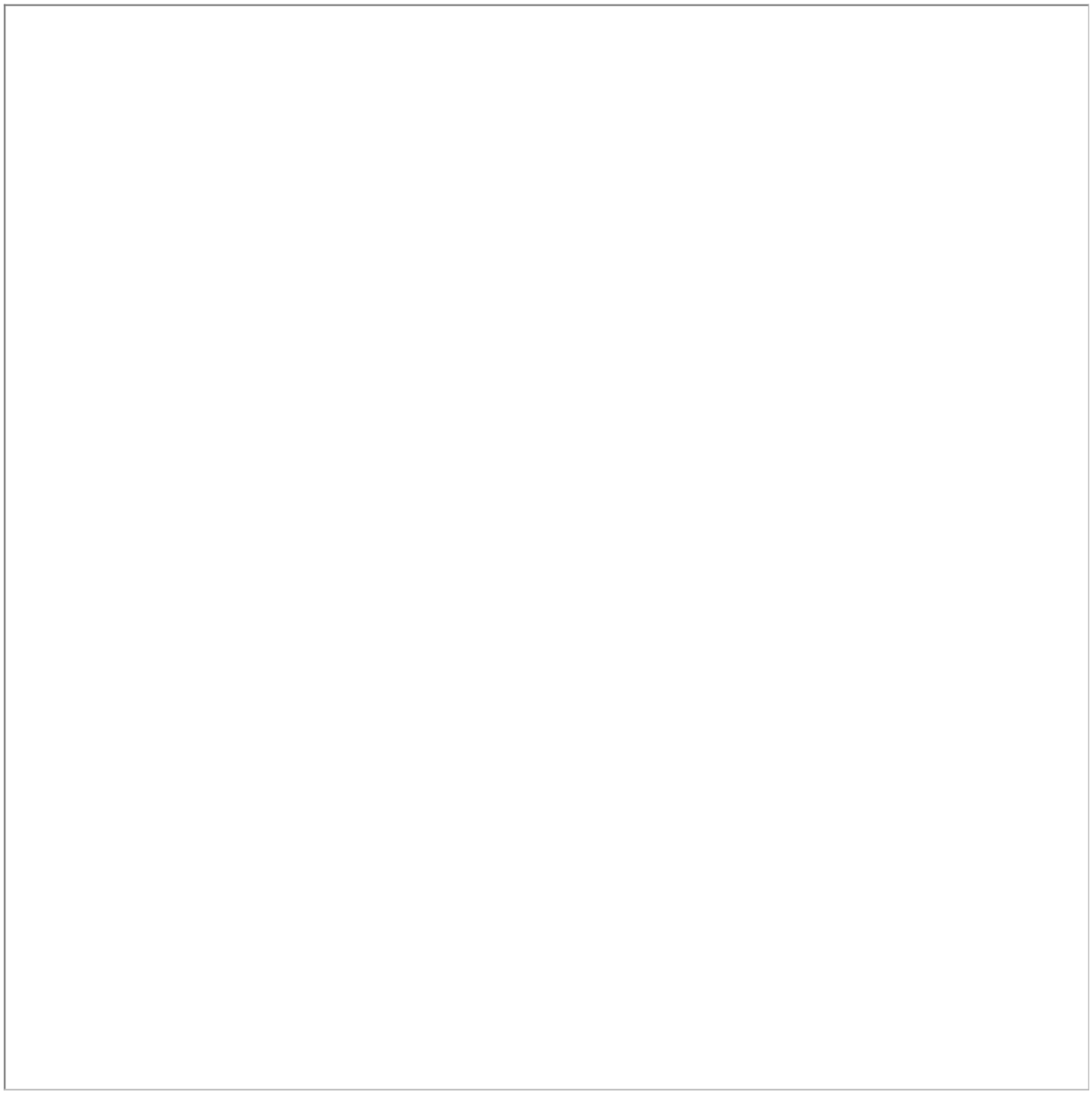
Thus, if your Delphix Engine or host environments are running off of an incorrect (slow) time configuration, then your DSP connections will not work until the offending engine or host's time advances past all incoming certificate's "Valid From (or Not Before)" time.

If correcting the Delphix Engine's or host environment's time configuration may cause issues, then you can workaround this issue by creating and using a certificate with a "Valid From (or Not Before)" date which is before your slowest Engine or host.

1. From the **Network Security** panel select the **Actions** (**...**) menu and select **Replace Certificate**.



2. In the **Replace Certificate** window, select **Upload certificate as files**.



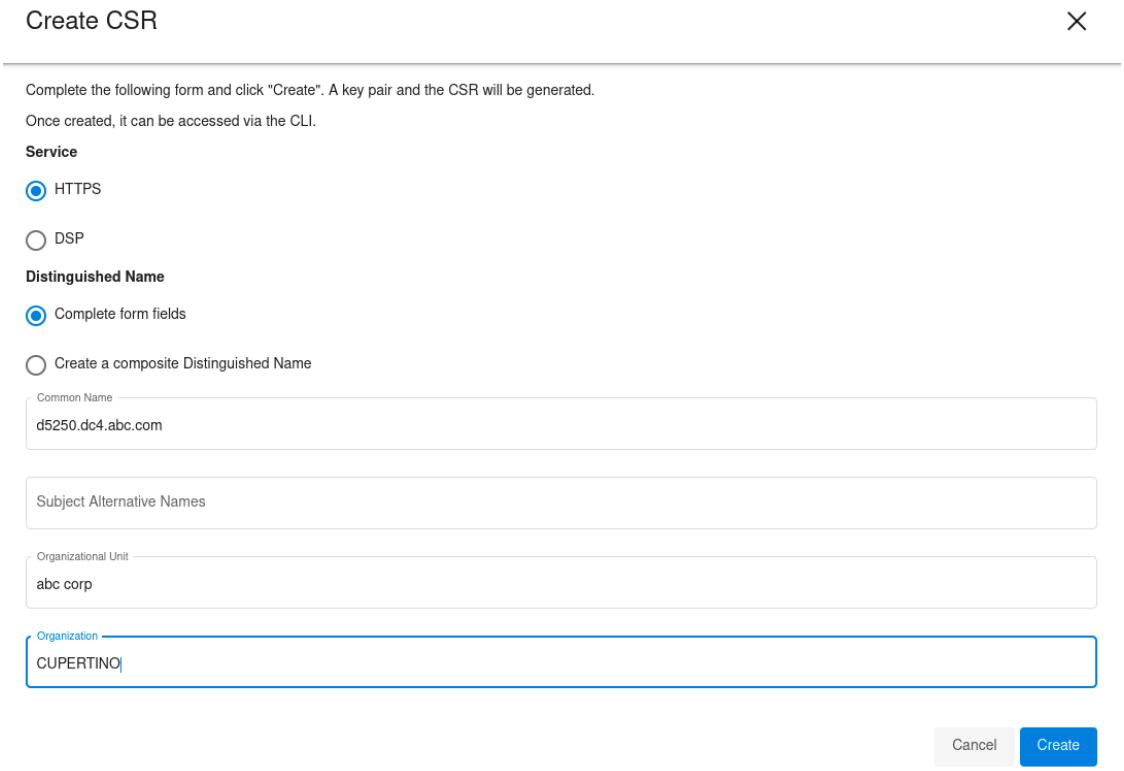
   a. The **Certificate Alias** field is where the key pair and certificate is saved in your JKS or PKCS #12 store.
   b. The **Keystore Passcode** field is the keystore's password.
   c. The **Key Pair Passcode** field is the password for the given alias' key. If not set, it uses the keystore's password.
3. Click **Next** to view a summary. Then click **Submit**.

Delphix provided key pair configuration

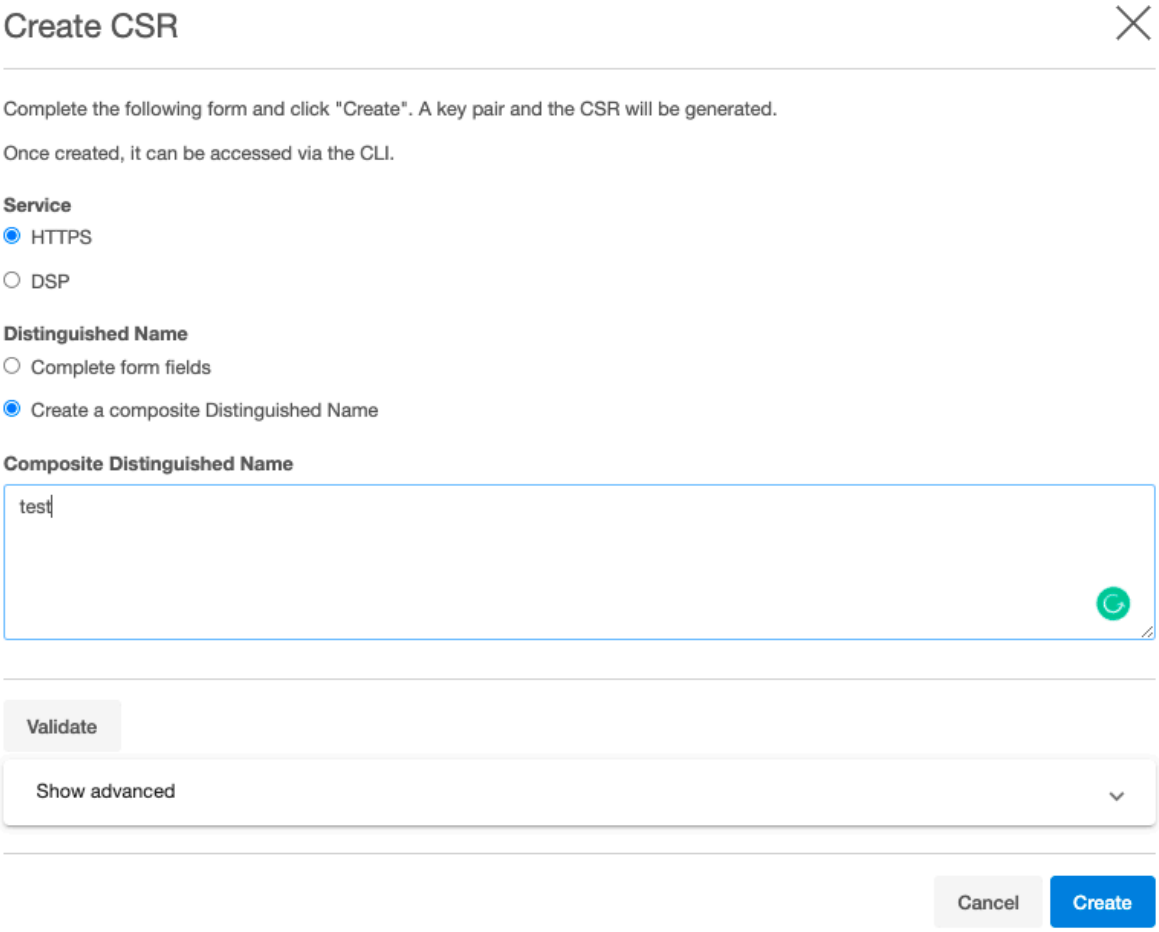
Creating CSR

Use the following instructions to provide an HTTPS or DSP (Delphix Session Protocol) certificate chain for a key pair created by the Delphix Engine.

1. Connect to the Delphix Engine HTTP://< your engine>/login/index/html#serverSetup.
2. From the **Network Security** panel select the **Actions** (**...**) menu and select **Create CSR**.
3. In the **Create CSR window** provide options for your CSR (certificate signing request). Fill up the following fields:
    a. The **Common Name** (CN) is required. If the CN is a valid hostname, then the **Subject Alternative Name** (SAN) can be left blank, otherwise a SAN should be provided. When the CN is a wildcard Common Name (e.g. *.abc.com), the SAN must be populated with every possible domain name covered by the certificate as a comma delimited list. A valid SAN is required for most browDsers to accept the certificate.



    b. Select the service you want to create a CSR for (HTTPS or DSP).
    c. Select how to enter the distinguished name (using the provided fields, or as a composite string).
4. The Validate button will verify that your provided distinguished name follows a legal format.



5. [Optional] select the **Show advanced link** to provide extra options.
    a. **Force Replace**: By default, this is false and means Delphix will not replace the active key pair and certificate with the newly generated key pair and self-signed certificate. If you want to replace the active key pair right away before the signed certificate has been created this can be set to true.
    b. **Keypair Algorithm**: Choose whether the created keypair will use the RSA or ECDSA algorithm. Once an algorithm has been chosen the user can customize the key size and the signature algorithm used.
        i. **Key size:** The valid sizes for RSA are between 2048 and 4096 inclusive. The valid sizes for ECDSA are between 256 and 571 inclusive.
        ii. **Signature algorithm:** The available signature algorithms for RSA are "SHA256withRSA", "SHA384withRSA", "SHA512withRSA". The available signature algorithms for ECDSA are "SHA256withECDSA", "SHA384withECDSA", "SHA512withECDSA".
6. Click **Create**. Once submitted, the CSR is created, and will show up in the **Open CSRs** tab. You can select the CSR and click on View PEM to obtain the CSR in a PEM format.

If creating a CSR for the DSP application, note that when the DSP Server or Client authenticates each incoming DSP connection (if enabled), it will validate that each certificate of the incoming connection's identity chain has a "Valid From (or Not Before)" date that is after its own time.

Thus, if your Delphix Engine or host environments are running off of an incorrect (slow) time configuration, then your DSP connections will not work until the offending engine or host's time advances past all incoming certificate's "Valid From (or Not Before)" time.

If correcting the Delphix Engine's or host environment's time configuration may cause issues, then you can workaround this issue by signing the CSR with a "Valid From (or Not Before)" date which is before your slowest Engine or host.

Replacing a certificate

After the CSR has been signed and turned into an X.509 Certificate, you can replace the certificate using the Replace Certificate option

1. From the **Network Security** panel select the **Actions** (**...**) menu and select **Replace Certificate**.
2. In the Replace Certificate window, select PEM file contents. You can then paste the PEM response in the text box.

⚠ The PEM contents must contain a list of the entire trust chain from the newly generated end-entity certificate to the root CA. You can do this by just pasting each certificate back to back in the text box as long as they are separated by the begin and end cert tags. The order that the PEM certificates are added to the list does not matter

3. Click **Next**.

4 .The **Summary** tab provides a description of your selections. Click **Submit**.

Host DSP configuration

In Delphix Engine to remote host DSP (Delphix Session Protocol) communication, the Server is the engine and the Client is the host. The Add Environment wizard allows you to tell the Delphix Engine how the KeyStores and TrustStores have been set up on the remote hosts.

When the Server or Client authenticates each incoming DSP connection (if enabled), it will validate that each certificate of the incoming connection's identity chain has a "Valid From (or Not Before)" date that is after its own time.

Thus, if your Delphix Engine or host environments are running off of an incorrect (slow) time configuration, then your DSP connections will not work until the offending engine or host's time advances past all incoming certificate's "Valid From (or Not Before)" time.

If correcting the Delphix Engine's or host environment's time configuration may cause issues, then you can workaround this issue by creating and using certificates with a "Valid From (or Not Before)" date which is before your slowest Engine or host.

Adding a new single instance environment after DSP configuration changes

- If server authentication for remote host communication or engine to host throughput tests is desired, make sure the appropriate config is set. For more details refer to Configuring Network Security Settings. You will need to create a JKS or PKCS#12 keystore on the remote host with the full CA chain of the DSP key in the keystore. By default, the key will just the signed by the Delphix CA, but you can replace the DSP key if you wish. Refer to KeyStore Settings for more details.
- If client authentication for remote host communication or engine to host throughput tests is also desired, make sure the appropriate config is set. For more details refer to Configuring Network Security Settings You will need to create another JKS or PKCS#12 keystore on the remote host with the desired key pair. Make sure the created keystore has permissions such that it is readable by all environment users. Then, add the full CA chain of the remote host's key pair to the TrustStore on the engine. For more details, refer to TrustStore Settings

Once the appropriate toggles are enabled, and the remote host is all set up, you can now add the environment as shown below. If only server authentication was desired, only the TrustStore fields need to be filled in. If client authentication was also desired, then the KeyStore fields will also need to be filled in

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Click the **Actions (...)** menu next to Environments and select **Add Environment**.
5. In the **Host and Server** tab, select **Unix/Linux**.
6. Select **Standalone Host** or **Oracle Cluster**, depending on the type of environment you are adding.
7. Click **Next.**
8. In the Environment Settings tab enter your DSP configurations.
9. Select **Submit.**

Modifying an existing single instance environment after DSP configuration changes

1. If an environment already exists after enabling server/client DSP authentication, you will need to modify its attributes for host communication to continue working. As detailed in the above section Adding a new single instance environment after DSP configuration changes, you will need to set up the appropriate stores on the remote host.
2. Once this is done, the **Details** page of your environment will show the existing DSP attributes.
3. Click the **pencil** icon in the top right corner to edit the DSP KeyStore and TrustStore fields accordingly.

4. Edit the DSP fields as required.
5. Once you are done select the checkmark. 

> This will need to be done for all existing environments after DSP server/client authentication is enabled otherwise many host communication features will not work.

Adding a new Unix/Linux cluster environment after DSP configuration changes

Adding a new cluster environment is similar to adding a single instance environment, see Adding a new single instance environment after DSP configuration changes. The steps for setting up the TrustStore and/or KeyStore will need to be done on ALL nodes of the cluster. For a new cluster, each node must also be set up to have the exact same path, password, an alias, because the Delphix Engine will use the same configuration for every auto discovered node. If desired, the path, password, and alias configuration can be changed for each node, but only AFTER the cluster has been added. See Modifying an existing Unix/Linux cluster environment after DSP configuration changes for more details.

Modifying an existing Unix/Linux cluster environment after DSP configuration changes

Similar to single instance environments, an existing cluster can be modified if DSP server/client authentication is enabled. In the cluster environment's Details page, each node can be selected and modified individually. You can also use this to change the path, password, and/or alias to be different across nodes for a cluster that was just added.

DSP for windows clusters

> This will need to be done for all existing environments after DSP server/client authentication is enabled otherwise many host communication features will not work.

Before adding a Windows cluster, each node must have already been added as a single instance. See Adding a new single instance environment after DSP configuration changes if adding a new Windows cluster. If modifying a Windows cluster after a DSP configuration change, you can modify each Windows node on its environment details page See Modifying an existing single instance for more details, or just use the cluster environment details which will allow you to select which node to modify. See Modifying an existing Unix/Linux cluster environment after DSP configuration changes.

KeyStore settings

The KeyStore tab in the Network Security panel displays the public key certificates that the Delphix Engine uses for its identity in HTTPS and DSP (Delphix Session Protocol) communication. For a selected service, the tab will show the full certificate chain starting from the end-entity certificate and ending with the root CA.+

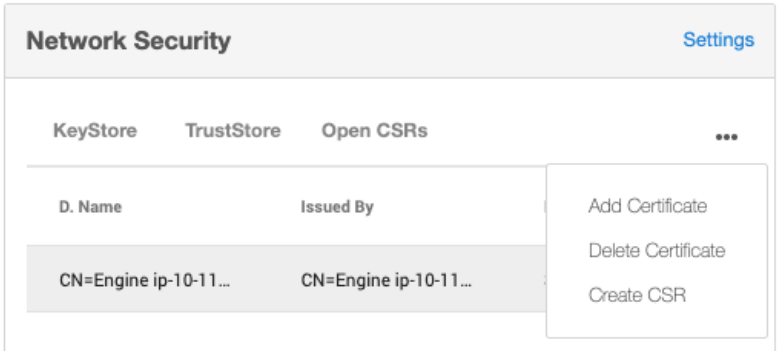
There are two methods of replacing a certificate. The key difference between the two is whether Delphix or the user is providing the key pair (public and private key). For more information see Configuring Network Settings.

Delphix Engine generates alerts when certificates in the Keystore or truststore are expired or about to expire:

- A warning level alert is generated if certificates are expiring in 60 days.
- Critical level alerts are generated if certificates expire in 14 days or have already expired.

TrustStore settings

The TrustStore tab in the Network Security panel displays all the CA certificates that the Delphix Engine trusts. Click on the Actions (...) menu in the top right to reveal the options for editing the TrustStore contents:



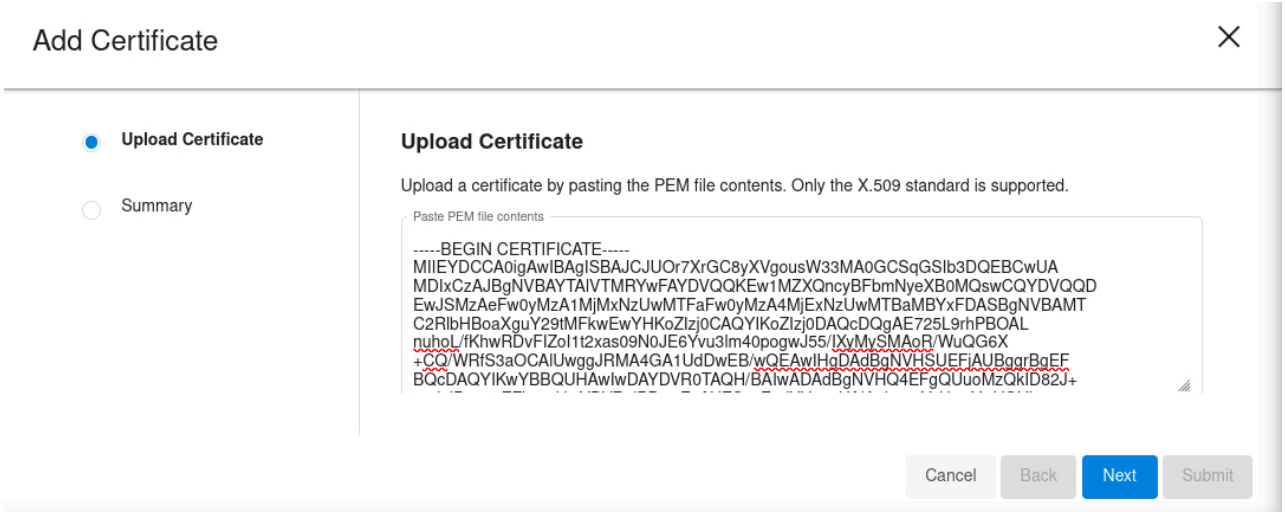
Delphix Engine generates alerts when certificates in the Keystore or truststore are expired or about to expire:

- A warning level alert is generated if certificates are expiring in 60 days.
- Critical level alerts are generated if certificates expire in 14 days or have already expired.

The truststore is used for a variety of connections from the engine to external hosts, such as secure SMTP, and HTTPS proxy connections.

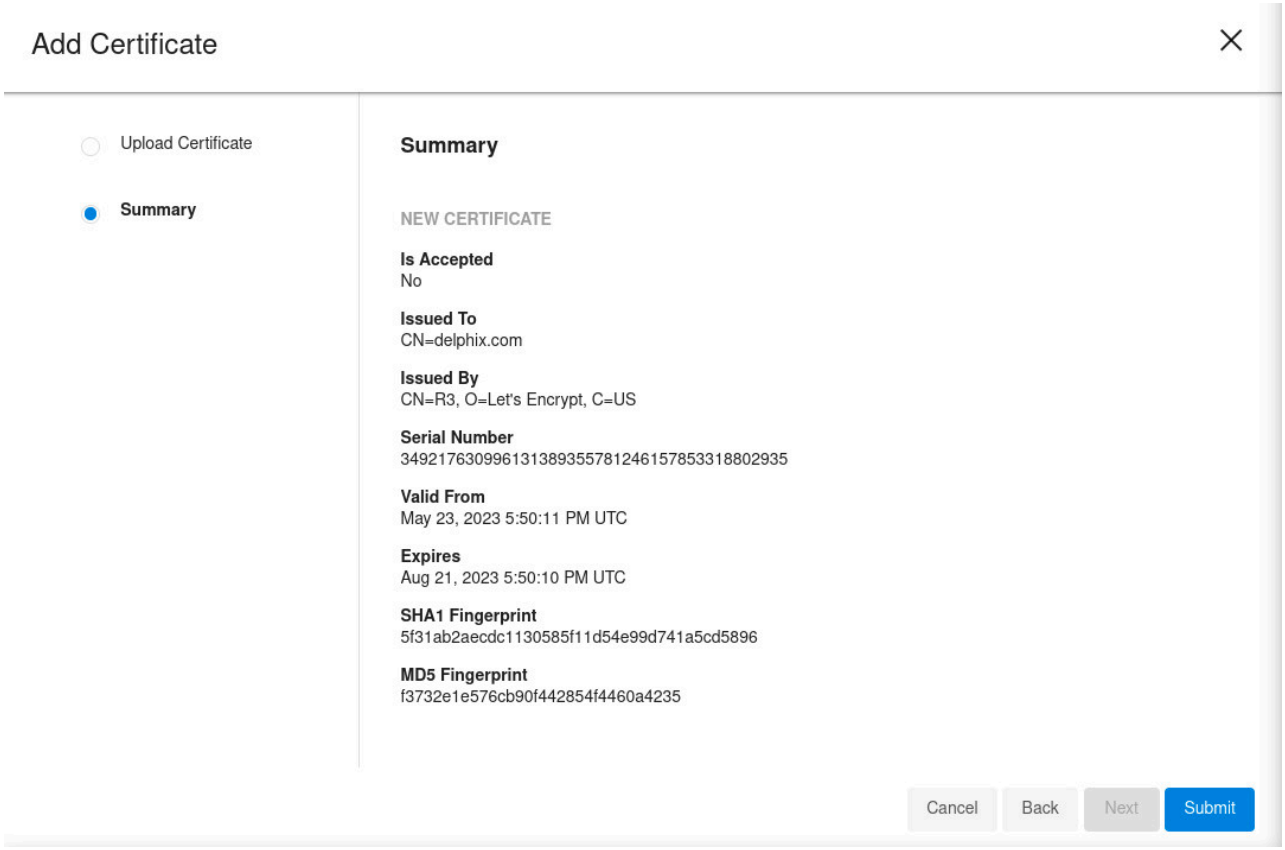
Adding a certificate

1. From the Actions menu select Add Certificate.
2. In the Add Certificate wizard paste the PEM contents of the CA certificate you want to add. The PEM contents must have the appropriate header and footer included.



3. If you are adding a non-root CA certificate, its signer must already exist in the truststore. So, if you are adding a chain with multiple certificates, you must add them individually starting from the root CA. If not, you will get an error saying that we could not establish a chain of trust.

4. Click **Next** to view a Summary tab where you can confirm the certificate contents.



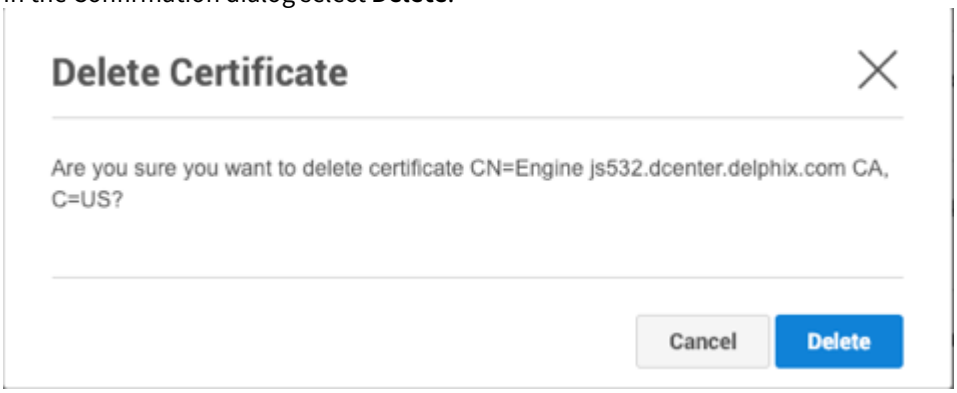
5. Click **Submit**.

Deleting a certificate

Use this option to delete the selected CA certificate.

⚠ Deleting any certificate that breaks an existing chain of trust is not allowed.

1. From the Actions menu select **Delete Certificate** .
2. The default Delphix CA cannot be deleted. Note that deleting any certificate that breaks an existing chain of trust is also not allowed.
3. In the Confirmation dialog select **Delete**.



⚠ Unlike the network security settings, any changes to the TrustStore will not require a stack restart.

Replication security

Choose encrypt and compress when replicating

Delphix provides encryption capabilities when replicating data from one Delphix Engine to another.

Upon selecting the **Encrypted** option under **Traffic Options** the Delphix Engine intelligently compresses before encrypting data. This ordering leads to lower CPU utilization and higher throughput compared to using encryption alone, with the same level of protection. See Configuring Replication for details. The flag can be set at any time. Once set, it results in sending subsequent replication traffic streams as encrypted.

Train your Delphix Administrators to choose both options.

Object security

This section covers the following topics:

- User management
- Source database security
- Source and target host security

User management

Secure user management

Secure user management is best achieved by integration with your centralized authentication service. Once the integration is complete, create LDAP authenticated named users to facilitate separation of duties, least privileges, and auditing. Disable the out-of-the-box generic ADMIN and SYSADMIN accounts.

Use LDAP for authentication

As described under System Configuration above, enable LDAP authentication to leverage your enterprise authentication service and enable SSL/TLS to secure LDAP connections.

Create named users

Do not create generic functional accounts such as “QA,” “DEV,” or “TEST.” Such accounts will not leave a proper audit trail and violate the separation of duties principle. Instead, create LDAP authenticated named users.

Assign least privileges

Restrict the **admin** and **sysadmin** roles to 1-2 trusted named users each. These roles are highly privileged and must be carefully managed. These roles typically map to a **DBA** and **System Administrator** respectively.

For subordinate users who need to refresh VDBs, assign “Data Operator” privilege on the VDB and “Reader” privilege on the dSource.

For subordinate users who need to provision new VDBs from dSources, assign “Provisioner” privilege on the dSource and “Provisioner” privilege on the Group to which they will assign the VDB.

Consider Delphix self-service functionality

The Delphix Self-Service functionality is targeted towards developer and tester self-service, and it contains a more sophisticated privilege model. With this functionality, Delphix Self-Service users do not have access to the Administrator GUI.

Administrators can define multiple data sources as a complete template. They also allocate server resources as a “data container.” The end-user has the ability to update data from the source, from peers using the same source, and from prior images of the source that they have created.

Disable ADMIN and SYSADMIN

Once you have established named Delphix Administrators and Systems Administrators, disable the out-of-the-box `sysadmin` and `admin` accounts. You can disable accounts through the CLI.

When engines created before 5.3.1 are upgraded to 5.3.1 or later they will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then Disabling delphix_admin.

Source database security

Choose minimum privileges for Delphix DB user

The Delphix Engine requires a DB user (**delphix_db)** on your source databases. This account is necessary to detect the state of the source and stay in sync. Do not give unnecessary privileges to this user. Leverage the script provided by Delphix in the hostchecker bundle to create a user with the minimum required privileges. The DB user (e.g., **delphix_db**, which is the example used on this page) account can have the same or different user name on each of your source databases.

Protect the Delphix DB user password

Because the **delphix_db** user has access to sensitive data dictionary information, take steps to protect access to this account.

Use database encryption functionality to encrypt sensitive data at rest

Database vendors provide tools to encrypt data at rest. This encrypts data on disk in order to provide protection at rest. Use the database encryption on your source databases to encrypt sensitive or all data. Delphix integrates seamlessly with database encryption to provide protection at rest.

Choose encrypt and compress when linking

Delphix provides encryption capabilities when linking against your source databases. Encrypting while linking can lead to higher CPU utilization and higher throughput. This overhead can be reduced by selecting both the Compress and Encrypt options, in order to compress the data before it is encrypted. Train your Delphix Administrators to choose both options.

Source and target host security

Oracle on UNIX

Delphix support for Oracle on UNIX requires an OS account (**delphix_os**) on source database servers and on target servers that will host virtual databases or files. The Delphix Engine uses SSH to send commands to this user, which performs operations on the host. Some of these commands require elevated privileges.

> There is no actual requirement that the account be named **delphix_os** on both sources and targets. You can name the account anything you want; you can also use separate accounts on every source and target.

- **Restrict su - delphix_os to named Delphix admins and system administrators**

Other users of the system do not need access to the delphix_os user. Your Delphix Admins and System Administrators should retain su ability to facilitate troubleshooting.

- **Use SSH Key exchange to allow the Delphix engine to communicate with targets**

Implement public/private key exchange instead of username/password. This allows you to keep the password of **delphix_os** completely secret.

- **Put delphix_os on password rotation** Rotate the delphix_os password in accordance with your enterprise security policy for application software accounts. You should either:
  - implement SSH Key exchange prior to placing delphix_os on password rotation, or
  - script CLI commands to update the password inside the engine as part of the rotation process.

Delphix Professional Services can assist you in integrating Delphix with your enterprise password rotation system.

- **Restrict elevated privilege commands to the lowest level needed** The Delphix Engine uses elevated privileges to provide core features as well as optional features. The Delphix docs describe in detail which privileges are absolutely necessary, as well as techniques for further restricting the commands that can be used. The Delphix Engine ships with support for “sudo” as the privilege elevation system, but also allows for integration with third-party and custom centralized privilege management systems.

Windows

Delphix support for SQL Server requires two OS accounts for Windows:

- **delphix_src** – used on the source database server
- **delphix_trgt** – used on the servers which host Virtual Databases Both are required for the Validated Sync target

> There is no actual requirement that the account is named **delphix_src/delphix_trgt**. You can name the account anything you want; you can also use separate accounts on every source and target. Finally, you can create a single account for use everywhere, but this is not recommended since it violates the separation of duties.

Restrict privileged commands to the lowest level needed

The Delphix user or domain account should have exactly the privileges required in the Delphix documentation. Do not grant additional privileges.

Put delphix_src and delphix_trgt on password rotation

Change the user or domain account password at regular intervals or in accordance with security policies for application software accounts. Use CLI scripts to quickly modify the password across the Delphix ecosystem. Delphix Professional Services can assist you in scripting and integrating Delphix with your enterprise password rotation system.

Use minimum privileges on your SMB share

Consult http://technet.microsoft.com/en-us/library/cc754178.aspx to understand how shared folder privileges work. Use the minimum privileges.

Use windows authentication for SQL server

SQL Server allows authentication via Windows or Mixed mode. Mixed mode allows authentication via Windows or SQL Server.

Windows authentication is more secure; it uses Kerberos security protocol, provides password policy enforcement with regard to complexity validation for strong passwords, provides support for account lockout, and supports password expiration. http://msdn.microsoft.com/en-us/library/ms144284.aspx

System configuration

There are a number of configuration options available in the System Administration area that help you to secure the Delphix Engine. You can configure these during installation through the setup wizard, or later by accessing the Delphix Setup screens.

Maintain system time with NTP

Establish at least one, but preferably three, corporate NTP servers and sync your Delphix Engine to them. This ensures that audit and error messages display the correct time.

When configuring Delphix Engine on VMware, be sure to configure the NTP client on the ESX host to use the same servers that you enter here. On a vSphere client, the NTP client is set in the **Security Profile** section of the configuration process.

Enable phone home

Phone Home service will send critical information about the Delphix Engine to Delphix Support using HTTPS, on a periodic basis. The use of a Web Proxy Server is fully supported. Phone Home data allows Delphix Support to proactively detect Delphix Engines affected by critical vulnerabilities.

Register your Delphix engine

Registration is fast and easy, and you can do it with or without Internet connectivity from the Delphix Engine. Failing to register the Delphix Engine will impact its supportability and security in future versions.

Enable LDAP for authentication

The LDAP protocol is used by enterprise authentication services. Enabling LDAP authentication allows your Delphix Engine to leverage the password control features of these products, such as expiration, lockout, and complexity.

Import your LDAP server certificate into your Delphix Engine, and enable SSL/TLS.

Enable SMTP and/or SNMP monitoring

When the Delphix Engine encounters errors, it issues **alerts**. Configure SMTP and/or SNMP to forward **alerts** to your central monitoring system.

GUI security

Overview

The sections in this article cover securing the Delphix GUI, which is similar to securing other web consoles. Some of these solutions include reducing the session timeout threshold, creating a signed certificate, and disabling HTTP access.

Reduce inactive session timeout to 15 minutes

This means that a user will be booted from the session after 15 minutes of inactivity. This is done with a CLI command on a per-user basis by modifying the `sessionTimeout` property of the **User** object, as shown below. The default inactive timeout happens after 30 minutes.

```
myhost.delphix.com>  cd  user
myhost.delphix.com user> select delphix_admin
myhost.delphix.com user 'delphix_admin'> update
myhost.delphix.com user 'delphix_admin' update *> set sessionTimeout=15
myhost.delphix.com user 'delphix_admin' update *> commit
```

Use a URL from your domain and create a signed certificate

Do not use IP Addresses to access the Delphix Engine. Create a hostname and DNS entry, such as “delphix1.mycompany.com”. Delphix Support can assist in converting the engine from a self-signed certificate to a signed certificate that maps to your domain name. Please file a Support ticket to proceed.

Disable HTTP access

Disabling HTTP or configuring HTTP to redirect connections to HTTPS is recommended to protect in-flight user credentials and connections with the engine. This can be done via the command line or through the GUI.

Repave Delphix Engine

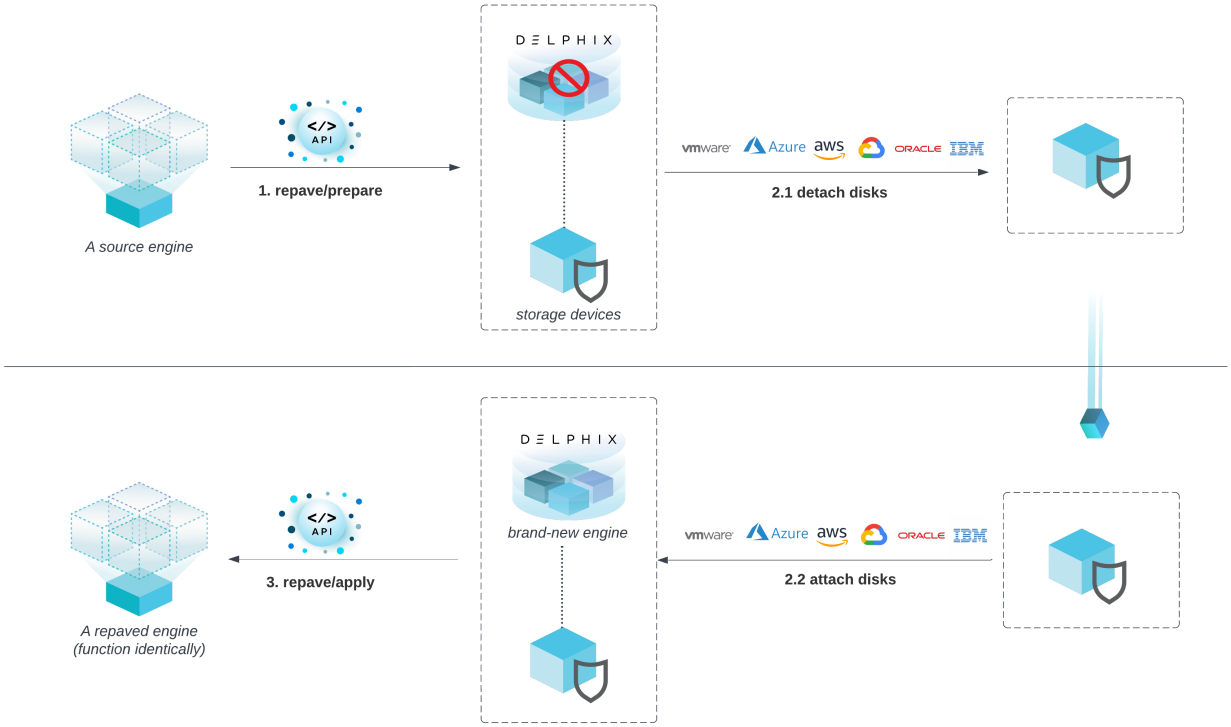
Overview

As the rate of new security vulnerabilities rise, Delphix continues developing solutions to mitigate potential risks. Repave is an added solution applied to a Delphix Engine to get rid of any potentially compromising complications. It transfers datasets and required metadata to a fresh, new target engine, without carrying over any compromised binaries. The target engine will function identically to the source engine.

Prerequisites

- Repave is only supported on the Continuous Data (virtualization) engine with block storage.
- You would need to enable the feature flag `REPAVE` before using it on engine version 13.0. After 14.0, the feature flag is enabled by default.
- The target engine shall be in the same version as the source engine.

How to Repave a Delphix Engine



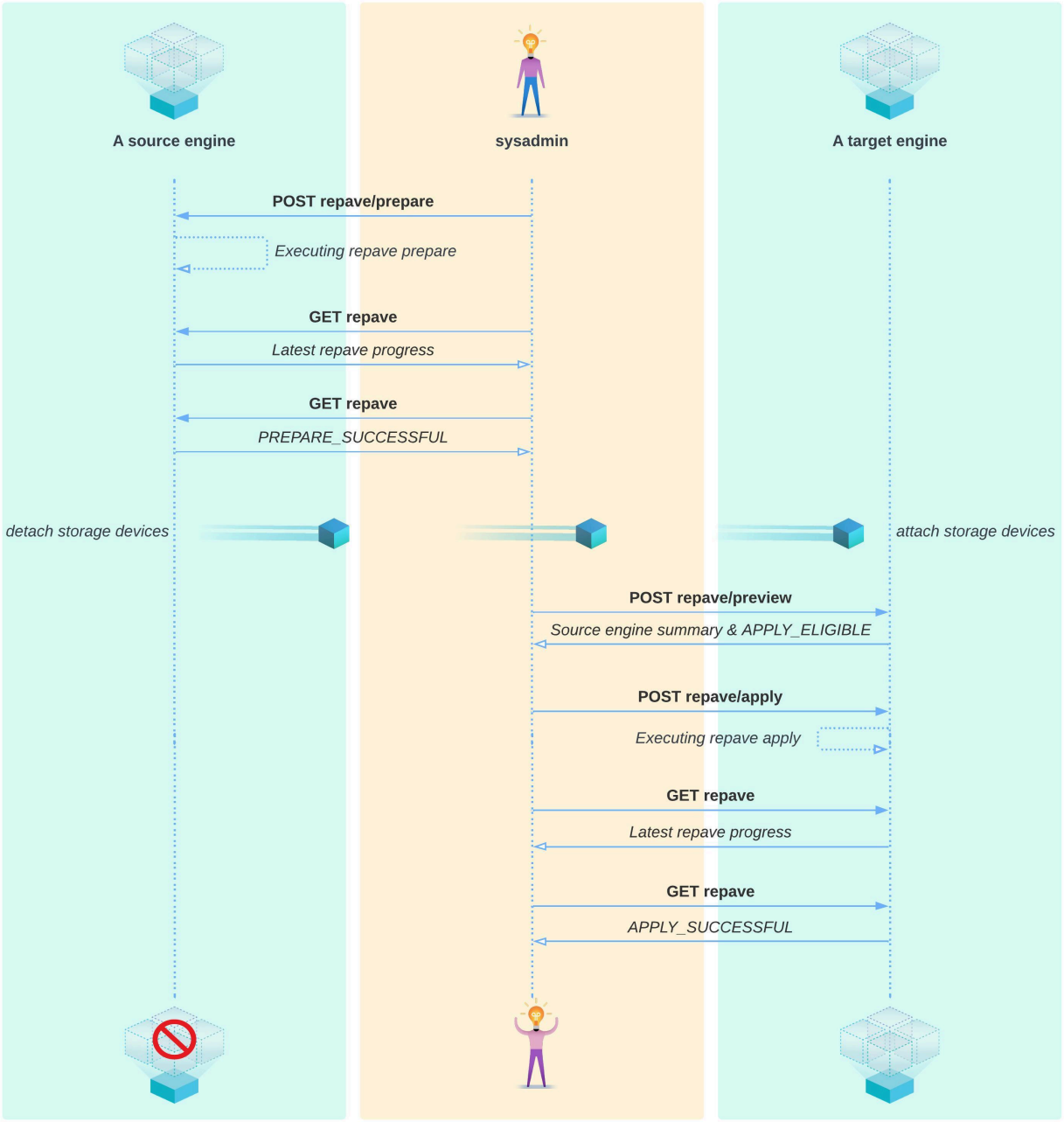
1. A sysadmin initiates **repave/prepare** on the source engine. All Delphix datasets are disabled and the Delphix management service is shut down.
2. A sysadmin detaches storage devices from the Delphix engine and attaches them to a new Delphix engine. This can be done through cloud APIs or manual operation.
3. A sysadmin calls **repave/apply** on the new engine. All previously disabled data sources are enabled again.

Repave API calls

- `POST repave/prepare` initiates the repave preparation on the source engine
- `POST repave/apply` starts the process of applying repave on the target engine.
- `GET repave` returns the repave state, configurable metadata, and engine summary of the current engine.
- `POST repave/preview` provides a preview of the configurable metadata and engine summary of the source engine, as well as determines its eligibility for applying repave on the target engine.

The representation below is a general API workflow of repave.

> You might need to call APIs `Establish Session`, `Login` and `Enable Feature Flag` before calling repave APIs.



Tips and usage guidelines

- `repave/prepare` and `repave/apply` returns a job id. However, when the Delphix management service is down, the job API will not work. The other way to check the repave status is by calling `GET repave`.
- All running sources on the engine will be disabled by repave. However, if an environment was disabled before repave, repave might fail to enable the corresponding dSources and vDBs back. Manually enabling the environment and then enabling the dSources and vDBs should resolve the problem.

- If an engine gets stuck in an ongoing repave state for a very long time (like a few hours), restarting the Delphix management stack can forcibly reset repave to a failed state so users are able to try again.
- If the source engine has a hot fix installed, the target engine shall have the same hotfix installed as well before applying repave.

POST calls

Establish session

`http://{{delphix_engine_url}}/resources/json/delphix/session`

Establish an API session with Delphix, which is **required** before calling repave APIs.

```
{
    "type": "APISession",
    "version": {
        "type": "APIVersion",
        "major": {{api_version_major}},
        "minor": {{api_version_minor}},
        "micro": {{api_version_micro}}
    }
}
```

Login

`http://{{delphix_engine_url}}/resources/json/delphix/login`

Login as `sysadmin`, which is **required** before calling repave APIs.

```
{
   "type": "LoginRequest",
   "username": "{{sysadmin_user}}",
   "password": "{{sysadmin_pwd}}"
}
```

Enable feature flag

`http://{{delphix_engine_url}}/resources/json/delphix/system/enableFeatureFlag`

Enable feature flag `REPAVE`, which is **required** before calling repave APIs.

```
{
   "type": "FeatureFlagParameters",
   "name": "REPAVE"
}
```

Repave prepare

`http://{{delphix_engine_url}}/resources/json/delphix/repave/prepare`

Introduction

Delphix sysadmin calls **repave/prepare** to start preparing for repave on the source engine compromised.

- All the sources will be disabled.
- All datasets and required metadata will be transferred to domain storage devices.
- The Delphix management service will be shut down.

Request parameters

- `ignoreDisableSourcesFailures`
  If `true`, a failure to disable sources will not block the repave. The default is `false`, if not provided.
- `enableSourcesOnFailure`
  If `true`, when repave fails, data source disabled by repave will be enabled again. The default is `false`, if not provided.

Response

A job id and action id will be returned if the API is called successfully.

Please make sure repave state is `PREPARE_SUCCESSFUL` before detaching domain0 storage devices.

Technical details

The Delphix engine will go through four main steps during this phase:

- **Quiesce engine**
  All domain users will be forcibly kicked out. All automatic replication will be turned off. All running jobs will be cancelled or suspended. All policies will be paused. All sources will be disabled. A Repave state `PREPARE_QUIESCE_ENGINE_FAILED` indicates that the engine is failed to quiesce the engine.
- **Clean up environments**
  Windows environments will be cleaned up, this step is mainly to clean up the iSCSI setting of environment hosts. Any failure will be ignored since this step does not block repave, but users might see legacy iSCSI settings on their Windows environment hosts.
- **Extract metadata**
  Metadata that presents the identity of the current engine will be extracted and stored in domain0. A Repave state `PREPARE_EXTRACT_METADATA_FAILED` indicates that the engine has failed to extract metadata.
- **Export Domain0**
  Services like Delphix management stack and Postgres will be stopped before exporting the domain0 pool. The repave state `PREPARE_EXPORT_DOMAIN0_FAILED` indicates that the current engine has failed to stop services or export the domain0 pool.

```
{
    "type": "RepavePrepareParameters",
    "ignoreDisableSourcesFailures":false,
    "enableSourcesOnFailure":false
```

```
}
```

Repave apply

`http://{{delphix_engine_url}}/resources/json/delphix/repave/apply`

Introduction

Delphix sysadmin calls **repave/apply** on the new engine. The Delphix management service will be restarted and all previously disabled data sources will be enabled again.

- Before calling this API, make sure all domain0 pool storage devices have been attached to the target engine correctly.

Request parameters

One type of parameter is supported for version 13.0.

- `BlockStorageRepaveApplyParameters` is for Delphix Continuous Data engines with block storage.

Response

A job id and action id will be returned if the API is called successfully.

Technical Details

The engine will go through six main steps during the this phase. `APPLY_SUCCESSFUL` indicates the engine has applied repave successfully. You can always call `repave/apply` again if there is any failure during repaving.

- **Import Domain0**
  The domain0 pool from the source engine will be imported to the target engine. `APPLY_IMPORT_DOMAIN0_FAILED` indicates the repave apply failed to import domain0. Delphix management stack will be stopped after this step.
- **Check eligibility**
  Check if the target engine is eligible to apply repave. If the target engine version is different from the source engine, or any required hotfix is not installed, or not enough space is in rpool storage, checking eligibility will fail.
- **Setup MDS**
  Repave will shut down the Delphix management stack and call a backend task to set up MDS (MetaData Service) in this step, the MDS was originally from the source engine. It will also rsync `/var/delphix`, snapshot-based metadata and system tunable from domain0 to rpool. `APPLY_SETUP_MDS_FAILED` indicates repave fails in setting up MDS. The Delphix management stack will be restarted after this step.
- **Generate metadata**
  The configurable metadata stored in domain0 will be generated on the target engine so the target engine will behave identically as the source engine. `APPLY_GENERATE_METADATA_FAILED` indicates repave fails to generate metadata.
- **Refresh environments**
  All environments will be refreshed in parallel during this step. New iSCSI settings will be set up again on Windows environment hosts. `APPLY_REFRESH_ENV_FAILED` indicates repave fails to refresh environments.
- **Unquiesce engine**
  In this step, all sources disabled by repave will be enabled again, all scheduled replication jobs will be

resumed and policy execution will be resumed. `APPLY_UNQUIESCE_ENGINE_FAILED` indicates repave fails to unquiesce engine.

```
{
    "type": "BlockStorageRepaveApplyParameters"
}
```

Repave status

`http://{{delphix_engine_url}}/resources/json/delphix/repave`

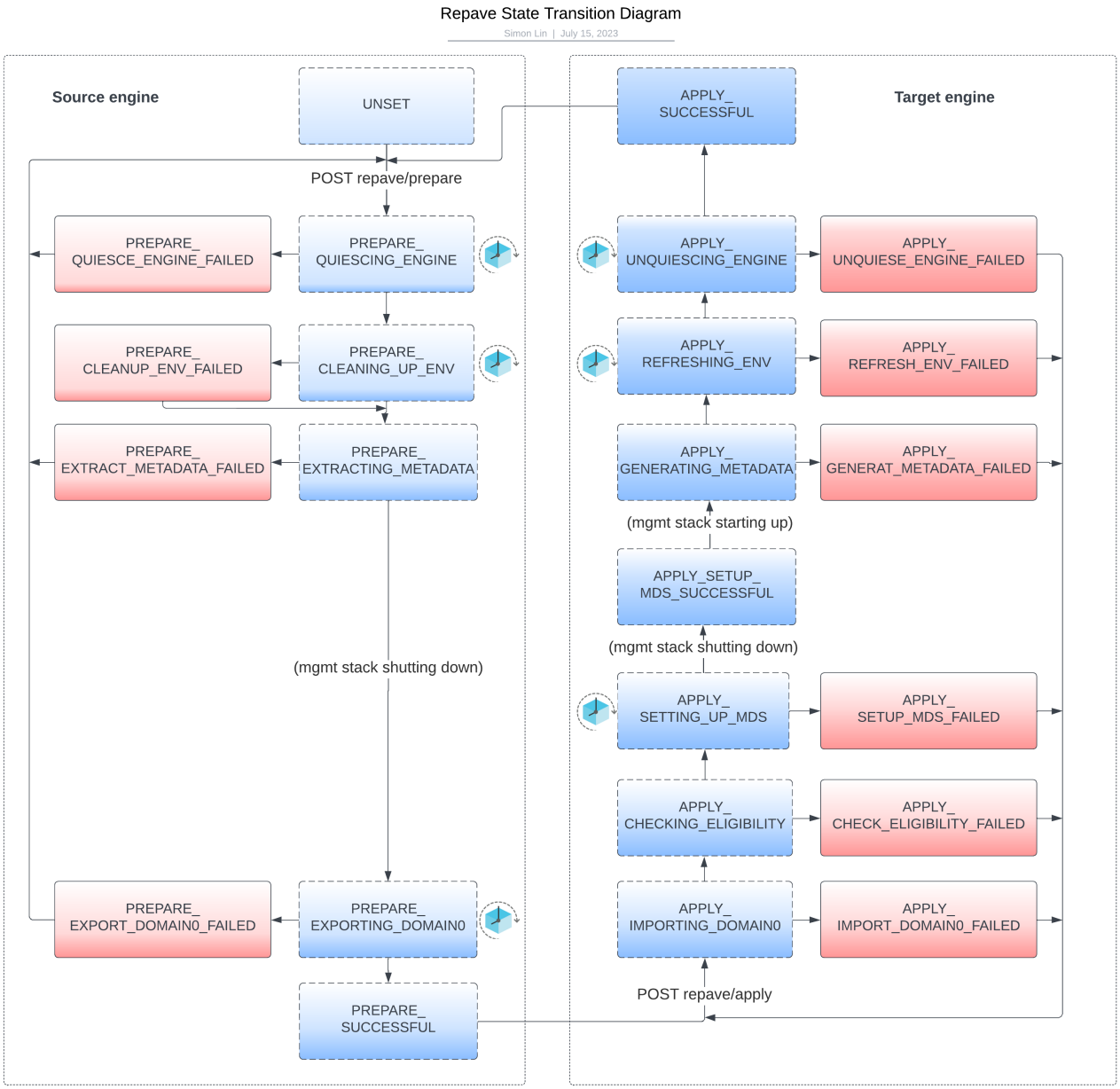
Introduction

`GET /repave` will show:

- Latest state of Repave.
- Summary of the current engine.
- Metadata of the current engine that will be migrated to the target engine.

Tips

- Always use `GET /repave` to check repave progress.
- When the Delphix management service is restarting, `GET /repave` might not respond, wait for a few minutes and try again.
- Use the repave state transition diagram below for reference.



Repave preview

`http://{{delphix_engine_url}}/resources/json/delphix/repave/preview`

Introduction

- `repave/preview` is for users to preview the summary and metadata of a source engine.
- The `APPLY_ELIGIBLE` state indicates the current engine is eligible to apply repave.
- The `APPLY_CHECK_ELIGIBILITY_FAILED` state indicates the current engine is not eligible to apply repave, the actual reason can be found from `stateDetail`.

Request parameters

One type of parameter is supported for version 13.0.

- `BlockStorageRepavePreviewParameters` is for engines with block storage.

Response

The engine summary and metadata of the source engine.

Technical details

- The engine summary and metadata of the source engine is stored in the domain0 rpool. `repave/preview` will try to import domain0 pool before previewing them, then domain0 pool will be exported again.

```
{
    "type": "BlockStorageRepavePreviewParameters"
}
```

Masking sensitive data

Encryption does not protect data that is accessed through applications and database clients, the most likely attack vector. Masking sensitive data before it gets to non-production systems is a critical tool in the security arsenal.

Delphix provides an add-on masking product for simple cost-effective integration. It is also possible to integrate Delphix with any masking technology.

There are many topologies to consider, and the complete explanation of their pros and cons is outside the scope of this document. Delphix Professional Services can assist you in analyzing various masking solutions.

Audit logs

Review audit logs monthly

Conduct a monthly review of audit logs on your Delphix Engine. Pay particular attention to provisioning operations of unmasked databases, which creates new copies of your production data. See Accessing Audit Logs for instructions.

Forward audit logs to central server via syslog

Forward audit logs to a central audit server using syslog techniques. Delphix Professional Services can assist you with scripts that facilitate this. See also Setting Syslog Preferences for configuration instructions.

Support security

At least every six months, regenerate the registration code and re-register the engine. See Regenerating the Delphix engine registration code for instructions and the reasons why this is important for security.

Delphix operating system (DxOS)

Delphix Support accesses the DxOS for deep diagnostics and troubleshooting. Access to the Delphix Engine requires access to your network. Typically this is granted via shared troubleshooting sessions over Webex, with full transparency.

If desired, you can enable additional control so that access can only take place when you provide a token.

If you disable Support Access, do not have the token, and you are unable to login as a system administrator, it can become impossible for Delphix Support to login to repair your system. Example: the management stack crashes, the login system becomes unavailable, and you have disabled Support Access and do not have the token. For this reason, Delphix strongly recommends leaving Support Access enabled at all times. If you wish to disable access, generate a unique token once a month and place it in a secure location separate from the Delphix Engine.

Disable support access (optional)

Support Access Control is managed as a system administrator in the Server Setup area.

- When set to DISABLED, it is impossible for Delphix Support to login to the DxOS.
- When set to ENABLED (the default), Delphix Support can login to the DxOS.
- When set to ENABLED and calendar time is set and a token generated, Delphix Support can only login with the token during that timeframe, which you provide. Generate the token once/month for an entire month and store it in a secure location separate from your Delphix Engine. When requested, provide the token through a secure means: in your support ticket, via email or SMS to a trusted entity, etc.

Password policies

Getting started

The password policy feature allows users to create their own custom password policies and enforce the password policy on non-LDAP Delphix Engine users.

Understanding password policies

A password policy is a named password policy that can be assigned to a user. It is a set of requirements that passwords must satisfy.

- minLength - A password must be longer than this length.
- reuseDisallowLimit - The user should not reuse old passwords. This tells the number of last used passwords disallowed to be reused as the new passwords.
- uppercaseLetter - A password must have at least one capital letter.
- lowercaseLetter - A password must have at least one lower case letter.
- digit - A password must have at least one digit.
- symbol - A password must have at least one symbol.
- disallowUsernameAsPassword - A password should not be the same as the user name.

Password policy requirements

When you set a password, it must differ from the most recent password and contain:

- at least 5 characters
- at least one uppercase letter
- at least one lowercase letter
- at least one numeric digit
- at least one symbol such as #, $, !
- do not use username or reverse username

This policy applies to non-LDAP Delphix Engine users. This includes the default users, **delphix_admin** and **sysadmin**. The password policy does not apply to LDAP users.

Default password policy

By default, the Delphix Engine enforces the password policy named **NONE**, which enforces the least possible constraint.

Passwords must contain at least one character.

Changing the password policy

To change the current password policy from the default policy **NONE**, create a custom password policy and select it instead of **NONE**.

Who can change password policy for whom

- Domain administrators can change the current password policy for all domain users.
- System users can change the current password policy for all system users.
- Domain regular users (non-administrators) users can only view the password policy.

What operations can be done by administrators

- Create custom password policies
- Update custom password policies
- Delete custom password policies
- Change the current password policy to any of the available password policies
- View available password policies
- View current password policy requirements

Password policy parameters

When you create a password policy, you can set the following parameters:

- Unique name for the password policy
- Minimum length of the password
- Whether password must differ from the last password
- Whether password must not contain the username or reverse user name
- Whether password must contain at least one uppercase letter
- Whether password must contain at least one lowercase letter
- Whether password must contain at least one numeric digit
- Whether password must contain at least one symbol such as #, $, !

Restrictions

- Restrictions for default password policy's modification (named **NONE**):
    - not allowed to delete the default password policy from available list of password policies.
    - not allowed to update any parameters of the default password policy.
- Cannot delete the password policy which is set as current password policy.

Additional topics

The Delphix Engine provides robust, enterprise-quality security controls. Performing the steps listed in this document will allow you to easily bring your Delphix Engines into compliance with your organization's security policies.

Perform a yearly audit

At least once annually, audit one or more Delphix Engines to ensure compliance with your security policies.

Port scan

Delphix fully supports network security scans, using a tool of your choosing.

Security testing

Many companies require security testing of applications in their environment using a Port Scanner or other Security Penetration Test tools. Delphix supports the use of these security tools with the application credentials available for the engine (e.g., delphix_admin). The Delphix Engine is a closed appliance, and OS credentials on the appliance are not provided for these tests.

Security banner

You can configure a custom security banner that will be displayed to all the users prior to login.

1. Login to the CLI using the sysadmin **username** and **password**.

```
ssh sysadmin@yourdelphixengine
```

2. Set the configuration and commit the changes.

```
delphix > service security
delphix service security > update
delphix service security update * > set banner="Your Message Here"
delphix service security update * > commit
```

The string set in the banner is in plain text only. You can also generate banners with multiple lines, however, this can only be done using the "securityConfig" API. Refer to the Delphix KB article for additional information.

Virtual database security

The Delphix Engine provides advanced storage capabilities and automation to allow rapid provisioning of virtual databases (VDBs), which use only a fraction of the physical storage used by full database copies. Nonetheless, a VDB is equivalent to a physical database **and must be properly secured like any other database**.

By far the most dangerous attack vectors in the Delphix ecosystem are the same ones that existed pre-Delphix: unauthorized access to your non-production systems containing sensitive production data. **You must perform all the same actions to harden virtual databases as you would to harden physical clones**.

For information on securing your virtual databases, consult vendor-specific material and security guides.

Datasets

Datasets

A key part of using Delphix is how we integrate with various data sources. In the Getting Started section, you can find information related to general dataset usage, that is, that usage that is not specific to any data platform. This encompasses topics such as adding environments, creating their users, and using Delphix objects such as virtual databases (VDBs). Separately, there are also data platform-specific features that are only applicable to a certain database. If you need to learn more about the requirements or workflows of a particular data platform, you can find them in the named sections below.

- Getting started
- IBM Db2 environments and data sources
- MySQL environments and data sources
- Oracle environments and data sources
- Oracle E-Business Suite (EBS) environments and data sources
- PostgreSQL environments and data sources
- SAP ASE environments and data sources
- SAP HANA environments and data sources
- SQL Server environments and data sources
- Unstructured files and app data

Getting started

In order to start using Delphix, you'll need to connect one or more data sources and appropriate hosts for an engine. These are key concepts you'll need to understand as an administrator using Delphix with any data source. An outline of the contents in this section:

This section covers the following topics:

Managing environments and hosts

In Delphix, an environment is either a single instance host or cluster of hosts that run database software. For example, a Linux system running Postgres. This may either be an individual instance, or a cluster like Oracle RAC. Environments can either be a source (where data comes from), staging (where data are prepared/masked) or target (where data are delivered and used by developers and testers).

Each environment has its own properties and information depending on the type of environment it is. In this section, you can learn about how to add environments to Delphix as well as what attributes and configurations that environments have.

Environment attributes and configurations

This topic describes the attributes of an environment such as name, host address, ssh port, or toolkit path. For more advanced attributes for specific data platforms, please navigate to that data platform's documentation section.

Environment attributes

Attribute	Description
Environment user(s)	The user(s) for the environment. These are the users who have permission to ssh into an environment (Unix-based environments) or access the environment through the Delphix Connector (Windows-based environments).
Host address	The fully qualified domain name or IP address for the environment.
NFS addresses (Unix-based hosts only)	A list of IP addresses to be used for NFS mount for this environment. If left empty, Delphix will automatically use the Host Address parameter above for NFS requests
Delphix session protocol (DSP) Configurations	If an environment already exists after enabling server/client DSP authentication, you will need to modify its attributes for host communication to continue working, you will need to set up the appropriate stores on the remote host.
SSH port	Environment port used for SSH, usually port 22. This parameter is only used for Unix-based environments.
Toolkit path	Location of the toolkit. For Windows hosts, this will be specified once installing the Delphix Windows Connector.

Attribute	Description
Java path	Location of the Java Development Kit (JDK) used for the host. Only specified if the feature to provide your own JDK is enabled, otherwise, the defaults are used per our Java Support Matrix. (link) Delphix uses Eclipse Adoptium (formerly known as AdoptOpenJDK) by default. You can provide your own Java when adding an environment and in the environment configuration page. This includes Oracle JDKs, Adoptium Temurin JDKs, and JDKs specific to HPUX and AIX hosts.
Notes	Any other information you want to add about the environment

Environment users

Not to be confused with Delphix administrator users and sysadmin users, environment users must be created for each environment to perform actions on each host. These users are configured for each data platform to interface with the database instance on the host.

1. Login to the Delphix Management application using Delphix Admin credentials.
2. Click **Manage**.
3. Select **Environments**.
4. Click on the existing environment name you want to modify and open the environment information screen.
5. In the **Details** tab, click the **Plus** icon located next to Environment users.
6. There are two ways that can be used for the Delphix Engine to login into the environment.
    a. Enter the Username and Password for the OS user in that environment and click **Validate**.
    b. If you want to use a public key for logging into your environment:
        i. Select 'Username and Public Key' for the Login Type.
        ii. Copy the public key that is displayed, and append it to the end of ~/.ssh/authorized_keys file of the new user being added. If this directory or file does not exist, you will need to create it.
        iii. Run chmod 600 authorized_keys to enable only the file owner with read and write privileges.
        iv. Run chmod 755 ~ to make your home directory writable only by your user and no other user may write to it.
        v. The public key needs to be added only once per user and per environment.
7. Click the **Add** icon to save the new user.
8. To change the primary user for this environment, select the environment. Then click the **'star'** icon next to **Environment Users**. Only the primary user will be used for environment discovery.
9. To delete a user, click the **Trash** icon next to their username.

Environment operations

Environments managed by Delphix have two operations, refresh and delete.

1. **Environment refresh**: will update the metadata associated with that environment and sends a new toolkit to the host.
2. **Environment delete**: will remove all metadata associated with an environment on the engine.

Environment refresh

After you make changes to an environment that you have already set up in the Delphix Management application, such as installing a new database home, creating a new database, or adding a new listener, you may need to refresh the environment to reflect these changes.

During environment discovery and environment refreshes, Delphix pushes a new copy of the toolkit to each host environment and will replace the old toolkit with the newer one. Included in the toolkit are:

- A Java Runtime Environment (JRE)
- Delphix jar files
- The HostChecker utility
- Scripts for managing the environment and/or VDBs
- Delphix Connector log files

When you refresh the environment it will push the toolkit back to the directory identified as the "Toolkit Path" for the given environment. Once this completes, you should be able to bring the dSource back online.

Delphix then executes some of these scripts to discover information about the objects in your environment (where the databases are installed, the database names, information required to connect to these databases, etc.). In some environments (Windows in particular), the scripts are customized to fit your environment.

> An environment refresh or discovery operation does not alter the source configuration of manually added databases. If you have added the databases manually, then Delphix does not update its source configuration upon discovering a change.

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. In the Environments panel, click the name of the environment you want to refresh.
5. Select the **Refresh** icon.
6. In the Refresh confirmation dialog select **Refresh**.

Environment delete

Deleting an environment in Delphix will remove all data and configurations related to that environment. This action only affects the environment metadata stored in the Delphix Engine. It will not affect your database installations or homes on the hosts or clusters that the environment is referencing.

Before you can delete an environment, you must first delete all dependencies such as dSources and virtual databases (VDBs)

1. Login to the **Delphix Management** application.
2. Select **Environments**.
3. In the Environments panel, select the environment you want to delete.
4. Click the Actions (...) menu, and select **Delete**.
5. In the Delete Environment confirmation dialog click **Delete**.

Using HostChecker to validate environments

HostChecker is a standalone program which validates that host machines are configured correctly before the Delphix Engine uses them as data sources and provision targets.

Please note that HostChecker does not communicate changes made to hosts back to the Delphix Engine. If you reconfigure a host, you must refresh the host in the Delphix Engine in order for it to detect your changes.

You can run the tests contained in the HostChecker individually, or all at once. You must run these tests on both the source and target hosts to verify their configurations. As the tests run, you will either see validation messages that the test has completed successfully, or error messages directing you to make changes to the host configuration.

HostChecker is distributed as a set of Java files and executables. You can find these files and executables in five distinct tarballs, each containing a different jdk corresponding to a particular platform (OS + processor). Together, these tarballs comprise the set of engines supported by Delphix.

When validating hosts during a new deployment, it is important to download the appropriate tarball for the host you are validating. Tarballs follow the naming convention "hostchecker_<OS>_<processor>.tar." For example, if you are validating a linux x86 host, you should download the tarball named hostchecker_linux_x86.tar.

HostChecker is also included in the Delphix Toolkit which is pushed to every environment managed by the Delphix Engine. It can be found in /<toolkit-path>/<Delphix_COMMON>/client/hostchecker.

Privilege elevation profiles

Privilege Elevation Profiles exist to provide the Delphix Engine with a mechanism for running privileged commands in a secure way to achieve the following:

- Mount and Unmount NFS filesystems
- Create and Remove directories in paths not owned by the Delphix OS user
- Examine the running process list
- Run commands as root

This is an advanced CLI topic and intended for advanced end-users and Delphix Professional Services consultants.

Support for privilege elevation profiles

Writing and troubleshooting scripts, such as those required for Privilege Elevation Profiles, is out of scope and not covered by Delphix Support.

How do privilege elevation profiles work?

Privilege Elevation Profiles need to be tailor-made to work with non-standard environments that may use third-party or proprietary a privilege elevation mechanism other than sudo. Customers are strongly encouraged to work with Delphix Professional Services to formulate reliable profile scripts.

Privilege Elevation Profiles exist within a two-tier cascading hierarchy. This means there is one default profile for the entire Delphix Engine that should contain scripts for all the operations that require privilege elevation. Additional profiles may contain a subset of the scripts. When a non-default profile is used, the Delphix Engine uses that profile's scripts where they exist and reverts to the scripts in the default profile if no script for the operation exists. By default, the Delphix Engine ships with simple scripts that pass commands to the standard UNIX sudo command.

All Environments added to the Delphix Engine get added with the default Privilege Elevation Profile. The profile can be assigned on a per-host basis. Below shows how a host using a non-standard profile will use scripts in the cascading model.

In order to create a privilege elevation profile, you must create both a profile and a profileScript. Scripts exist for particular operations, which include:

- dlpx_mount
- dlpx_umount
- dlpx_rmdir
- dlpx_mkdir
- dlpx_ps
- dlpx_pfexec

There are three parameters when creating a new profile:

1. name:
2. contents:
3. profile:

The Delphix toolkit

The Delphix Toolkit is a group of files that allow communication between Delphix engines and remote hosts. Included in the Delphix Toolkit are:

- Database and Host Scripts
- Delphix Session Protocol (DSP) binaries
- Delphix HostChecker
- Java Development Kit (JDK)

The Delphix Toolkit is automatically sent to Environments at the specified Toolkit Path whenever adding or refreshing an environment.

Toolkit size and predicted growth

Delphix Toolkit's size depends on two factors:

1. The number of Delphix client applications running on the environment
2. The number of VDBs on the environment (if any)

There are currently three Delphix client-side applications: SnapSyncClient, the Delphix Connector, and V2P. Each of the clients that run from the client-side toolkit generates their own logs - info, trace, debug, error. Each level of logging is restricted to a maximum of 10 log files and these log files are capped at 10 MB each.

Therefore the maximum space occupied by the toolkit directory on the Source server is its initial size of 400 MB + 1200 MB = 1.6 GB.

However, on the target server, the maximum toolkit size is initial size 400 MB + 800 MB + (Number of VDBs * 1MB)

Providing your own Java

Delphix allows the ability for users to provide their own JDKs for each host. Customers who want to use a more recent version than the one that comes with Delphix can do so when adding an environment to a given engine. Note that the JDK is provided per node or host, not per cluster or environment. Initially, when creating a cluster all hosts must have Java at the same absolute path. However, once the cluster has been discovered in Delphix each host's path can be adjusted. Supported JDKs include Adoptium Temurin OpenJDK, AIX, HPUX, and Oracle. Customers who have a Java license with Oracle can follow the instructions on the Providing Your Own Oracle Java page.

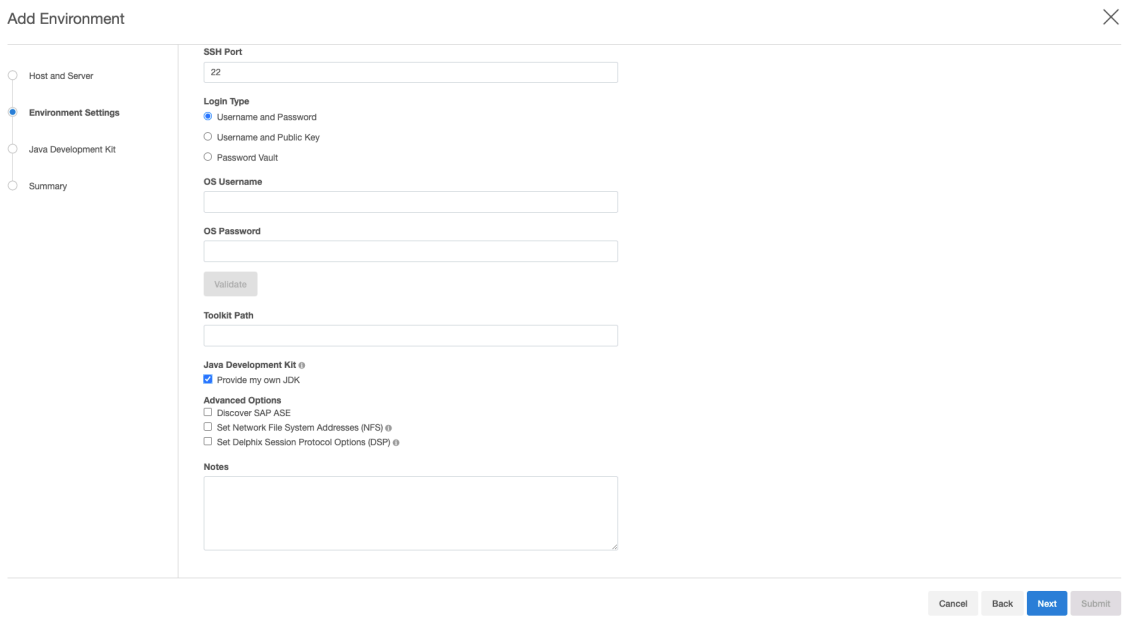
⚠ All Delphix environment users on the host require read and execute permissions on the provided JDK, its subfolders, and files.

⚠ Delphix only supports using custom JDKs for Oracle, Adoptium Temurin, AIX, and HPUX. All other JDKs are not supported.

Adding a JDK

By default, Delphix comes with Adoptium's OpenJDK. To modify the JDK, follow the below steps:

1. log in to the **Delphix management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Click the **Actions (...)** menu next to **Environments and** select **Add environment.**
5. In the **Environment settings** tab, select the **Provide my own JDK** checkbox, and click **Next**.
   This action will remove the previous built-in JDK and will initiate an Environment refresh operation after the path is changed.



⚠ Select Reset to Default to reset your JDK to the OpenJDK default.

6. In the **Java development kit** tab, the currently selected JDK kit will be shown (default is OpenJDK). Provide an absolute path to the JDK root and click **Next.**
7. Verify the configured JDK and environment settings and click **Submit.**

Updating a JDK path

For existing environments, users can add or update JDK paths by following the steps below:

1. Log in to the **Delphix management** application.
2. Select **Manage > Environments**.
3. Click on the name of an environment to view its basic information.
4. In the Details tab next to the **Java development kit**, click the **Pencil** icon.

5. In the **Java development kit dialog**, provide an absolute path to the JDK root and click **Next.**

> Do not place the JDK inside the Delphix Toolkit.

6. Click the **Check** icon to save your changes.

Finding a JDK

Custom JDKs need to be present on a host being added to Delphix and can be sourced directly from the supported vendors. These JDKs must be of Java major version 8 and come from one of the following:

- Oracle
- Adoptium Temurin OpenJDK
- HP-UX Java
- IBM Java for AIX

Delphix toolkit native Java support matrix

This matrix describes the supported versions of Java we package with the Delphix toolkit for each operating system. This is the default option if you do not use the feature to provide your own Java for each host.

	RHEL	**SLES**	**Solaris x64**	**Solaris sparc9**	**AIX**	**HP-UX**	**Windows**
AdoptOpenJDK 8u332-b09	Supported	Supported	Supported	N/A	N/A	N/A	Supported

	RHEL	SLES	Solaris x64	Solaris sparc9	AIX	HP-UX	Windows
AdoptOpenJDK 8u322-b06	N/A	N/A	N/A	Supported	N/A	N/A	N/A
IBM Java 8.0.0.620	N/A	N/A	N/A	N/A	Supported	N/A	N/A
HP Java 8.0.21	N/A	N/A	N/A	N/A	N/A	Supported	N/A

Java support policy

Delphix is committed to testing and certifying the JDK versions that are included with the product, along with specific versions provided by Oracle (see the Providing Your Own Oracle Java page for details). While using custom JDKs from Adoptium Temurin, AIX, and HPUX is generally supported, there is no certification process for beyond those detailed in the native Java support matrix above.

Managing data sources and syncing data

A dSource is a virtualized representation of a physical or logical source database, which cannot be accessed using database tools. In order to use dSource snapshots, you must create a virtual database (VDB), an independent, writable copy of a dSource snapshot.

Once an environment has been added, Delphix will automatically discover any data sources on the host. These appear as dSources on the Databases page of each environment. This section will describe how to manage these dSources and sync data from your sources. The following operations and concepts exist for dSources:

- Link/Sync
- Timeflows
- Snapshots
- SnapSync and LogSync
- Enable/Disable
- Detach/Reattach
- Delete
- Upgrade

Linking dSources

Once you have discovered your data source environments, you will see a list of data sources on that host. The first step to sync data is to link a dSource from the host.

Linking a dSource will ingest data from the source and create a dSource object on the engine. The dSource is an object that the Delphix Virtualization Engine uses to create and update virtual copies of your database. As a virtualized representation of your source data, it cannot be managed, manipulated, or examined by database tools.

Once a dSource has been linked, you can begin to perform dSource operations on it, such as enable, detach, delete, and more.

The dSource timeflow

Both dSources and VDBs have Timeflows, which are a collection of snapshots for a particular data set. Virtual databases can be provisioned from any snapshots in a Timeflow. In the UI, a Timeflow is represented vertically with a series of individual snapshots as rows.

The dSource snapshot

Snapshots of dSources preserve specific points in time for use later during the virtual database provisioning workflow. Taking a snapshot will create a new snapshot entry in the dSource's Timeflow.

SnapSync and LogSync

This dSource Timeflow described above is maintained through the use of SnapSync and LogSync.

SnapSync

SnapSync will pull over the complete data set during the initial load using standard database protocols.

Subsequent SnapSync operations will pull only the incremental changes and store them with optimal storage efficiency. At the end of each SnapSync operation, a snapshot is created that serves as the base point for provisioning operations.

When provisioning a VDB, the closer the origin point is to a snapshot created via SnapSync, the sooner the provisioning operation will complete.

LogSync

In addition to SnapSync, LogSync will periodically connect to the host(s) running the source database via standard protocols and pull over any log files associated with the database. These log files are stored separately from the SnapSync data and are used to provision from points in between SnapSync snapshots.

When provisioning from a point between snapshots, the additional time it takes to provision is directly proportional to the time difference between the provision point and the last snapshot. The rate of change on the source database dictates the amount of data that must be replayed to bring a virtual database to the correct point in time.

Enabling and disabling dSources

Some operations, such as dSource upgrade, are not available unless the dSource is disabled. Disabling a dSource turns off communication between the Delphix Engine and the source database, but it does not remove the configuration that enables communication and updates data. When a disabled dSource is later enabled, it will resume communication and incremental data updates from the source database according to the original policies and data management configurations that you set.

Disabling a dSource is also a prerequisite for several other operations, such as:

- database migration
- upgrading the dSource metadata after the upgrade of the associated data source
- restoring the source database from a backup

Disabling a dSource will stop further operations on the Delphix Engine related to the dSource.

1. Login to the **Delphix Management** application.
2. Select the dSource you want to disable.
3. In the upper right-hand corner, from the Actions menu (...) select **Disable**.
4. In the Disable dialog, select **Disable**.

When you are ready to **enable** the dSource again, from the Actions menu (...) select **Enable**, and the dSource will continue to function as it did previously.

Detaching and reattaching dSources

Each dSource contains metadata that associates it with the source database, as well as the data it has ingested from the source database in the form of snapshots up to that point.

You may want to detach a dSource from a data source and reattach it to another source. It is possible to detach, or unlink, a dSource from its source database. This breaks the association with the source database without affecting the data within the Delphix Engine.

Detached dSources and their source databases have these properties:

- A detached dSources can still be used to provision a virtual database (VDB).
- You can re-link the source database as a different dSource.
- Any child VDBs that were provisioned from this dSource will only be able to be refreshed from the most recent snapshot available on the dSource.
- If you need a VDB from a newer snapshot, you will need to provision a new VDB. Once you have provisioned the new VDB you can delete the old VDBs provisioned from this dSource. You can delete the old dSource when it is no longer needed.

Detaching a dSource

1. Login to the **Delphix Management** application.

2. Select the dSource you want to detach.
3. From the Actions menu (...), select **Unlink dSource**.
   A warning message will appear.
4. Click **Unlink** to confirm. The status of the dSource will show as Detached.

When you are ready to **attach** the dSource again, from the Actions menu (...) select **Link dSource**, and the dSource.

Deleting dSources

Deleting a dSource will delete the dSource metadata for a particular source database, along with all associated snapshots, logs, and policies stored in Delphix. If a dSource is deleted, it will not affect your source database.

Prerequisites

You cannot delete a dSource that has dependent virtual databases (VDBs). Before deleting a dSource, ensure that you have deleted all dependent VDBs as described in *Deleting a VDB on* Provisioning and Managing Virtual Databases. Otherwise, the delete option will be grayed out.

> For dSources, you cannot delete a dSource that has dependent virtual databases (VDBs) or virtual Pluggable databases (vPDBs). Before deleting a dSource, ensure that you have deleted all dependent VDBs or vPDBs. Otherwise, the delete option will be grayed out. When you delete all the dSources of PDBs, the associated CDBs are automatically deleted.

Procedure

1. Login to the **Delphix Management** application.
2. Select the dSource you want to delete.
3. From the Actions menu (...), select **Delete**.
4. Click **Delete** to confirm.

Upgrading dSources

Occasionally, you may upgrade the source database that a dSource is associated with. In this case, you will need to update the dSource's metadata to associate a dSource with a more recent database version, thus performing an 'upgrade' of the dSource's known version.

While this feature does not upgrade the actual database, it is important that Delphix associates this dSource with the correct version of the source database to function properly.

This process of upgrading a dSource will modify the database installation path to use the path of the newer database installation.

The process for the upgrade of dSources will vary depending on the database. Refer to the following pages for database-specific procedures:

- Upgrading dSources after an Oracle Upgrade
- Upgrading a dSource after a SQL Server Upgrade
- Upgrading a SAP ASE dSource

Prerequisites

- The data source instance has been upgraded to a more recent installation version.
- The dSource must be disabled.

Provisioning and managing virtual databases

After you have added a data source from an environment and created snapshots of a dSource, you can begin to provision virtual databases (VDBs), one of the key parts of using Delphix.

As discussed before, a dSource is a virtualized representation of a physical or logical source database, which cannot be accessed or manipulated using database tools. In order to use dSource snapshots, you must create a VDB, an independent, writable copy of a dSource snapshot.

You can create VDBs from other VDBs. Once a VDB has been provisioned to a target environment, you can also create a snapshot policy for that VDB, to capture changes within it as if it were any other logical or physical database.

The following operations and concepts exist for VDBs:

- Provision
- Start/Stop
- Refresh
- Rewind (Rollback)
- Enable/Disable
- Delete
- Force delete
- Migrate
- Upgrade
- Virtual to physical (V2P)
- VDB configuration templates

Provisioning virtual databases

Virtual databases are a key data management concept for Delphix. In order to create or provision a virtual database, you will need a linked dSource from a source host and a compatible target environment, as described in the overview for Managing environments and hosts. Since provisioning VDBs varies significantly across data platforms, details on this process can be found within the documentation for a specific data platform.

At their core, virtual databases are fully functional copies of data sources that can be created and managed at a fraction of the storage and time that are typically required. To end-users, VDBs act and perform just like a standard database.

The VDB timeflow

Both dSources and VDBs have timeflows, which are a collection of snapshots for a particular data set. Virtual databases can be provisioned from any snapshots in a Timeflow. The Virtual database's filesystem will be dependent upon the snapshot used to Provision, and as such, that snapshot can not be removed until the dependency is removed. In the product UI, a Timeflow is represented vertically with a series of individual snapshots as rows.

The VDB snapshot

Snapshots of dSources preserve specific points in time for use later during the virtual database provisioning workflow. Taking a snapshot will create a new snapshot entry in the VDBs Timeflow.

Refreshing virtual databases

Refreshing a VDB will re-provision it from the dSource. As with the normal provisioning process, you can choose to refresh the VDB from a snapshot or a specific point in time. However, you should be aware that refreshing a VDB will delete any changes that have been made to it over time. When you refresh a VDB, it will revert to its past state in the specified snapshot or point in time.

When performing a VDB refresh, there is an option to choose between either the faster or more accurate point in time for that database.

> Although the VDB no longer contains the previous contents, the previous Snapshots and TimeFlow still remain in Delphix and are accessible through the Command Line Interface (CLI).

Prerequisites

To refresh a VDB, you must have the following permissions:

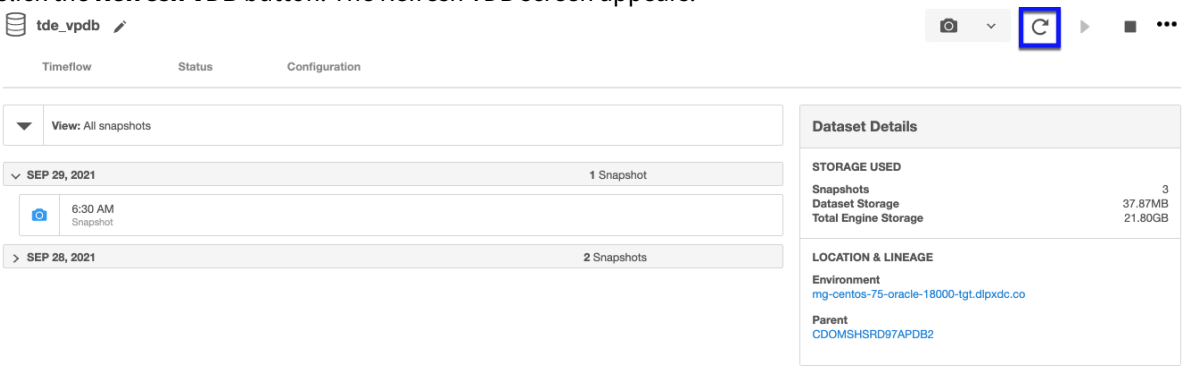
- **PROVISIONER** permissions on the dSource associated with the VDB
- **PROVISIONER** permissions on the group that contains the VDB
- **Owner** permissions on the VDB itself
- **Data** is a role that allows DB_ROLLBACK, DB_REFRESH, READ_ACTION, DB_SYNC, JOB_CANCEL**.**
- **Read** is a role that allows the user to inspect objects via the READ_ACTION permission.

A user with admin credentials can perform a VDB Refresh on any VDB in the system.

Procedure

Perform the following steps to refresh a VDB.

1. Login to the **Delphix Management** application.
2. Navigate to **Manage** > **Datasets**.
3. Select the VDB that you want to refresh.
4. Click the **Refresh VDB** button. The Refresh VDB screen appears.



5. Select one of the refresh options and click **Next**.
    a. **Faster** - This option will utilize the most recent snapshot in the timeflow.
    b. **More accurate** - This option allows you to select a snapshot, a point in time, or a specific log ID.
6. Click **Submit** to confirm.
7. Click the **Actions** button to watch the progress of the refresh job.
8. If you want to know when the VDB was last refreshed/provisioned, check the Time Point on the **Status** page.

Rewinding virtual databases

Rewinding a VDB rolls it back to a previous point in its Timeflow and re-provisions the VDB. The VDB will no longer contain changes after the rewind point.

> **Rewind VDB**
> Although the VDB no longer contains changes after the rewind point, the rolled-over Snapshots and Timeflow still remain in Delphix and are accessible through the Command Line Interface (CLI). See the topic CLI Cookbook: Rolling forward a VDB for instructions on how to use these snapshots to refresh a VDB to one of its later states after it has been rewound.

> Delphix clones a new Timeflow from the closest Snapshot older than or equal to the rewind point. This creates a dependency between the new Timeflow and the parent Snapshot and Timeflow. The parent Snapshot and Timeflow cannot be deleted because of this dependency. The VDB must first be refreshed before the parent Snapshot and Timeflow can be removed.

Prerequisites

To rewind a VDB, you must have the following permissions:

- **Auditor** permissions on the dSource associated with the VDB
- **Owner** permissions on the VDB itself

You do NOT need **owner** permissions for the group that contains the VDB. A user with admin credentials can perform a VDB Rewind on any VDB in the system.

Procedure

1. Login to the **Delphix Management** application.
2. Select the VDB you want to rewind.
3. Select the **Timeflow** tab.
4. Select the rewind point as a snapshot or a point in time.
5. Click **Rewind**.
6. If you want to use login credentials on the target environment other than those associated with the environment user, click **Provide Privileged Credentials**.
7. Click **Rewind** to confirm.

> **Using Timeflow bookmarks**
> You can use TimeFlow bookmarks as the rewind point when using the CLI. Bookmarks can be useful to:
> - Mark where to rewind to - before starting a batch job on a VDB for example.
> - Provide a semantic point to revert back to in case the chosen rewind point turns out to be incorrect.
>
> For a CLI example using a TimeFlow bookmark, see CLI Cookbook: Provisioning a VDB from a Timeflow Bookmark

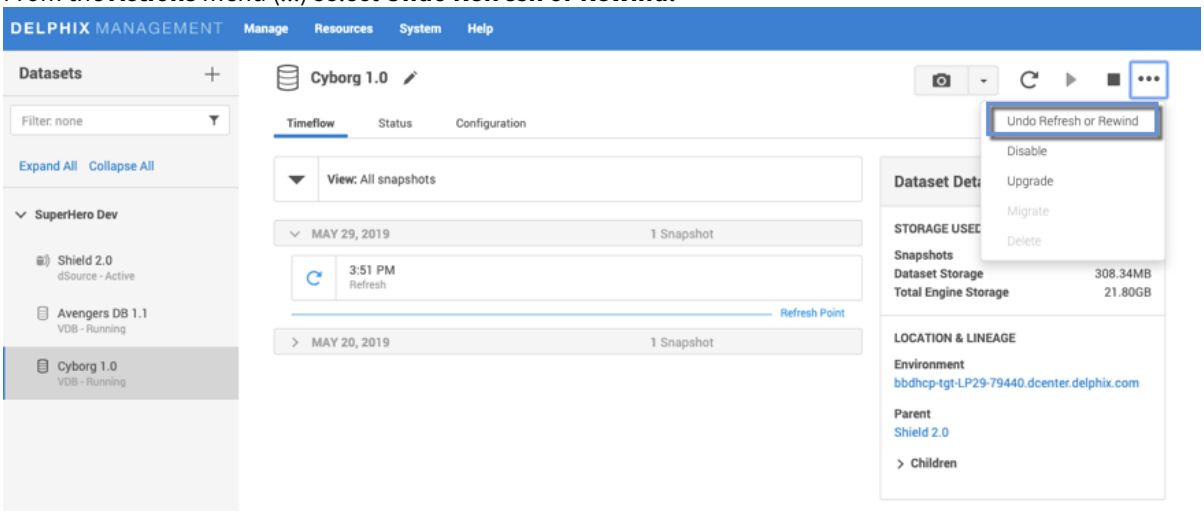
Undoing a rewind or refresh

An accidental refresh of one or more virtual databases (VDBs) from a source database has removed important data. To correct this situation users can undo VDB refresh or rewind actions.

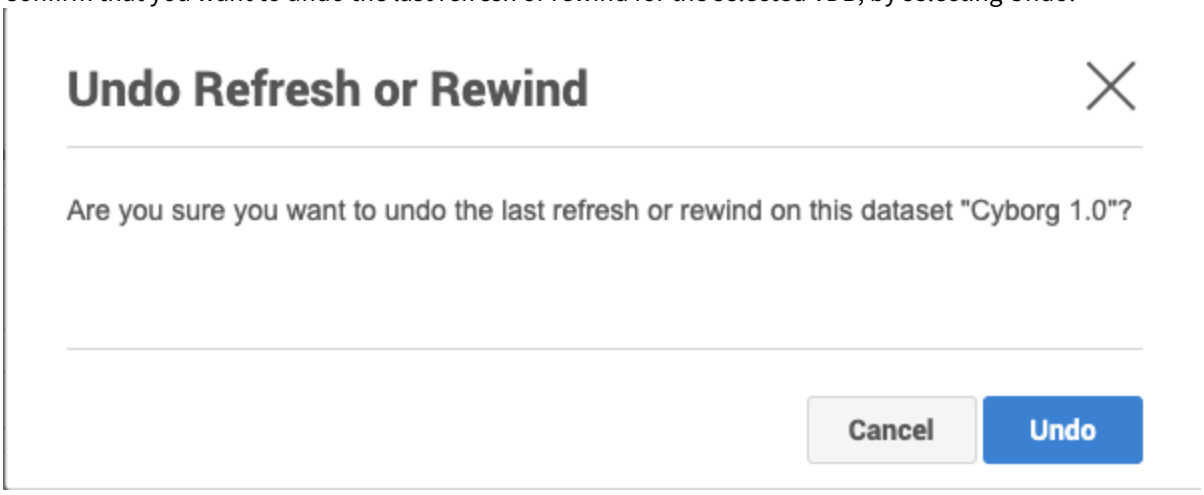
Procedure

To undo a refresh or rewind via the Delphix Management application:

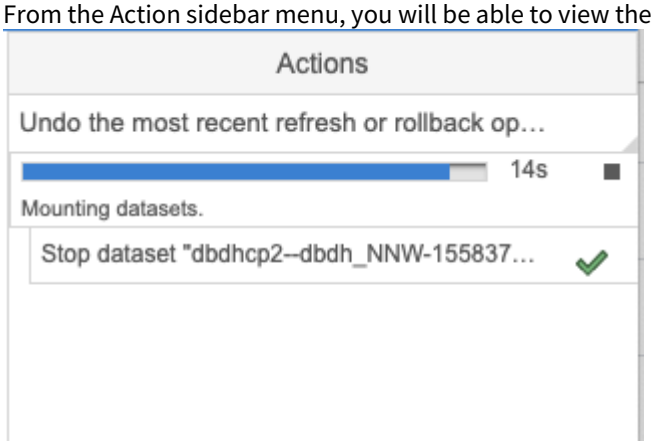
1. Login to the **Delphix Management** application.
2. Under **Datasets**, select the **VDB** you want to undo your rewind/refresh for.
3. From the **Actions** menu (...) select **Undo Refresh or Rewind.**



4. Confirm that you want to undo the last refresh or rewind for the selected VDB, by selecting Undo.



From the Action sidebar menu, you will be able to view the undo action.



Starting and stopping virtual databases

Stopping your virtual databases is necessary if you are performing any actions on the database that require it to be offline. In addition to properly stopping the database, the Delphix stop and start actions also unmount and mount the filesystems that are provided by the Delphix Engine as well as performing any Delphix-specific actions during startup or shutdown.

1. Login to the **Delphix Management** application.
2. Select the VDB you want to manage.
3. To stop an active VDB, click the **stop** icon in the top right-hand corner.
   To start an inactive VDB, click the **start** icon in the top right-hand corner.
4. If stopping or starting the VDB requires particular credentials for the target environment other than those of the default environment user:
   a. Check Provide Privileged Credentials.
   b. Enter the username and password.
   c. Click Validate Credentials.
5. Click Yes to stop or start the VDB.
6. You can view the status of the stopping or starting of the VDB in several ways:
   View the **Active Jobs** on the right of the screen or monitor the progress bar from the VDB Status tab.

Enabling and disabling virtual databases

Disabling a VDB is a prerequisite for procedures such as VDB migration or upgrade. Disabling a VDB removes all traces of it, including any configuration files, from the target environment to which it was provisioned. However, the VDB itself, as well as the metadata, will still exist on the engine.

When you later enable the VDB again, these configuration files are restored on the target environment.

1. Login to the **Delphix Management** application.
2. Select the VDB you want to disable.
3. From the Actions menu select **Disable**.
4. Click **Disable** to acknowledge the warning.

When you are ready to enable the VDB again, from the Actions menu select **Enable**, and the VDB will continue to function as it did previously.

Deleting virtual databases

Deleting a VDB will remove it and it's Timeflow and snapshots from the Delphix Engine entirely. This operation is non-reversible.

Prerequisites

You cannot delete a VDB that has dependent VDBs. Before deleting a VDBs, make sure that you have deleted all dependent VDBs. Otherwise, the delete option will be grayed out.

> ⚠ Deleting a VDB is an unrecoverable operation. Proceed only if you want to permanently destroy the unique data that was created in the VDB.

Procedure

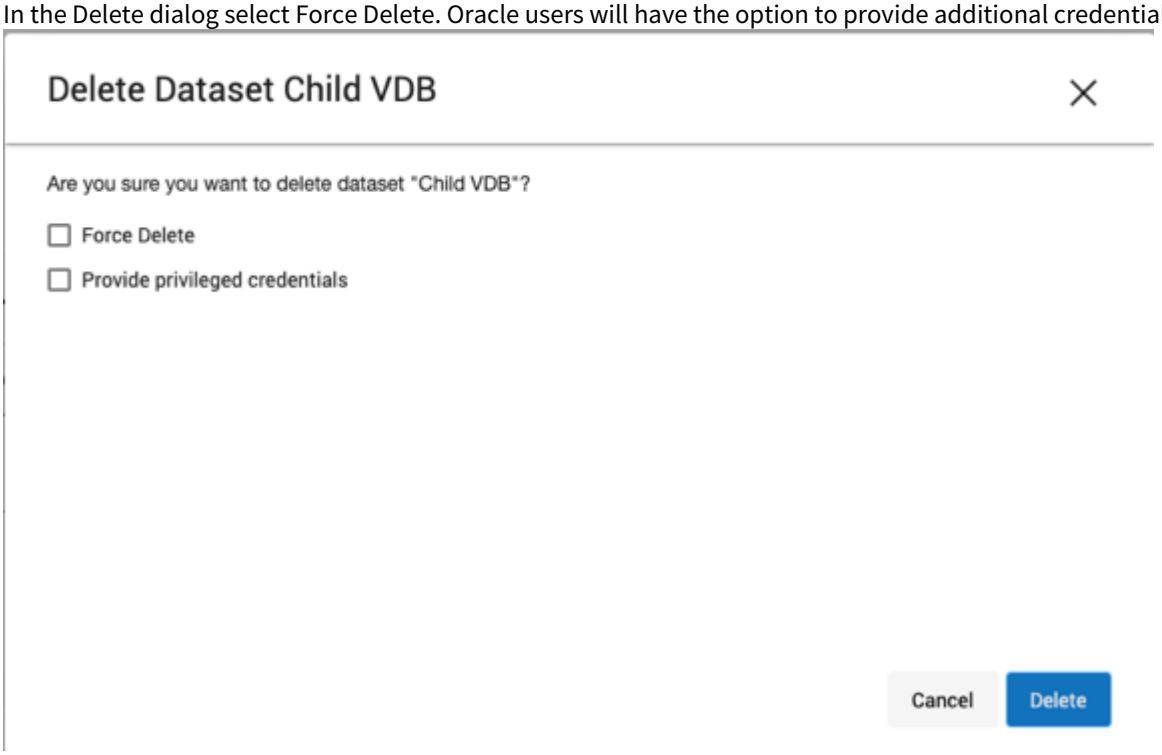
1. Login to the **Delphix Management** application.
2. Select the VDB you want to delete.
3. From the **Actions** menu select Delete.
4. Click **Delete** to confirm.

Force Delete
Deleting a vFiles may fail if it cannot be unmounted successfully from all target environments. You can use the **Force Delete** option to ignore all failures during unmount.

Using force delete

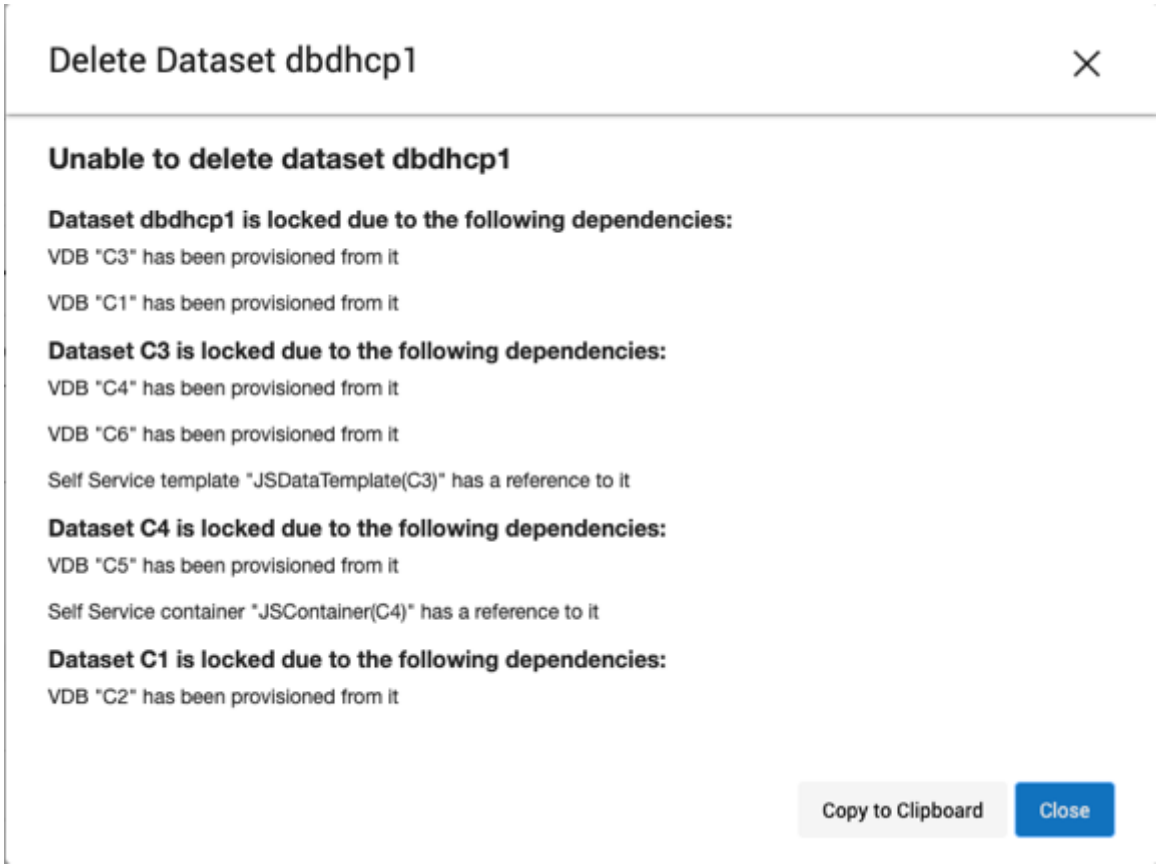
Deleting unused or outdated objects should be a regular part of Delphix Engine administration. This is especially important to prevent low space errors, which can cause the Delphix Engine to stop. The Delphix Engine holds a maximum of 400 objects.

1. Log into the **Delphix Management** application.
2. Select **Resources > Storage Capacity**.
3. Next to the object you want to delete select the **Trashcan**.
4. In the Delete dialog select Force Delete. Oracle users will have the option to provide additional credentials.



5. Click Delete.

Dependencies
If there are dependencies on the snapshot, you will not be able to delete the snapshot free space; the dependencies rely on the data associated with the snapshot. These items are displayed with a lock icon next to the name.



Migrating virtual databases

There may be situations in which you want to migrate a virtual database to a new target environment, for example when upgrading the host on which the VDB resides, or as part of a general data center migration. Using the VDB migrate functionality allows you to change the VDB's location from one target environment to another.

This is easily accomplished by first disabling the database, then using the Migrate VDB feature to select a new target environment.

Prerequisites

- You should have already set up a new target environment that is compatible with the VDB that you want to migrate.
- Ensure that the VDB has first been disabled

Procedure

1. Login to the **Delphix Management** application.
2. Select the VDB you want to migrate.
3. From the Actions menu select **Migrate**.
4. Select the new target environment for the VDB, the user for that environment, and the database installation where the VDB will reside.
5. Select **Migrate** to confirm your selections.
6. From the **Actions** menu select **Enable**.
7. Click **Enable** to confirm.

Within a few minutes, your VDB will re-start in the new environment, and you can continue to work with it as you would any other VDB.

Upgrading virtual databases

Similar to upgrading a data source, upgrading a VDB changes the installation path that is associated with that data platform. While this feature does not upgrade the actual VDB, it is important that Delphix associates this VDB with the correct database version to function properly.

By performing this upgrade, you can update the version of the database installation used for running the VDB.

Procedure

1. Login to the **Delphix Management** application.
2. Select the VDB you want to upgrade.
3. From the **Actions** menu select **Upgrade**.

Virtual to physical (V2P)

Virtual to Physical, or V2P, is the process of exporting a virtual database to a physical one.

After you have created a dSource or a VDB, you can export its contents and log files to a physical database. This process creates a set of directories on the target environment and populates them with the database data, log files, and scripts that are used to recover the physical database.

You can automatically start the physical database recovery process as part of V2P, or you can use the scripts for manual recovery. When the export process completes, the target environment will contain a copy of the database in its unvirtualized size, so before you begin the process, make sure the target directories you specify during V2P have enough capacity to hold the database. For more information, visit the following page: Virtual to physical

Procedure

1. Login to the **Delphix Management** application.
2. Select the dSource or VDB you want to export.
3. Select the snapshot of the dSource or VDB state you want to export.
4. From the **Actions** menu select **Virtual to Physical**.
5. Select the target environment, and fill the other input fields depending on the data platform of your object.

Provisioning with replication

This topic describes how to provision from a replicated dSource or virtual database (VDB).

The process for provisioning from replicated objects is the same as the typical VDB provisioning process, except that first you need to select the replica namespace containing the replicated object.

Prerequisites

- You must have a dSource or VDB that has been replicated from one Delphix Engine to another, as described in Replication overview
- The Delphix Engine containing the replicated dSource or VDB must have a compatible target environment that it can use to provision a VDB from the replicated dSource or VDB.
- On the Delphix Engine containing the replicated dSource or VDB, login to the Delphix Management application.

Procedure

1. From the list of replica namespaces, select the replica namespace that contains the dSource or VDB from which you want to provision.
2. The provisioning process is now identical to the process for provisioning standard objects.

Virtual to physical

This section covers the following topics:

- Customizing target directory structure for database export
- Manually recovering a database after V2P
- V2P with unstructured files

Customizing target directory structure for database export

This topic describes how to customize the target directory layout for database export.

In the V2P export process, it may be necessary to customize the target directory structure which the files will be exported to. The following is the default directory structure:

- Data files: `<target directory>/data`
- Archive files: `<target directory>/archive`
- Temp files: `<target directory>/temp`
- External files: `<target directory>/external`
- Script files: `<target directory>/script`

Note: The example on this page uses `/` for file separators which is relevant for Unix and Linux environments. If the target environment is Windows, the file separator will be `\`.

The following procedure describes how to customize the directory layout.

Procedure

1. During the virtual to physical export process, click **Advanced** in the V2P Wizard to see the target directory options.
2. You can customize any of the following:
    - **Data directory**
    - **Archive directory**
    - **Temp directory**
    - **External directory**
    - **Script directory**
3. Each directory will then be concatenated to the **Target directory** separated by the appropriate separator.

> Any one of **Target directory, Data directory**, **Archive directory**, **Temp directory**, **External directory**, **Script directory** can be blank. However, the combination of the fields must form an absolute path.
>
> - Data files: `<target directory>/<data directory>`
> - Archive files: `<target directory>/<archive directory>`
> - Temp files: `<target directory>/<archive directory>`
> - External files: `<target directory>/<external directory>`
> - Script files: `<target directory>/<script directory>`

Examples

Target directory is not empty

This means all target directories have a common root.

Input	Final directory layout
• **Target directory:** `/mytarget` • **Data directory:** `/mydata` • **Archive directory:** `/myarchive` • **Temp directory:** `/mytemp` • **External directory:** `/myexternal` • **Script directory:** `/myscript`	• Data files: `/mytarget/mydata` • Archive files: `/mytarget/myarchive` • Temp files: `/mytarget/mytemp` • External files: `/mytarget/myexternal` • Script files: `/mytarget/myscript`

Target directory is empty

All target directories may not have a common root. Note that external files and temp files share the same common root.

Input	Final Directory Layout
• **Target directory:** • **Data directory:** `/mydata` • **Archive directory:** `/myarchive` • **Temp directory:** `/mytarget/temp` • **External directory:** `/mytarget/external` • **Script directory:** `/myscript`	• Data files: `/mydata` • Archive files: `/myarchive` • Temp files: `/mytarget/temp` • External files: `/mytarget/external` • Script files: `/myscript`

Target directory is empty and data directory `/`

Combined with Customizing VDB file mappings, exporting data files to separate file systems is possible. In this example, `a.dbf` and `b.dbf` can be exported to `/filesystem1` and `/filesystem2` respectively. For Oracle VDBs, it is necessary to specify the parameter `useAbsolutePathForDataFiles` in order to export data files to separate file systems. See CLI Cookbook: V2P (virtual to physical) of a single instance Oracle database with datafiles on separate file systems for details.

Input	Final directory layout

• **Target directory:**
• **Data directory:**
• **Archive directory:** `/myarchive`
• **Temp directory:** `/mytarget/temp`
• **External directory:** `/mytarget/myexternal`
• **Script directory:** `/myscript`
• **File mappings:**
  • `a.dbf : /filesystem1/a.dbf`
  • `b.dbf : /filesystem2/b.dbf`
	• Data files:/
	• `/filesystem1/a.dbf`
	• `/filesystem2/b.dbf`
	• Archive files: `/myarchive`
	• Temp files: `/mytarget/mytemp`
	• External files: `/mytarget/myexternal`
	• Script files: `/myscript`

Target directory is empty and one of the subdirectories is empty would result in an error.

Input	Final Directories
• **Target directory:** • **Data directory:** `/mydata` • **Archive directory:** • **Temp directory:** `/mytarget/temp` • **External directory:** `/mytarget/myexternal` • **Script directory:** `/myscript`	• INVALID

For parallel execution of V2P operations from the same target, make sure that the scripts directory name is unique.

Manually recovering a database after V2P

This article describes how to manually recover a database after the V2P process.

> For the Oracle database, while running V2P, Delphix copies the datafiles and other required files on the physical storage. After that it performs the database recovery, followed by some post recovery steps and openDatabase.
>
> While running V2P from CLI, you can set `recoverDatabase=false` and `openDatabase=false`.
>
> While running V2P from UI, you can select not to openDatabase.
>
> If `recoverDatabase=false` is set, run both `recover-vdb.sh` and `open-vdb.sh`.
>
> If the V2P was performed with `recoverDatabase=true` and `openDatabase=false`, only run `open-vdb.sh`.

Procedure

1. In the V2P target environment, navigate to the scripts directory for the exported database instance. You can find the scripts in a sub-directory named for that specific database instance. For Oracle databases, the path is `<target_directory>/<db_unique_name>/script/<instance name>`. For SQL Server databases, the path is `<target_directory>\<db_name>\scripts`.
2. For **Oracle** databases, locate the `recover-vdb.sh` and `open-vdb.sh` scripts, then run the required script(s) as described above (as `sh ./recover-vdb.sh` and `sh ./open-vdb.sh`).

   For **SQL server** databases, locate the script `Provision.ps1` and run it.
3. For **Oracle** databases, add the recovered database to /etc/oratab and **Refresh** the target environment for it to discover the recovered database. For **SQL server** databases, when the script completes, **Refresh** the target environment for it to discover the recovered database.

V2P with unstructured files

This topic describes the procedure for performing virtual to physical (V2P) operations with unstructured files.

> V2P Not Supported for Unstructured Files on Windows.
> V2P is not supported for unstructured files on Windows environments. Similar results to V2P may be achieved by provisioning a vFiles and copying data out of the vFiles to the local machine.

Procedure

1. Log into the **Delphix management** application.
2. Select the dataset you want to export.
3. Select the snapshot you want to export.
4. From the **Actions** (...) menu select **Virtual to physical**.
5. Select the target environment.
6. Enter the **Mount path** for the export.
   The directory you enter here must exist in the target environment, and you must have permission to write to it.
7. Click **Next**.
8. Review the **Target environment** configuration information, and then click **Submit**.

Hook scripts for automation and customization

Hook operations allow you to execute custom operations at select points during linking sources and managing virtual datasets. For example, you may want to prevent your monitoring systems from triggering during VDB startup and shutdown. In this case, you would leverage pre- and post-hooks to run required scripts for VDB start/stop operations.

Unix-based operating systems will execute hook scripts in bash, while Windows operating systems will execute using PowerShell.

Each Delphix object will have its own set of hooks you can create. To easily manage hooks across multiple datasets, you can create hook templates to apply the same hook to many objects.

Setting hook operations

You can construct hook operation lists through the Delphix Management application or the command-line interface (CLI). You can either define the hooks as part of the provisioning process or edit them on virtual datasets that already exist.

Hook operations can be created in the Hooks tab of the Add dSource or Add VDB wizards.

To edit hook operations on a dataset, navigate to the datasets page and find the object's configuration. You will see a 'Hooks' tab where you can manage and modify hook operations.

Hook operation templates

You can use templates to store commonly used operations, which allows you to avoid repeated work when an operation is applicable to more than a single virtual dataset. You manage templates through the Delphix Management application.

Hook operations list

dSource hooks

Hook	Description
Pre sync	Pre Sync hook operations are executed before a Snapsync of a dSource. These operations can quiesce data to be captured during the Snapsync, or stop application processes that may interfere with the Snapsync operation.
Post sync	Post Sync hook operations are executed after a Snapsync operation of a dSource. These operations can undo any changes made by the pre-sync hooks. They will run regardless of the success of pre-sync hook operations or the Snapsync itself.

Virtual database hooks

Hook	Description
Configure clone	Operations performed after initial provision or after a refresh. This hook will run after the virtual dataset has been started. During a refresh, this hook will run before the Post Refresh hook.
Pre refresh	Operations performed before a refresh. This hook will run before the virtual dataset has been stopped. These operations can cache data from the virtual dataset to be restored after the refresh completes.
Post refresh	Operations performed after a refresh. This hook will run after the virtual dataset has been started and after the Configure Clone hook. This hook will not run if the refresh or Pre Refresh hook operations fail. These operations can restore cached data after the refresh completes.
Pre rollback	Operations performed before a rollback, also known as rewind. This hook will run before the virtual dataset has been stopped. These operations can cache data from the virtual dataset to be restored after the rewind completes.
Post rollback	Operations performed after a rollback, also known as rewind. This hook will not run if the rewind or Pre Rewind hook operations fail. This hook will run after the virtual dataset has been started. This hook will not run if the rewind or Pre Rewind hook operations fail. These operations can restore cached data after the rewind completes.
Pre snapshot	Operations performed before a snapshot. These operations can quiesce data to be captured during the snapshot, or stop processes that may interfere with the snapshot.
Post snapshot	Operations performed after a snapshot. This hook will run regardless of the success of the snapshot or Pre Snapshot hook operations. These operations can undo any changes made by the Pre Snapshot hook.

Hook	Description
Pre start	Operations performed before the startup of a VDB or vFile. These operations can be used to initialize configuration files or stop processes that might interfere with the virtual dataset.
Post start	Operations performed after the startup of a VDB of vFile. These operations can be used to clean up any temporary files, or restart processes that may have been stopped by a Pre-Start hook, or log notifications.
Pre stop	Operations performed before the shutdown of a VDB or vFile. These operations can quiesce data or processes prior to the virtual dataset shutdown.
Post stop	Operations performed after the shutdown of a VDB or vFile. These operations can be used to log notifications, clean up any temporary files, or stop/ restart related processes.

Order of execution of hooks for various VDB Operations

Provision	Refresh	Rollback	Snapshot	Start	Stop
Pre start	Pre refresh	Pre Rollback	Pre snapshot	Pre start	Pre stop
Post start	Pre stop	Pre stop	Post snapshot	Post start	Post stop
Configure clone	Post stop	Post stop			
Run masking job	Pre start	Pre start			
Pre snapshot	Post start	Post start			
Post snapshot	Configure clone	Configure clone			
	Run masking job	Run masking job			
	Post refresh	Post rollback			
	Pre snapshot	Pre snapshot			
	Post snapshot	Post snapshot			

Staging server hooks

Staging Server hooks are currently applicable to SAP ASE and staged plugins (Db2, HANA, and PostgreSQL) environments, and will execute on the configured staging server.

Hook	Description
Pre validated sync	Operations performed on a staging server before validated sync. This hook will run before the validated sync operation whenever the validated sync run is triggered.
Post validated sync	Operations performed on staging server after validated sync. This hook will run after the validated sync operation whenever the validated sync run is triggered.

Self-service interaction with hooks

These are the operations performed by the various Self-Service operations:

Rollback
Reset, Undo, and any Restore or Branch operation using source point-in-time or a bookmark that's part of the current container.

Refresh
Refresh and any Restore or Create Branch operation from any shared bookmark or when using the Template as the source of the restore (bookmark or point-in-time).

The Virtual Database Hooks described above will run accordingly. Snapshots are taken, for recovery purposes, before and after every operation that is performed. Bookmark operations will also initiate a snapshot.

Setting hook operations

You can construct hook operation lists through the Delphix Management application or the command-line interface (CLI). You can either define the operation lists as part of the provisioning process or edit them on virtual datasets that already exist.

Setting hook operations through the Delphix management application

> The provisioning wizard still imports hook operations from templates.

Hook operations can be provisioned in the **Hooks** tab of the Add dSource or Add VDB wizards.

1. Select the **type of operation** and enter a name, operation type, and script.
2. To remove an operation from the list, click the **Trash** icon on the operation.
3. When you have set all hook operations, click **Next** to continue with the provisioning process.

To edit hook operations on a virtual dataset:

> From the **Datasets** panel, you can create hook operations from a template.

1. In the **Datasets** panel, click the virtual dataset.
2. Click the **Configuration** tab.
3. Within the **Configuration** tab, click the **Hooks** tab.
4. Select the **hook** to edit.
5. The current operations at this hook will be displayed. To edit this list of operations, click the **Pencil** icon in the top right-hand corner of the tab.
6. Click the **Plus** icon to add a new operation.
7. Select the **type of operation** or click  to load a hook operation template.
8. Click the **text area** and edit the contents of the operation.
9. To remove an operation from the list, click the **Trash** icon on the operation.
10. When you have set all hook operations, click the **checkmark** to save the changes.

> The provisioning wizard still imports hook operations from templates.

Setting hook operations through the CLI

To specify hook operations during linking, edit the relevant hook's array of operations defined on the `LinkingParameters > Source > Operations` object.

To specify hook operations during provisioning, edit the relevant hook's array of operations defined on the `ProvisionParameters > Source > Operations` object.

To edit hook operations on a dSource that already exists, edit the relevant hook's array of operations defined on the `Source > Operations` object.

To edit hook operations on a virtual dataset that already exists, edit the relevant hook's array of operations defined on the `Source > Operations` object.

For more information about these CLI objects, see the following documentation in the **Help** menu of the **Delphix management** application:

- `LinkedSourceOperations`

- `VirtualSourceOperations`
- `RunCommandOnSourceOperation`
- `RunExpectOnSourceOperation` API

Example of editing hook operations through the CLI

1. Navigate to the relevant source's `VirtualSourceOperations` object.
2. Select a **hook**to edit.

```
delphix> source
delphix source> select "pomme"
delphix source "pomme" > update
delphix source "pomme" update *> edit operations
delphix source "pomme" update operations *> edit postRefresh
```

3. Add an operation at index 0.

```
delphix source "pomme" update operations postRefresh *> add
delphix source "pomme" update operations postRefresh 0 *> set
type=RunCommandOnSourceOperation
delphix source "pomme" update operations postRefresh 0 > set command= "
echo Refresh completed."
delphix source "pomme" update operations postRefresh 0 > ls
Properties
  type: RunCommandOnSourceOperation ( )
  command: echo Refresh completed. ( )
delphix source "pomme" update operations postRefresh 0 *> commit
```

4. Add another operation at index 1 and then delete it.

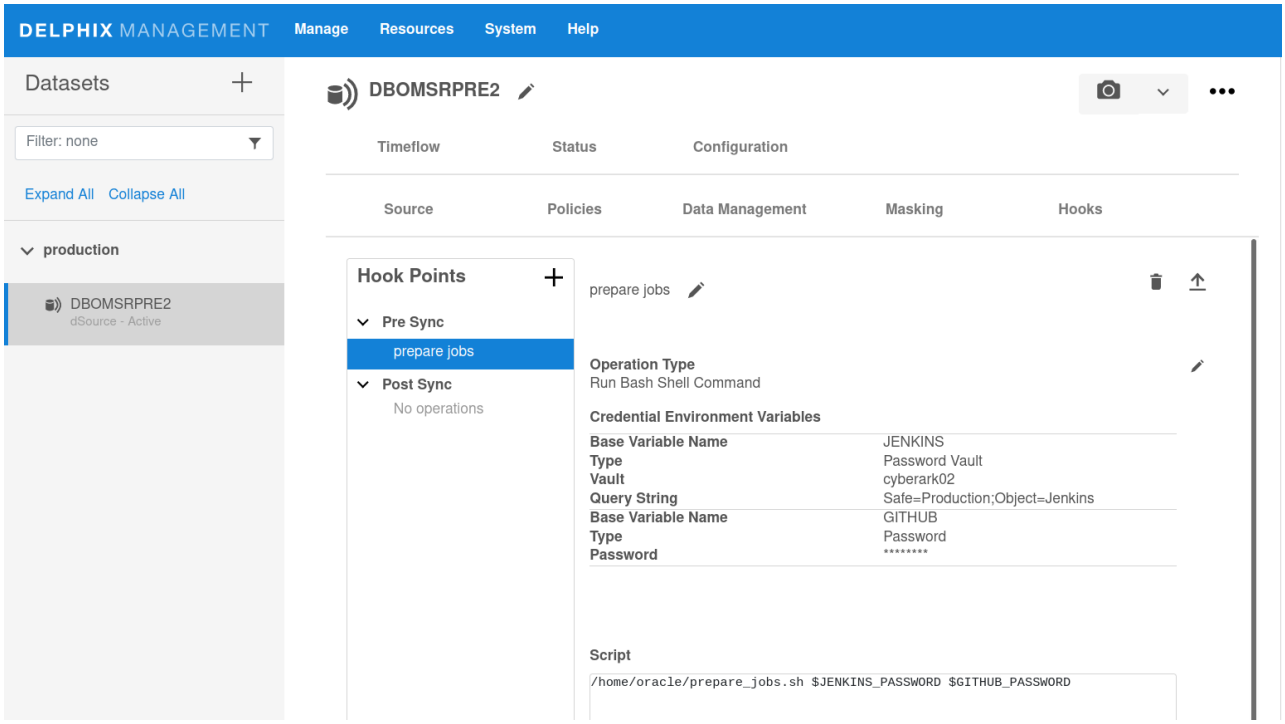
```
set command="echo Refresh completed."delphix source "pomme" update
operations postRefresh *> add
delphix source "pomme" update operations postRefresh 1 *> set
type=RunCommandOnSourceOperation
delphix source "pomme" update operations postRefresh 1 *> set command= "
echo Refresh completed."
delphix source "pomme" update operations postRefresh 1 *> back
delphix source "pomme" update operations postRefresh *> unset 1
delphix source "pomme" update operations postRefresh *> commit
```

Passing credentials securely to hook operations

Sometimes, commands in hook operations need credentials to perform tasks such as making API calls or managing local services. To avoid hard-coding credentials in the hook code, you can pass credentials securely to hooks via environment variables using a new list property of each hook operation. The elements in this list define a set of environment variables and the credentials to pass to the hook in those variables.

In the browser, the list can be viewed and updated in the **Credential environment variables** section of a hook operation. (Hook operations can be accessed in the dataset and hook template creation dialogs and view/edit pages.) For example, to view or edit hook credentials for a dataset:

1. Login to the **Delphix management** application.
2. Under **Datasets**, select a **dSource** or **VDB**.
3. Select the **Configuration** tab and then the **Hooks** tab.
4. Select a hook operation.
5. To make any changes, such as adding, modifying or deleting credentials, click on the edit icon to the right of **Operation type**.



In the API and CLI, the credentials list property of each hook operation is an array called `credentialsEnvVarsList`.

Hook credentials can be configured directly as a password or key (entered by the user) or indirectly by selecting a password vault and a location in that vault. Credentials entered directly are managed securely by the engine in the same way as other passwords and keys, which are encrypted in disk and neither logged nor exposed by the API, whereas vault credentials are managed securely and externally to the engine by a specialized product such as CyberArk or HashiCorp Vault that has already been configured in the engine. Credentials stored in a vault are always retrieved by the engine just before executing the hook, so they will always contain the latest values. Any number of credentials of any type can be defined for an operation in the `credentialsEnvVarsList` property.

Four types of credentials can be defined for hooks:

- `CyberArkVaultCredential`
- `HashiCorpVaultCredential`
- `PasswordCredential`
- `KeyPairCredential` (supported only via the API and CLI at the moment)

For more information on each of these types, visit your engine's API page for that type at https://<engine address>/api/#<type name> (you must log in first).

To illustrate, consider a hook that needs to authenticate to two different APIs, one using an API key entered directly by the user and another using a username and password managed by a vault. The user can configure the credential variables as follows:

```
delphix source "pomme" update operations postRefresh *> edit
0.credentialsEnvVarsList
delphix source "pomme" update operations.postRefresh 0 credentialsEnvVarsList
*> adddelphix source "pomme" update operations.postRefresh 0
credentialsEnvVarsList 0 *> set baseVarName=API1delphix source "pomme" update
operations.postRefresh 0 credentialsEnvVarsList 0 *> edit credentialsdelphix
source "pomme" update operations.postRefresh 0 credentialsEnvVarsList 0
credentials *> set type=PasswordCredentialdelphix source "pomme" update
operations.postRefresh 0 credentialsEnvVarsList 0 credentials *> set
password="API-KEY-02a0b73f"delphix source "pomme" update
operations.postRefresh 0 credentialsEnvVarsList 0 credentials *> backdelphix
source "pomme" update operations.postRefresh 0 credentialsEnvVarsList *>
adddelphix source "pomme" update operations.postRefresh 0
credentialsEnvVarsList 1 *> set baseVarName=API2delphix source "pomme" update
operations.postRefresh 0 credentialsEnvVarsList 1 *> edit credentialsdelphix
source "pomme" update operations.postRefresh 0 credentialsEnvVarsList 1
credentials *> set type=CyberArkVaultCredentialdelphix source "pomme" update
operations.postRefresh 0 credentialsEnvVarsList 1 credentials *> set
queryString="Safe=Apis;Folder=Root\Service2;Object=ApiClient"delphix source
"pomme" update operations.postRefresh 0 credentialsEnvVarsList 1 credentials
*> set vault=MyCyberArk1delphix source "pomme" update operations.postRefresh 0
credentialsEnvVarsList 1 credentials *> commit
```

When this hook executes, it will receive two sets of environment variables containing the corresponding credentials: `API1_PASSWORD` with value `API-KEY-02a0b73f` and variables `API2_USER` and `API2_PASSWORD` containing the values stored in the vault `MyCyberArk1` at the location of the provided query string. The `API2` credentials are retrieved from the vault just before the hook executes, so they will always contain the latest values.

The complete list of environment variables that may be available to a hook is:

- `<base name>_USER` : user name. Present only if the credentials are of a vault credential type and the entry in the vault contains a user name.
- `<base name>_PASSWORD` : a password. Present for `PasswordCredential` . Also present in case of a vault credential type and the vault entry contains a password.
- `<base name>_PRIVKEY` : a private key. Present for `KeyPairCredential` . Also present in case of a vault credential type and the vault entry contains a private key.
- `<base name>_PUBKEY` : a public key. Present for `KeyPairCredential` . Also present in case of a vault credential type and the vault entry contains a public key.

Hook operation templates

You can use templates to store commonly used operations, which allows you to avoid repeated work when an operation is applicable to more than a single virtual dataset. You manage templates through the Delphix Management application.

> **Hook operations templates not available via CLI**
> Hook operation templates cannot be fully utilized from the CLI. Manage and use hook operations through the Delphix Management application.

Creating a hook operation template

> The provisioning wizard still imports hook operations from templates.

1. Login to the **Delphix management** application.
2. Click **Manage**.
3. Select **Operation templates**.
4. Click the **Plus** icon to add a new operation template.
5. Enter a **Name** for the template.
6. Select an operation **Type**.
7. Enter a **Description** detailing what the operation does or how to use it.
8. Enter operation **Contents** to implement the operation partially or fully.
9. Click **Create**.

Importing a hook operation template

To import a hook operation template:

1. In the **Datasets** panel, select a dataset.
2. Click the **Configuration** tab.
3. Within the **Configuration** tab, click the **Hooks** tab.
4. Select the **hook** to edit.
5. Click the **Plus** icon to add a new operation.
6. Click **Import**.
7. Select the **template** to import.
8. Click **Import**.
9. When you have set all hook operations, click **Check** to save the changes.

Exporting a hook operation template

To export a hook operation template:

1. In the **Datasets** panel, select a dataset.
2. Click the **Configuration** tab.
3. Within the **Configuration** tab, click the **Hooks** tab.
4. Select the **hook** to edit.
5. Click the **Plus** icon to add a new operation.
6. Select the **type of operation**.
7. Click the **text area** and edit the contents of the operation.
8. Click **Export**.
9. Enter a **Name** for the template.
10. Enter a **Description** detailing what the operation does or how to use it.
11. Click **Export.**

Considerations for Delphix hook operations on Oracle RAC VDBs

When performing data management operations like SnapSync, refresh, and so on with Delphix data sources such as VDB and dSources, Delphix offers "hooks" as an ability to perform tasks prior to and after these data management operations. For an introduction and more details about hooks refer to, Oracle hooks for data sources.

Mostly the above article discusses shell operations within the hooks. But sometimes you might need to run a SQL script within the hook to capture or alter the database state before or after a Delphix operation on that source. In a sample scenario, we might be connecting to the database and running some SQL script that does the operations and perhaps generates some files for later use.

In such a case, there is no need to set the Oracle environment in the script, and in fact, setting the ORACLE_SID value in the script can be a problem for RAC instances. From the article linked above, these are the details of hook operations as they relate to RAC VDBs:

Oracle RAC

When linking from, or provisioning to Oracle RAC environments, hook operations will not run once on each node in the cluster. Instead, the Delphix Engine picks a node in the cluster at random and guarantees all operations within any single hook will execute serially on this node.

> The Delphix Engine does not guarantee the same node is chosen for the execution of every hook but does guarantee that Pre-/Post- hook pairs (such as Pre-Sync and Post-Sync) will execute on the same node.

If the data source (or VDB) is a single instance Oracle, explicitly setting variables in the script should not be a problem, since the local listener will be aware of the SID in the connect string. But for a RAC, since we will not be sure which node the hook will run against beforehand, do not statically allocate the variables, the engine will determine the connection string based on the databases section of the environments page of the admin app GUI.

Troubleshooting

Hook operations do not work for Delphx RAC installations.

Resolution

Remove any oracle environment setting parameters in the script and consider that the hook will run arbitrarily against any of the RAC nodes. Validate that the connect string works for the **Environment>Databases** section for the dSource in question.

Additional information

For more information about validating and setting connection strings refer to, Adding JDBC connection string

Cookbook of common scripts for hooks on SQL server

Customization is a huge topic and there are hundreds of ways to customize Delphix operations using hooks, but here are some example scripts which have shown up as common, and are presented here to provide guidance for building hooks.

Common scripts for hooks

What Are hooks?

Hooks are programmatic call-outs embedded before and after dataset manipulation operations within the Delphix Engine. These programmatic call-outs provide the ability for the Delphix Engine to run custom-built PowerShell scripts directly on the Windows database server.

Hooks and data operations

Delphix dSources are manipulated by "Sync" operations, in which the Delphix Engine captures data changes from backups of the source database appends them to the dSource. The pre-sync and post-sync hooks are executed in PowerShell by the "environment user" Windows domain account (delphix_trgt) on the staging target. Since the dSource is a database in recovery mode within the SQL instance on the staging target, no database operations can be performed on the dSource itself, but file-operations on the staging target are possible.

Delphix **VDBs** are manipulated by Provision, Refresh, Rewind, Snapshot, Start, and Stop operations. The Configure Clone, Pre-Refresh, Post-Refresh, Pre-Rewind, Post-Rewind, Pre-Snapshot, Post-Snapshot, Pre-Start, Post-Start, Pre-Stop, and Post-Stop hooks are executed in PowerShell by the Windows "environment user" Windows domain account (delphix_trgt) on the VDB on the target.

Here are some example use-cases and solutions using hooks:

- Your organization might require that a notification be sent to a distribution list five minutes before anyone refreshes or rewinds a VDB that is used for development or testing. You can write and test that logic in a PowerShell script and then embed it into the Pre-Refresh, Pre-Rewind, and Pre-Stop hooks on all VDBs. This PowerShell script can send email to a specified distribution list, and then pause for 300 seconds (5 minutes) before allowing the operation to continue.
    - PowerShell scripting example for email notification during Pre-Refresh and Pre-Rewind hooks
- Update data within a VDB when it is first provisioned and again after each refresh from the dSource. A very common example of this is data masking or anonymization, to prevent confidential data from being visible to the users and administrators of the VDB. Delphix has integrated data masking, but you can still initiate the appropriate job and poll until it completes using a PowerShell script.
- During a refresh of a VDB from the dSource, save the non-production database and application passwords, then restore them after the refresh is complete.
    - PowerShell scripting example for stored procedure execution during Pre-Refresh and Post-Refresh hooks
- Debug the sequence and timing of dataset manipulation operations within the Delphix Engineby logging hook executions to a file
    - PowerShell scripting example for debugging output to be generated during all VDB hooks

There are many more potential use-cases for hooks, but hopefully, these examples will begin to show the way.

Example PowerShell script for debugging

Description

This is the Windows PowerShell code for a script named **display.ps1**, which you can insert into any pre-script, post-script, or hook to display a timestamp and the values of the Windows environment variables to a log file located in the **C:\TEMP** directory on the target host. One purpose of this script, if inserted into every hook, is to better understand the order of operations within the Delphix Engine. It would also help you understand the duration of those operations.

Of course, this basic script is intended to be augmented for more custom diagnostic output, as desired.

> Disclaimer
>
> This sample code is offered for illustration purposes only, and does not carry any warranty or guarantee. Employing copies of this code, in whole or in part, indicates acceptance of all risks.

Source code for script

Downloadable copy display.ps1

```
#==============================================================================
#  File:      display.ps1
#  Type:      Powershell script
#  Date:      04-Nov 2016
#  Author:    Delphix
#
#  Licensed under the Apache License, Version 2.0 (the "License");
#  you may not use this file except in compliance with the License.
#  You may obtain a copy of the License at
#
#    http://www.apache.org/licenses/LICENSE-2.0
#
#  Unless required by applicable law or agreed to in writing, software
#  distributed under the License is distributed on an "AS IS" BASIS,
#  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
#  See the License for the specific language governing permissions and
#  limitations under the License.
#
#  Copyright (c) 2016 by Delphix. All rights reserved.
#
#  Description:
#
#    Windows Powershell script intended to be used for debugging purposes, by
#    logging timestamped variable values with the name of the hook from which
#    it was called to an output file on the user's desktop.
#
#  Usage:
#
#    display.ps1 <hook-name>
#
```

```
#    where:
#        <hook-name>    the name of the dSource or VDB hook from
#                which this script is called:
#                    For dSources...
#                        Pre-Sync, Post-Sync
#                    For VDBs...
#                        Configure-Clone,
#                        Pre-Refresh, Post-Refresh,
#                        Pre-Rewind, Post-Rewind,
#                        Pre-Snapshot, Post-Snapshot
#
#    If the hook-name does not have the correct text, then the script will not
#    not recognize it and will emit an error message.
#
#  Modifications:
#=============================================================================
param([string]$hookName)
#
#-----------------------------------------------------------------------------
# Set up variables for use within the script...
#-----------------------------------------------------------------------------
$outDir = "C:\TEMP"
$timeStamp = Get-Date -format o
#
#-----------------------------------------------------------------------------
# Output values for Windows environment variables set with hook session by the
# Delphix virtualization engine...
#
# If the name of the hook implies a VDB hook, then output the values for the
# relevant variables...
#-----------------------------------------------------------------------------
if ( ( $hookName.ToUpper() -eq "CONFIGURE-CLONE" ) -or
     ( $hookName.ToUpper() -eq "PRE-REFRESH" ) -or
     ( $hookName.ToUpper() -eq "POST-REFRESH" ) -or
     ( $hookName.ToUpper() -eq "PRE-REWIND" ) -or
     ( $hookName.ToUpper() -eq "POST-REWIND" ) -or
     ( $hookName.ToUpper() -eq "PRE-SNAPSHOT" ) -or
     ( $hookName.ToUpper() -eq "POST-SNAPSHOT" ) ) {
    $outFile = $outDir + "\HOOK_OUTPUT_" + $env:VDB_DATABASE_NAME + ".TXT"
    $timeStamp + "|" + $hookName + ": VDB_INSTANCE_HOST is '" +
$env:VDB_INSTANCE_HOST + "'" | Out-File -FilePath $outFile -Append
    $timeStamp + "|" + $hookName + ": VDB_INSTANCE_PORT is '" +
$env:VDB_INSTANCE_PORT + "'" | Out-File -FilePath $outFile -Append
    $timeStamp + "|" + $hookName + ": VDB_INSTANCE_NAME is '" +
$env:VDB_INSTANCE_NAME + "'" | Out-File -FilePath $outFile -Append
    $timeStamp + "|" + $hookName + ": VDB_DATABASE_NAME is '" +
$env:VDB_DATABASE_NAME + "'" | Out-File -FilePath $outFile -Append
} else {
    #
    #-------------------------------------------------------------------------
    # If the name of the hook implies a dSource hook, then output the values
    # for the relevant variables...
```

```
    #-------------------------------------------------------------------------
    if ( ( $hookName.ToUpper() -eq "PRE-SYNC" ) -or
         ( $hookName.ToUpper() -eq "PRE-SYNC" ) ) {
        $outFile = $outDir + "\HOOK_OUTPUT_" + $env:SOURCE_DATABASE_NAME + ".TXT"
        $timeStamp + "|" + $hookName + ": SOURCE_INSTANCE_HOST is '" +
$env:SOURCE_INSTANCE_HOST + "'" | Out-File -FilePath $outFile -Append
        $timeStamp + "|" + $hookName + ": SOURCE_INSTANCE_PORT is '" +
$env:SOURCE_INSTANCE_PORT + "'" | Out-File -FilePath $outFile -Append
        $timeStamp + "|" + $hookName + ": SOURCE_INSTANCE_NAME is '" +
$env:SOURCE_INSTANCE_NAME + "'" | Out-File -FilePath $outFile -Append
        $timeStamp + "|" + $hookName + ": SOURCE_DATABASE_NAME is '" +
$env:SOURCE_DATABASE_NAME + "'" | Out-File -FilePath $outFile -Append
    } else {
        #
        #-----------------------------------------------------------------
        # If the name of the hook is not recognized, then output an error
        # message to that effect...
        #-----------------------------------------------------------------
        $outFile = $outDir + "\HOOK_OUTPUT_UNKNOWN.TXT"
        $timeStamp + ": unknown hook name '" + $hookName + "'" | Out-File -FilePath
$outFile -Append
    }
}
```

Output

The name of the output file created by this script includes the name of the database, so all output from all hooks associated with this database name will be appended to the same file.

Example PowerShell script for email notification

Description

The PowerShell script (below) sends an email using SMTP attributes passed on the command-line (i.e. from-Email-Address, to-Email-Address, and SMTP-Server). The script then pauses for 5 minutes before returning control to the Delphix Engine.

Source code for script

Downloadable copy email_5mins.ps1

```
Description
The PowerShell script (below) sends an email using SMTP attributes passed on the
command-line (i.e. from-Email-Address, to-Email-Address, and SMTP-Server). The script
then pauses for 5 minutes before returning control to the Delphix Engine.

Source Code for Script
Downloadable copy email_5mins.ps1.

#============================================================================
# File:         email_5mins.ps1
# Type:         powershell script
# Author:    Delphix Professional Services
# Date:         04-Nov 2016
#
# Copyright and license:
#
#       Licensed under the Apache License, Version 2.0 (the "License"); you may
#       not use this file except in compliance with the License.
#
#       You may obtain a copy of the License at
#
#               http://www.apache.org/licenses/LICENSE-2.0

#
#       Unless required by applicable law or agreed to in writing, software
#       distributed under the License is distributed on an "AS IS" basis,
#       WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
#
#       See the License for the specific language governing permissions and
#       limitations under the License.
#
#       Copyright (c) 2015,2016 by Delphix.  All rights reserved.
#
# Description:
#
#    Script will send an email using the SMTP attributes passed as parameters
#    on the command-line, and then pause for 5 minutes.
#
#
# Command-line parameters:
```

```
#
#    fromEmailAddr    - originating email address (used for "reply-to")
#    toEmailAddr    - destination email address
#    smtpServer    - IP address or hostname of SMTP server
#    opName        - free-format text describing the Delphix operation
#              about to occur (i.e. "REFRESH", "REWIND", "STOP", etc)
#
#
# Environment inputs expected:
#
#    VDB_DATABASE_NAME
#
# Note:
#
# Modifications:
#    TGorman    04nov16    first version
#=============================================================================
param(   [string]$fromEmailAddr = "~~~"
    ,[string]$toEmailAddr = "~~~"
    ,[string]$smtpServer = "~~~"
    ,[string]$opName = "~~~"
)
#
#-----------------------------------------------------------------------------
# Compose the email within a "Net.Mail.MailMessage" object, then send it...
#-----------------------------------------------------------------------------
$message = New-Object Net.Mail.MailMessage
$message.From = $fromEmailAddr
$message.To.Add($toEmailAddr)
$message.Subject = "VDB '" + $env:VDB_DATABASE_NAME + "' will be " + $opName + " in 5
mins"
$message.Body = "<H1>Delphix operation " + $opName + " will occur in 5 mins</H1>"
$message.Body = $message.Body + "<BR><BR>Please save any work in this database or it
will be lost."
$message.Body = $message.Body + "<BR><BR>Contact IT Operations at x5555 if you have
any questions or concerns"
$message.IsBodyHTML = "True"
$smtp = New-Object Net.Mail.SmtpClient($smtpServer)
$smtp.Send($message)
#
#-----------------------------------------------------------------------------
# Go to sleep for 300 seconds (5 minutes)...
#-----------------------------------------------------------------------------
Start-Sleep -s 300
exit 0
```

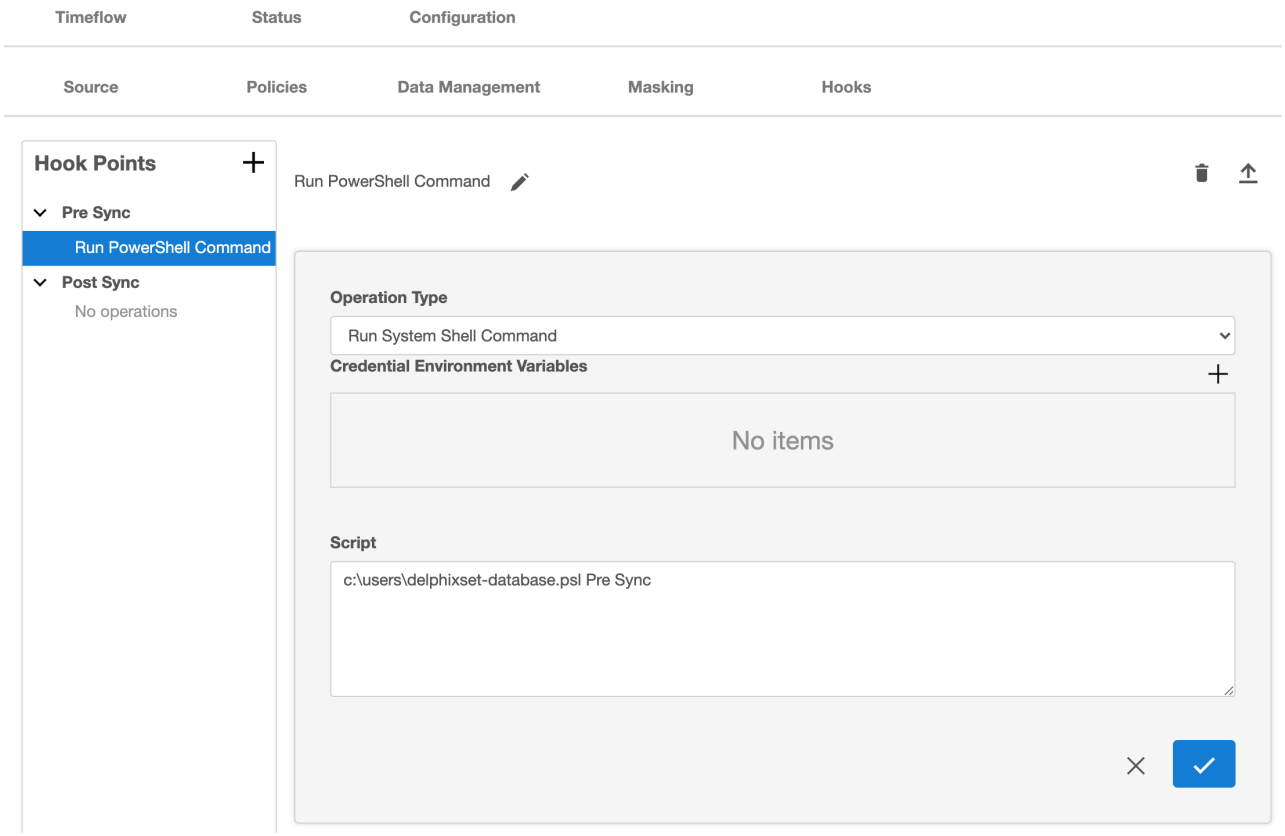
Related Topics
Cookbook of Common

Example PowerShell script for executing stored procedures

Description

The PowerShell script (below) makes a call to the stored procedure, whose fully-qualified name is passed as a parameter. This particular script is written so that the stored procedure expects only one parameter named "@DatabaseName". Please note that pre and post-scripts can only be executed on VDBs on the VDB target host. If "@DatabaseName" is not unique enough, then the script can be modified to add "@InstanceName" as well.

The script logs debugging information to a log file created within the "Desktop" directory of the Delphix "environment user" Windows account. This log file is directed to "C:\TEMP" in this example, but of course, can be modified as suitable in your environment.



Source code for script

Downloadable copy callsp.ps1

```
#============================================================================
# File:         callsp.ps1
# Type:         powershell script
# Author:     Delphix Professional Services
# Date:         02-Nov 2015
#
# Copyright and license:
#
#       Licensed under the Apache License, Version 2.0 (the "License"); you may
```

```
#       not use this file except in compliance with the License.
#
#       You may obtain a copy of the License at
#
#               http://www.apache.org/licenses/LICENSE-2.0
#
#       Unless required by applicable law or agreed to in writing, software
#       distributed under the License is distributed on an "AS IS" basis,
#       WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
#
#       See the License for the specific language governing permissions and
#       limitations under the License.
#
#       Copyright (c) 2015 by Delphix.  All rights reserved.
#
# Description:
#
#    Call the appropriate stored procedure within the DBO schema in the MSDB
#    databse on behalf of the VDB.  The stored procedure name the name of the
#    database as a parameter called "@DatabaseName"..
#
# Command-line parameters:
#
#    $fqSpName        fully-qualified stored procedure name
#
# Environment inputs expected:
#
#    VDB_DATABASE_NAME    SQL Server database name for the VDB
#    VDB_INSTANCE_NAME    SQL Server instance name for the VDB
#    VDB_INSTANCE_PORT    SQL Server instance port number for the VDB
#    VDB_INSTANCE_HOST    SQL Server instance hostname for the VDB
#
# Note:
#
# Modifications:
#    TGorman    02nov15    first version
#=============================================================================
param([string]$fqSpName = "~~~")
#
#-----------------------------------------------------------------------------
# Verify the "$dirPath" and "$fqSpName" command-line parameter values...
#-----------------------------------------------------------------------------
if ( $fqSpName -eq "~~~" ) {
    throw "Command-line parameter 'fqSpName' not found"
}
#
#-----------------------------------------------------------------------------
# Clean up a log file to capture future output from this script...
#-----------------------------------------------------------------------------
$dirPath = [Environment]::GetFolderPath("Desktop")
$timeStamp = Get-Date -UFormat "%Y%m%d_%H%M%S"
$logFile = $dirPath + "\" + $env:VDB_DATABASE_NAME + "_" + $timeStamp + "_SP.LOG"
"logFile is " + $logFile
```

```
#
#------------------------------------------------------------------------------
# Output the variable names and values to the log file...
#------------------------------------------------------------------------------
"INFO: dirPath = '" + $dirPath + "'" | Out-File $logFile
"INFO: fqSpName = '" + $fqSpName + "'" | Out-File $logFile -Append
"INFO: env:VDB_INSTANCE_HOST = '" + $env:VDB_INSTANCE_HOST + "'" | Out-File $logFile
-Append
"INFO: env:VDB_INSTANCE_NAME = '" + $env:VDB_INSTANCE_NAME + "'" | Out-File $logFile
-Append
"INFO: env:VDB_INSTANCE_PORT = '" + $env:VDB_INSTANCE_PORT + "'" | Out-File $logFile
-Append
"INFO: env:VDB_DATABASE_NAME = '" + $env:VDB_DATABASE_NAME + "'" | Out-File $logFile
-Append
#
#------------------------------------------------------------------------------
# Housekeeping: remove any existing log files older than 15 days...
#------------------------------------------------------------------------------
"INFO: removing log files older than 15 days..." | Out-File $logFile -Append
$ageLimit = (Get-Date).AddDays(-15)
$logFilePattern = $env:VDB_DATABASE_NAME + "_*_SP.LOG"
"INFO: logFilePattern = '" + $logFilePattern + "'" | Out-File $logFile -Append
Get-ChildItem -Path $dirPath -recurse -include $logFilePattern |
    Where-Object { !$_.PSIsContainer -and $_.CreationTime -lt $ageLimit } |
    Remove-Item
#
#------------------------------------------------------------------------
# Run the stored procedure...
#------------------------------------------------------------------------
"INFO: Running stored procedure '" + $fqSpName + "' within database '" +
    $env:VDB_DATABASE_NAME + "'..." | Out-File $logFile -Append
try {
    "INFO: open SQL Server connection..." | Out-File $logFile -Append
    $sqlServer = $env:VDB_INSTANCE_HOST + "\" + $env:VDB_INSTANCE_NAME + ", " +
$env:VDB_INSTANCE_PORT
    "INFO: sqlServer = '" + $sqlServer + "'" | Out-File $logFile -Append
    [System.Reflection.Assembly]::LoadWithPartialName("Microsoft.SqlServer.Smo") |
Out-Null;
    $conn = New-Object System.Data.SqlClient.SqlConnection
    $conn.ConnectionString = "Server=$sqlServer; Database=MSDB; Integrated
Security=SSPI;"
    "INFO: conn.ConnectionString = '" + $conn.ConnectionString + "'" | Out-File
$logFile -Append
    $conn.Open()
    $cmd1 = New-Object System.Data.SqlClient.SqlCommand($fqSpName, $conn)
    $cmd1.CommandType = [System.Data.CommandType]::StoredProcedure
    $cmd1.Parameters.Add('@DatabaseName', $env:VDB_DATABASE_NAME) | Out-null
    "INFO: calling " + $fqSpName + ", @DatabaseName = " + $env:VDB_DATABASE_NAME |
Out-File $logFile -Append
    $exec1 = $cmd1.ExecuteReader()
    $exec1.Close()
    $conn.Close()
```

```
} catch { Throw $Error[0].Exception.Message | Out-File $logFile -Append }
#
"INFO: completed stored procedure '" + $fqSpName + "' within database '" +
    $env:VDB_DATABASE_NAME + "' successfully" | Out-File $logFile -Append
#
#-------------------------------------------------------------------------
# Exit with success status...
#-------------------------------------------------------------------------
exit 0
```

Sample configure clone hook for SQL server

How the configure clone hook works

The Delphix Engine executes the **Configure clone** hook after the initial provisioning of a virtual database (VDB) and after each refresh of that VDB. However, during refresh operations, the **Configure clone** hook runs before the **Post-refresh** hook.

Because it executes after the Delphix Engine brings fresh data from the dSource into the VDB, the Configure Clone hook is an ideal place for data masking and other obfuscation and reconfiguration operations to occur.

The Windows PowerShell script (shown below) is intended to demonstrate how to call the HTTP interface of the Delphix DmSuite v4.7.3 from a Configure Clone hook.

Disclaimer

This sample code is offered for illustration purposes only, and does not carry any warranty or guarantee. Employing copies of this code, in whole or in part, indicates acceptance of all risks.

```
#===============================================================================
# File:         mask.ps1
# Type:         powershell script
# Author:       Delphix Professional Services
# Date:         02-Nov 2015
#
# Copyright and license:
#
#       Licensed under the Apache License, Version 2.0 (the "License"); you may
#       not use this file except in compliance with the License.
#
#       You may obtain a copy of the License at
#
#               http://www.apache.org/licenses/LICENSE-2.0
#
#       Unless required by applicable law or agreed to in writing, software
#       distributed under the License is distributed on an "AS IS" basis,
#       WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
#
#       See the License for the specific language governing permissions and
#       limitations under the License.
#
#       Copyright (c) 2015 by Delphix. All rights reserved.
#
# Description:
#
#    Powershell script to automate the call to Delphix/AXIS data-masking when
#    the appropriate parameters are passed.
#
#    Otherwise, this script just returns without doing anything.
#
# Command-line parameters:
#
```

```
#    $dmSuiteJobId        Delphix/AXIS DmSuite Job ID value
#    $dmSuiteAppName        Delphix/AXIS DmSuite Application Name
#    $dmSuiteEnvName        Delphix/AXIS DmSuite Target Environment Name
#    $dmSuiteConnName    Delphix/AXIS DmSuite Target Connector Name
#
# Environment inputs expected:
#
#    VDB_DATABASE_NAME    SQL Server database name for the VDB
#    VDB_INSTANCE_NAME    SQL Server instance name for the VDB
#    VDB_INSTANCE_PORT    SQL Server instance port number for the VDB
#    VDB_INSTANCE_HOST    SQL Server instance hostname for the VDB
#
# Note:
#
# Modifications:
#=============================================================================================
param([string]$dmSuiteJobId = "~~~"
    ,[string]$dmSuiteAppName = "~~~"
    ,[string]$dmSuiteEnvName = "~~~"
    ,[string]$dmSuiteConnName = "~~~"
)
#
#---------------------------------------------------------------------------------------------
# Set fixed variables for later use when submitting the Delphix/AXIS DmSuite job...
#---------------------------------------------------------------------------------------------
$dirPath = [Environment]::GetFolderPath("Desktop")
$dmSuiteServer = "XXXXXXXXX"
$dmSuitePort = "8282"
$dmSuiteUser = "XXXXXXXX"
$dmSuitePwd = "XXXXXXXXX"
$dmSuiteBaseURL = "http://" + $dmSuiteServer + ":" + $dmSuitePort + "/dmsuite/apiV4"
$dmSuiteWaitSecs = 5
#
#---------------------------------------------------------------------------------------------
# If no Delphix/AXIS DmSuite Job ID is provided, then this database is not to be
# masked, and so then don't do anything...
#---------------------------------------------------------------------------------------------
if ( $dmSuiteJobId -eq "~~~" -or $dmSuiteJobId -eq "0" ) {
    exit 0
}
#
#---------------------------------------------------------------------------------------------
# Otherwise, if we're going to mask this database, then first let's create a log
# file to capture output from this script and also create an XML file to store
# responses from the Delphix/AXIS DmSuite engine...
#---------------------------------------------------------------------------------------------
$timeStamp = Get-Date -UFormat "%Y%m%d_%H%M%S"
$logFile = $dirPath + "\" + $env:VDB_DATABASE_NAME + "_" + $timeStamp + "_MASK.LOG"
"logFile is " + $logFile
$xmlFile = $dirPath + "\" + $env:VDB_DATABASE_NAME + "_" + $timeStamp + "_MASK.XML"
"INFO: xmlFile = '" + $xmlFile + "'" | Out-File $logFile
$errFile = $dirPath + "\" + $env:VDB_DATABASE_NAME + "_" + $timeStamp + "_MASK.ERR"
"INFO: errFile = '" + $errFile + "'" | Out-File $logFile -Append
```

```
#
#-------------------------------------------------------------------------------
# Housekeeping: remove any existing log files older than 15 days...
#-------------------------------------------------------------------------------
"INFO: removing log files older than 15 days..." | Out-File $logFile -Append
$ageLimit = (Get-Date).AddDays(-15)
$logFilePattern = $env:VDB_DATABASE_NAME + "_*_MASK.LOG"
"INFO: logFilePattern = '" + $logFilePattern + "'" | Out-File $logFile -Append
Get-ChildItem -Path $dirPath -recurse -include $logFilePattern |
    Where-Object { !$_.PSIsContainer -and $_.CreationTime -lt $ageLimit } |
    Remove-Item
#
#-------------------------------------------------------------------------------
# Record variables and parameters to the log file...
#-------------------------------------------------------------------------------
"INFO: dirPath = '" + $dirPath + "'" | Out-File $logFile -Append
"INFO: dmSuiteJobId = '" + $dmSuiteJobId + "'" | Out-File $logFile -Append
"INFO: dmSuiteAppName = '" + $dmSuiteAppName + "'" | Out-File $logFile -Append
"INFO: dmSuiteEnvName = '" + $dmSuiteEnvName + "'" | Out-File $logFile -Append
"INFO: dmSuiteConnName = '" + $dmSuiteConnName + "'" | Out-File $logFile -Append
"INFO: env:VDB_INSTANCE_HOST = '" + $env:VDB_INSTANCE_HOST + "'" | Out-File $logFile
-Append
"INFO: env:VDB_INSTANCE_NAME = '" + $env:VDB_INSTANCE_NAME + "'" | Out-File $logFile
-Append
"INFO: env:VDB_INSTANCE_PORT = '" + $env:VDB_INSTANCE_PORT + "'" | Out-File $logFile
-Append
"INFO: env:VDB_DATABASE_NAME = '" + $env:VDB_DATABASE_NAME + "'" | Out-File $logFile
-Append
"INFO: dmSuiteServer = '" + $dmSuiteServer + "'" | Out-File $logFile -Append
"INFO: dmSuitePort = '" + $dmSuitePort + "'" | Out-File $logFile -Append
"INFO: dmSuiteUser = '" + $dmSuiteUser + "'" | Out-File $logFile -Append
"INFO: dmSuiteBaseURL = '" + $dmSuiteBaseURL + "'" | Out-File $logFile -Append
"INFO: xmlFile = '" + $xmlFile + "'" | Out-File $logFile -Append
"INFO: errFile = '" + $errFile + "'" | Out-File $logFile -Append
#
#-------------------------------------------------------------------------------
# Verify that the CURL executable is available...
#-------------------------------------------------------------------------------
$curlExe = $dirPath.ToUpper() + "\CURL.EXE"
if (-Not (test-path $curlExe )) {
    "ERROR: Executable '" + $curlExe + "' not found; aborting..." | Out-File $logFile
-Append
    Throw "ERROR: Executable '" + $curlExe + "' not found; aborting..."
}
"INFO: curlExe = '" + $curlExe + "'" | Out-File $logFile -Append
#
#-------------------------------------------------------------------------------
# Obtain the CURL authentication token for the username and encrypted password...
#-------------------------------------------------------------------------------
"INFO: obtaining auth_token for username/encyrpted-password..." | Out-File $logFile
-Append
$cmdLine = $curlExe + " -sIX GET `"" + $dmSuiteBaseURL + "/login?user=" +
$dmSuiteUser + "&password=" + $dmSuitePwd + "`" 2>&1"
```

```
"INFO: powershell command-line is '" + $cmdLine + "'" | Out-File $logFile -Append
$output = invoke-expression $cmdLine
if ( $LastExitCode -ne 0) {
    "ERROR: GET auth_token failed; aborting..." | Out-File $logFile -Append
    $output | Out-File $logFile -Append
    Throw $output
}
if ($output | where {$_.ToUpper() -match "ERROR" -or $_.ToUpper() -match "FAILURE"})
{
    "ERROR: GET auth_token returned an error; aborting..." | Out-File $logFile
-Append
    $output | Out-File $logFile -Append
    Throw $output
}
$curlAuthTokenLine = $output | Select-String -pattern auth_token
$curlAuthToken = $curlAuthTokenLine -split '\s+' | Select-Object -Last 1
if ([string]::IsNullOrEmpty($curlAuthToken)) {
    "ERROR: NULL authentication token; aborting..." | Out-File $logFile -Append
    $output | Out-File $logFile -Append
    Throw $output
}
"INFO: auth_token '" + $curlAuthToken + "' obtained" | Out-File $logFile -Append
#
#-------------------------------------------------------------------------------
# Retrieve the Delphix/AXIS DmSuite environment ID...
#-------------------------------------------------------------------------------
"INFO: retrieving DmSuite environment ID for dmSuiteAppName '" + $dmSuiteAppName +
    "..." | Out-File $logFile -Append
$cmdLine = $curlExe + " -H `"auth_token:" + $curlAuthToken + "`" `"" +
    $dmSuiteBaseURL + "/environments`" > " + $xmlFile + " 2> " + $errFile
"INFO: powershell command-line is '" + $cmdLine + "'" | Out-File $logFile -Append
$output = invoke-expression $cmdLine
if ( $LastExitCode -ne 0) {
    "ERROR: retrieving DmSuite environment ID failed; aborting..." | Out-File
$logFile -Append
    $output | Out-File $logFile -Append
    cat $xmlFile | Out-File $logFile -Append
    cat $errFile | Out-File $logFile -Append
    Remove-Item $xmlFile
    Remove-Item $errFile
    Throw $output
}
if ($output | where {$_.ToUpper() -match "ERROR" -or $_.ToUpper() -match "FAILURE"})
{
    "ERROR: retrieving DmSuite environment ID returned an error; aborting..." | Out-
File $logFile -Append
    $output | Out-File $logFile -Append
    cat $xmlFile | Out-File $logFile -Append
    cat $errFile | Out-File $logFile -Append
    Remove-Item $xmlFile
    Remove-Item $errFile
    Throw $output
}
```

```
[xml]$xml = Get-Content $xmlFile
$environmentURL = ""
$xml.SelectNodes("//Environment") | % {
    if (($_.Name -eq $dmSuiteEnvName) -and ($_.Application -eq $dmSuiteAppName)) {
        $environmentURL = $_.Link.href
    }
}
if ([string]::IsNullOrEmpty($environmentURL)) {
    "ERROR: retrieving DmSuite environment ID returned an error; aborting..." | Out-
File $logFile -Append
    cat $xmlFile | Out-File $logFile -Append
    cat $errFile | Out-File $logFile -Append
    Remove-Item $xmlFile
    Remove-Item $errFile
    Throw "ERROR: retrieving DmSuite environment ID returned an error; aborting..."
}
"INFO: environmentURL = '" + $environmentURL + "'" | Out-File $logFile -Append
Remove-item $xmlFile
Remove-Item $errFile
#
#-------------------------------------------------------------------------------
# Retrieve the Delphix/AXIS DmSuite connector ID...
#-------------------------------------------------------------------------------
"INFO: retrieving DmSuite connector ID for dmSuiteConnName '" + $dmSuiteConnName +
    "..." | Out-File $logFile -Append
$cmdLine = $curlExe + " -H `"auth_token:" + $curlAuthToken + "`" `"" +
    $dmSuiteBaseURL + "/" + $environmentURL + "/connectors`" > " + $xmlFile + " 2> "
+ $errFile
"INFO: powershell command-line is '" + $cmdLine + "'" | Out-File $logFile -Append
$output = invoke-expression $cmdLine
if ( $LastExitCode -ne 0) {
    "ERROR: retrieving DmSuite connector ID failed; aborting..." | Out-File $logFile
-Append
    $output | Out-File $logFile -Append
    cat $xmlFile | Out-File $logFile -Append
    cat $errFile | Out-File $logFile -Append
    Remove-Item $xmlFile
    Remove-item $errFile
    Throw $output
}
if ($output | where {$_.ToUpper() -match "ERROR" -or $_.ToUpper() -match "FAILURE"})
{
    "ERROR: retrieving DmSuite connector ID returned an error; aborting..." | Out-
File $logFile -Append
    $output | Out-File $logFile -Append
    cat $xmlFile | Out-File $logFile -Append
    cat $errFile | Out-File $logFile -Append
    Remove-Item $xmlFile
    Remove-item $errFile
    Throw $output
}
[xml]$xml = Get-Content $xmlFile
$connectorURL = ""
```

```
$xml.SelectNodes("//Connector") | % {
    if ( $_.Name -eq $dmSuiteConnName ) {
        $connectorURL = $_.Link.href
    }
}
if ([string]::IsNullOrEmpty($connectorURL)) {
    "ERROR: retrieving DmSuite connector ID returned an error; aborting..." | Out-
File $logFile -Append
    cat $xmlFile | Out-File $logFile -Append
    cat $errFile | Out-File $logFile -Append
    Remove-Item $xmlFile
    Remove-item $errFile
    Throw "ERROR: retrieving DmSuite connector ID returned an error; aborting..."
}
"INFO: connectorURL = '" + $connectorURL + "'" | Out-File $logFile -Append
Remove-item $xmlFile
Remove-item $errFile
#
#-------------------------------------------------------------------------------
# Start the Delphix/AXIS DmSuitemasking job...
#-------------------------------------------------------------------------------
"INFO: starting DmSuite JobID " + $dmSuiteJobId + "..." | Out-File $logFile -Append
$cmdLine = $curlExe + " -X POST -H `"auth_token:" + $curlAuthToken +
    "`" -H `"Content-Type:application/xml`" -d `"<MaskingsRequest>" +
###
### The two following lines are needed if the Delphix/AXIS DmSuite masking job is
"multitenant"...
###
###    "<Links>" + "<Link rel=`"SourceConnector`" href=`"" + $connectorURL + "`"/>" +
###    "<Link rel=`"TargetConnector`" href=`"" + $connectorURL + "`"/>" + "</Links>"
 +
    "</MaskingsRequest>`" `"" + $dmSuiteBaseURL + "/applications/" + $dmSuiteAppName
+
    "/maskingjobs/" + $dmSuiteJobId + "/run`" > " + $xmlFile + " 2> " + $errFile
"INFO: powershell command-line is '" + $cmdLine + "'" | Out-File $logFile -Append
$output = invoke-expression $cmdLine
if ( $LastExitCode -ne 0) {
    "ERROR: DmSuite job start returned failure exit code; aborting..." | Out-File
$logFile -Append
    $output | Out-File $logFile -Append
    cat $xmlFile | Out-File $logFile -Append
    cat $errFile | Out-File $logFile -Append
    Remove-Item $xmlFile
    Remove-Item $errFile
    Throw "ERROR: DmSuite job start returned failure exit code; aborting..."
}
[xml]$xml = Get-Content $xmlFile
if ($xml.MaskingsResponse.ResponseStatus.Status -ne "SUCCESS") {
    "ERROR: DmSuite job run failed; aborting..." | Out-File $logFile -Append
    $output | Out-File $logFile -Append
    cat $xmlFile | Out-File $logFile -Append
    cat $errFile | Out-File $logFile -Append
    Remove-Item $xmlFile
```

```
    Remove-item $errFile
    Throw "ERROR: DmSuite job start failed; aborting..."
}
Remove-Item $xmlFile
Remove-item $errFile
#
#-------------------------------------------------------------------------------
# Check status of the submitted Delphix/AXIS DmSuite masking job...
#-------------------------------------------------------------------------------
$dmSuiteStatus = "RUNNING"
while ( $dmSuiteStatus -match "RUNNING" ) {
    #
    Start-Sleep -s $dmSuiteWaitSecs
    $totalWaitSecs = $totalWaitSecs + $dmSuiteWaitSecs
    #
    "INFO: Checking DmSuite JobID " + $dmSuiteJobId + "status after " +
         $totalWaitSecs + " secs..." | Out-File $logFile -Append
    $cmdLine = $curlExe + " -H `"auth_token:" + $curlAuthToken +
        "`" -H `"Content-Type:application/xml`" `"" +
        $dmSuiteBaseURL + "/applications/" + $dmSuiteAppName +
        "/maskingjobs/" + $dmSuiteJobId + "/results`" > " +
        $xmlFile + " 2> " + $errFile
    "INFO: powershell command-line is '" + $cmdLine + "'" | Out-File $logFile -Append
    $output = invoke-expression $cmdLine
    if ( $LastExitCode -ne 0) {
        "ERROR: DmSuite job status returned failure exit status; aborting..." | Out-
File $logFile -Append
        $output | Out-File $logFile -Append
        cat $xmlFile | Out-File $logFile -Append
        cat $errFile | Out-File $logFile -Append
        Remove-Item $xmlFile
        Remove-Item $errFile
        Throw "ERROR: DmSuite job status returned failure exit status; aborting..."
    }
    [xml]$xml = Get-Content $xmlFile
    if ($xml.MaskingsResponse.ResponseStatus.Status -ne "SUCCESS") {
        "ERROR: DmSuite job status failed; aborting..." | Out-File $logFile -Append
        $output | Out-File $logFile -Append
        cat $xmlFile | Out-File $logFile -Append
        cat $errFile | Out-File $logFile -Append
        Remove-Item $xmlFile
        Remove-item $errFile
        Throw "ERROR: DmSuite job status failed; aborting..."
    }
    $dmSuiteStatus = $xml.MaskingsResponse.Maskings.Masking.Status
    #
}
#
#-------------------------------------------------------------------------------
# Return final exit status of completed Delphix/AXIS DmSuite masking job...
#-------------------------------------------------------------------------------
if ($dmSuiteStatus -eq "SUCCESS") {
    #
```

```
    "INFO: DmSuite JobID '" + $dmSuiteJobId + "' SUCCESS after " +
        $totalWaitSecs + " secs..." | Out-File $logFile -Append
    Remove-Item $xmlFile
    Remove-item $errFile
    exit 0
    #
} else {
    #
    cat $xmlFile | Out-File $logFile -Append
    cat $errFile | Out-File $logFile -Append
    "ERROR: DmSuite JobID '" + $dmSuiteJobId + "' failed with '" + $dmSuiteStatus +
        " after " + $totalWaitSecs + " secs running..." | Out-File $logFile -Append
    Throw "ERROR: DmSuite JobID '" + $dmSuiteJobId + "' failed with '" +
        $dmSuiteStatus + " after " + $totalWaitSecs + " secs running..."
    #
}
```

Sample post-start hook to add CDC jobs

Sample post-start hook for CDC

This script will add Change Data Capture (CDC) capture and cleanup jobs on provisioned virtual databases.

```
$VDB_NAME = $env:VDB_DATABASE_NAME

$VDB_INSTANCE = $env:VDB_INSTANCE_NAME

SQLCMD -b -d $VDB_NAME -r0 -Q "EXEC sys.sp_cdc_add_job @job_type = N'capture';EXEC
sys.sp_cdc_add_job @job_type = N'cleanup'" -U <Username> -P <Password> -S ".\
$VDB_INSTANCE"

exit $LastExitCode
```

Sample post-start hook for CDC if provisioning is done from a lower database version to SQL2016 or above

This script will add CDC capture, cleanup jobs, and upgrade CDC metadata on provisioned virtual databases.

```
$VDB_NAME = $env:VDB_DATABASE_NAME

$VDB_INSTANCE = $env:VDB_INSTANCE_NAME

SQLCMD -b -d $VDB_NAME -r0 -Q "EXEC sys.sp_cdc_add_job @job_type = N'capture';EXEC
sys.sp_cdc_add_job @job_type = N'cleanup';EXEC sys.sp_cdc_vupgrade;" -U <Username> -P
<Password> -S ".\$VDB_INSTANCE"

exit $LastExitCode
```

⚠ Place real username and password at respective placeholders while using the scripts.

Sample pre- and post-refresh hook for SQL server

How Pre- and Post-refresh hooks work

The Delphix Engine executes the Pre-Refresh hook before each refresh operation on a virtual database (VDB). It executes the Post-Refresh hook after each refresh operation. If a Configure Clone hook is also defined, then the Post-Refresh hook executes after all Configure Clone hooks.

Generally, after the initial provisioning of a VDB, changes are made to that VDB which should be saved across refreshes. Examples include resetting passwords to production accounts, adding non-production accounts, database references to other databases in production or non-production, and so on.

The Pre-Refresh hook is the first step of a process to capture the data to be saved, ending with the Post-Refresh hook to re-apply all that was captured, after the refresh has completed. So, the order of execution is:

1. Zero, one, or more **Pre-refresh** hook(s)
2. The **Refresh** operation itself on the VDB
3. Zero, one, or more **Post-refresh** hook(s)

The Windows PowerShell script (shown below) is intended to demonstrate how to call a SQL Server stored procedure, taking a single parameter for the VDB database name. In a Pre-Refresh hook, a custom-built stored procedure (named MSDB.DBO.CAPTURE_DB_SETTINGS) can be called to capture all data, and store it in (for example) the MSDB system database, or in a file on the Windows target host server. In a Post-Refresh hook, a custom-built stored procedure (named MSDB.DBO.APPLY_DB_SETTINGS) can be called to access all the data captured by the Pre-Refresh hook and re-apply it into the newly-refreshed VDB.

So, assuming that the Windows PowerShell script below is stored in a file named CALLSP.PS1 within a directory on the VDB target host named D:\DELPHIX\SCRIPTS, the call syntax within the Pre-Refresh hook might look something like this:

```
D:\Delphix\Scripts\callsp.ps1 MSDB.DBO.CAPTURE_DB_SETTINGS
```

...and the call syntax within the Post-Refresh hook might look something like this:

```
D:\Delphix\Scripts\callsp.ps1 MSDB.DBO.APPLY_DB_SETTINGS
```

Be aware that Delphix hooks set the following Windows environments from a VDB:

- VDB_DATABASE_NAME – the name of the Delphix VDB (not the SQL Server database)
- VDB_INSTANCE_NAME – the name of the SQL Server instance
- VDB_INSTANCE_PORT – the port number of the SQL Server instance
- VDB_INSTANCE_HOST – the name of the Windows host on which the SQL Server instance resides

You can access the values in these environment variables within PowerShell using the $env:*variable-name* syntax, as shown in the code below.

Sample code

Below is a sample code for the **callsp.ps1** script.

Disclaimer

This sample code is offered for illustration purposes only, and does not carry any warranty or guarantee. Employing copies of this code, in whole or in part, indicates acceptance of all risks.

```
#==============================================================================
# File:         callsp.ps1
# Type:         powershell script
# Author:       Delphix Professional Services
# Date:         02-Nov 2015
#
# Copyright and license:
#
#       Licensed under the Apache License, Version 2.0 (the "License"); you may
#       not use this file except in compliance with the License.
#
#       You may obtain a copy of the License at
#
#               http://www.apache.org/licenses/LICENSE-2.0
#
#       Unless required by applicable law or agreed to in writing, software
#       distributed under the License is distributed on an "AS IS" basis,
#       WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
#
#       See the License for the specific language governing permissions and
#       limitations under the License.
#
#       Copyright (c) 2015 by Delphix. All rights reserved.
#
# Description:
#
#    Call the appropriate stored procedure within the DBO schema in the MSDB
#    databse on behalf of the VDB. The stored procedure name the name of the
#    database as a parameter called "@DatabaseName"..
#
# Command-line parameters:
#
#    $fqSpName        fully-qualified stored procedure name
#
# Environment inputs expected:
#
#    VDB_DATABASE_NAME    SQL Server database name for the VDB
#    VDB_INSTANCE_NAME    SQL Server instance name for the VDB
#    VDB_INSTANCE_PORT    SQL Server instance port number for the VDB
#    VDB_INSTANCE_HOST    SQL Server instance hostname for the VDB
#
# Note:
#
# Modifications:
#==============================================================================
param([string]$fqSpName = "~~~")
#
#------------------------------------------------------------------------------
# Verify the "$dirPath" and "$fqSpName" command-line parameter values...
#------------------------------------------------------------------------------
if ( $fqSpName -eq "~~~" ) {
    throw "Command-line parameter 'fqSpName' not found"
```

```
}
#
#-------------------------------------------------------------------------------
# Clean up a log file to capture future output from this script...
#-------------------------------------------------------------------------------
$dirPath = [Environment]::GetFolderPath("Desktop")
$timeStamp = Get-Date -UFormat "%Y%m%d_%H%M%S"
$logFile = $dirPath + "\" + $env:VDB_DATABASE_NAME + "_" + $timeStamp + "_SP.LOG"
"logFile is " + $logFile
#
#-------------------------------------------------------------------------------
# Output the variable names and values to the log file...
#-------------------------------------------------------------------------------
"INFO: dirPath = '" + $dirPath + "'" | Out-File $logFile
"INFO: fqSpName = '" + $fqSpName + "'" | Out-File $logFile -Append
"INFO: env:VDB_INSTANCE_HOST = '" + $env:VDB_INSTANCE_HOST + "'" | Out-File $logFile
-Append
"INFO: env:VDB_INSTANCE_NAME = '" + $env:VDB_INSTANCE_NAME + "'" | Out-File $logFile
-Append
"INFO: env:VDB_INSTANCE_PORT = '" + $env:VDB_INSTANCE_PORT + "'" | Out-File $logFile
-Append
"INFO: env:VDB_DATABASE_NAME = '" + $env:VDB_DATABASE_NAME + "'" | Out-File $logFile
-Append
#
#-------------------------------------------------------------------------------
# Housekeeping: remove any existing log files older than 15 days...
#-------------------------------------------------------------------------------
"INFO: removing log files older than 15 days..." | Out-File $logFile -Append
$ageLimit = (Get-Date).AddDays(-15)
$logFilePattern = $env:VDB_DATABASE_NAME + "_*_SP.LOG"
"INFO: logFilePattern = '" + $logFilePattern + "'" | Out-File $logFile -Append
Get-ChildItem -Path $dirPath -recurse -include $logFilePattern |
    Where-Object { !$_.PSIsContainer -and $_.CreationTime -lt $ageLimit } |
    Remove-Item
#
#-------------------------------------------------------------------------
# Run the stored procedure...
#-------------------------------------------------------------------------
"INFO: Running stored procedure '" + $fqSpName + "' within database '" +
    $env:VDB_DATABASE_NAME + "'..." | Out-File $logFile -Append
try {
    "INFO: open SQL Server connection..." | Out-File $logFile -Append
    $sqlServer = $env:VDB_INSTANCE_HOST + "\" + $env:VDB_INSTANCE_NAME + ", " +
$env:VDB_INSTANCE_PORT
    "INFO: sqlServer = '" + $sqlServer + "'" | Out-File $logFile -Append
    [System.Reflection.Assembly]::LoadWithPartialName("Microsoft.SqlServer.Smo") |
Out-Null;
    $conn = New-Object System.Data.SqlClient.SqlConnection
    $conn.ConnectionString = "Server=$sqlServer; Database=MSDB; Integrated
Security=SSPI;"
    "INFO: conn.ConnectionString = '" + $conn.ConnectionString + "'" | Out-File
$logFile -Append
    $conn.Open()
```

```
    $cmd1 = New-Object System.Data.SqlClient.SqlCommand($fqSpName, $conn)
    $cmd1.CommandType = [System.Data.CommandType]::StoredProcedure
    $cmd1.Parameters.Add('@DatabaseName', $env:VDB_DATABASE_NAME) | Out-null
    "INFO: calling " + $fqSpName + ", @DatabaseName = " + $env:VDB_DATABASE_NAME |
Out-File $logFile -Append
    $exec1 = $cmd1.ExecuteReader()
    $exec1.Close()
    $conn.Close()
} catch { Throw $Error[0].Exception.Message | Out-File $logFile -Append }
#
"INFO: completed stored procedure '" + $fqSpName + "' within database '" +
    $env:VDB_DATABASE_NAME + "' successfully" | Out-File $logFile -Append
#
#-------------------------------------------------------------------------
# Exit with success status...
#-------------------------------------------------------------------------
exit 0
```

Managing policies

These topics describe creating and managing SnapSync, LogSync, Retention, and VDB Refresh policies.

- Policies for scheduled jobs
- Creating custom policies
- Policies and time zones
- Configuring retention on individual snapshots

Policies for scheduled jobs

Introduction

Creating policies is a great way to automate the management of datasets in Delphix. Whether for syncing data, taking snapshots, or refreshing virtual databases, policies ensure that data is ready when needed.

In order to create policies, navigate to the 'Manage' dropdown and select 'Policies'.

There are five categories of policies that the Delphix Engine uses in conjunction with datasets objects:

- **SnapSync** – How often snapshots of a source database are taken for a dSource.
- **VDB snapshot** – How often snapshots are taken of the virtual database (VDB).
- **Retention** – How long snapshots and log files are retained for dSources and VDBs.
- **VDB refresh** – Automatic refresh of a VDB, either with the most recent snapshot or latest Timeflow logs. The default setting for this policy is **None**.
- **Replica retention** – How long snapshots are retained on replicated namespaces for dSources and VDBs after they have been deleted on the replication source. Retention of snapshot applies as long as the database is not deleted on the source. If the database is deleted, then the next replication update will delete the database and the retained snapshots on the replication target.

> **Avoid adding conflicting policies**
> The following example case can occur if the Snapshot policy is set to NONE, this has a direct impact on the retention policy and potential VDB data growth: An engine encountered a domain0 space issue that resulted from the archive logs on a single VDB taking up 25% of the pool. That engine was set to use a 'None' Snapshot policy along with a 1-week log retention policy. Because the archive logs were needed to provision from the one and only snapshot that existed, the engine was unable to enforce the retention policy.

There can be default or custom policies for each of these categories.

Policy type	Description	User access
Default	Default policies exist at the domain level and are applied across all objects in a category. The settings of a default policy in a category can be modified, but their names cannot be changed.	• Users with Delphix Admin credentials
Custom	Custom policies allow for the creation of unique policies to fit any schedule requirements. These can be setup with varying time intervals, ranging from minutes to days.	• Users with Delphix Admin credentials • Group and object owners

Setting different policies for objects in a group

Policies applied at the group level will affect all objects in that group. To set different policies for objects in a group, apply the policies at the group level first, then apply policies at the object level.

SnapSync Policies

SnapSync policies determine how often snapshots of the source database are taken.

In the Default SnapSync policy, a snapshot is taken daily at 3:30 AM local time and will cancel if not completed within four hours. If SnapSync does not complete within this four hour period, it will resume at the next scheduled daily time until the process is complete.

Click the 'Edit' icon to change the Default SnapSync policy, or click the 'Add' icon to create a new SnapSync policy.

Policies may be configured using the provided Schedule date picker, by Interval, or by using a Quartz cron expression when selecting Custom.

Retention policy

A Retention Policy defines how long the Delphix Engine retains snapshots and log files, which are used to rewind or provision objects from past points in time. The retention time for snapshots must be equal to or longer than the retention time for logs.

To support longer retention times, more storage may need to be allocated to the Delphix Engine. The retention policy – in combination with the SnapSync policy – can have a significant impact on the performance and storage consumption of the Delphix Engine. Retention policies can be customized to retain snapshots and logs for longer periods, enabling usage to specific points further back in time.

Replica retention policy

Replica Retention policy defines how long the snapshots are retained on replicated namespaces for dSources and VDBs, after they have been deleted on the replication source.

Normally, the snapshots that have been deleted on the replication source engine are also deleted on the replication target engine. A new retention policy is introduced to provide an extended lifetime of such snapshots on the replication target. The Replica Retention Policy is targetted and defined on the replication target. This policy can be applied to an entire replica namespace or could target specific groups or dSource/VDBs within the replica namespace.

Use the replica snapshots on the replication target to provision or refresh VDBs. Point-in-time provisioning may not be possible for these snapshots, this is done to optimize the disk space usage on the replication target.

The replica retention policy can only be used to extend (**not to reduce**) the lifetime of the replica snapshots.

Extended replica snapshots can be deleted in the following cases:

- Similar to replicated snapshots, Extended replica snapshots cannot be deleted manually. They can only be removed by adjusting the policy duration so the snapshot is no longer covered.
- If the entire dSource or VDB is deleted, the extended replica snapshot will be deleted on the replication target as well.
- Oracle virtual PDBs require their CDB to be present and, if the CDB is deleted or is not replicated, then the extended replica snapshots on both the deleted CDB as well as the dependent virtual PDB will be deleted on the replication target. This can happen in a rare scenario where an Oracle virtual PDB is migrated from one CDB to another. Once the virtual PDBs snapshots that reference the old CDB are deleted, the old CDB can also be deleted. The snapshots on the virtual PDB that depend on the old CDB will not be covered by extended replica retention on the replication target.

The replica retention policy runs automatically on a schedule to cleanup expiring snapshots or whenever a replication receive job is executed.

Benefits of longer retention

With increased retention time for snapshots and logs, a longer (older) rollback period for data is allowed.

Common use cases for longer retention include:

- SOX compliance
- Frequent application changes and development
- Caution and controlled progression of data
- Reduction of project risk
- Speed of rollback or restoring to older points in time

Creating custom policies

This topic describes creating custom policies based on cron expressions for specific database objects or groups.

Custom policies are created by editing a policy associated with a database object, either during its creation or through the **Policy management** screen after it has been created. For information about creating custom policies for dSources and VDBs during the linking and provisioning processes, see the **Linking** and **Provisioning** topics listed for each data platform.

> Policies may be configured using the provided Schedule date picker, by Interval, or by using a Quartz cron expression by selecting Custom.

Procedure

1. Login to the **Delphix management** application.
2. Click **Manage**.
3. Select **Policies**.
4. Select the **tab** for the object or group for which you want to create a policy.
5. In the selected tab click the **Add** button located on the right. For example, to create a new VDB Snapshot policy:
    a. Select the **VDB snapshot** tab.
    b. Click **+ VDB snapshot**.
6. Enter **Name** for the policy.
7. Select a **Time Zone**, if this is not selected Delphix picks up the system default time.
8. Select **Timeout**. When a job is run if it does not finish in the selected time it will be killed.
9. Select a **Schedule**.
10. Select **Next**.
11. Select the object(s) to which you want to assign this policy.
12. Click **Submit**.

Policies and time zones

Policies and time zones

You can configure the SnapSync, VDB Snapshot, and VDB Refresh policies with the time zone in which the policy should be scheduled.

To edit the time zone of a policy:

1. Login to the **Delphix management application.**
2. Click **Manage**.
3. Select **Policies**.
4. Click on the **pencil** icon located next to the selected policy.
5. Select the appropriate time zone from the drop-down list.
6. Click **Submit**.

⚠ Retention and Quota policies are not schedulable and do not need a time zone.

Upgrading to version 4.2 or higher:

Prior to version 4.2, a policy operated under the time zone of the policy's target. For example, a SnapSync Policy scheduled for 4:00 am every day that targeted a dSource in Eastern Standard Time (EST) and a dSource in Pacific Standard Time (PST) fired twice a day: once at 4:00 am EST and once at 4:00 am PST.

To maintain the same behavior of the Delphix Engine after an upgrade, the upgrade process clones existing policies with these clones differing only in their time zone. After upgrading, you may notice that the names of policies change to include the time zones in which they operate.

⚠ Default policies are not cloned and always operate under the timezone of the Delphix Engine.

Example of an upgrade engine

In this example, the dSources and VDBs originally operated under either EST (America/New_York) or CST (America/Mexico_City), and new policies were created to reflect this.

Original policy	New policies
UserSnapSync	UserSnapSync (America/Mexico_City) UserSnapSync (America/New_York)
SnapshotTest	SnapshotTest (America/Mexico_City) SnapshotTest (America/New_York)
UserRefresh	UserRefresh (America/Mexico_City) UserRefresh (America/New_York)

After an upgrade, ensure that the policies are configured as expected; it may have been unclear prior to this upgrade when policies were actually firing.

Also, after upgrading to 4.2 or higher, you may consolidate/clean-up the clones and these changes will persist through future upgrades. If you go to the policy tab, and click on a policy you should see a timezone field. This timezone field is editable. So for example, if you had "VDB_SNAP (US/Arizona)" and "VDB_SNAP (America/

Phoenix)", you could delete one of the duplicates (they are both from the same time zone in this case), make sure the timezone field is set to the desired time zone and rename the remaining policy to "VDB_SNAP".

Configuring retention on individual snapshots

This topic describes adding a custom retention definition for individual snapshots. This value will override that of the policy currently assigned to the container, for example, if 'forever' is selected then the snapshot will no longer be deleted via the retention policy.

Procedure

1. Log into the **Delphix management** application.
2. Select **Resources > storage capacity**.
3. Expand the object (dsource or vdb) to modify.
4. Expand the snapshots. (it may take a few minutes for the individual snapshots to appear)
5. To configure the desired value in the 'keep until' column, click the figure in the column.
6. In the Keep For dialog either enter the **number of days** or tick **forever**.
7. Click **Save**.

Accessing the Delphix engine

The admin and sysadmin user roles

After installation, you will have two users: **sysadmin** and **admin**. Email addresses are required inputs for both accounts.

> ⚠ When engines created before 5.3.1 are upgraded to 5.3.1 or later they will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then Disabling delphix_admin.

You can create additional sysadmin and admin users, as well as other users with more restricted privileges, as described in the topics under Managing System Administrators.

User	Initial password	Abilities and duties
sysadmin	sysadmin	Typical system administration duties such as: • modifying NTP, SNMP, SMTP settings • managing storage • downloading support logs for the Delphix Engine • performing upgrades and patches. Launches the initial **Server Setup** configuration application Has access to the Command Line Interface (CLI)
admin	delphix	Manages data objects, all collectively referred to as the Delphix Engine "domain": • dSources • virtual databases (VDBs) • users • groups • related policies and resources Manages the Delphix Engine domain using either the browser-based **Engine Admin** application or the Command Line Interface (CLI).

Access methods

There are three main interfaces to a Delphix Engine:

- Web browser (http or https)
- Command Line Interface (CLI) through ssh
- Delphix Web APIs

Accessing a Delphix engine with a web browser

In order to reach the main Engine Administrator and Delphix Management application (GUI), you can use http or https:

http://<engine IP or hostname>

or

https://<engine IP or hostname>

> ⚠ In order to use your hostname, your DNS Administrator must add the hostname and IP Address to your DNS system.

The Delphix management application

By default, the GUI login screen of a Delphix Engine will log you into the Delphix Management application, as that is where a majority of the day-to-day activities will take place, such as attaching of source databases and VDB provisioning. There is a link (Setup) to the Delphix Server Setup application.

If you are already logged into the Delphix Setup application, then you can switch to the Delphix Management login by clicking “Management” in the upper-right corner of the Delphix Setup screen:

The Delphix setup GUI

If accessing the Delphix Engine to perform system administrator actions (adding storage, changing system-level parameters), you will need to switch the login screen to the Delphix Setup login by clicking **Server setup** on the default login screen.

If you are already logged into the Delphix Management GUI, you can switch to the Server Setup login screen by clicking the username in the upper-right corner of the GUI and selecting “Setup”:

Accessing a Delphix engine using the CLI

Using the Delphix CLI is useful for various situations:

- Limited bandwidth connections where a GUI is too “heavy”
- Flash is prohibited on browsers
- Users who prefer not to use a GUI
- Automation scripts

You access the Delphix CLI through an SSH session. Like the GUI, there are separate CLIs for the Engine Administrator and Delphix Sysadmin roles. The appropriate CLI is automatically selected based on the role that has been granted to you.

Using your SSH client of choice, access the Delphix Engine in a method similar to the following:

```
ssh <user>@<engine IP or hostname>
```

Where <user> is a pre-defined user on the Delphix Engine, and <engine IP> is the IP address of a Delphix Engine.

```
ssh delphix_admin@dlpxengine1
```

All Delphix functionality is available via the CLI. For further information on using the Delphix CLI, see Command Line Interface Guide.

Accessing a Delphix engine using the Delphix web API

Delphix offers a set of RESTful web service APIs with which you can administer a Delphix Engine. Using Web API calls allows you to create powerful, complex automation, often in coordination with other technologies like Jenkins,

Puppet, Chef, and Ansible. All Delphix functionality is available via the Web API. The use of the Web API is outside the scope of this document, but there is additional information in the Delphix Web Service API Guide.

IBM Db2 environments and data sources

In this section, we will describe all the different Delphix and Db2 components that are part of the Continuous Data solution for IBM Db2 database.

This section covers the following topics:

IBM Db2 overview

Introduction to Db2

Db2 for Linux, UNIX, and Windows (LUW) is a database server product developed by IBM. Db2 LUW is the "Common Server" product member of the Db2 family, designed to run on most popular operating systems.

The version numbers in Db2 are non-sequential with v11.1 and 11.5 being the two most recent releases. Specifics of Db2 versions and platforms supported on Delphix are located in the Db2 Compatibility Matrix

IBM Db2 authentication

Db2 User and group authentication is managed in a facility external to Db2 LUW, such as the operating system, a domain controller, or a Kerberos security system. This is different from other database management systems (DBMSs), such as Oracle and SQL Server, where user accounts may be defined and authenticated in the database itself, as well as in an external facility such as the operating system.

Any time a user ID and password is explicitly provided to Db2 LUW as part of an instance attachment or database connection request, Db2 attempts to authenticate that user ID and password using this external security facility. If no user ID or password is provided with the request, Db2 implicitly uses the user ID and password that were used to login to the workstation where the request originated. More information on Db2 authentication and authorization is available via IBM documentation

Delphix Db2 authentication

Delphix for Db2 requires that the staging and target hosts must already have the necessary users and authentication systems created/installed on them. Delphix will neither create users nor change database passwords as part of the provisioning process.

Db2 database level support

Delphix supports a number of Db2 database-level features. An overview of Db2 database level support is as follows:

- Support for multiple databases linking in a Single Instance, which allows Delphix Engine users to utilize an available instance on the target more efficiently.
- Support for using a customer-supplied directory for the Delphix Plugin, Db2 Mount Points, and Db2 Delphix files and logs.
- Support for the in-memory BLU feature of Db2.
- Support for Kerberos environments.
- Support for VDB provisioning of the same OS version level or one version higher than the source and staging instances.
- Intelligent handling of HADR logs.

High Availability Disaster Recovery (HADR)

The HADR feature of IBM Db2 provides a high availability solution for both partial and complete site failures. It protects against data loss by replicating data changes from a source database, called the primary, to one or more target databases, called the standby.

Delphix HADR support

HADR replication takes place at a database level, not at the instance level. Therefore, a standby instance can have multiple databases from multiple different primary servers/instances on it. If the instance ID on the Delphix standby is not the same as the instance ID on the primary, the Delphix standby instance ID must have database permissions **secadm** and **dbadm** granted to it on the primary database. These permissions and all HADR settings must be implemented on the primary database before you take the backup on the primary database.

Log transmitting

All changes that take place at the primary database server are written into log files. The individual log records within the log files are then transmitted to the secondary database server, where the recorded changes are replayed to the local copy of the database. This procedure ensures that the primary and the secondary database servers are in a synchronized state. Using two dedicated TCP/IP communication ports and a heartbeat, the primary and the standby databases track where they are processing currently, the current state of replication, and whether the standby database is up-to-date with the status of the primary database. When a log record is "closed" (still in memory, but has yet to be written to disk on the primary), it is immediately transmitted to the HADR standby database(s). Transmission of the logs to the standbys may also be time-delayed.

Multiple standby

Beginning in Db2 v10.1, the HADR feature supports multiple standby databases. This enables an advanced topology where you can deploy HADR in multiple standby mode with up to three standby databases for a single primary. One of the databases is designated as the principal HADR standby database, with the others termed as auxiliary HADR standby databases. As with the standard HADR deployment, both types of HADR standbys are synchronized with the HADR primary database through a direct TCP/IP connection. Furthermore, both types support the reads on standby feature and can be configured for time-delayed log replay. It is possible to issue a forced or unforced takeover on any standby, including the delphix auxiliary standby. However, you should never use the Delphix auxiliary standby as a primary, because this will impact Delphix performance.

Delphix HADR synchronization

The Delphix for Db2 uses the HADR capability of Db2 to synchronize data from a production Db2 database into a Delphix-controlled Db2 "standby" server. By using this mature and existing Db2 capability, the Delphix Engine is able to ingest data and keep the standby server in sync with only a minimal impact on production. The HADR connection is configured to Super-Asynchronous (SUPERASYNC) mode where log writes are considered successfully transmitted when the log records are sent from the primary database. Because the primary database does not wait for acknowledgments from the standby database, there is no delay on the primary and transactions are considered committed regardless of the state of the replication of that transaction. For further information on Delphix synchronization, see Linking a dSource from a Db2 Database: An Overview

Database partitioning feature

Database Partitioning Feature (DPF) lets you partition your database across multiple servers. Since you can add new machines and spread your database across them, this allows users to scale their database. This means more CPUs, more memory, and more disks from each of the additional machines are available for your database. DPF can be used to manage large databases for a variety of use cases including data warehousing, data mining, online analytical processing (OLAP), or online transaction processing (OLTP) workloads.

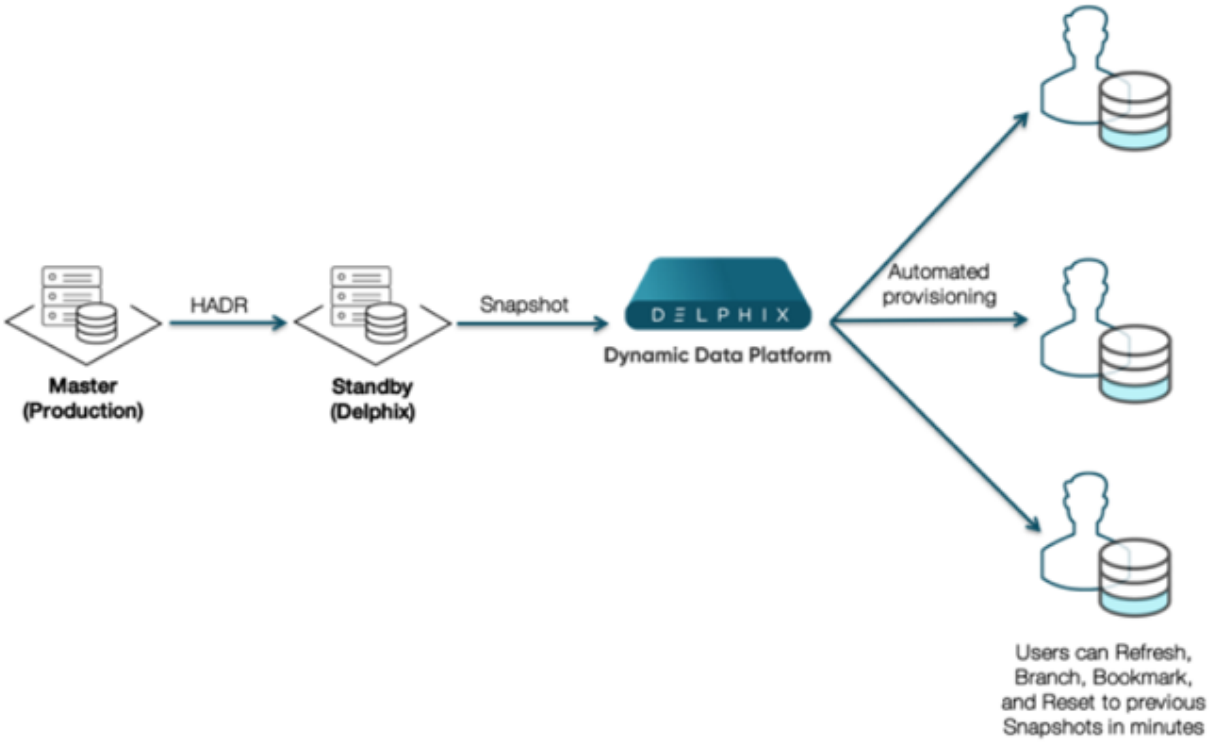
DPF enables the user to divide a database into database partitions, a database partition is a part of a database that consists of its own data, indexes, configuration files, and transaction logs. Each database partition can be configured on the different physical server having its own set of computing resources, including CPU and storage. When a query is processed, the request is divided so each database partition processes the rows that it is responsible for. DPF can maintain consistent query performance as the table grows by providing the capability to

add more processing power in the form of additional database partitions. This capability is often referred to as providing linear scalability using Db2s shared-nothing architecture.

DPF is an approach to sizing and configuring an entire database system. Please follow the recommended practices for optimal performance, reliability, and capacity growth. Please refer to IBM documentation of DPF for more details in IBM knowledge center

Delphix architecture with Db2

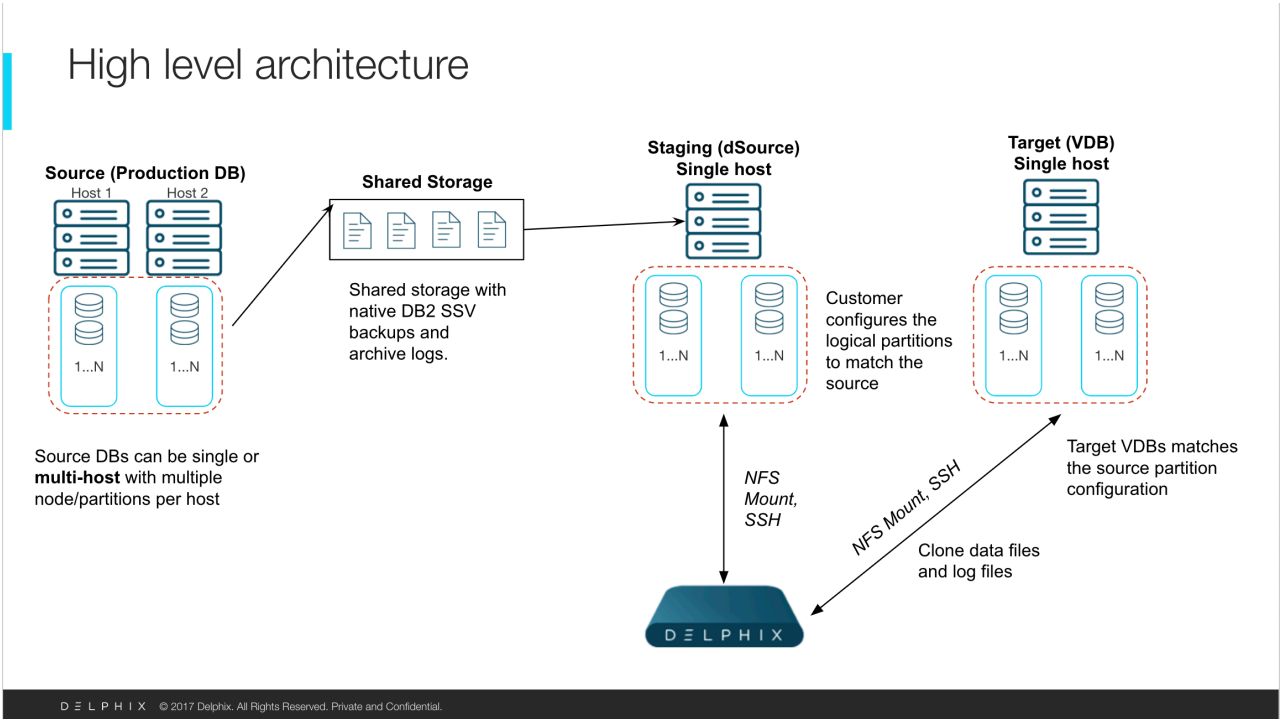
This topic describes the high-level process for adding Db2 environments, linking Db2 instances to the Delphix Engine, and provisioning virtual databases.



Delphix uses a Standby server called “Staging” to support the data ingestion process. There are multiple benefits of having a Staging server between the Db2 Source Instances and the Delphix Dynamic Data Platform:

- Allows the database instances to stay in sync with the source database, so end users can create or update their virtual databases as needed. The Delphix Engine snapshots the Staging server and the snapshots can be provisioned out to one or more target servers.
- Supports multiple ways of ingesting data, via HADR, Backups, Archive Logs, and Staging Push.
- In the case of the Staging Push feature, the user will control the restore and roll forward of the staging database.
- Prevents the end-users and their related processes to be connected to the primary database instances, making the architecture more secure and reducing primary databases' potential point of failure and performance degradation.
- In the case of Database Partitioning Feature (DPF), the staging and target environments will be single-node environments and all the portions on the source (logical and physical) will be converted to logical partitions on a single host.

Database partitioning feature architecture



The snapshot and provision process occurs on the instance level; all databases that exist on the standby server will be provisioned out to the target machines. Similarly, actions such as bookmark, rewind, and refresh will simultaneously apply to all the databases in the instance.

Db2 support and requirements

In this section, we will describe all the different Delphix and Db2 components that are part of the Continuous Data solution for IBM Db2 database.

- Network and connectivity requirements for Db2 environments
- Requirements for Db2 hosts and databases
- Sudo privilege requirements for Db2 environments
- Sudo file configuration examples for Db2 environments
- How to use dlpx_db_exec privilege elevation script

To view the Db2 compatibility matrix, see IBM Db2 matrix.

Network and connectivity requirements for Db2 environments

Overview

This topic outlines the network and connectivity requirements for the Delphix Engine and Db2 standby and target environments.

Port allocations specific to DB2

The Delphix Engine makes use of the following network ports for Db2 standby and target:

Inbound to the Delphix engine port allocation

Protocol	Port number	Use
TCP/UDP	111	Remote Procedure Call (RPC) port mapper used for NFSv3 mounts Note: RPC calls in NFSv3 use additional fixed ports for supporting services (lockd, mountd and statd) seen below.
TCP	1110	NFS Server daemon status and NFS server daemon keep-alive (client info)
TCP	2049	NFS Server daemon from vFiles to the Delphix Engine
TCP	54043	NFSv3 mount daemon
TCP	54044	NFSv3 stat daemon (lock state notification service)
TCP	54045	NFSv3 lock daemon/manager
UDP	33434 - 33464	Traceroute from standby and target hosts to the Delphix Engine (optional)

Outbound from a standby or target environment port allocation

Protocol	Port numbers	Use
TCP	873	Rsync connections used during V2P.
TCP	8415	DSP connections used for monitoring and script management. Typically DSP runs on port 8415.

Inbound to a standby or target environment port allocation

Protocol	Port numbers	Use
TCP	22	SSH connections to the target environment

HADR service ports

The HADR ports set for HADR_LOCAL_SVC and HADR_REMOTE_SVC on the Db2 Master and Standby hosts. The specific ports used at the customers' discretion and need to be specified during the linking process. It is highly recommended that these ports also be defined in the /etc/services file to ensure that they are only used by Db2 for the specified databases.

General outbound from the Delphix engine port allocation

Protocol	Port numbers	Use
TCP	25	Connection to a local SMTP server for sending email
TCP/UDP	53	Connections to local DNS servers
UDP	123	Connection to an NTP server
UDP	162	Sending SNMP TRAP messages to an SNMP Manager
TCP	443	HTTPS connections from the Delphix Engine to the Delphix Support upload server
TCP/UDP	636	Secure connections to an LDAP server
TCP	8415	Connections to a Delphix replication target. See Configuring Replication.
TCP	50001	Connections to source and target environments for network performance tests.

General inbound to the Delphix engine port allocation

Protocol	Port number	Use
TCP	22	SSH connections to the Delphix Engine
TCP	80	HTTP connections to the Delphix GUI

Protocol	Port number	Use
UDP	161	Messages from an SNMP Manager to the Delphix Engine
TCP	443	HTTPS connections to the Delphix Management Application
TCP	8415	Delphix Session Protocol connections from all DSP-based network services including Replication, SnapSync for Oracle, V2P, and the Delphix Connector.
TCP	50001	Connections from source and target environments for network performance tests via the Delphix CLI.

Firewalls and intrusion detection systems (IDS)

Production databases on source environments (for dSources) are often separated from the non-production environment by firewalls. Firewalls can add milliseconds to the latency between servers. Accordingly, for best performance, there should be no firewalls between the Delphix Engine and the virtual database (VDB) target environments. If the Delphix Engine is separated from a source environment by a firewall, the firewall must be configured to permit network connections between the Delphix Engine and the source environments for the application protocols (ports) listed above.

Intrusion detection systems (IDSs) should also be made permissive to the Delphix Engine deployment. IDSs should be made aware of the anticipated high volumes of data transfer between dSources and the Delphix Engine.

AppData port requirementsThe use of AppData requires the following ports/protocols.
Two important notes about these specifications:

1. The next release of the Delphix Engine will significantly augment the port/protocol utilization of AppData. The upcoming-only requirements have been marked with a *.
2. AppData V2P uses RSYNC to export to the target. RSYNC between the target and Delphix Engine is not required for general virtualization usage. The V2P-only requirements have been marked with a ^.

From Source to Delphix Engine	From Delphix Engine to Source	From Target to Delphix Engine	From Delphix Engine to Target
RSYNC (TCP Port 873)	RSYNC (TCP Port 873)	DSP (Default TCP Port 8415)	DSP (Default TCP Port 8415)
DSP (Default TCP Port 8415)	SSH (TCP Port 22)	NFS	SSH (TCP Port 22)
*NFS	DSP (Default TCP Port 8415)	^RSYNC (TCP Port 873)	^RSYNC (TCP Port 873)

Requirements for Db2 hosts and databases

Source Db2 hosts are servers that have Db2 binaries installed and have Db2 instances created on them. The hosts that contain the data that we wish to ingest are referred to as the source environment. Hosts with empty instances (no dbs in instance) are used as either staging or target hosts. This topic describes the requirements for creating connections between the Delphix Engine and Db2 hosts and instances.

Requirements for Db2 source hosts and instances

Each Db2 Source host must have IBM Db2 installed and at least one instance created on the machine. Depending on the way you intend to ingest data into the Staging server, you will also need to ensure that the following requirements are met:

- For HADR Ingestion: HADR settings for each database to be used with the standby server should be preset before the linking process begins as described in Linking a Db2 dSource
- For Backup and Log Ingestion, no additional requirements are needed at this point.

Requirements for Db2 staging and target hosts and instance

- The staging environment that the Delphix Engine uses must have access to an existing full backup of the source database on disk to create the first full copy. Delphix recommends using compressed backups as that will reduce storage needs and speed up ingest. Once this first full backup is available, subsequent refreshes can be accomplished via HADR or Archive Logs.
- The staging and target Db2 instances that you wish to use must already exist on the host. We can use the same instance for the dSource and VDB creation and can also have multiple VDBs on the same instance.
- The available instances on each host can be verified by going to the databases tab for the environment in question.
- Instance level configuration values such as the bufferpool value will need to be managed by the customer independent of Delphix.
- The instances used for staging and target environments must be compatible with the source Db2 instance.

Requirements for Db2 DPF staging and target hosts and instances

> It is highly recommended that the Database Partitioning Feature (DPF) for Db2 staging and target should be configured on separate hosts due to reason that the DPF environment consumes a lot of resources.

- The staging and target hosts should be configured with the same number of logical partition nodes as the source.
- The logical portion configuration should be added in $HOME/sqllib/db2nodes.cfg file where $HOME is the home directory of the instance user.
- The hostname to be used should be a fully qualified domain name.
- db2 nodes.cfg file configured for 4 logical partitions should look like:

```
0 <FQDN hostname> 0
1 <FQDN hostname> 1
2 <FQDN hostname> 2
3 <FQDN hostname> 3
```

- Each logical partition should have a corresponding port entry in /etc/services of hosts. In the example below, db2inst1 is the instance the user and port number from 60000 to 60003 are reserved for inter-partition communication.

```
DB2_db2inst1 60000/tcp
DB2_db2inst1_1 60001/tcp
DB2_db2inst1_2 60002/tcp
DB2_db2inst1_END 60003/tcp
```

- Registry variables DB2RSHCMD and DB2COMM should be updated with values as shown below:

```
SSHCMD=$(which ssh)
db2set DB2RSHCMD="$SSHCMD"
db2set DB2COMM=TCPIP
```

- After the environment is configured on staging and target, execute **db2_all echo hello** on both staging and target servers to add host key in the known_hosts file. This will also help to check for any ssh issues on these servers.
- The backup files (or log directory if using **Customer Supplied Archive Logs**) should be consistent with the partitions configured on staging.

Additional environment requirements

- There must be an operating system user (delphix_os) with these privileges:
  - Ability to login to the staging and target environment via SSH
  - Ability to run mount, umount, mkdir, and rmdir as a super-user. If the staging and target host is an AIX system, permission to run the nfso command as a super-user. See Sudo Privilege Requirements for Db2 Environments for further explanation of the commands and Sudo File Configuration Examples for Db2 Environments for examples of the /etc/sudoers file on different Outdent operating systems.
- There must be a directory on the staging and target environment where you can install the Delphix Engine Plugin. For example, /var/opt/delphix/toolkit .
  - The delphix_os user must own the toolkit path directory (__/var/opt/delphix/toolkit).
  - If the delphix_os user and instance users (responsible for the Delphix Engine operations such as linking and provisioning) are sharing any common group, then the toolkit path directory (__/var/opt/delphix/toolkit) must have permissions rwxrwx-- (0770), but you can also use more permissive settings. This allows instance users who are part of the same group as the delphix_os user's group to be able to create directories inside the toolkit path directory.
  - If the delphix_os user and instance users (responsible for the Delphix Engine operations such as linking and provisioning) are not sharing any common group, then the toolkit path directory must have -rwxrwxrwx (0777) permissions.
  - The directory should have 1.5GB of available storage: 400MB for the Plugin and 400MB for the set of logs generated by each Db2 instance that runs on the host
  - In Db2 Plugin, plugin directory space will be used as the base location for the mount point
  - When multiple engines access the same host environment, create a separate Db2 plugin directory for each engine that will discover this environment.
- When configuring staging and target environment for multiple partitions (dpf), the `ClientAliveInterval` option must be set to 0 or commented out to prevent the SSH connection from getting severed in the midst of command executions.

ⓘ For Db2 plugin version 4.0.0 and onwards, follow the steps below to change the toolkit directory.

1. Disable all the datasets related to that environment

2. The instance of the dataset must have the DIR_CACHE configuration parameter set to No, as shown below.
   db2 update dbm cfg using DIR_CACHE NO
3. Upload the new vSDK-based plugin. The plugin has upgrade logic implemented in it and will upgrade the existing Lua-based plugin.
4. Refresh the environment to allow the plugin to create plugin paths and files under the scratch path.
5. Enable all the datasets and complete the migration process.

- The Delphix Engine must be able to initiate an SSH connection to the staging and target environment
- NFS client services must be running on the staging and target environment
- Staging and Target host instances must have the same Db2 instance code page as the production instance. The instance owner must have a compatible value defined in LC_ALL or LC_CTYPE for the Db2 code page. Preferably derived from the value of LC_ALL, LC_CTYPE, or LANG of the instance owner on the production host.
- The primary environment user should share its primary group with the instance user. For example, if the delphix_os is the primary environment user which is used for environment addition and its primary group is also delphix_os then instance users (responsible for the Delphix Engine operations such as linking and provisioning) should share group delphix_os as their secondary group.

Custom configurations in Db2 plugin

DB2 plugin v3.x

The advanced config file is named **advanceConfFlag.conf** and is created under the **<Toolkit directory>** during discovery or environment refresh operation if the file does not pre-exist already. This file will have the following configurable parameters:

- **Common group (*notCommonGroupFlag):***
  - This parameter sets the permission of "mnts", "logs" and "code" directories under the **<Toolkit dir> / db2="">**location.
  - Set ***notCommonGroupFlag*** to false, if the primary group of the primary environment user (user which is used to do discovery) is shared with the instance user.
  - Set ***notCommonGroupFlag*** to true, if the primary environment user and the instance user do not share any user group.
  - Once the necessary changes are made, refresh the environment in order to apply the changes.

By default, the ***notCommonGroupFlag*** parameter is commented out in the file. This means that the plugin implicitly assumes its value as false. The valid values for this parameter are true or false.

- **Allow Source Database on Same Instance** (***allowSourceDBonSameInst***) By default, the DB2 Plugin restricts the provisioning of a VDB on a DB2 instance which contains a database with the same name as the VDB's source database For example:
  - When provisioning a VDB from a VDB then the latter is treated as the source database.
  - When provisioning a VDB from a dSource then the staging database representing the dSource is treated as the source database.

Set ***allowSourceDBonSameInst*** to true, if the user wants to provision a VDB on an instance which contains a database with the same name as the VDB's source database. The valid values for this parameter are true or false.

- Restore Pipeline limit (***restorePipelineLimit***): Db2 DPF Plugin performs restoration of all non-catalog partitions in parallel. The parameter ***restorePipelineLimit*** allows the user to configure the number of partitions that could be restored in parallel by the plugin. When the new **advanceConfFlag.conf** file is created during environment refresh or discovery operation, the default value of ***restorePipelineLimit*** is set to 10. The user can modify the value of this parameter to a desired value. The valid value for this field is any positive integer.

A default config file created during discovery or environment refresh operation:

```
# If the user is not sharing a common group between primary environment user and
instance user.

# Then the user needs to set parameter notCommonGroupFlag as true.

# notCommonGroupFlag=true

# During provision operation check if instance contains a database name which is
identical to the source DB name  used for provisioning operation.

# By default plugin will never allow creating a VDB on the instance where source
database (or other database which is identical in name with source DB) already
exists.

# If the user still wants to create a VDB on the instance which contains a database
name which is identical to the source DB name used for provisioning operation,

# then the user needs to set parameter allowSourceDBonSameInst as true.

# allowSourceDBonSameInst=true

# Limit for parallel restores

restorePipelineLimit=10
```

DB2 plugin v4.x and onwards

DB2 plugin config file is named **db2_plugin.conf** and is created under the **<Toolkit directory>/<Delphix_COMMON>/plugin/DB2_18f4ff11-b758-4bf2-9a37-719a22f5a4b8/** directory during discovery or environment refresh operation if the file does not already exist. This file has two sections represented by the headers - plugin_custom_parameters and plugin_logging_parameters.

It is important for the users to maintain two section headers in the db2_plugin.conf file.
If any of the sections get deleted, the plugin will pick the default value for the parameters under the deleted section.

Plugin custom parameters

Common group (*usersHaveCommonGroup)*

- This parameter sets the permission for the "mnts" directory located under **<Toolkit dir>/DB2** directory and for the "logs" directory located under the **<Toolkit directory>/<Delphix_COMMON>/plugin/DB2_18f4ff11-b758-4bf2-9a37-719a22f5a4b8/** directory.
- Set the ***usersHaveCommonGroup*** parameter as ***True***, if the primary group of the primary environment user (user which is used to do discovery) is shared with the instance user.
- Set the ***usersHaveCommonGroup*** parameter as ***False***, if the primary environment user and the instance user do not share any user group.
- Once the necessary changes are made, refresh the environment in order to apply the changes.

By default, the ***usersHaveCommonGroup*** parameter is set to ***True***. The valid values for this parameter are true or false.

Allow Source Database on Same Instance (*allowSourceDbOnSameInstance*)

By default, the DB2 Plugin restricts the provisioning of a VDB on a DB2 instance which contains a database with the same name as the VDB's source database For example:

- When provisioning a VDB from a VDB then the latter is treated as the source database.
- When provisioning a VDB from a dSource then the staging database representing the dSource is treated as the source database.

If the user wants to provision a VDB on an instance which contains a database with the same name as the VDB's source database, then the parameter ***allowSourceDbOnSameInstance*** must be set to true. The valid values for this parameter are true or false.

Restore pipeline limit (*restorePipelineLimit)*

Db2 DPF Plugin performs restoration of all non-catalog partitions in parallel.
The parameter ***restorePipelineLimit*** allows the user to configure the number of partitions that could be restored in parallel by the plugin. When the new **advanceConfFlag.conf** file is created during environment refresh or discovery operation, the default value of ***restorePipelineLimit*** is set to 10. The user can modify the value of this parameter to the desired value. The valid value for this field is any positive integer.

Skip archive log validation (*skipArchiveLogValidation*)

Currently, before rolling forward, the Db2 plugin checks all archive logs to assess the integrity of the archive log files and to decide if the log files are healthy and can be used when rolling forward the database. This process is repeated every time during a resync or when creating a DSource snapshot.

This option allows the user to set the plugin to skip the validation process and perform a partial validation. By default, the plugin is configured to examine all archive logs and the parameter value is set to ***False***. To be able to use this functionality, the parameter value must be set to ***True***.

Plugin logging parameters

This feature provides three options to configure and control enhanced logging by the plugin:

1. To set debug log levels.
2. To set active log file size.
3. To set log retention levels.

Delphix users can modify the above parameters defined under the plugin_logging_parameters section of this file.

Setting Log Levels (level)

Delphix users can set the logging level of the plugin-generated logs by setting the parameter ***level*** under the plugin_logging_parameters header. Higher logging levels will help to expedite debugging issues. There are two levels of logging which are: Info and Debug. The configured logging level will apply to all the objects (both dSources and VDBs) present on a staging/target host. For example; if we have 4 dSources on a host associated with a Delphix Engine then the log level will apply to all the dSources.

Log level description:

- level=INFO: This level will print only informational logs. This is the default log level.
- level=DEBUG: This level will print informational logs and debug statements.

Setting maximum size of active log file (logFileSize)

Delphix users can set the maximum size of the active log file by defining the value of the logFileSize parameter under the plugin_logging_parameters header of the db2_plugin.conf file. Once this maximum size limit is reached, the plugin will rotate the log as per the retention property described below. This parameter only takes in a positive integer value. The minimum value of this parameter is 1 MB and the maximum value is 10 MB. The default value is 1 MB.

Setting log retention levels (retention)

Delphix users can set a retention level for plugin-generated logs for each dataset (<Db name>.diag.log) using the parameter **retention** under the plugin_logging_parameters header. As per this parameter, the log files are moved (archived), renamed or deleted once they reach the value set in the logFileSize parameter. New incoming log data is directed into a new fresh file (at the same location).

By default, the value for this parameter will be set to a minimum value of 2. The user can change this value and set its value within the range 2 and 50. For example; if ***retention*** is set to 4, the plugin will have the following log files: <DB Name>.diag.log, <DB Name>.diag.log.1, <DB Name>.diag.log.2, <DB Name>.diag.log.3, <DB Name>.diag.log.4.

- File <DB Name>.diag.log is the active log file
- File <DB Name>.diag.log.1 is the most recent archive log file
- File <DB Name>.diag.log.4 is the oldest one

The DB2 plugin will ignore values for a parameter if the values do not fit the description above and will continue the operation with the default set of values for that particular parameter.

db2_plugin.conf

```
#
# The parameters must not be left padded with spaces. The plugin will otherwise not
accept the specified parameter
# values and will use the default values.
#

[plugin_custom_parameters]
#
# If the user has not shared a common group between primary environment user and
instance user.
# Then the user needs to set parameter usersHaveCommonGroup as False.
#

usersHaveCommonGroup=True

#
# During provision operations the plugin checks if the target instance contains a
database name which is identical to
# the source DB name from which the VDB is to be created. By default the plugin will
never allow creating a VDB on the
# instance where the source database already exists, or on any instance that has a
database with the same name. If the
# users still want to create a VDB on an instance which contains a database with the
same name as the source DB then
# they need to set parameter allowSourceDbOnSameInstance as True.
```

```
#

allowSourceDbOnSameInstance=False

#
# If the users want to control the number of parallel restores then they can tune the
parameter restorePipelineLimit.
# By default the value of this parameter would be 10.
#

restorePipelineLimit=10

#
# The archive logs validation process can be skipped if the end user so chooses.
Instead, a speedier technique can be used,
# in which the archive logs are only validated for the first and last logs and the
file sequences in between are confirmed.
#

skipArchiveLogValidation=False

[plugin_logging_parameters]
#
# This flag will set the debug level of plugin logs on the remote server (both
staging and target). There are two levels:
# Info
# Debug
# The above are the only valid values that can be assigned. If any other value is
assigned, the plugin will set the
# default level as Info. the string is case insensitive so info, Info and INFO are
acceptable.
#

level=INFO

#
# This parameter will set the maximum size of the active log file in MB. Once this
 limit is reached, the plugin will
# rotate the log as per the retention property defined below. The minimum value of
this parameter is 1 MB and maximum
# value is 10 MB. This parameter only takes in a positive integer value. The default
 value is 1 MB
# logFileSize=<positive integer>
#

logFileSize=1

#
# Whenever the size of <DB Name>.diag.log file exceeds the value provided by
logFileSize parameter then the plugin will
# rename the active log file to <DB Name>.diag.log.<number> and a new log file with
name <DB Name>.diag.log will be
```

```
# generated. For example, if LogRetention is set to 4, the plugin will have the
following log files: <DB Name>.diag.log,
# <DB Name>.diag.log.1, <DB Name>.diag.log.2, <DB Name>.diag.log.3, <DB
Name>.diag.log.4.
# File <DB Name>.diag.log.4 will be the oldest one.
# File <DB Name>.diag.log.1 will be the most recent archive log file.
# File <DB Name>.diag.log will be the active log file.
# The minimum and default value for retention flag is 2.
# retention=<positive integer>
#

retention=2
```

Instance user requirements

- The instance owner of each instance you wish to use within staging or a target host must be added as an environment user within the Delphix engine. See Managing Db2 Users and Instance Owners
- For HADR synced dSources the staging instance owner must be able to "read" the ingested database contents as Delphix will check the validity of the database by querying tables before each dSource snapshot.

If a container is added or deleted, the dSource will have to be resynced.

- Ensure that the following database configurations are set to default values:
  - LOGPRIMARY, LOGSECOND - default values are set by DB2 configuration advisor and shouldn't be changed. For more info, see logsecond - Number of secondary log files configuration parameter
  - LOGARCHCOMPR1, LOGARCHCOMPR2- the default values are set to OFF so that the workflows can work properly.

Database Container Requirements

- All Db2 database container types are fully supported with the exception of Db2 raw containers.
- It is recommended not to use special characters for source database container paths as this may create problems while parsing such container paths and may also result in failures during dSource and VDB creation.

Sudo privilege requirements for Db2 environments

This topic describes the rationale behind specific `sudo` privilege requirements for virtualizing Db2 Databases.

Privilege	Sources	Targets and Staging	Rationale
`mkdir/rmdir`	Not Required	**Required**	Delphix dynamically makes and removes directories under the provisioning directory during VDB operations. This privilege is optional, provided the provisioning directory permissions allow the delphix os user to make and remove directories.
`mount/umount`	Not Required	**Required**	Delphix dynamically mounts and unmounts directories under the provisioning directory during VDB operations. This privilege is required because `mount` and `umount` are typically reserved for superuser.

It is required to specify the NOPASSWD qualifier within the "sudo" configuration file, as shown here: Sudo File Configuration Examples for Db2 Environments. This ensures that the "sudo" command does not demand the entry of a password, even for the "display permissions" (i.e. "sudo -l") command.

Delphix issues "sudo -l" in some scripts to detect if the operating system user has the correct sudo privileges. If it is unable to execute this command, some actions may fail and Delphix will raise an alert suggesting it does not have the correct sudo permissions. Restricting the execution of "sudo -l" by setting “listpw=always” in the “/etc/sudoers” file when the Delphix operating system user is configured to use public key authentication will cause the Delphix operating system user to be prompted for a password which will fail certain Delphix actions. Use a less restrictive setting for listpw than "always" when the Delphix operating system user is using public-key authentication.

Sudo file configuration examples for Db2 environments

Configuring `sudo` access on AIX for Db2 source and target environments

Sudo access to `ps` on the AIX operating system is required for the detection of listeners with non-standard configurations on both source and target environments. Super-user access level is needed to determine the `TNS_ADMIN` environment variable of the user running the listener (typically **db2**, the installation owner). From `TNS_ADMIN`, the Delphix OS user **delphix_os** can derive connection parameters.

Example: AIX /etc/sudoers entries for a Delphix Source

```
Defaults:delphix_os !requiretty
delphix_os ALL=NOPASSWD:/bin/ps
```

In addition to sudo access to the `mount`, `umount`, and `ps` commands on AIX target hosts, Delphix also requires `sudo` access to `nfso`. This is required on target hosts for the Delphix Engine to monitor the NFS read-write sizes configured on the AIX system. Super-user access level is needed to run the `nfso` command.

Example: AIX /etc/sudoers File for a Delphix Target

```
Defaults:delphix_os !requiretty
delphix_os ALL=NOPASSWD: \
/usr/sbin/mount, \
/usr/sbin/umount, \
/usr/sbin/nfso, \
/usr/bin/ps
```

Configuring `sudo` access on Linux for Db2 source and target environments

On a Linux target, sudo access to `mount`, `umount`, `mkdir`, and `rmdir` is required.
Example: Linux /etc/sudoers file for a Delphix target for DB2

```
Defaults:delphix_os !requiretty
delphix_os ALL=NOPASSWD: \
/bin/mount, /bin/umount, /bin/mkdir, /bin/rmdir
```

Examples of limiting `sudo` access for the Delphix OS user

In situations where security requirements prohibit giving the Delphix user root privileges to mount, unmount, make directory, and remove directory on the global level, it is possible to configure the `sudoers` file to provide these privileges only on specific mount points or from specific Delphix Engines, as shown in these two examples.

The Delphix Engines tests its ability to run the `mount` command using `sudo` on the target environment by issuing the `sudo mount` command with no arguments. Many of the examples shown in this topic do not allow that. This causes a warning during environment discovery and monitoring, but otherwise does not cause a problem. If your VDB operations succeed, it is safe to Ignore this warning.
However, some users configure the security on the target environments to monitor `sudo` failures and lock out the offending account after some threshold. In those situations, the failure of the sudo commands might cause the **delphix_os** account to become locked. One work-around for this situation is to increase the threshold for locking out the user account. Another option is to modify `/etc/sudoers` to permit the **delphix_os** user to run `mkdir, rmdir, umount` and `mount` command without parameters.

Note that the following examples are for illustrative purposes and the sudo file configuration options are subject to change.

Example 1

This example restricts the **delphix_os** user's use of `sudo` privileges to the directory `/db2`.

Note that wildcards are allowed for the options on `mount` and `umount` because those commands expect a fixed number of arguments after the options. The option wildcard on the `mount` command also makes it possible to specify the file-system being mounted from the Delphix Engine.

However, wildcards are not acceptable on `mkdir` and `rmdir` because they can have any number of arguments after the options. For those commands, you must specify the exact options (`-p` , `-p -m 755`) used by the Delphix Engine.

Example /etc/sudoers File Configuration on the Target Environment for sudo Privileges on the VDB Mount Directory Only (Linux OS)

```
Defaults:delphix_os !requiretty
delphix_os ALL=(root) NOPASSWD: \
/bin/mount  *        /db2/*, \
/bin/umount *        /db2/*, \
/bin/umount          /db2/*, \
/bin/mkdir -p        /db2/*, \
/bin/mkdir -p -m 755 /db2/*, \
/bin/mkdir           /db2/*, \
/bin/rmdir           /db2/*
```

Example 2

This example restricts the **delphix_os** user's use of `sudo` privileges to the directory / `db2` , restricts the mount commands to a specific Delphix Engine hostname and IP, and does not allow user-specified options for the `umount` command.

This configuration is more secure, but there is a tradeoff with deployment simplicity. This approach would require a different sudo configuration for targets configured for different Delphix Engines.

A Second Example of Configuring the /etc/sudoers File on the Target Environment for Privileges on the VDB Mount Directory Only, and Allows Mounting Only from a Single Server (Linux OS)

```
Defaults:delphix_os !requiretty
delphix_os ALL=(root) NOPASSWD: \
/bin/mount           <delphix-server-name>* /db2/*, \
/bin/mount *         <delphix-server-name>* /db2/*, \
/bin/mount           <delphix-server-ip>*   /db2/*, \
/bin/mount *         <delphix-server-ip>*   /db2/*, \
/bin/mount "", \
/bin/umount          /db2/*, \
/bin/umount *        /db2/*, \
/bin/mkdir [*]       /db2/*, \
/bin/mkdir           /db2/*, \
/bin/mkdir -p        /db2/*, \
/bin/mkdir -p -m 755 /db2/*, \
/bin/rmdir           /db2/*
```

How to use dlpx_db_exec privilege elevation script

The following section describes how the Db2 Plugin works with a Kerberos environment. Customers use principal (A user in Kerberos is called a principal) to connect with DE. All operations where Plugin needs to execute commands using database users, in that case, privilege elevation script will be used.

The following example shows how the Db2 Plugin works in the Kerberos enabled environment for a customer:

1. All SSH connections to remote hosts use a single environment user that is principal in the case of a Kerberos environment.
2. All Db2 commands which need to be executed using a particular OS user (in the case of DB2 it'll be an instance user) will pass through a privilege elevation script. Commands requiring privilege elevation will be executed under a script dlpx_db_exec, with the first parameter being the user to execute as, and the remaining parameters being the command to execute. This script may be customized by the customer, but it must always return the results of the executed command and exit with the return code from the executed command.
3. When a command is invoked with dlpx_db_exec then we'll pass the instance IDs which was discovered during the discovery phase.

Implementation

1. The customer will specify the same primary environment user for the execution of all Delphix object requests.
2. All SSH connections will then be made with this user.
3. A new utility dlpx_db_exec has been added to the Plugin.
    a. The first param to this command will be the db2 instance user to execute the remaining args.
    b. This script will be customizable by the customer to use their internal utility or command to essentially sudo to the requested instance user.
4. The Db2 Plugin is updated to pass the instance ID string to the dlpx_db_exec script such that the customer can update this to use sudo (or some other custom elevation utility).
5. The Db2 Plugin scripts are updated to prefix all commands required to be executed as the instance ID with dlpx_db_exec.
6. All Db2 changes are based on top of the Db2 DB level changes.

Sample content of dlpx_db_exec script

```
#

# This script allows customization of command execution with an alternate user

# account.

# Arg $1 contains "-u<optional user account>" for the desired user under

# which database commands will be executed.

# By default this argument is ignored and the script is executed as the default

# account.

#
```

```
if [[ $1 != -u* ]]; then

   echo "Incorrect command line parameters, -u<optional user account> is required as
the first parameter"

   exit 1

fi

user_id=`echo $1 | sed -e "s/^-u//"`

echo "$user_id" >> /tmp/test.log

shift 1

echo "$user_id and $DB2_DB_NAME" >> /tmp/test.log

if [[ $user_id != "delphix_os" ]]; then

command=$(printf "%s " "$@")

sudo su - $user_id -c "cd /home/delphix_os;export DB2DBDFT=$DB2DBDFT;$command"

else

$@

fi
```

Db2 plugin installation and upgrade

The Db2 plugin is provided as a zip file on the Delphix download site. Follow the steps below to download the plugin.

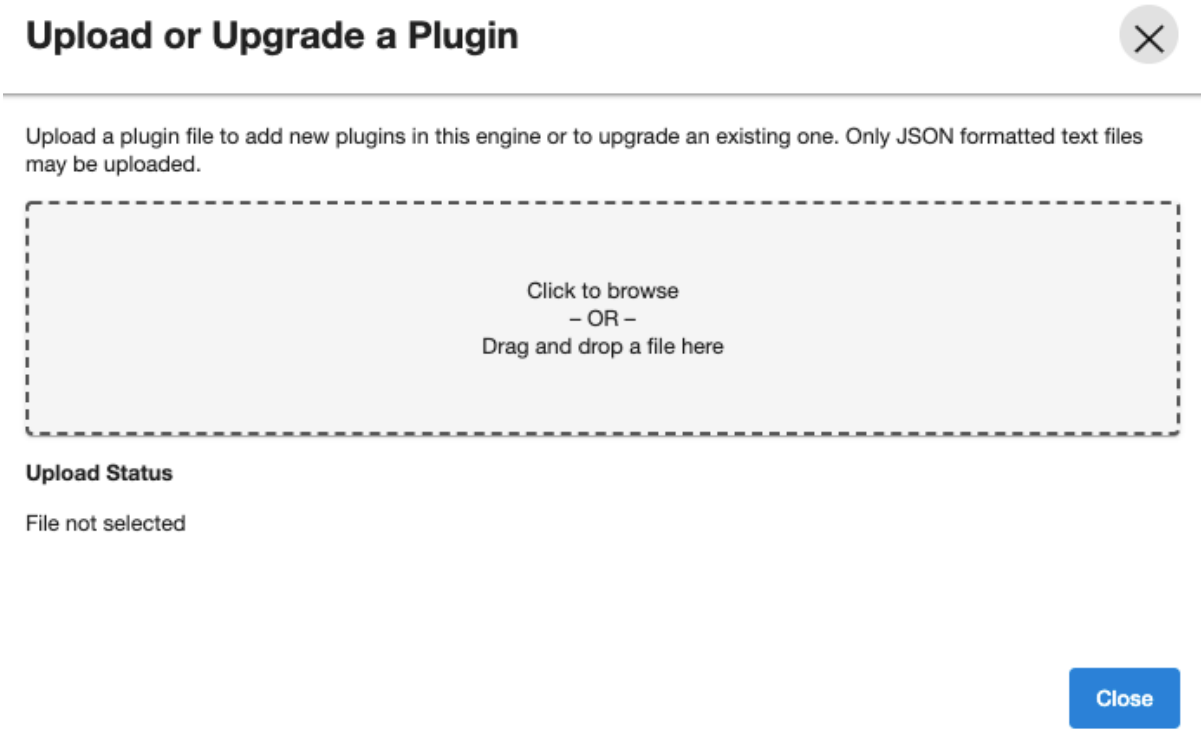
1. In the web browser, go to the Delphix download site.
2. Login to the download site using email and password credentials.
3. Navigate to the required version number of Delphix Engine.
4. Go to the Plugins > Db2 > Plugin_<version_number>.zip folder and download the zip file.

For initial installation and upgrading to a higher version refer to the section Delphix Engine Plugin Management for further details. For upgrading the Lua-based plugin to the vSDK plugin, refer to the following section.

Upgrading the Lua-based plugin to the vSDK-based plugin

The steps to upgrade from the Lua-based plugin to the new vSDK-based plugin are described below.

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. You need to disable all the datasets in the Lua environment before upgrading. Click the dSource you want to disable.
5. From the **Actions** menu (...) select **Disable**.
6. Repeat the above step to disable all the VDB datasets.
7. Select **Manage > Plugins**.
8. To upload a vSDK Plugin, click the icon. This opens the Upload or Upgrade a Plugin dialog. Upload the <file_name>.json vSDK plugin.
9. 

To start using the plugin, refresh target and source environments to re-discover repositories.
You will notice the changes in the toolkit directory structure as compared to the Lua plugin environment.
Note: This plugin upgrade operation will read the advanceConfFlag.conf and plugin-logging.properties files present in the Lua plugin environment and combines them to create db2_plugin.conf. This operation also creates the plugin_3.x_archives directory and moves all the Lua-specific files such as the logs directory, the code directory, and the configuration files (advanceConfFlag.conf and plugin-logging.properties) into this directory. The plugin archive directory is for reference purposes only. You can delete the same if not required.

10. Click **Manage**.
11. Select **Datasets**.
12. From the **Actions** menu (...) select **Enable**. This will enable the existing datasets under the new vSDK environment.
13. The upgrade is now complete. You can now check the logs for upgrade-related information.

Quick start guide for Db2

This quick start guide, which is excerpted from the larger User Guide, is intended to provide you with a quick overview of working with DB2 database objects in the Delphix Engine. It does not cover any advanced configuration options or best practices, which can have a significant impact on performance. It assumes that you are working in a Lab/Dev setting and attempting to quickly test Delphix functionality. It assumes you will use the VMware Hypervisor.

Overview

In this guide, we will walk through deploying a Delphix Engine, starting with configuring Source and Target database environments. We will then create a dSource, and provision a VDB.

For purposes of the QuickStart, you can ignore any references to Replication or Masking.

Deploy OVA on VMware

Use the Delphix-supplied OVA file to install the Delphix Engine. The OVA file is configured with many of the minimum system requirements. The underlying storage for the install is assumed to be redundant SAN storage.

1. Download the OVA file from https://download.delphix.com. You will need a support login from your sales team or a welcome letter.
    a. Navigate to the Delphix Product Releases/<Current Version>/Appliance Images page.
2. Login using the vSphere client to the vSphere server (or vCenter Server) where you want to install the Delphix Engine.
3. In the vSphere Client, click **File**.
4. Select **Deploy OVA Template**.
5. Browse to the OVA file.
6. Click **Next**.
7. Select a **hostname** for the Delphix Engine.
   This hostname will also be used in configuring the Delphix Engine network.
8. Select the **data center** where the Delphix Engine will be located.
9. Select the **cluster** and the **ESX host**.
10. Select one (1) **data store** for the **Delphix OS**. This datastore can be **thin-provisioned** and must have enough free space to accommodate the 127GB comprising the Delphix operating system.
11. Select four (4) or more **data stores** for Database Storage for the Delphix Engine. The Delphix Engine will stripe all of the Database Storage across these VMDKs, so for optimal I/O performance, each VMDK must be equal in size and be configured **Thick Provisioned - Eager Zeroed**. Additionally, these VMDKs should be distributed as evenly as possible across all four SCSI I/O controllers.
12. Select the **virtual network** you want to use.
    If using multiple physical NICs for link aggregation, you must use vSphere NIC teaming. Do not add multiple virtual NICs to the Delphix Engine itself. The Delphix Engine should use a single virtual network. For more information, see Optimal Network Architecture for the Delphix Engine.
13. Click **Finish**.
    The installation will begin and the Delphix Engine will be created in the location you specified.
14. Once the installation has completed, power on the Delphix Engine and proceed with the initial system configuration as described in Setting Up Network Access to the Delphix Engine.

> ⓘ If your source database is 4 TB, you probably need 4 TB of storage for the Delphix Engine. Add at least 4 data disks of similar size for the Delphix VM. For example: for a source database of 4 TB, create 4 VMDKs of 1 TB each.

For a full list of requirements and best practice recommendations, see Virtual Machine Requirements for VMware Platform.

Setup network access to the Delphix engine

1. Power on the Delphix Engine and open the Console.
2. Wait for the Delphix Management Service and Delphix Boot Service to come online.
   This might take up to 10 minutes during the first boot. Wait for the large orange box to turn green.
3. Press any key to access the sysadmin console.
4. Enter `sysadmin@SYSTEM` for the username and `sysadmin` for the password.
5. You will be presented with a description of available network settings and instructions for editing.

```
Delphix Engine Network Setup

To access the system setup through the browser, the system must first be
configured for networking in your
environment. From here, you can configure the primary interface, DNS, hostname,
and default route. When DHCP is
configured, all other properties are derived from DHCP settings.

To see the current settings, run "get." To change a property, run "set =." To
commit your changes,
run "commit." To exit this setup and return to the standard CLI, run "discard."

    defaultRoute    IP address of the gateway for the default route -- for
 example, "1.2.3.4."

    dhcp            Boolean value indicating whether DHCP should be used for
 the primary interface. Setting this value
                    to "true" will cause all other properties (address,
hostname, and DNS) to be derived from the DHCP
                    response

    dnsDomain       DNS Domain -- for example, "delphix.com"

    dnsServers      DNS server(s) as a list of IP addresses -- for example,
"1.2.3.4,5.6.7.8."

    hostname        Canonical system hostname, used in alert and other logs --
for example, "myserver"

    primaryAddress  Static address for the primary interface in CIDR notation
-- for example, "1.2.3.4/22"

Current settings:

    defaultRoute: 192.168.1.1
    dhcp: false
    dnsDomain: example.com
```

```
    dnsServers: 192.168.1.1
    hostname: Delphix
    primaryAddress: 192.168.1.100/24
```

6. Configure the `hostname`. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set hostname=<hostname>
```

Note : Use the same `hostname` you entered during the server installation.

7. Configure DNS. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set dnsDomain=<domain>
delphix network setup update *> set dnsServers=<server1-ip>[,<server2-ip>,...]
```

8. Configure either a static or DHCP address.
DHCP Configuration

```
delphix network setup update *> set dhcp=true
```

Static Configuration

```
delphix network setup update *> set dhcp=false
delphix network setup update *> set primaryAddress=<address>/<prefix-len>
```

Note: The static IP address must be specified in CIDR notation (for example, `192.168.1.2/24`)

9. Configure a default gateway. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set defaultRoute=<gateway-ip>
```

10. Commit your changes. Note that you can use the `get` command prior to committing to verify your desired configuration.

```
delphix network setup update *> commit
Successfully committed network settings. Further setup can be done through the
browser at:

        http://<address>

Type "exit" to disconnect, or any other commands to continue using the CLI.
```

11. Check that you can now access the Delphix Engine through a Web browser by navigating to the displayed IP address, or hostname if using DNS.
12. Exit setup.

```
delphix> exit
```

Setting up the Delphix engine

The setup procedure uses a wizard process to take you through a set of configuration screens:

- Administrators
- Time
- Network
- Network Security
- Storage
- Outbound Connectivity
- Authentication
- Network Authorization
- Registration
- Summary

1. Connect to the Delphix Engine at `http://<Delphix Engine>/login/index.html#serverSetup.`
   The **Delphix Setup** application will launch when you connect to the server.
   Enter your **sysadmin** login credentials, which initially defaults to the username **sysadmin**, with the initial default password of **sysadmin**. On first login, you will be prompted to change the initial default password.
2. Click **Next**.

Administrators

The Delphix Engine supports two types of administrators:

- System Administrator (sysadmin) - this is the engine system administrator. The sysadmin password is defined here.
- Engine Administrator (admin) - this is typically a DBA who will administer all the data managed by the engine.

On the Administrators tab, you set up the sysadmin password by entering an email address and password. The details for the admin are displayed for reference.

> ⚠ The default domain user created on Delphix Engines is now **admin** instead of delphix_admin. When engines created before 5.3.1 are upgraded to 5.3.1 or later they will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then Disabling delphix_admin.

System time

The engine time is used as the baseline for setting policies that coordinate between virtual databases and external applications.

Choose your option to set up system time in this section.

For a Quick Start, simply set the time and your timezone. You can change this later.

Network

The initial out-of-the-box network configuration in the OVA file is set to use a single VMXNET3 network adapter.

You have already configured this in the initial configuration. Delphix supports more advanced configurations, but you can enable those later.

Storage

You should see the data storage VMDKs or RDMs you created during the OVA installation. Click Next to configure these for data storage.

Serviceability

Choose your options to configure serviceability settings.

For a Quick Start, accept the defaults. You can change this later.

Authentication

Choose your options to configure authentication services.

For a Quick Start, accept the defaults. You can change this later.

Registration

If the Delphix Engine has access to the external Internet (either directly or through a web proxy), then you can auto-register the Delphix Engine:

1. Enter your **Support Username** and **Support Password**.
2. Click **Register**.

If external connectivity is not immediately available, you must perform manual registration.

1. Copy the **Delphix Engine registration code** in one of two ways:
    a. Manually highlight the registration code and copy it to clipboard. Or,
    b. Click **Copy Registration Code to Clipboard**.
2. Transfer the Delphix Engine's registration code to a workstation with access to the external network Internet. For example, you could e-mail the registration code to an externally accessible e-mail account.
3. On a machine with access to the external Internet, please use your browser to navigate to the Delphix Registration Portal at http://register.delphix.com.
4. Login with your Delphix support credentials (username and password).
5. Paste the **Registration Code**.
6. Click **Register**.

Although your Delphix Engine will work without registration, we strongly recommend that you register each Delphix Engine as part of the setup. Failing to register the Delphix Engine will impact its supportability and security in future versions.

To regenerate the registration code for a Delphix Engine please refer to, Regenerating the Delphix Engine Registration Code. Delphix strongly recommends that you regenerate this code and re-register the engine regularly to maximize the Support Security of the Delphix Engine. Delphix recommends doing this every six months.

Summary

The final summary tab will enable you to review your configurations for System Time, Network, Storage, Serviceability, and Authentication.

1. Click the **Back** button to go back and to change the configuration for any of these server settings.
2. If you are ready to proceed, then click **Submit**.
3. Click **Yes** to confirm that you want to save the configuration.
4. Click **Setup** to acknowledge the successful configuration.
5. There will be a wait of several minutes as the Delphix Engine completes the configuration.

Db2 source and target environment requirements

Source Db2 hosts are servers that have Db2 binaries installed and have Db2 instances created on them. The hosts that contain the data that we wish to ingest are referred to as the source environment. Hosts with empty instances (no dbs in instance) are used as either staging or target hosts. This topic describes the requirements for creating connections between the Delphix Engine and Db2 hosts and instances.

Requirements for Db2 source hosts and instances

Each Db2 Source host must have IBM Db2 installed and at least one instance created on the machine. Depending on the way you intend to ingest data into the Staging server, you will also need to ensure that the following requirements are met:

- For HADR Ingestion: HADR settings for each database to be used with the standby server should be preset before the linking process begins as described in Linking a Db2 dSource.
- For Backup and Log Ingestion, no additional requirements are needed at this point.

Requirements for Db2 staging and target hosts and instance

- The staging environment that the Delphix Engine uses must have access to an existing full backup of the source database on disk to create the first full copy. Delphix recommends using compressed backups as that will reduce storage needs and speed up ingest. It is recommended to test the integrity of the backup before using it for ingestion. Once this first full backup is available, subsequent refreshes can be accomplished via HADR or Archive Logs.
- The staging and target Db2 instances that you wish to use must already exist on the host. We can use the same instance for the dSource and VDB creation and can also have multiple VDBs on the same instance.
- The available instances on each host can be verified by going to the databases tab for the environment in question.
- Instance level configuration values such as the bufferpool value will need to be managed by the customer independent of Delphix.
- The instances used for staging and target environments must be compatible with the source Db2 instance.

Requirements for Db2 DPF staging and target hosts and instances

> It is highly recommended that the Database Partitioning Feature (DPF) for Db2 staging and target should be configured on separate hosts due to reason that the DPF environment consumes a lot of resources.

- The staging and target hosts should be configured with the same number of logical partition nodes as the source.
- The logical portion configuration should be added in $HOME/sqllib/db2nodes.cfg file where $HOME is the home directory of the instance user.
- The hostname to be used should be a fully qualified domain name.
- db2 nodes.cfg file configured for 4 logical partitions should look like:

```
0 <FQDN hostname> 0
1 <FQDN hostname> 1
2 <FQDN hostname> 2
3 <FQDN hostname> 3
```

- Each logical partition should have a corresponding port entry in /etc/services of hosts. In the example below, db2inst1 is the instance the user and port number from 60000 to 60003 are reserved for inter-partition communication.

```
DB2_db2inst1 60000/tcp
DB2_db2inst1_1 60001/tcp
DB2_db2inst1_2 60002/tcp
DB2_db2inst1_END 60003/tcp
```

- Registry variables DB2RSHCMD and DB2COMM should be updated with values as shown below:

```
SSHCMD=$(which ssh)
db2set DB2RSHCMD="$SSHCMD"
db2set DB2COMM=TCPIP
```

- After the environment is configured on staging and target, execute **db2_all echo hello** on both staging and target servers to add host key in the known_hosts file. This will also help to check for any ssh issues on these servers.
- The backup files (or log directory if using **Customer Supplied Archive Logs**) should be consistent with the partitions configured on staging.

Additional environment requirements

- There must be an operating system user (delphix_os) with these privileges:
  - Ability to login to the staging and target environment via SSH
  - Ability to run mount, umount, mkdir, and rmdir as a super-user. If the staging and target host is an AIX system, permission to run the nfso command as a super-user. See Sudo Privilege Requirements for Db2 Environments for further explanation of the commands and Sudo File Configuration Examples for Db2 Environments for examples of the /etc/sudoers file on different Outdent operating systems.
- There must be a directory on the staging and target environment where you can install the Delphix Engine Plugin. For example, /var/opt/delphix/toolkit .
  - The delphix_os user must own the toolkit path directory (__/var/opt/delphix/toolkit).
  - If the delphix_os user and instance users (responsible for the Delphix Engine operations such as linking and provisioning) are sharing any common group, then the toolkit path directory (__/var/opt/delphix/toolkit) must have permissions rwxrwx-- (0770), but you can also use more permissive settings. This allows instance users who are part of the same group as the delphix_os user's group to be able to create directories inside the toolkit path directory.
  - If the delphix_os user and instance users (responsible for the Delphix Engine operations such as linking and provisioning) are not sharing any common group, then the toolkit path directory must have -rwxrwxrwx (0777) permissions.
  - The directory should have 1.5GB of available storage: 400MB for the Plugin and 400MB for the set of logs generated by each Db2 instance that runs on the host
  - In Db2 Plugin, plugin directory space will be used as the base location for the mount point
  - When multiple engines access the same host environment, create a separate Db2 plugin directory for each engine that will discover this environment.
- When configuring staging and target environment for multiple partitions (dpf), the `ClientAliveInterval` option must be set to 0 or commented out to prevent the SSH connection from getting severed in the midst of command executions.

For Db2 plugin version 4.0.0 and onwards, follow the steps below to change the toolkit directory.

1. Disable all the datasets related to that environment

2. The instance of the dataset must have the DIR_CACHE configuration parameter set to No, as shown below.
   db2 update dbm cfg using DIR_CACHE NO
3. Upload the new vSDK-based plugin. The plugin has upgrade logic implemented in it and will upgrade the existing Lua-based plugin.
4. Refresh the environment to allow the plugin to create plugin paths and files under the scratch path.
5. Enable all the datasets and complete the migration process

- The Delphix Engine must be able to initiate an SSH connection to the staging and target environment
- NFS client services must be running on the staging and target environment
- Staging and Target host instances must have the same Db2 instance code page as the production instance. The instance owner must have a compatible value defined in LC_ALL or LC_CTYPE for the Db2 code page. Preferably derived from the value of LC_ALL, LC_CTYPE, or LANG of the instance owner on the production host.
- The primary environment user should share its primary group with the instance user. For example, if the delphix_os is the primary environment user which is used for environment addition and its primary group is also delphix_os then instance users (responsible for the Delphix Engine operations such as linking and provisioning) should share group delphix_os as their secondary group.

Custom configurations in Db2 plugin

DB2 plugin v3.x

The advanced config file is named **advanceConfFlag.conf** and is created under the during discovery or environment refresh operation if the file does not pre-exist already. This file will have the following configurable parameters:

- **Common Group (*notCommonGroupFlag):***
  - This parameter sets the permission of "mnts", "logs" and "code" directories under the **/db2="">**location.
  - Set ***notCommonGroupFlag*** to false, if the primary group of the primary environment user (user which is used to do discovery) is shared with the instance user.
  - Set ***notCommonGroupFlag*** to true, if the primary environment user and the instance user do not share any user group.
  - Once the necessary changes are made, refresh the environment in order to apply the changes.

By default, the ***notCommonGroupFlag*** parameter is commented out in the file. This means that the plugin implicitly assumes its value as false. The valid values for this parameter are true or false.

- **Allow Source Database on Same Instance** (***allowSourceDBonSameInst***)
  By default, the DB2 Plugin restricts the provisioning of a VDB on a DB2 instance which contains a database with the same name as the VDB's source database
  For example:
  - When provisioning a VDB from a VDB then the latter is treated as the source database.
  - When provisioning a VDB from a dSource then the staging database representing the dSource is treated as the source database.

Set ***allowSourceDBonSameInst*** to true, if the user wants to provision a VDB on an instance which contains a database with the same name as the VDB's source database. The valid values for this parameter are true or false.

- Restore Pipeline limit (***restorePipelineLimit***):
  Db2 DPF Plugin performs restoration of all non-catalog partitions in parallel.
  The parameter ***restorePipelineLimit*** allows the user to configure the number of partitions that could be restored in parallel by the plugin. When the new **advanceConfFlag.conf** file is created during environment

refresh or discovery operation, the default value of ***restorePipelineLimit*** is set to 10. The user can modify the value of this parameter to a desired value. The valid value for this field is any positive integer.

A default config file created during discovery or environment refresh operation:

```
# If the user is not sharing a common group between primary environment user and
instance user.

# Then the user needs to set parameter notCommonGroupFlag as true.

# notCommonGroupFlag=true

# During provision operation check if instance contains a database name which is
identical to the source DB name  used for provisioning operation.

# By default plugin will never allow creating a VDB on the instance where source
database (or other database which is identical in name with source DB) already
exists.

# If the user still wants to create a VDB on the instance which contains a database
name which is identical to the source DB name used for provisioning operation,

# then the user needs to set parameter allowSourceDBonSameInst as true.

# allowSourceDBonSameInst=true

# Limit for parallel restores

restorePipelineLimit=10
```

DB2 plugin v4.x and onwards

DB2 plugin config file is named **db2_plugin.conf** and is created under the **>>** directory during discovery or environment refresh operation if the file does not already exist. This file has two sections represented by the headers - plugin_custom_parameters and plugin_logging_parameters.

It is important for the users to maintain two section headers in the db2_plugin.conf file.
If any of the sections get deleted, the plugin will pick the default value for the parameters under the deleted section.

Plugin custom parameters

Common group (*usersHaveCommonGroup)*

- This parameter sets the permission for the "mnts" directory located under the **/db2="">**directory and for the "logs" directory located under the **<delphix_common> plugin/db2_18f4ff11-b758-4bf2-9a37-719a22f5a4b8/=""></delphix_common>>**directory.
- Set the ***usersHaveCommonGroup*** parameter as ***True***, if the primary group of the primary environment user (user which is used to do discovery) is shared with the instance user.
- Set the ***usersHaveCommonGroup*** parameter as ***False***, if the primary environment user and the instance user do not share any user group.
- Once the necessary changes are made, refresh the environment in order to apply the changes.

By default, the ***usersHaveCommonGroup*** parameter is set to ***True***. The valid values for this parameter are true or false.

Allow source database on same instance (*allowSourceDbOnSameInstance*)

By default, the DB2 Plugin restricts the provisioning of a VDB on a DB2 instance which contains a database with the same name as the VDB's source database
For example:

- When provisioning a VDB from a VDB then the latter is treated as the source database.
- When provisioning a VDB from a dSource then the staging database representing the dSource is treated as the source database.

If the user wants to provision a VDB on an instance which contains a database with the same name as the VDB's source database, then the parameter ***allowSourceDbOnSameInstance*** must be set to true. The valid values for this parameter are true or false.

Restore pipeline limit (*restorePipelineLimit)*

Db2 DPF Plugin performs restoration of all non-catalog partitions in parallel.
The parameter ***restorePipelineLimit*** allows the user to configure the number of partitions that could be restored in parallel by the plugin. When the new **advanceConfFlag.conf** file is created during environment refresh or discovery operation, the default value of ***restorePipelineLimit*** is set to 10. The user can modify the value of this parameter to the desired value. The valid value for this field is any positive integer.

Plugin logging parameters

This feature provides three options to configure and control enhanced logging by the plugin:

1. To set debug log levels.
2. To set active log file size.
3. To set log retention levels.

Delphix users can modify the above parameters defined under the plugin_logging_parameters section of this file.

Setting log levels (level)

Delphix users can set the logging level of the plugin-generated logs by setting the parameter ***level*** under the plugin_logging_parameters header. Higher logging levels will help to expedite debugging issues. There are two levels of logging which are: Info and Debug. The configured logging level will apply to all the objects (both dSources and VDBs) present on a staging/target host. For example; if we have 4 dSources on a host associated with a Delphix Engine then the log level will apply to all the dSources.

Log level description:

- level=INFO: This level will print only informational logs. This is the default log level.
- level=DEBUG: This level will print informational logs and debug statements.

Setting maximum size of active log file (logFileSize)

Delphix users can set the maximum size of the active log file by defining the value of the logFileSize parameter under the plugin_logging_parameters header of the db2_plugin.conf file. Once this maximum size limit is reached, the plugin will rotate the log as per the retention property described below. This parameter only takes in a positive integer value. The minimum value of this parameter is 1 MB and the maximum value is 10 MB. The default value is 1 MB.

Setting log retention levels (retention)

Delphix users can set a retention level for plugin-generated logs for each dataset (.diag.log)="" using="" the="" parameter="">***retention*** under the plugin_logging_parameters header. As per this parameter, the log files are moved (archived), renamed or deleted once they reach the value set in the logFileSize parameter. New incoming log data is directed into a new fresh file (at the same location).

By default, the value for this parameter will be set to a minimum value of 2. The user can change this value and set its value within the range 2 and 50. For example; if retention is set to 4, the plugin will have the following log files: <DB Name>.diag.log, <DB Name>.diag.log.1, <DB Name>.diag.log.2, <DB Name>.diag.log.3, <DB Name>.diag.log.4.

- File <DB Name>.diag.log is the active log file
- File <DB Name>.diag.log.1 is the most recent archive log file
- File <DB Name>.diag.log.4 is the oldest one

The DB2 plugin will ignore values for a parameter if the values do not fit the description above and will continue the operation with the default set of values for that particular parameter.

db2_plugin.conf

```
#
# The parameters must not be left padded with spaces. The plugin will otherwise not
accept the specified parameter
# values and will use the default values.
#

[plugin_custom_parameters]
#
# If the user has not shared a common group between primary environment user and
instance user.
# Then the user needs to set parameter usersHaveCommonGroup as False.
#

usersHaveCommonGroup=True

#
# During provision operations the plugin checks if the target instance contains a
database name which is identical to
# the source DB name from which the VDB is to be created. By default the plugin will
never allow creating a VDB on the
# instance where the source database already exists, or on any instance that has a
database with the same name. If the
# users still want to create a VDB on an instance which contains a database with the
same name as the source DB then
# they need to set parameter allowSourceDbOnSameInstance as True.
#

allowSourceDbOnSameInstance=False

#
# If the users want to control the number of parallel restores then they can tune the
parameter restorePipelineLimit.
# By default the value of this parameter would be 10.
#
```

```
restorePipelineLimit=10

[plugin_logging_parameters]
#
# This flag will set the debug level of plugin logs on the remote server (both
staging and target). There are two levels:
# Info
# Debug
# The above are the only valid values that can be assigned. If any other value is
assigned, the plugin will set the
# default level as Info. the string is case insensitive so info, Info and INFO are
acceptable.
#

level=INFO

#
# This parameter will set the maximum size of the active log file in MB. Once this
 limit is reached, the plugin will
# rotate the log as per the retention property defined below. The minimum value of
this parameter is 1 MB and maximum
# value is 10 MB. This parameter only takes in a positive integer value. The default
 value is 1 MB
# logFileSize=<positive integer>
#

logFileSize=1

#
# Whenever the size of <DB Name>.diag.log file exceeds the value provided by
logFileSize parameter then the plugin will
# rename the active log file to <DB Name>.diag.log.<number> and a new log file with
name <DB Name>.diag.log will be
# generated. For example, if LogRetention is set to 4, the plugin will have the
following log files: <DB Name>.diag.log,
# <DB Name>.diag.log.1, <DB Name>.diag.log.2, <DB Name>.diag.log.3, <DB
Name>.diag.log.4.
# File <DB Name>.diag.log.4 will be the oldest one.
# File <DB Name>.diag.log.1 will be the most recent archive log file.
# File <DB Name>.diag.log will be the active log file.
# The minimum and default value for retention flag is 2.
# retention=<positive integer>
#

retention=2
```

Instance user requirements

- The instance owner of each instance you wish to use within staging or a target host must be added as an environment user within the Delphix engine. See Managing Db2 Users and Instance Owners.

- For HADR synced dSources the staging instance owner must be able to "read" the ingested database contents as Delphix will check the validity of the database by querying tables before each dSource snapshot. **Note:** If a container is added or deleted, the dSource will have to be resynced.
- Ensure that the following database configurations are set to default values:
    - LOGPRIMARY, LOGSECOND - default values are set by DB2 configuration advisor and shouldn't be changed. For more info, see logsecond - Number of secondary log files configuration parameter.
    - LOGARCHCOMPR1, LOGARCHCOMPR2- the default values are set to OFF so that the workflows can work properly.

Database container requirements

- All Db2 database container types are fully supported with the exception of Db2 raw containers.
- It is recommended not to use special characters for source database container paths as this may create problems while parsing such container paths and may also result in failures during dSource and VDB creation.

Sudo privilege requirements for Db2 environments

This topic describes the rationale behind specific `sudo` privilege requirements for virtualizing Db2 Databases.

Privilege	Sources	Targets and staging	Rationale
`mkdir/rmdir`	Not Required	**Required**	Delphix dynamically makes and removes directories under the provisioning directory during VDB operations. This privilege is optional, provided the provisioning directory permissions allow the delphix os user to make and remove directories.
`mount/umount`	Not Required	**Required**	Delphix dynamically mounts and unmounts directories under the provisioning directory during VDB operations. This privilege is required because `mount` and `umount` are typically reserved for superuser.

It is required to specify the NOPASSWD qualifier within the "sudo" configuration file, as shown here: Sudo File Configuration Examples for Db2 Environments. This ensures that the "sudo" command does not demand the entry of a password, even for the "display permissions" (i.e. "sudo -l") command.

Delphix issues "sudo -l" in some scripts to detect if the operating system user has the correct sudo privileges. If it is unable to execute this command, some actions may fail and Delphix will raise an alert suggesting it does not have the correct sudo permissions. Restricting the execution of "sudo -l" by setting "listpw=always" in the "/etc/sudoers" file when the Delphix operating system user is configured to use

public key authentication will cause the Delphix operating system user to be prompted for a password which will fail certain Delphix actions. Use a less restrictive setting for listpw than "always" when the Delphix operating system user is using public-key authentication.

Adding a Db2 environment

Prerequisites

- Make sure that the staging environment in question meets the requirements described in Requirements for Db2 Hosts and Databases.

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Next to **Environments**, click the **Actions (...) menu and** select **Add Environment**.
5. In the Add Environment dialog, select **Unix/Linux** in the menu.
6. Select **Standalone Host**.
7. Click **Next**.
8. Enter a **Name** for the environment.Enter the **Host IP address** or **hostname**.
9. Enter the **SSH port**.
   The default value is 22.
10. Enter an **OS Username** for the environment.
    For more information about the environment user requirements, see Requirements for Db2 Hosts and Databases.
11. Select Login Type.
    — Username and Password - enter the OS username and password
    — Username and Public Key - enter the OS username.
    — Password Vault - select from an existing Enterprise Password Vault
    Note: Currently, the Enterprise Password Vault option is not supported.
    Info: Using Public Key Authentication
    If you want to use public-key authentication for logging into your Unix-based environment, there are two options: use the engine's key pair or provide a key pair for this environment.
    To use the engine's key pair:
    a. Select **Public Key** for the **Login Type.**
    b. Click **View Public Key**.
    c. Copy the public key that is displayed, and append it to the end of your `~/.ssh/authorized_keys` file. If this file does not exist, you will need to create it.
        i. Run chmod 600 ~/.ssh/authorized_keys to allow only the file's owner to read and write to it (make sure the file is owned by the user).
        ii. Run chmod 755 ~ to restrict access to the user's home directory so no other user may write to it.
        iii. Run chmod 700 ~/.ssh so that others cannot write to it. The ~/.ssh directory cannot be writable by group or other users. Otherwise, authentication will fail.
    As an alternative, you can provide a key pair specific for this environment via the API or CLI. See Option 2 in this CLI Cookbook article for instructions.
12. For Password, enter the password associated with the user in step 11.
13. Using Public-Key Authentication:

   a. Run chmod 600 ~/.ssh/authorized_keys to allow only the file's owner to readand write privileges to it (make sure the file is owned by the user).
   b. Run chmod 755 ~ to restrict access to the user's home directory so no other user may write to it.
   c. Run chmod 700 ~/.ssh so that others cannot write to it. The ~/.ssh directory cannot be writable by group or other users. Otherwise, authentication will fail.
14. You can also add public key authentication to an environment user's profile by using the command-line interface, as explained in the topic CLI Cookbook: Setting Up SSH Key Authentication for UNIX Environment Users.
15. For Password Login, click Verify Credentials to test the username and password.
16. Enter a **Toolkit Path** (make sure the toolkit path does not have spaces).
    For more information about the toolkit directory requirements, see Requirements for Db2 Hosts and Databases.
17. To provide your own Oracle Java select the **Provide my own JDK** checkbox and click **Next**.
18. In the Java Development Kit tab enter the absolute path to your Oracle JDK and click **Next**.
19. Click **Submit**.

As the new environment is added, you will see two jobs running in the Delphix platform Job History, one to Create and Discover an environment, and another to Create an environment. When the jobs are complete, you will see the new environment added to the list in the Environments tab. If you do not see it, click the Refresh icon in your browser.

Managing DB instances

When you add an environment with the Delphix Management Application, all existing DB2 instances on the host are automatically discovered by Delphix. A list of all instances and databases are available to Delphix based on the environment discovery process.

View instances

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. In the Environments panel, click on the **name of the environment** you want to refresh.
5. Select the **Databases** tab to see a list of all DB2 instances found in the environment.

Linking a data source (dSource)

Prerequisites

- Be sure that the source and staging instances meet the host requirements and the databases meet the container requirements described in Requirements for Db2 Hosts and Databases.

Source database preparation

- **Instance Owner Permissions**
  Delphix uses the Db2 instance owner account on the dSource for multiple operations, including verifying the data inside the databases. In order to ingest a database on the staging server with different instances, we need permissions on the source database in order to perform a restore on the staging server. As an example, if we have an instance named "auto1051" at the source with a database name "delphix" and we want to create a dSource on the "auto1052" instance on the staging server, then you must explicitly grant DBADM and SECADM to the dSource instance "auto1052" on the source instance using the following steps:
  a. Connect to the source database as the source instance owner.
     i. *connect to <DB_NAME> user <SOURCE_INSTANCE_OWNER>*
  b. Issue database grant command

        i. *grant DBADM, SECADM on the database to user <DSOURCE_INSTANCE_OWNER>*
        ii. *grant DBADM, SECADM on the database to user <VDB_INSTANCE_OWNER>*
    c. Repeat step 2 for every database to be included in the dSource, on the corresponding source database.

Determine if your dSource will be a non-HADR instance, a HADR single standby instance, or a HADR multiple standby instance. Non-HADR dSources can only be updated via a full dSource resync from a newer backup file

Non-HADR Database

1. See the "InstanceOwner Permissions" section above.
2. Ensure that the source database has the necessary user permissions for the provisioned VDBs as described in Database Permissions for Provisioned Db2 VDBs.

HADR single standby database

1. All items in Non-HADR Database section above.
2. The following database configuration settings must be set:
    a. *update db cfg for <DB_NAME> using HADR_LOCAL_HOST <PRIMARY_IP> HADR_LOCAL_SVC <PRIMARY_PORT > immediate*
    b. *update db cfg for <DB_NAME> using HADR_REMOTE_HOST <STANDBY_IP> HADR_REMOTE_SVC <STANDBY_PORT> immediate*
    c. *update db cfg for <DB_NAME> using HADR_REMOTE_INST <STANDBY_INSTANCE_NAME> immediate*
    d. *update db cfg for <DB_NAME> using HADR_SYNCMODE SUPERASYNC immediate*
3. If database configuration parameter LOGINDEXBUILD is set to OFF, do the following:
    a. *update db cfg for <DB_NAME> using LOGINDEXBUILD ON*
    b. Force off all connections to the database and reactivate the database
4. If database configuration parameter LOGARCHMETH1 is set to OFF, do the following:
    a. *update db cfg for <DB_NAME> using LOGARCHMETH1 XXXX (must be a valid log archiving method)*
    b. Take an offline backup
5. If LOGARCHMETH1 points to a third-party backup server (i.e. TSM or Netbackup) define LOGARCHMETH2 to disk
    a. *update db cfg for <DB_NAME> using LOGARCHMETH2 DISK:<full path to archive log directory>*
        i. Log files in the directory must be available from the time of the backup until the restore has successfully completed on the dSource.
6. *db2 start hadr on db <DB_NAME> as primary by force*
7. Take a full online backup as defined in the "Backup Source Database" section below.
8. Record the following information, as it must be entered on the Delphix Engine while creating the dSource.
    a. HADR Primary hostname
    b. HADR Primary SVC
    c. HADR Standby SVC (auxiliary standby port)

HADR multiple standby databases

This **assumes** a single standby database HADR setup already exists. The existing standby will be referred to as the main standby. The new delphix standby will be referred to as the auxiliary standby.

1. The following database configuration settings must be set on the primary database:
    a. *update db cfg for <DB_NAME> using HADR_SYNCMODE <SYNC MODE> immediate – set whichever sync mode you wish to use on your main standby.*
    b. *update db cfg for <DB_NAME> using HADR_TARGET_LIST "<MAIN_STANDBY_IP:MAIN_STANDBY_PORT|AUXILIARY_STANDBY_IP:AUXILIARY_STANDBY_PORT>" immediate*
        i. You may have up to two auxiliary standbys defined separated by a '|'; one of which must be the delphix dSource.
2. stop hadr on db <DB_NAME>
3. *sart hadr on db <DB_NAME> as primary by force*

4. Take a full online backup as defined in the "Backup Source Database" section below. While this backup is running, you may continue with step 5.
5. The following database configuration settings must be set on the existing main standby database:
    a. *update db cfg for <DB_NAME> using HADR_SYNCMODE <same mode as defined in 1.a above.> – It must be the same value used for the primary database.*
    b. *update db cfg for <DB_NAME> using HADR_TARGET_LIST "<PRIMARY_IP:PRIMARY_PORT|MAIN_STANDBY_IP:MAIN_STANDBY_PORT>"*
6. *stop hadr on db <DB_NAME>*
7. *start hadr on db <DB_NAME> as standby*
8. Record the following information, as it must be entered on the Delphix Engine while creating the dSource (the auxiliary standby database):
    a. HADR Primary hostname
    b. HADR Primary SVC
    c. HADR Standby SVC (auxiliary standby port)
    d. HADR_TARGET_LIST <PRIMARY_IP:PRIMARY_PORT|MAIN_STANDBY_IP:MAIN_STANDBY_PORT>

Delphix recovery with customer supplied archive logs

The Delphix Recovery with Customer Supplied Archive Logs (formerly known as Delphix Backup and Logs ingestion), feature allows users to provide Delphix with the location of the source database archive logs. Full backups only have to be restored during dSource creation, or during RESYNC operations when there are exceptions, for example, lost/corrupt log files. Archived logs get applied during the snapshot operation.

A full backup is required to create a dSource and prior to the snapshot operation, users can provide the archive logs to update the dSource during the snapshot operation, user-provided logs will be applied to the staging database. Applying the logs on a regular basis allows users to keep their dSource in sync with their source database. The staging database will always be in a roll forward pending state. The Plugin performs validation before applying the logs. Plugin managed Pre and Post Hooks can be used for preparing and purging the full backups and archive logs. A list of logs that are available for purging is stored in the snapshot metadata of the dSource. In case DPF is enabled, users need to place archive logs inside a folder with a name as NODE<Partition number> where <Partition number> is a four-digit number. For example, if the source environment has two partitions, then the user-provided log path will have folder names as NODE0000 and NODE0001. Both these folders will have respective archive logs. Snapshot operation will be used to apply the archive logs.

HADR and non-HADR configured dSources can be moved to this feature by performing a RESYNC operation.

Delphix supports Customer Supplied Archive Logs with Db2 version 10.5 and 11.1. To use Customer Supplied Archive Logs simply check the checkbox provided during dSource creation. Customer Supplied Archive Logs require that:

- A full backup must be accessible on the staging server, it could be a shared location but must be mounted on the staging host and visible as a file system.
- Archived logs must also be accessible on the staging server
    - The following command can be used to generate archive logs from the source database;
        - db2 archive log for database <database name>
    - Instance users must have read permissions on logs so that Delphix-managed recover can apply the logs.
    - Instance users must have read permissions on logs and additional write permissions on customer-provided log location if Delphix is planning to use customer-provided log location for applying the logs.
    - If the user has placed the archive logs at a user-provided log location, then the logs must be valid and the first active log of the staging database must be present in the user-provided log location.
- If database configuration parameter LOGINDEXBUILD is set to OFF, do the following:
    - update db cfg for <DB_NAME> using LOGINDEXBUILD ON
    - Force off all connections to the database and reactivate the database

- The customer does log purging from the provided log location. Delphix only provides the list of logs that are eligible for purging.

Limitations:

- Continuous updates to the dSource are not available. The DB_SYNC operations apply new logs from the log location and keep the dSource updated with changes on the source.
- Read access to the staging database is not available.

Staging push

With the "Staging Push" feature, you can now manage the restore and rollforward operations from the CLI. During dSource creation (or RESYNC), the plugin will create an empty mount point and will provide a restore and rollforward template that will allow you to restore the database. You can restore the dSource using the steps mentioned in the readme or through any other method of their choice. You can also configure the dSource with HADR by following the steps provided in the "HADR Single/Multiple Standby Database" sections above.

Once the dSource is ready, you can perform restore and rollforward operations from the CLI. After the restore, the staging database should remain in the rollforward pending state, and prior to the next snapshot, the user can do a rollforward to a specific Point-in-Time (by providing a timestamp to the rollforward command) against the available archive logs. In the case of HADR dSource, you will have to stop HADR before taking a snapshot and starting it back on after the snapshot operation is complete.

You can create a pre-snapshot and post-snapshot hook to perform HADR stop and start operations on the staging database respectively so that you don't have to connect to the staging host every time a snapshot is captured.

Prior to the snapshot operation, the user should validate the backups and the archive logs before applying them to the staging database. Below are the user side validations:

User side validation prior to snapshot operation

- The user must validate the backup files using ***db2ckbkp*** utility on the staging host.
- The user must validate the archive logs using ***db2cklog*** utility on the staging host.
- The authenticity of archive logs would be managed by the user.
- The user has to provide the first active log at the staging database log directory location after applying the logs.
- If there is a scenario where a user performed some load copy (non-logged transactions) operations on the source side then it’s the responsibility of the user to make those non-logged transactions available on the staging database.

Backup source database

New feature: Source database with Raw DEVICE type storage

Several users use raw device-based tablespaces for source Db2 databases. To leverage these environments with Delphix, Delphix has built a workflow using Db2s native tools that allow Delphix to discover and convert a raw device-based tablespace into an automatic storage-based tablespace during ingestion. Once the data is ingested into staging, customers will be able to provision VDBs of the automatic storage-based database.

In order to complete the linking process, the Standby dSource must have access to a full backup of the source Db2 databases on disk. This should be a compressed online Db2 backup and must be accessible to the dSource instance owner on disk. Delphix is currently not set up to accept Db2 backups taken using third-party sources such as Netbackup or TSM. All three features of data ingestion, namely HADR, Non-HADR, and Customer Supplied Archive Logs backups must also include logs. Starting with the Db2 plugin version 3.1.1, we are also supporting full online backups, which are taken using the "exclude logs" syntax. This support is only applicable to the dSources created using the Customer Supplied Archive Logs (Backup and Logs) ingestion mechanism on a single partition.

Support for named pipe type DB2 backup

Users can also provide Db2 backups of type named-pipe. Db2 plugin will check the type (type will be either named-pipe or a file) of backup and will use the restore syntax accordingly.

Example backup command: db2 backup database <DB_NAME> online compress include logs.

Backup types supported in DPF

The type of backups supported by Db2 plugin when used with DPF are:

- SSV backups through backup files or named pipes.
- Non-SSV backups through backup files or named pipes.

Example backup command for DB2 SSV backups: db2_all "db2 BACKUP DATABASE<DB_NAME> ON all dbpartitionnums online to <Backup File Location> compress include logs"
Example backup command for DB2 Non SSV backups: db2_all "db2 backup database <DB_NAME> online to <Backup File Location> compress include logs"

Best practices for taking a backup

The following best practices can help improve backup and restore performance:

1. Compression should be enabled
2. Following parameters should be optimally configured:
    a. Utility Heap Size (UTIL_HEAP_SZ)
    b. No. of CPUs
    c. No. of Table Spaces
    d. Extent Size
    e. Page Size
3. Parallelism & Buffer configuration may be used to improve backup performance. Parameters that should be configured are:
    a. Parallelism
    b. Buffer Size
    c. No. of Buffers

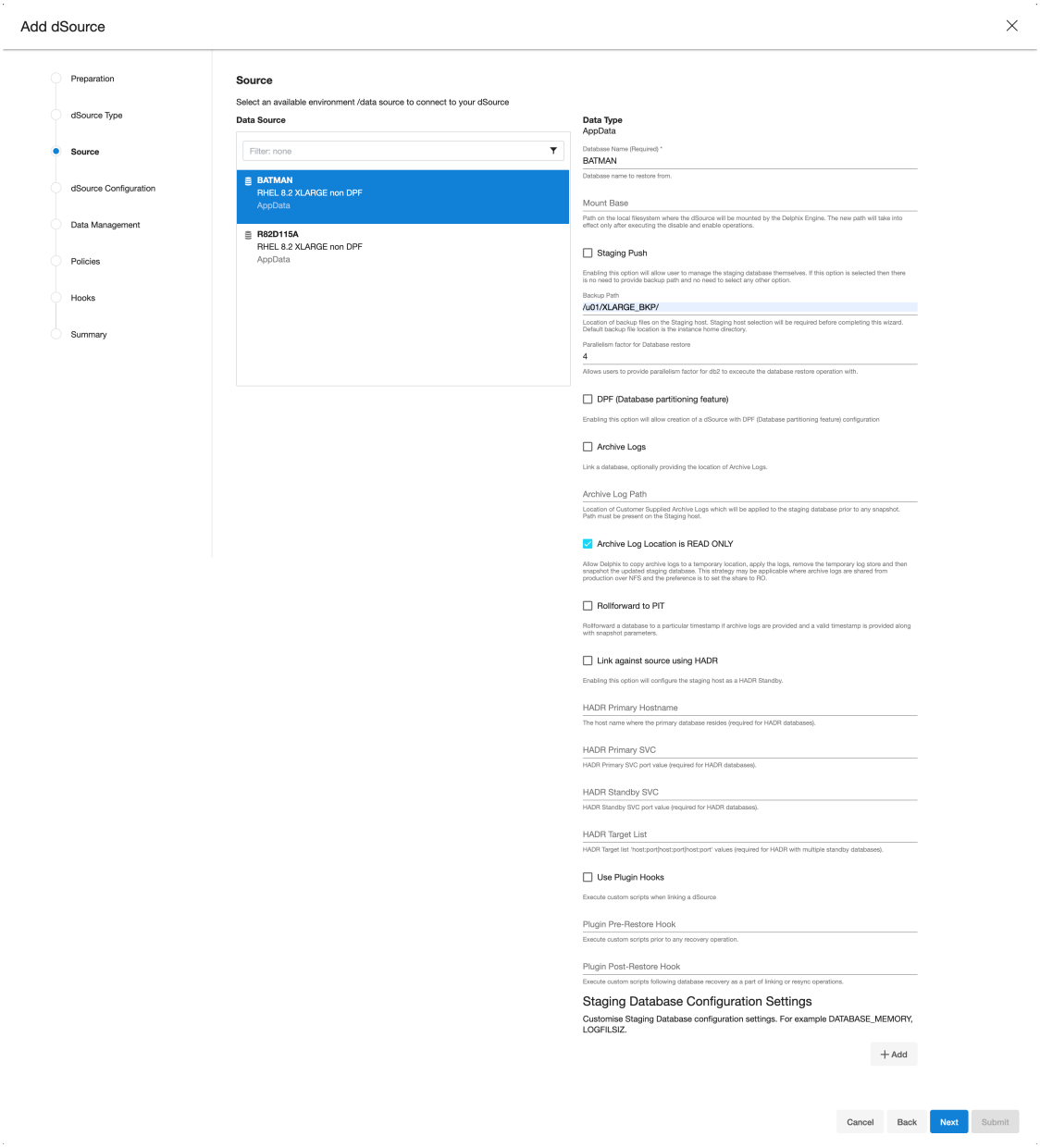
More information about backup best practices is available in IBM Knowledge Center.

Procedure

1. Login to the **Delphix Management Application**.
2. On the **Databases** tab of the Environment Management screen, add a source config against the discovered staging instance.
3. Then, click **Manage** and select **Datasets.**
4. Click the **Plus (+)** icon and select **Add dSource**, you'll get a list of available source configs using which you can go for dSource creation.
5. In the **Add dSource** wizard, select the required source configuration.

Consider the following when choosing an ingestion method.

a. Users can specify their own mount path to host the dSource dataset by specifying the path in the **Mount Base** field.
b. The restore and rollforward operations on the staging database can be managed by selecting the "**Use Staging Push**" checkbox. If the "**Use Staging Push**" checkbox is selected, the other checkboxes must not be selected.
c. If you are working with a HADR setup and want to ingest the database on a staging using HADR configurations, you need to check the **Link against source using HADR** checkbox and uncheck the **Customer Supplied Archive Logs** checkboxes. However, if you want to setup a dSource using HADR configurations using Staging Push, you need to only select the **Staging push** checkbox and handle restore and HADR configuration manually.
d. To use a non-HADR method for database ingestion, you need to uncheck the **Link against the source using HADR** and **Customer Supplied Archive Logs** checkboxes.
e. To use the **Customer Supplied Archive Logs** method for database ingestion check the **Customer Supplied Archive Logs** checkbox and uncheck the **Link against source using HADR** checkbox and enter the path in **Archive Log Path.**
f. To use the **Database partition feature** check **Use DPF** checkbox and uncheck the **Link against the source using HADR** checkbox. For more information on DPF please view IBM Db2 Overview.

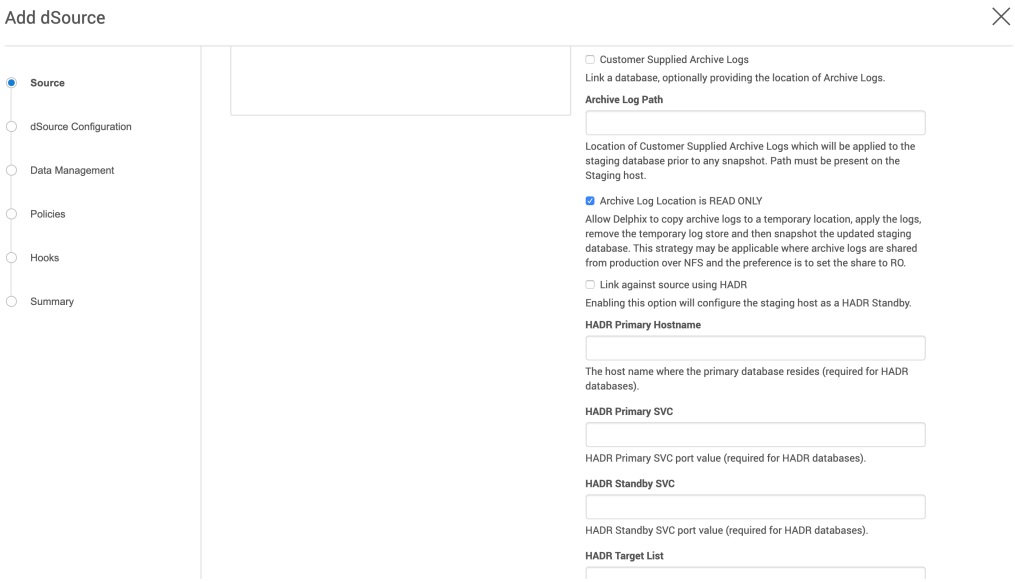


6. The **database name** is mandatory and must be unique for a given instance. This is the name that the database was on the instance it was restored from.
7. If the user is planning to create a dSource with the "**Staging Push**" feature then the user must need to select the "**Use Staging Push**" checkbox.
8. Enter the complete **Backup Path** where the database backup file resides. If no value is entered, the default value used is the instance home directory. If there are multiple backup files for a non-DPF database on the backup path, the most current one will be used. If there are multiple sets of backup files for a DPF database on the backup path, the most recent one will be used.
9. Optionally, you can add the **Parallelism factor for Database restore** to provide the degree of parallelisation to run the database restore operation. If no value is entered, the Db2 plugin inherently assigns parallelism factor.
   Info: Info :

   a. Parallelism factor depends on the resources provided on the staging host, you may not see any difference in performance if there aren't enough resources for Db2 plugin to implement the parallelism.
   b. You can provide any positive integer value for parallelism factor. Db2 plugin will provide that level of parallelism given that the system has appropriate resources to offer.
   c. Parallelism factor is inversely proportional to the Database restore time, which implies that a higher parallelism factor will result in reduced restoration time.
   d. For optimal performance during restoration, make sure to provide the same parallelism factor while creating the backups.
10. If the user wants to roll forward a snapshot to a user-specified timestamp, then the user must select the **Rollforward to PIT** checkbox. Users can also manually do this through the snapshot option.
    Info: The user-specified timestamp needs to be provided along with the timezone. For example, 2021-11-14-04.15.00 **UTC**. If the timezone is not entered, then the user's local timezone is considered.
11. Optionally, users can set the database configuration parameters during the linking operation in the **Config Settings** section.
12. Optionally, users can provide a bash script as a **Plugin** defined hook. Delphix has two Plugin managed Hooks:
    a. **Plugin Pre-Restore Hook:** The bash script entered in this field will get executed before restoration. This hook is useful in such cases where the users want to prepare the full backup before the restore operation.
    b. **Plugin Post-Restore Hook:** The bash script entered in this field will get executed after completion of the restore operation. This hook is useful in such cases where the user wants to purge the full backup file after the restore operation.

```
1. Plugin managed pre-Hook - User can copy the required full backup file from some remote location to staging host. Below is an example of such bash script (name of below script is copyBackup.sh)
#!/bin/bash

# Copying backup file from a remote host remote.delphix.com
scp auto1051@remote.delphix.com:/backuplocation/R74D105A.0.auto1051.DBPART000.20180510073639.001 /db2backup/

The user can provide the above script in Plugin managed pre-Hook field as "sh copyBackup.sh"

2. Plugin managed post-Hook - User can purge the full backup file from the staging host. Below is an example of such bash script (name of below script is purgeBackup.sh)
#!/bin/bash

# Purging full backup file from staging host for saving the staging storage
rm -f /db2backup/R74D105A.0.auto1051.DBPART000.20180510073639.001

The user can provide the above script in Plugin managed post-Hook field as "sh purgeBackup.sh"
```

13. If the dSource is to use **HADR** please enter the following fields. For more information about HADR please view Linking a dSource from a Db2 Database: An Overview.
    a. Enter a fully qualified HADR Primary Hostname. This is a required field for HADR and must match the value set for HADR_LOCAL_HOST on the master.
    b. Enter the port or /etc/services name for the HADR Primary SVC. This is a required field for HADR and uses the value set for HADR_LOCAL_SVC on the master.
    c. Enter the port or /etc/services name for the HADR Standby SVC. This is a required field for HADR and uses the value set for HADR_REMOTE_SVC on the master.
14. If you are using **Customer Supplied Archive Logs** for dSource creation complete the following fields:
    a. Check the Customer Supplied Archive logs checkbox
    b. Provide the log location of the archive logs. In this location, users can load copy files by appending schema and table name to it so that plugin can process those load copy files during log apply process. Archive logs will be used for applying the logs during the snapshot operation. This location must be a part of the staging server and this location must be different than full backup location.



c. The **Archive Log Location is READ ONLY**(default) option indicates whether Delphix should copy the log files to the dSource's filesystem prior to the log apply process. If this is not selected the staging database will read the log files directly from the user-provided log location and perform the log apply operation.
Note: Special considerations for Archive Log Location is READ ONLY
If this operation is performed on the source database and user want the Plugin to process the load files place the load copy files in your location and append the schema and table names (for example: <DB Name>.<Number>.<Instance>.DBPART<partNumber>.<Timestamp>.001.<schema name>.<table name>) to the load copy file names
If you do not want to process load copy files via the Plugin place the load copy files in a similar structure as it is on the source or set a DB2 registry parameter DB2LOADREC.
Note: Special consideration during dSource upgrade from older version plugin to Customer Supplied Archive Logs feature enabled plugin
Suppose the user is having a dSource with an older version of the plugin and now wants to upgrade that dSource to **Customer Supplied Archive Logs** feature then the user needs to perform the following steps :
1. Go to the configuration tab of dSource and then click on the custom sub-tab.
2. Click on the edit icon.
3. Check the **Customer Supplied Archive Logs** and **Archive Log Location is READ ONLY** checkbox.
4. Click on submit and then go for RESYNC operation to upgrade the dSource.

15. Click **Next**.
16. Enter a name and select a group for your dSource.
Adding a dSource to a dataset group lets you set Delphix Domain user permissions for that database and its objects, such as snapshots. See the topics under Users and Groups for more information.
17. Click **Next** and set the **Staging Environment** to be the same as the dSource host. Select the **Staging Environment User** to be the same as the instance owner of the dSource instance.
Info : Changing the Environment User
If you need to change or add an environment user for the dSource instance, see Managing Db2 Users and Instance.
Note: Ensure that the Staging environment must be the environment that you have chosen to create the source config.
Select the **User** from the drop-down and set the **SNAPSHOT PARAMETERS** for the dSource snapshot.
    a. Click on the **Timestamp** input field to enter the timestamp at which the snapshot will be taken.
    The user-specified timestamp needs to be provided along with the timezone in YYYY-MM-DD-

HH.MM.SS format. For example, 2022-03-12-04.15.00**UTC**. If the timezone is not entered, then the user's local timezone is considered.
Note: Providing the timestamp will only work if the user has checked the **Rollforward to PIT** option in step 9 above.

b. Select **Resynchronize dSource** to resynchronize the dSource. This will force a non-incremental load of data from the source similar to when the dSource was created. This action avoids duplicating storage requirements and maintains a Timeflow history.

18. The Delphix Engine will initiate two jobs to create the dSource: DB_Link and DB_Sync. You can monitor these jobs by clicking Active Jobs in the top menu bar, or by selecting System > Event Viewer. When the jobs have completed successfully, the database icon will change to a dSource icon on the Environments > Host > Databases screen, and the dSource will also appear in the list of Datasets under its assigned group. Then, click **Next** and you'll get the Policies section. Set the desired **Snapsync Policy** for the dSource. For more information on policies see Data Management Settings for Db2 dSources.
19. Click **Next** and specify any desired pre- and post-scripts. For details on pre- and post-scripts, refer to Customizing Db2 Management with Hook Operations.
20. Click **Next**. Review the dSource Configuration and Data Management information in the Summary section.
21. Click **Submit**.

The dSource configuration screen
After you have created a dSource, the **dSource Configuration tab** allows you to view information about it and make modifications to its policies and permissions. In the **Datasets** panel, select the dSource you wish to examine. You can now choose the configuration tab to see information such as the **Source files, Data Management** configuration, and **Hook Operations**.

Purging of archive logs after snapshot operation

Logs can be purged from your log location after the snapshot operation. The Plugin will preserve the list of logs that are eligible for purging in the snapshot's metadata. The snapshot metadata can be accessed via the Delphix CLI or API.

Backup Source Database

Source database with Raw DEVICE type storage

Several users use raw device-based tablespaces for source DB2 databases. To leverage these environments with Delphix, Delphix has built a workflow using Db2s native tools that allow Delphix to discover and convert a raw device-based tablespace into an automatic storage-based tablespace during ingestion. Once the data is ingested into staging, customers will be able to provision VDBs of the automatic storage-based database.

In order to complete the linking process, the Standby dSource must have access to a full backup of the source Db2 databases on disk. This should be a compressed online DB2 backup and must be accessible to the dSource instance owner on disk. Delphix is currently not setup to accept DB2 backups taken using third-party sources such as Netbackup or TSM. All three features of data ingestion, namely HADR, Non-HADR, and Customer Supplied Archive Logs backups must also include logs. Starting with Db2 plugin version 3.1.1, we are also supporting full online backups, which are taken using the "exclude logs" syntax. This support is only applicable to the dSources created using the Customer Supplied Archive Logs (Backup and Logs) ingestion mechanism on a single partition.

Example backup command: db2 backup database <DB_NAME> online compress include logs

Best practices for taking a backup

The following best practices can help improve backup and restore performance:

1. Compression should be enabled
2. Following parameters should be optimally configured:
    a. Utility Heap Size (UTIL_HEAP_SZ)
    b. No. of CPUs
    c. No. of Table Spaces
    d. Extent Size
    e. Page Size
3. Parallelism & Buffer configuration may be used to improve backup performance. Parameters that should be configured are :
    a. Parallelism
    b. Buffer Size
    c. No. of Buffers

More information about backup best practices is available in IBM Knowledge Center.

Procedure

1. Login to the Delphix Management Application using Delphix Engine credentials or as the owner of the database from which you want to provision the dSource.
2. On the **Databases** tab of Environment Management screen, add a source config against discovered staging instance.
3. Then, click **Manage**.
4. Select **Datasets**.
5. Click the Plus (+) icon and select **Add dSource**, you'll get a list of available source configs using which you can go for dsource creation.
6. In the **Add dSource** wizard, select the required source configuration.
7. If you are working with an HADR setup, please leave the HADR checkbox checked unless you're working with Staging push dSource.
8. The **database name** is mandatory and must be unique for a given instance. This is the name that the database was on the instance it was restored from.
9. Enter the complete **Backup Path** where the database backup file resides. If no value is entered, the default value used is the instance home directory. If there are multiple backup files for a database on the backup path, the most current one will be used.
10. Enter the **Log Archive Method1** you wish to use for the database. If no value is entered, the default value used is DISK:/mountpoint/dbname/arch
11. Optionally, users can set the database configuration parameters during the linking operation in the **Config Settings** section.
12. If the dSource is to use HADR please enter the following fields. If it will not use HADR skip ahead to step 13. For more information about HADR please view Linking a dSource from a Db2 Database: An Overview.
    a. Enter a fully qualified HADR Primary Hostname. This is a required field for HADR and must match the value set for HADR_LOCAL_HOST on the master.
    b. Enter the port or /etc/services name for the HADR Primary SVC. This is a required field for HADR and uses the value set for HADR_LOCAL_SVC on the master.
    c. Enter the port or /etc/services name for the HADR Standby SVC. This is a required field for HADR and uses the value set for HADR_REMOTE_SVC on the master.
13. Click **Next**.
14. Select a **dSource Name** and **Database Group** for the dSource.
15. Click **Next**.
    You will get Data Management section where you need to specify staging environment and user which will be used for dsource creation.
16. Set the **Staging Environment** to be the same as the dSource host.
17. Select the **Staging Environment User** to be the same as the instance owner of the dSource instance.

Info: Changing the Environment User
If you need to change or add an environment user for the dSource instance, see Managing Db2 Users and Instance Owners.

18. Then, click **Next** and you'll get the Policies section. Set the desired **Snapsync Policy** for the dSource.
19. Click **Next**.
20. Specify any desired pre- and post-scripts. For details on pre- and post-scripts.
21. Click **Next**.
22. Review the dSource Configuration and Data Management information in the summary section.
23. Click **Submit**.

The Delphix Engine will initiate two jobs to create the dSource: DB_Link and DB_Sync. You can monitor these jobs by clicking Active Jobs in the top menu bar, or by selecting System > Event Viewer. When the jobs have completed successfully, the database icon will change to a dSource icon on the Environments > Host > Databases screen, and the dSource will also appear in the list of Datasets under its assigned group.

The dSource configuration screen
After you have created a dSource, the **dSource Configuration tab** allows you to view information about it and make modifications to its policies and permissions. In the **Datasets** panel, select the dSource you wish to examine. You can now choose the configuration tab to see information such as the **Source files, Data Management** configuration, and **Hook Operations**.

Provisioning a virtual database (VDB)

Prerequisites

- You will need to have linked a dSource from a staging instance, as described in Linking a Db2 dSource, or have created a VDB from which you want to provision another VDB
- You should have set up the Db2 target environment with necessary requirements as described in Requirements for Db2 Hosts and Databases.
- Make sure you have the required Instance Owner permissions on the target instance and environment as described in Managing Db2 Users and Instance Owners.
- The method for Database Permissions for Provisioned Db2 VDBs is decided before the provisioning

You can take a new snapshot of the dSource by clicking the **Camera** icon on the dSource card. Once the snapshot is complete you can provision a new VDB from it.

Please note while creating a VDB on an instance

By default, DB2 Plugin will not allow provisioning a VDB on an instance which contains a database that has an identical name to the source database.

For e.g.;

1. When we are provisioning a VDB from a VDB then source VDB will be treated as the source database.
2. When we are provisioning a VDB from a dSource then a staging database will be treated as the source database.

If the user wants to provision a VDB on an instance which contains a source database, then the user needs to follow the below steps :

1. Check if any config file with name **advanceConfFlag.conf** exists at location **<Toolkit directory>**/. If it does not exist, then the user should create a file with name **advanceConfFlag.conf** at location **<Toolkit directory>**/. Instance user must have read permission to this file. Once the file is created, user should proceed to step 2. If the file already exists at this location with the required permission, the user should directly proceed to step 2.
2. Add the below line in this config file.
    a. allowSourceDBonSameInst=**true**
3. Then create/Refresh the VDB again.

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select a **dSource**.
5. Select a **snapshot** from which you want to provision.
6. Click **Provision VDB** icon to open Provision VDB wizard.
7. Select a target environment from the left pane.
8. Select an **Installation** to use from the dropdown list of available DB2 instances on that environment.
9. Set the **Environment User** to be the Instance Owner. Note: The picking of instance owner is only possible if you have multiple environment users set on that host.
10. Provide VDB Name as database name as parameter.
11. Optionally, set the database configuration parameters for the VDB.
12. Click **Next**.
13. Select a **Target Group** for the VDB.
    Click the green **Plus** icon to add a new group, if necessary.
14. Select a **Snapshot Policy** for the VDB.
15. Click **Next**.
16. Specify any desired hook operations.
17. Click **Next**.
18. Review the **Provisioning Configuration** and **Data Management** information.
19. Click **Submit**.
    When provisioning starts, you can review the progress of the job in the **Databases** panel, or in the **Job History** panel of the **Dashboard**. When provisioning is complete, the VDB will be included in the group you designated and listed in the **Databases** panel. If you select the VDB in the Databases panel and click the **Open** icon, you can view its card, which contains information about the database and its Data Management settings.
    Once the VDB provisioning has successfully completed, if the source and target instance ids are not the same, you may want to grant secadm and dbadm on the database to the target instance id. Refer to Database Permissions for Provisioned Db2 VDBs for more information.

Next steps

Congratulations! You have provisioned your first virtual database!

Now, perform some simple functional tests with your application. You can connect your app to the VDB using standard TNS/JDBC techniques. Delphix has already registered the VDB for you on the target listener.

We suggest the following next steps:

1. Drop a table and use the VDB Rewind feature to test the recovery of your VDB.
2. Take a snapshot of your dSource and refresh your VDB to quickly get fresh production data.

3. Provision a new VDB from your VDB to test sharing data quickly with other teams.
4. Mask your new VDB to protect sensitive data. Provision new VDBs from that masked VDB to quickly provide safe data to development and QA teams.

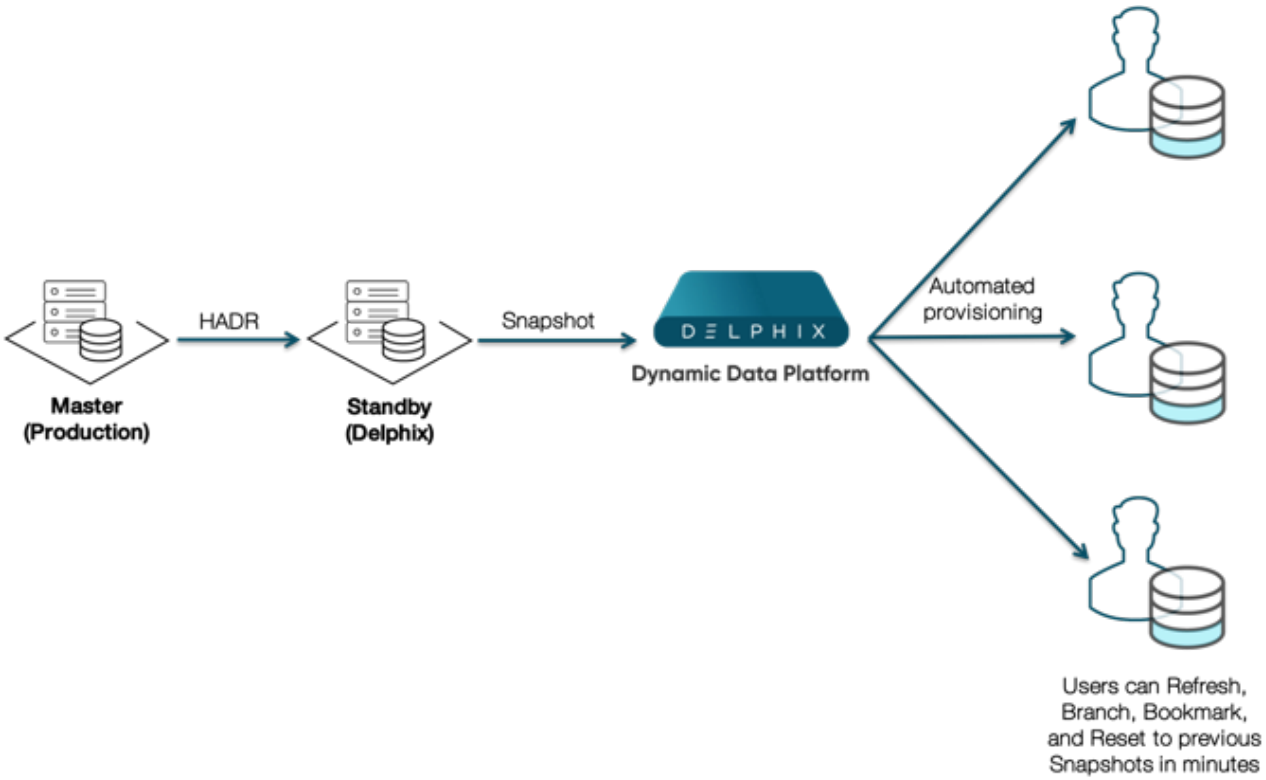
Managing Db2 environments and hosts

This section covers the following topics:

Setting up Db2 environments: An overview

This topic describes the high-level process for adding Db2 environments, linking Db2 instances to the Delphix Engine, and provisioning virtual databases.

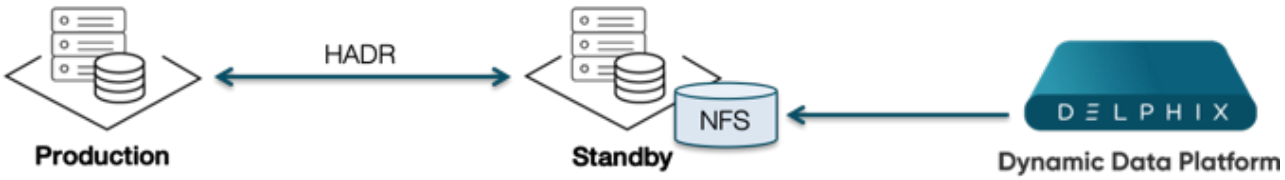
Delphix for Db2 architecture



Delphix uses a Standby server model along with Db2s High Availability Disaster Recovery feature to ingest data and stay in sync with the source database. The standby server is then snapshotted by the Delphix Engine and the snapshots can be provisioned out to one or more target servers.

> The snapshot and provision process occurs on the instance level, all databases that exist on the standby server will be provisioned out to the target machines. Similarly, actions such as bookmark, rewind, and refresh will simultaneously apply to all the databases in the instance.

Block diagram of linking architecture between Db2 environments and the Delphix engine



The linking process sets up a database inside an instance on the standby server and uses this as a HADR standby for the primary database. The staging instance must have access to a recent backup copy of the database that will be used to restore the initial copy of the dSource. Once the restoration process is complete, Delphix will issue the

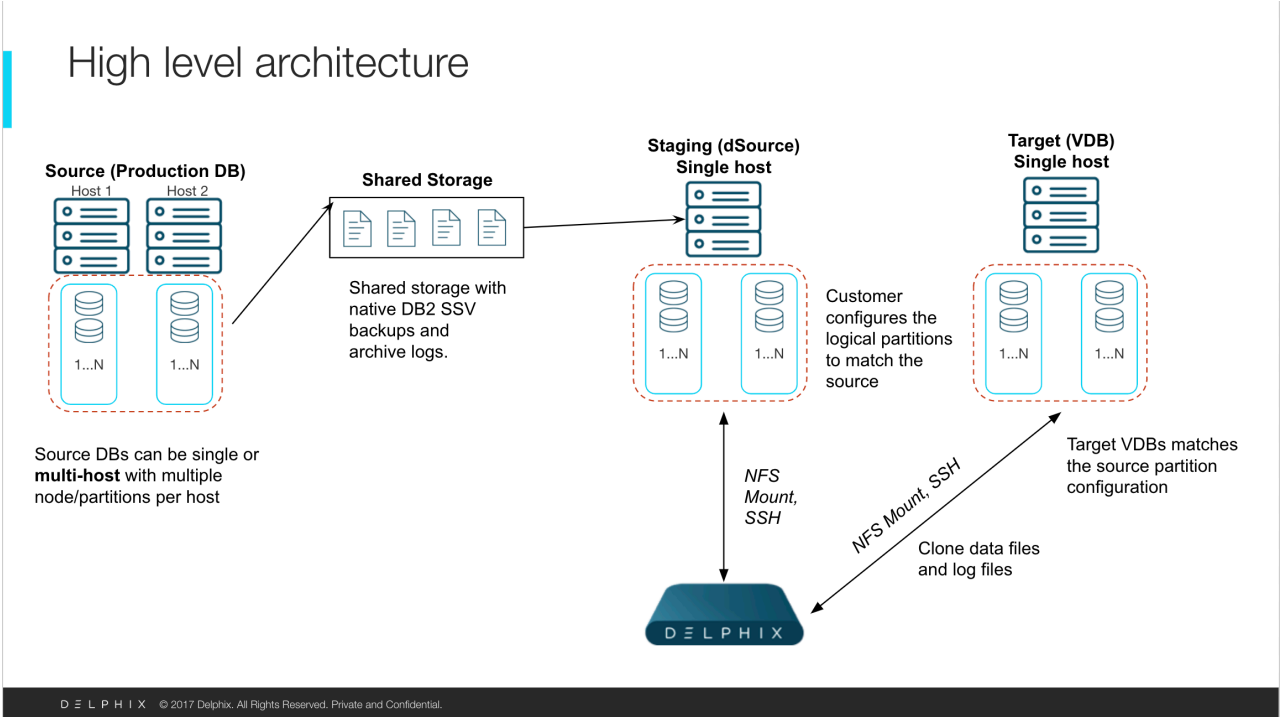
HADR standby commands for the given database and ensure that the health of the HADR connection stays within the acceptable threshold values.

Db2 staging instance set up

The database level feature enables you to have 1-many mappings between instance and databases. It is also possible to set up databases from multiple primary hosts to use the same standby instance.

The choice of databases on the staging server should also take into account the expected network traffic that HADR will create between the source and staging environments.

Db2 DPF block diagram



While using DPF, the DPF environment for Db2 staging and target should be configured on separate hosts. Also, staging and target should be configured with the same number of logical partitions as the number of partitions in the source node.

Adding a Db2 environment

This topic describes how to add a Db2 Staging environment.

Prerequisites

- Make sure that the staging environment in question meets the requirements described in Requirements for Db2 Hosts and Databases

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Next to **Environments**, click the **Actions (...) menu and** select **Add Environment**.
5. In the Add Environment dialog, select **Unix/Linux** in the menu.
6. Select **Standalone Host**.
7. Click **Next**.
8. Enter a **Name** for the environment.
9. Enter the **Host IP address** or **hostname**.
10. Enter the **SSH port**. The default value is 22.
11. Enter an **OS Username** for the environment. For more information about the environment user requirements, see Requirements for Db2 Hosts and Databases
12. Select Login Type. — Username and Password - enter the OS username and password — Username and Public Key - enter the OS username. — Password Vault - select from an existing Enterprise Password Vault
    Note: Currently, the Enterprise Password Vault option is not supported.
    Info: Using Public Key Authentication
    If you want to use public-key authentication for logging into your Unix-based environment, there are two options: use the engine's key pair or provide a key pair for this environment.
    To use the engine's key pair:
    a. Select **Public Key** for the **Login Type.**
    b. Click **View Public Key**.
    c. Copy the public key that is displayed, and append it to the end of your `~/.ssh/authorized_keys` file. If this file does not exist, you will need to create it.
        i. Run chmod 600 ~/.ssh/authorized_keys to allow only the file's owner to read and write to it (make sure the file is owned by the user).
        ii. Run chmod 755 ~ to restrict access to the user's home directory so no other user may write to it.
        iii. Run chmod 700 ~/.ssh so that others cannot write to it. The ~/.ssh directory cannot be writable by group or other users. Otherwise, authentication will fail.
    As an alternative, you can provide a key pair specific for this environment via the API or CLI. See Option 2 in this CLI Cookbook article for instructions.
13. For Password, enter the password associated with the user in step 11.
14. Using Public-Key Authentication:
    a. Run chmod 600 ~/.ssh/authorized_keys to allow only the file's owner to readand write privileges to it (make sure the file is owned by the user).
    b. Run chmod 755 ~ to restrict access to the user's home directory so no other user may write to it.
    c. Run chmod 700 ~/.ssh so that others cannot write to it. The ~/.ssh directory cannot be writable by group or other users. Otherwise, authentication will fail.
15. You can also add public key authentication to an environment user's profile by using the command-line interface, as explained in the topic CLI Cookbook: Setting Up SSH Key Authentication for UNIX Environment Users

16. For Password Login, click Verify Credentials to test the username and password.
17. Enter a **Toolkit Path** (make sure the toolkit path does not have spaces).
    For more information about the toolkit directory requirements, see Requirements for Db2 Hosts and Databases
18. To provide your own Oracle Java select the **Provide my own JDK** checkbox and click **Next**.
19. In the Java Development Kit tab enter the absolute path to your Oracle JDK and click **Next**.
20. Click **Submit**.

As the new environment is added, you will see two jobs running in the Delphix platform Job History, one to Create and Discover an environment, and another to Create an environment. When the jobs are complete, you will see the new environment added to the list in the Environments tab. If you do not see it, click the Refresh icon in your browser.

Post-requisites

To view information about an environment after you have created it:

1. Click **Manage**.
2. Select **Environments**.
3. Select the **environment name**.

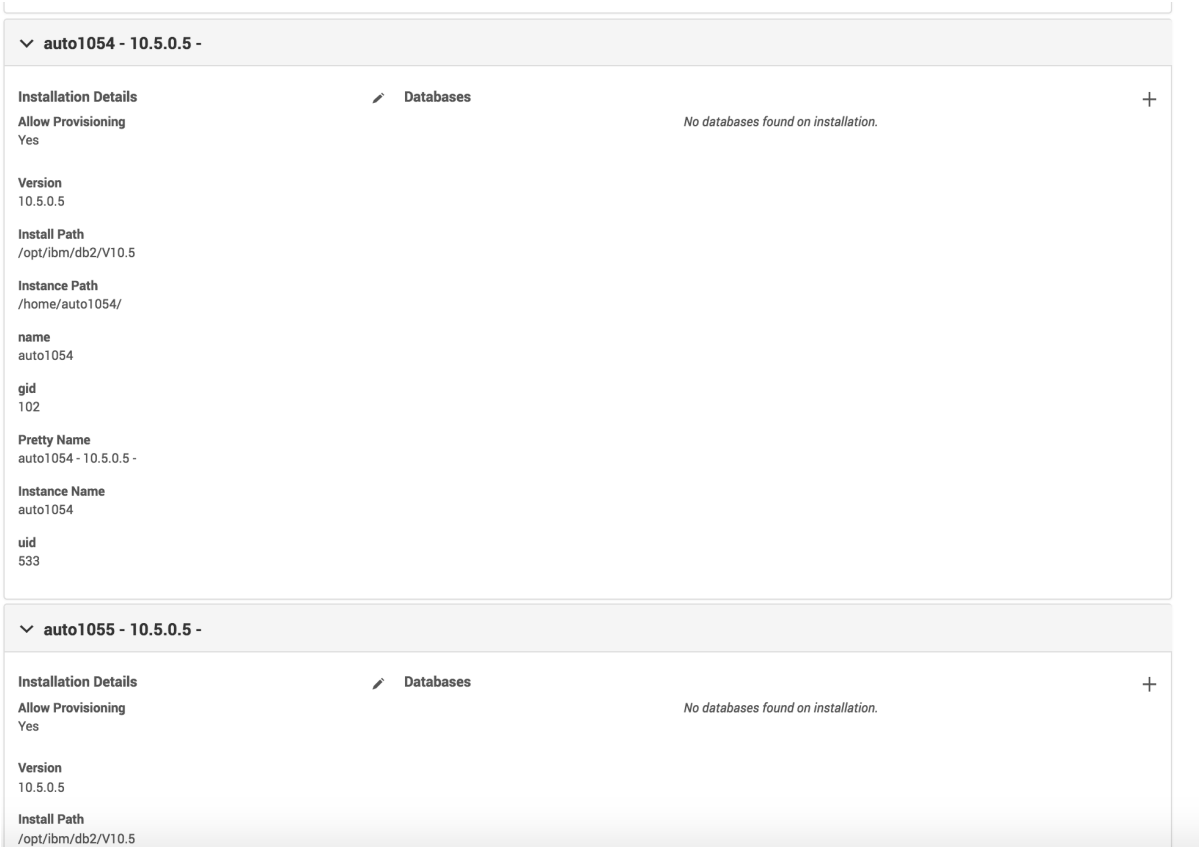
Managing Db2 instances

This topic explains how to manage DB2 instances on a host.

When you add an environment with the Delphix Management Application, all existing DB2 instances on the host are automatically discovered by Delphix. A list of all instances and databases are available to Delphix based on the environment discovery process.

View instances

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. In the **Environments** panel, click on the **name of the environment** to you want to refresh.
5. Select the **Databases** tab to see a list of all DB2 instances found in the environment.



Updating instances on a host

If you add or remove any DB2 instances from a host you must refresh the environment using the steps listed in Environment Refresh to update the list of instances available to Delphix. Additionally, the environment users must be updated to match the DB2 instance owners as detailed in Managing DB2 Users and Instance Owners

Managing Db2 users and instance owners

All DB2 Delphix operations such as linking, snapshot, and provisioning always occur on the database level. As a result, we require that any DB2 instance that you wish to use either as Staging or Target must have its associated instance owner added as an environment user to the host within Delphix. Please perform the following steps to add each DB2 instance owner you to use on the host.

It is important to note that Delphix will not create new instances on the host and can only use existing DB2 instances that it finds on the host machine. For more information on instance management refer to Managing Db2 Instances

Adding environment users

Procedure

1. Login to the Delphix Management application.
2. Click **Manage**.
3. Select **Environments**.
4. Click on the existing environment name you want to modify and open the environment information screen.
5. In the Details tab, click the **Plus** icon located next to Environment users.
6. Enter the Username and Password for the OS user in that environment.

1. If you want to use public-key encryption for logging into your Unix-based environment:
    a. Select **Public Key** for the **Login Type.**
    b. Click **View Public Key**.
    c. Copy the public key that is displayed, and append it to the end of your `~/.ssh/authorized_keys` file. If this file does not exist, you will need to create it.
        i. Run `chmod 600 authorized_keys` to enable read and write privileges for your user.
        ii. Run `chmod 755 ~` to make your home directory writable only by your user.

    The public key needs to be added only once per user and per environment.

7. Click the **Check** icon to save the new user.

8. To change the primary user for this environment, click the **Star** icon next to Environment Users. Only the primary user will be used for environment discovery.

9. To delete a user, click the **Trash** icon next to their username.

Environment attributes for hosts with Db2

This topic describes the more advanced attributes for Db2 data platforms.

Common editable attributes

Attribute	Description
Environment Users	The users for that environment. These are the users who have permission to ssh into an environment or access the environment through the Delphix Connector. For more information on the environment user requirements, see the **Requirements** topics for specific data platforms.
Host Address	The IP address of the environment host.
SSH Port	The SSH port number used for the environment.
Toolkit Path	The plugin path for the environment. Delphix uses the Toolkit path (Toolkit is a synonym for Plugin) for logs, mount points, and keeping lib files.
Notes	Any other information you want to add about the environment

Linking data sources and syncing data with Db2

This section covers the following topics:

Linking a dSource from a Db2 database: An overview

This topic describes basic concepts behind the creation of dSources from Db2 instances.

Database level operation

Associated with an instance is the concept of an instance owner. This is the user who "owns" that instance and has SYSADM authority over the instance and all databases inside that instance. SYSADM authority is the highest level of authority in Db2 and lets this user perform several database management activities such as upgrade, restore, and editing configurations. More information about instances is located in the IBM knowledge center

Db2 database level support

Delphix operates at the database level and requires that

- The staging and target hosts must have the empty instances created prior to Delphix using them
- The desired OS user to execute commands related to dSource and VDB operation for each instance has been added as an environment user

Note:

- Make sure the staging instance used for linking doesn't have an existing restoring database with the same name.
- The dSource ingestion or VDB provisioning should not be performed on the source instance to maintain the integrity of the source database.

Data Ingestion

Db2 for Delphix ingests data by using a staging database created on a discovered Standby instance of Db2. The dSource uses the staging database to stay in sync with the production database. This is done by first going through the linking process during which a full backup is used to recover an initial copy of the production database to the staging database. Storage for the staging database is provided by Delphix via an NFS mount to storage exported by the Delphix system.

A single standby instance can contain data from multiple source databases.

Pipelining logic for implementing parallel restores

The plugin employs a parallel pipeline methodology so that the restore operation of non-catalog partitions can be performed in parallel in the Database Partitioning Feature (DPF). The number of parallel restores is determined by the value of “restorePipelineLimit” (default value is 10) in <Toolkit directory>/advanceConfFlag.conf. For more details on this please refer to Requirements for Db2 Hosts and Databases. The parameter “restorePipelineLimit” is configurable by end-users. The plugin performs parallel restore for all non-catalog partitions. For e.g. If the total number of non-catalog partitions is 15, and the "restorePipelineLimit" parameter is set to 10, the first set of 10 restores will happen in parallel. The plugin will track the restore of each partition present in the pipe. Whenever a restore of a partition completes, it will move out from the pipe and a new partition will enter into that pipe. Thus, the plugin ensures that the pipe will always have the user-configured number of partitions being restored (default=10).

In case of a restore failure for a particular partition, the plugin will track that failure and will print the same in the plugin-generated logs so that the user can take the necessary action. If the user received failure during restore operation for a few partitions and then decided to manually initiate the restores for failed partitions, the plugin will

verify the datapaths during snapshot operation. In case it finds wrong path information (such as paths outside the mount point), then the plugin will error out of the operation.

The plugin will track the status of each restore in a file named <DB Name>_metadata.json The plugin can handle a scenario where the staging host reboots in-between the parallel restores. In this case, the user can perform a “Resynchronisation” operation again. At this time, the remaining restores will resume from scratch. For more details on this please refer to Requirements for Db2 Hosts and Databases

Data synchronization

Delphix provides the following options to keep your dSource in sync with the source database:

1. During the linking process, you can set up a HADR connection between the original source database and the Standby instance. This allows the Standby instance using HADR for log shipping to always keep its databases in sync with the source. It is important to note that a single Standby instance (dSource) can contain multiple databases from multiple different servers and instances as long as each database has a unique name.
2. The users can opt for the "**Staging Push**" feature by selecting the checkbox "**Use Staging Push**". With this feature, the user can manage the restore and roll forward of a staging database without involving the Delphix Engine.
3. Users can opt for a non-HADR method of database ingestion. The database will be ingested to a standalone staging host and the user can trigger a resynchronization operation with a most recent online full backup to keep the staging database up to date with the source database.
4. Users can opt for the Delphix Recovery with the **Customer Supplied Archive Logs** method. During an update, the dSource user can trigger a snapshot operation where user-provided archive logs will get applied to the staging database.
5. Optionally, when the DPF feature is in use, users can use the “Customer Supplied Archive Logs” feature. Users need to place archive logs inside a folder with a name as NODE<Partition number> where <Partition number> is a four-digit number. For example, if the source environment has two partitions then the user-provided log path will have folder names NODE0000 and NODE0001. Both the folders will have respective archive logs. Snapshot operation will be used to apply the archive logs.

⚠ Delphix HADR standby configuration may not meet Db2 DR requirements and should only be used for Delphix use cases.

Linking a Db2 dSource

This topic describes how to link a Db2 staging dSource.

Prerequisites

Be sure that the source and staging instances meet the host requirements and the databases meet the container requirements described in Requirements for Db2 Hosts and Databases

Source database preparation

Instance owner permissions

Delphix uses the Db2 instance owner account on the dSource for multiple operations, including verifying the data inside the databases. In order to ingest a database on the staging server with different instances, we need permissions on the source database in order to perform a restore on the staging server. As an example, if we have an instance named "auto1051" at the source with a database name "delphix" and we want to create a dSource on the "auto1052" instance on the staging server, then you must explicitly grant DBADM and SECADM to the dSource instance "auto1052" on the source instance using the following steps:

1. Connect to the source database as the source instance owner.
    a. *connect to <DB_NAME> user <SOURCE_INSTANCE_OWNER>*
2. Issue database grant command
    a. *grant DBADM, SECADM on the database to user <DSOURCE_INSTANCE_OWNER>*
    b. *grant DBADM, SECADM on the database to user <VDB_INSTANCE_OWNER>*
3. Repeat step 2 for every database to be included in the dSource, on the corresponding source database.

Determine if your dSource will be a non-HADR instance, a HADR single standby instance, or a HADR multiple standby instance. Non-HADR dSources can only be updated via a full dSource resync from a newer backup file

Non-HADR Database

1. See the "InstanceOwner Permissions" section above.
2. Ensure that the source database has the necessary user permissions for the provisioned VDBs as described in Database Permissions for Provisioned Db2 VDBs

HADR single standby database

1. All items in Non-HADR Database section above.
2. The following database configuration settings must be set:
    a. *update db cfg for <DB_NAME> using HADR_LOCAL_HOST <PRIMARY_IP> HADR_LOCAL_SVC <PRIMARY_PORT > immediate*
    b. *update db cfg for <DB_NAME> using HADR_REMOTE_HOST <STANDBY_IP> HADR_REMOTE_SVC <STANDBY_PORT> immediate*
    c. *update db cfg for <DB_NAME> using HADR_REMOTE_INST <STANDBY_INSTANCE_NAME> immediate*
    d. *update db cfg for <DB_NAME> using HADR_SYNCMODE SUPERASYNC immediate*
3. If database configuration parameter LOGINDEXBUILD is set to OFF, do the following:
    a. *update db cfg for <DB_NAME> using LOGINDEXBUILD ON*
    b. Force off all connections to the database and reactivate the database
4. If database configuration parameter LOGARCHMETH1 is set to OFF, do the following:
    a. *update db cfg for <DB_NAME> using LOGARCHMETH1 XXXX (must be a valid log archiving method)*
    b. Take an offline backup
5. If LOGARCHMETH1 points to a third-party backup server (i.e. TSM or Netbackup) define LOGARCHMETH2 to disk
    a. *update db cfg for <DB_NAME> using LOGARCHMETH2 DISK:<full path to archive log directory>*

        i. Log files in the directory must be available from the time of the backup until the restore has successfully completed on the dSource.
6. *db2 start hadr on db <DB_NAME> as primary by force*
7. Take a full online backup as defined in the "Backup Source Database" section below.
8. Record the following information, as it must be entered on the Delphix Engine while creating the dSource.
    a. HADR Primary hostname
    b. HADR Primary SVC
    c. HADR Standby SVC (auxiliary standby port)

HADR multiple standby databases

This **assumes** a single standby database HADR setup already exists. The existing standby will be referred to as the main standby. The new delphix standby will be referred to as the auxiliary standby.

1. The following database configuration settings must be set on the primary database:
    a. *update db cfg for <DB_NAME> using HADR_SYNCMODE <SYNC MODE> immediate – set whichever sync mode you wish to use on your main standby.*
    b. *update db cfg for <DB_NAME> using HADR_TARGET_LIST "<MAIN_STANDBY_IP:MAIN_STANDBY_PORT|AUXILIARY_STANDBY_IP:AUXILIARY_STANDBY_PORT>" immediate*
        i. You may have up to two auxiliary standbys defined separated by a '|'; one of which must be the delphix dSource.
2. *stop hadr on db <DB_NAME>*
3. *start hadr on db <DB_NAME> as primary by force*
4. *Take a full online backup as defined in the "Backup Source Database" section below. While this backup is running, you may continue with step 5.*
5. *The following database configuration settings must be set on the existing main standby database:*
    a. *update db cfg for <DB_NAME> using HADR_SYNCMODE <same mode as defined in 1.a above.> – It must be the same value used for the primary database.*
    b. *update db cfg for <DB_NAME> using HADR_TARGET_LIST "<PRIMARY_IP:PRIMARY_PORT|MAIN_STANDBY_IP:MAIN_STANDBY_PORT>"*
6. *stop hadr on db <DB_NAME>*
7. *start hadr on db <DB_NAME> as standby*
8. *Record the following information, as it must be entered on the Delphix Engine while creating the dSource (the auxiliary standby database):*
    a. HADR Primary hostname
    b. HADR Primary SVC
    c. HADR Standby SVC (auxiliary standby port)
    d. HADR_TARGET_LIST <PRIMARY_IP:PRIMARY_PORT|MAIN_STANDBY_IP:MAIN_STANDBY_PORT>

Delphix recovery with customer supplied archive logs

The Delphix Recovery with Customer Supplied Archive Logs (formerly known as Delphix Backup and Logs ingestion), feature allows users to provide Delphix with the location of the source database archive logs. Full backups only have to be restored during dSource creation, or during RESYNC operations when there are exceptions, for example, lost/corrupt log files. Archived logs get applied during the snapshot operation.

A full backup is required to create a dSource and prior to the snapshot operation, users can provide the archive logs to update the dSource during the snapshot operation, user-provided logs will be applied to the staging database. Applying the logs on a regular basis allows users to keep their dSource in sync with their source database. The staging database will always be in a roll forward pending state. The Plugin performs validation before applying the logs. Plugin managed Pre and Post Hooks can be used for preparing and purging the full backups and archive logs. A list of logs that are available for purging is stored in the snapshot metadata of the dSource. In case DPF is enabled, users need to place archive logs inside a folder with a name as NODE<Partition number> where <Partition number> is a four-digit number. For example, if the source environment has two partitions, then the user-provided log path

will have folder names as NODE0000 and NODE0001. Both these folders will have respective archive logs. Snapshot operation will be used to apply the archive logs.

HADR and non-HADR configured dSources can be moved to this feature by performing a RESYNC operation.

Delphix supports Customer Supplied Archive Logs with Db2 version 10.5 and 11.1. To use Customer Supplied Archive Logs simply check the checkbox provided during dSource creation. Customer Supplied Archive Logs require that:

- A full backup must be accessible on the staging server, it could be a shared location but must be mounted on the staging host and visible as a file system.
- Archived logs must also be accessible on the staging server
    - The following command can be used to generate archive logs from the source database;
        - db2 archive log for database <database name>
    - Instance users must have read permissions on logs so that Delphix-managed recover can apply the logs.
    - Instance users must have read permissions on logs and additional write permissions on customer-provided log location if Delphix is planning to use customer-provided log location for applying the logs.
    - If the user has placed the archive logs at a user-provided log location, then the logs must be valid and the first active log of the staging database must be present in the user-provided log location.
- If database configuration parameter LOGINDEXBUILD is set to OFF, do the following:
    - update db cfg for <DB_NAME> using LOGINDEXBUILD ON
    - Force off all connections to the database and reactivate the database
- The customer does log purging from the provided log location. Delphix only provides the list of logs that are eligible for purging.

Limitations:

- Continuous updates to the dSource are not available. The DB_SYNC operations apply new logs from the log location and keep the dSource updated with changes on the source.
- Read access to the staging database is not available.

Staging push

With the "Staging Push" feature, you can now manage the restore and rollforward operations from the CLI. During dSource creation (or RESYNC), the plugin will create an empty mount point and will provide a restore and rollforward template that will allow you to restore the database. You can restore the dSource using the steps mentioned in the readme or through any other method of their choice. You can also configure the dSource with HADR by following the steps provided in the "HADR Single/Multiple Standby Database" sections above. Staging push is now supported on both DPF and non-DPF Db2 instances.

Once the dSource is ready, you can perform restore and rollforward operations from the CLI. After the restore, the staging database should remain in the rollforward pending state, and prior to the next snapshot, the user can do a rollforward to a specific Point-in-Time (by providing a timestamp to the rollforward command) against the available archive logs. In the case of HADR dSource, you will have to stop HADR before taking a snapshot and starting it back on after the snapshot operation is complete.

You can create a pre-snapshot and post-snapshot hook to perform HADR stop and start operations on the staging database respectively so that you don't have to connect to the staging host every time a snapshot is captured.

Prior to the snapshot operation, the user should validate the backups and the archive logs before applying them to the staging database. Below are the user side validations:

User side validation prior to snapshot operation

- The user must validate the backup files using ***db2ckbkp*** utility on the staging host.
- The user must validate the archive logs using ***db2cklog*** utility on the staging host.
- The authenticity of archive logs would be managed by the user.
- The user has to provide the first active log at the staging database log directory location after applying the logs.
- If there is a scenario where a user performed some load copy (non-logged transactions) operations on the source side then it's the responsibility of the user to make those non-logged transactions available on the staging database.

Backup source database

New feature: Source database with Raw DEVICE type storage

Several users use raw device-based tablespaces for source Db2 databases. To leverage these environments with Delphix, Delphix has built a workflow using Db2s native tools that allow Delphix to discover and convert a raw device-based tablespace into an automatic storage-based tablespace during ingestion. Once the data is ingested into staging, customers will be able to provision VDBs of the automatic storage-based database.

In order to complete the linking process, the Standby dSource must have access to a full backup of the source Db2 databases on disk. This should be a compressed online Db2 backup and must be accessible to the dSource instance owner on disk. Delphix is currently not set up to accept Db2 backups taken using third-party sources such as Netbackup or TSM. All three features of data ingestion, namely HADR, Non-HADR, and Customer Supplied Archive Logs backups must also include logs. Starting with the Db2 plugin version 3.1.1, we are also supporting full online backups, which are taken using the "exclude logs" syntax. This support is only applicable to the dSources created using the Customer Supplied Archive Logs (Backup and Logs) ingestion mechanism on a single partition.

Support for named pipe type DB2 backup

Users can also provide Db2 backups of type named-pipe. Db2 plugin will check the type (type will be either named-pipe or a file) of backup and will use the restore syntax accordingly.

Example backup command for DB2 Non SSV backups: db2_all "db2 backup database <DB_NAME> online to <Backup File Location> compress include logs"

Backup types supported in DPF

The type of backups supported by Db2 plugin when used with DPF are:

- SSV backups through backup files or named pipes.
- Non-SSV backups through backup files or named pipes.

Example backup command for DB2 SSV backups: db2_all "db2 BACKUP DATABASE<DB_NAME> ON all dbpartitionnums online to <Backup File Location> compress include logs"
Example backup command for DB2 Non SSV backups: db2_all "db2 backup database <DB_NAME> online to <Backup File Location> compress include logs"

Best practices for taking a backup

The following best practices can help improve backup and restore performance:

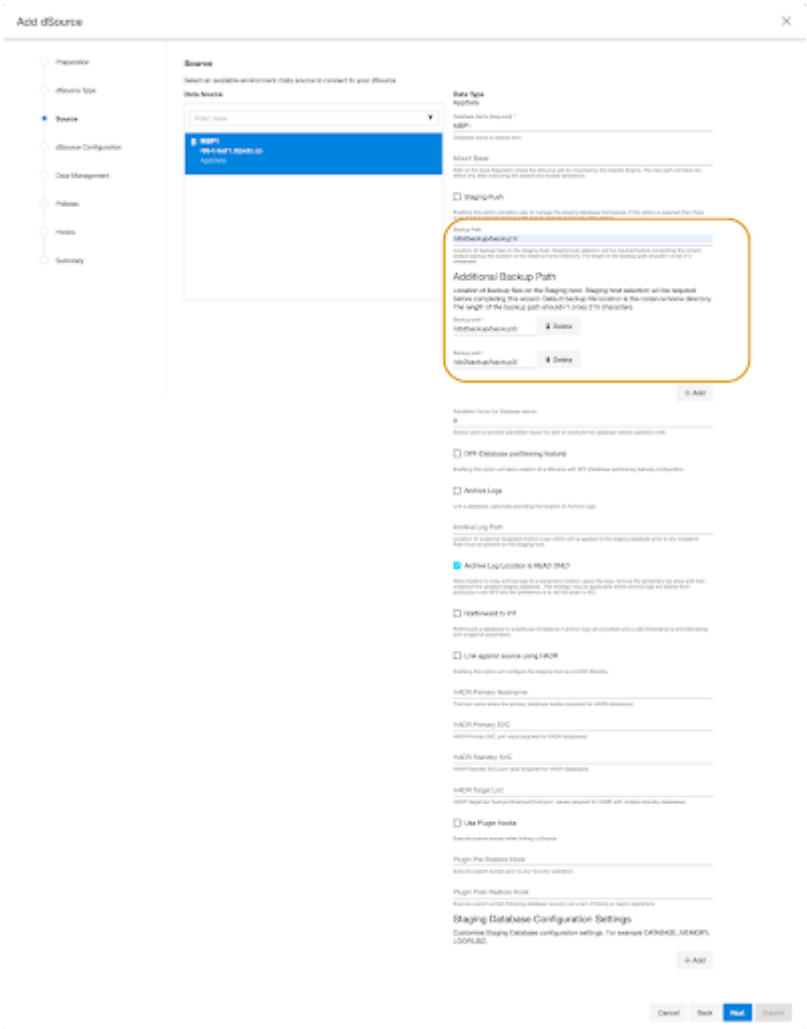
1. Compression should be enabled

2. Following parameters should be optimally configured:
    a. Utility Heap Size (UTIL_HEAP_SZ)
    b. No. of CPUs
    c. No. of Table Spaces
    d. Extent Size
    e. Page Size
3. Parallelism & Buffer configuration may be used to improve backup performance. Parameters that should be configured are:
    a. Parallelism
    b. Buffer Size
    c. No. of Buffers

More information about backup best practices is available in IBM Knowledge Center

Procedure

1. Login to the **Delphix Management Application**.
2. On the **Databases** tab of the Environment Management screen, add a source config against the discovered staging instance.
3. Then, click **Manage** and select **Datasets.**
4. Click the **Plus (+)** icon and select **Add dSource**, you'll get a list of available source configs using which you can go for dSource creation.
5. In the **Add dSource**wizard, select the required source configuration.
   Consider the following when choosing an ingestion method.
    a. Users can specify their own mount path to host the dSource dataset by specifying the path in the **Mount Base** field.
    b. The restore and rollforward operations on the staging database can be managed by selecting the "**Use Staging Push**" checkbox. If the "**Use Staging Push**" checkbox is selected, the other checkboxes must not be selected, unless you are creating a DPF dSource.
    c. If you are working with a HADR setup and want to ingest the database on a staging using HADR configurations, you need to check the **Link against source using HADR** checkbox and uncheck the **Customer Supplied Archive Logs** checkboxes. However, if you want to setup a dSource using HADR configurations using Staging Push, you need to only select the **Staging push** checkbox and handle restore and HADR configuration manually.
    d. To use a non-HADR method for database ingestion, you need to uncheck the **Link against the source using HADR** and **Customer Supplied Archive Logs** checkboxes.
    e. To use the **Customer Supplied Archive Logs** method for database ingestion check the **Customer Supplied Archive Logs** checkbox and uncheck the **Link against source using HADR** checkbox and enter the path in **Archive Log Path.**
    f. To use the **Database partition feature** check **Use DPF** checkbox and uncheck the **Link against the source using HADR** checkbox. For more information on DPF please view IBM Db2 Overview.



6. The **database name** is mandatory and must be unique for a given instance. This is the name that the database was on the instance it was restored from.
7. If the user is planning to create a dSource with the "**Staging Push**" feature then the user must need to select the "**Use Staging Push**" checkbox.
8. Enter the complete **Backup Path** where the database backup file resides. If no value is entered, the default value used is the instance home directory. If there are multiple backup files for a non-DPF database on the backup path, the most current one will be used. If there are multiple sets of backup files for a DPF database on the backup path, the most recent one will be used.
9. If the backup files are split across multiple filesystems, you can enter multiple backup paths by dynamically adding an **Additional Backup Path** field by clicking on the **Add** button. Make sure to enter the path containing the first part of the backup under the **Backup Path** field and the subsequent paths in **Additional Backup Paths**.
10. Optionally, you can add the **Parallelism factor for Database restore** to provide the degree of parallelisation to run the database restore operation. If no value is entered, the Db2 plugin inherently assigns parallelism factor.

    Info:

    a. Parallelism factor depends on the resources provided on the staging host, you may not see any difference in performance if there aren't enough resources for Db2 plugin to implement the parallelism.

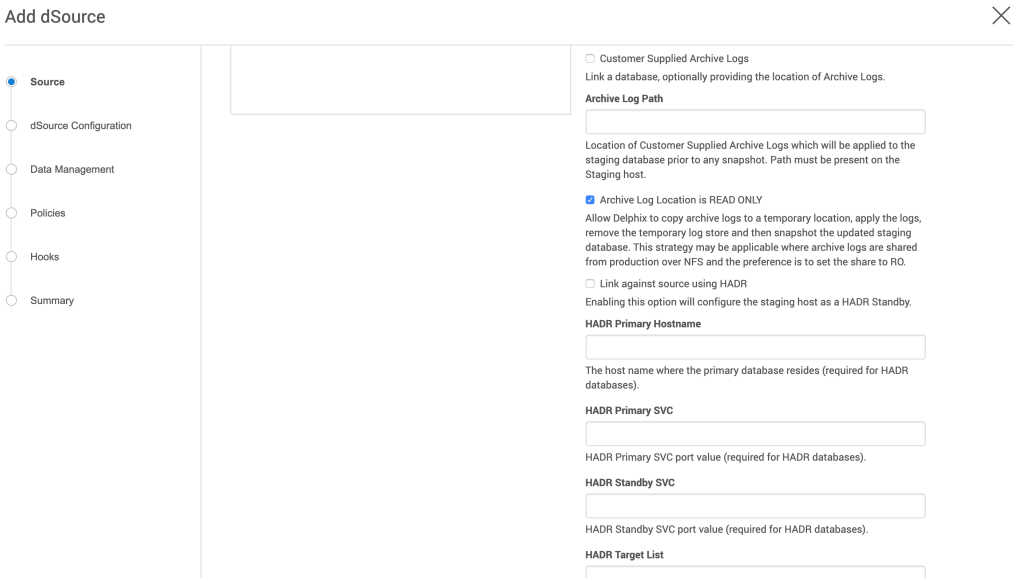
   b. You can provide any positive integer value for parallelism factor. Db2 plugin will provide that level of parallelism given that the system has appropriate resources to offer.
   c. Parallelism factor is inversely proportional to the Database restore time, which implies that a higher parallelism factor will result in reduced restoration time.
   d. For optimal performance during restoration, make sure to provide the same parallelism factor while creating the backups.
11. If the user wants to roll forward a snapshot to a user-specified timestamp, then the user must select the **Rollforward to PIT**checkbox. Users can also manually do this through the snapshot option.
    Info: The user-specified timestamp needs to be provided along with the timezone. For example, 2021-11-14-04.15.00 **UTC**. If the timezone is not entered, then the user's local timezone is considered.
12. Optionally, users can set the database configuration parameters during the linking operation in the **Config Settings**section.
13. Optionally, users can provide a bash script as a **Plugin**defined hook. Delphix has two Plugin managed Hooks:
    a. **Plugin Pre-Restore Hook:** The bash script entered in this field will get executed before restoration. This hook is useful in such cases where the users want to prepare the full backup before the restore operation.
    b. Plugin Post-Restore Hook: The bash script entered in this field will get executed after completion of the restore operation. This hook is useful in such cases where the user wants to purge the full backup file after the restore operation.

```
1. Plugin managed pre-Hook - User can copy the required full backup file
from some remote location to staging host. Below is an example of such bash
script (name of below script is copyBackup.sh)
#!/bin/bash


# Copying backup file from a remote host remote.delphix.com
scp auto1051@remote.delphix.com:/backuplocation/
R74D105A.0.auto1051.DBPART000.20180510073639.001 /db2backup/


The user can provide the above script in Plugin managed pre-Hook field as "s
h copyBackup.sh"


2. Plugin managed post-Hook - User can purge the full backup file from the
staging host. Below is an example of such bash script (name of below script
is purgeBackup.sh)
#!/bin/bash


# Purging full backup file from staging host for saving the staging storage
rm -f /db2backup/R74D105A.0.auto1051.DBPART000.20180510073639.001


The user can provide the above script in Plugin managed post-Hook field as "
sh purgeBackup.sh"
```

14. If the dSource is to use **HADR** please enter the following fields. For more information about HADR please view Linking a dSource from a Db2 Database: An Overview
    a. Enter a fully qualified HADR Primary Hostname. This is a required field for HADR and must match the value set for HADR_LOCAL_HOST on the master.

b. Enter the port or /etc/services name for the HADR Primary SVC. This is a required field for HADR and uses the value set for HADR_LOCAL_SVC on the master.
c. Enter the port or /etc/services name for the HADR Standby SVC. This is a required field for HADR and uses the value set for HADR_REMOTE_SVC on the master.

15. If you are using **Customer Supplied Archive Logs**for dSource creation complete the following fields:
    a. Check the Customer Supplied Archive logs checkbox
    b. Provide the log location of the archive logs. In this location, users can load copy files by appending schema and table name to it so that plugin can process those load copy files during log apply process. Archive logs will be used for applying the logs during the snapshot operation. This location must be a part of the staging server and this location must be different than full backup location.



    c. The **Archive Log Location is READ ONLY**(default) option indicates whether Delphix should copy the log files to the dSource's filesystem prior to the log apply process. If this is not selected the staging database will read the log files directly from the user-provided log location and perform the log apply operation.
    Note: Special considerations for Archive Log Location is READ ONLY
    If this operation is performed on the source database and user want the Plugin to process the load files place the load copy files in your location and append the schema and table names (for example: <DB Name>.<Number>.<Instance>.DBPART<partNumber>.<Timestamp>.001.<schema name>.<table name>) to the load copy file names
    If you do not want to process load copy files via the Plugin place the load copy files in a similar structure as it is on the source or set a DB2 registry parameter DB2LOADREC
    Note: Special consideration during dSource upgrade from older version plugin to Customer Supplied Archive Logs feature enabled plugin
    Suppose the user is having a dSource with an older version of the plugin and now wants to upgrade that dSource to **Customer Supplied Archive Logs** feature then the user needs to perform the following steps :
        i. Go to the configuration tab of dSource and then click on the custom sub-tab.
        ii. Click on the edit icon.
        iii. Check the **Customer Supplied Archive Logs** and **Archive Log Location is READ ONLY** checkbox.
        iv. Click on submit and then go for RESYNC operation to upgrade the dSource.

16. Click **Next**.

17. Enter a name and select a group for your dSource.
    Adding a dSource to a dataset group lets you set Delphix Domain user permissions for that database and its objects, such as snapshots. See the topics under Users and Groups for more information.
18. Click **Next** and set the **Staging Environment** to be the same as the dSource host. Select the **Staging Environment User** to be the same as the instance owner of the dSource instance.
    Info: Changing the Environment User
    If you need to change or add an environment user for the dSource instance, see Managing Db2 Users and Instance
    Info: Ensure that the Staging environment must be the environment that you have chosen to create the source config.
    Select the **User** from the drop-down and set the **SNAPSHOT PARAMETERS** for the dSource snapshot.
    a. Click on the **Timestamp** input field to enter the timestamp at which the snapshot will be taken. The user-specified timestamp needs to be provided along with the timezone in YYYY-MM-DD-HH.MM.SS format. For example, 2022-03-12-04.15.00**UTC**. If the timezone is not entered, then the user's local timezone is considered.
       Providing the timestamp will only work if the user has checked the **Rollforward to PIT** option in step 9 above.
    b. Select **Resynchronize dSource** to resynchronize the dSource. This will force a non-incremental load of data from the source similar to when the dSource was created. This action avoids duplicating storage requirements and maintains a Timeflow history.
19. The Delphix Engine will initiate two jobs to create the dSource: DB_Link and DB_Sync. You can monitor these jobs by clicking Active Jobs in the top menu bar, or by selecting System > Event Viewer. When the jobs have completed successfully, the database icon will change to a dSource icon on the Environments > Host > Databases screen, and the dSource will also appear in the list of Datasets under its assigned group. Then, click **Next** and you'll get the Policies section. Set the desired **Snapsync Policy** for the dSource. For more information on policies see Advanced Data Management Settings for Db2 dSources
20. Click **Next** and specify any desired pre- and post-scripts. For details on pre- and post-scripts, refer to Customizing Db2 Management with Hook Operations
21. Click **Next**. Review the dSource Configuration and Data Management information in the Summary section.
22. Click **Submit**.

The dSource configuration screen
After you have created a dSource, the **dSource Configuration tab** allows you to view information about it and make modifications to its policies and permissions. In the **Datasets** panel, select the dSource you wish to examine. You can now choose the configuration tab to see information such as the **Source files, Data Management** configuration, and **Hook Operations**.

Purging of archive logs after snapshot operation

Logs can be purged from your log location after the snapshot operation. The Plugin will preserve the list of logs that are eligible for purging in the snapshot's metadata. The snapshot metadata can be accessed via the Delphix CLI or API.

Data management settings for Db2 dSources

This topic describes advanced data management settings for dSources.

When linking a dSource, you can use custom data management settings to improve overall performance and match your specific server and data environment's needs. If no specific settings are required, leverage default data management settings.

Accessing data management settings

There are three ways to set or modify data management settings for dSources:

1. During the dSource linking process, in the Policies tab of the Add dSource wizard, select the **Retention Policy**.
2. Or select **Manage**, then select **Policies**.
3. In the Policies screen, select the **Retention** tab. To manage retention on target after the dSource snapshots have been deleted from the replication source, select the **Replica Retention** tab.
4. To add a new retention policy click the **+Retention** button, and to add a new replica retention policy on the replication target, click the **+Replica Retention** button.
5. This will open the Retention or the Replica Retention Policy wizard, depending upon the above selection. Enter a policy name along with required retention details and click **Submit**. For more information, see Policies for Scheduled Jobs

Retention policies - retention/replica retention

Retention policy defines the length of time Delphix Engine retains snapshots within its storage. To support longer retention times, you may need to allocate more storage to the Delphix Engine. The retention policy – in combination with the SnapSync policy – can significantly impact the performance and storage consumption of the Delphix Engine.

Replica Retention policies define how long the snapshots are retained on replicated namespaces for dSources and VDBs after they have been deleted on the replication source. Normally, the snapshots that have been deleted on the replication source engine are also deleted on the replication target engine. A new retention policy is introduced to provide an extended lifetime of such snapshots on the replication target.

Benefits of longer retention

With increased retention time for snapshots and logs, you allow a longer (older) rollback period for your data.

Common use cases for longer retention include:

- SOX compliance
- Frequent application changes and development
- Caution and controlled progression of data
- Reduction of project risk
- Restore to older snapshots

Backup source database

Source database with Raw DEVICE type storage

Several users use raw device-based tablespaces for source DB2 databases. To leverage these environments with Delphix, Delphix has built a workflow using Db2s native tools that allow Delphix to discover and convert a raw device-based tablespace into an automatic storage-based tablespace during ingestion. Once the data is ingested into staging, customers will be able to provision VDBs of the automatic storage-based database.

In order to complete the linking process, the Standby dSource must have access to a full backup of the source Db2 databases on disk. This should be a compressed online DB2 backup and must be accessible to the dSource instance owner on disk. Delphix is currently not setup to accept DB2 backups taken using third-party sources such as Netbackup or TSM. All three features of data ingestion, namely HADR, Non-HADR, and Customer Supplied Archive Logs backups must also include logs. Starting with Db2 plugin version 3.1.1, we are also supporting full online backups, which are taken using the "exclude logs" syntax. This support is only applicable to the dSources created using the Customer Supplied Archive Logs (Backup and Logs) ingestion mechanism on a single partition.

Example backup command: db2 backup database <DB_NAME> online compress include logs

Best practices for taking a backup

The following best practices can help improve backup and restore performance:

1. Compression should be enabled
2. Following parameters should be optimally configured:
    a. Utility Heap Size (UTIL_HEAP_SZ)
    b. No. of CPUs
    c. No. of Table Spaces
    d. Extent Size
    e. Page Size
3. Parallelism & Buffer configuration may be used to improve backup performance. Parameters that should be configured are :
    a. Parallelism
    b. Buffer Size
    c. No. of Buffers

More information about backup best practices is available in IBM Knowledge Center

Procedure

1. Login to the Delphix Management Application using Delphix Engine credentials or as the owner of the database from which you want to provision the dSource.
2. On the **Databases** tab of Environment Management screen, add a source config against discovered staging instance.
3. Then, click **Manage**.
4. Select **Datasets**.
5. Click the Plus (+) icon and select **Add dSource**, you'll get a list of available source configs using which you can go for dsource creation.
6. In the **Add dSource** wizard, select the required source configuration.
7. If you are working with an HADR setup, please leave the HADR checkbox checked unless you're working with Staging push dSource.
8. The **database name** is mandatory and must be unique for a given instance. This is the name that the database was on the instance it was restored from.

9. Enter the complete **Backup Path** where the database backup file resides. If the field is kept blank, the instance home directory will be used as the default value. If there are multiple backup files for a database on the backup path, the latest one will be used. If the backup files are split across multiple filesystems, you can provide them as additional backup paths with the first path being populated in the **Backup path** field.
10. Enter the **Log Archive Method1** you wish to use for the database. If no value is entered, the default value used is DISK:/mountpoint/dbname/arch
11. Optionally, users can set the database configuration parameters during the linking operation in the **Config Settings** section.
12. If the dSource is to use HADR please enter the following fields. If it will not use HADR skip ahead to step 13. For more information about HADR please view Linking a dSource from a Db2 Database: An Overview
    a. Enter a fully qualified HADR Primary Hostname. This is a required field for HADR and must match the value set for HADR_LOCAL_HOST on the master.
    b. Enter the port or /etc/services name for the HADR Primary SVC. This is a required field for HADR and uses the value set for HADR_LOCAL_SVC on the master.
    c. Enter the port or /etc/services name for the HADR Standby SVC. This is a required field for HADR and uses the value set for HADR_REMOTE_SVC on the master.
13. Click **Next**.
14. Select a **dSource Name** and **Database Group** for the dSource.
15. Click **Next**. You will get Data Management section where you need to specify staging environment and user which will be used for dsource creation.
16. Set the **Staging Environment** to be the same as the dSource host.
17. Select the **Staging Environment User** to be the same as the instance owner of the dSource instance. **Info**: Changing the Environment User
    If you need to change or add an environment user for the dSource instance, see Managing Db2 Users and Instance Owners
18. Then, click **Next** and you'll get the Policies section. Set the desired **Snapsync Policy** for the dSource.
19. Click **Next**.
20. Specify any desired pre- and post-scripts. For details on pre- and post-scripts.
21. Click **Next**.
22. Review the dSource Configuration and Data Management information in the summary section.
23. Click **Submit**.

The Delphix Engine will initiate two jobs to create the dSource: DB_Link and DB_Sync. You can monitor these jobs by clicking Active Jobs in the top menu bar, or by selecting System > Event Viewer. When the jobs have completed successfully, the database icon will change to a dSource icon on the Environments > Host > Databases screen, and the dSource will also appear in the list of Datasets under its assigned group.

> **The dSource configuration screen**
> After you have created a dSource, the **dSource Configuration tab** allows you to view information about it and make modifications to its policies and permissions. In the **Datasets** panel, select the dSource you wish to examine. You can now choose the configuration tab to see information such as the **Source files, Data Management** configuration, and **Hook Operations**.

Working with Db2 snapshots

Taking a snapshot creates a new snapshot entry in the dSource's Timeflow. The plugin-managed dSource allows you to select the parameters before taking the snapshot. These parameters are provided as an input to pre-snapshot and post snapshot functions. On the other hand, in the default snapshot, the parameters are pre-defined in the dSource environment.

Snapshot (Default)

This section lists the steps to take a snapshot and delete the same.

1. Login to the **Delphix Management** application.
2. Click **Manage** and select **Datasets** from the dropdown list.
3. Select the dSource you want to Snapshot.
4. Click the **Camera** icon. Alternatively, click the arrow next to the Camera icon and select **Snapshot (default)**.
5. From the Snapshot dialog, select **Perform Snapshot**.
6. Click **View:** under **Timeflow** to verify the Snapshot you just created.
7. To delete, select the snapshot you just created, and from the Actions menu (...) select **Delete Snapshot**.
8. In the Delete Snapshot dialog, select **Delete**.
9. Click **View:** under **Timeflow** to verify the Snapshot, you just deleted.

Source sizing implementation for datasets

The Source Sizing feature aligns the data source experience with Delphix database standards that will improve your ingestion reporting experience on the per TB pricing model. This feature enables you to calculate the dataset size based on the total allocated space on the mount point provided by the Delphix Engine for the dataset.

Procedure

- For all datasets, the size of the dataset can be calculated using `du -sb <dataset>` command that provides the correct volume and apparent size of the datasets.
- For AIX hosts, the size of the dataset can be calculated by executing any one of the following commands depending on whether the database is open for connections or not.
    a. `select sum(TBSP_TOTAL_SIZE_KB)*1024 from sysibmadm.TBSP_UTILIZATION,` if the database is open for connections and it is possible to connect to the database.
    b. `db2dart <dbname> /DTSF`, if the database is not open for connections and is in rollforward pending state.

- For AIX hosts, the plugin won't be able to display the size of the database during the creation of dSource, since it needs to be restored to read the size of the database.
- If the plugin is unable to connect to the database or the database is yet to be restored, then the status tab under dSource displays **Not available**.

Provisioning and managing VDBs from Db2 dSources

we do not support Migrate dataset option for DB2

> ⚠ We currently **do not** support Migrate dataset option for DB2.

This section covers the following topics:

Provisioning Db2 VDBs: An overview

This topic describes the basic concepts involved with provisioning VDBs from DB2.

A dSource is a virtualized representation of a physical or logical source database. As a virtual representation, it cannot be accessed or manipulated using database tools. Instead, you must create a virtual database (VDB) from a dSource snapshot. A VDB is an independent, writable copy of a dSource snapshot. You can also create VDBs from other VDBs. Once you have provisioned a VDB to a target environment, you can also implement snapshot and retention policies for the VDB, which will determine how frequently Delphix Engine will take a database snapshot and how long the snapshots will be retained for recovery and provisioning purposes. For an overview of the high-level components involved in provisioning a DB2 VDB refer to Setting Up Db2 Environments: An Overview

Snapshots accumulate over time. To view a snapshot:

1. From the Datasets panel, click the group containing the dSource.
2. Select dSource.
3. Click the TimeFlow tab.

The TimeFlow appears as a list of dates, each of which expands to show snapshots from that date. Times, when the VDB has been refreshed, are marked by a blue line between dates.

On the TimeFlow, you can also filter by type of snapshot. To do so, click the filter button, which is shaped like an eye.

You can scroll through these lists to select the one you want, or you can enter a date and time to search for a specific snapshot.

Once you have provisioned a VDB, you can also take snapshots of it. As with the dSource snapshots, you can find these when you select the VDB in the Datasets panel. You can then provision additional VDBs from these VDB snapshots.

Dependencies

If there are dependencies on the snapshot, you will not be able to delete the snapshot free space; the dependencies rely on the data associated with the snapshot.

Provisioning a Db2 VDB

This topic describes how to provision a virtual database (VDB) from a DB2 dSource.

Prerequisites

- You will need to have linked a dSource from a staging instance, as described in Linking a Db2 dSource, or have created a VDB from which you want to provision another VDB
- You should have set up the Db2 target environment with necessary requirements as described in Requirements for Db2 Hosts and Databases
- Make sure you have the required Instance Owner permissions on the target instance and environment as described in Managing Db2 Users and Instance Owners
- The method for Database Permissions for Provisioned Db2 VDBs is decided before the provisioning
- [Optional] Once you have restored the databases into your target environment, If the version of IBM® Db2® in your source environment is at a different fix pack level than the version of Db2 in your target environment, then you need to update the 11.1 version databases with `db2updv111` command and 11.5 databases with `db2updv115` command utility. Run the below command as a Pre-snapshot hook:

```
db2updv111 -d <DBNAME> -a
db2updv115 -d <DBNAME> -a
```

ⓘ The optional pre-requisite is applicable only when the target environment is using a different Db2 fix pack level than the source. If your source environment is using version 11.1.x and your target is using version 11.5.x, you can neglect the above-mentioned point.

Consider the following scenarios for the optional prerequisite mentioned above:

1. if your source environment uses Db2 version 11.5.5 and your target environment uses Db2 11.5.8 or a later fix pack, you need to update the databases in the target environment.
2. If your databases were created or upgraded to Db2 Version 11.1 GA and you applied Db2 Version 11.1 Fix Pack 1 or later, then running the above command will automatically apply all updates required from Version 11.1 GA up to and including the fix pack level that you are installing.

ⓘ
1. To know how to update databases, see db2updv115 - Update database to Version 11.5 mod pack command
2. Certain database or instance operations are restricted while an online fix pack update is in progress.
3. See db2updv111 - Update database to Version 11.1 fix pack command, which mentions it is mandatory to run db2updv111 on database if one is coming from higher version of fix pack to Lower OR vice versa.
4. For more information, you can also refer: Update the databases for a Db2 11.1.x fix pack

ⓘ You can take a new snapshot of the dSource by clicking the **Camera** icon on the dSource card. Once the snapshot is complete you can provision a new VDB from it.

Please note while creating a VDB on an instance By default, DB2 Plugin will not allow provisioning a VDB on an instance which contains a database that has an identical name to the source database.

For e.g.;

1. When we are provisioning a VDB from a VDB then source VDB will be treated as the source database.
2. When we are provisioning a VDB from a dSource then a staging database will be treated as the source database.

If the user wants to provision a VDB on an instance which contains a source database, then the user needs to follow the below steps :

1. Check if any config file with name db2_plugin.conf at location <Toolkit directory>/Delphix_COMMON/plugin/<Identifier>/. If it does not exist, then the user should create a file with name db2_plugin.conf at location <Toolkit directory>/Delphix_COMMON/plugin/<Identifier>/. The instance user must have read permission to this file. Once the file is created, user should proceed to step 2. If the file already exists at this location with the required permission, the user should directly proceed to step 2.
2. Add the below line in this config file.
    a. allowSourceDbOnSameInstance=True
3. Then create/Refresh the VDB again.

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select a **dSource**.
5. Select a **snapshot** from which you want to provision.
6. Click **Provision VDB** icon to open Provision VDB wizard.
7. Select a target environment from the left pane.
8. Select an **Installation** to use from the dropdown list of available DB2 instances on that environment.
9. Set the **Environment User** to be the Instance Owner. Note: The picking of instance owner is only possible if you have multiple environment users set on that host.
10. Provide VDB Name as database name as parameter.
11. Optionally, provide the VDB Mount Base for the VDB.
12. Optionally, set the database configuration parameters for the VDB.
13. Click **Next**.
14. Select a **Target Group** for the VDB. Click the green **Plus** icon to add a new group, if necessary.
15. Select a **Snapshot Policy** for the VDB.
16. Click **Next**.
17. Specify any desired hook operations.
18. Click **Next**.
19. Review the **Provisioning Configuration** and **Data Management** information.
20. Click **Submit**. When provisioning starts, you can review the progress of the job in the **Databases** panel, or in the **Job History** panel of the **Dashboard**. When provisioning is complete, the VDB will be included in the group you designated and listed in the **Databases** panel. If you select the VDB in the Databases panel and click the **Open** icon, you can view its card, which contains information about the database and its Data Management settings. Once the VDB provisioning has successfully completed, if the source and target instance ids are not the same, you may want to grant secadm and dbadm on the database to the target instance id. Refer to Database Permissions for Provisioned Db2 VDBs for more information.

Database permissions for provisioned Db2 VDBs

This topic describes the database permissions on provisioned Db2 virtual instances

DB2 authentication

Authentication is the process of validating a supplied user ID and password using a security mechanism. User and group authentication is managed in a facility external to Db2 LUW, such as the operating system, a domain controller, or a Kerberos security system. This is different from other database management systems (DBMSs), such as Oracle and SQL Server, where user accounts may be defined and authenticated in the database itself, as well as in an external facility such as the operating system.

Any time a user ID and password is explicitly provided to Db2 LUW as part of an instance attachment or database connection request, Db2 attempts to authenticate that user ID and password using this external security facility. If no user ID or password is provided with the request, Db2 implicitly uses the user ID and password that were used to login to the workstation where the request originated. More information on Db2 authentication and authorization is available via IBM documentation

Delphix Db2 Authentication Delphix for Db2 requires that the staging and target hosts must already have the necessary users and authentication systems created/installed on them. Delphix will neither create users nor change database passwords as part of the provisioning process.

While the terminology used within the Delphix Management application refers to a VDB, the ingestion, snapshot and provisioning process for Db2 on Delphix always occurs on the Database level. Thus when a virtual Db2 database is provisioned by Delphix, it contains the Db2 database that were in the source instance with the identical user permissions as they had on the source. This means that if the target instance name is different from the source instance name then that instance owner will NOT have DBADM or SECADM permissions unless they were specifically granted to that instance owner on the source instance. The instance owner will however always have SYSADM permissions on all databases in the instance.

LDAP authentication

If your Db2 instances and applications use LDAP authentication then they will work seamlessly as long as LDAP had been configured on the VDB Target Instance.

OS authentication

If your DB2 instances and applications are using OS authentication then it is important to ensure that the relevant OS accounts exist on the target machine.

Generic accounts

If the Db2 applications are using generic (non-instance owner) accounts, they will then be able to continue using them as long as those OS accounts exist on the host machine. It is important to note that the passwords for the same account may be different on different hosts, or if they use different LDAP servers (i.e. prod vs dev LDAP servers).

Instance owner usage

If the Db2 applications typically use the instance owner accounts to access data then it is important to note that if the new instance name is different from the source instance name then that instance owner will NOT have DBADM or SECADM permissions in the VDB instance and may not be able to read and write data. In such a case there are three possible workarounds:

1. Connect to the databases as the source instance owner. This requires that the target host has a UNIX account with the same name as the source instance.

2. Grant the necessary permissions to the VDB instance owner on the source instance before taking the snapshot that is to be provisioned.
3. Grant the permissions to a known "delphix" user on the source instance and then use that account to grant the permissions to the target VDB instance after provisioning.

Upgrading Db2 VDBs

This topic describes how to upgrade a Db2 VDB to a higher version of Db2.

Before upgrading the VDB, please refer to the Delphix Db2 support matrix to ensure that you upgrade to the supported Db2 versions only.

Procedure for VDB in-place upgrade

1. Remove any VDB Refresh Policy assigned to the VDB.
2. Upgrade the target Db2 instance.
3. Refresh the target environment.

Provisioning from a replicated Db2 VDB

This topic describes how to provision from a replicated dSource or virtual database (VDB).

The process for provisioning from replicated objects is the same as the typical VDB provisioning process, except that first you need to select the replica namespace containing the replicated object.

Prerequisites

- You must have a dSource or VDB that has been replicated from one Delphix Engine to another, as described in Replication Overview
- The Delphix Engine containing the replicated dSource or VDB must have a compatible target environment that it can use to provision a VDB from the replicated dSource or VDB.

Procedure

1. On the Delphix Engine containing the replicated dSource or VDB, login to the **Delphix Management** application.
2. In the top menu bar, click **Manage**.
3. Select **Datasets**.
4. From the list of replica namespaces, select the **replica namespace** that contains the dSource or VDB from which you want to provision.
5. The provisioning process is now identical to the process for provisioning standard objects.

Post-requisites

Once the provisioning job has started, the user interface will automatically display the new VDB in the live system.

Db2 hook operations

Shell operations

RunCommand operation

The RunCommand operation runs a shell command on a Unix environment using whatever binary is available at `/bin/sh`. The environment user runs this shell command from their home directory. The Delphix Engine captures and logs all output from this command. If the script fails, the output is displayed in the Delphix Management application and command-line interface (CLI) to aid in debugging.

If successful, the shell command must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Examples of RunCommand operations

You can input the full command contents into the RunCommand operation.

```
remove_dir="$DIRECTORY_TO_REMOVE_ENVIRONMENT_VARIABLE"

if test -d "$remove_dir"; then
    rm -rf "$remove_dir" || exit 1
fi

exit 0
```

If a script already exists on the remote environment and is executable by the environment user, the RunCommand operation can execute this script directly.

```
/opt/app/oracle/product/10.2.0.5/db_1/dbs/myscript.sh "$ARG_ENVIRONMENT_VARIABLE"
"second argument in double quotes" 'third argument in single quotes'
```

RunBash operation

The RunBash operation runs a Bash command on a Unix environment using a `bash` binary provided by the Delphix Engine. The environment user runs this Bash command from their home directory. The Delphix Engine captures and logs all output from this command. If the script fails, the output is displayed in the Delphix Management application and command-line interface (CLI) to aid in debugging.

If successful, the Bash command must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Example of RunBash operations

You can input the full command contents into the RunBash operation.

```
remove_dir="$DIRECTORY_TO_REMOVE_ENVIRONMENT_VARIABLE"

# Bashisms are safe here!
```

```
if [[ -d "$remove_dir" ]]; then
    rm -rf "$remove_dir" || exit 1
fi

exit 0
```

Shell operation tips

Using `nohup`

You can use the `nohup` command and process backgrounding from resource in order to "detach" a process from the Delphix Engine. However, if you use `nohup` and process backgrounding, you MUST redirect `stdout` and `stderr.`

Unless you explicitly tell the shell to redirect `stdout` and `stderr` in your command or script, the Delphix Engine will keep its connection to the remote environment open while the process is writing to either `stdout` or `stderr` . Redirection ensures that the Delphix Engine will see no more output and thus not block waiting for the process to finish.

For example, imagine having your `RunCommand` operation background a long-running Python process. Below are the bad and good ways to do this.

Bad examples

- `nohup python file.py & # no redirection`
- `nohup python file.py 2>&1 & # stdout is not redirected`
- `nohup python file.py 1>/dev/null & # stderr is not redirected`
- `nohup python file.py 2>/dev/null & # stdout is not redirected`

Good examples

- `nohup python file.py 1>/dev/null 2>&1 & # both stdout and stderr redirected, Delphix Engine will not block`

Other operations

RunExpect operation

The RunExpect operation executes an Expect script on a Unix environment. The Expect utility provides a scripting language that makes it easy to automate interactions with programs which normally can only be used interactively, such as `ssh` . The Delphix Engine includes a platform-independent implementation of a subset of the full Expect functionality.

The script is run on the remote environment as the environment user from their home directory. The Delphix Engine captures and logs all output of the script. If the operation fails, the output is displayed in the Delphix Management application and CLI to aid in debugging.

If successful, the script must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Example of a RunExpect operation

Start an `ssh` session while interactively providing the user's password.

```
spawn ssh user@delphix.com
expect {
    -re {Password: } {
        send "${env(PASSWORD_ENVIRONMENT_VARIABLE)}\n"
    }
    timeout {
        puts "Timed out waiting for password prompt."
        exit 1
    }
}
exit 0
```

DB2 environment variables

Operations that run user-provided scripts have access to environment variables. For operations associated with specific dSources or virtual databases (VDBs), the Delphix Engine will always set environment variables so that the user-provided operations can use them to access the dSource or VDB.

dSource environment variables

Environment variable	Description
`DLPX_DATA_DIRECTORY`	The primary mount path provided by the user on the UI. The first one, in case multiple mount specifications are provided.
`USER`	The OS user used to create the dSource.

VDB Environment Variables

Environment variable	Description
`DLPX_DATA_DIRECTORY`	The primary mount path provided by the user on the UI. The first one, in case multiple mount specifications are provided.
`USER`	The OS user used to provision the VDB.

Enhanced logging for Db2 Plugin

DB2 plugin v3.x

This feature adds three options to provide enhanced logging:

1. To set debug log levels
2. To set log retention levels
3. To set trace size limit

Each host (added as an environment to DE) will have its own property file **<Toolkit directory>/DB2/plugin-logging.properties.** Delphix users can modify the parameters within this property file to configure the logging mechanism.

Setting debug log levels

Delphix users can set log levels for plugin generated logs by setting the parameter ***level***. Higher logging levels will help to expedite debugging issues. There are three levels of logging which are: Level 1 (info), Level 2 (debug) and Level 3 (trace). All the objects (both dSources and VDBs) present on a staging/target host will refer to a single property file. For example; if we have 4 dSources on a host associated with a Delphix Engine then the log level applies to all the dSources.

Log level description:

- level=1: This level will print only informational logs.
- level=2: This level will print informational logs and debug statements. This is the default log level.
- level=3: Besides informational and debug logs, this level will enable the traces.

Setting log retention levels

Delphix users can set a retention level for diag logs (.diag.log)="" using="" parameter="">***retention*** in *the* file. As per this parameter, the log files are moved (archived), renamed or deleted once they are too old or too big. New incoming log data is directed into a new fresh file (at the same location).

By default, this value will be set to a minimum value of 2. The user can change this value and set its value within the range 2 and 50. For example; if ***retention*** is set to 4, the plugin will have the following log files: *<DB Name>.diag.log, <DB Name>.diag.log.1, <DB Name>.diag.log.2, <DB Name>.diag.log.3, <DB Name>.diag.log.4.*

- *File <DB Name>.diag.log is the active log file.*
- File <DB Name>.diag.log.1 is the most recent archive log file
- File <DB Name>.diag.log.4 is the oldest one

Setting trace size limit

In case trace level logging is enabled, users can set the size of the trace folder using parameter **traceFolderLimit**. By default, the value of this parameter will be 10, which signifies that the size limit for this folder will be 10 MB. This parameter is configurable and the user can set its value to any positive integer. The traces are stored within a separate folder for traces created under directory **<Toolkit directory>/DB2/logs/<User>/.** The filename format for traces would be <DB Name>.<sciprt name>.trace.<epoch time>.

If the file is accidentally removed from the location, the plugin will generate the property file **<Toolkit directory>/DB2/plugin-logging.properties** with default values at runtime. For example, when the user will upload the plugin to DE, the first call to log function will check for the existence of **<Toolkit directory>/DB2/plugin-logging.properties**. If the plugin does not find this property file, it will create a property file with the content shown

below. The logs will start generating per the level defined inside the file. Below is the sample **<Toolkit directory>/DB2/plugin-logging.properties** file:

plugin-logging.properties

```
# This flag will set the debug level of logging. There are three levels:

# 1: Logging will be in info mode.

# 2: Logging will be in Debug mode.

# 3: Traces will be enabled along with debug level logging.

# The default value is 2.

# level=<positive integer>

level=2

# Whenever the size of <DB Name>.diag.log file exceeds 1MB the plugin will rename the
active log file to  <DB Name>.diag.log.<number> and a new log file with name <DB
Name>.diag.log will be generated.

# For example, if LogRetention is set to 4, the plugin will have the following  log
files: <DB Name>.diag.log, <DB Name>.diag.log.1, <DB Name>.diag.log.2, <DB
Name>.diag.log.3, <DB Name>.diag.log.4.

# File <DB Name>.diag.log.4 will be the oldest one.

# File <DB Name>.diag.log.1 will be the most recent archive log file.

# File <DB Name>.diag.log will be the active log file.

# For example, if LogRetention is set to 4, the plugin will have the following  log
files: <DB Name>.diag.log, <DB Name>.diag.log.1, <DB Name>.diag.log.2, <DB
Name>.diag.log.3, <DB Name>.diag.log.4.

# The default value for retention flag is 2, minimum value is 2 and maximum value is
50.

# retention=<positive integer>

retention=2

# The value for this property is configurable and must be a positive integer.

# The default value is 10.

# traceFolderLimit=<positive number>

traceFolderLimit=10
```

DB2 plugin v4.x and onwards

This feature adds three options to provide enhanced logging:

1. To set debug log levels.
2. To set active log file size.
3. To set log retention levels.

Delphix users can modify the parameters defined under plugin_logging_parameters section header within the db2_plugin.conf file located at **<Toolkit directory>/<Delphix_COMMON>/plugin/DB2_18f4ff11-b758-4bf2-9a37-719a22f5a4b8/db2_plugin.conf** to configure the logging mechanism.

Setting log levels (level)

Delphix users can set log levels for plugin generated logs by setting the parameter ***level*** under plugin_logging_parameters header. Higher logging levels will help to expedite debugging issues. There are two levels of logging which are: Info and Debug. All the objects (both dSources and VDBs) present on a staging/target host will refer to a single property file. For example; if we have 4 dSources on a host associated with a Delphix Engine then the log level applies to all the dSources.

Log level description:

- level=INFO: This level will print only informational logs. This is the default log level.
- level=DEBUG: This level will print informational logs and debug statements.

Setting maximum size of active log file (logFileSize)

Delphix users can set this parameter under the plugin_logging_parameters header of the db2_plugin.conf file to set the maximum size of the active log file in MB. Once this limit is reached, the plugin will rotate the log as per the retention property defined below. The minimum value of this parameter is 1 MB and maximum value is 10 MB. This parameter only takes in a positive integer value. The default value is 1 MB.

Setting log retention levels (retention)

Delphix users can set a retention level for diag logs (.diag.log)="" using="" parameter="">***retention*** under the plugin_logging_parameters header. As per this parameter, the log files are moved (archived), renamed or deleted once they reach the value set in logFileSize parameter. New incoming log data is directed into a new fresh file (at the same location).

By default, this value will be set to a minimum value of 2. The user can change this value and set its value within the range 2 and 50. For example; if ***retention*** is set to 4, the plugin will have the following log files: *<DB Name>.diag.log, <DB Name>.diag.log.1, <DB Name>.diag.log.2, <DB Name>.diag.log.3, <DB Name>.diag.log.4.*

- *File <DB Name>.diag.log is the active log file.*
- File <DB Name>.diag.log.1 is the most recent archive log file
- File <DB Name>.diag.log.4 is the oldest one

db2_plugin.conf

```
#
# The parameters must not be left padded with spaces. The plugin will otherwise not
accept the specified parameter
# values and will use the default values.
#

[plugin_custom_parameters]
#
# If the user has not shared a common group between primary environment user and
instance user.
```

```
# Then the user needs to set parameter usersHaveCommonGroup as False.
#

usersHaveCommonGroup=True

#
# During provision operations the plugin checks if the target instance contains a
database name which is identical to
# the source DB name from which the VDB is to be created. By default the plugin will
never allow creating a VDB on the
# instance where the source database already exists, or on any instance that has a
database with the same name. If the
# users still want to create a VDB on an instance which contains a database with the
same name as the source DB then
# they need to set parameter allowSourceDbOnSameInstance as True.
#

allowSourceDbOnSameInstance=False

#
# If the users want to control the number of parallel restores then they can tune the
parameter restorePipelineLimit.
# By default the value of this parameter would be 10.
#

restorePipelineLimit=10

#
# The archive logs validation process can be skipped if the end user so chooses.
Instead, a speedier technique can be used,
# in which the archive logs are only validated for the first and last logs and the
file sequences in between are confirmed.
#

skipArchiveLogValidation=True

[plugin_logging_parameters]
#
# This flag will set the debug level of plugin logs on the remote server (both
staging and target). There are two levels:
# Info
# Debug
# The above are the only valid values that can be assigned. If any other value is
assigned, the plugin will set the
# default level as Info. the string is case insensitive so info, Info and INFO are
acceptable.
#

level=INFO

#
# This parameter will set the maximum size of the active log file in MB. Once this
 limit is reached, the plugin will
```

```
# rotate the log as per the retention property defined below. The minimum value of
this parameter is 1 MB and maximum
# value is 10 MB. This parameter only takes in a positive integer value. The default
 value is 1 MB
# logFileSize=<positive integer>
#

logFileSize=1

#
# Whenever the size of <DB Name>.diag.log file exceeds the value provided by
logFileSize parameter then the plugin will
# rename the active log file to <DB Name>.diag.log.<number> and a new log file with
name <DB Name>.diag.log will be
# generated. For example, if LogRetention is set to 4, the plugin will have the
following log files: <DB Name>.diag.log,
# <DB Name>.diag.log.1, <DB Name>.diag.log.2, <DB Name>.diag.log.3, <DB
Name>.diag.log.4.
# File <DB Name>.diag.log.4 will be the oldest one.
# File <DB Name>.diag.log.1 will be the most recent archive log file.
# File <DB Name>.diag.log will be the active log file.
# The minimum and default value for retention flag is 2.
# retention=<positive integer>
#

retention=2
```

MySQL environments and data sources

This section covers the following topics:

- Quick start guide for MySQL
- MySQL prerequisites
- MySQL connector installation
- MySQL environment discovery
- Linking Data Sources with MySQL
- Provisioning VDBs from MySQL dSources
- Upgrading a dSource,VDB after a MySQL database upgrade
- MySQL references

To view the MySQL support matrix, see MySQL matrix.

Quick start guide for MySQL

MySQL connector has been developed to ingest and virtualize MySQL data sources using Delphix.

What does MySQL connector do?

The Delphix MySQL connector can virtualize a single instance MySQL database and create multiple virtual copies as required. There are two modes of operation of the connector based on how the MySQL dSource is created and kept in-sync with source database.

Replication with bin log

In the Replication with binlog mode, Delphix uses MySQL's built-in replication with binlog technology to keep the Delphix dSource in-sync with the MySQL source databases. Creation of the dSource requires a full backup of the source database(s) and there are two options to provide this backup.

1. Delphix Managed Backup In this option, Delphix is provided with the required information and privileges to take a full backup of the source database.
2. User Provided Backup In this option, user provides Delphix with an existing full backup. Delphix initializes the staging database using the full backup and configures the staging database as a replicated slave of the source database. MySQL Replication with binlog keeps the data in the staging database in-sync with the source.

Manual Ingestion

In the Manual Ingestion mode, the data in Delphix dSource is managed manually by the end users. Delphix creates a seed MySQL staging database which can be managed via Delphix. The end user assumes the responsibility of keeping the data in the staging database in-sync with the source database.

Where to start?

By now, you must have an overall idea of what is possible with the MySQL plugin. Before you can get started with virtualizing your MySQL databases, there are some pre-requisites you need to take care of. Please visit the Pre-Requisites page for more details.

Questions?

If you have questions, bugs or feature requests reach out to us via the MySQL Github or at Delphix Community Portal.

MySQL prerequisites

General prerequisites

Given below are the general pre-requisites for MySQL virtualization. Environments For MySQL virtualization, Delphix requires only Staging and Target Hosts to be added as environments. Refer to Delphix Docs for configuration best practices. ...

Updated on : 25 May 2023

Environment requirements for replication with binlog as an ingestion mechanism

Given below are the pre-requisites for MySQL virtualization when using Replication with binlog mode. Source Environment Requirements Source environment is where the source MySQL databases are running. Connectivity Delphix staging user must be...

Updated on : 25 May 2023

Environment requirements for manual ingestion

Given below are the pre-requisites for MySQL virtualization when using Manual Ingestion mode. Source Environment Requirements Source environment is where the source MySQL databases are running. Source DB User In Manual Ingestion mode, Delphix...

Updated on : 25 May 2023

General prerequisites

Given below are the general pre-requisites for MySQL virtualization.

Environments

For MySQL virtualization, Delphix requires only Staging and Target Hosts to be added as environments.

Refer to Delphix Docs for configuration best practices.

Staging Host/Environment

The MySQL connector is a "Staged" connector - which means that in order to create a dSource, Delphix requires a staging environment. This environment must have MySQL binaries installed and the version of MySQL must match the source MySQL database(s).

Target Host/Environment

If you are familiar with Delphix, you must already know what a Target Host is. It is simply another host with MySQL binaries installed where Delphix creates virtual databases. Again, the version of MySQL must match your source database(s).

Storage

- Empty folder on Staging & Target host to hold delphix toolkit [approximate 2GB free space]
- If Delphix is managing backups, the Staging Host must have enough storage space to hold your source database backup.

MySQL Version

- MySQL Binary version on Staging and Target must match the version on the source database(s)

Environment requirements for replication with bin log as an ingestion mechanism

Given below are the pre-requisites for MySQL virtualization when using Replication with binlog mode.

Source environment requirements

Source environment is where the source MySQL databases are running.

Connectivity

Delphix staging user must be able to connect to source environment from staging and take a backup of the source database(s) using the *mysqldump* utility.

Source DB user

- A Source DB user with the following permissions.
    - Can connect to the source database from staging host as well as locally. This user is required to take backup of the source database.

```
mysql>CREATE USER 'delphix_os'@'<staging_host>' IDENTIFIED BY 'delphix_user_passwd';
```

```
mysql>CREATE USER 'delphix_os'@'localhost' IDENTIFIED BY 'delphix_user_passwd';
```

- Has at the minimum, the following permissions on the source database(s).

SELECT, SHUTDOWN, SUPER, RELOAD ,SHOW VIEW, EVENT, TRIGGER, REPLICATION CLIENT,REPLICATION SLAVE

```
mysql>GRANT SELECT, SHUTDOWN, SUPER, RELOAD ,SHOW VIEW, EVENT, TRIGGER, REPLICATION
CLIENT,REPLICATION SLAVE on *.* to 'delphix_os'@'staging-host';
```

```
mysql>GRANT SELECT, SHUTDOWN, SUPER, RELOAD ,SHOW VIEW, EVENT, TRIGGER on *.* to
'delphix_os'@'localhost';
```

You can also grant more permissive privileges

```
mysql>GRANT ALL PRIVILEGES ON *.* TO 'user'@'%';
```

> **Note**
> Remember, this is the user that Delphix uses to manage the Staging database. So, it is recommended that you create a dedicated source db user for Delphix with the privileges mentioned above.

Binary logging

- In order to set up replication, binary logging should be enabled on the source database. You can check the status of binary logging as follows

```
mysql> SHOW VARIABLES LIKE 'log_bin';
```

If binary logging is enabled, you should see the following status

```
    +---------------+-------+
    | Variable_name | Value |
    +---------------+-------+
    | log_bin       | ON    |
    +---------------+-------+
```

Server-Id

The source database must have a non zero server-id and the server-id value should be different than the server-id value on the source.

Staging environment requirements

Staging OS user

This is a typical Delphix OS staging user. Key requirements for this user are given below.

Please refer to Delphix Docs for more detailed requirements.

- A Delphix OS user with the elevated permissions to run ps, mount, umount, mkdir, rmdir commands without requiring a password. Below is an example where delphix_os will be used for the mounting purpose.

```
/etc/sudoers
Defaults:delphix_os !requiretty
delphix_os ALL=NOPASSWD: \
/bin/mount, /bin/umount, /bin/mkdir, /bin/rmdir, /bin/ps
```

- Delphix OS user should be in the same primary and secondary groups as mysql user (or the MySQL binary owner)
- Delphix OS user must have execute access on all files within MySQL installation folder - Min permission level 775 recommended.

Storage

- Staging Host must have enough storage space to hold the source backup file.
- Empty folder on staging host to hold delphix toolkit [approximate 2GB free space]

MySQL version & configuration

- MySQL Binary version must match the version on the source database(s)
- [Recommended] As every organization's MySQL configuration is different, User can place a starter my.cnf file to be present in Delphix Toolkit Directory when creating a staging database. Delphix will use this my.cnf file and modify it as per the configuration provided during the dsource creation process. This is recommended to reduce the possibility of errors while restoring the backup from the source database. However, having the my.cnf in the toolkit directory is not mandatory. Connector has the ability to create my.cnf file on its own.

Target environment requirements

Target OS user

This is a typical Delphix OS target user. Key requirements for this user are given below.

Please refer to Delphix Docs for more detailed requirements.

- A Delphix OS user with the elevated permissions to run ps, mount, umount, mkdir, rmdir commands without requiring a password. Below is an example where delphix_os will be used for the mounting purpose.

```
/etc/sudoers

Defaults:delphix_os !requiretty
delphix_os ALL=NOPASSWD: \
/bin/mount, /bin/umount, /bin/mkdir, /bin/rmdir, /bin/ps
```

- Delphix OS user should be in the same primary and secondary groups as mysql user (or the MySQL binary owner)
- Delphix OS user must have execute access on all files within MySQL installation folder - Min permission level 775 recommended.

Done, what's next?

All the pre-requisites are now taken care of, next step is to install the connector.

Environment requirements for manual ingestion

Given below are the pre-requisites for MySQL virtualization when using Manual Ingestion mode.

Source environment requirements

Source environment is where the source MySQL databases are running.

Source DB user

In Manual Ingestion mode, Delphix does not interact with the source database. Delphix creates a staging database and the backups are ingested manually by a customer user. However, Delphix needs to be able to manage this staging database for snapshots and other time travel operations. Hence, we require a user with the following permissions on the staging database.

SELECT, SHUTDOWN, SUPER, RELOAD ,SHOW VIEW, EVENT, TRIGGER

This can be achieved by following:

1. Create this user on your source database so that when the backup is restored, this user is present on the staging database.
2. Plugin will attempt to create this user and assign the required privileges on the staging database during the dSource creation. Plugin will fail if it is not able to create this user or assign the required permissions on the staging DB. User needs to ensure that this user is always present in the staging database and has the necessary privileges. This is required as delphix will use this user to perform any time travel operations.

   ```
   PowerShell
   mysql>CREATE USER 'delphix_os'@'localhost' IDENTIFIED BY 'delphix_user_passwd';
   ```

3. Grant the necessary privileges to the user.

```
mysql> GRANT SELECT, SHUTDOWN, SUPER, RELOAD ,SHOW VIEW, EVENT, TRIGGER on *.* to
'delphix_os'@'localhost';
```

You can also grant more permissive privileges

```
mysql>GRANT ALL PRIVILEGES ON *.* TO 'user'@'%';
```

> **Note**
> Remember, this is the user that Delphix uses to manage the Staging database. So, you need to create this user on the source database.

Staging environment requirements

Staging OS user

This is a typical Delphix OS staging user. Key requirements for this user are given below.

- A Delphix OS user with the elevated permissions to run ps, mount, umount, mkdir, rmdir commands without requiring a password. In the example below permissions are provided to delphix_os user.

```
/etc/sudoers
Defaults:delphix_os !requiretty
delphix_os ALL=NOPASSWD: \
/bin/mount, /bin/umount, /bin/mkdir, /bin/rmdir, /bin/ps
```

- Delphix OS user should be in the same primary and secondary groups as mysql user (or the MySQL binary owner)
- Delphix OS user must have execute access on all files within MySQL installation folder - Min permission level 775 recommended.

Storage

- Empty folder on staging host to hold Delphix toolkit [approximate 2GB free space]

MySQL version & configuration

- MySQL Binary version must match the version on the source database(s)
- Starter my.cnf file [Recommended]
    - As every organization's MySQL configuration is different, User can place a starter my.cnf file to be present in Delphix Toolkit Directory when creating a staging database.
    - Delphix will use this my.cnf file and modify it as per the configuration provided during the the dsource creation process.
    - This is recommended to reduce the possibility of errors while restoring the backup from the source database.
    - Having a my.cnf in the toolkit directory is not mandatory, if this file is not provided, Delphix will create a my.cnf file from scratch.

Target environment requirements

Target OS user

This is a typical Delphix OS target user. Key requirements for this user are given below.

Please refer to Delphix Docs for more detailed requirements.

- A Delphix OS user with the elevated permissions to run ps, mount, umount, mkdir, rmdir commands without requiring a password. In the example below permissions are provided to delphix_os user.

```
/etc/sudoers
Defaults:delphix_os !requiretty
delphix_os ALL=NOPASSWD: \
/bin/mount, /bin/umount, /bin/mkdir, /bin/rmdir, /bin/ps
```

- Delphix OS user should be in the same primary and secondary groups as mysql user (or the MySQL binary owner)
- Delphix OS user must have execute access on all files within MySQL installation folder - Min permission level 775 recommended.

Done, what's next?

All the pre-requisites are now taken care of, next step is to install the connector.

MySQL connector installation

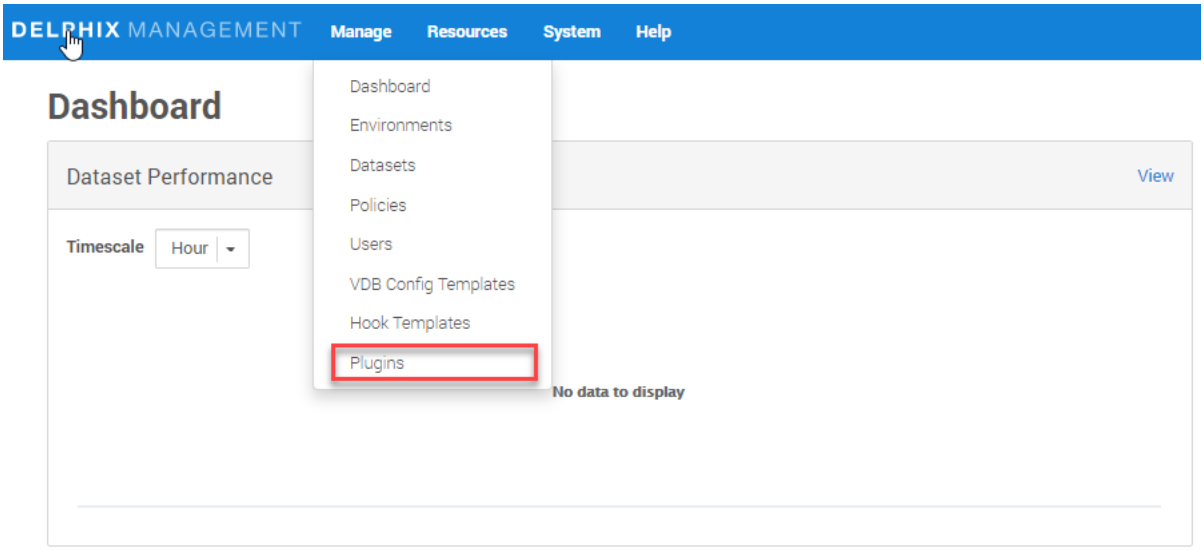
Prerequisites

- Delphix Engine of version 6.0.x
- Install MySQL Binaries on source, staging, and target servers

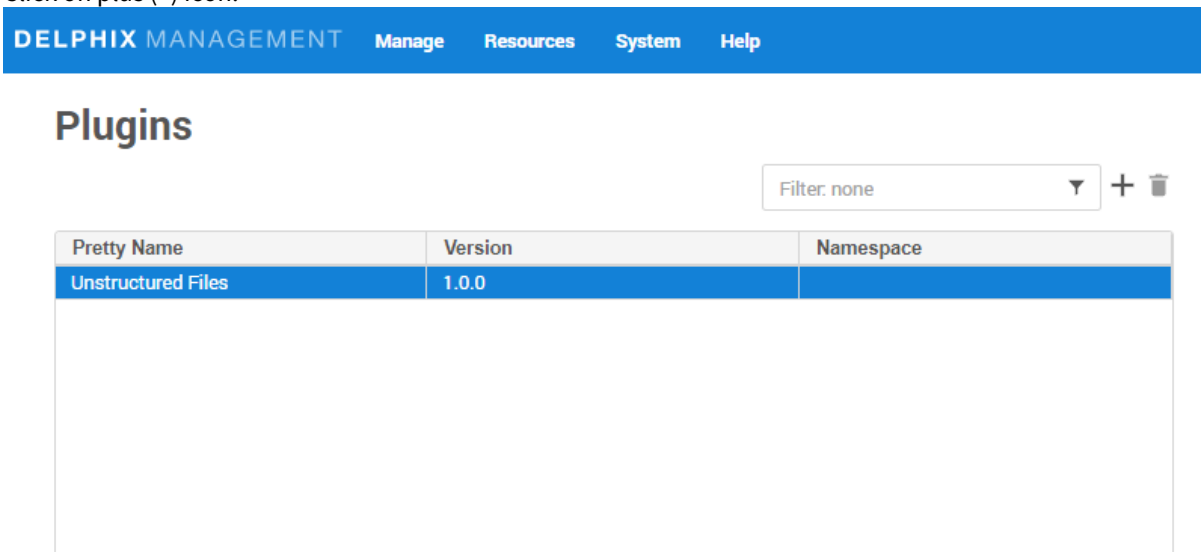
Installing the MySQL connector

Method 1: Using GUI

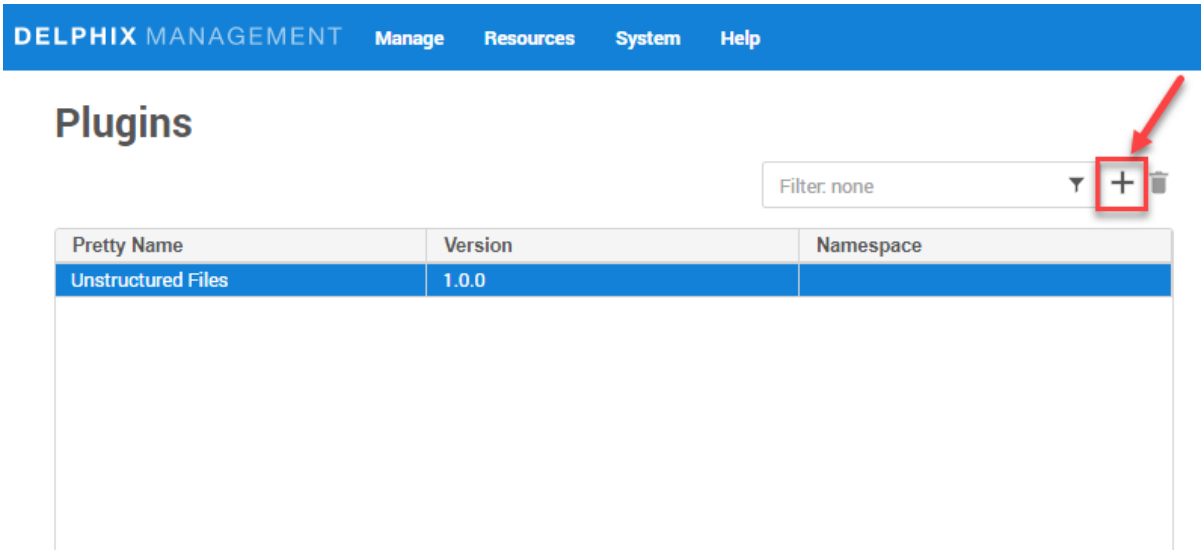
- Click on **Manage** (present on top-left of the below page) and then **Plugins**.



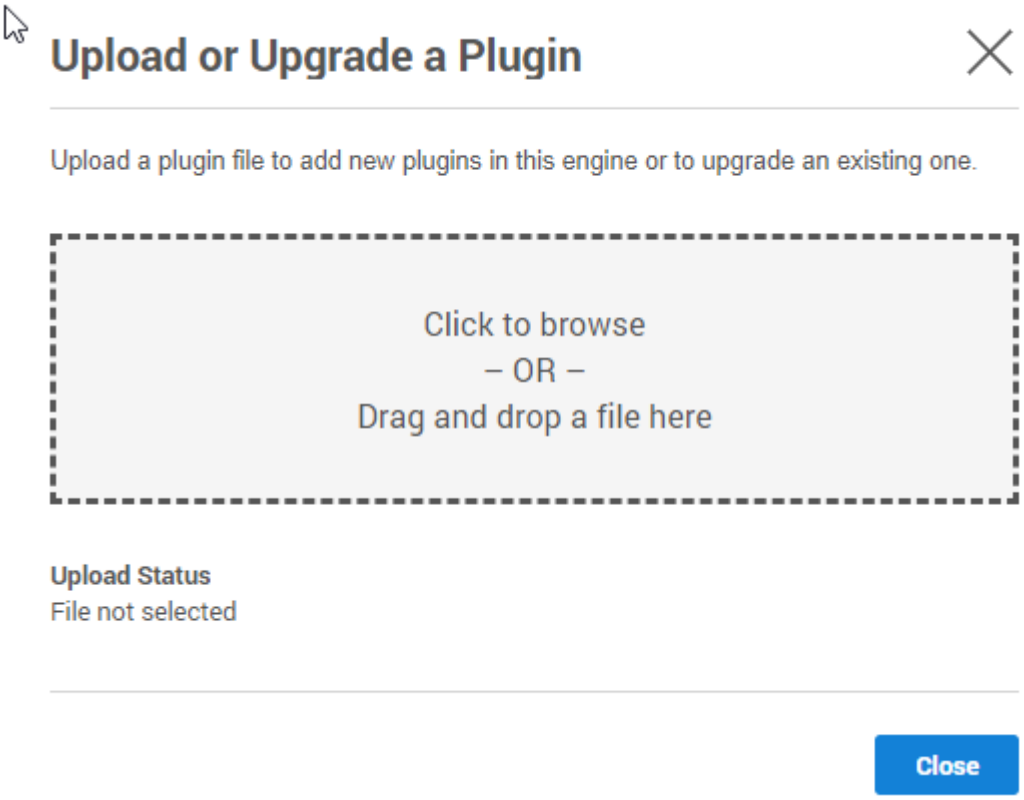
- Click on plus (+) icon.



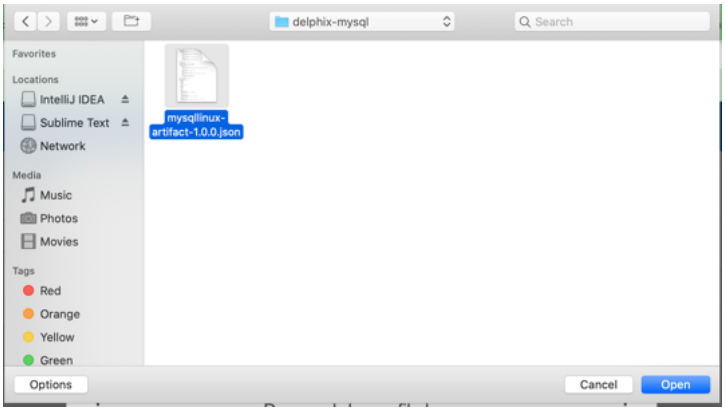
- Click on **Upload** a plugin.



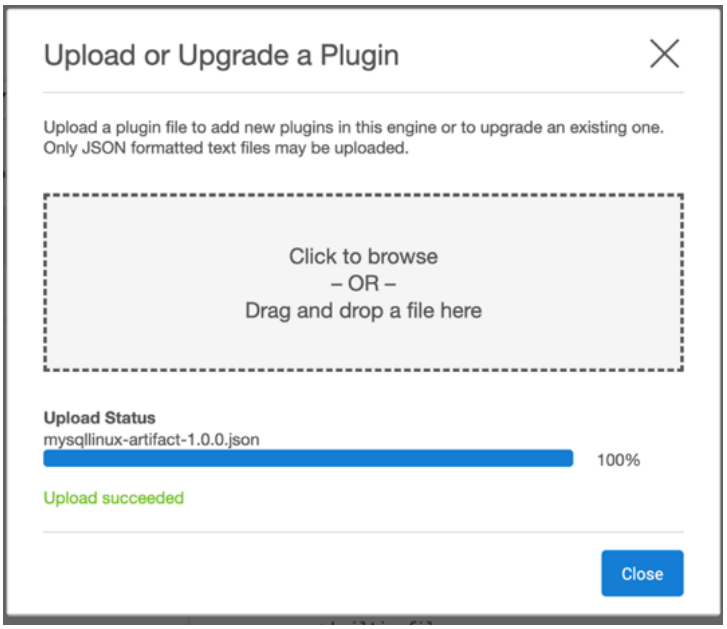
- Select the `build(artifacts.json)` from your device.



- Click on **Close** button.



- See the plugin version in **Plugins** section.



Method 2: using dvp command

```
dvp upload -e <Delphix_Engine_Name> -u <username> --password <password>
```

Delphix Engine's documentation on installing plugins: Plugin Management.

MySQL environment discovery

Environment discovery/refresh is a process that enables the MySQL Plugin to determine MySQL installation details on a host. Database discovery/refresh is initiated during the environment set up process.

Whenever there is any change (installing a new database home) to an already set up environment in the Delphix application, we need to perform an environment discovery/refresh.

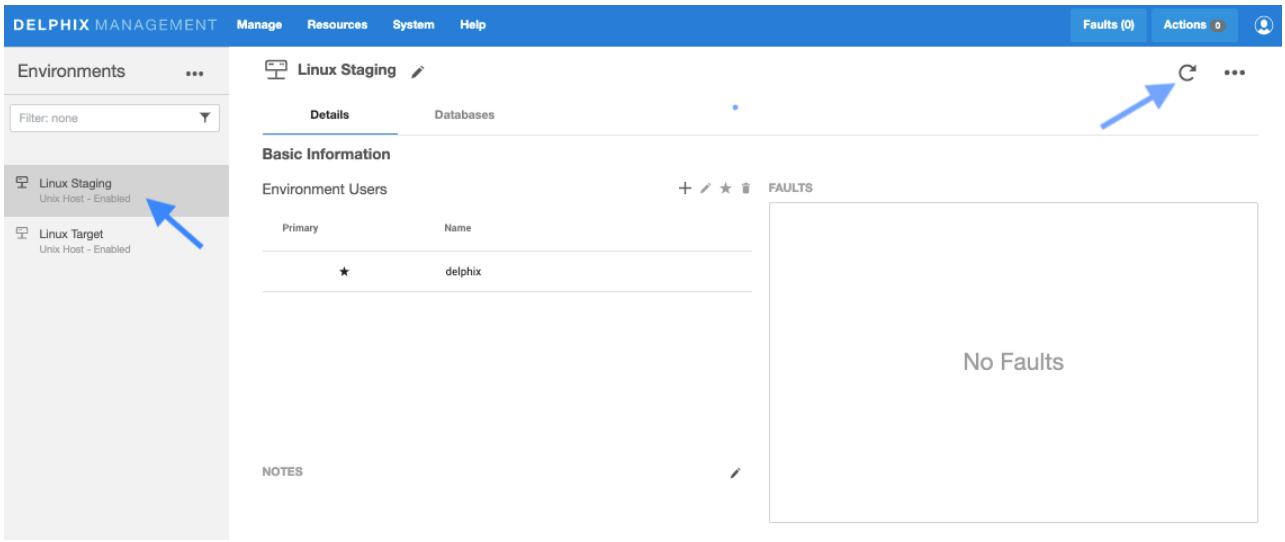
Prerequisites

Installation of the MySQL Plugin is required before the Discovery.

Refreshing an environment

Environment refresh will update the metadata associated with that environment and push Delphix Toolkit onto the host.

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. In the Environments panel, click the name of the environment you want to refresh.
5. Select the **Refresh** icon.
6. In the Refresh confirmation dialog select **Refresh**.



Once an environment refresh completes successfully, Delphix will discover all MySQL installations on the environment. These installations are referred to as "repositories".

Add source config

As noted above, environments contain `repositories`, that are MySQL installations in the environment. Each environment may have any number of repositories associated with it.

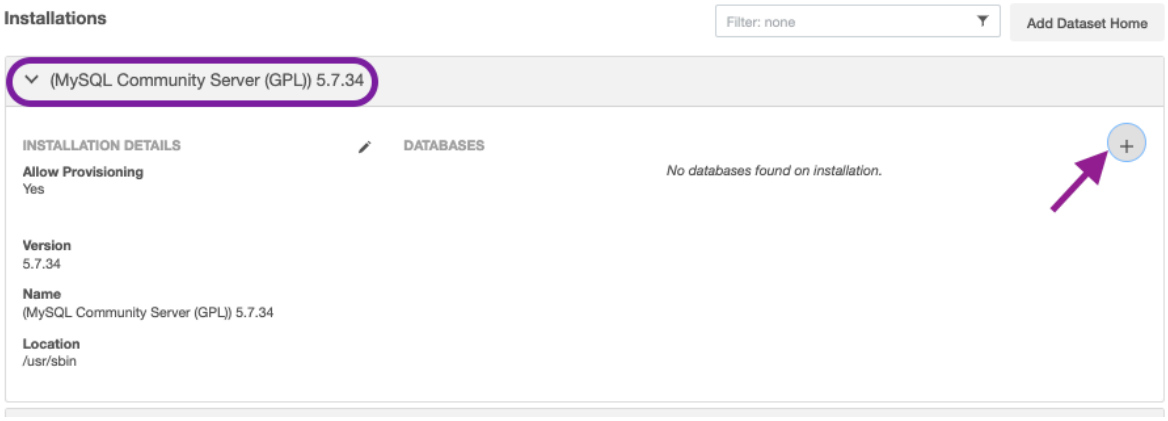
The next step in the virtualization process is to add a `SourceConfig`. A `SourceConfig` object defines the configuration of the dSource and is required to create a dSource. You can create any number of `SourceConfig` objects using a repository, which represents known database instances.

For the MySQL plugin, the Source config must be created manually.

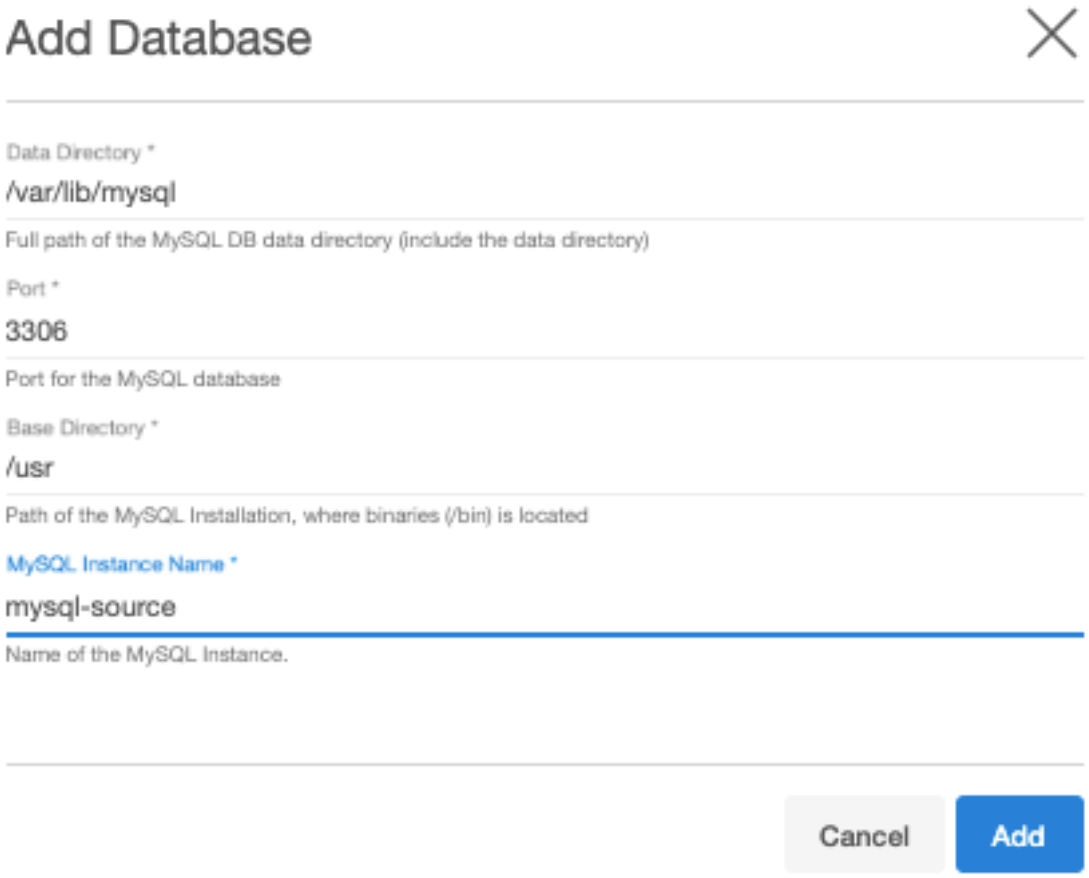
How to create source configs

> Note
> Source Config is created on the Staging Environment

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Select the repository.
5. Click on plus(+)icon.



6. Add required details in the **Add database** section.
   - Enter source data directory in section **Data Directory**.
   - Enter source port number in **Port** section.
   - Enter MySQL base directory on the source host in **Base Directory** section.
   - Enter dSource name in **MySQL dSource Nam**e section.



What's next?

Now that your environments are all added and a Source Config has been created, please proceed to Linking Data Sources with MySQL to see how to create a dSource.

Linking Data Sources with MySQL

Linking using replication with bin log mode

In Replication Mode, Delphix creates a staging database using an initial backup from the source database. This initial backup can be taken by Delphix or can be provided by the end user. In order to ensure that the staging database remains in-sync with the source database, Delphix can set up a MySQL native replication from the sourcedatabase to the stagingdatabase.

Pre-requisites

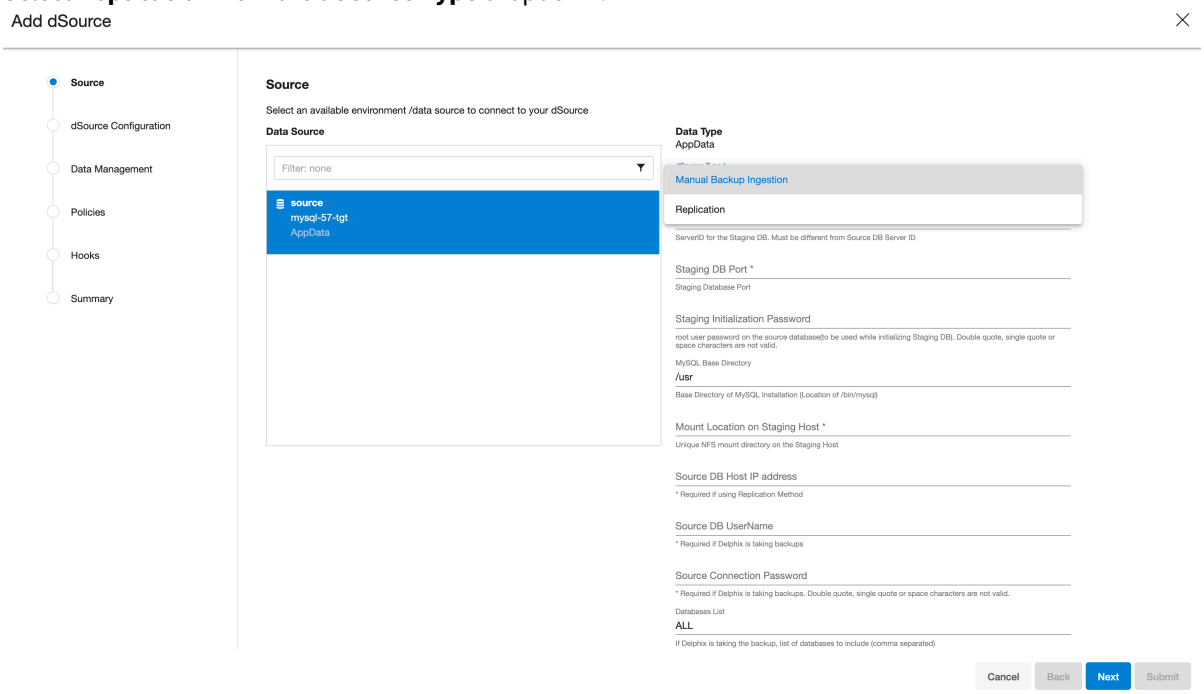
- Staging environment must be added to Delphix.
- A Source Config must be created on the staging environment-MySQL repository.

This is recommended to reduce the possibility of errors while restoring the backup from the source database.

> Warning
> It may take upto 5 minutes after successful dSource creation for the status to show as Active.

Creating dSource

1. Login to **Delphix Management** application.
2. Click **Manage > Datasets**.
3. Select **Add dSource**.
4. In the **Add dSource** wizard, select the MySQL source configuration which is created on the staging host.
5. Select **Replication** from the **dSource Type** dropdown.

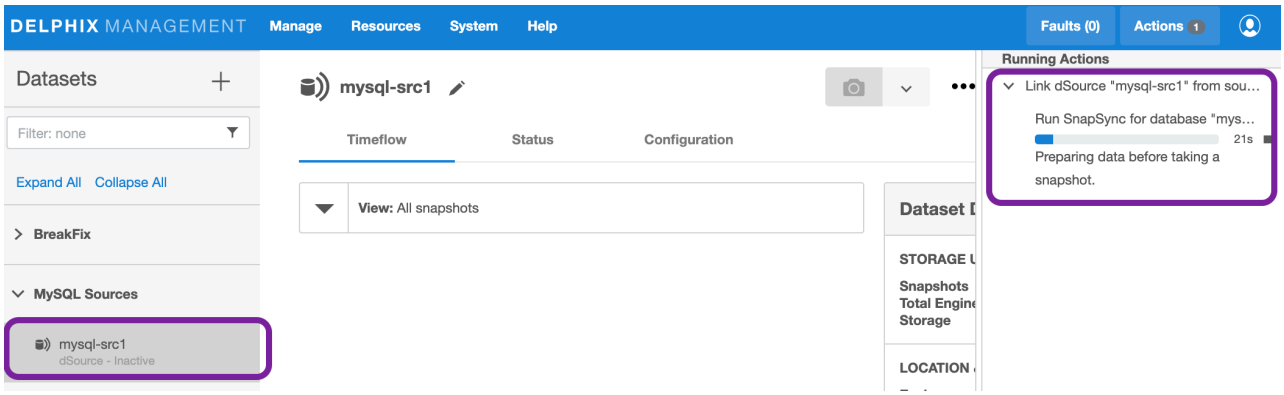


6. Provide the additional details required for dSource creation
    a. **Staging DB Server ID** Server ID for the dsource (staging db. For Replication Mode, this server id must be greater than the source db server id.
    b. **Staging DB Port** Port for the dsource (staging db). This port should not be used by any other application or MySQL server.
    c. **Staging Initialization Password** Password to use while initializing the dsource (staging db). This password will be assigned to the 'root'@'localhost'. We need to provide Password of the root user

over source. Delphix does not connect to the source db using the root user. Source root user password will be inherited for this instance once the data ingestion starts.

d. **MySQL Base Directory** MySQL installation directory. This generally has location of /bin/mysql
e. **Mount Location on Staging Host** This is the mount directory for Delphix on the staging host. This location should be unique and empty.
f. **Source DB Host IP Address** IP Address of the source db host.
g. **Source DB UserName** Delphix database user on the source database. This user will be created in the dsource when the initial backup is created or restored. If the backup generation over source is driven by Delphix, then Delphix will be using the Source DB USerName to connect to the source database. Delphix will be using this db user to manage the dSource and other time travel operations.
h. **Source DB Password** Password for the source database user.
i. **Databases List** Selective database replication is not supported. Leave blank or fill in as ALL which indicates that all databases on the source instance will be brought in.
j. **MySQL Backup Path** Full path, including the filename, of the full source db backup to use for the dsource creation. If this is provided, Delphix will not create a backup.
k. **LogSync** Whether to enable replication or not. If checked, Delphix will set up the dsource (staging db) as a replicated slave of the source database.
l. **Replication UserName** Delphix DB user with replication privileges to set up replication from source to staging db. Currently, Delphix uses the Source DB user for replication.
m. **Replication User Password** Password for the replication database user.

7. On **dSource Configuration** screen, select the dataset group where the dSource will be placed and click **Next**.
8. On the **Data Management** screen, select the staging environment and the environment user and click **Next**.
9. On the **Policies** screen, select the Snapsync and Retention policies for the dSource and click Next.
10. On the **Hooks** screen, add any pre-sync and (or) post-sync hooks as required and click Next.
11. Review the dSource configuration on the **Summary** screen and **Submit**. The Linking process has commenced.



Done, what's next?

You have created a MySQL dSource. Next step - Provision a VDB.

Linking using manual ingestion mode

In Manual Ingestion Mode, Delphix creates a seed (empty) staging database. Customer users will be responsible for manually ingesting data into this staging database. All other management operations can be performed through Delphix. A snapshot can be taken either manually or via a SnapSync policy once the data is restored into the seed staging database.

> Info
> While ingesting the data for Manual Backup Ingestion, make sure to ingest the data on the MySQL instance which has been created as a part of dSource creation.

Pre-requisites

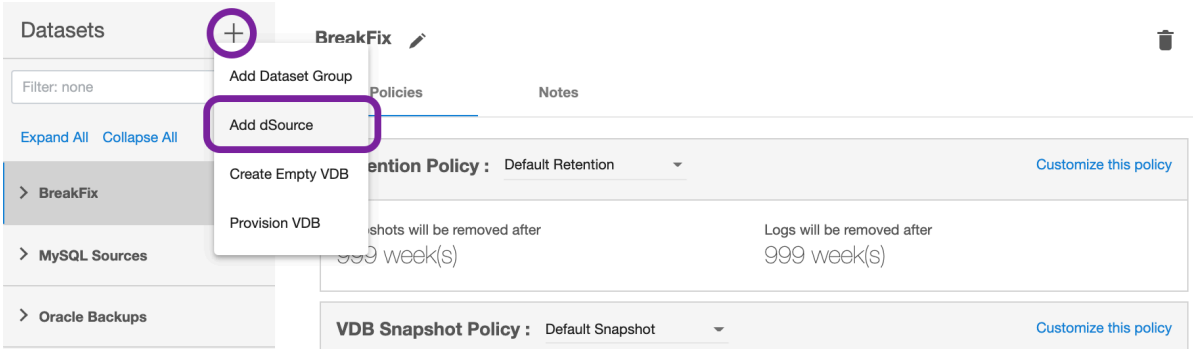
- Staging environment must be added to Delphix.
- A Source Config must be created on the staging environment-MySQL repository.

This is recommended to reduce the possibility of errors while restoring the backup from the source database.

> Warning
> It may take upto 5 minutes after successful dSource creation for the status to show as Active.

Creating dSource

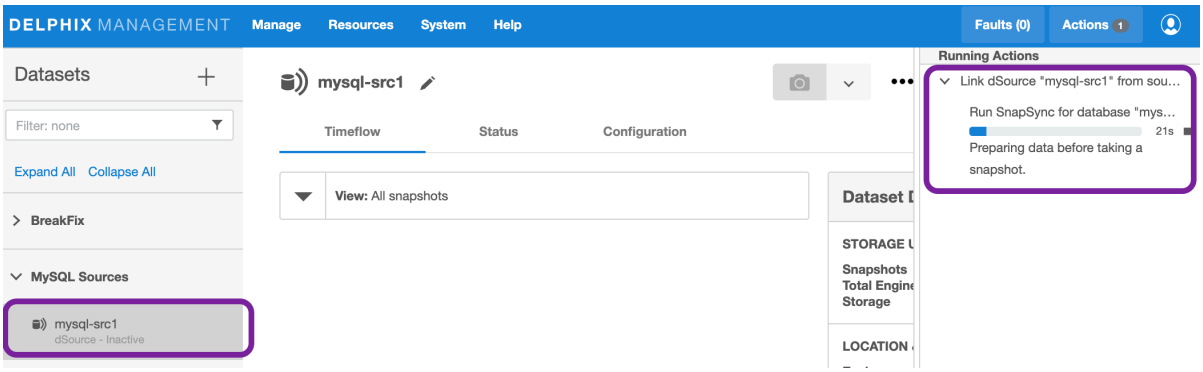
1. Login to **Delphix Management** application.
2. Click **Manage > Datasets**.
3. Select **Add dSource**.



4. In the **Add dSource** wizard, select the MySQL source configuration which is created on the staging host.
5. Select **Manual Ingestion** from the **dSource Type** dropdown.
6. Provide the additional details required for dsource creation
    a. **Staging DB Server ID**
    Server ID for the dsource stagingdatabase.
    b. **Staging DB Port**
    Port for the dSource (stagingdatabase). This port should not be used by any other application or MySQL server
    c. **Staging Initialization Password**
    Password to use while initializing the dSource (stagingdatabase). This password will be assigned to the `root'@'localhost`. We need to provide Password of the root user over source at the time of backup initiation. While Linking using the Manual Ingestion Mode, Delphix does not connect to the source db. Source root user password will be inherited for this instance once the data ingestion starts onto the staging instance.

d. **MySQL Base Directory** MySQL installation directory. This generally has location of /bin/mysql
e. **Mount Location on Staging Host** This is the mount directory for Delphix on the staging host. This location should be unique and empty.
f. **Source DB UserName**
Delphixdatabase user on the source database. In Manual Ingestion mode, Delphix does not connect to the sourcedatabase. Delphix will be using thisdatabase user to manage the dSource and other time travel operations. This user must be part of the sourcedatabase backup that will be restored into the stagingdatabase. Plugin will also attempt to create this user on the staging database and provide the necessary permissions.
g. **Source DB Password**
Password for the sourcedatabase user.

7. On **dSource Configuration** screen, select the dataset group where the dSource will be placed and click **Next**.
8. On the **Data Management** screen, select the staging environment and the environment user and click **Next**.
9. On the **Policies** screen, select the Snapsync and Retention policies for the dSource and click **Next**.
10. On the **Hooks** screen, add any pre-sync and (or) post-sync hooks as required and click **Next**.
11. Review the dSource configuration on the **Summary** screen and **Submit**. The Linking process has commenced.



Once the dSource creation is successful, the **Timeflow** tab should show the initial snapshot.

Done, what's next?

You have created a MySQL dSource. Next step - Provision a VDB.

Provisioning VDBs from MySQL dSources

Virtual Databases (VDBs) are virtualized copies of a dSource. A VDB can be created using a snapshot from the dSource timeflow.

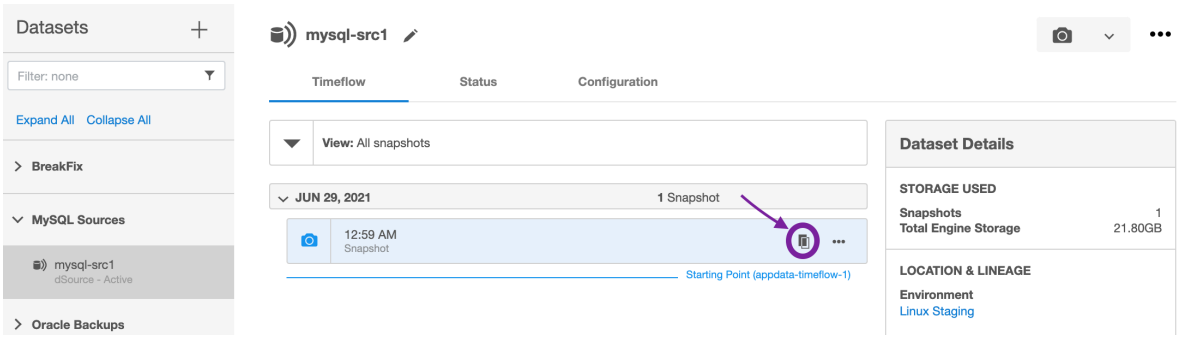
Prerequisites

- Require a linked dSource with at least 1 snapshot.
- Require a target environment added to Delphix.
- A MySQL binary with the same version as the source database must be installed on the Target environment.

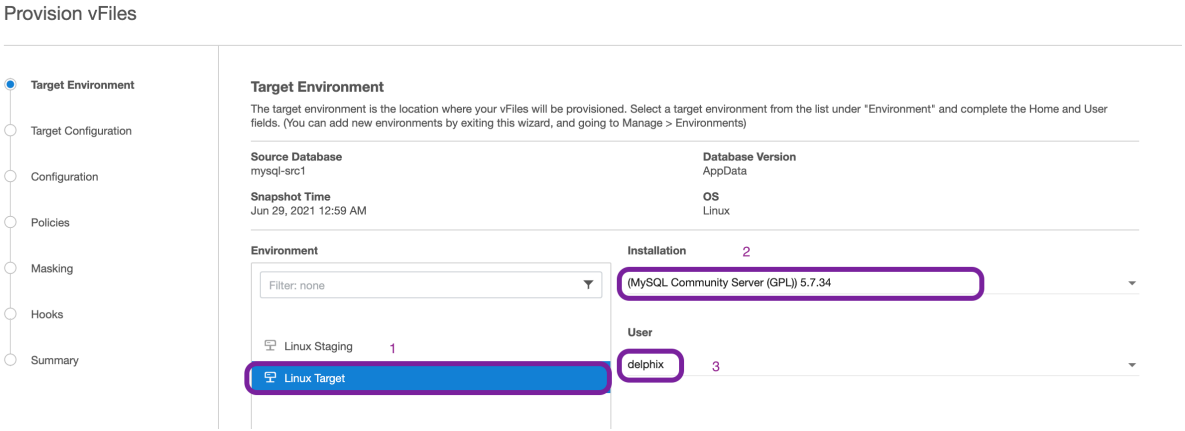
It may take up to 5 minutes after successful dSource creation for the status to show as Active.

Provisioning a VDB

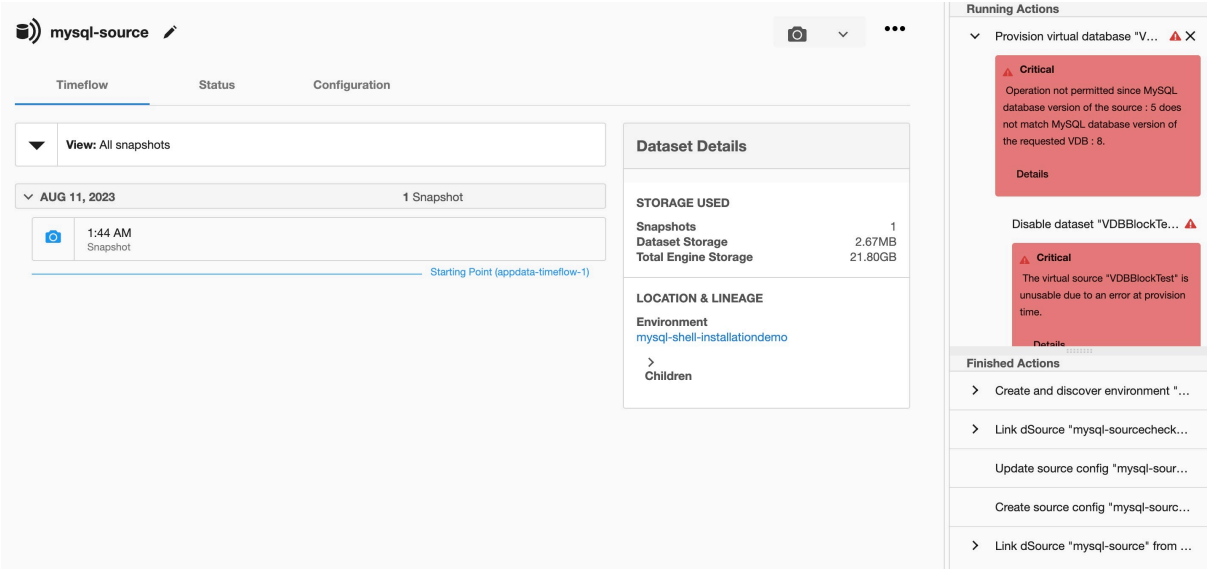
1. On the dSource Timeflow, click on the Provision action on the snapshot that you want to use for the VDB as shown below.



2. Select the target environment from the dropdown on which VDB needs to be created. If there are multiple MySQL repositories on the host, select the one that matches the source database version. If there are multiple OS users on the host, select the user you want to use.



3. The VDB target environment MySQL version should not equal to the MySQL version of the dSource environment. If not, you will see the below error in the UI

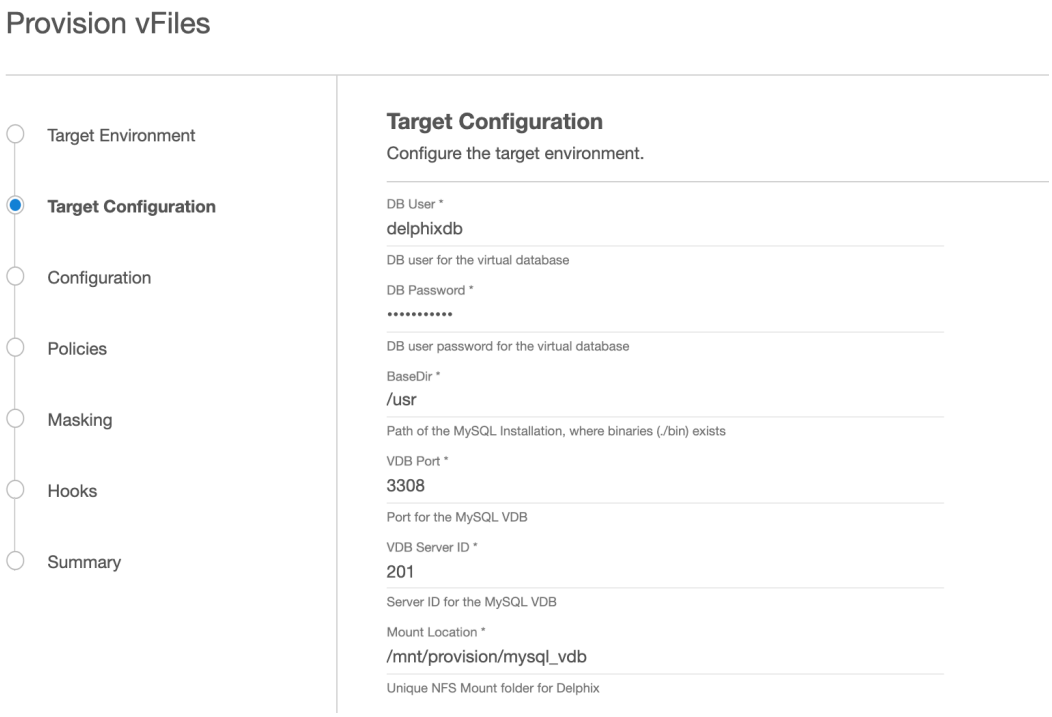


This same behavior will be observed in case of VDB refresh and rollback.

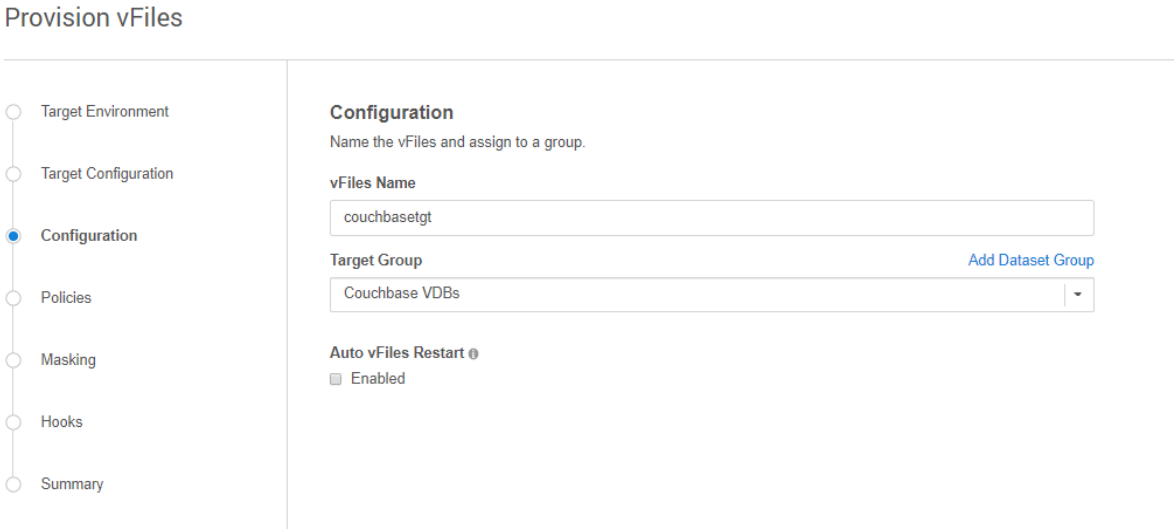
Note:

Do not create/rollback/refresh VDB on a higher MySQL version installed on the target environment from a dSource snapshot from a lower version of MySQL and vice versa. The existing dSource prior to release 4.1.0 will not have the above check.

4. Enter the following values for the target configuration:
    a. **DB User** This is the database user for the VDB. Delphix will use this user to manage the VDB. User needs to provide the same user account that was used by the staging database.
    b. **DB Password** Password for the VDB database user.
    c. **BaseDir** MySQL installation directory. This is the location of */b/mysql*
    d. **VDB Port** MySQL database port for the VDB. This port should not be used by any other application or MySQL server
    e. **VDB Server ID** MySQL server id for the VDB. This server-id must be non-zero.
    f. **Mount Location** This is the mount directory for Delphix VDB on the Target host. This location should be unique and empty.



5. On the **Configuration** screen, add the VDB name, select **Target Group**, check the **Auto vFiles Restart** checkbox, and click **Next**



6. On the **Policies** screen, add any required policies and click Next.
7. If you want to Mask the VDB, select the required configuration on this screen and click Next.
8. On the **Hooks** screen, add any required Hooks and click Next.
9. Preview the summary and click **Submit**.

Once the VDB is created successfully, you can review the datasets on **Manage** > **Datasets** screen in the lefthand panel. You have now provisioned a MySQL VDB.

Upgrading a dSource,VDB after a MySQL database upgrade

Overview

The following document describes the general methodology of upgrading your MySQL landscape. The “In-Place” upgrade method is the only recommended MySQL upgrade approach. This process replaces the old MySQL installation on the Target environment with the new version, using the same data directories as the previous version.

Perform the MySQL upgrade in the following order:

1. Target Environments and their VDBs
2. Staging (Replica) and Source Environments

System requirements

Review the following requirements before performing any environment upgrade steps.

- Plan to install compatible versions on the staging and target environments. To know the compatible versions, see MySQL connector source and target compatibility section in MySQL matrix.
- Install MySQL Select Connector v4.1.0 or greater.

Pre-requisites

You can follow the below prerequisites before performing any activity as per this documentation

- Review and familiarize with the MySQL upgrade guidelines and the upgrade prerequisites.
    - Delphix recommends validating these steps in a separate test environment before following the upgrade steps below.

⚠

- The MySQL upgrade from v5.7 to v8 introduced many incompatibility issues. We strongly recommend you install and run the MySQL Shell: Upgrade Checker Utility to identify those problems within all Source, Staging, Target, and VDB databases you plan to upgrade. Upgrading MySQL without fixing these issues can cause database initialization failure which might be unrecoverable and might need a rollback .
- Following a successful MySQL platform upgrade, you will need to take precautions against dSource and VDB corruption. Unlike other database types, MySQL staging and target environments are not backward compatible. We recommend you avoid the following scenarios:
    - Refresh or rollback a VDB to a snapshot with a lower version [ie 5.7] after an upgrade to a higher version [ie 8.0] with incompatible data as defined by the upgrade checker utility.
    - Enable a dSource or VDB when the latest snapshot of the dSource or VDB’s MySQL contains incompatible data as defined by the upgrade checker utility.

For the above scenario, follow troubleshooting guidelines in below section to analyze and correct the VDB or dSource

How to upgrade MySQL Target Environments and their VDBs

These directions outline how you can preserve the MySQL target environment and VDBs during an upgrade.

The target environment and VDBs must match the originating dSource to maintain full refresh and provision functionality. Therefore, it should be expected that users experience a temporary loss of the provision and refresh

features. During this time, VDBs can be enabled and running, however, no refreshes should be performed until the following section is followed and the dSource version matches.

> ⚠ Delphix do not support an automated way to upgrade VDBs. In general, Delphix recommends deleting and re-provisioning each VDB from a new snapshot. However, if you wish to preserve certain VDBs, follow the steps below. If you would rather not modify this target environment and its VDBs, you may also create a completely new target environment as well.

Directions:

1. Select the Target environment that you plan to upgrade and its set of VDBs.
2. Identify all VDBs that you wish to preserve and **delete** all VDBs that you do not.
3. Run MySQL Utility Checker on the remaining VDBs. This utility will identify incompatibility issues that can occur while upgrading from version 5.7 to 8.0.
4. For each VDB, solve the compatibility issues reported by MySQL Utility Checker.
5. Take a backup of the entire database.
6. **Remove** any VDB Refresh Policy assigned to the VDB.
7. **Stop** all VDBs.
8. By following the official MySQL in-place upgrade guidelines, perform an upgrade of MySQL installation on the Target environment.
9. **Refresh** the Target environment on the GUI.
10. **Start** all VDBs.
11. **Take a new snapshot** for each VDB.
    a. We recommend all prior snapshots are deleted as well to eliminate the chance of accidental rollback to a prior version.
12. Validate the upgraded VDBs.

At this point, the Target environment should be prepared to provision new VDBs from a matching dSource, and VDBs should be upgraded. Follow the next section to re-enable VDB refreshes.

How to upgrade MySQL Source and Staging Environments

Follow the steps below to upgrade dSources using the “In-Place” method. Each set of steps is organized based on the ingestion method.

> ⚠ If the issues reported by the MySQL Utility checker is not fixed for a dSource and upgrade is performed on the staging environment then the dSource will not start. To resolve this please check issue 2 in the troubleshooting section

Manual Ingestion

1. Run MySQL Utility Checker on all dSource on the staging instance and also source to find all the incompatibility issues that can occur while upgrading from version 5.7 to 8.0.
2. Solve all the compatibility issues reported by MySQL Utility Checker.
3. Take a backup of the entire database.
4. **Create** a new snapshot on an existing dSource.
5. **Disable** all the dSource on the staging instance and MySQL database on source instance
6. By following the official MySQL in-place upgrade guidelines, upgrade the MySQL staging instance.
7. Perform an upgrade of the source MySQL instance as per the official MySQL in-place upgrade guidelines.
8. **Refresh** the source and staging environments in the GUI.
9. **Enable** all the dSource on the staging instance and also start MySQL database on source instance
10. **Take a new snapshot** on an existing dSource.
11. **Validate** the dSource

Replication with Delphix initiated backup

1. Run MySQL Utility Checker on all dSources on the instance to find all the incompatibility issues that can occur while upgrading from version 5.7 to 8.0.
2. Solve all the compatibility issues reported by MySQL Utility Checker.
3. Take a backup of the entire database.
4. **Create** a new snapshot on an existing dSource.
5. **Disable** all the dSource on the staging instance and MySQL database on source instance
6. By following the official MySQL in-place upgrade guidelines, perform an upgrade of the replica instances. so here it is a staging instance.
7. By following the official MySQL in-place upgrade guidelines, perform an upgrade of the MySQL source instance.
8. **Refresh** the staging and source environments in the Delphix Engine UI.
9. **Enable** all the dSource on the staging instance and also start MySQL database on source instance
10. **Take a new snapshot** on an existing dSource.
11. **Validate** the dSource

Replication with customer-provided backup

1. Run MySQL Utility Checker on all dSources on the instance to find all the incompatibility issues that can occur while upgrading from version 5.7 to 8.0.
2. Solve all the compatibility issues reported by MySQL Utility Checker.
3. Take a backup of the entire database.
4. **Create** a new snapshot on an existing dSource.
5. By following the official MySQL in-place upgrade guidelines, perform an upgrade of the replica instances. So here it is the staging instance.
6. By following the official MySQL in-place upgrade guidelines, perform an upgrade of the MySQL Source instance.
7. **Take a new Backup** of the source database, and place it in the staging environment.
8. **Refresh** the staging and source environments in the GUI.
9. **Edit** the backup file location in the staging dSource configuration on Delphix Engine UI or **replace** the backup file on the staging environment.
10. **Enable** all the dSource on the staging instance and also start MySQL database on source instance
11. **Take a new snapshot** on an existing dSource.
12. **Validate** the dSource

Troubleshooting

- **Issue 1: VDB/dSource fails to start after enabling or creating a new snapshot**
  Solution: This occurs after an upgrade if you refresh or rollback the VDB to an older version of a snapshot or currently pointing to the latest snapshot which contains incompatible data. To resolve this, refresh or rollback to the correct version of the snapshot that was taken immediately after the upgrade.
  The logs in MySQL 8.0 will show that the database initialization failure occurred and because of this, the database was not able to start. An example of such an error is as below. Here there was a table named `` `catalogs` `` in MySQL schema of MySQL 5.7 database which is a default table in MySQL 8.0

```
023-07-26T15:02:39.477811Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld
(mysqld 8.0.34) starting as process 15891
2023-07-26T15:02:39.513413Z 1 [System] [MY-011012] [Server] Starting upgrade of
data directory.
2023-07-26T15:02:39.513453Z 1 [System] [MY-013576] [InnoDB] InnoDB
initialization has started.
```

```
2023-07-26T15:02:40.265033Z 1 [System] [MY-013577] [InnoDB] InnoDB
initialization has ended.
2023-07-26T15:02:40.270816Z 1 [ERROR] [MY-010781] [Server] Found ./mysql/
catalogs.frm file in mysql schema. DD will create .ibd file with same name.
Please rename table and start upgrade process again.
2023-07-26T15:02:40.270838Z 1 [ERROR] [MY-010336] [Server] Found .frm file with
same name as one of the Dictionary Tables.
2023-07-26T15:02:40.271004Z 0 [ERROR] [MY-010020] [Server] Data Dictionary
initialization failed.
2023-07-26T15:02:40.271028Z 0 [ERROR] [MY-013236] [Server] The designated data
directory /temp/vdbtarget/data/ is unusable. You can remove all files that the
server added to it.
2023-07-26T15:02:40.271039Z 0 [ERROR] [MY-010119] [Server] Aborting
2023-07-26T15:02:40.819837Z 0 [System] [MY-010910] [Server] /usr/sbin/mysqld:
Shutdown complete (mysqld 8.0.34)  MySQL Community Server - GPL.
2023-07-26T15:02:40.820669Z 0 [ERROR] [MY-010065] [Server] Failed to shutdown
components infrastructure.
```

- **Issue 2: dSource is not starting after an upgrade**
  Solution : When the dSource is enabled after an upgrade below error is shown on the UI of the engine.

```
Traceback (most recent call last): File "plugin/platform_server.py",
line 66, in _run_helper response = internal_func(request) File "./_linked.py",
line 517, in _internal_start_staging File "./plugin_runner.py",
line 75, in start_staging File "./pluginops/pluginops.py",
line 172, in start_staging File "./libs.py",
line 174, in run_bash File "./libs.py",
line 91, in _check_exit_code
dlpx.virtualization.libs.exceptions.PluginScriptError: The script failed
with exit code
```

Here the error depicts that dSource was not able to start . One of the reasons why the dSource won't start after an upgrade is it might have some incompatible data which does not work in MySQL 8.0. Commonly, the issues reported by MySQL Upgrade Checker were ignored and the upgrade was performed. Please follow the below steps if you encounter any error as above.

1. Uninstall MySQL version 8.0
   Note: Please make sure you don't delete your data directory
2. By following the official MySQL guidelines install MySQL version 5.7
3. Enable the dSource and Refresh the environment
4. Connect to the dSource and fix the incompatibilities
5. Take a new snapshot on the dSource
6. Disable the dSource on the GUI
7. Again upgrade the staging to MySQL version 8.0
8. Refresh the staging environment in the GUI.
9. Enable the dSource [dSource should be able to start since there is no incompatible data in the snapshot] on the GUI
10. Take a new snapshot [This snapshot will be of version 8.0]

MySQL references

This section covers the following topics:

- MySQL plugin exit codes
- Troubleshooting

MySQL plugin exit codes

Detailed below are the exit codes that the MySQL Plugin throws if an error occurs.

Code	Description	Possible Reason
1	General Error	Check logs
2	General Error	Check logs
3	Unable to start MySQL	1. Delphix user does not have the necessary permissions on the host 2. Mount location provided is in use 3. Database credentials provided are not accurate 4. Invalid server-id 5. Provided port is not in use
4	Missing config file	A `my.cnf` file was not provided and Delphix was unable to create one.
5	Unable to change `root` password	Delphix was unable to change the password for the root user after creating the staging db. Please check logs.
6	Unable to restore backup	Delphix was unable to restore the full backup into staging db. Please check logs.
7	Connect failure post restore	Delphix is unable to connect to staging database after backup is restored. 1. The Source DB username and password provided may be incorrect. 2. The MySQL Database did not restart after the restore. Check logs for more information on the error.
8	Delphix backup failure	Delphix could not take a source database backup. 1. Staging Host may not be able to connect to Source Host 2. Source DB credentials may be incorrect 3. Source DB user may not have the required permissions 4. Databases provided in the list may not be present on the source MySQL instance. Check logs for more information on the error.

Code	Description	Possible Reason
9	Customer backup failure	There was an issue with the customer provided backup. 1. Backup location may not exist. 2. The backup file may not exist 3. Backup file may be empty. Check logs for more information on the error.
10	Invalid binary path	`mysql` was not found under the provided installation directory.
11	Connect failure after restart	In Backup Ingestion Mode, the Delphix source user could not connect to the MySQL instance after reboot. Check logs for more information on the error.
12	Unable to create user	In Backup Ingestion Mode, there was an error while creating the delphix user after staging db was initialized. Check logs for more information on the error.

Troubleshooting

If you run into an error while using the MySQL plugin and have an exit code in the error, refer to ExitCodes for further information.

Logs

There are 2 sets of logs for the MySQL plugin

1. Plugin Logs Plugin logs are part of the Delphix Engine. Refer to [How to Retrieve Logs] (https://developer.delphix.com/References/Logging/#how-to-retrieve-logs) to find out how to get the Delphix Plugin Logs.
2. MySQL Shell Operation Logs When the plugin performs certain operations (such as Take backup, Link, Provision etc), logs are written to log files on the staging or target host under the Delphix Toolkit directory

Following are the logs to review

- *delphix_mysql_debug.log*
- *delphix_mysql_info.log*
- *delphix_mysql_error.log*

Common MySQL commands

- Connecting to a MySQL instance

```
delphixos> /usr/bin/mysql -uUserName -pPassword --protocol=TCP --port=1234
```

- List databases in MySQL

```
mysql> show databases;
```

- Check replication status in MySQL

```
mysql> show slave status;
```

- Check binary logging

```
mysql> SHOW VARIABLES LIKE 'log_bin';
```

- Check user permissions

```
mysql> show grants for 'delphixdb'@'%';
```

Oracle environments and data sources

This section covers the following topics:

Delphix architecture with Oracle

Overview

Various Oracle configurations ranging from Oracle RAC to Oracle multi-tenant can be used with Delphix. This article contains an overview of how Delphix works with Oracle.

There are three key concepts when using Delphix with any data platform:

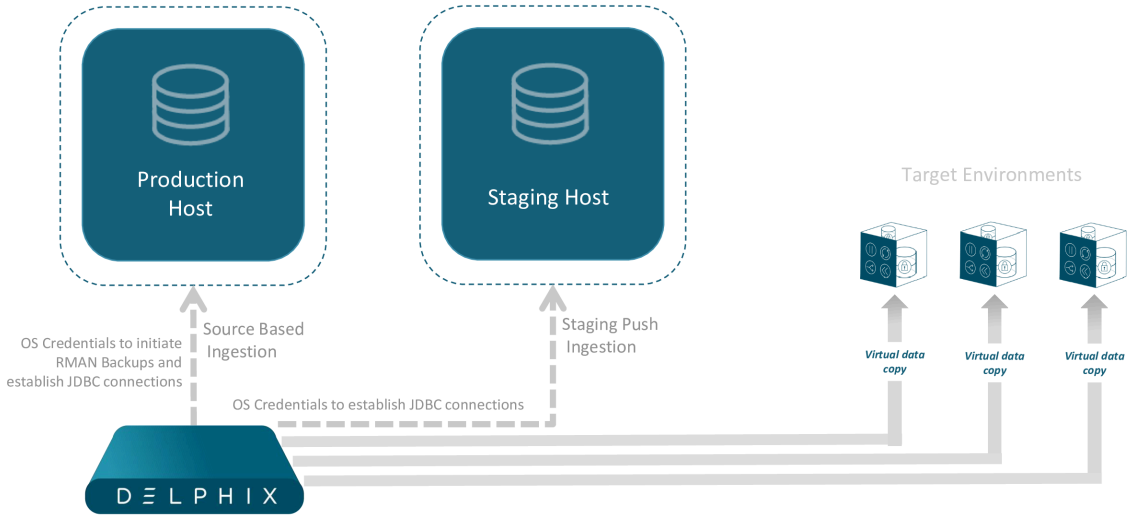
1. Environments: The server and software required to run a data set. For Oracle, this will be an operating system host with Oracle instances running on it.
    a. Source Environment: Source data to be ingested into Delphix. These will be used to create dSources.
    b. Target Environment: Target hosts to provision VDBs. These need Oracle installations that correspond to the versions of the Source environments, per our Oracle support matrix.
2. dSources: A database that the Delphix Virtualization Engine uses to create and update virtual copies of your database
3. VDBs: A database provisioned from either a dSource or another VDB which is a copy of the source data. A VDB is created and managed by the Delphix Virtualization Engine.

With these concepts in mind, explore how Delphix connects to Oracle environments and creates Oracle dSources and VDBs.

Delphix is not a replacement for an archived log backup solution and should not be relied on to do so. This can cause issues where Delphix is unable to start a vPDB that was plugged into a physical CDB if Delphix is having to determine and deliver the archived logs. You will need to find a separate archived log backup solution workflow to own and be responsible for physical CDB archived logs.

Environment linking and provisioning architecture

As shown in the diagram below, there are two ways by which Delphix Engine can ingest data from an Oracle dSource.



Source based ingestion

In this method, Delphix Engine begins by ingesting data from your source databases in order to create dSources. Once an environment is added, Delphix Engine will automatically 'discover' databases on it, which are compatible to ingest from, to create new dSources.

When a dSource is to be added, we will leverage RMAN and JDBC to create snapshots of the source by taking a full database backup. The newer snapshots take incremental backups of the database to ingest incremental data between snapshots. SnapSync and LogSync policies can be leveraged for automated snapshots. The result is a TimeFlow with various snapshots from which you can provision a VDB.

Staging push ingestion

Staging Push introduces the concept of a staging instance for Oracle dSource ingestion. Storage for the staging instance is provided by the Delphix Engine via NFS. The staging database can be populated with production data after which a dSource Snapshot can be taken.

The staging database needs to be active only while taking dSource snapshots. It can be shut down after the snapshot to save system resources on the Staging host. Pre-sync and Post-sync hooks can be used to populate the staging database with production data before taking snapshots.

When you later provision a VDB, you can specify any environment as a target, including the environment that contains the source/staging database. However, we recommend choosing a different target environment for the best performance. It must have an operating system that is compatible with the one running on the source/staging environment.

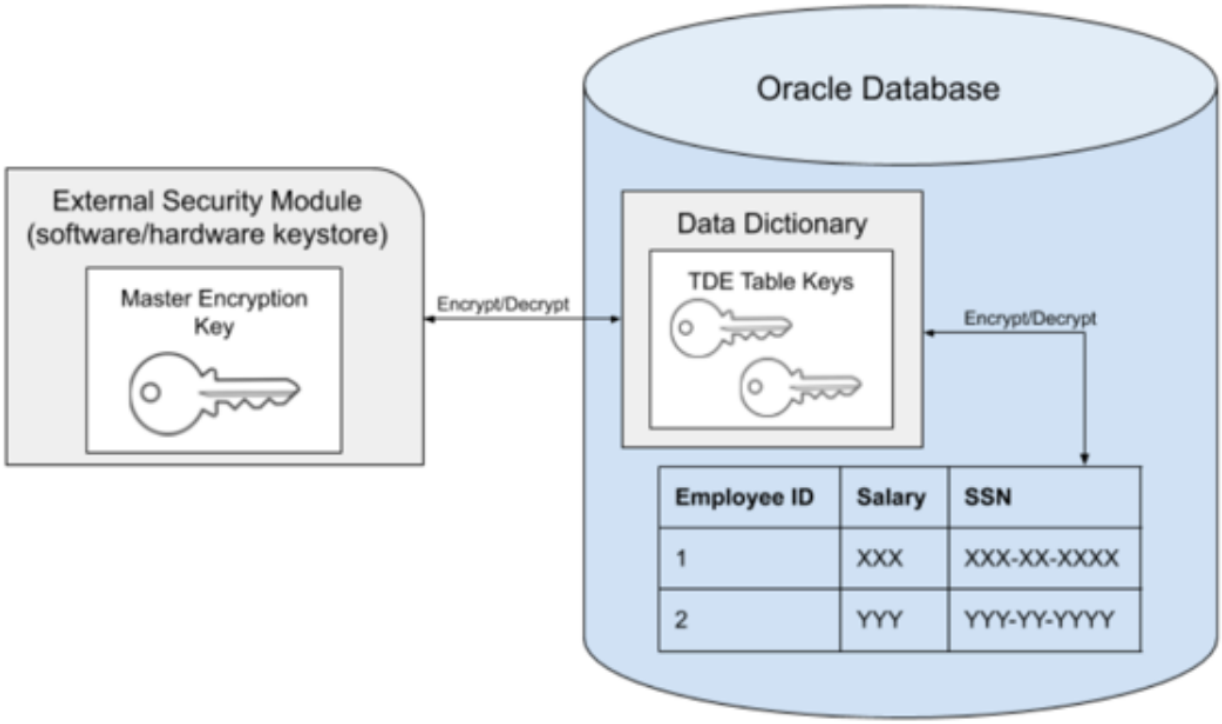
Transparent data encryption and Delphix

Overview

The Oracle Transparent Data Encryption feature encrypts the sensitive data (database tables and tablespaces) stored on the disk. This prevents misuse of the data if the disks or storage mediums are lost or stolen. The data is transparently decrypted for the authorized users when they access the data.

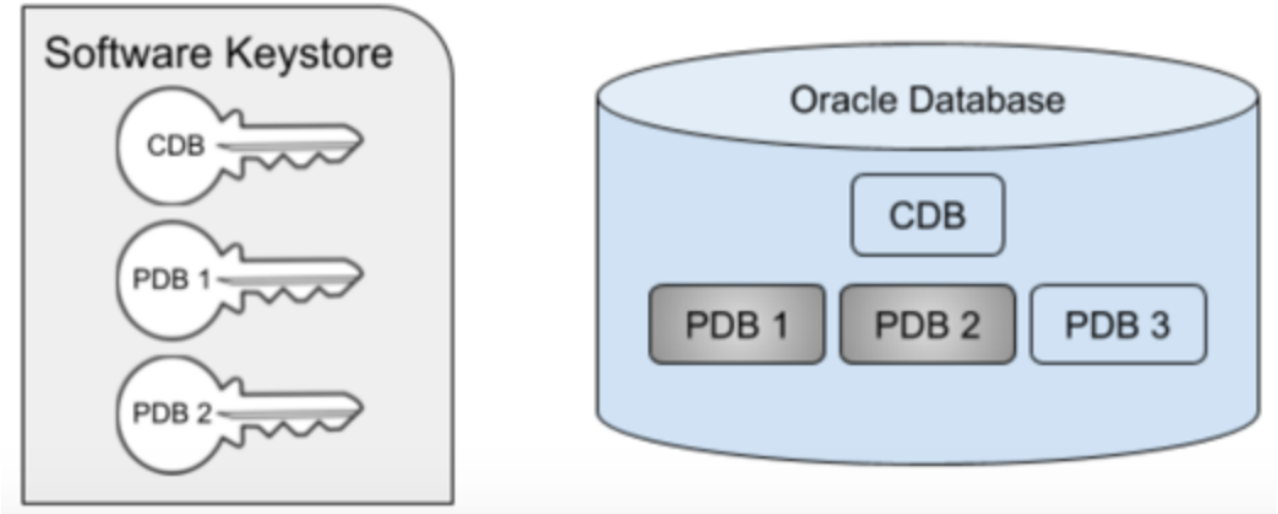
Data is encrypted with the help of encryption keys which are stored in an external module or file, known as the *wallet* or *keystore*. The keystore is managed by an authorized user and can be either a hardware or software keystore. In order to decrypt the data successfully, the encryption keys must be made available to the database by the authorized user.

In the Oracle database, the data is organized into tables that are located within a tablespace, which is in turn made up of one or more files on disk. With TDE either individual tables or an entire tablespace can be encrypted. The keys used to encrypt the data are stored within the database itself, in the data dictionary. The keys themselves are further encrypted using the wallet keys. By using 2 layers of encryption in this manner, access to all of the encrypted data can be achieved with just access to the keys in the keystone.

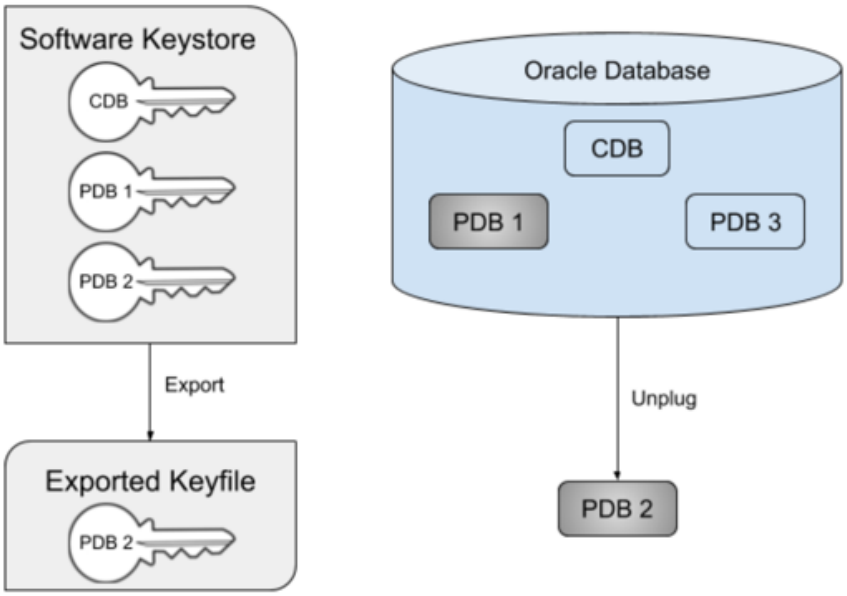


In the remainder of this document, the term *key* will refer to the master key stored in the external keystore, rather than the table keys stored in the data dictionary. In the case of a multi-tenant database, the same principle applies, namely that the keys are stored in the keystore which is located external to the database. However, not all of the PDBs within the container need to be encrypted. The CDB itself must always have a key associated with it, even if the CDB datafiles themselves are not encrypted. This is the case because Oracle will encrypt both the datafiles belonging to the PDB, as well as the archive logs, which are located in the CDB, as illustrated in the following diagram.

In the diagram below, PDBs 1 and 2 are encrypted, but PDB 3 is not. Note that the keystore contains all of the keys for the entire multitenant database, in this case, keys for the CDB, PDB 1, and PDB 2.



In order to unplug a TDE-enabled PDB, the key(s) for that PDB must be exported from the keystore into a new file, known as an *exported keyfile*. Oracle will prevent the PDB from being unplugged unless the keys have first been exported. The exported keyfile is encrypted with a password (known as the *keyfile secret*) that is specified during the export. The diagram below illustrates the scenario where PDB 2 has been unplugged. Note that the key for PDB 2 still remains in the keystore. This illustrates an important point about Oracle keystores - once added, a key cannot be easily removed from a keystore.



You can plug back a PDB into a new CDB before importing the new keys. This results in the PDB being plugged in in restricted mode with plugin violations. Importing the keys is necessary to resolve the plugin violations and start up the PDB in unrestricted READ WRITE mode.

The design of TDE allows for the keys for a given CDB or PDB to be changed, via the `ADMINISTER KEY MANAGEMENT SET KEY` command. This process is known as *key rotation* and will update the master encryption key in the keystore while leaving the table keys in the data dictionary. The table keys themselves will be encrypted with the new master encryption key, and any future updates to that data will use the new key. Existing data already written to the datafile or archive log will still be encrypted with the original key, however. For this reason, keys are

never removed from an existing wallet, only new keys are added. In order to decrypt all of the data in a given database, all current and prior keys will be needed.

For more information on TDE and how it is configured and enabled, see the Oracle documentation.

Delphix implementation of TDE

Oracle recommends that the keystore be stored on a separate disk from the datafiles. In accordance with this recommendation, neither keystores nor exported keyfiles are stored on Delphix storage. Rather, they are placed on customer storage. Exported keyfiles generated by Delphix are stored in the *artifact directory* under the *keystores root directory*, while keystores generated by Delphix are stored in the location specified by sqlnet.ora for the target container database. It is the customer's responsibility to maintain these storage locations and ensure they are backed up as needed, just like database files. If the keystore or exported keyfile is lost, the data in the associated vPDB may not be recoverable and the vPDB will cease to operate.

Delphix support for TDE has the following requirements:

- The Oracle version must be 12cR2 or higher (Oracle 12cR1 is not supported).
- Parent keystores of the source PDB must not be on ASM storage. Target CDB keystores, however, can be on ASM storage.
- Migration of a TDE-enabled vPDB from one CDB to another requires manual steps that must be completed for migrate vPDB to be successful and to support refresh and rewind operations on the migrated vPDB.
- Only software keystores on the same host as the database files are supported, in particular, Oracle Key Vault is not supported.
- The dSource from which the initial provision is done must be encrypted when it is linked. Existing dSources can not be encrypted without unlinking and creating a new dSource.
- There is currently no supported transition path from existing TDE-enabled vPDBs using the TDE workaround to the full product solution. The TDE workaround continues to be supported for approved customers.
- For linked provisions, the target container database should have an autologin Keystore configured. For a cluster target, the autologin keystore is shared. For vCDB provisions, Delphix will create an autologin Keystore when configuring the vCDB Keystore.

Some or all of these restrictions may be relaxed in future versions of Delphix.

Definitions

The following terms are used throughout TDE documentation and are summarized here for clarity. Note that the first occurrence of these terms may be on other documentation pages.

Term	Definition
Keystore/wallet	File found on the Oracle host which stores the keys used to encrypt and decrypt the internal table keys in a database. Every keystore has a password which is set when it is first created, and must be supplied for operations on it.
Parent keystore	Keystore with the keys used to encrypt the dSource PDB files.
Target keystore	Keystore for the target CDB into which the TDE-enabled vPDB is plugged.
Artifact directory	Directory on the target system (not on Delphix storage) which stores keys needed to support Delphix workflows on TDE-enabled vPDBs. It is located under the keystores root directory.

Term	Definition
Exported keyfile	File located on the target Oracle host which contains keys that have been exported from the keystore. It is encrypted with a secret that is specified when it is exported. The exported keyfile itself cannot be used as a keystore, but its contents can be imported into a new keystore.
Key rotation	Process for changing the master encryption key in the keystore via the `ADMINISTER KEY MANAGEMENT SET KEY` command. This does not remove the original key, rather it adds a new key to the wallet and future data will be encrypted with the new key.
Auxiliary container database (CDB)	Provisioning an Oracle vPDB requires running recovery to bring the snapshotted datafiles into a consistent state. This needs to be done in the context of a container database, which is created on the target system. After recovery is complete, the vPDB is unplugged and plugged into the target container, and the auxiliary container is deleted.
Keyfile secret	Password used to encrypt an exported keyfile.
Keystores root directory	User-specifiable location on the target system under which all TDE related artifacts such as keystores and exported keyfiles created by Delphix are stored. This includes both the artifact directories used for vPDBs and temporary directories used for auxiliary CDB keystores.

Wallet location configuration

Oracle requires that the keystore location be specified to the database so that it can be accessed when reading from or writing to the database files. This location can be specified in 2 ways:

1. The ENCRYPTION_WALLET_LOCATION parameter in `sqlnet.ora`.
2. The `wallet_root` initialization parameter is available starting in Oracle 18, while ENCRYPTION_WALLET_LOCATION is available in Oracle 12.2. Oracle has indicated that support for ENCRYPTION_WALLET_LOCATION will end starting with Oracle 20.

Delphix supports both configurations for the appropriate releases (i.e. ENCRYPTION_WALLET_LOCATION only in 12.2, and both ENCRYPTION_WALLET_LOCATION and `wallet_root` in 18+). When using ENCRYPTION_WALLET_LOCATION, Delphix recommends referencing an environment variable, for example:

Encryption wallet location

```
ENCRYPTION_WALLET_LOCATION=
 (SOURCE=
  (METHOD=FILE)
   (METHOD_DATA=
    (DIRECTORY=/u03/app/ora12201/admin/$ORACLE_UNQNAME/wallet/)))
```

As there is only one `sqlnet.ora` file found under $ORACLE_HOME, it will be used for all databases that use that home. Specifying an environment variable such as $ORACLE_UNQNAME allows a different location for each database in the same $ORACLE_HOME. Any environment variable referenced in `sqlnet.ora` must always be set in the environment for the Oracle user. Delphix explicitly sets only $ORACLE_HOME, $ORACLE_SID, and $ORACLE_UNQNAME in the connections which are established by the Delphix engine, so it is recommended that only these variables be referenced in `sqlnet.ora`. For a 12.2 TDE vPDB provision, Delphix creates a unique `sqlnet.ora` file for the use of the auxiliary database during the provision. For provisions to vCDB targets, Delphix will set the `wallet_root` parameter to a user-provided path for versions 18c or higher and will use the path in `sqlnet.ora` for version 12.2.

Required O/S permissions for the Delphix user

The provision itself is executed within the context of the environment user specified during the provision. This user does not have to be the Oracle user, and in fact, often is not. The Delphix user must be a member of the oracle group. During a TDE-enabled vPDB provision, the parent keystore is merged from a user-specified location to a location under the keystores root directory. The Delphix user does this copy via `ADMINISTER KEY MANAGEMENT` command. Since the Oracle user will do this, the Oracle user must be able to also create files in the wallet location.

The privilege requirements are satisfied by ensuring that the parent keystore has group read privileges, and the keystores root directory (owned by the Delphix user) has group write privileges.

Oracle environment management

In order for the Delphix Engine to ingest data or provision virtual databases, your Oracle environments must comply with Delphix environment configuration standards.

Quick start guide for Oracle on Linux

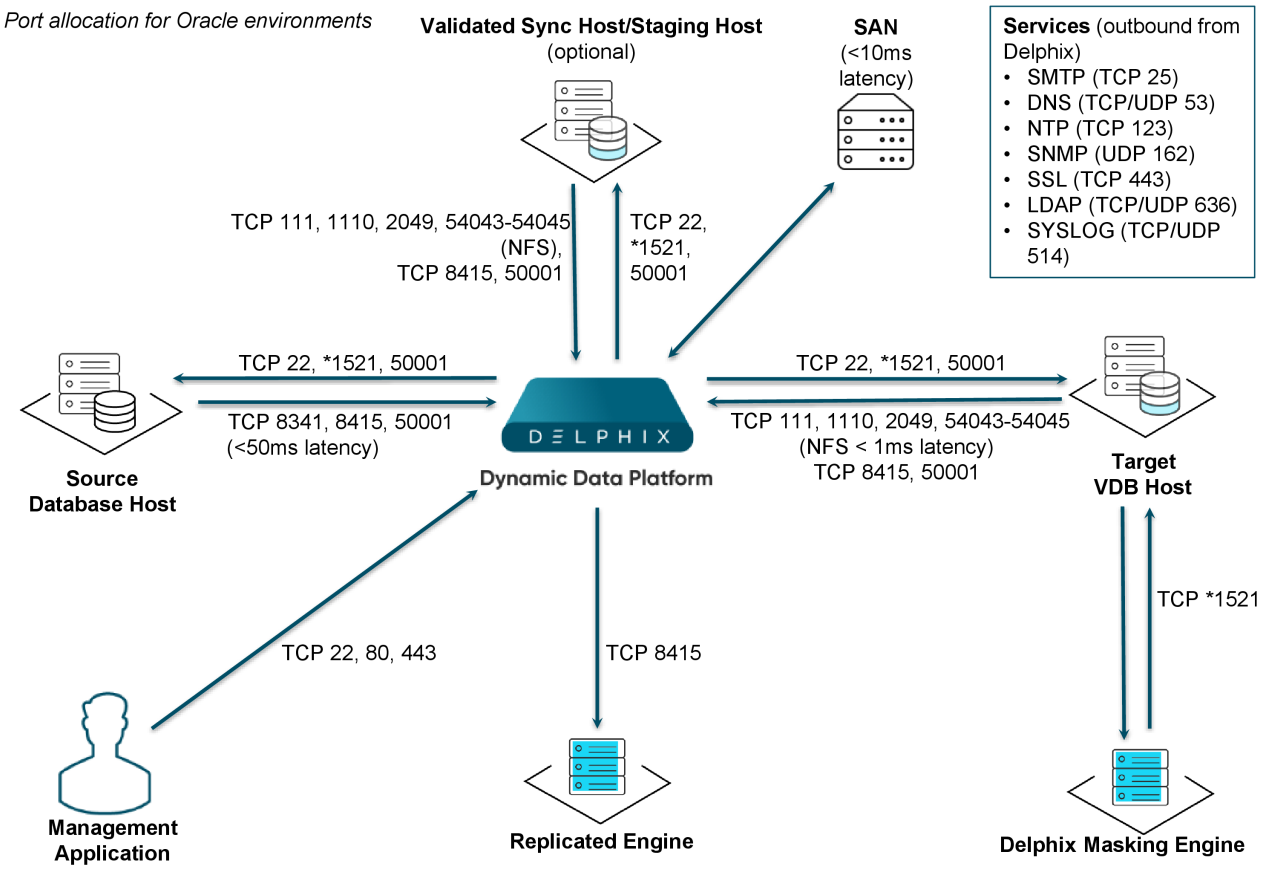
Overview

This quick start guide, which is excerpted from the larger User Guide, is intended to provide you with a quick overview of working with Oracle database objects in the Delphix Engine. It does not cover advanced configuration options including Oracle RAC, Linking to Standby, or best practices for performance. It assumes that you are working in a Lab/Dev setting and attempting to quickly test Delphix Engine functionality. It assumes you will use the VMware Hypervisor.

In this guide, we will walk through deploying a Delphix Engine, configuring Oracle Source and Target environments on Linux servers, creating a dSource, and provisioning a VDB.

The following diagram describes the engine topology for Oracle environments. It illustrates the recommended ports to be open from the engine to remote services, to the engine, and to the Source and Target Environments.

For this Quick Start Guide, you can ignore the following components: Validated Sync Host, Replicated Engine, and Delphix Masking Engine.



Note: *Oracle listener typically runs on TCP port 1521. In cases where other ports are used, substitute for 1521 above. Starting with 6.0.7.0 release, this port is required only if database user credentials are provided. The port need not be open if no database user credentials are provided.

Deploy OVA on VMware

Use the Delphix-supplied OVA file to install the Delphix Engine. The OVA file is configured with many of the minimum system requirements. The underlying storage for the install is assumed to be redundant SAN storage.

1. Download the OVA file from https://download.delphix.com. You will need a support login from your sales team or a welcome letter.
    a. Navigate to the Delphix Product Releases/ /appliance="" images="">
2. Login using the vSphere client to the vSphere server (or vCenter Server) where you want to install the Delphix Engine.
3. In the vSphere Client, click **File**.
4. Select **Deploy OVA Template**.
5. Browse to the OVA file.
6. Click **Next**.
7. Select a **hostname** for the Delphix Engine. This hostname will also be used in configuring the Delphix Engine network.
8. Select the **data center** where the Delphix Engine will be located.
9. Select the **cluster** and the **ESX host**.
10. Select one (1) **data store** for the **Delphix OS**. This datastore can be **thin-provisioned** and must have enough free space to accommodate the 127GB comprising the Delphix operating system.
11. Select four (4) or more **data stores** for Database Storage for the Delphix Engine. The Delphix Engine will stripe all of the Database Storage across these VMDKs, so for optimal I/O performance, each VMDK must be equal in size and be configured **Thick Provisioned - Eager Zeroed**. Additionally, these VMDKs should be distributed as evenly as possible across all four SCSI I/O controllers.
12. Select the **virtual network** you want to use. If using multiple physical NICs for link aggregation, you must use vSphere NIC teaming. Do not add multiple virtual NICs to the Delphix Engine itself. The Delphix Engine should use a single virtual network. For more information, see Optimal Network Architecture for the Delphix Engine
13. Click **Finish**. The installation will begin and the Delphix Engine will be created in the location you specified.
14. Once the installation has completed, power on the Delphix Engine and proceed with the initial system configuration as described in Setting Up Network Access to the Delphix Engine

> ⓘ If your source database is 4 TB, you probably need 4 TB of storage for the Delphix Engine. Add at least 4 data disks of similar size for the Delphix VM. For example: for a source database of 4 TB, create 4 VMDKs of 1 TB each.

> ⓘ For a full list of requirements and best practice recommendations, see Virtual Machine Requirements for VMware Platform

Setup network access to Delphix engine

1. Power on the Delphix Engine and open the Console.
2. Wait for the Delphix Management Service and Delphix Boot Service to come online. This might take up to 10 minutes during the first boot. Wait for the large orange box to turn green.
3. Press any key to access the sysadmin console.
4. Enter `sysadmin@SYSTEM` for the username and `sysadmin` for the password.
5. You will be presented with a description of available network settings and instructions for editing.

```
Delphix Engine Network Setup
```

```
To access the system setup through the browser, the system must first be
configured for networking in your
environment. From here, you can configure the primary interface, DNS, hostname,
and default route. When DHCP is
configured, all other properties are derived from DHCP settings.

To see the current settings, run "get." To change a property, run "set =." To
commit your changes,
run "commit." To exit this setup and return to the standard CLI, run "discard."

    defaultRoute    IP address of the gateway for the default route -- for
 example, "1.2.3.4."

    dhcp            Boolean value indicating whether DHCP should be used for
 the primary interface. Setting this value
                    to "true" will cause all other properties (address,
hostname, and DNS) to be derived from the DHCP
                    response

    dnsDomain       DNS Domain -- for example, "delphix.com"

    dnsServers      DNS server(s) as a list of IP addresses -- for example,
"1.2.3.4,5.6.7.8."

    hostname        Canonical system hostname, used in alert and other logs --
for example, "myserver"

    primaryAddress  Static address for the primary interface in CIDR notation
-- for example, "1.2.3.4/22"

Current settings:

    defaultRoute: 192.168.1.1
    dhcp: false
    dnsDomain: example.com
    dnsServers: 192.168.1.1
    hostname: Delphix
    primaryAddress: 192.168.1.100/24
```

6. Configure the `hostname`. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set hostname=<hostname>
```

warning: Use the same `hostname` you entered during the server installation.

7. Configure DNS. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set dnsDomain=<domain>
delphix network setup update *> set dnsServers=<server1-ip>[,<server2-ip>,...]
```

8. Configure either a static or DHCP address.
DHCP Configuration

```
delphix network setup update *> set dhcp=true
```

Static Configuration

```
delphix network setup update *> set dhcp=false
delphix network setup update *> set primaryAddress=<address>/<prefix-len>
```

warning:The static IP address must be specified in CIDR notation (for example, `192.168.1.2/24`)

9. Configure a default gateway. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set defaultRoute=<gateway-ip>
```

10. Commit your changes. Note that you can use the `get` command prior to committing to verify your desired configuration.

```
delphix network setup update *> commit
Successfully committed network settings. Further setup can be done through the
browser at:

        http://<address>

Type "exit" to disconnect, or any other commands to continue using the CLI.
```

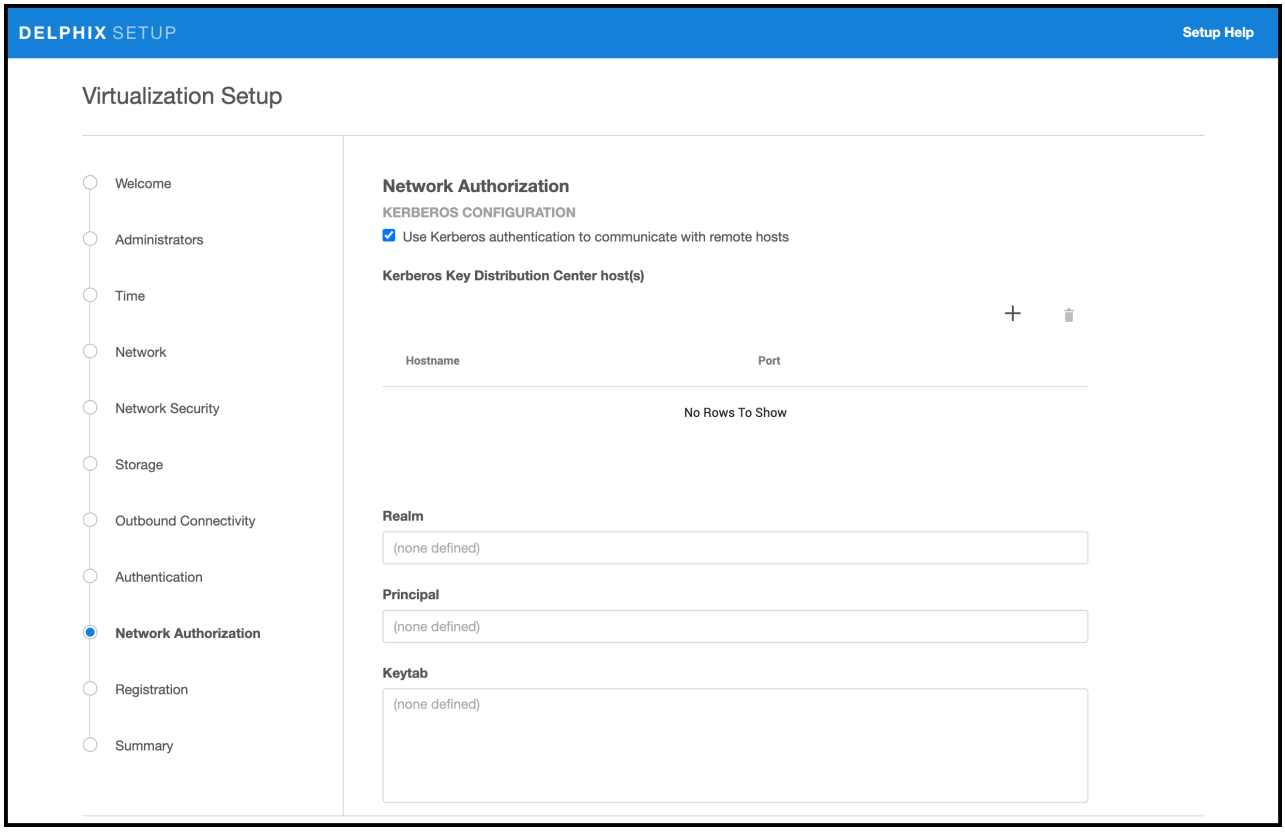
11. Check that you can now access the Delphix Engine through a Web browser by navigating to the displayed IP address, or hostname if using DNS.
12. Exit setup.

```
delphix> exit
```

Setting up the Delphix engine

Once you setup the network access for Delphix Engine, navigate to the Delphix Engine URL in your browser for server setup.

The welcome screen below will appear for you to begin your Delphix Engine setup.



The setup procedure uses a wizard process to take you through a set of configuration screens:

- Administrators
- Time
- Network
- Network Security
- Storage
- Outbound Connectivity
- Authentication
- Network Authorization
- Registration
- Summary

1. Connect to the Delphix Engine at `http:///login/index.html#serverSetup`.
   The **Delphix Setup** application will launch when you connect to the server. Enter your **sysadmin** login credentials, which initially defaults to the username **sysadmin**, with the initial default password of **sysadmin**. On first login, you will be prompted to change the initial default password.
2. Click **Next**.

Administrators

The Delphix Engine supports two types of administrators:

- System administrator (**sysadmin**) - this is the engine system administrator. The sysadmin password is defined here**.**
- Engine administrator (**admin)** - this is typically a DBA who will administer all the data managed by the engine.

On the Administrators tab, you set up the sysadmin password by entering an email address and password. The details for the admin are displayed for reference.

⚠ The default domain user created on Delphix Engines from 5.3.1 is known as **admin** instead of delphix_admin. When engines created before 5.3.1 are upgraded to 5.3.1 or later they will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then Disabling delphix_admin.

Time

The engine time is used as the baseline for setting policies that coordinate between virtual databases and external applications

Choose your option to setup system time in this section. For a Quick Start, simply set the time and your timezone. You can change this later.

Network

The initial out-of-the-box network configuration in the OVA file is set to use a single VMXNET3 network adapter.

You have already configured this in the initial configuration. Delphix supports more advanced configurations, but you can enable those later.

Storage

You should see the data storage VMDKs or RDMs you created during the OVA installation. Click **Next** to configure these for data storage.

Serviceability

Choose your options to configure serviceability settings.

For a Quick Start, accept the defaults. You can change this later.

Authentication service

Choose your options to configure authentication services.

For a Quick Start, accept the defaults. You can change this later.

Registration

If the Delphix Engine has access to the external Internet (either directly or through a web proxy), then you can auto-register the Delphix Engine:

1. Enter your **Support username** and **Support password**.
2. Click **Register**.

If external connectivity is not immediately available, you must perform manual registration.

1. Copy the **Delphix engine registration code** in one of two ways:
    a. Manually highlight the registration code and copy it to clipboard. Or,
    b. Click **Copy registration code to clipboard**.
2. Transfer the Delphix Engine's registration code to a workstation with access to the external network Internet. For example, you could e-mail the registration code to an externally accessible e-mail account.
3. On a machine with access to the external Internet, please use your browser to navigate to the Delphix Registration Portal at http://register.delphix.com
4. Login with your Delphix support credentials (username and password).
5. Paste the **Registration code**.
6. Click **Register**.

⚠ Although your Delphix Engine will work without registration, we strongly recommend that you register each engine as part of the setup. Failing to register the Delphix Engine will impact its supportability and security in future versions.

To regenerate the registration code for a Delphix Engine please refer to, Regenerating the Delphix Engine Registration Code. Delphix strongly recommends that you regenerate this code and re-register the engine regularly to maximize the Support Security of the Delphix Engine. Delphix recommends doing this every six months.

Summary

The final summary tab will enable you to review your configurations for System Time, Network, Storage, Serviceability, and Authentication.

1. Click the **Back** button to go back and to change the configuration for any of these server settings.
2. If you are ready to proceed, then click **Submit**.
3. Click **Yes** to confirm that you want to save the configuration.
4. Click **Setup** to acknowledge the successful configuration.
5. There will be a wait of several minutes as the Delphix Engine completes the configuration.

Requirements for Oracle hosts and databases

In order to begin using Oracle environments with Delphix, you will need to configure the source and target hosts with the requirements described on this page.

Oracle hosts and databases

On each host with Oracle, there must be an operating system user configured to the required specifications for Delphix, as explained in the table below. This user (i.e. delphix_os) can easily be created by using the createDelphixOSUser.sh script (located at the bottom of the page).

These requirements apply to both source and target environments. However, target environments have additional requirements which are detailed in the 'Target Host Requirements' section below.

Source host requirement

Host requirement	Explanation
Profile and privileges should be the same as the Oracle user (e.g. oracle) on the host.	For example, delphix_os should have the same environment variable settings ($PATH, $ORACLE_HOME, etc.), umask, and ulimit settings, as the user oracle.

Host requirement	Explanation
The Delphix software owner account (e.g. delphix_os) must have the same primary OS group as the Oracle software owner account (e.g. oracle).	Delphix recommends giving the delphix_os user the same primary OS group as the Oracle home owner. This ensures the Delphix engine can fully and automatically discover Oracle homes, databases, and listeners. Often, this is an OS group named oinstall. However, the oinstall group is not always necessary depending on your Oracle configuration. This user requires access to the libobk_proxy.so library in the toolkit for the child processes triggered by RMAN.
The delphix_os user must have the Oracle OSDBA group (typically dba) as a primary or secondary OS group.	The OSDBA group allows "OS authentication" when connecting to an Oracle database instance by specifying either username or password (i.e. rman target /), thus eliminating the need to store or retrieve a SYSDBA password. **Oracle 12c** For Oracle 12c and later versions of Oracle, the delphix_os user can also use OSBACKUPDBA as its secondary group. This is typically the BACKUPDBA group on the host.
For secondary group requirements, the Delphix OS user must be a member of the SYSBACKUP OS group (12.1 or higher) or the SYSDBA OS group (11.2 and lower).	This ensures that the Delphix engine is able to take snapshots of source databases using RMAN.
There must be a directory on the source host where the Delphix Engine Toolkit can be installed, for example: /var/opt/delphix/toolkit	The delphix_os user and primary OS group (i.e. oinstall) must own the directory. The directory must have at least permissions -rwxrwx--- (0770) The delphix_os user must have read and execute permissions on each directory in the path leading to the toolkit directory. At least 1.6 GB of storage is needed at the time of setting up the environment and at least 500MB of free space is required to allow refreshes and maintenance of the toolkit, especially during upgrades.
The Delphix Engine must be able to make an SSH connection to the source host (typically port 22).	

Host requirement	Explanation
Read access to $ORACLE_HOME and all underlying files and directories.	Delphix needs to run locally available Oracle tools such as sqlplus and RMAN. Those executables, as well as various required libraries, reside inside the Oracle home

Additional requirements for RAC environments

If the source host is part of a RAC cluster, Delphix will attempt to use all nodes and crsctl for its operations.

RAC environment requirement	Explanation
The delphix_os user and Delphix Toolkit configuration must be the same on each node in the RAC cluster	• delphix_os must have execute permission on crsctl and srvctl on each node in the cluster. • access to olsnodes is needed. • The Delphix Toolkit must be installed in the same directory path on each of the nodes in the source cluster. • All data files, archive logs, and database control file must be located on storage shared by all of the cluster nodes. • Each node in the cluster must be able to access archive logs and the database control file from all other nodes. • At least 1.6 GB of storage is needed at the time of setting up the environment and at least 500MB of free space is required to allow refreshes and maintenance of the toolkit, especially during upgrades.

Masked Provisioning is not supported on Oracle RAC.

Auto-discovery requirements

The preferred way to find source databases it to allow Delphix to automatically discover your Oracle Homes and Databases by examining the inventory and oratab files and the listener setup. Successful autodiscovery requires read access to these and related files.

In **most** environments, delphix_os group membership is sufficient to perform auto-discovery.

However, if you have overridden Oracle's group permission structure, you may need to modify privileges to allow auto-discovery.

Auto-discovery requirements	Explanation
The ORATAB file must exist (typically in /etc/oratab or /var/opt/oracle/oratab) and be readable by delphix_os. Read access to: • /etc/orainst.loc or /var/opt/oracle/orainst.loc. • the Oracle inventory file (inventory.xml)	The ORATAB is used to determine the location of the Oracle installations on your host. Note: The Oracle inventory file is identified by the contents of orainst.loc (for example, $INVENTORY_HOME/ContentsXML/inventory.xml).
Permission to run pargs on Solaris hosts and ps on AIX, HP-UX, and Linux hosts, as super-user. This permission is usually granted via sudo authorization of the commands.	Unless you have used a custom TNS_ADMIN setting, elevated access to ps (pargs on Solaris) is not required See the topic Sudo Privilege Requirements for Oracle Environments for further explanation of this requirement, and Sudo File Configuration Examples for Oracle Environments for examples of file configurations

Source database requirements

For each source database, there are specific configurations required for Delphix to ingest data.

Source database requirement	Explanation
For each Oracle Home, the delphix_os user should have execute permission for the programs in $ORACLE_HOME/bin.	If symlinks are configured, Delphix must be configured with the same $ORACLE_HOME path as was used when starting the instance. Ensure the PermitUserEnvironment configuration parameter = "yes" in the sshd_config file The $ORACLE_HOME/bin/oracle executable must have the SETUID and SETGID flags set. Permissions on the oracle binary must be at least -rwsr-s–x (06751).
Source databases must be in ARCHIVELOG mode to ensure that redo logs are archived.	Archive logs are required to make SnapSyncs consistent and provisionable.
Enable Block Change Tracking (BCT). (Highly Recommended).	Enabling BCT will improve SnapSync operation time. Without BCT, incremental SnapSyncs must scan the entire database.

Source database requirement	Explanation
Enable FORCE LOGGING. (Highly Recommended)	If you do not enable FORCE LOGGING and NOLOGGING operations take place, you will get a Fault from the Delphix Engine. If you must use NOLOGGING to meet specific performance criteria, take a new snapshot of the source database after doing the NOLOGGING operations to bring the dSource up-to-date before provisioning VDBs.

Operating system specific requirements

Solaris

On a Solaris host, gtar must be installed. Delphix uses gtar to handle long file names when extracting the toolkit files into the toolkit directory on a Solaris host. The gtar binary should be installed in one of the following directories:

- /bin:/usr
- /bin:/sbin:/usr
- /sbin:/usr/contrib
- /bin:/usr/sfw
- /bin:/opt/sfw
- /bin:/opt/csw/bin

Additional target /staging host requirements

This topic describes the user privileges and environment requirements that are required for Oracle target hosts and databases collectively referred to as target environments.

These are in addition to the 'Oracle Hosts and Database Requirements' called out above.

Target/Staging host requirement	Explanation
The Delphix OS user must be a member of the SYSDBA OS group.	This ensures that the Delphix engine is able to create new VDBs on target hosts.
There must be a directory (e.g. "/mnt/provision/" or"/mnt/staging/") that will be used as a container for the NFS mount points that are created when provisioning a VDB.	The delphix_os user and primary OS group (i.e. oinstall or dba) must own the directory. The directory must have at least permissions -rwxrwx--- (0770). There must be no symbolic links in the path of this directory, because NFS can mount into a directory with symlinks in its path, but cannot unmount.

Target/Staging host requirement	Explanation
Permission to run: • mount, umount, mkdir, rmdir as super-user. • pargs on Solaris hosts • ps on Linux, AIX, and HP-UX as super-user. • nfso on AIX hosts as super-user	The following permissions are usually granted via sudo authorization of the commands. See Sudo Privilege Requirements for Oracle Environments for further explanation of the commands, and Sudo File Configuration Examples for Oracle Environments for examples of the /etc/sudoers file on different operating systems.
(Optional) Write permission to the $ORACLE_BASE_CONFIG/dbs directory.	(i.e. chmod g+w $ORACLE_BASE_CONFIG/dbs)
An Oracle listener process running on the target host for provisioning a VDB	The listener's version should be equal to or greater than the highest Oracle version that will be used to provision a VDB.
Required packages on target hosts: 1. portmapper / rpcbind 2. status daemon (rpc.statd) 3. NFS lock manager (rpc.lockd/lockmgr)	As the Delphix Engine uses NFSv3 for mounting target host filesystems, the prerequisite packages to support NFSv3 client communication are required for normal operation, and the required services to support NFS client communications (including file locking) must be running. NFSv3 is enabled by default and to enable NFSv4, see NFSv4 Configuration.
If using ASM, the delphix_os user must have the Oracle ASM ownership groups as secondary OS groups (typically asmadmin and asmdba)	This ensures the Delphix engine can fully and automatically discover Oracle homes, databases, and listeners.

Deploy Hostchecker to Validate Delphix Requirements

Delphix has developed a hostchecker script that contains standardized checks for source and target hosts - these checks generally fall into three buckets

- OS and Host permissions/access
- Network Port Checks
- DB-specific functionality

OS and Host permissions/access and network port checks can (and should) be performed prior to Delphix installation to ensure a smooth deployment.

Each DB should have a specifically associated hostchecker - you can find detailed documentation on the DB-specific hostchecker page.

1. Download the appropriate **HostChecker tarball** for your engine from https://download.delphix.com/ Tarballs follow the naming convention "hostchecker_<OS>_<processor>.tar". For example, if you are validating a linux x86 host you should download the hostchecker_linux_x86.tar tarball.
2. Create a working directory and extract the **HostChecker files** from the **HostChecker tarball**.

```
mkdir dlpx-host-checker
cd dlpx-host-checker/
```

```
tar -xf hostchecker_linux_x86.tar
```

3. Run the `sh` script contained within:

   ```
   sh hostchecker.sh
   ```

   This will extract the JDK included in the tarball (if necessary) and invoke the HostChecker.

   ```
   ora10205@bbdhcp:/home/ora10205/hostchecker-> sh hostchecker.sh
   Extracting the JDK from the tarball jdk-6u45-linux-i586.tar.gz.
   ```

   warning: **Don't Run as Root** Do not run the HostChecker as root; this will cause misleading or incorrect results from many of the checks.
4. Select which **checks** you want to run.
   Info: **Run Tests without the Interface**
   You can also run checks without spawning the interface. Enter `--help` to get a list of arguments you can pass to the HostChecker.
5. As the checks are made, enter the requested **arguments**.
6. Read the output of the check. The general format is that severity increases as you scroll down the output. First comes informational output, then warnings, then errors. **warning** : Internal Errors If you see a message that starts with `Internal Error`, forward it to Delphix Support immediately. This represents a potential bug in the HostChecker, and not necessarily a problem with your environment.
7. Error or warning messages will explain any possible problems and how to address them. Resolve the issues that the HostChecker describes. Do not be surprised or undo your work if more errors appear the next time you run HostChecker, because the error you just fixed may have been masking other problems.
8. Repeat steps 3 - 7 until all the checks return no errors or warnings.

Adding Oracle source and target environments

Follow the steps below to add **both** source and target environments for Oracle.

1. Login to the **Delphix management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Click the **Plus** icon next to **Environments**.
5. In the **Add environment** dialog, select **Unix/Linux**.
6. Select **Standalone host** or **Oracle cluster**, depending on the type of environment you are adding.
7. For standalone Oracle environments enter the **Host IP** address.
8. For Oracle RAC environments, enter the **Node address** and **Cluster home.**
9. Enter an optional **Name** for the environment.
10. Enter the **SSH** port. The default value is **22**.
11. Enter a **Username** for the environment. See Requirements for Oracle Target Hosts and Databases for more information on the required privileges for the environment user.
12. Select a Login Type: — Username and Password - enter the OS username and password, or — Username and Public Key - enter the OS username. — Password Vault - select from an existing Enterprise Password Vault
    Info : Using Public Key Authentication
    If you want to use public-key encryption for logging into your Unix-based environment:
    a. Select **Public key** for the **Login type.**
    b. Click **View public key**.

c. Copy the public key that is displayed, and append it to the end of your `~/.ssh/authorized_keys` file. If this file does not exist, you will need to create it.
    i. Run chmod 600 ~/.ssh/authorized_keys to allow only the file's owner to read and write to it (make sure the file is owned by the user).
    ii. Run chmod 755 ~ to restrict access to the user's home directory so no other user may write to it.
    iii. Run chmod 700 ~/.ssh so that others cannot write to it. The ~/.ssh directory cannot be writable by group or other users. Otherwise, authentication will fail.

The public key needs to be added only once per user and per environment.

13. For **Password login**, click **Verify credentials** to test the username and password.
14. Enter a **Toolkit path**. The toolkit directory stores scripts used for Delphix Engine operations, and should have a persistent working directory rather than a temporary one. The toolkit directory will have a separate subdirectory for each database instance. The toolkit path must have 0770 permissions and at least 345MB of free space.
15. Click **Submit**.

Linking an Oracle data source

1. Login to the **Delphix management** application.
2. Navigate to **Manage > Datasets.**
3. Click the plus icon and select **Add dSource**.
4. In the **Add dSource** wizard, select the source database with the correct environment user-specified.
   Info: For linking an Oracle staging push dSource, follow the procedure in Staging Push Implementation for Oracle
5. Enter your login credentials for the source database and click **Verify credentials**. If you are linking a mounted standby, Click **Next**. See the topics under Linking Oracle Physical Standby Databases for more information about how the Delphix Engine uses non-SYS login credentials.
6. Enter a name and select a group for your dSource. Adding a dSource to a dataset group lets you set Delphix Domain user permissions for that database and its objects, such as snapshots. See the topics under Users and Groups for more information.
7. Select the **Data Management** settings needed. For more information, visit Data Management Settings for Oracle Data Sources
8. Assign existing policies to the new dSource. New policies can be created and associated later.
9. Enter any scripts that should be run using the **Hooks** page.
10. Review the dSource Configuration and Data Management information, and then click **Submit**.

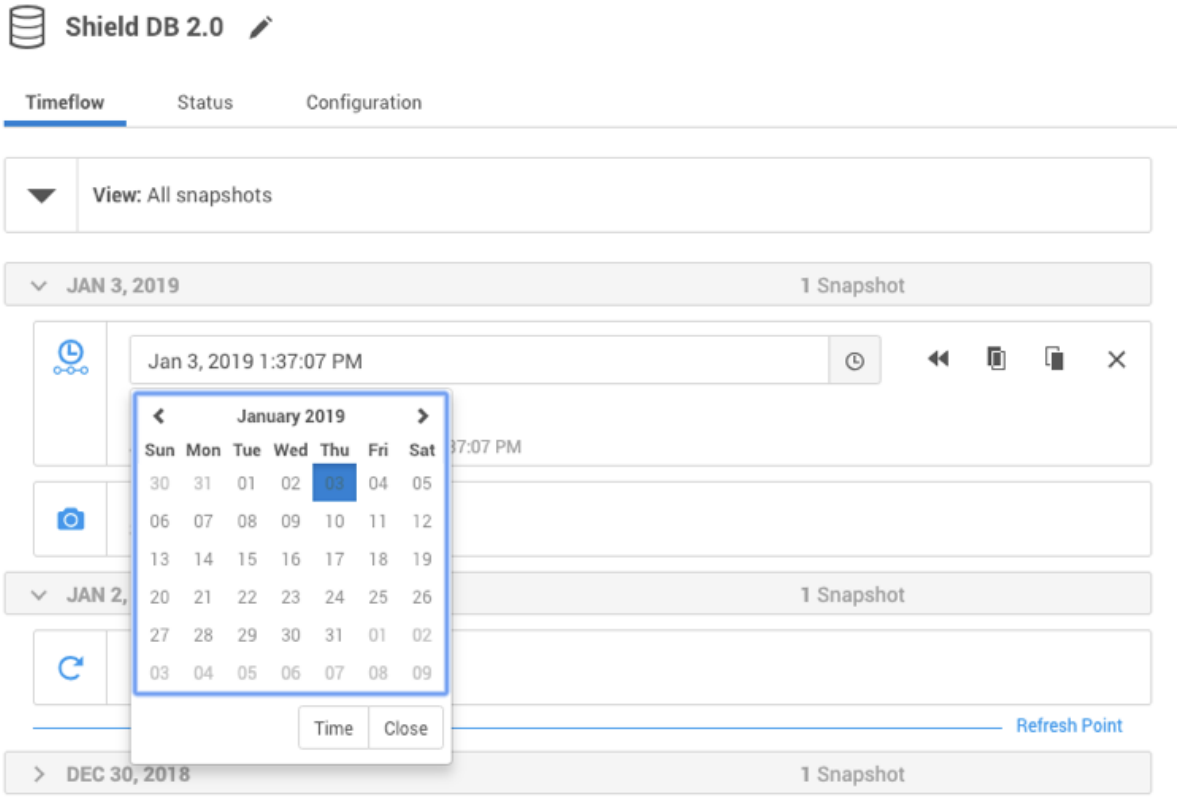
Once the action to add a dSource has been submitted, the Delphix Engine will initiate a DB_Link job to create the dSource. If the *Load Immediately* option was selected in the data management page a DB_Sync job will also be executed to ingest data from the source, otherwise, this first DB_Sync job will run as per the associated SnapSync policy.

When the jobs have successfully completed, the database icon will change to a dSource icon on the Environments > Databases screen, and the dSource will be added to the list of Datasets under its assigned group.

Provisioning an Oracle VDB

1. Login to the **Delphix management** application.
2. Select **Manage > Datasets**.
3. In the **Datasets** panel on the left-hand side, click the **group** containing the dSource or VDB from which you want to provision.
4. Click the **TimeFlow** tab.
5. Select a **snapshot**.
   You can take a snapshot of the dSource from which to provision. To do so, click the **Camera** icon.

6. **Optional:** Select to open LogSync timeline.



7. Select to provision from a point of time within a snapshot. You can select by date or time.
8. Click and the **Provision VDB**wizard will open:
    a. For **Oracle single instance** the fields **Installation home**, **Database unique name**, **SID**, **Database name**, **Mount base**, and **Environment user** will auto-populate with information from the parent.
    b. For **Oracle RAC** the fields **Installation home**, **Database unique name**, **SID**, **Database name**, **Mount base**, **Instance number, Instance name** and **Environment user** will auto-populate with information from the parent.
    Editable Fields in the VDB Provision Wizard
    The following fields are editable:
    Installation Home (need to have an additional compatible target)
    Database Unique Name
    SID
    Database Name
    Mount Base
    Instance Number (RAC Only)
    Instance Name (RAC Only)
9. If you need to add a new target environment for the VDB, click the green **Plus** icon next to the **Filter target** field, and follow the instructions in Oracle Single Instance or RAC Environment
10. Review the information for **Installation home**, **Database unique name**, **SID**, and **Database name**. Edit as necessary.
11. Review the **Mount base** and **Environment user**. Edit as necessary. The Environment User must have permission to write to the specified Mount Base, as described in Requirements for Oracle Target Hosts and

Databases. You may also want to create a new writeable directory in the target environment with the correct permissions and use that as the Mount Base for the VDB.

12. Select **Provide privileged credentials** if you want to use login credentials on the target environment that are different from those associated with the **Environment user**.
13. Click **Advanced** to customize the VDB online log size and log groups, archivelog mode, local_listener parameter (TCP/IPC protocol addresses), additional VDB configuration settings or file mappings, or custom environment variables. For more information, see Customizing Oracle VDB Configuration Settings, Customizing VDB File Mappings, and Customizing Oracle VDB Environment Variables
    If you are provisioning to a target environment that is running a Linux OS, you will need to compare the `SGA_TARGET` configuration parameter with the shared memory size in `/dev/shm`. The shared memory configured on the target host should match the SGA memory target. You can check the Linux OS shared memory size with the command df -k /dev/shm and the SGA_TARGET configuration parameter by opening the **Advanced** settings and then finding the value for `SGA_TARGET` under **VDB Configuration templates**.
14. Click **Next**.
15. Select a **Target Group** for the VDB.
16. Enable Auto VDB Restart to allow the VDB to be automatically restarted when the target host reboot is detected by Delphix.
17. Click Next.
18. Select a **Snapshot policy** for the VDB.
19. Click **Next**.
20. Enter any operations that should be run at Hooks during the provisioning process. Click **Next**.
21. Click **Submit**.

When provisioning starts, you can review the progress of the job by selecting the VDB and clicking on the **Status** tab, or by selecting **Manage/Dashboards** and viewing the **Job history** panel. Alternatively, you could see this in the **Actions sidebar.** When provisioning is complete, the VDB will be included in the group you designated and listed in the **Datasets** panel. If you select the VDB in the **Datasets** panel and click the **Configuration** tab, you can view information about the database and its Data Management settings.

Next steps

Congratulations! You have provisioned your first virtual database!

Now, perform some simple functional tests with your application. You can connect your app to the VDB using standard TNS/JDBC techniques. Delphix has already registered the VDB for you on the target listener.

We suggest the following next steps:

1. Drop a table and use the VDB Rewind feature to test the recovery of your VDB.
2. Take a snapshot of your dSource and refresh your VDB to quickly get fresh production data.
3. Provision a new VDB from your VDB to test the sharing data quickly with other teams.
4. Mask your new VDB to protect sensitive data. Provision new VDBs from that masked VDB to quickly provide safe data to development and QA teams.

Script

- createDelphixOSUser.sh

Quick start guide for Oracle on Solaris SPARC

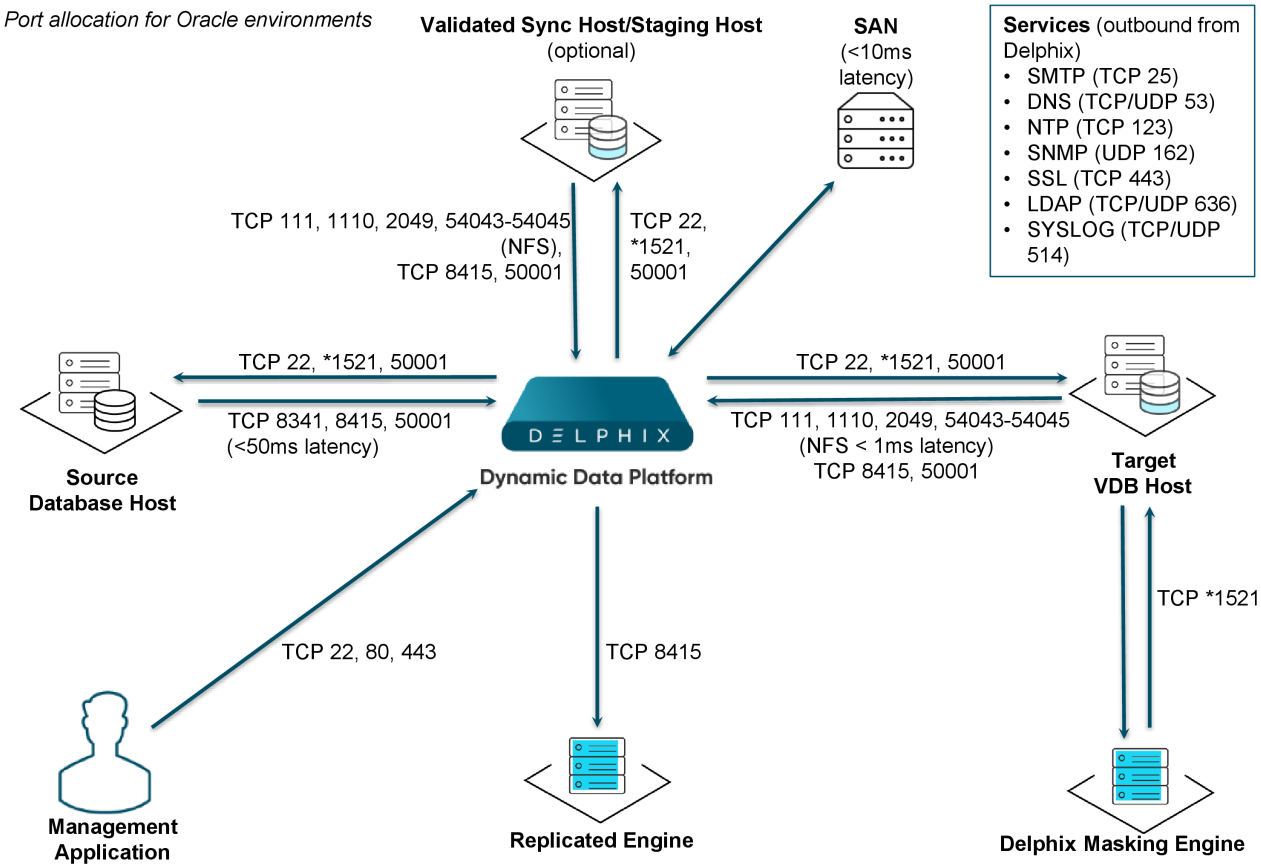
This quick start guide, which is excerpted from the larger User Guide, is intended to provide you with a quick overview of working with Oracle database objects in the Delphix Engine. It does not cover advanced configuration options including Oracle RAC, Linking to Standby, or Best Practices for Performance. It assumes that you are working in a Lab/Dev setting and attempting to quickly test Delphix functionality. It assumes you will use the VMware Hypervisor.

Overview

In this guide, we will walk through deploying a Delphix Engine, configuring Oracle Source and Target environments on Solaris SPARC servers, creating a dSource, and provisioning a VDB.

The following diagram describes the Delphix topology for Oracle environments. It illustrates the recommended ports to be open from Delphix to remote services, to the Delphix Engine, and to the Source and Target Environments.

For this Quick Start Guide, you can ignore the following components: Validated Sync Host, Replicated Engine, and Delphix Masking Engine.



Note: *Oracle listener typically runs on TCP port 1521. In cases where other ports are used, substitute for 1521 above. Starting with 6.0.7.0 release, this port is required only if database user credentials are provided. The port need not be open if no database user credentials are provided.

Deploy OVA on VMware

Use the Delphix-supplied OVA file to install the Delphix Engine. The OVA file is configured with many of the minimum system requirements. The underlying storage for the install is assumed to be redundant SAN storage.

1. Download the OVA file from https://download.delphix.com. You will need a support login from your sales team or a welcome letter.
    a. Navigate to the Delphix Product Releases/<Current Version>/Appliance Images page.
2. Login using the vSphere client to the vSphere server (or vCenter Server) where you want to install the Delphix Engine.
3. In the vSphere Client, click **File**.
4. Select **Deploy OVA template**.
5. Browse to the OVA file.
6. Click **Next**.
7. Select a **hostname** for the Delphix Engine. This hostname will also be used in configuring the Delphix Engine network.
8. Select the **data center** where the Delphix Engine will be located.
9. Select the **cluster** and the **ESX host**.
10. Select one (1) **data store** for the **Delphix OS**. This datastore can be **thin-provisioned** and must have enough free space to accommodate the 127GB comprising the Delphix operating system.
11. Select four (4) or more **data stores** for Database Storage for the Delphix Engine. The Delphix Engine will stripe all of the Database Storage across these VMDKs, so for optimal I/O performance, each VMDK must be equal in size and be configured **Thick Provisioned - eager zeroed**. Additionally, these VMDKs should be distributed as evenly as possible across all four SCSI I/O controllers.
12. Select the **virtual network** you want to use. If using multiple physical NICs for link aggregation, you must use vSphere NIC teaming. Do not add multiple virtual NICs to the Delphix Engine itself. The Delphix Engine should use a single virtual network. For more information, see Optimal Network Architecture for the Delphix Engine
13. Click **Finish**. The installation will begin and the Delphix Engine will be created in the location you specified.
14. Once the installation has completed, power on the Delphix Engine and proceed with the initial system configuration as described in Setting Up Network Access to the Delphix Engine

ⓘ If your source database is 4 TB, you probably need 4 TB of storage for the Delphix Engine. Add at least 4 similarly sized data disks for Delphix VM e.g. for 4 TB create 4 VMDKs of 1 TB each.

ⓘ For a full list of requirements and best practice recommendations, see Virtual Machine Requirements for VMware Platform

Setup network access to Delphix engine

1. Power on the Delphix Engine and open the Console.
2. Wait for the Delphix Management Service and Delphix Boot Service to come online. This might take up to 10 minutes during the first boot. Wait for the large orange box to turn green.
3. Press any key to access the sysadmin console.
4. Enter `sysadmin@SYSTEM` for the username and `sysadmin` for the password.
5. You will be presented with a description of available network settings and instructions for editing.

```
Delphix Engine Network Setup
To access the system setup through the browser, the system must first be
configured for networking in your
```

```
environment. From here, you can configure the primary interface, DNS, hostname,
and default route. When DHCP is
configured, all other properties are derived from DHCP settings.

To see the current settings, run "get." To change a property, run "set
<property>=<value>." To commit your changes,
run "commit." To exit this setup and return to the standard CLI, run "discard."

    defaultRoute    IP address of the gateway for the default route -- for
 example, "1.2.3.4."

    dhcp            Boolean value indicating whether DHCP should be used for
 the primary interface. Setting this value
                    to "true" will cause all other properties (address,
hostname, and DNS) to be derived from the DHCP
                    response
    dnsDomain       DNS Domain -- for example, "delphix.com"
    dnsServers      DNS server(s) as a list of IP addresses -- for example,
"1.2.3.4,5.6.7.8."
    hostname        Canonical system hostname, used in alert and other logs --
for example, "myserver"
    primaryAddress  Static address for the primary interface in CIDR notation
-- for example, "1.2.3.4/22"
Current settings:
    defaultRoute: 192.168.1.1
    dhcp: false
    dnsDomain: example.com
    dnsServers: 192.168.1.1
    hostname: Delphix
    primaryAddress: 192.168.1.100/24
```

6. Configure the `hostname`. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set hostname=<hostname>
```

Note: Use the same `hostname` you entered during the server installation

7. Configure DNS. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set dnsDomain=<domain>
delphix network setup update *> set dnsServers=<server1-ip>[,<server2-ip>,...]
```

8. Configure either a static or DHCP address.
DHCP Configuration

```
delphix network setup update *> set dhcp=true
```

Static Configuration

```
delphix network setup update *> set dhcp=false
```

```
delphix network setup update *> set primaryAddress=<address>/<prefix-len>
```

The static IP address must be specified in CIDR notation (for example, `192.168.1.2/24`)

9. Configure a default gateway. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set defaultRoute=<gateway-ip>
```

10. Commit your changes. Note that you can use the `get` command prior to committing to verify your desired configuration.

```
delphix network setup update *> commit
Successfully committed network settings. Further setup can be done through the
browser at:
        http://<address>
Type "exit" to disconnect, or any other commands to continue using the CLI.
```

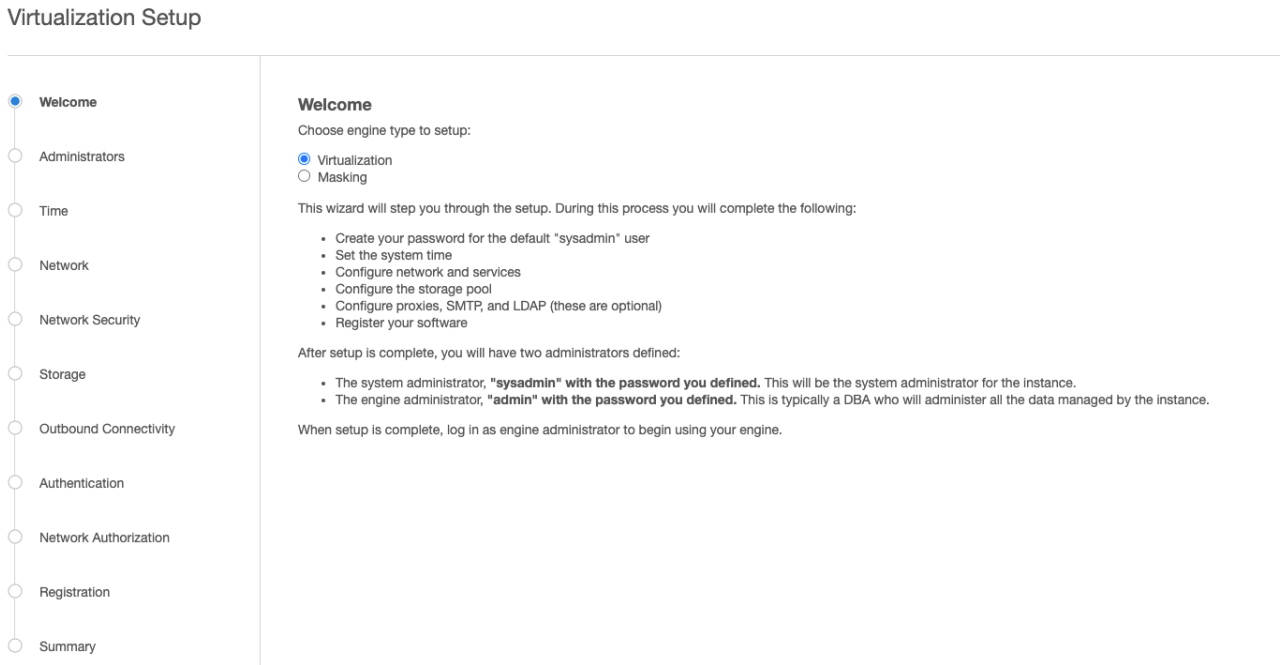
11. Check that you can now access the Delphix Engine through a Web browser by navigating to the displayed IP address, or hostname if using DNS.
12. Exit setup.

```
delphix> exit
```

Setting up the Delphix engine

Once you setup the network access for Delphix Engine, enter Delphix Engine URL in your browser for server setup.

The welcome screen below will appear for you to begin your Delphix Engine setup.



The setup procedure uses a wizard process to take you through a set of configuration screens:

- Administrators
- Time
- Network
- Network Security
- Storage
- Outbound Connectivity
- Authentication
- Network Authorization
- Registration
- Summary

1. Connect to the Delphix Engine at `http:///login/index.html#serverSetup`. The **Delphix setup** application will launch when you connect to the server. Enter your **sysadmin** login credentials, which initially defaults to the username **sysadmin**, with the initial default password of **sysadmin**. On first login, you will be prompted to change the initial default password.
2. Click **Next**.

Administrators

The Delphix Engine supports two types of administrators:

- System Administrator (**sysadmin**) - this is the engine system administrator. The sysadmin password is defined here**.**
- Engine Administrator (**admin)** - this is typically a DBA who will administer all the data managed by the engine.

On the Administrators tab, you set up the sysadmin password by entering an email address and password. The details for the delphix_admin are displayed for reference.

> The default domain user created on Delphix Engines is now **admin** instead of delphix_admin. When engines created before 5.3.1 are upgraded to 5.3.1 or later they will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then Disabling delphix_admin.

System time

Choose your option to setup system time in this section.

For a Quick Start, simply set the time and your timezone. You can change this later.

Network

The initial out-of-the-box network configuration in the OVA file is set to use a single VMXNET3 network adapter.

You've already configured this in the initial configuration. Delphix supports more advanced configurations, but you can enable those later.

Storage

You should see the data storage VMDKs or RDMs you created during the OVA installation. Click Next to configure these for data storage.

Outbound connectivity

Choose your options to configure outbound connectivity settings.

For a Quick Start, take the defaults. You can change this later.

Authentication

Choose your options to configure authentication services.

For a Quick Start, take the defaults. You can change this later.

Registration

If the Delphix Engine has access to the external Internet (either directly or through a web proxy), then you can auto-register the Delphix Engine:

1. Enter your **Support username** and **Support password**.
2. Click **Register**.

If external connectivity is not immediately available, you must perform manual registration.

1. Copy the **Delphix engine registration code** in one of two ways:
    a. Manually highlight the registration code and copy it to clipboard. Or,
    b. Click **Copy registration Code to Clipboard**.
2. Transfer the Delphix Engine's registration code to a workstation with access to the external network Internet. For example, you could e-mail the registration code to an externally accessible e-mail account.
3. On a machine with access to the external Internet, please use your browser to navigate to the Delphix Registration Portal at http://register.delphix.com
4. Login with your Delphix support credentials (username and password).
5. Paste the **Registration code**.
6. Click **Register**.

> ⚠ While your Delphix Engine will work without registration, we strongly recommend that you register each Delphix Engine as part of the setup. Failing to register the Delphix Engine will impact its supportability and security in future versions.

To regenerate the registration code for a Delphix Engine please refer to, Regenerating the Delphix Engine Registration Code. Delphix strongly recommends that you regenerate this code and re-register the engine regularly to maximize the Support Security of the Delphix Engine. Delphix recommends doing this every six months.

Summary

The setup procedure uses a wizard process to take you through a set of configuration screens:

- Administrators
- Time
- Network
- Network Security
- Storage
- Outbound Connectivity
- Authentication
- Network Authorization
- Registration
- Summary

1. Connect to the Delphix Engine at `http:///login/index.html#serverSetup`. The **Delphix Setup** application will launch when you connect to the server. Enter your **sysadmin** login credentials, which initially defaults to the username **sysadmin**, with the initial default password of **sysadmin**. On first login, you will be prompted to change the initial default password.
2. Click **Next**.

Requirements for Oracle hosts and databases

Oracle hosts and databases

To begin using Oracle environments in Delphix, reference this article as a provided outline of the configuration requirements for the source and target hosts.

On each host with Oracle, there must be an operating system user configured to the required specifications for Delphix, as explained in the table below. This user (i.e. delphix_os) can easily be created by using the createDelphixOSUser.sh script (located at the bottom of the page).

These requirements apply to both source and target environments. However, target environments have additional requirements which are detailed in the 'Target Host Requirements' section below.

Source host requirement

Host requirement	**Explanation**
Profile and privileges should be the same as the Oracle user (e.g. oracle) on the host.	For example, delphix_os should have the same environment variable settings ($PATH, $ORACLE_HOME, etc.), umask, and ulimit settings, as the user oracle.
The Delphix software owner account (e.g. delphix_os) must have the same primary OS group as the Oracle software owner account (e.g. oracle).	Delphix recommends giving the delphix_os user the same primary OS group as the Oracle home owner. This ensures the Delphix engine can fully and automatically discover Oracle homes, databases, and listeners. Often, this is an OS group named oinstall. However, the oinstall group is not always necessary depending on your Oracle configuration. This user requires access to the libobk_proxy.so library in the toolkit for the child processes triggered by RMAN.

Host requirement	Explanation
The delphix_os user must have the Oracle OSDBA group (typically dba) as a primary or secondary OS group.	The OSDBA group allows "OS authentication" when connecting to an Oracle database instance by specifying neither username or password (i.e. rman target /), thus eliminating the need to store or retrieve an SYSDBA password. **Oracle 12c** For Oracle 12c and later versions of Oracle, the delphix_os user can also use OSBACKUPDBA as its secondary group. This is typically the BACKUPDBA group on the host.
For secondary group requirements, the Delphix OS user must be a member of the SYSBACKUP OS group (12.1 or higher) or the SYSDBA OS group (11.2 and lower).	This ensures that the Delphix engine is able to take snapshots of source databases using RMAN.
There must be a directory on the source host where the Delphix Engine Toolkit can be installed, for example: /var/opt/delphix/toolkit	The delphix_os user and primary OS group (i.e. oinstall) must own the directory. The directory must have at least permissions -rwxr-x--- (0750) The delphix_os user must have -rwxr-x-- (0750) permissions on each directory in the path leading to the toolkit directory. At least 1.6 GB of storage is needed at the time of setting up the environment and at least 500MB of free space is required to allow refreshes and maintenance of the toolkit, especially during upgrades.
The Delphix Engine must be able to make an SSH connection to the source host (typically port 22).	
Read access to $ORACLE_HOME and all underlying files and directories.	Delphix needs to run locally available Oracle tools such as sqlplus and RMAN. Those executables, as well as various required libraries, reside inside the Oracle home

Additional requirements for RAC environments

If the source host is part of a RAC cluster, Delphix will attempt to use all nodes and crsctl for its operations.

RAC environment requirement	Explanation
The delphix_os user and Delphix Toolkit configuration must be the same on each node in the RAC cluster	• delphix_os must have a execute permission on crsctl and srvctl on each node in the cluster. • access to olsnodes is needed. • The Delphix Toolkit must be installed in the same directory path on each of the nodes in the source cluster. • All data files, archive logs, and database control file must be located on storage shared by all of the cluster nodes. • Each node in the cluster must be able to access archive logs and the database control file from all other nodes. • At least 1.6 GB of storage is needed at the time of setting up the environment and at least 500MB of free space is required to allow refreshes and maintenance of the toolkit, especially during upgrades.

Masked Provisioning

Masked Provisioning is supported on Oracle RAC only when used with "script-based masking".

Auto-discovery requirements

The preferred way to find source databases it to allow Delphix to automatically discover your Oracle Homes and Databases by examining the inventory and oratab files and the listener setup. Successful autodiscovery requires read access to these and related files.

In **most** environments, delphix_os group membership is sufficient to perform auto-discovery.

However, if you have overridden Oracle's group permission structure, you may need to modify privileges to allow auto-discovery.

Auto-discovery requirements	Explanation
The ORATAB file must exist (typically in /etc/oratab or /var/opt/oracle/oratab) and be readable by delphix_os. Read access to: • /etc/orainst.loc or /var/opt/oracle/orainst.loc. • the Oracle inventory file (inventory.xml)	The ORATAB is used to determine the location of the Oracle installations on your host. Note: The Oracle inventory file is identified by the contents of orainst.loc (for example, $INVENTORY_HOME/ContentsXML/inventory.xml).
Permission to run pargs on Solaris hosts and ps on AIX, HP-UX, and Linux hosts, as super-user. This permission is usually granted via sudo authorization of the commands.	Unless you have used a custom TNS_ADMIN setting, elevated access to ps (pargs on Solaris) is not required See the topic Sudo Privilege Requirements for Oracle Environments for further explanation of this requirement, and Sudo File Configuration Examples for Oracle Environments for examples of file configurations

Source database requirements

For each source database, there are specific configurations required for Delphix to ingest data.

Source database requirement	Explanation
For each Oracle Home, the delphix_os user should have execute permission for the programs in $ORACLE_HOME/bin.	If symlinks are configured, Delphix must be configured with the same $ORACLE_HOME path as was used when starting the instance. Ensure the PermitUserEnvironment configuration parameter = "yes" in the sshd_config file The $ORACLE_HOME/bin/oracle executable must have the SETUID and SETGID flags set. Permissions on the oracle binary must be at least -rwsr-s–x (06751).
Source databases must be in ARCHIVELOG mode to ensure that redo logs are archived.	Archive logs are required to make SnapSyncs consistent and provisionable.
Enable Block Change Tracking (BCT). (Highly Recommended).	Enabling BCT will improve SnapSync operation time. Without BCT, incremental SnapSyncs must scan the entire database.
Enable FORCE LOGGING. (Highly Recommended)	If you do not enable FORCE LOGGING and NOLOGGING operations take place, you will get a Fault from the Delphix Engine. If you must use NOLOGGING to meet specific performance criteria, take a new snapshot of the source database after doing the NOLOGGING operations to bring the dSource up-to-date before provisioning VDBs.
If the online redo log files are located on RAW or ASM devices, then LogSync can only operate in Archive Only mode.	See the topics Advanced Data Management Settings for Oracle dSources and Linking Oracle Physical Standby Databases for more information.

Operating system specific requirements

Solaris

On a Solaris host, gtar must be installed. Delphix uses gtar to handle long file names when extracting the toolkit files into the toolkit directory on a Solaris host. The gtar binary should be installed in one of the following directories:

- / bin:/usr
- / bin:/sbin:/usr
- / sbin:/usr/contrib
- / bin:/usr/sfw
- / bin:/opt/sfw
- / bin:/opt/csw/bin

Additional target /staging host requirements

This topic describes the user privileges and environment requirements that are required for Oracle target hosts and databases collectively referred to as target environments.

These are in addition to the 'Oracle Hosts and Database Requirements' called out above.

Target/Staging host requirement	**Explanation**
The Delphix OS user must be a member of the SYSDBA OS group.	This ensures that the Delphix engine is able to create new VDBs on target hosts.
There must be a directory (e.g. "/mnt/provision/"or "/mnt/staging/") that will be used as a container for the NFS mount points that are created when provisioning a VDB.	The delphix_os user and primary OS group (i.e. oinstall or dba) must own the directory. The directory must have at least permissions -rwxrwx--- (0770). There must be no symbolic links in the path of this directory, because NFS can mount into a directory with symlinks in its path, but cannot unmount.
Permission to run: • mount, umount, mkdir, rmdir as super-user. • pargs on Solaris hosts • ps on Linux, AIX, and HP-UX as super-user. • nfso on AIX hosts as super-user	The following permissions are usually granted via sudo authorization of the commands. See Sudo Privilege Requirements for Oracle Environments for further explanation of the commands, and Sudo File Configuration Examples for Oracle Environments for examples of the /etc/sudoers file on different operating systems.
Optional write permission to the $ORACLE_BASE_CONFIG/dbs directory	If write permission is granted, an instance init file will be written to this directory during provisioning that will simplify manual instance startup. If write permission is not granted, manual instance startup must specify the instance init file. See Manually Starting a VDB
An Oracle listener process running on the target host for provisioning a VDB	The listener's version should be equal to or greater than the highest Oracle version that will be used to provision a VDB.
Required packages on target hosts: 1. portmapper / rpcbind 2. status daemon (rpc.statd) 3. NFS lock manager (rpc.lockd/lockmgr)	As the Delphix Engine uses NFSv3 for mounting target host filesystems, the prerequisite packages to support NFSv3 client communication are required for normal operation, and the required services to support NFS client communications (including file locking) must be running. NFSv3 is enabled by default and to enable NFSv4, see NFSv4 Configuration

Target/Staging host requirement	Explanation
If using ASM, the delphix_os user must have the Oracle ASM ownership groups as secondary OS groups (typically asmadmin and asmdba)	This ensures the Delphix engine can fully and automatically discover Oracle homes, databases, and listeners.

Deploy hostchecker to validate Delphix requirements

Delphix has developed a hostchecker script that contains standardized checks for source and target hosts - these checks generally fall into three buckets

- OS and Host permissions/access
- Network Port Checks
- DB-specific functionality

OS and Host permissions/access and network port checks can (and should) be performed prior to Delphix installation to ensure a smooth deployment.

Each DB should have a specifically associated hostchecker - you can find detailed documentation on the DB-specific hostchecker page.

1. Download the appropriate **HostChecker tarball** for your engine from https://download.delphix.com/ Tarballs follow the naming convention "hostchecker_<OS>_<processor>.tar". For example, if you are validating a linux x86 host you should download the hostchecker_linux_x86.tar tarball.
2. Create a working directory and extract the **HostChecker files** from the **HostChecker tarball**.

   ```
   mkdir dlpx-host-checker
   cd dlpx-host-checker/
   tar -xf hostchecker_linux_x86.tar
   ```

3. Run the `sh` script contained within:

   ```
   sh hostchecker.sh
   ```

   This will extract the JDK included in the tarball (if necessary) and invoke the HostChecker.

   ```
   ora10205@bbdhcp:/home/ora10205/hostchecker-> sh hostchecker.sh
   Extracting the JDK from the tarball jdk-6u45-linux-i586.tar.gz.
   ```

   Warning:
   Don't run as root
   Do not run the HostChecker as root; this will cause misleading or incorrect results from many of the checks.
4. Select which **checks**you want to run.
   Note:
   Run Tests without the Interface
   You can also run checks without spawning the interface. Enter `--help` to get a list of arguments you can pass to the HostChecker.
5. As the checks are made, enter the requested **arguments**.

6. Read the output of the check. The general format is that severity increases as you scroll down the output. First comes informational output, then warnings, then errors.
   Note:
   Internal Errors
   If you see a message that starts with `Internal Error`, forward it to Delphix Support immediately. This represents a potential bug in the HostChecker, and not necessarily a problem with your environment.
7. Error or warning messages will explain any possible problems and how to address them. Resolve the issues that the HostChecker describes. Do not be surprised or undo your work if more errors appear the next time you run HostChecker, because the error you just fixed may have been masking other problems.
8. Repeat steps 3 - 7 until all the checks return no errors or warnings.

Adding Oracle source and target environments

Follow the steps below to add **both** source and target environments for Oracle.

1. Login to the **Delphix management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Click the **Plus** icon next to **Environments**.
5. In the **Add environment** dialog, select **Unix/Linux**.
6. Select **Standalone host** or **Oracle cluster**, depending on the type of environment you are adding.
7. For standalone Oracle environments enter the **Host IP** address.
8. For Oracle RAC environments, enter the **Node address** and **Cluster home.**
9. Enter an optional **Name** for the environment.
10. Enter the **SSH** port. The default value is **22**.
11. Enter a **Username** for the environment. See Requirements for Oracle Target Hosts and Databases for more information on the required privileges for the environment user.
12. Select a Login Type: — Username and Password - enter the OS username and password, or — Username and Public Key - enter the OS username. — Password Vault - select from an existing Enterprise Password Vault
    Note:
    Using public key authentication
    If you want to use public-key encryption for logging into your Unix-based environment:
    a. Select **Public key** for the **Login type.**
    b. Click **View public key**.
    c. Copy the public key that is displayed, and append it to the end of your `~/.ssh/authorized_keys` file. If this file does not exist, you will need to create it.
       i. Run chmod 600 ~/.ssh/authorized_keys to allow only the file's owner to read and write to it (make sure the file is owned by the user).
       ii. Run chmod 755 ~ to restrict access to the user's home directory so no other user may write to it.
       iii. Run chmod 700 ~/.ssh so that others cannot write to it. The ~/.ssh directory cannot be writable by group or other users. Otherwise, authentication will fail.
    The public key needs to be added only once per user and per environment.
13. For **Password login**, click **Verify credentials** to test the username and password.
14. Enter a **Toolkit path**. The toolkit directory stores scripts used for Delphix Engine operations, and should have a persistent working directory rather than a temporary one. The toolkit directory will have a separate subdirectory for each database instance. The toolkit path must have 0770 permissions and at least 345MB of free space.
15. Click **Submit**.

Linking an Oracle data source

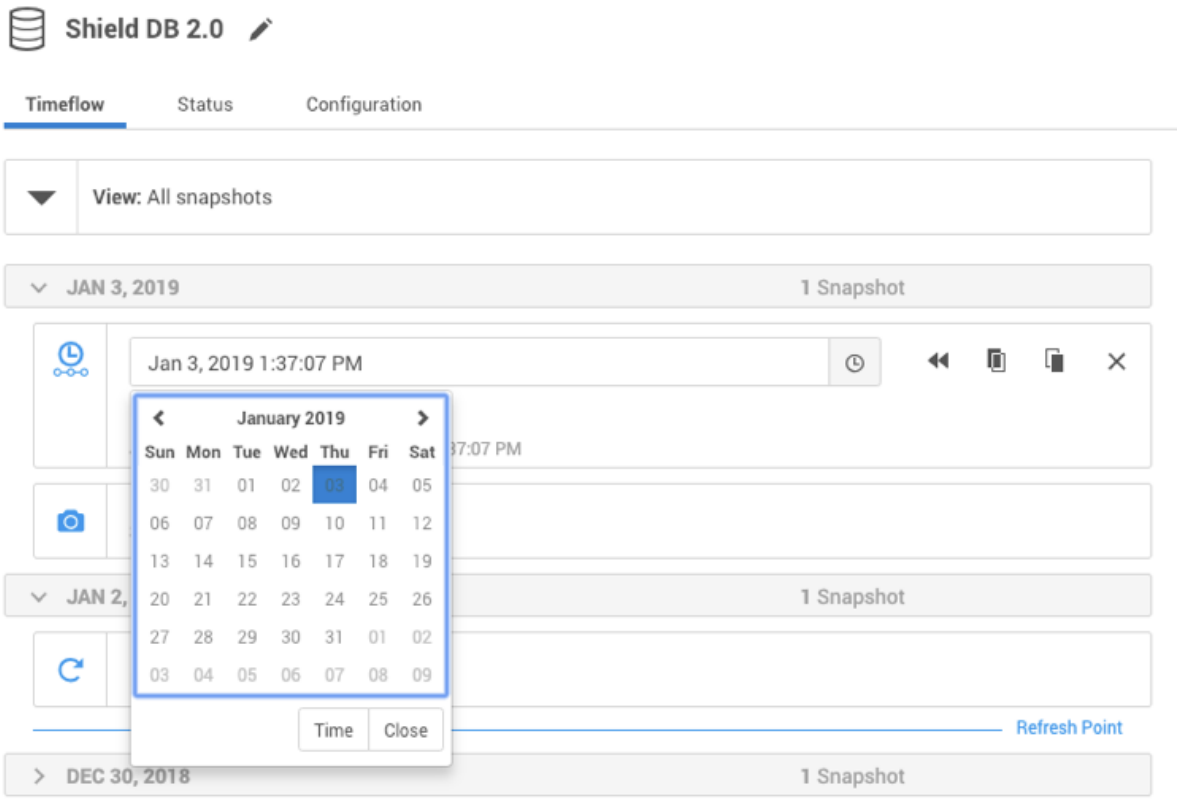
1. Login to the **Delphix management** application.
2. Navigate to **Manage > Datasets.**
3. Click the plus icon and select **Add dSource**.
4. In the **Add dSource** wizard, select the source database with the correct environment user-specified.
5. Enter your login credentials for the source database and click **Verify credentials**. If you are linking a mounted standby, Click **Next**. See the topics under Linking Oracle Physical Standby Databases for more information about how the Delphix Engine uses non-SYS login credentials.
6. Enter a name and select a group for your dSource. Adding a dSource to a dataset group lets you set Delphix Domain user permissions for that database and its objects, such as snapshots. See the topics under Users and Groups for more information.
7. Select the **Data management** settings needed. For more information, visit Data Management Settings for Oracle Data Sources
8. Assign existing policies to the new dSource. New policies can be created and associated later.
9. Enter any scripts that should be run using the **Hooks** page.
10. Review the dSource Configuration and Data Management information, and then click **Submit**.

Once the action to add a dSource has been submitted, the Delphix Engine will initiate a DB_Link job to create the dSource. If the *Load Immediately* option was selected in the data management page a DB_Sync job will also be executed to ingest data from the source, otherwise, this first DB_Sync job will run as per the associated SnapSync policy.

When the jobs have successfully completed, the database icon will change to a dSource icon on the Environments > Databases screen, and the dSource will be added to the list of Datasets under its assigned group.

Provisioning an Oracle VDB

1. Login to the **Delphix management** application.
2. Select **Manage > Datasets**.
3. In the **Datasets** panel on the left-hand side, click the **group** containing the dSource or VDB from which you want to provision.
4. Click the **TimeFlow** tab.
5. Select a **snapshot**.
   Note:
   You can take a snapshot of the dSource from which to provision. To do so, click the **Camera** icon.
6. **Optional:** Select  to open LogSync timeline.



7. Select [ ] to provision from a point of time within a snapshot. You can select by date or time.
8. Click [ ] and the **Provision VDB**wizard will open:
    a. For **Oracle single instance** the fields **Installation home**, **Database unique name**, **SID**, **Database name**, **Mount base**, and **Environment user** will auto-populate with information from the parent.
    b. For **Oracle RAC** the fields **Installation home**, **Database unique name**, **SID**, **Database name**, **Mount base**, **Instance number, Instance name** and **Environment user** will auto-populate with information from the parent.
    Note:
    Editable Fields in the VDB Provision Wizard
    The following fields are editable:
    Installation Home (need to have an additional compatible target)
    Database Unique Name
    SID
    Database Name
    Mount Base
    Instance Number (RAC Only)
    Instance Name (RAC Only)
9. If you need to add a new target environment for the VDB, click the green **Plus** icon next to the **Filter target** field, and follow the instructions in Adding an Oracle Single Instance or RAC Environment
10. Review the information for **Installation home**, **Database unique name**, **SID**, and **Database name**. Edit as necessary.
11. Review the **Mount base** and **Environment user**. Edit as necessary. The Environment User must have permission to write to the specified Mount Base, as described in Requirements for Oracle Target Hosts and Databases. You may also want to create a new writeable directory in the target environment with the correct permissions and use that as the Mount Base for the VDB.

12. Select **Provide privileged credentials** if you want to use login credentials on the target environment that are different from those associated with the **Environment user**.
13. Click **Advanced** to customize the VDB online log size and log groups, archivelog mode, local_listener parameter (TCP/IPC protocol addresses), additional VDB configuration settings or file mappings, or custom environment variables. For more information, see Customizing Oracle VDB Configuration Settings, Customizing VDB File Mappings, and Customizing Oracle VDB Environment Variables
    If you are provisioning to a target environment that is running a Linux OS, you will need to compare the `SGA_TARGET` configuration parameter with the shared memory size in `/dev/shm`. The shared memory configured on the target host should match the SGA memory target. You can check the Linux OS shared memory size with the command df -k /dev/shm and the SGA_TARGET configuration parameter by opening the **Advanced** settings and then finding the value for `SGA_TARGET` under **VDB Configuration templates**.
14. Click **Next**.
15. Select a **Target group** for the VDB.
16. Enable Auto VDB Restart to allow the VDB to be automatically restarted when the target host reboot is detected by Delphix.
17. Click Next.
18. Select a **Snapshot policy** for the VDB.
19. Click **Next**.
20. Enter any operations that should be run at Hooks during the provisioning process. Click **Next**.
21. Click **Submit**.

When provisioning starts, you can review the progress of the job by selecting the VDB and clicking on the **Status** tab, or by selecting **Manage/Dashboards** and viewing the **Job history** panel. Alternatively, you could see this in the **Actions sidebar.** When provisioning is complete, the VDB will be included in the group you designated and listed in the **Datasets** panel. If you select the VDB in the **Datasets** panel and click the **Configuration** tab, you can view information about the database and its Data Management settings.

Next steps

Congratulations! You've provisioned your first virtual database!

You should attempt some simple functional tests with your application. You can connect your app to the VDB using standard TNS/JDBC techniques. Delphix has already registered the VDB for you on the Target listener.

We suggest the following next steps:

- dropping a table and using the VDB Rewind feature to test recovery of your VDB
- Take a snapshot of your dSource and Refresh your VDB to quickly get fresh production data
- Provision a new VDB from your VDB to test sharing data quickly with other teams
- Mask your new VDB to protect sensitive data. Provision new VDBs from that Masked VDB to quickly provide safe data to development and QA teams.

Script

- createDelphixOSUser.sh

Oracle support and requirements

This section covers the following topics:

- Requirements for Oracle hosts and databases
- Network and connectivity requirements for Oracle environments
- Sudo privilege requirements for Oracle environments
- Sudo file configuration examples for Oracle environments

To view Oracle support matrix, see Oracle matrix.

Requirements for Oracle hosts and databases

Oracle hosts and databases

To begin using Oracle environments in Delphix, reference this article as a provided outline of the configuration requirements for the source and target hosts.

On each host with Oracle, there must be an operating system user configured to the required specifications for Delphix, as explained in the table below. This user (i.e. delphix_os) can easily be created by using the createDelphixOSUser.sh script (located at the bottom of the page).

These requirements apply to both source and target environments. However, target environments have additional requirements which are detailed in the 'Target Host Requirements' section below.

Source host requirement

Host requirement	**Explanation**
Profile and privileges should be the same as the Oracle user (e.g. oracle) on the host.	For example, delphix_os should have the same environment variable settings ($PATH, $ORACLE_HOME, etc.), umask, and ulimit settings, as the user oracle.
The Delphix software owner account (e.g. delphix_os) must have the same primary OS group as the Oracle software owner account (e.g. oracle).	Delphix recommends giving the delphix_os user the same primary OS group as the Oracle home owner. This ensures the Delphix engine can fully and automatically discover Oracle homes, databases, and listeners. Often, this is an OS group named oinstall. However, the oinstall group is not always necessary depending on your Oracle configuration. This user requires access to the libobk_proxy.so library in the toolkit for the child processes triggered by RMAN.
The delphix_os user must have the Oracle OSDBA group (typically dba) as a primary or secondary OS group.	The OSDBA group allows "OS authentication" when connecting to an Oracle database instance by specifying neither username or password (i.e. rman target /), thus eliminating the need to store or retrieve an SYSDBA password. **Oracle 12c** For Oracle 12c and later versions of Oracle, the delphix_os user can also use OSBACKUPDBA as its secondary group. This is typically the BACKUPDBA group on the host.
For secondary group requirements, the Delphix OS user must be a member of the SYSBACKUP OS group (12.1 or higher) or the SYSDBA OS group (11.2 and lower).	This ensures that the Delphix engine is able to take snapshots of source databases using RMAN.

Host requirement	Explanation
There must be a directory on the source host where the Delphix Engine Toolkit can be installed, for example: /var/opt/delphix/toolkit	The delphix_os user and primary OS group (i.e. oinstall) must own the directory. The directory must have at least permissions -rwxr-x--- (0750) The delphix_os user must have -rwxr-x-- (0750) permissions on each directory in the path leading to the toolkit directory. At least 1.6 GB of storage is needed at the time of setting up the environment and at least 500MB of free space is required to allow refreshes and maintenance of the toolkit, especially during upgrades.
The Delphix Engine must be able to make an SSH connection to the source host (typically port 22).	
Read access to $ORACLE_HOME and all underlying files and directories.	Delphix needs to run locally available Oracle tools such as sqlplus and RMAN. Those executables, as well as various required libraries, reside inside the Oracle home

Additional requirements for RAC environments

If the source host is part of a RAC cluster, Delphix will attempt to use all nodes and crsctl for its operations.

RAC environment requirement	Explanation
The delphix_os user and Delphix Toolkit configuration must be the same on each node in the RAC cluster	• delphix_os must have a execute permission on crsctl and srvctl on each node in the cluster. • access to olsnodes is needed. • The Delphix Toolkit must be installed in the same directory path on each of the nodes in the source cluster. • All data files, archive logs, and database control file must be located on storage shared by all of the cluster nodes. • Each node in the cluster must be able to access archive logs and the database control file from all other nodes. • At least 1.6 GB of storage is needed at the time of setting up the environment and at least 500MB of free space is required to allow refreshes and maintenance of the toolkit, especially during upgrades.

Masked Provisioning

Masked Provisioning is supported on Oracle RAC only when used with "script-based masking".

Auto-Discovery Requirements

The preferred way to find source databases it to allow Delphix to automatically discover your Oracle Homes and Databases by examining the inventory and oratab files and the listener setup. Successful autodiscovery requires read access to these and related files.

In **most** environments, delphix_os group membership is sufficient to perform auto-discovery.

However, if you have overridden Oracle's group permission structure, you may need to modify privileges to allow auto-discovery.

Auto-discovery requirements	**Explanation**
The ORATAB file must exist (typically in /etc/oratab or /var/opt/oracle/oratab) and be readable by delphix_os. Read access to: • /etc/orainst.loc or /var/opt/oracle/orainst.loc. • the Oracle inventory file (inventory.xml)	The ORATAB is used to determine the location of the Oracle installations on your host. Note: The Oracle inventory file is identified by the contents of orainst.loc (for example, $INVENTORY_HOME/ContentsXML/inventory.xml).
Permission to run pargs on Solaris hosts and ps on AIX, HP-UX, and Linux hosts, as super-user. This permission is usually granted via sudo authorization of the commands.	Unless you have used a custom TNS_ADMIN setting, elevated access to ps (pargs on Solaris) is not required See the topic Sudo Privilege Requirements for Oracle Environments for further explanation of this requirement, and Sudo File Configuration Examples for Oracle Environments for examples of file configurations

Source database requirements

For each source database, there are specific configurations required for Delphix to ingest data.

Source database requirement	**Explanation**
For each Oracle Home, the delphix_os user should have execute permission for the programs in $ORACLE_HOME/bin.	If symlinks are configured, Delphix must be configured with the same $ORACLE_HOME path as was used when starting the instance. Ensure the PermitUserEnvironment configuration parameter = "yes" in the sshd_config file The $ORACLE_HOME/bin/oracle executable must have the SETUID and SETGID flags set. Permissions on the oracle binary must be at least -rwsr-s–x (06751).

Source database requirement	Explanation
Source databases must be in ARCHIVELOG mode to ensure that redo logs are archived.	Archive logs are required to make SnapSyncs consistent and provisionable.
Enable Block Change Tracking (BCT). (Highly Recommended).	Enabling BCT will improve SnapSync operation time. Without BCT, incremental SnapSyncs must scan the entire database.
Enable FORCE LOGGING. (Highly Recommended)	If you do not enable FORCE LOGGING and NOLOGGING operations take place, you will get a Fault from the Delphix Engine. If you must use NOLOGGING to meet specific performance criteria, take a new snapshot of the source database after doing the NOLOGGING operations to bring the dSource up-to-date before provisioning VDBs.
If the online redo log files are located on RAW or ASM devices, then LogSync can only operate in Archive Only mode.	See the topics Advanced Data Management Settings for Oracle dSources and Linking Oracle Physical Standby Databases for more information.

Operating system specific requirements

Solaris

On a Solaris host, gtar must be installed. Delphix uses gtar to handle long file names when extracting the toolkit files into the toolkit directory on a Solaris host. The gtar binary should be installed in one of the following directories:

- / bin:/usr
- / bin:/sbin:/usr
- / sbin:/usr/contrib
- / bin:/usr/sfw
- / bin:/opt/sfw
- / bin:/opt/csw/bin

Additional target /staging host requirements

This topic describes the user privileges and environment requirements that are required for Oracle target hosts and databases collectively referred to as target environments.

These are in addition to the ‘Oracle Hosts and Database Requirements’ called out above.

Target/Staging host requirement	Explanation
The Delphix OS user must be a member of the SYSDBA OS group.	This ensures that the Delphix engine is able to create new VDBs on target hosts.

Target/Staging host requirement	Explanation
There must be a directory (e.g. "/mnt/provision/" or "/mnt/staging/") that will be used as a container for the NFS mount points that are created when provisioning a VDB or linking a staging push dSource.	The delphix_os user and primary OS group (i.e. oinstall or dba) must own the directory. The directory must have at least permissions -rwxrwx--- (0770). There must be no symbolic links in the path of this directory, because NFS can mount into a directory with symlinks in its path, but cannot unmount.
Permission to run: • mount, umount, mkdir, rmdir as super-user. • pargs on Solaris hosts • ps on Linux, AIX, and HP-UX as super-user. • nfso on AIX hosts as super-user	The following permissions are usually granted via sudo authorization of the commands. See Sudo Privilege Requirements for Oracle Environments for further explanation of the commands, and Sudo File Configuration Examples for Oracle Environments for examples of the /etc/sudoers file on different operating systems.
Optional write permission to the $ORACLE_BASE_CONFIG/dbs directory	If write permission is granted, an instance init file will be written to this directory during provisioning that will simplify manual instance startup. If write permission is not granted, manual instance startup must specify the instance init file. See Manually Starting a VDB
An Oracle listener process running on the target host for provisioning a VDB	The listener's version should be equal to or greater than the highest Oracle version that will be used to provision a VDB.
Required packages on target hosts: 1. portmapper / rpcbind 2. status daemon (rpc.statd) 3. NFS lock manager (rpc.lockd/lockmgr)	As the Delphix Engine uses NFSv3 for mounting target host filesystems, the prerequisite packages to support NFSv3 client communication are required for normal operation, and the required services to support NFS client communications (including file locking) must be running. NFSv3 is enabled by default and to enable NFSv4, see NFSv4 Configuration
If using ASM, the delphix_os user must have the Oracle ASM ownership groups as secondary OS groups (typically asmadmin and asmdba)	This ensures the Delphix engine can fully and automatically discover Oracle homes, databases, and listeners.

Scripts

- createDelphixOSUser.sh
- createDelphixDBUser.sh

Network and connectivity requirements for Oracle environments

General port allocation

The Delphix Engine makes use of the following network ports regardless of the type of database platform:

General outbound from the Delphix engine port allocation

Protocol	Port numbers	Use
TCP	25	Connection to a local SMTP server for sending email
TCP/UDP	53	Connections to local DNS servers
UDP	123	Connection to an NTP server
UDP	162	Sending SNMP TRAP messages to an SNMP Manager
TCP	443	HTTPS connections from the Delphix Engine to the Delphix Support upload server
TCP/UDP	636	Secure connections to an LDAP server
TCP	8415	Connections to a Delphix replication target. See Configuring Replication
TCP	50001	Connections to source and target environments for network performance tests.

General inbound to the Delphix engine port allocation

Protocol	Port number	Use
TCP	22	SSH connections to the Delphix Engine
TCP	80	HTTP connections to the Delphix GUI
UDP	161	Messages from an SNMP Manager to the Delphix Engine
TCP	443	HTTPS connections to the Delphix Management Application

Protocol	Port number	Use
TCP	8415	Delphix Session Protocol connections from all DSP-based network services including Replication, SnapSync for Oracle, V2P, and the Delphix Connector.
TCP	50001	Connections from source and target environments for network performance tests via the Delphix CLI.

Firewalls and intrusion detection systems (IDS)

Production databases on source environments (for dSources) are often separated from the non-production environment by firewalls. Firewalls can add milliseconds to the latency between servers. Accordingly, for best performance, there should be no firewalls between the Delphix Engine and the virtual database (VDB) target environments. If the Delphix Engine is separated from a source environment by a firewall, the firewall must be configured to permit network connections between the Delphix Engine and the source environments for the application protocols (ports) listed above.

Intrusion detection systems (IDSs) should also be made permissive to the Delphix Engine deployment. IDSs should be made aware of the anticipated high volumes of data transfer between dSources and the Delphix Engine.

SSHD configuration

Both source and target Unix environments are required to have `sshd` running and configured such that the Delphix Engine can connect over `ssh`.

The Delphix platform expects to maintain long-running, highly performant `ssh` connections with remote Unix environments. The following `sshd` configuration entries can interfere with these `ssh` connections and are therefore disallowed:

Disallowed `sshd` configuration entries
`ClientAliveInterval`
`ClientAliveCountMax`

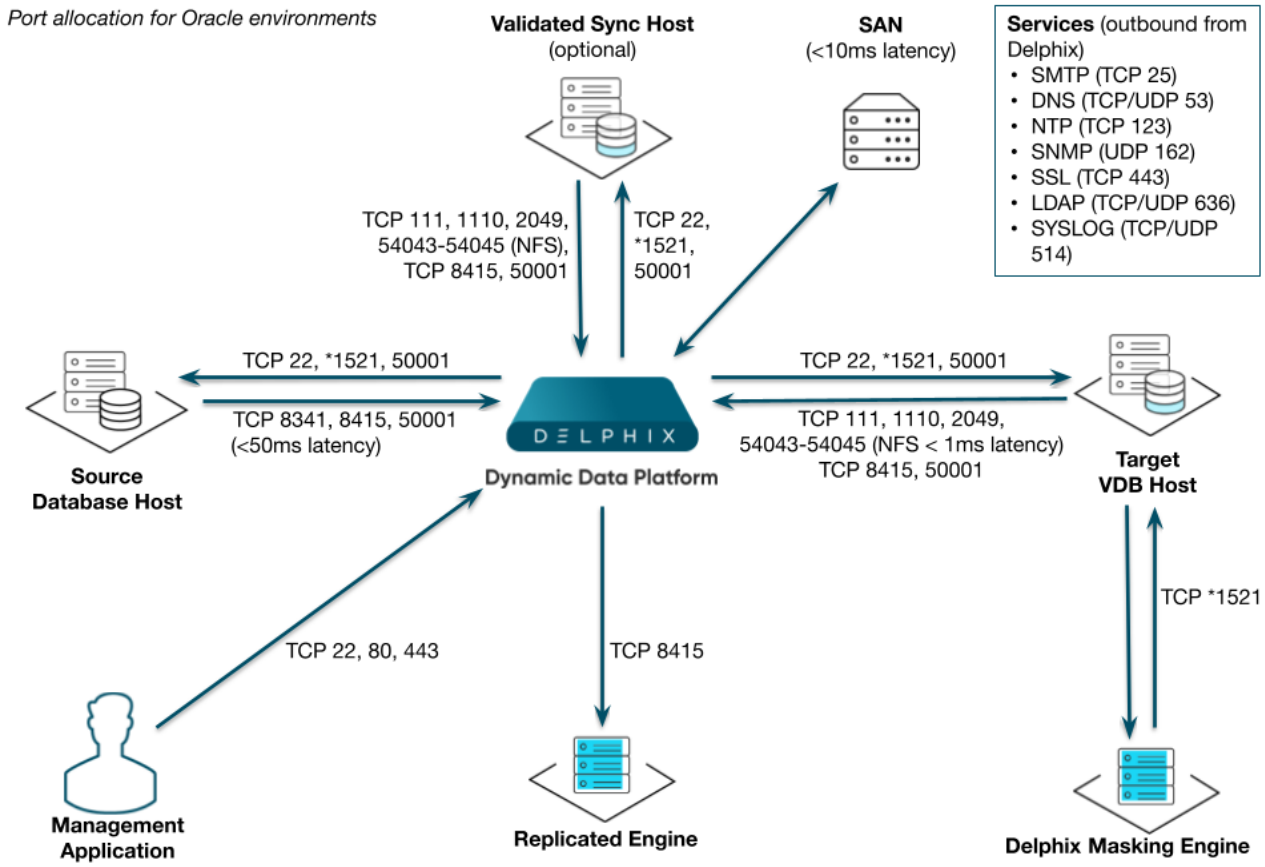
Network and connectivity requirements for Oracle

- IP connections must exist between the Delphix Engine and the source and target environments.
- For source environments, Delphix Engine uses an **SSH** connection to each source host, an **HTTP** connection from each source environment to Delphix Engine, and a DSP connection to the Delphix Engine. The Delphix Engine uses ***SQLNet*** *connections to the DBMS on the source environment.*
- *For target environments, Delphix uses an **SSH** connection to each target environment and an **NFS** connection to Delphix Engine. Delphix Engine uses **SQLNet*** connections to the virtual databases on the target environment.

> **scp Availability**
> The **scp** program must be available in the environment in order to add an environment.

Port allocation for Oracle environments

The following diagram describes the port allocations for Oracle environments. It illustrates the ports that we recommend to be open from Delphix to remote services, to the Delphix Engine, and to the Target Environments.



Note: *Oracle listener typically runs on TCP port 1521. In cases where other ports are used, substitute for 1521 above. Starting with 6.0.7.0 release, this port is required only if database user credentials are provided. The port need not be open if no database user credentials are provided.

The Delphix Engine makes use of the following network ports for Oracle dSources and VDBs:

Outbound from the Delphix engine port allocation

Protocol	Port numbers	Use
TCP	22	SSH connections to source and target environments
TCP	xxx	Connections to the Oracle SQL*Net Listener on the source and target environments (typically port 1521)

Inbound to the Delphix engine port allocation

Protocol	Port number	Use
TCP/UDP	111	Remote Procedure Call (RPC) port mapper used for NFSv3 mounts Note: RPC calls in NFSv3 use additional fixed ports for supporting services (lockd, mountd and statd) seen below.
TCP	1110	NFSv3 Server daemon status and NFS server daemon keep-alive (client info)
TCP/UDP	2049	NFS Server daemon from VDB to the Delphix Engine (used for both NFSv3 and NFSv4)
TCP	8341	Sending data from source to the Delphix Engine (for LogSync)
TCP	8415	SnapSync control and data from source to the Delphix Engine V2P control and data from the target environment to the Delphix Engine.
TCP	54043	NFSv3 mount daemon
TCP	54044	NFSv3 stat daemon (lock state notification service)
TCP	54045	NFSv3 lock daemon/manager
UDP	33434 - 33464	Traceroute from source and target database servers to the Delphix Engine (optional)

AppData port requirements

The use of AppData requires the following ports/protocols. Two important notes about these specifications:

1. The next release of the Delphix Engine will significantly augment the port/protocol utilization of AppData. The upcoming-only requirements have been marked with a *.
2. AppData V2P uses RSYNC to export to the target. RSYNC between the target and Delphix Engine is not required for general virtualization usage. The V2P-only requirements have been marked with a ^.

From source to Delphix engine	From Delphix engine to source	From target to Delphix engine	From Delphix engine to target
RSYNC (TCP Port 873)	RSYNC (TCP Port 873)	DSP (Default TCP Port 8415)	DSP (Default TCP Port 8415)
DSP (Default TCP Port 8415)	SSH (TCP Port 22)	NFS	SSH (TCP Port 22)
*NFS	DSP (Default TCP Port 8415)	^RSYNC (TCP Port 873)	^RSYNC (TCP Port 873)

Sudo privilege requirements for Oracle environments

This topic describes the rationale behind specific `sudo` privilege requirements for virtualizing Oracle Databases.

Privilege	Sources	Targets	Rationale
`ps | pargs`	Optional, Strongly Recommended	Optional, Strongly Recommended	Delphix auto-discovery uses the `TNS_ADMIN` environment variable of Oracle Listener processes with non-standard configurations to derive their connection parameters. A different user (oracle), rather than **delphix_os** user normally owns the oracle listener. The Delphix Engine needs `sudo` access to `pargs` on the Solaris OS or `ps` on other OSes to determine the environment variables of those Listener processes. This privilege is required for Auto-Discovery with non-default `TNS_ADMIN` locations. It is optional when using a standard `TNS_ADMIN` location, or if you choose to manually configure Oracle Homes, databases and listeners.
`mount/umount`	Not Required	**Required**	The Delphix Engine dynamically mounts and unmounts directories under the provisioning directory during VDB operations. This privilege is required because `mount` and `umount` are typically reserved for superuser.
`nfso` (AIX only)	Not Required	**Required**	The Delphix Engine monitors NFS read and write sizes on an AIX target host. It uses the `nfso` command to query the sizes in order to optimize NFS performance for VDBs running on the target host. Only a superuser can issue the `nfso` command.

⚠ It is required to specify the NOPASSWD qualifier within the "sudo" configuration file, as shown here: Sudo File Configuration Examples for Oracle Environments. This ensures that the "sudo" command does not demand the entry of a password, even for the "display permissions" (i.e. "sudo -l") command. Delphix issues "sudo -l" in some scripts to detect if the operating system user has the correct sudo privileges. If it is unable to execute this command, some actions may fail and Delphix will raise an alert suggesting it does not have the correct sudo permissions. Restricting the execution of "sudo -l" by setting “listpw=always” in the “/etc/sudoers” file ***when the Delphix operating system user is configured to use***

public key authentication will cause the Delphix operating system user to be prompted for a password which will fail certain Delphix actions. Use a less restrictive setting for listpw than "always" when the Delphix operating system user is using public-key authentication.

Oracle mount options for RAC	
AIX	`cio,rw,bg,hard,nointr,timeo=600,proto=tcp,nosuid,noac`
HPUX	`rw,bg,hard,rsize=1048576,wsize=1048576,nointr,timeo=600 ,proto=tcp,nosuid,forcedirectio,noac`
Solaris	`rrw,bg,hard,rsize=1048576,wsize=1048576 ,nointr,proto=tcp,nosuid,forcedirectio,noac`
For the above platforms, depending on NFS version used, additional options vers=3 or vers=4.x is added (x varies depending on what that platform supports. e.g. vers=4 or vers=4.1)	
Linux (NFSv3)	`-t nfs rw,bg,hard,rsize=1048576,wsize=1048576,nointr,timeo=600,tcp ,nosuid,sec=sys,vers=3,actimeo=0`
Linux (NFSv4)	`-t nfs4 rw,bg,hard,rsize=1048576,wsize=1048576,nointr,timeo=600,tc p,nosuid,sec=sys,actimeo=0`

1. For both Single instance and RAC, "port=2049" option is added for all platforms.
2. For AIX, rsize=<value>,wsize=<value> options are added depending on the value returned by "/usr/sbin/nfso -o nfs_max_read_size" and "/usr/sbin/nfso -o nfs_max_write_size" commands.

Oracle unmount options	"-f" is used for all platforms. For Linux, "-lf" is used.
Oracle mount options for single instance	
AIX	`cio,rw,bg,hard,intr,timeo=600,proto=tcp,nosuid`

Oracle mount options for single instance	
HPUX	`rw,bg,hard,rsize=1048576,wsize=1048576,nointr,timeo=600,proto=tcp,nosuid,forcedirectio`
Solaris	`rw,bg,hard,rsize=1048576,wsize=1048576,nointr,proto=tcp,nosuid,forcedirectio`
For the above platforms, depending on NFS version used, additional options vers=3 or vers=4.x is added (x varies depending on what that platform supports. e.g. vers=4 or vers=4.1)	
Linux (NFSv3)	`-t nfs rw,bg,hard,rsize=1048576,wsize=1048576,nointr,timeo=600,tcp,nosuid,sec=sys,vers=3`
Linux (NFSv4)	`-t nfs4 rw,bg,hard,rsize=1048576,wsize=1048576,nointr,timeo=600,tcp,nosuid,sec=sys`

Sudo file configuration examples for Oracle environments

This topic provides a sample `sudo` file privilege configurations for using the Delphix Engine with various operating systems and the Oracle RDBMS.

> **Requiretty settings**
> Delphix requires that the `requiretty` setting be disabled for all Delphix users with `sudo` privileges.

Configuring `sudo` access on Solaris SPARC for Oracle source and target environments

Sudo access to `pargs` on the Solaris operating system is required for the detection of listeners with non-standard configurations on both source and target environments. Super-user access level is needed to determine the `TNS_ADMIN` environment variable of the user running the listener (typically **oracle**, the installation owner). From `TNS_ADMIN`, the Delphix OS user **delphix_os** can derive connection parameters.

Example: Solaris /etc/sudoers entries for a Delphix Source

```
Defaults:delphix_os !requiretty
delphix_os ALL=NOPASSWD:/usr/bin/pargs
```

On a Solaris target, `sudo` access to `mount` and `umount` is also required.

Example: Solaris /etc/sudoers entries for a Delphix Target

```
# Delphix issues sudo -l so we need to allow it via listpw. Never set it to always
when using public key authentication
Defaults      listpw=all

User_Alias DELPHIX_USER=delphix_os

Cmnd_Alias DELPHIX_CMDS= \
/usr/bin/mount, \
/usr/bin/umount, \
/usr/bin/mkdir, \
/usr/bin/rmdir, \
/usr/bin/pargs

DELPHIX_USER ALL=(ALL) NOPASSWD: DELPHIX_CMDS
```

Configuring `sudo` access on Linux for Oracle source and target environments

Sudo access to `ps` on the Linux operating system is required for the detection of listeners with non-standard configurations on both source and target environments. Super-user access level is needed to determine the `TNS_ADMIN` environment variable of the user running the listener (typically **oracle**, the installation owner). From `TNS_ADMIN`, the Delphix OS user **delphix_os** can derive connection parameters.

Example: Linux /etc/sudoers entries for a Delphix Source

```
Defaults:delphix_os !requiretty
delphix_os ALL=NOPASSWD:/bin/ps
```

On a Linux target, sudo access to `mount` and `umount` is also required.

Example: Linux /etc/sudoers file for a Delphix Target

```
Defaults:delphix_os !requiretty
delphix_os ALL=NOPASSWD: \
/bin/mount, /bin/umount, /bin/mkdir, /bin/rmdir, /bin/ps
```

Configuring `sudo` access on AIX for Oracle source and target environments

Sudo access to `ps` on the AIX operating system is required for the detection of listeners with non-standard configurations on both source and target environments. Super-user access level is needed to determine the `TNS_ADMIN` environment variable of the user running the listener (typically **oracle**, the installation owner). From `TNS_ADMIN`, the Delphix OS user **delphix_os** can derive connection parameters.

Example: AIX /etc/sudoers entries for a Delphix Source

```
Defaults:delphix_os !requiretty
delphix_os ALL=NOPASSWD:/bin/ps
```

In addition to sudo access to the `mount`, `umount`, and `ps` commands on AIX target hosts, Delphix also requires `sudo` access to `nfso`. This is required on target hosts for the Delphix Engine to monitor the NFS read-write sizes configured on the AIX system. Super-user access level is needed to run the `nfso` command.

Example: AIX /etc/sudoers File for a Delphix Target

```
Defaults:delphix_os !requiretty
delphix_os ALL=NOPASSWD: \
/usr/sbin/mount, \
/usr/sbin/umount, \
/usr/sbin/mkdir, \
/usr/sbin/rmdir, \
/usr/sbin/nfso, \
/usr/bin/ps
```

Configuring `sudo` access on HP-UX for Oracle source and target environments

No `sudo` privileges are required on source environments running HP-UX. The HP-UX OS does not allow the **delphix_os** user to determine the `TNS_ADMIN` environment variable setting for the **oracle** user. This means that the Delphix Engine cannot auto-discover non-standard listener configurations with non-default `TNS_ADMIN` settings.

On the HP-UX target, `sudo` access to `mount` and `umount` is required as with other operating systems.

Example: HP-UX /etc/sudoers file for a Delphix target

Example: AIX /etc/sudoers File for a Delphix target

```
Defaults:delphix_os !requiretty
delphix_os ALL=NOPASSWD:/sbin/mount, /sbin/umount, /sbin/mkdir, /sbin/rmdir
```

Examples of Limiting `sudo` Access for the Delphix OS User

In situations where security requirements prohibit giving the Delphix user root privileges to mount, unmount, make a directory, and remove directory on the global level, it is possible to configure the `sudoers` file to provide these privileges only on specific mount points or from specific Delphix Engines, as shown in these two examples.

The Delphix Engine tests its ability to run the `mount` command using `sudo` on the target environment by issuing the `sudo mount` command with no arguments. Many of the examples shown in this topic do not allow that. This causes a warning during environment discovery and monitoring but otherwise does not cause a problem. If your VDB operations succeed, it is safe to ignore this warning.
Similarly, the `ps` or `pargs` command is used for target environment operations such as initial discovery and refresh. The most restrictive sudo setups might not allow the commands `ps (pargs).` Delphix can still function without these privileges, although auto-discovery may not work.
However, some users configure the security on the target environments to monitor `sudo` failures and lockout the offending account after some threshold. In those situations, the failure of the sudo commands might cause the **delphix_os** account to become locked. One workaround for this situation is to increase the threshold for locking out the user account. Another option is to modify `/etc/sudoers` to permit the **delphix_os** user to run `ps (pargs),` and `mount` command without parameters.

Note that the following examples are for illustrative purposes and the sudo file configuration options are subject to change.

Example 1

This example restricts the **delphix_os** user's use of `sudo` privileges to the directory `/oracle`.

Note that wildcards are allowed for the options on `mount` and `umount` because those commands expect a fixed number of arguments after the options. The option wildcard on the `mount` command also makes it possible to specify the file-system being mounted from the Delphix Engine.

Example /etc/sudoers File Configuration on the Target Environment for sudo Privileges on the VDB Mount Directory Only (Linux OS)

```
Defaults:delphix_os !requiretty
delphix_os ALL=(root) NOPASSWD: \
/bin/mount  *        /oracle/*, \
/bin/umount *        /oracle/*, \
/bin/umount          /oracle/*, \
/bin/mkdir -p        /oracle/*, \
```

```
/bin/mkdir -p -m 755 /oracle/*, \
/bin/mkdir           /oracle/*, \
/bin/rmdir           /oracle/*, \
/bin/ps
```

Example /etc/sudoers File Configuration on the Source Environment to grant Super-User privileges when running PS

```
Defaults:delphix_os !requiretty
delphix_os ALL=(root) NOPASSWD: /bin/ps
```

Example 2

This example restricts the **delphix_os** user's use of `sudo` privileges to the directory / `oracle` , restricts the mount commands to a specific Delphix Engine hostname and IP, and does not allow user-specified options for the `umount` command.

This configuration is more secure, but there is a tradeoff with deployment simplicity. This approach would require a different sudo configuration for targets configured for different Delphix Engines.

A Second Example of Configuring the /etc/sudoers File on the Target Environment for Privileges on the VDB Mount Directory Only (Linux OS)

```
Defaults:delphix_os !requiretty
delphix_os ALL=(root) NOPASSWD: \
/bin/mount           <delphix-server-name>* /oracle/*, \
/bin/mount *         <delphix-server-name>* /oracle/*, \
/bin/mount           <delphix-server-ip>*   /oracle/*, \
/bin/mount *         <delphix-server-ip>*   /oracle/*, \
/bin/mount "", \
/bin/umount          /oracle/*, \
/bin/umount *        /oracle/*, \
/bin/mkdir, \
/bin/rmdir, \
/bin/ps
```

Managing Oracle environments and hosts

This section describes the attributes of Oracle-specific environments such as RAC Cluster Users, Listeners, and RAC configurations and covers the following topics:

- Managing environment users
- Environment attributes for hosts with Oracle
- Adding an Oracle single instance or RAC environment
- Adding Oracle standalone and RAC targets in Kerberos
- Adding a database installation home to an Oracle environment
- Adding a database to an Oracle environment
- Adding a listener to an Oracle environment
- Enabling linking and provisioning for Oracle databases
- Linking an Oracle RAC source database using a standalone host environment
- Listener and JDBC verification
- Providing your own Oracle Java
- Changing the hostname or IP address for Oracle source and target environments
- How to change ORACLE_HOME
- CLI support for partial-full backup in Oracle

Managing environment users

Procedure

1. Login to the **Delphix management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Click on the existing **environment name** you want to modify.
5. In the Details tab under **Basic information**, click the **Plus** icon next to Environment Users to add a user.
6. Enter the **Username** and **Password** for the OS user in that environment.
7. Select **Add** to save the new user.
8. To change the primary user for this environment, click the **Pencil** icon next to **Environment users**. Only the primary user will be used for environment discovery.
9. To delete a user, click the **Trash** icon next to their username.

Environment attributes for hosts with Oracle

This topic describes the attributes of Oracle-specific environments such as RAC Cluster Users, Listeners, and RAC configurations. You can manage these settings by the procedure described below.

1. Navigate to Manage > Environments.
2. In the Environments panel, click the name of an environment to view its attributes.
3. Next to Attributes, click the Pencil icon to edit an attribute.
    - For Oracle RAC environments select your node then click the Pencil icon.

Common environment attributes

Attribute	Description
Environment users	The users for that environment. These are the users who have permission to ssh into an environment or access the environment through the Delphix Connector. For more information on the environment user requirements, see the Requirements topics for specific data platforms.
Host address	The IP address of the environment host.
SSH Port	The SSH port number used for the environment.
Login Type	Select from: • Username and Password • Username and Public Key • Password Vault
Toolkit Path	The plugin path for the environment. Delphix uses the Toolkit path (Toolkit is a synonym for Plugin) for logs, mount points, and keeping lib files.
Notes	Any other information you want to add about the environment.

Oracle environment attributes

Attribute	Description
Environment Name (RAC)	The Environment Name field under Attributes is used to provide the name of the environment host in the case of cluster environments. This field defaults to the IP address of the host unless you specify another name.

Attribute	Description
Cluster Name (RAC)	The name of the RAC cluster as retrieved during auto-discovery from the Oracle Clusterware Repository (OCR).
Cluster Home (RAC)	This is the full pathname of the directory in which the cluster or grid ORACLE_HOME resides, which contains the executables for the Oracle Clusterware. Besides containing the software for the Oracle CRS (Cluster Ready Services), this directory also contains the binaries for Oracle ASM (Automatic Storage Management).
Version	The version of the RAC cluster as retrieved during auto-discovery from the Oracle Clusterware Repository (OCR).
Cluster User (RAC)	The OS user account with read-execute permission to access the ORACLE_HOME for the cluster/grid, as well as the ORACLE_HOME for the database.
Host Address (RAC)	Public IP hostname or IP address for the specific node (or host) within the RAC cluster.
NFS Address	In the event that the network link to which “public hostname” of the RAC nodes reside is to be used only for command-and-control traffic (i.e. SSH, JDBC, etc), and not for network-attached storage, then this field is where the comma-separated list of IP addresses for the alternate network dedicated to NFS for network-attached storage.
Virtual IP (RAC)	Oracle RAC requires setting up virtual IPs to manage failover. This field informs Delphix which IP addresses to use when a failure is detected. Click the + to add another virtual IP domain and IP address. For more details view Overview of Virtual IP Addresses
Listeners	The listener is used to connect incoming client requests to the database. See Adding a Listener to an Oracle Environment for more information.

Attribute	Description
Remote Listener	An Oracle TNS name resolves to an address or address list of Oracle Net remote listeners. This is the Oracle initialization parameter REMOTE_LISTENER, which must be the same on all RAC cluster instances. Click the + to add a remote listener.
SCAN	Single Client Access Name is used to allow clients to access cluster databases.
SCAN Listener	Listener used with SCAN to establish client connections to the database instances in the RAC cluster.

Adding an Oracle single instance or RAC environment

This topic describes how to add a new Oracle or Oracle RAC environment.

Prerequisites

- See the topics Requirements for Oracle Hosts and Databases
- There can be one Oracle unique database name (DB_UNIQUE_NAME) per Delphix Engine. For example, if you provision a VDB with a database unique name "ABC" and later try to add an environment that has a source database that also has a database unique name of "ABC", errors will occur.

Procedure

1. Login to the **Delphix management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Click the **Actions (...)** menu next to **Environments** and select **Add environment.**
5. In the **Host and Server** tab, select **Unix/Linux**.
6. Select **Standalone host** or **Oracle cluster**, depending on the type of environment you are adding.
7. Click **Next**.
8. For standalone Oracle environments: Enter the **Environment name** Enter **Host IP address**
9. For cluster Oracle RAC environments: Enter the **Environment name,** Enter **Cluster home, and** Enter **Node address**
10. For NFS Addresses (Optional): Enter one or more comma-separated **IP Address/Hostname**
    In the case of the Oracle RAC environment, ensure that the NFS Address list includes IP Addresses from all the cluster nodes. NFS address given at the time of environment creation is applied to all the discovered RAC nodes. If some nodes in the cluster need a different IP Address list, users can edit the specific host and update the NFS Address after the environment has been created.

> If specified, Delphix Engine only allows NFS requests (mount, etc) originated from IP Addresses specified for the host.

11. If server authentication for remote host communication or engine to host throughput tests is desired, make sure the appropriate config is set. For more details refer to Configuring Network Security Settings. You will need to create a JKS or PKCS#12 keystore on the remote host with the full CA chain of the DSP key in the keystore. By default, the key will just the signed by the Delphix CA, but you can replace the DSP key if you wish. Refer to KeyStore Settings for more details.
12. If client authentication for remote host communication or engine to host throughput tests is also desired, make sure the appropriate config is set. For more details refer to Configuring Network Security Settings You will need to create another JKS or PKCS#12 keystore on the remote host with the desired key pair. Make sure the created keystore has permissions such that it is readable by all environment users. Then, add the full CA chain of the remote host's key pair to the TrustStore on the engine. For more details, refer to TrustStore Settings
13. Enter the SSH port. The default value is 22.
14. Select a Login Type:
    - Username and Password - enter the OS username and password, or
    - Username and Public Key - enter the OS username, or
    - Password Vault - select from an existing Enterprise Password Vault

> **Using Public Key Authentication**

If you want to use public-key authentication for logging into your Unix-based environment, there are two options: use the engine's key pair or provide a key pair for this environment.

To use the engine's key pair:

- Select **Public key** for the **Login type.**
- Click **View public key**.
- Copy the public key that is displayed, and append it to the end of your `~/.ssh/authorized_keys` file. If this file does not exist, you will need to create it.
    - Run chmod 600 ~/.ssh/authorized_keys to allow only the file's owner to read and write to it (make sure the file is owned by the user).
    - Run chmod 755 ~ to restrict access to the user's home directory so no other user may write to it.
    - Run chmod 700 ~/.ssh so that others cannot write to it. The ~/.ssh directory cannot be writable by group or other users. Otherwise, authentication will fail.

As an alternative, you can provide a key pair specific for this environment via the API or CLI. See Option 2 in this CLI Cookbook article for instructions.

15. For **Password login**, click **Verify credentials** to test the username and password.
16. Enter **Toolkit path**. The toolkit directory stores scripts used for Delphix Engine operations, and should have a persistent working directory rather than a temporary one. The toolkit directory will have a separate subdirectory for each database instance and must have 0770 permissions. At least 1.6 GB of storage is needed at the time of setting up the environment and at least 500MB of free space is required to allow refreshes and maintenance of the toolkit, especially during upgrades.
17. To provide your own Oracle Java select the **Provide my own JDK** checkbox and click **Next**.
18. In the Java Development Kit, tab enter the absolute path to your Oracle JDK and click **Next**. For more information, see OpenJDK on Delphix Engine. Adding an Oracle single instance or RAC environment Procedure
19. Click **Submit**.

Post-requisites

After you create the environment, you can view information about it:

1. Click **Manage**.
2. Select **Environments**.
3. Select the environment name.

Adding Oracle standalone and RAC targets in Kerberos

This topic describes how to add an Oracle standalone and RAC targets in Kerberos.

Prerequisites

- Configure Kerberos credentials through the System Setup interface.
- Kerberized Oracle host operations are only supported in Delphix Engine release 5.2.3.0 and later.
- You must define only one principle in the Kerberos configuration and this principle must be used for all Kerberized host environments.

Procedure

When adding a new environment or editing the configuration of an existing environment, it is a general Kerberos authentication requirement that you configure the host address using a fully qualified domain name (FQDN). The login type of Kerberos authentication is available after you apply the Kerberos configuration in the System Setup interface.

In the example below, the Kerberos principal krbuser has previously been configured using System Setup, so this is automatically populated when the radio button is selected.

Perform the following steps to add an Oracle standalone and RAC targets in Kerberos.

1. Login to the Delphix Management application.
2. Click **Manage**.
3. Select **Environments**.
4. Click the **Actions (...)** menu next to **Environments** and select **Add environment**.
5. In the **Host and server** tab, select **Unix/Linux** and click **Next**.
6. In the **Environment settings** tab, do the following:
    a. Enter an environment name and the host address.
    b. Under **Login Type**, select the **Kerberos authentication**radio button.

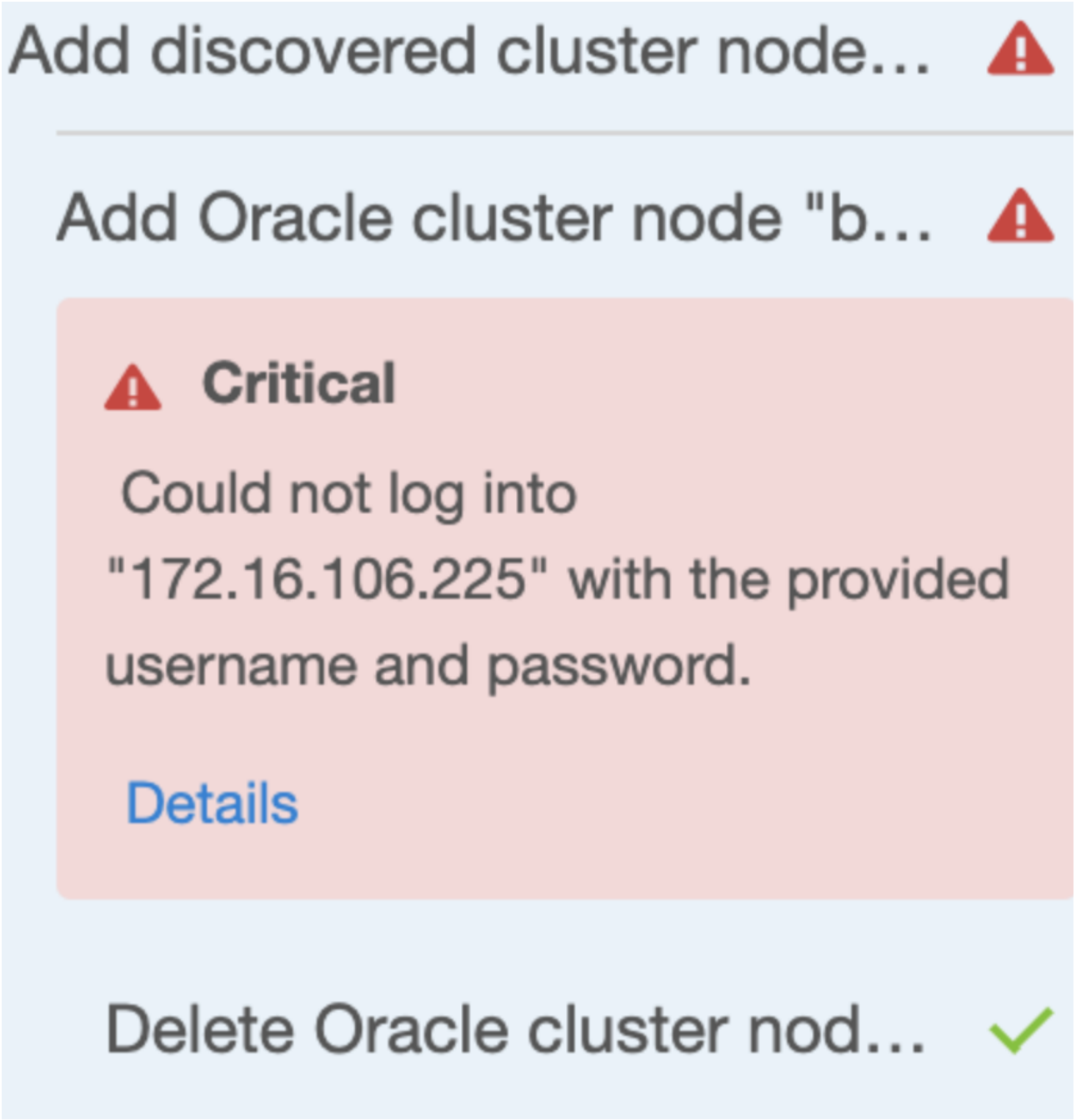
> The Kerberos principal **krbuser** has previously been configured using the System Setup interface, so this is automatically populated when the **Kerberos authentication** radio button is selected.

c. Click Next.

d. The **Summary** tab enables you to review your configurations. Review the configuration and click **Submit**.

Configuring Oracle RAC using the GUI

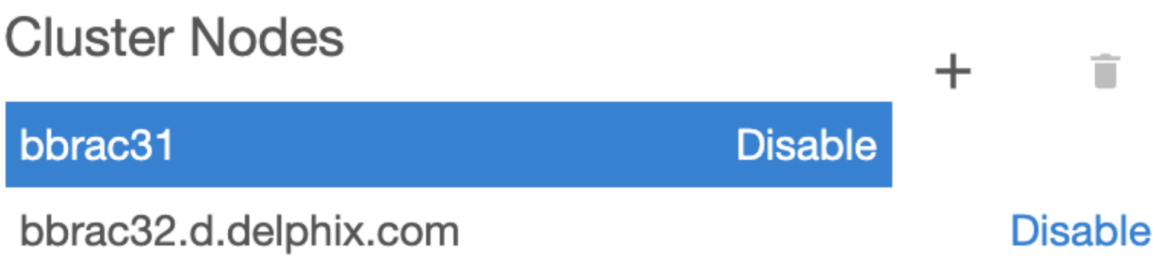
As the discovery logic attempts to add other nodes via IP address (based on cluster interrogation results which return the additional node IP addresses), automatic discovery of additional nodes will initially fail, causing Kerberos authentication to fail with "Could not log into <IP_address> with the provided username and password". It is therefore also expected to see one or more failed actions under the Discovery job indicating "Add discovered cluster node" and "Add Oracle cluster node <node name>" were unsuccessful.



As such, all nodes beyond the first must be added manually. The cluster can be added initially using the GUI or CLI as desired to discover the first node, configured via FQDN. Each additional node is added manually via CLI or GUI.

To add all the additional nodes via GUI, perform the following steps:

1. Login to the Delphix Management application.
2. Click **Manage**.
3. Select **Environments**.
4. Select the cluster node to view the basic information of the respective node.
5. In the **Cluster node** widget, click the plus icon to add a node. In this example, **bbrac31** was discovered normally, and **bbrac32.d.delphix.com** was manually added after the Environment Discovery process was completed.



6. Click the new node in the Cluster Nodes list.
7. Click the plus icon next to the **Virtual IPs** and add the virtual IP name and address for the node.
8. Click the plus icon next to the **Listeners** and add the name and protocol address of a listener on the node. Protocol address should be in the following format. This example uses bbrac15-vip.delphix.com as the host virtual IP address.

```
(ADDRESS=(PROTOCOL=tcp)(HOST=bbrac15-vip.delphix.com)(PORT=1521))
```

Adding an Oracle RAC cluster using the CLI

You can use the CLI to create the cluster, specifying one of the nodes for discovery. Make sure to set the credential type to KerberosCredential. This procedure uses the following node names bbrac14, bbrac15, and bbrac16 for example purposes. "bbrac14" will be the primary node used for discovery. Make sure that you use an FQDN for the host address (it is a must Kerberos requirement for authentication):

```
delphix.engine> /environment create
delphix.engine environment create *> set type=OracleClusterCreateParameters
delphix.engine environment create *> set cluster.name=bbrac1416
delphix.engine environment create *> set cluster.crsClusterHome=/u01/app/11.2.0/grid
delphix.engine environment create *> set
primaryUser.credential.type=KerberosCredential
delphix.engine environment create *> edit nodes
delphix.engine environment create nodes *> add
delphix.engine environment create nodes 0 *> set name=bbrac14.delphix.com
delphix.engine environment create nodes 0 *> set
hostParameters.host.address=bbrac14.delphix.com
delphix.engine environment create nodes 0 *> set hostParameters.host.toolkitPath=/
work
delphix.engine environment create nodes 0 *> commit
```

Now, use the above procedure described in the **Configuring Oracle RAC using the GUI** section to manually add the node.

Adding a database installation home to an Oracle environment

When you add an environment with the Delphix Admin application, all database installation homes on it are automatically discovered. However, if a database installation home is not automatically discovered, you can add it manually to the environment.

Procedure

1. Login to the **Delphix management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Select an Environment.
5. Click the **Databases** tab**.**
6. Click the **Add dataset home** button.
7. Enter the **Installation home**.
8. Enter the **Version** of the Installation Home.
9. Enter the **Oracle base** of the Installation Home.
10. Enter the **Bits** of the Oracle Home.
11. When finished, click **Add**.

Troubleshooting

If the environment user has oinstall permissions, Delphix will be able to discover the **Version**, **Oracle Base**, and **Bits**. If any of these fields are found to be different than those provided by the user, a fault will be raised on the repository.

Oracle version

The version can be found in the comps.xml file in $ORACLE_HOME/inventory/ContentsXML/comps.xml. Example of Oracle Home with version 12.1.0.2:

```
<COMP NAME="oracle.server" VER="12.1.0.2.0" BUILD_NUMBER="0" REP_VER="0.0.0.0.0"
 RELEASE="Production" INV_LOC="Components/oracle.server/12.1.0.2.0/1/" LANGS="en"
 XML_INV_LOC="Components21/oracle.server/12.1.0.2.0/" ACT_INST_VER="12.1.0.2.0"
 DEINST_VER="11.2.0.0.0" INSTALL_TIME="2016.Apr.14 12:42:09 PDT" INST_LOC="/u01/app/
oracle/product/12.1.0/dbhome_1/oracle.server">
```

Oracle base

This can be found as a property in $ORACLE_HOME/inventory/ContentsXML/oraclehomeproperties.xml. Example of Oracle Home with Oracle Base "/u03/app/ora11202":

```
<PROPERTY NAME="ORACLE_BASE" VAL="/u03/app/ora11202"/>
```

Bits

This can be found by running

```
file $ORACLE_HOME/bin/rman
```

The output will indicate if the Oracle Home is 32 bit or 64 bit.

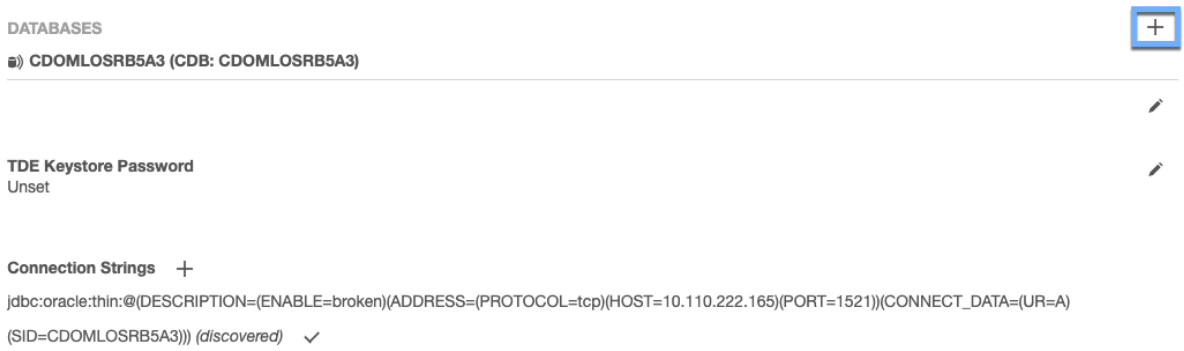
Adding a database to an Oracle environment

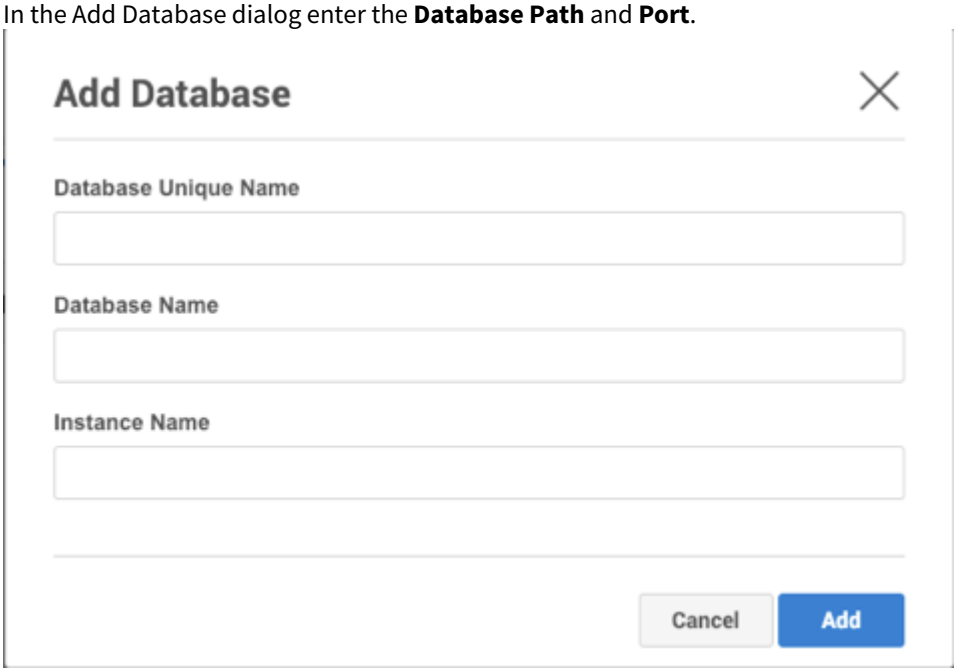
Prerequisites

- Make sure your source database meets the requirements described in Requirements for Oracle Hosts and Databases
- Before adding a database, the installation home of the database must exist in the environment. If the installation home does not exist in the environment, follow the steps in Adding a Database Installation Home to an Oracle Environment

Procedure

1. Login to the **Delphix management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Select an Environment.
5. Click the **Databases** tab.
6. Choose the installation home where the database is installed.
7. Click the **Plus** icon.



8. In the Add Database dialog enter the **Database Path** and **Port**.
9. 

When finished, click **Add**.

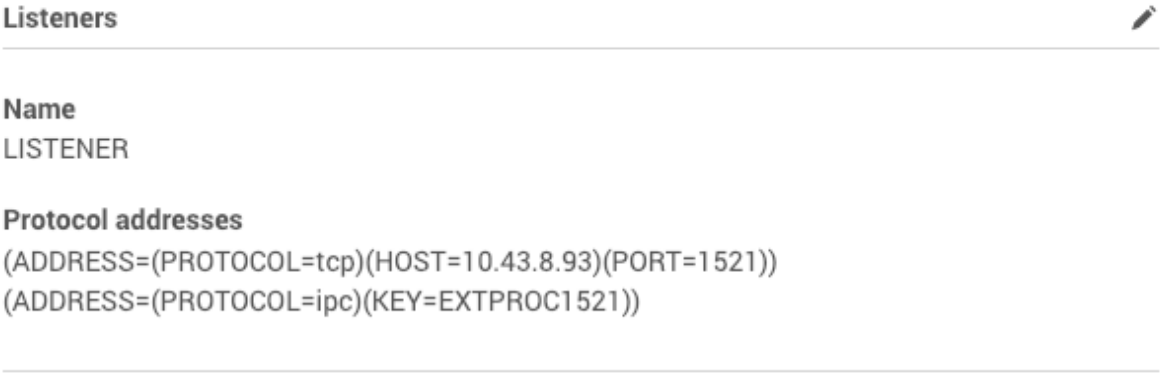
Adding a listener to an Oracle environment

This topic describes how to add listeners for an Oracle environment.

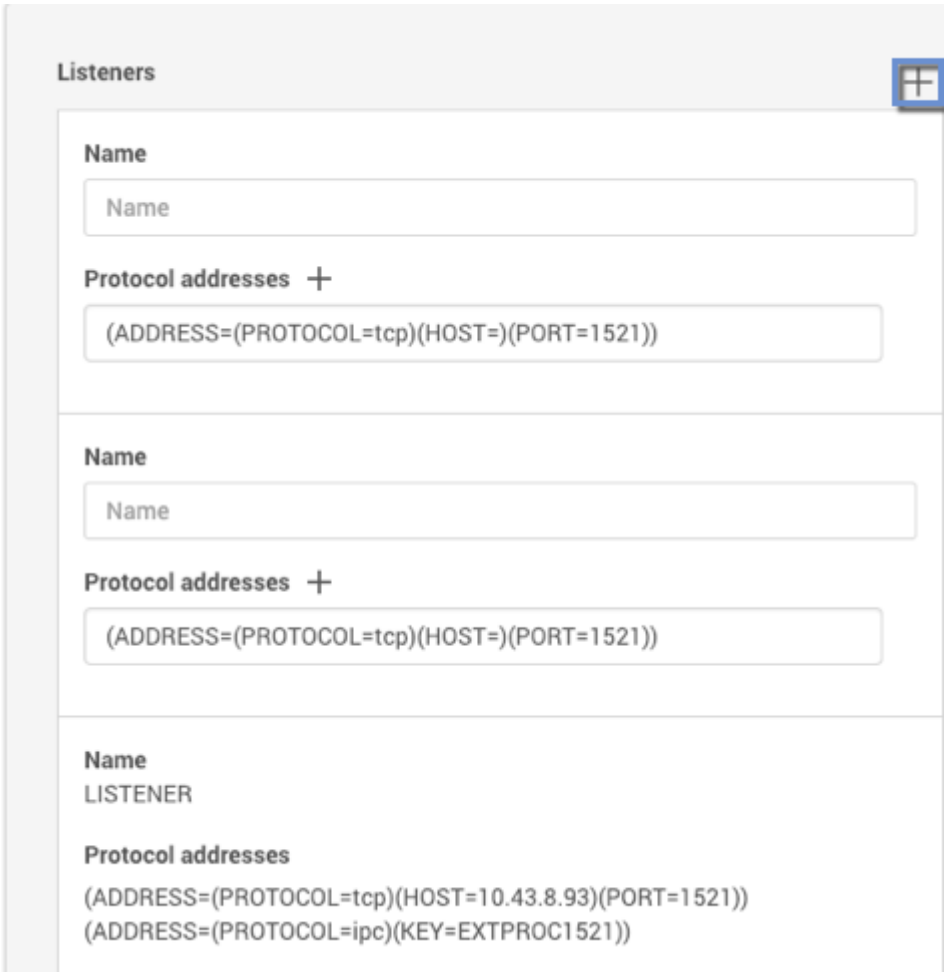
When you add an environment with the Delphix Management application, all listeners that are running on it are automatically discovered. However, if a listener is not automatically discovered, you can add it manually to the environment.

Procedure

1. Log into the **Delphix management** application.
2. Select **Manage > Environments**.
3. Click on the name of an environment to view its basic information.
4. In the Details tab next to Listeners, click the **Pencil** icon to edit the list of listeners.



5. In the Listeners panel click the **Plus** icon.



6. Enter a **Name** for the new listener, and a **Protocol address** of the listener. The Delphix Engine currently supports TCP and IPC protocol addresses. An example TCP protocol address is (ADDRESS=(PROTOCOL=TCP)(HOST=10.43.17.92)(PORT=1521)) and an example IPC protocol address is (ADDRESS=(PROTOCOL=IPC)(KEY=DELPHIX))
7. Click the **Plus** icon next to **Protocol addresses** to enter additional protocol addresses.
8. Click the **Check** icon to save your changes.

Enabling linking and provisioning for Oracle databases

This topic describes how to enable and disable provisioning, and linking for databases on an environment.

Before you can use a database as a dSource, you must first make sure that you have defined an environment and that linking is allowed on it in the Delphix Environment. Similarly, before you can provision a virtual database (VDB) to a target database, you must make sure that you have allowed provisioning to the host or cluster to which the Delphix Environment is attached.

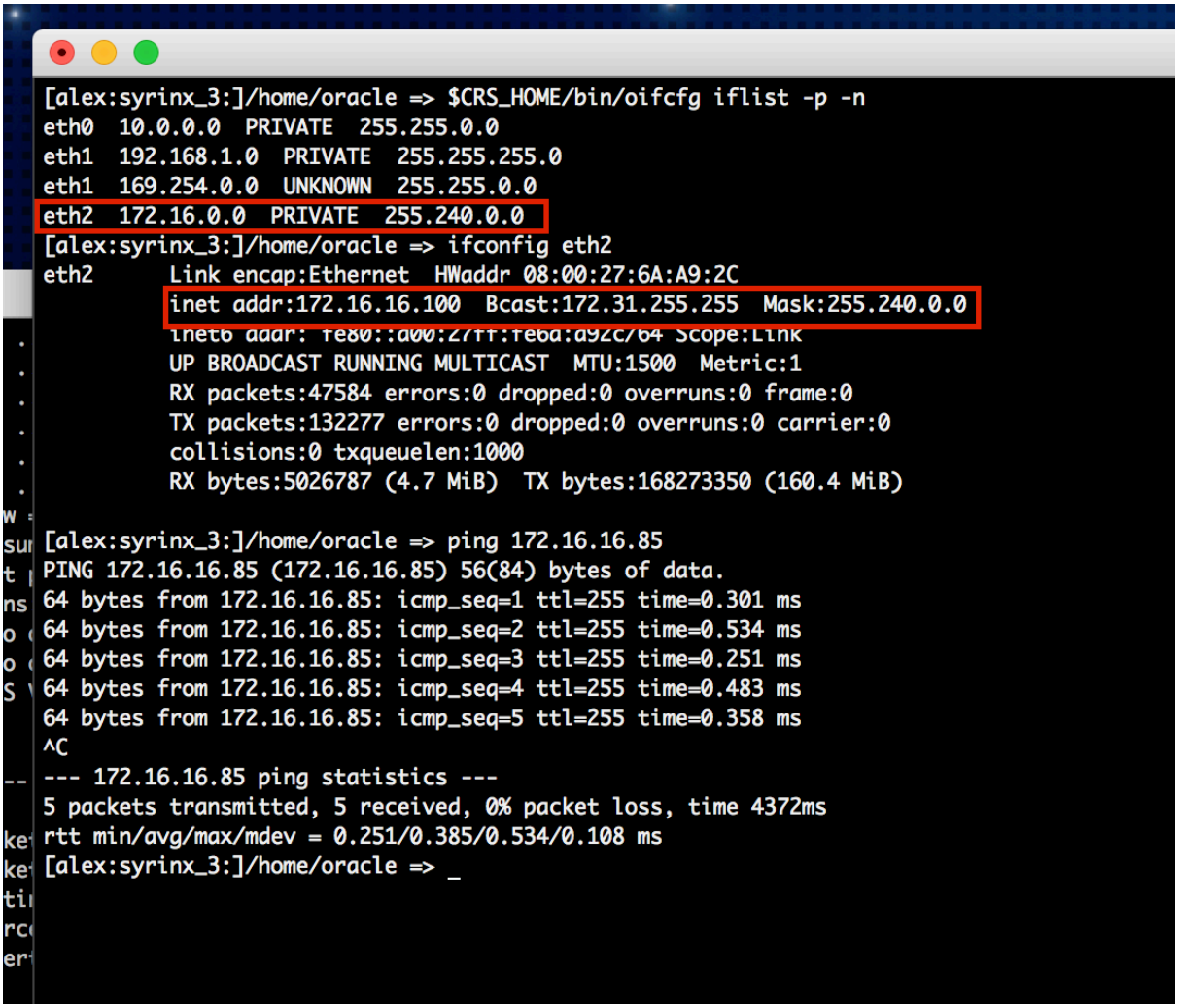
Procedure

1. Login to the **Delphix management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Click **Databases**.
    a. **For provisioning**: On the installation, you wish to work on select Edit for the **Installation details** and toggle the **Allow provisioning** checkbox and click save.
    b. **For linking**: Under in the database section of the installation which contains the DB you wish to link click edit for **Allow linking** and toggle the checkbox then click save.

Linking an Oracle RAC source database using a standalone host environment

The standard way to link an Oracle RAC database as a dSource in the Delphix workflow is to provide the name or IP of one of the cluster nodes and then discovery will find the other cluster members, listeners, and databases. This is done by adding an 'Oracle Cluster' type Environment in the Add Environment wizard. The cluster nodes will be added to the Environment using the hostname/IP returned by the `'olsnodes'` command and in most cases, this will be sufficient. There may, however, be cases when this is undesirable, such as wanting to run snapsync/logsync traffic over a second network. Using an 'Oracle Cluster' type Environment currently stores a single IP for each cluster node and this IP will be the one associated with the node name returned by the `'olsnodes'` command. Using a Standalone Environment is one way to work around this situation by defining the Environment based on the IP on the network snapsync/logsync should run over. The Delphix Engine should have a NIC connected to this network also. When both Delphix Engine and source servers have NICs on this second network there is no need for static routes to be configured on the Delphix Engine or the source server.

1. Ensure that OS and DB users for Delphix have been configured and meet requirements. Create the OS user on each node using the exact same settings. Always use the hostchecker to verify the source and target environments are ready for use with Delphix. Even though a single node will be used for linking, it may be desirable to unlink and relink the source to a different node in the future, so make sure the Delphix OS account is configured identically on all nodes.
2. Create a toolkit directory in the same location on each node and ensure permissions are set to 0770.
3. Identify the local IP address on the second network of the node which will be used for linking. Use the `'oifcfg'` and `'ifconfig'` commands to find this information. Then verify by pinging the Delphix Engine using it's IP on the second network. Don't worry about the output from `'oifcfg'` with respect to `PUBLIC,PRIVATE or UNKNOWN` values for the interfaces. These are not actual CRS usage, but rather based on the IP address range (RFC1918) - in the image they all show as `PRIVATE`.



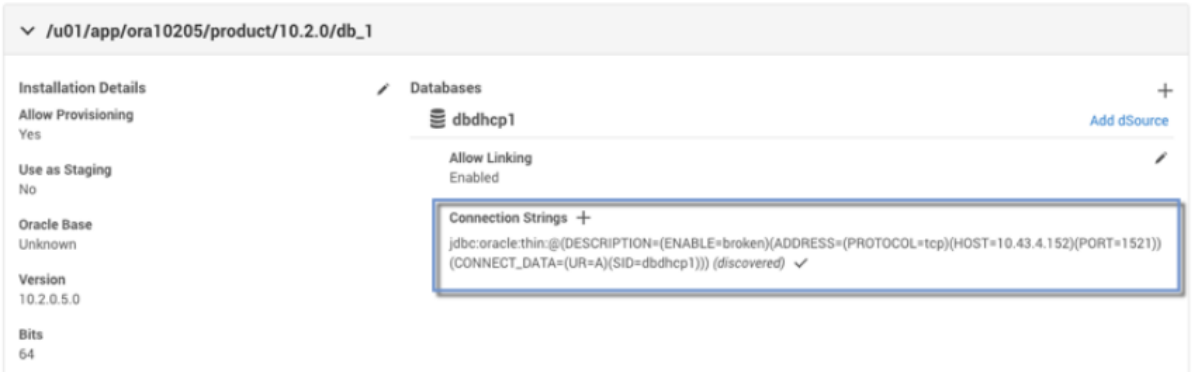
4. Add a 'Standalone Host' type environment using the chosen cluster node's second network IP.
5. When discovery is finished, the Oracle homes on the cluster node should be discovered along with the listeners that run on that node. Once that is done add the database that will eventually be a dSource to by providing the db_unique_name, db_name, and the instance name that runs on the node that was added.
6. Then add a jdbc connect string using the node VIP and a service name.
7. Now click on the add dSource link. Follow the steps in Adding a dSource making sure to set the Snapshot controlfile location in RMAN to a location that's visible to all cluster nodes, or snapsync will fail. An ASM diskgroup is a suitable location. Even if the Snapshot controlfile is not located in a shared location, it is still possible to link the source - the first snapsync will fail with a fault that relates to the Snapshot controlfile location, but subsequent Snapsync operations should succeed.
   JDBC connections will go over the public network and snapsync / logsync / environment monitor traffic will go over the second network.

Listener and JDBC verification

As with any database, a VDB needs to have a listener in order for external connections to be made. The Delphix Engine also uses a JDBC connection string in order to connect to a source or target database. If the engine does not have the proper listener or JDBC connect string defined, then connection errors can result.

Verifying the listener configuration

1. Verify that a listener is running on the source or target environment that you are investigating.
2. Login to the Delphix Management application.
3. Click **Manage**.
4. Select **Environments**.
5. Click on the **environment in** which you are troubleshooting.
6. On the **Configuration** tab, locate the **Listeners** section. Verify that the listener for the source or target system is listed there.
7. If a listener is not listed, it may be due to insufficient privileges on the part of the environment user that the engine is using. Verify that the proper sudo permissions have been granted for the user, or adjust them as necessary.

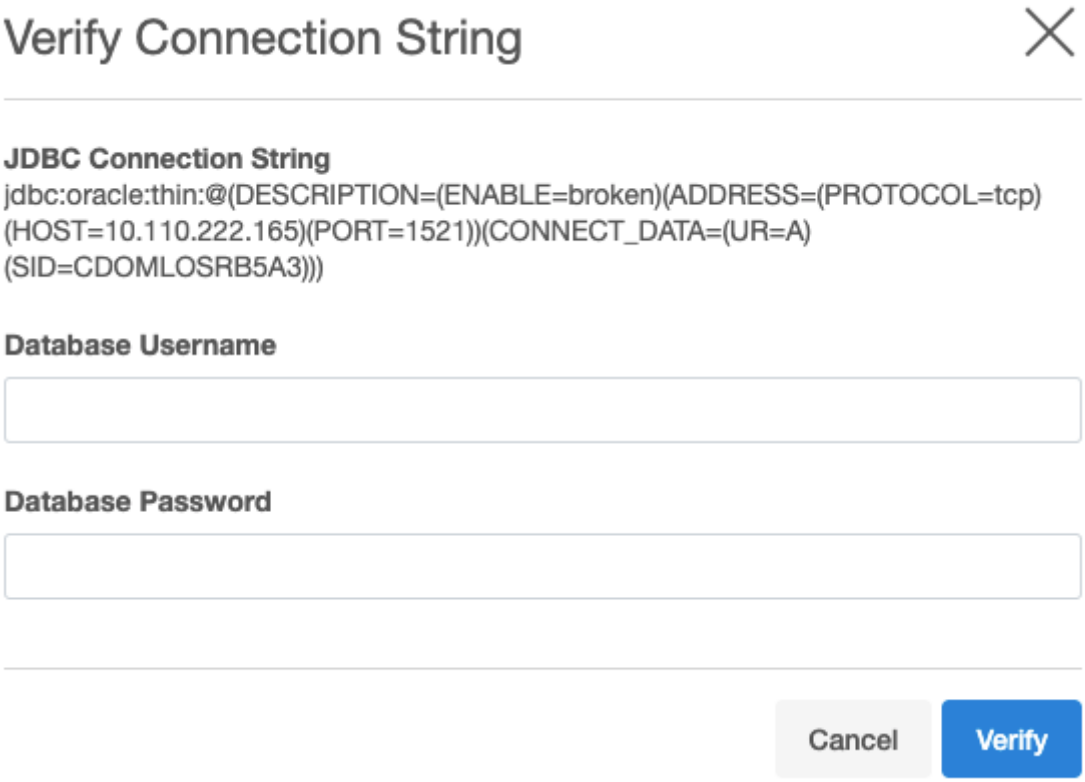


8. If you need to add another listener, you can do so manually.
    a. Click the plus (+) sign.
    b. Enter the appropriate values using the above image as a reference.

Verifying the JDBC connection string

Each source database and VDB has a connection string defined. If any parameters have changed, you may need to adjust these connection strings.

1. Verify that a listener is running on the source or target environment you are investigating.
2. Login to the Delphix Management application.
3. Click **Manage**.
4. Select **Environments**.
5. Click the environment in which you are troubleshooting.
6. Select the **Databases** tab.
7. The source database or VDB which you are investigating will display the JDBC connection string being used for the given database.
8. To verify that the connection string works, click the **checkmark** to the right of the connection string. You will then see the username and password text boxes.
9. Enter the Oracle username and password used by the Delphix Engine.



If no errors are encountered and the username/password dialogs simply go away, then a successful connection was made to the database.
If errors are encountered, you must investigate them on a case-by-case basis, just as you would do with any connection errors to an Oracle database.

Adding JDBC connection string

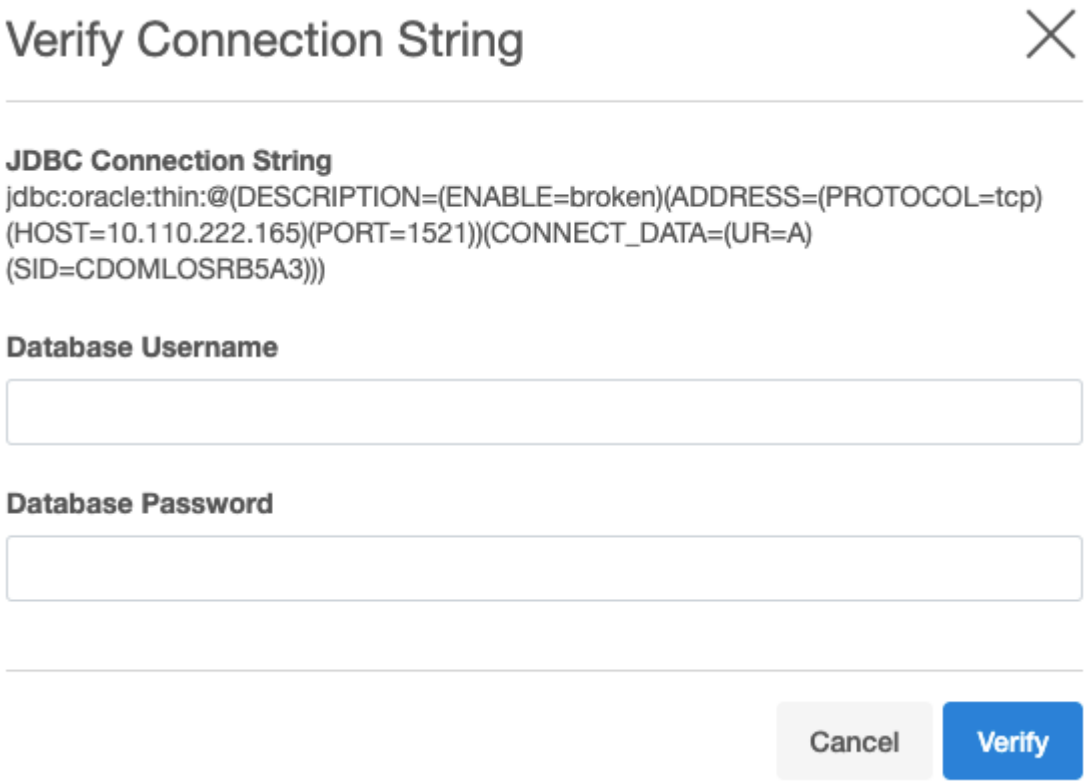
If you need to define an additional JDBC connection string for any reason, take the following steps:

1. Follow the same steps as listed in Verifying the JDBC Connection String to reach the JDBC connection string definition(s).
2. Click the plus (+) sign to the right of **Connection strings** to define an additional connection string. 
   Add Connection String Icon
   In the screenshot below, the user is adding a connection string for the “test2” service on port 1530 instead of the default 1521.
   
3. Click the **Add** to save the changes.
4. Follow the remaining steps in Verifying the JDBC Connection String to validate your newly added connection string.

Verifying the JDBC connection string

Each source database and VDB has a connection string defined. If any parameters have changed, you may need to adjust these connection strings.

1. Verify that a listener is running on the source or target environment you are investigating.
2. Login to the Delphix Management application.
3. Click **Manage**.
4. Select **Environments**.
5. Click the environment in which you are troubleshooting.
6. Select the **Databases** tab.
7. The source database or VDB which you are investigating will display the JDBC connection string being used for the given database.
8. To verify that the connection string works, click the **checkmark** to the right of the connection string. You will then see the username and password text boxes.
9. Enter the Oracle **username** and **password** used by the Delphix Engine.



Database Username and Password

If no errors are encountered and the username/password dialogs simply go away, then a successful connection was made to the database.
If errors are encountered, you must investigate them on a case-by-case basis, just as you would do with any connection errors to an Oracle database.

Verifying the listener configuration

1. Verify that a listener is running on the source or target environment that you are investigating.
2. Login to the Delphix Management application.
3. Click **Manage**.
4. Select **Environments**.
5. Click on the **environment in** which you are troubleshooting.
6. On the **Configuration** tab, locate the **Listeners** section. Verify that the listener for the source or target system is listed there.
7. If a listener is not listed, it may be due to insufficient privileges on the part of the environment user that the engine is using. Verify that the proper sudo permissions have been granted for the user, or adjust them as necessary.
8. If you need to add another listener, you can do so manually.
    a. Click the plus (+) sign.
    b. Enter the appropriate values using the above image as a reference.

Providing your own Oracle Java

Delphix supports the ability for users to provide their own Oracle JDK for each host. This feature is intended for customers who have a Java license with Oracle and want to use a more recent version than the one that comes with Delphix. Note that the JDK is provided per node or host, not per cluster or environment. Initially, when creating a cluster all hosts must have Java at the same absolute path. However, once the cluster has been discovered in Delphix each host's path can be adjusted.

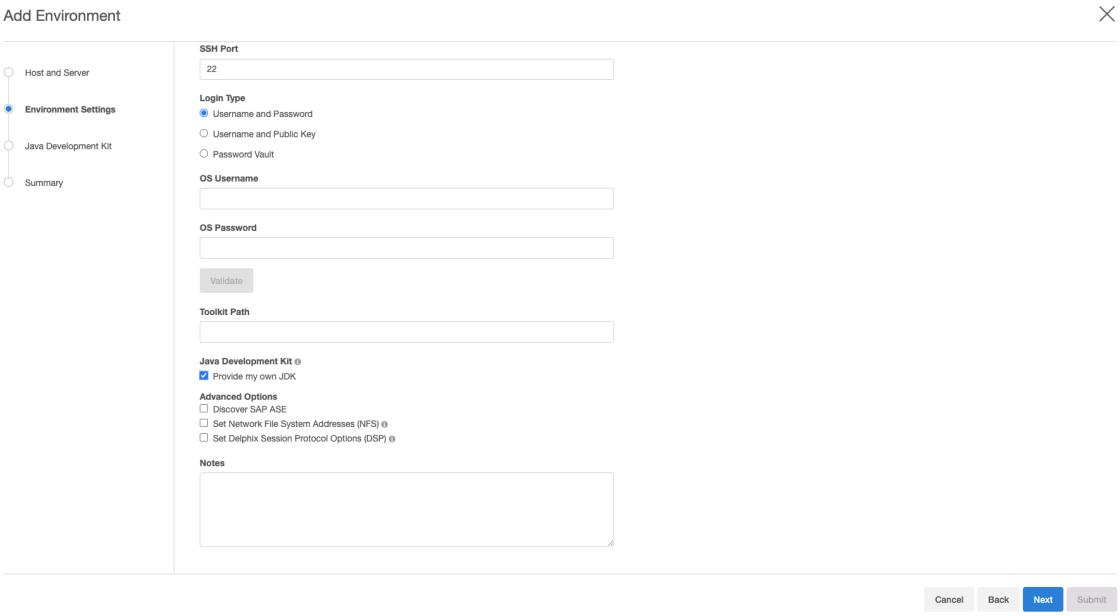
For details about which versions of Oracle Java are supported see the Java Support Policy section below.

> ⚠ All Delphix environment users on the host require read and execute permissions on the provided JDK, its subfolders, and files.

Adding an Oracle JDK

By default, Delphix comes with the OpenJDK. To modify the JDK, follow the below steps:

1. log in to the **Delphix management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Click the **Actions (...)** menu next to **Environments and** select **Add environment.**
5. In the **Environment settings** tab, select the **Provide my own JDK** checkbox, and click **Next**.
   This action will remove the previous built-in JDK and will initiate an Environment refresh operation after the path is changed.



> ⚠ Select the Reset to Default to reset your JDK to the OpenSource JDK (default).

6. In the **Java development kit** tab, the currently selected JDK kit will be shown (default is OpenJDK). Provide an absolute path to the JDK root and click **Next.**
7. Verify the configured JDK and environment settings and click **Submit.**

Updating a JDK path

For existing environments users can add or update JDK paths by following the steps below:

1. Log in to the **Delphix management** application.
2. Select **Manage > Environments**.
3. Click on the name of an environment to view its basic information.
4. In the Details tab next to the **Java development kit**, click the **Pencil** icon.

5. In the **Java development kit dialog**, provide an absolute path to the JDK root and click **Next.**

Do not place the JDK inside the Delphix Toolkit.

1. Click the

Check icon to save your changes.

Java support policy

For each release of the Delphix engine, we will support Java version(s) depending on the available Java builds provided by Oracle in that time period. The following statements apply for Java support for specific Delphix versions:

1. In accordance with our security policy, every six months, we will certify the latest update release of the currently supported LTS Oracle Java.
2. We will support at least one commercially supported LTS release per Delphix version for 1 year, at which point we will update the minimum supported version.
3. Introducing support for new LTS versions of Java will be considered and evaluated with each major version of Delphix.
4. At least 1 year before Oracle ends support for an LTS version of Java, Delphix will also end support for that version and update it to a newer LTS version.

Supported operating systems

This feature is supported with use for Linux, Solaris, and Windows environments, as described in the support matrix. AIX and HP-UX provide us with a specific Java version to run on those hosts.

Only Java versions equal to or greater than U242 and equal to or lesser than U281 are tested and certified by Delphix.

	RHEL	SLES	Solaris	AIX	HP-UX	Windows
Oracle Java 8 u242	Supported	Supported	Supported	N/A	N/A	Supported
Oracle Java 8 u281	Supported	Supported	Supported	N/A	N/A	Supported
Oracle Java 8 u333	Supported	Supported	Supported	N/A	N/A	Supported

	RHEL	SLES	Solaris	AIX	HP-UX	Windows
Oracle Java 8 u351*	Supported	Supported	Supported	N/A	N/A	Supported

- New in 8.0.0.0

Delphix toolkit native Java support matrix

This matrix describes the supported versions of Java we package with the Delphix toolkit for each operating system. This is the default option if you do not use the feature to provide your own Java for each host.

	RHEL	SLES	Solaris x64	Solaris sparc9	AIX	HP-UX	Windows
AdoptOpenJDK 8u332-b09	Supported	Supported	Supported	N/A	N/A	N/A	Supported
AdoptOpenJDK 8u322-b06	N/A	N/A	N/A	Supported	N/A	N/A	N/A
IBM Java 8.0.0.620	N/A	N/A	N/A	N/A	Supported	N/A	N/A
HP Java 8.0.21	N/A	N/A	N/A	N/A	N/A	Supported	N/A

Changing the hostname or IP address for Oracle source and target environments

Procedure

For source environments

1. Disable the dSource as described in Enabling and Disabling Oracle dSources
2. If the **Host Address** field contains an IP address, edit the IP address.
3. If the **Host Address** field contains a hostname, update your Domain Name Server to associate the new IP address to the hostname. The Delphix Engine will automatically detect the change within a few minutes.
4. In the **Environments** screen of the Delphix Engine, refresh the host.
5. Enable the dSource.

For VDB target environments

1. Disable the VDB as described in Enabling and Disabling Oracle VDBs
2. If the **Host address** field contains an IP address, edit the IP address.
3. If the **Host address** field contains a hostname, update your Domain Name Server to associate the new IP address to the hostname. The Delphix Engine will automatically detect the change within a few minutes.
4. In the **Environments** screen of the Delphix Engine, refresh the host.
5. Enable the VDB.

For the Delphix engine

1. To stop running your VDB select the red **Stop** button located on the VDB **Configuration** tab.
2. Disable all dSources as described in Enabling and Disabling Oracle dSources
3. You can use either the command-line interface or the Delphix Setup application to change the IP address of the Delphix Engine.
    a. To use the command-line interface, press **F2** and follow the instructions described in Setting Up Network Access to the Delphix Engine
    b. To use the Delphix Setup application, go to **Delphix management > Engine setup** in the Delphix Management application, or click **Server setup** in the Delphix Engine login screen.
        i. In the **Network** panel, click **Modify**.
        ii. Under **DNS services**, enter the new IP address.
        iii. Click **OK**.
4. Refresh all Environments by clicking the **Refresh** symbol on the **Environments** screen.
5. Enable all dSources as described in Enabling and Disabling Oracle dSources
6. To start all VDBs, click the **Start** button located on the VDB **Configuration** tab.

> **Using custom init.ora or spfile.ora files**
> If you are are using custom `init.ora` or `sfpfile.ora` files with your Oracle VDBs, you should use the Oracle command-line interface (`sqlplus/srvctl`) to shut down any active VDBs and copy the parameter files to a backup location. Complete the steps above, then replace the files and re-start the VDB from the Oracle command line to restore your custom settings. See Customizing VDB File Mappings for more information about customizing init.or and other configuration files.

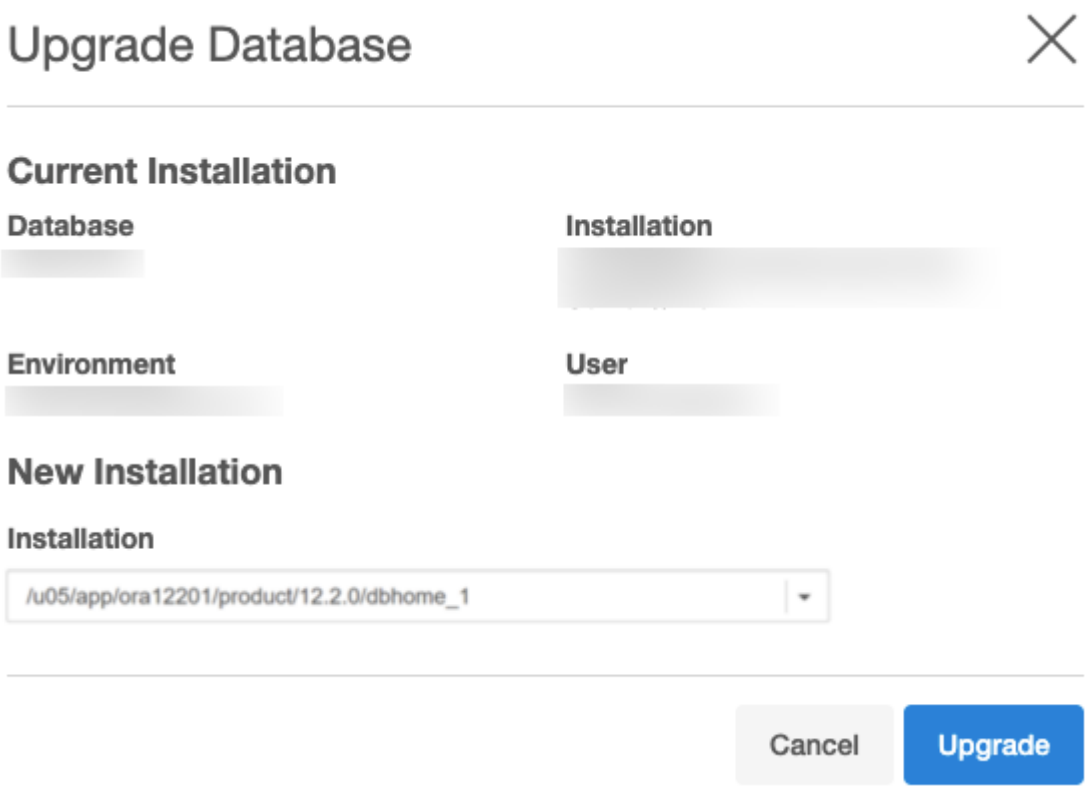
How to change ORACLE_HOME

Prerequisites

- know the new ORACLE_HOME user and password
- know the VDB you want to change
- have delphix_admin privileges to the VDB

Procedure

1. Login to the **Delphix management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select your VDB.
5. Select the **Configuration tab.**
6. In the upper right-hand corner of the **Source** sub-tab, click the **Actions menu (...)** and select **Upgrade**.
7. Select the new **Installation** from the drop-down menu.



8. Click **Upgrade**.

CLI support for partial-full backup in Oracle

Overview

Delphix Engine version 6.0.8.0 introduces an option to perform partial full snapsync via the CLI. A full backup of specified datafiles will be performed, along with an incremental backup of the remaining files.

Prerequisites

1. Requested datafiles must exist in the database for which SnapSync is being run.
2. In case of an MT, Snapsync must be performance for the PDB, the datafiles can belong to either the PDB or its CDB.
3. Requested datafiles must be online.
4. Partial full SnapSync can not be requested for an initial SnapSync.
5. Partial full SnapSync can not be run if there is a pending SnapSync.

Command

```
ip-00-000-000-000 database> select CDOMLOSR1CEBPDB3

ip-00-000-000-000 database 'CDOMLOSR1CEBPDB3'> sync

ip-00-000-000-000 database 'CDOMLOSR1CEBPDB3' sync *> set filesForFullBackup=1,2

ip-00-000-000-000 database 'CDOMLOSR1CEBPDB3' sync *> commit
```

Linking data sources and syncing data with Oracle

Linking a data source will create a dSource object on the engine and allow Delphix to ingest data from this source. The dSource is an object that the Continuous Data Engine uses to create and update virtual copies of your database. As a virtualized representation of your source data, it cannot be managed, manipulated, or examined by database tools.

Once a dSource has been linked, you can begin to perform dSource operations on it such as enable, detach, delete, and more.

This section covers the following topics:

Linking data sources with Oracle

Linking a dSource will ingest data from the source and create a dSource object on the engine. The dSource is an object that the Continuous Data Engine uses to create and update virtual copies of your database. As a virtualized representation of your source data, it cannot be managed, manipulated, or examined by database tools.

Once a dSource has been linked, you can begin to perform dSource operations on it such as enable, detach, delete, and more.

For an overview of all dSource related actions, please visit Provisioning and Managing Virtual Databases

1. Login to the **Delphix management** application.
2. Navigate to **Manage > Datasets.**
3. Click the plus icon and select **Add dSource**.
4. In the **Add dSource** wizard, select the source database with the correct environment user-specified.
5. Enter your login credentials for the source database and click **Verify credentials**. If you are linking a mounted standby, Click **Next**. See the topics under Linking Oracle Physical Standby Databases for more information about how the Delphix Engine uses non-SYS login credentials.
6. Enter a name and select a group for your dSource. Adding a dSource to a dataset group lets you set Delphix Domain user permissions for that database and its objects, such as snapshots. See the topics under Users and Groups for more information.
7. Select the **Data management** settings needed. For more information, visit Data Management Settings for Oracle Data Sources
8. Assign existing policies to the new dSource. New policies can be created and associated later.
9. Enter any scripts that should be run using the **Hooks** page.
10. Review the dSource Configuration and Data Management information, and then click **Submit**.

Once the action to add a dSource has been submitted, the Delphix Engine will initiate a DB_Link job to create the dSource. If the *Load Immediately* option was selected in the data management page a DB_Sync job will also be executed to ingest data from the source, otherwise, this first DB_Sync job will run as per the associated SnapSync policy.

When the jobs have successfully completed, the database icon will change to a dSource icon on the Environments > Databases screen, and the dSource will be added to the list of Datasets under its assigned group.

Linking an Oracle pluggable database

This topic describes how to link an Oracle 12c pluggable database to the Delphix Engine to create a dSource.

Prerequisites

- Make sure the Delphix Engine has already discovered the multitenant container database and its pluggable databases. If the container database does not exist in the environment, follow the steps in Adding a Database to an Oracle Environment. If the pluggable database you want to link does not exist in the environment.
- You should have Block Change Tracking (BCT) enabled for the container database, as described in Requirements for Oracle Hosts and Databases
- The container database should be in ARCHIVELOG mode and the NOLOGGING option should be disabled, as described in Requirements for Oracle Hosts and Databases

Procedure

1. Log into the **Delphix management** application.
2. Select **Manage > Datasets.**
3. Click **the Plus** icon and select **Add dSource.**
4. In the **Add dSource** wizard, select the source pluggable database. If the container database is shown but the pluggable database is not, select the **container database**, enter its **database credentials**, and click **Verify credentials**. The Delphix Engine will discover and list all pluggable databases in the container database. Select the **pluggable database** from the list. Alternatively, you can use the **Environment management** screen to discover pluggable databases in any of the discovered container databases.
5. Enter your **login credentials** for the source database and click **Verify credentials**.
6. Click **Next**.
7. Select a **Database group** for the dSource.
8. Click **Next**.
9. Select an **Initial load** option. By default, the initial load takes place upon completion of the linking process. Alternatively, you can set the initial load to take place according to the SnapSync policy. For example, you can set the initial load to take place when the source database is not in use, or after a set of operations have taken place.
10. Select a **SnapSync** policy. See Advanced Data Management Settings for Oracle dSources for more information.
11. Click **Next**.
12. Review the **dSource configuration** and **Data management** information.
13. Click **Submit**. The Delphix Engine will initiate two jobs, **DB_Link** and **DB_Sync**, to create the dSource. You can monitor these jobs by clicking **Active Jobs** in the top menu bar, or by selecting **System > Event Viewer**. When the jobs have completed successfully, the database icon will change to a dSource icon on the **Environments > Datasets** screen, and the dSource will be added to the list of **Datasets** under its assigned group. Link/Sync of the Multitenant Container Database The **DB_Link** job will also link the pluggable database's multitenant container database if it has not been linked yet.

Staging push implementation for Oracle

This section covers the following topics:

Oracle staging push-overview

Introduction

The staging push feature allows you to ingest production data into the Delphix Engine without the need for the Delphix-initiated backups. It uses a staging database instance that can be populated with production data, by restoring your existing backups or by setting it up as a physical standby, using storage provided by the Delphix Engine. Delphix Engine then uses the database files on this storage to take dSource snapshots. Thus, eliminating the need to access the production environment. Since the restore process is not controlled by Delphix Engine, any backup vendor can be used to restore production database backups on the staging database.

Workflow

Ingesting data from Staging Push dSources consists of the following steps:

- Linking an Oracle Staging Push dSource
- Populating staging database with production data
- Taking a snapshot

The staging database can be populated further with incremental data from the production database to take incremental snapshots.

- Staging Database should be in the MOUNT, READ ONLY or READ ONLY WITH APPLY mode for taking a snapshot.
- Disable or enable operations are allowed only at the CDB level. Disabling a CDB dSource will also disable all the PDB dSources.
- Detach or attach operations are not allowed on the staging push dSources.
- Delete operation is allowed for both CDB and PDB dSources. However, since PDBs cannot be unplugged/dropped from an Oracle database in the MOUNT state, you will be responsible for unplugging/dropping the PDB from the CDB. Deleting a CDB dSource will also delete all the PDB dSources.
- A Non Multi Tenant Staging Push dSource can not be used to ingest a multi-tenant database. For a multi-tenant database, separate Staging Push dSources should be created for CDB and individual PDBs.

Linking an Oracle staging push dSource

This section covers the following topics:

- Linking an Oracle non multi tenant staging push dSource
- Linking an Oracle multi tenant staging push dSource

Linking an Oracle non multi tenant staging push dSource

You can perform the following steps to link a Non Multi Tenant dSource using the staging push mechanism.

1. Login to the **Delphix management** application.
2. Go to **Manage > Datasets**.
3. Click the plus (+) icon and select **Add dSource**.
4. On the **Preparation** tab, click **Next**.
5. From the **dSource type** tab, select the **Oracle staging push** as dSource type and click **Next**.
6. On the **dSource configuration** step, provide the following dSource configurations:
    a. **dSource name** - This name will be shown on the Delphix Engine interface.
    b. **Target group** - Select a target group from the drop-down list.
    c. **Database type** - Select the database type as **Non Multi Tenant**.
    d. **Staging environment** - Select an environment from the drop-down list. The staging database will be hosted in this environment.
    e. **User** - Select an environment user from the drop-down list.
    f. **Repository** - Select a repository (Oracle installation) from the drop-down list.
    g. **Database name:** Provide the Database Name for the staging dSource.
    Info : Consider the following while providing the database name:
        - This is the name of the staging database that is created on the staging host after linking.
        - The Database Name should be the same as the source database's database name. The source database is the database whose data will be populated on this staging database.
        - Database Name is case-sensitive.
        - You can run either of the two queries on the source database to get the database name:
            - `show parameter db_name;`
            - `select name from v$database;`
    h. **Database unique name** - Provide a database unique name for the staging database. It can be different from the database unique name of the source database.
    i. **SID** - Provide the SID for the staging database.
    j. **Mount base** - Provide a mount path. This is the location on the staging host at which the Delphix Engine will mount its storage for the staging database. To know about the permissions that the mount path directory must-have, see the Additional Target/Staging Host Requirements section of this article.
    k. **Custom environment variables** - Configure the custom environment variables, similar to VDBs, if needed.
    l. Optionally, you can configure **Auto staging push restart**, **Configure staging database parameters**, **Physical standby**, and **Validate snapshot by Opening database in read only mode**, as per requirement.
        i. **Auto staging push restart** indicates whether this staging database should be automatically restarted when the staging host reboot is detected.
        ii. **Configure staging database parameters** to override Oracle database configuration parameters.
        iii. **Physical standby** indicates that the staging database will be configured as a physical standby.
        Info: The staging database is not configured as a Physical Standby by Delphix Engine. The setup should be done outside Delphix Engine before initiating the Snapshot operation.
        iv. **Validate Snapshot by Opening Database in Read Only Mode** indicates whether the staging database snapshots will be validated by opening the database in read-only mode.
        Info: All these options are disabled by default.
    m. Click **Next**.
7. From the **Staging database parameters** tab, select a template and click **Next**.

8. From the **Data Management** tab, select **LogSync** checkbox to **enable** the logsync for Staging Push dSource. For more information on **LogSync**, see Staging Push LogSync, click **Next.**
9. From the **Policies** tab, select any policy for the new dSource. **SnapSync Policy** is used as a default policy for taking snapshots. For more information on SnapSync policy, see Policies for Scheduled Jobs Click **Next**.
10. From the **Hooks** tab, enter any script that should be run before the snapshot operation (pre-sync), after snapshot but before post-sync hook (pre-log-sync) or after the snapshot operation (post-sync).
11. From the **Summary** tab, review the staging Push dSource configuration profile and click **Submit**. When all the above steps are executed on the staging host, a new database is instantiated with the above provided Database Name, Database Unique Name, and SID, and the database files are expected to be placed at the location: `MOUNT_BASE/<Database_Unique_Name>/datafile`. Once linking is done, you can see this Staging Push dSource under the **Datasets**.

Linking an Oracle multi tenant staging push dSource

Oracle Multi Tenant Staging Push dSource can be linked with the following steps:

1. Link a CDB staging push dSource
2. Link a PDB staging push dSource

- A CDB Staging Push dSource must be created before creating a PDB Staging Push dSource.
- A CDB dSource ingests data for the CDB and PDB$SEED databases.
- Except for PDB$SEED, a PDB dSource must be created for each PDB associated with the CDB.

Linking a CDB staging push dSource

You can perform the following steps to link a CDB dSource using the staging push mechanism.

1. Login to the **Delphix management** application.
2. Go to **Manage > Datasets**.
3. Click the plus (+) icon and select **Add dSource**.
4. On the **Preparation** tab, click **Next**.
5. From the **dSource Type** tab, select the **Oracle staging push** as dSource type and click **Next**.
6. On the **dSource configuration** step, provide the following dSource configurations:
    a. **dSource name** - This name will be shown on the Delphix Engine interface.
    b. **Target group** - Select a target group from the drop-down list.
    c. **Database type** - Select the database type as CDB.
    d. **Staging environment** - Select an environment from the drop-down list. The staging database will be hosted in this environment.
    e. **User** - Select an environment user from the drop-down list.
    f. **Repository** - Select a repository (Oracle installation) from the drop-down list.
    g. **CDB name:** Provide the Database Name for the staging CDB dSource.
    h. **CDB unique name:** Provide CDB unique name for the staging database. It can be different from the database unique name of the source CDB database.
    Info: Consider the following while providing the CDB name:
        - This is the name of the staging container database that is created on the staging host after linking.
        - It should be the same as the source CDB's Database Name
        - It is case-sensitive
        - You can run either of the two queries on the source database to get the CDB database name:
            - `show parameter db_name;`
            - `select name from v$database;`
    i. **SID** - Provide the SID for the staging database.
    j. **Mount base** - Provide a mount path. This is the location on the staging host at which the Delphix Engine will mount its storage for the staging database. To know about the permissions that the mount path directory must have, see Additional Target/Staging Host Requirements section of this article.
    k. **Custom environment variables** - Configure the custom environment variables, similar to VDBs, if needed.
    l. Optionally, you can configure **Auto staging push restart**, **Configure staging database parameters**, **Physical standby**, and **Validate snapshot by opening Database in read only mode**, as per requirement.
        i. **Auto Staging Push Restart** indicates whether this staging database should be automatically restarted when the staging host reboot is detected.
        ii. **Configure Staging Database Parameters** to override Oracle database configuration parameters.
        iii. **Physical Standby** indicates that the staging database will be configured as a physical standby.
        Info: The staging database is not configured as a Physical Standby by Delphix Engine. The setup should be done outside Delphix Engine before initiating the snapshot operation.
        iv. **Validate Snapshot by Opening Database in Read Only Mode** indicates whether the staging database snapshot will be validated by opening the database in read-only mode.
        Info: All these options are disabled by default
    m. Click **Next**.
7. From the **Staging Database Parameters** tab, select a template and click **Next**.

8. From the **Data Management** tab, select the **LogSync** checkbox to **enable** the logsync for Staging Push dSource. For more information on **LogSync**, see Staging Push LogSync, click **Next.**
9. From the **Policies** tab, select any policy for the new dSource. **SnapSync Policy** is used as a default policy for taking snapshots. For more information on SnapSync policy, see Policies for Scheduled Jobs Click **Next**.
10. From the **Hooks** tab, enter any script that should be run before the snapshot operation (pre-sync), after snapshot but before post-sync hook (pre-log-sync) or after the snapshot operation (post-sync).
11. From the **Summary** tab, review the Staging Push dSource configuration profile and click **Submit**.

When all the above steps are executed on the staging host, a new database is instantiated with the above provided Database Name, Database Unique Name, and SID, and the database files are expected to be placed at the location: `MOUNT_BASE/<CDB_Unique_Name>/datafile`. Once linking is done, you can see this Staging Push CDB dSource under the **Datasets**.

Linking a PDB staging push dSource

You can perform the following steps to link a PDB dSource using the staging push mechanism.

1. Login to the **Delphix Management** application.
2. Go to **Manage > Datasets**.
3. Click the plus (+) icon and select **Add dSource**.
4. On the **Preparation** tab, click **Next**.
5. From the **dSource Type** tab, select the **Oracle Staging Push** as dSource type and click **Next**.
6. On the **dSource Configuration** step, provide the following dSource configurations:
    a. **dSource Name** - This name will be shown on the Delphix Engine interface.
    b. **Target Group** - Select a target group from the drop-down list.
    c. **Database Type** - Select the database type as PDB.
    d. **Staging Environment** - Select an environment from the drop-down list. The staging database will be hosted in this environment.
    e. **Repository** - Select a repository (Oracle installation) from the drop-down list.
    f. **Container Database**: Select the Staging Push Container database to which the PDB will be plugged.
    g. **PDB Name:** Provide the PDB name for the staging PDB dSource.
    h. **Custom Environment Variables** - Configure the custom environment variables, similar to VDBs, if needed.
    i. Optionally, you can configure **Validate Snapshot by Opening Database in Read Only Mode**, as per requirement. This indicates whether the staging database snapshot will be validated by opening the database in read-only mode.
    j. Click **Next**.
7. From the **Staging Database Parameters** tab, select a template and click **Next**.
8. From the **Policies** tab, select any policy for the new dSource. **SnapSync Policy** is used as a default policy for taking snapshots. For more information on SnapSync policy, see Policies for Scheduled Jobs Click **Next**.
9. From the **Hooks** tab, enter any script that should be run before the snapshot operation (pre-sync), after snapshot but before post-sync hook (pre-log-sync) or after the snapshot operation (post-sync).
10. From the **Summary** tab, review the Staging Push dSource configuration profile and click **Submit**.

Upon submission, a new pluggable database will be created and plugged into the given Container Database on the staging host. The data files for this pluggable database are expected to be placed at the location: `MOUNT_BASE/<PDB_Name>-<CDB_Unique_Name>/datafile`. Once linking is done, you can see this Staging Push PDB dSource under the **Datasets**.

Populating staging database with production data

You can update the staging database with the latest data from the source database by either restoring source database backups over it or, by setting it up as a physical standby.

Pre-requisites

> While populating the data on the staging database, make sure that all the data files go to the below Delphix Engine mounted location:
> For Non Multi Tenant Databases:
> - *MOUNT_BASE/<Database_Unique_Name>/datafile*
>
> For Multi Tenant Databases:
> - For CDB and PDB$SEED: *MOUNT_BASE/<CDB_Unique_Name>/datafile*
> - For PDB: *MOUNT_BASE/<PDB_Name>-<CDB_Unique_Name>/datafile*

Restoring and recovering the source database

- For Non Multi Tenant database, after linking the dSource, restore your existing source database backups onto the mount location.
- For Multi Tenant databases, after linking the CDB and PDB dSources:
  - Restore CDB and PDB$SEED backups to the CDB mount location.
  - Restore PDB backup to the PDB mount location.

Automating the data population

You can automate the data population process on the staging database in the following ways:

- Create automation scripts and schedule/execute them outside of Delphix Engine.
- Configure pre/post-sync hooks to restore a backup or start/stop managed recovery process.

Taking Oracle staging push dSource snapshots

Taking a Snapshot

On initiating a sync, Delphix Engine checks the staging server for the database of the given name and if a match is found, it then takes a snapshot of its data files. Before and after taking the snapshot, Delphix Engine runs a few validations to check the readiness and consistency of the database for the snapshot.

Below are the checks performed by Delphix Engine during snapshot operation:

- The staging database is in the `MOUNT`, `READ ONLY` or `READ ONLY WITH APPLY` mode
- All data files are present on the Delphix Engine mounted storage
- For a non-standby staging database, checkpoint SCN of all online datafiles is the same
- The database should be recovered and no data files should be in `FUZZY` state
- Checkpoint SCN must advance since the last snapshot was taken
- Checkpoint SCN has not advanced during the snapshot operation

> CDB dSource snapshots cannot be taken manually. A PDB dSource snapshot operation takes a CDB dSource snapshot first before taking a PDB snapshot.

Taking an incremental snapshot

The staging database can be populated further by:

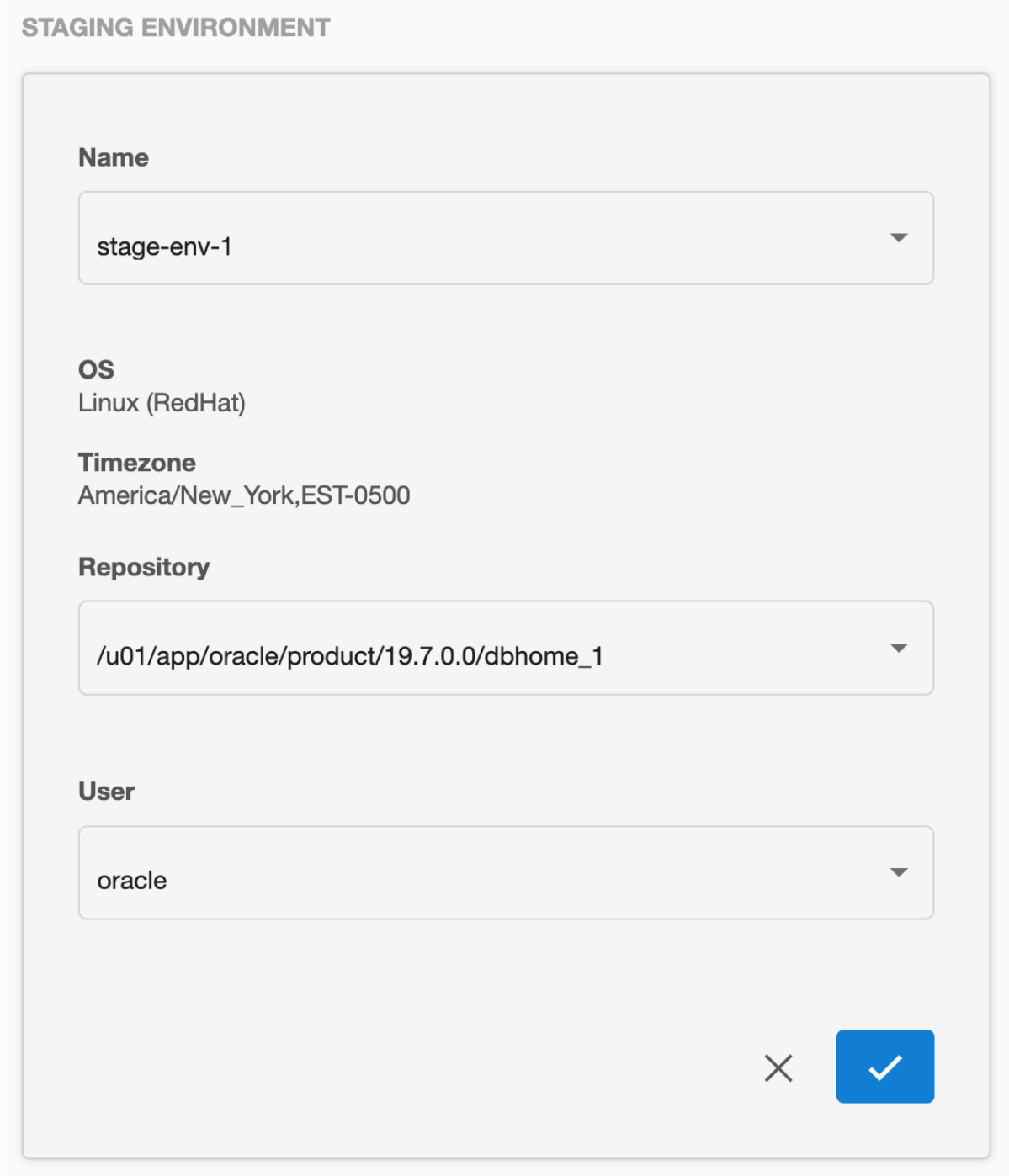
- Restoring incremental backups
- Through ongoing managed recovery in case of Physical Standby databases

After the staging database is populated with incremental changes, a snapshot operation will take an incremental snapshot of the dSource.

Migrating to another repository

The staging push database can be moved to another repository in the same or a different environment using the following steps:

1. Disable the staging push dSource.
2. Under **Datasets**, go to dSource **Configuration** tab.
3. Click on the pencil icon next to **Staging Environment** to get into edit mode.
4. Update the Environment **Name** (if migrating to another environment), **Repository** and **User**.



5. Enable the dSource.

Staging Push LogSync

Introduction

Staging Push LogSync processes the archive logs available at the mounted **archive** file system on the staging host. Once the archived logs are processed, log ranges will appear on the snapshots which can be used for the point-in-time or SCN based provisioning from the Staging Push dSource snapshots.
The archived logs should be placed in the mounted **archive** file system by configuring the archive destination (e.g. “log_archive_dest_1”) to the mounted dSource **archive** directory on the staging host.

- For a non-standby database - Restore the archived logs on the mounted **archive** directory.
- For a standby database - Configure the archive destination to the mounted ‘archive’ directory for the archived logs to be generated in this directory.

The archive filesystem is mounted to the following directories on the staging host:
For Non Multi Tenant Databases:

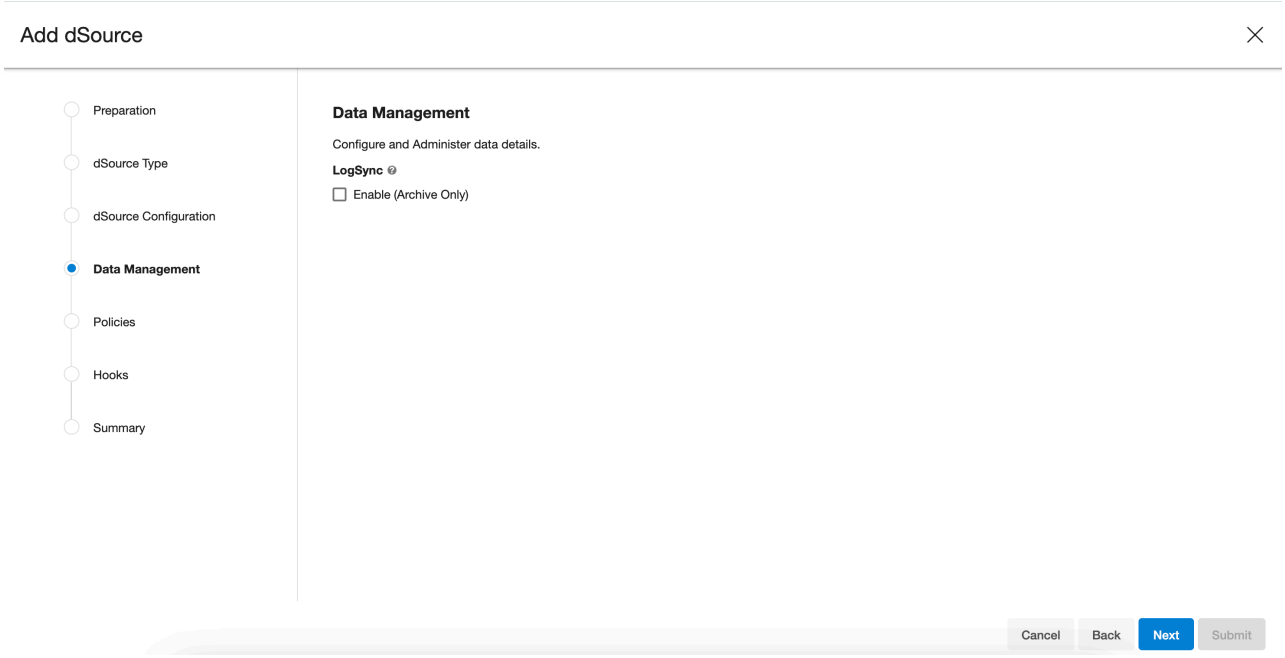
- *MOUNT_BASE/<Database_Unique_Name>/archive*

For Multi Tenant Databases:

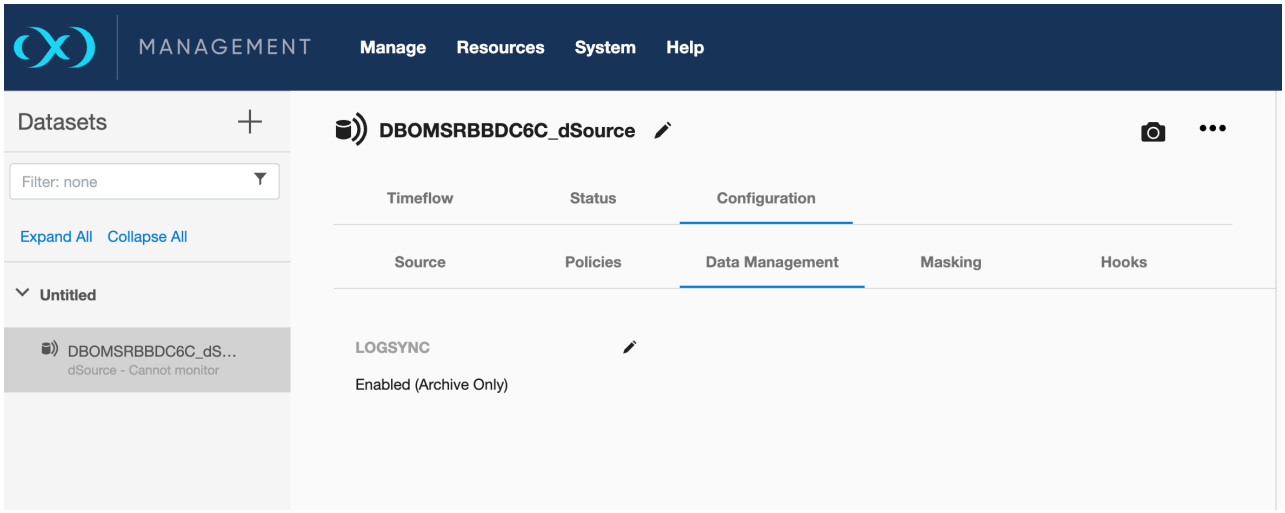
- *MOUNT_BASE/<CDB_Unique_Name>/archive*

Enable/Disable LogSync

You can enable LogSync from the **Data Management** screen by selecting the **LogSync** checkbox during the Staging Push dSource creation.



Once dSource is created, LogSync can be enabled/disabled from the **Data Management** screen under the **Configuration** tab by clicking on the pencil icon next to **LogSync** and then selecting/unselecting the **LogSync** checkbox.



Workflow

Staging Push LogSync operation runs every 5 minutes and consists of the following steps:

- Process the archived logs only if at least one staging push dSource snapshot exists.
- Read and process all the new archived log files available in the mounted **archive** file system after the last LogSync.
- Delete invalid archive log files. Online redo logs and SRLs are not processed and will be deleted.

PreLogSync Hook

PreLogSync hook can be used to switch the logs on the primary database in case of a standby staging push setup. It runs after the snapshot is successfully taken but before the post-sync hook.

Data management settings for Oracle data sources

Each dSource has its own data management settings, which can be configured during the linking workflow as well as in the configuration page for that dSource.

You can configure data management settings to improve overall performance and match the needs of your specific server and data environment.

The following settings are available for Oracle data sources:

Setting	Explanation
Initial Load	By default, the initial load takes place upon completion of the dSource linking wizard. Alternatively, you can set the initial load to take place according to a SnapSync policy. This might be desirable if you want the initial load to take place when the source database is under a lower load.
External Data Directory	An optional directory on the database server containing extra files associated with the database that are not included in normal SnapSync snapshots, for example, external table data files. Nothing in this directory will be modified, but the files will be loaded and provisioned along with the database.
Compression	Enable compression of backup data sent over the network. Compression is enabled by default.
Bandwidth Limit	Select the network bandwidth limit in units of MB/s between the source and the Delphix Engine. The default is 0, indicating that no bandwidth limit is enforced.
Number of Connections	Select the number of TCP connections to use between the Source and the Delphix engine. Multiple connections may improve network throughput especially over long distances and highly congested networks. The default is 1.
Encrypted Linking	Turn on encryption between the source and the Delphix Engine. Enabling encrypted linking leads to higher CPU utilization and higher throughput. Encrypted Linking is disabled by default.

Setting	Explanation
Data Load Channels	The channel settings determine the number of channels and data files per backup set. While these settings can be increased, you should consider potential adverse effects on the performance of database operations on the Source server • Number of Channels - set the number of RMAN channels used during SnapSync. The default is 2. • Files per Channel - maximum number of data files in a backup set. The default is 5. The product of files-per-channel and number of channels determines the maximum number of data files concurrently backed up by RMAN.
Block Checking	Enable logical block validation by RMAN. Checking is CPU intensive and will slow down SnapSync. Enabling block checking will detect logical inconsistencies while syncing with the source. Block checking is disabled by default.
Level Backup	Level backups should only be used to workaround Oracle bug 10146187 on physical standby sources. Switching from SCN to LEVEL mode will force a Level 0 backup. See Linking Oracle Physical Standby Databases for more information. Level Backups are disabled by default.
Skip Available Space Check	Skips checks for available space during the initial load in the Delphix Engine. The initial load will fail if the database needs more space than what is available in the Delphix Engine. This setting is only recommended after consulting with Delphix Support
Double SnapSync	Double SnapSync, or DoubleSync, takes two snapshots sequentially in order to improve the performance of subsequent operations, such as VDB provisioning and refresh. Performs the initial load without retrieving any archived logs, thereby creating a non-provisionable snapshot. At the completion of the initial load a second SnapSync that will retrieve logs will start. For more information, view the KB article

Setting	Explanation
LogSync Policy Settings	LogSync Policy Settings are available after the dSource has been added. On the Configuration page, you can select the following options: • Enabled - LogSync fetches log files from the source database, enabling the ability to provision a VDB from a specific point in time or, a database change number (SCN in the case of Oracle databases). LogSync must be enabled for this provisioning functionality to work. • Archive Only, Archive and Online Redo - these settings determine whether LogSync fetches logs from archive storage in the source database file system, or both the file system and online redo logs. Online Redo mode is not supported for physical standby or RAC databases. It is also not supported if the online redo logs are raw devices or Oracle Automatic Storage Management devices. If LogSync detects any of the restricted cases it will automatically enter into Archive Only mode, regardless of the mode that was chosen. Additionally, Online Redo mode is not supported for mounted but not open primary databases. LogSync policy settings for Oracle pluggable databases must be set at their corresponding container databases.
Validated Sync	Oracle validated sync is disabled by default. When enabled, validated sync is performed immediately after every subsequent SnapSync. See Enabling Validated Sync for Oracle for more information. Validated sync for Oracle pluggable databases is not supported in this release

Snapshot parameters

There are also parameters you can specify when creating a snapshot of an Oracle data source. These parameters are explained in the table below.

Parameter	Explanation
Force Full Backup	The Delphix Engine will by default perform an incremental backup, if for some reason a full backup is required (most often to resolve issues with datafile corruption), this option should be used.
Double Sync	Double SnapSync, or DoubleSync, takes two snapshots sequentially in order to improve the performance of subsequent operations, such as VDB provisioning and refresh. Performs the initial load without retrieving any archived logs, thereby creating a non-provisionable snapshot. At the completion of the initial load a second SnapSync that will retrieve logs will start. For more information, view the Using the Double Sync option for Oracle SnapSync

Parameter	Explanation
Do Not Resume	During the initial SnapSync, if a failure is encountered, the Delphix Engine can resume the SnapSync at a later date, this option will cause the engine to not resume, but rather to start the initial SnapSync over again.

Prepare and upgrade a Non-MT Oracle dSource to MT

Overview

This article describes how Delphix works with production databases that are converted from non-multitenant to multitenant, and the workflow for this operation. Once a dSource is converted to a multitenant PDB, it will be able to share storage blocks with its non-multitenant predecessor, and Delphix will only store the incremental changes to the database. This operation is known as STConvertedToPDBAttach and is only available via CLI.

Environment requirements

1. Delphix Engine 6.0.10.0 or later.
2. Source host with a 11g or 12c non-multitenant source database.
3. PDB that is created as a result of converting the above non-multitenant database to multitenant. This PDB can reside on a different host.
4. Source host with a CDB that hosts the above PDB.

Workflow steps

At the end of the steps below, there should be 2 containers:

- The detached non-MT container which contains all the non-MT snapshots.
- A new container representing the converted PDB and all its subsequent snapshots with the container name as the PDB name. By default the new MT container will have the name of the PDB, or a parameter can be set by the customer to specify its name during attach.

Use these steps to perform this functionality.

1. Link the non-multitenant Oracle 11g or 12c source database as a dSource within Delphix. Please note, 11g sources Patch 31220011 included in the July 2020 bundle is needed and XML must be installed to the DB.
2. Before performing the conversion to multitenant, detach the dSource and shut it down. Visit the Detach/Re-Attach Oracle dSource article for this step and step 7.
3. Convert the non-multitenant database to a PDB following procedures provided in the Oracle documentation. Delphix recommends that the new PDB have the same name as the original non-multitenant database. The new CDB name should be different from the original non-multitenant database.
4. Discover the CDB in Delphix Engine if not already present.
5. Refresh the environment to make sure the new PDB is discovered.
6. If the PDB name has the same name as the non-multitenant then the old non-MT container can be renamed or a parameter to give a different name to the PDB database should be used.
7. Attach converted PDB to non-multitenant dSource using CLI (attach-converted-pdb sample script).

Converted PDB rules BEFORE SnapSync

- Attachment of a converted PDB does not perform automatic SnapSync, even if linkNow=true is used. This is to allow detachment of a converted PDB without negative effects on a non-MT container.
- If for any reason it is necessary to re-convert the PDB (after re-opening to the non-MT database to perform changes), then the converted PDB should be removed instead of detached so that the STConvertedToPDBAttach can be done again to save space.
- Once the converted PDB is removed without performing SnapSync on it, it is possible to attach the non-MT container to its non-MT database.

Converted PDB Detach vs. delete BEFORE SnapSync

- Detaching a converted PDB will make it possible to attach the non-MT container to its non-MT database.
- However, if the non-MT database is reconverted and regular attach is used to attach the PDB, there will be no space de-duplication since the detach operation reverted the changes as it was before the STConvertedToPDBAttach operation.
- Detach of a converted PDB and subsequent attach should only be performed as long as the PDB has not been re-created/re-plugged to obtain space de-duplication.
- In general a converted PDB should be removed if no SnapSync is taken.

Converted PDB rules AFTER SnapSync

- Once a SnapSync is taken of a converted PDB, the non-MT container cannot be re-attached to any other container (neither to a non-MT database or to a converted pdb).
- This means that once SnapSync is taken, the non-MT container cannot be used to attach to a converted PDB anymore.
- Even if the converted PDB is removed after taking a SnapSync, attaching to the non-MT container will not be possible as the non-MT container is marked as already converted.

AttachSource/data properties

These are all the possible properties that can be configured during the attach of a converted PDB. Of these, only newPdbContainer and force have special meaning for a converted PDB. The rest are 'regular' attachment options.

```
machine.name database 'DBOMSR' attachSource attachData *> ls
Properties
    type: OracleSTConvertedToPDBAttachData (*)
    backupLevelEnabled: (unset)
    bandwidthLimit: (unset)
    checkLogical: (unset)
    compressedLinkingEnabled: (unset)
    config: (required)
    encryptedLinkingEnabled: (unset)
    environmentUser: (required)
    externalFilePath: (unset)
    filesPerSet: (unset)
    force: (unset)
    linkNow: (unset)
    newPdbContainerName: (unset)
    numberOfConnections: (unset)
    operations: (unset)
    oracleFallbackCredentials: (unset)
    oracleFallbackUser: (unset)
    rmanChannels: (unset)
```

attach-converted-pdb sample script

```
cd /database
select mydatabase
attachSource
edit attachData
set type=OracleSTConvertedToPDBAttachData
```

```
set newPdbContainerName=<name for the container instead of PDB name>
set config=<converted PDB name>
edit oracleFallbackCredentials
set type=PasswordCredential
set password=<database login password>
back
set oracleFallbackUser=<database login username>
set environmentUser=<environment user>
commit
```

Repository templates for Oracle databases

The primary use case and motivation for repository templates are to provide the Delphix administrator with control over the Oracle database parameters used during the staging phase of the VDB provisioning process. It is useful to be able to control these configuration parameters when the physical capabilities of the staging machine, such as CPU count and memory, are smaller than the physical capabilities of the machines hosting the source database repository.

The repository template is a relationship between three entities:

- A database repository – It is the Oracle home to which the vPDB will be provisioned on the target host.
- A database container – It is the source PDB.
- A VDB configuration template – A list of database configuration parameter names and values that you can save on the Delphix Engine to use at a later time. The VDB configuration template for the temporary CDB must contain at least the following database parameters:
  - **compatible**: Required for both Non-Multitenant and Multitenant sources. Must be set with the same value as in the dSource 'compatible' parameter.
  - **enable_pluggable_database**: Required for Multitenant sources only. Must be set to TRUE.

During the staging process, if you do not specify a repository template, then by default, the Delphix Engine will use the configuration parameters taken from the source database to configure the staged database. These parameters may not be appropriate, because the machine used for staging may be physically inferior to the machine hosting the source database.

Instead, the Delphix administrator can create a VDB configuration template, which would be appropriate for the physical machine hosting staging repository. (See VDB Config Templates) Then the admin can create a repository template entry that will bind together the VDB configuration template, database repository, and database container. This instructs the Delphix Engine to use configuration parameters from the VDB configuration template whenever the database container is staged on the database repository specified, instead of the parameters on the source database.

Currently, repository template relations can only be created via the command-line interface (CLI) in the repository > template.

Provisioning a VPDB using a Repository Template

Perform the following procedure to provision a VPDB using a repository template.

1. Create a VDB configuration template using the Delphix GUI called aux_cdb_params with the following parameters set as a bare minimum.

   ```
   compatible = 12.1.0.2
   enable_pluggable_database = true
   sga_target = 1551892480
   ```

   The parameters required may differ given a specific environment's requirements from this but this should allow the startup of the auxiliary database.
2. Create a repository template called aux_cdb_templ using the CLI.

   ```
   delphix repository template> create
   delphix repository template create *> set name=aux_cdb_templ
   delphix repository template create *> set container=PDB12C1
   delphix repository template create *> set repository=OELC9/'/u01/app/oracle/
   12.1'
   ```

```
delphix repository template create *> set template=aux_cdb_params
delphix repository template create *> commit
```

```
delphix repository template 'pb12c1rt'> ls
Properties
    type: SourceRepositoryTemplate
    name: aux_cdb_templ
    container: PDB12C1
    reference: REPOSITORY_TEMPLATE_REF-4
    repository: OELC9/'/u01/app/oracle/12.1'
    template: aux_cdb_params
```

In the above example:

a. **name** is what you want the template name to be.
b. the **container** is the source PDB.
c. the **repository** is the target environment/ORACLE_HOME that the CDB target is running from.
d. the **template** is the name of the database template created in the GUI.

3. Provision a virtual pluggable database (VPDB) via the Delphix GUI using the source PDB defined as the container in the repository template and provision the VPDB to the target environment defined by the repository (target environment/Oracle Home combination) in the repository template.

Using the PurgeLogs operation

Retention policy does not delete the archived logs which are

- Older than the log retention period(can be referred as expired logs) but are required to keep the snapshots provisionable
- Generated after the most recent snapshot
- Required to make retention-proof bookmark points provisionable

Under certain conditions, it may become necessary to delete more archived logs which cannot be deleted by retention policy. The purgeLogs operation can be used to perform this task.

PurgeLogs operation is capable of deleting the archived logs which are older than the log retention period but were not deleted by retention policy to keep the snapshots provisionable. If required, it can also delete the logs which are generated after the most recent snapshot. This operation, however, will always retain the archive logs required to keep the retention proof bookmark points or snapshots marked with 'keep forever' (pinned snapshots) provisionable.

PurgeLogs operation is a CLI only operation. For a multitenant dSource, this operation should be performed for CDB only as logs for CDB and all associated PDBs are managed by the CDB itself.

Example scenario

The retention policy does not delete the logs which are generated after the most recent snapshot. If no new snapshots are being taken within a reasonable period for any reason, the archive logs will continue to accumulate in the Delphix engine. This can lead to high memory and disk usage and can also make the engine unstable. This can also prevents new snapshots from being taken. By clearing these logs, the purgeLogs operation can help to free up the storage space in the Delphix engine.

> ⓘ The purgeLogs operation does not function with VDBs or any non-Oracle data platforms at this time.

Strategies for archive logs deletion

Strategy 1

- For non-multitenant dSources, it will delete all the expired logs associated with snapshots as required until the target space to reclaim is achieved.
- For multitenant dSources, it will delete all the expired CDB logs associated with the CDB and PDB snapshots as required until the target space to reclaim is achieved.

It deletes the oldest snapshot logs and moves to the newest one in order for each Timeflow starting from the current Timeflow first. The logs required for retention proof bookmarks and pinned snapshots are not deleted.

Strategy 2

- For non-multitenant dSources, it will delete the logs which are newer than the latest snapshot in the current Timeflow to free up space which retention cannot free.
- For multitenant dSources, it will delete the logs which are newer than the latest PDB snapshot in the current Timeflow to free up space which retention cannot free. Here, the latest PDB snapshot indicates the most recent snapshot among all PDBs in CDB.

Deletion is done in the order from the most recent logs to the oldest one because of the possibility of Timeflow repair at a later time. The deletion is done based on the actual compressed file size of the log sequence on the Delphix Engine. If there is a retention proof bookmark, this operation preserves the logs required to provision from the retention proof bookmark This operation also spares the online in-flight log sequence since chunks of this sequence will keep coming in (this only applies for logsync mode set to Archive + Online Redo).

PurgeLogs operation takes three arguments :

storageSpaceToReclaim - The amount of space user wants to free up (eg; 20K, 5M, 1G etc). This is mandatory and non-zero. Actual space reclaimed is dependent on whether there are enough logs to be deleted. By default, **Strategy 2** is always in effect. The user has the ability to run **strategy 1** if their target space reclaim is not met, the user can run **strategy 1** using the **deleteSnapshotLogs** argument.

deleteSnapshotLogs - This is set to false by default. If set to true, then logs are deleted based on **strategy 1**. It is possible that no more logs are available for deletion as per strategy 1 and the target storage space to reclaim is not yet achieved. In such a scenario, logs would be deleted as per strategy 2 until the target is achieved or there are no more logs to delete. If this is set to false, then deletion is done based on **strategy 2** only.

dryRun - This is a boolean argument and set to true by default. If it is true, then the operation does not actually delete the logs. This argument can be used to see which snapshots and parts of the Timeflow will be affected by the operation. This operation returns the time flow point beyond which logs might be deleted and the set of non MT/PDB snapshots affected by this operation (and will not be provisionable afterwards). When the operation is run in non-dryrun mode, we compute the time range of the snapshot and provide the endpoint as the end of the truncated Timeflow. In dryrun mode, we return the endpoint of the first log which survived as the Timeflow point beyond which it has been truncated.

- Use this operation with `setopt format=json`, otherwise, the list of snapshots may be truncated if there are more than 3.
- **Strategy 1** is an optional step and to be used in the event **Strategy 2** does not clear enough space desired by the user. Please note that any snapshots affected by this step enabled can no longer be provisioned from. The user will be required to restore archivelogs to re-enable provisioning of the affected snapshot. To repair this function please use the TimeFlow Repair Tool.
- VDB transaction logs can be manually removed from the **archive** filesystem mounted on the target for each VDB.

This example shows how to free 50GB of space from container "example_container" using strategies 1 and 2:

1. Log into the CLI.
2. Go to the container using `/database`.
3. Set the output format into json using this statement: `setopt format=json`.
4. Select the Oracle non-multitenant/CDB container using `select example_container`.
5. Select operation `purgeLogs`.
6. Set the target space to reclaim to 50GB using `set storageSpaceToReclaim=50G`.
7. ***(Optional Step)*** `deleteSnapshotLogs=true`. **Warning:** this step should be used only in the event that Strategy 2 does not free enough space. This will remove the ability to provision from affected snapshots or require the restoration of archivelogs to enable provisioning. Use `set dryRun=true` to view the snapshots which will be affected without deleting the logs. Affected snapshots will be listed under the section `affectedSnapshots`.
8. Use `set dryRun=false` to enable the operation.
9. `Commit` operation.

Example usage

```
delphixengine> /database
```

```
delphixengine database> select example_container <---- Non-MT/CDB Container ---->
delphixengine database "example"> purgeLogs
delphixengine database "example" purgeLogs *> set storageSpaceToReclaim=50G
delphixengine database "example" purgeLogs *> set deleteSnapshotLogs=true
delphixengine database "example" purgeLogs *> set dryRun=false
delphixengine database "example" purgeLogs *> commit
    type: PurgeLogsResult
    affectedSnapshots:
        0:
            type: OracleSnapshot
            name: '@2014-08-05T23:29:57.576Z'
            consistency: INCONSISTENT
            container: dbdhcp3
            creationTime: 2014-08-05T23:29:57.576Z
            firstChangePoint:
                type: OracleTimeflowPoint
                location: 2825963
                timeflow: dbdhcp3/default
                timestamp: 2014-08-05T23:31:15.000Z
            latestChangePoint:
                type: OracleTimeflowPoint
                location: 2826079
                timeflow: dbdhcp3/default
                timestamp: 2014-08-05T23:31:15.000Z
            missingNonLoggedData: false
            namespace: (unset)
            reference: ORACLE_SNAPSHOT-75
            retention: 0
            runtime: (unset)
            timeflow: dbdhcp3/default
            timezone: America/New_York,EDT-0400
            version: 11.2.0.2.0
    truncatePoint:
        type: OracleTimeflowPoint
        location: 3176794
        timeflow: dbdhcp3/default
        timestamp: 2014-08-07T19:58:55.000Z
```

This output shows that the affected snapshot is '@2014-08-05T23:29:57.576Z' and the snapshot will not be provision-able. The truncate point indicates the last SCN/timestamp that can be provisioned.

Advanced linking

This section covers the following topics:

- Enabling validated sync for Oracle
- Linking Oracle physical standby databases
- Specifying external data directories for Oracle dSources and VDBs
- Linking to Oracle dSources with RMAN compression or encryption enabled
- Upgrading dSources after an Oracle upgrade
- Oracle source continuity
- Oracle liveSources
- Oracle liveSource user workflows
- Oracle RAC dSource node addition or deletion
- Using the double sync option for Oracle snapSync
- Linking dSources from an encrypted Oracle database
- Detaching and re-attaching Oracle dSources
- Provisioning from a replicated Oracle dSource
- Detaching and re-attaching a PDB dSource from a data guard site
- Working with Oracle snapshots
- Moving the PDB to a new CDB

Enabling validated sync for Oracle

This topic describes the validated sync process for Oracle databases using both the Delphix Management application and Command Line Interface (CLI).

Traditional Oracle dSource snapshots require some recovery during provisioning. By configuring validated sync for Oracle, the Delphix Engine selects a compatible Oracle installation and applies the recovery necessary to provision a snapshot immediately after each SnapSync. Snapshots that have been through this validated sync process step do not require recovery during provisioning.

⚠ The Delphix Engine may be unable to perform validated sync on a physical standby database in Real-Time Apply mode. This is because the standby may apply changes before copying the logs that contain those changes. Without the logs necessary to perform recovery, validated sync cannot be executed. However, you can still provision the snapshot when the archive logs become available on the standby.

Prerequisite: designating a staging host

In order to validate an Oracle dSource snapshot after a sync, the Delphix Engine requires a host with an Oracle installation that is compatible with the dSource. This machine is known as the **staging** host. You must explicitly designate which machines you want the Delphix Engine to use as staging hosts. All machines that have been marked as staging hosts are added to a pool. During sync validation, the Delphix Engine will select a compatible host from the pool, export the requisite archived redo logs and data files, and execute Oracle media recovery on the host. Follow these steps to designate a staging host.

1. Log into the Delphix Management application.
2. Click **Manage**.
3. Select **Environments**.
4. In the **Environments** panel, select the environment you want to designate as staging.
5. Select the **Databases** tab.
6. Scroll down to the installations you want to designate as staging and edit the **Installation Details** by clicking on the **pencil** icon.
7. Select the **Use as staging** checkbox.
8. Select the green checkmark to confirm your change.

To configure validated sync for multiple dSources with different Oracle versions, you must designate a compatible staging source for each. If multiple compatible staging sites exist, the Delphix Engine will select one at random.

The validated sync process will consume some resources on the staging host when snapshots are taken. Designating a performance-critical host as a staging host is not recommended.

ⓘ The default OS user for the staging host must have access to the Oracle installation that will be used to perform recovery during validated sync.

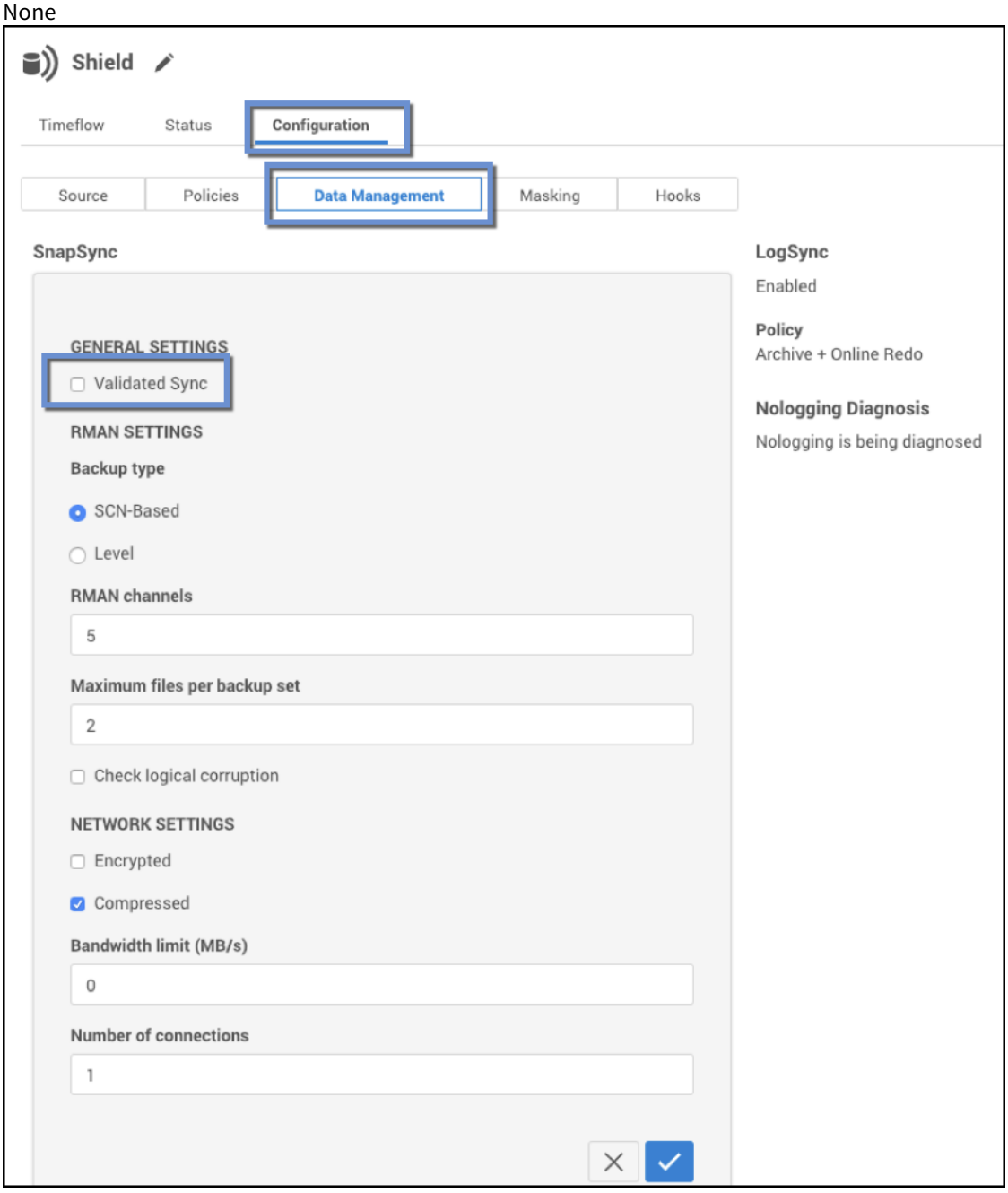
ⓘ **Oracle pluggable database**
Validated sync for Oracle pluggable databases is not supported in this release.

Enabling validated sync

Oracle validated sync can be enabled at link time or on any existing dSource. When adding the dSource (at link time), in the Data Management tab, select Show Advanced, and select the Enable checkbox for Validated Sync.

For an existing dSource:

1. Log into the Delphix Management application using **Delphix Admin** credentials.
2. Click **Manage**.
3. Select **Datasets**.
4. In the **Datasets** panel, select the dSource for which you want to enable sync validation.
5. Select the **Configuration** tab and then select the **Data Management** sub-tab.
6. Select the pencil icon located next to the **VALIDATED SYNC CONFIGURATION**.
7. Select one of the following from the **Validated Sync Mode**drop-down list:
    a. Transaction Log
    b. Full or Differential
    c. Full
    d. None



8. Select the checkmark to confirm your change.

Linking Oracle physical standby databases

This topic describes special considerations for linking Oracle physical standby databases.

The Delphix Engine supports linking both physical and logical standby databases. In previous versions of the Delphix Engine, limitations were placed upon support for Oracle RAC physical standby databases in Real-Time Apply mode. In version 3.0 of the Delphix Engine, these restrictions were lifted.

Using block change tracking (BCT) on a physical standby database
In general, Delphix recommends enabling Block Change Tracking (BCT) on a primary or standby source database. See Physical Standby Database Support Matrix below for restrictions on enabling BCT on a standby database.
BCT is available from Oracle release 11.1.0.6 onward for physical standby databases **only** if they are licensed for the Active Data Guard option.

- Release 11.1.0.6 is unstable for the BCT on physical standby feature
- Release 11.1.0.7 requires a patch for Oracle bugs 7613481, 9068088
- Release 11.2.0.2 requires patches for Oracle bugs 10170431, 12312133
- Release 11.2.0.3 requires patches for Oracle bugs 12312133, 16052165

Patches required
Enabling BCT on a physical standby database without these patches is **not recommended** because of serious performance and stability issues.

BCT on a primary database has been stable since Oracle version 10.2.0.5. In order to make use of BCT (>11.2.0.4), The Physical Standby Database must be in a "Managed Recovery Mode", i.e. achieved using "ALTER DATABASE RECOVER **MANAGED** STANDBY DATABASE'.

Physical standby database support matrix

Oracle Version	Apply Mode	Notes
11.x and 12.x in **Level Backup** mode; 12.x in **SCN Backup** mode	Archive Apply mode	No special restrictions.
	Real-Time Apply mode (RTA)	LogSync must be enabled for all database versions if RTA is enabled
11.x in **SCN Backup** mode	Archive Apply mode	If the Physical Physical Standby Database is at version 11.2.0.4 or above, no special actions are required. Due to Oracle bug 10146187, **Redo Apply** must be stopped and the database opened in **read-only** mode during SnapSync. See the section *Stopping and Restarting Redo Apply* below for more information.

Oracle Version	Apply Mode	Notes
Real-Time Apply mode	If the Physical Physical Standby Database is at version 11.2.0.4 or above, no special actions are required.	LogSync must be enabled. Due to Oracle bug 10146187, **Redo Apply** must be stopped and the database opened in **read-only** mode during SnapSync. See the section *Stopping and Restarting Redo Apply* below for more information. In addition, to avoid Oracle Bug 13075226, which results in a hang during the restart of Redo Apply, Delphix requires disabling using BCT on the standby database. The hang occurs when BCT is enabled on a standby database that uses SCN backup mode. ⚠ **Patch required** SnapSync will fail If running Oracle 11.2 before 11.2.0.4 when using SCN backups and real-time apply mode, Use level based backups instead. If the Oracle installation has already been patched for Oracle bug 13075226, or once the patch is applied, use the CLI to update the repository for this installation so that applied Patches include Oracle bug number 13075226. If the repository does not indicate that Oracle bug 13075226 for the repository has been addressed, SnapSync will not be possible when using SCN backups and real-time apply. See *Updating Repository for Oracle applied patches with the Command Line Interface* below for details on how to update the repository.
18.0.0.0 - 19.3.0.0	Real-Time Apply mode	Oracle patch 29056767 installed, Failure to install this patch may result in SnapSync failure with error.

Level backup mode for snapSync

By default, the Delphix Engine's SnapSync feature uses **SCN Backup** mode and is designed to not interfere with other backups that may already be in use. However, in cases where RMAN is not being used outside of the Delphix Engine, the Delphix Engine can use the **Level Backup** mode that improves SnapSync behavior on Oracle 11g physical standby databases. In this mode, **redo apply** does not have to be stopped during SnapSync. See Advanced Data Management Settings for Oracle dSources for more information about SnapSync settings.

Requirements for using level backup mode

Customer not backing up their physical standby with RMAN:

- Set CONTROL_FILE_RECORD_KEEP_TIME to 365

OR all of the following:

- Physical standby database running Oracle 11.2.0.2 or later version
- All RMAN backups must use tags
- RMAN CROSSCHECK commands must specify tags
- RMAN DELETE commands must specify tags

- RMAN DUPLICATE commands must specify tags
- Set CONTROL_FILE_RECORD_KEEP_TIME to 365

Failure to meet all of these requirements will cause external RMAN backups to be incomplete or result in corrupt SnapSync snapshots. Switching from SCN to LEVEL mode will force a new LEVEL 0 backup.

Stopping and restarting redo apply

Oracle bug 10146187 requires stopping of redo apply before an SCN-based incremental backup can be issued. These scripts can be used as pre- and post-scripts during the dSource linking process to stop and restart **Redo Apply**.

- SnapSync pre-script: disableStandby.sh.template
- SnapSync post-script: enableStandby.sh.template

These scripts must be modified for local use, particularly in regard to whether the physical standby database operates in MOUNTED or OPEN mode.

Failure to properly customize these scripts could violate your Oracle license terms by running redo apply on an open database, which requires an Oracle Active Data Guard license.

Linking and provisioning a mounted standby

Warning:
When you link a standby database in the mounted mode and have not been opened read-only, the data files for temporary tablespaces will be present in v$tempfile but will not actually be created yet, and therefore will indicate 0 file size. As a result, any VDB provisioned from a snapshot taken in this state will end up with the same tablespaces and datafiles created, but with a default file size of 52428800 bytes.

For databases that are in the **mounted** state, the Delphix database user account must be **SYS** (having the **SYSDBA** role), **SYSBACKUP** (having the **SYSBACKUP** role) or **SYSDG** (having the **SYSDG** role).

SYSBACKUP and **SYSDG** roles are only available in Oracle 12.1 and later releases.

However, for an **open** standby (Active Data Guard) database, only a regular database user account is required.

Connecting to a **mounted** standby with a **SYS** user account requires that the mounted standby be configured with a password file. Delphix does not capture the password file during SnapSync, and for this reason, cannot provision or sync validate a database with a SYS user. A secondary, regular database user account can be specified through either the **Delphix Management** application or CLI. This database user will then be used to connect to the database during provisioning and validated sync. Note that the **SYS** user is still required to perform snapshots of the source database.

In the **Delphix Management** application, the **non-SYS user** can be specified from within the **Add dSource** wizard, or on the back of the Oracle dSource after linking.

Configuring a standby PDB in mount mode

This procedure is optional and only applies if SCM is disabled.

To configure a standby PDB in the mount mode, you must also provide a non-SYS user for both the CDB and the PDB. The PDB non-SYS user can only be added via the CLI. You must perform a fresh SnapSync after adding the non-SYS user.

Pre-requisite

In order for Delphix Engine to connect, you must configure a static listener configuration for the PDB. You can configure a static listener by adding a configuration into listener.ora and restarting the listener.

```
SID_LIST_LISTENER=
(SID_DESC=
(GLOBAL_DBNAME=CDOMLOSR4F71PDB1)
(SID_NAME=stby18c)
(ORACLE_HOME=/u01/app/oracle/product/18.0.0.0/dbhome_1)
)
)
```

In the above example configuration, GLOBAL_DBNAME is the PDB name and SID_NAME is the SID of the CDB.

Procedure

Run the following commands to configure a PDB and CDB in the mount mode.

1. Update PDB non-SYS user.

```
# Update PDB nonsys user
delphix> /sourceconfig
delphix sourceconfig> select RH74PDB04
delphix sourceconfig 'RH74PDB04'> update
delphix sourceconfig 'RH74PDB04' update *> set nonSysUser=delphix
delphix sourceconfig 'RH74PDB04' update *> set
nonSysCredentials.type=PasswordCredential
delphix sourceconfig 'RH74PDB04' update *> set
nonSysCredentials.password=delphix
delphix sourceconfig 'RH74PDB04' update *> commit;
```

2. Update CDB non-SYS user.

```
# Update CDB nonsys user
delphix> /sourceconfig
delphix sourceconfig> select rh74cdb2
delphix sourceconfig 'rh74cdb2'> update
delphix sourceconfig 'rh74cdb2' update *> set nonSysUser=delphix
delphix sourceconfig 'rh74cdb2' update *> set
nonSysCredentials.type=PasswordCredential
delphix sourceconfig 'rh74cdb2' update *> set
nonSysCredentials.password=delphix
delphix sourceconfig 'rh74cdb2' update *> commit;
delphix sourceconfig 'rh74cdb2'>
```

3. Perform sync of the PDB.

```
# Perform sync of PDB
delphix> /database
delphix database> select RH74PDB04
delphix database 'RH74PDB04'> sync
```

Setting the Non-Sys User on the Oracle dSource

1. Create the delphix_db user in the primary database.
2. Log into the Delphix Management application.
3. From the **Manage** menu, select **Datasets.**
4. From the Configuration tab select the Oracle dSource for which you want to add a **non-SYS** user.
5. Click the dSource's icon to open the dSource information pane.
6. Click the Edit button next to **Non-SYS User**.
7. Enter a non-SYS user and credentials that exist on the standby.
8. Click the **Accept** button to save this user and associated credentials.

The non-SYS user will be used to connect to all VDBs provisioned from snapshots of this dSource that are created after the non-Sys user has been set.

Updating repository for applied patches with the command line interface

1. Select the repositoryof the database

```
delphix> repository select '/opt/app/oracle/product/11.2.0.2/db_1'
```

2. Execute the `update` command.

```
delphix repository ''/opt/app/oracle/product/11.2.0.2/db_1''> update
```

3. Set `appliedPatches` to list current patches applied to the repository.

```
delphix repository ''/opt/app/oracle/product/11.2.0.2/db_1''update *> set
appliedPatches=13075226
```

4. `Commit` the operation.

```
delphix repository ''/opt/app/oracle/product/11.2.0.2/db_1''update *> commit
```

Setting the Non-Sys User with the Command Line Interface

1. Select the `source config` of the mounted standby.

```
delphix> sourceconfig select pomme
```

2. Execute the `update` command.

```
delphix sourceconfig "pomme"> update
```

3. Set the `nonSysUser` and `nonSysCredentials` to a non-SYS user that exists on standby.

```
delphix sourceconfig "pomme" update *> set nonSysUser=<non-sys-username>
delphix sourceconfig "pomme" update *> set
nonSysCredentials.type=PasswordCredential
delphix sourceconfig "pomme" update *> set nonSysCredentials.password=<non-sys-
password>
```

4. Commit the operation.

```
delphix sourceconfig "pomme" update *> commit
```

Specifying external data directories for Oracle dSources and VDBs

This topic describes the process for including external data files with dSource snapshots and VDBs.

In the following places, you can specify the directory for any external data files that should be included with dSource snapshots:

- During the dSource linking process click on the **Advanced** section of the **Data Management** screen
- After you have created the dSource go to the **Configuration** tab

External file import for the Delphix engine and VDBs

The Delphix Engine will not fetch external tables or external data types such as BFILE. Instead, in order to link external data files to the source database and make it available to the Delphix Engine, you must create a directory in the file system and the database. Any data files in the directory you specify will be applied, recursively, to the dSource.

External data will be provisioned to each VDB that is created from this dSource. You will need to update the external file/data type definition to point to the new location after creating VDBs. Provisioning a VDB with external data creates a directory named **external** in the VDB mount point location.

Configuring the rsync command location for an environment

Files from the external data directory are fetched using the rsync command installed in the source environment. In order to SnapSync a dSource with an external data directory, rsync must be installed in the source environment. If rsync is installed in a non-standard location, the path to the rsync command can be configured in the **Environment Details** for the source environment on the **Environment Management** screen.

Example of attaching and redirecting external data files for Oracle databases

This example uses two environments:

1. **172.16.200.446** as the source environment **dinosaur** as the source database
2. **172.16.200.447** as the target environment **vdino** as the target database

Linking a dSource

1. Create an external data directory and an external data file, and attach the directory to the source database.
    a. Log into **172.16.200.446** as the environment user.
    b. Create a physical directory on the source environment. `$ mkdir /work/extdata`
    c. Create a directory in Oracle.

```
$ sqlplus / as sysdba
  SQL> create or replace directory extdata as '/work/extdata';
```

    d. Create a text file `/work/extdata/exttab.dat.`

```
$ cat > /work/extdata/exttab.dat
    1, aaa
    2, bbb
    3, ccc
    ^C
```

    e. Create an external table `exttab`.

```
$ sqlplus / as sysdba
    SQL> create table exttab (id number, text varchar2(10))
      2  organization external (default directory extdata location('exttab
.dat'));
```

f. Query the table.

```
SQL> select * from exttab;
        ID TEXT
---------- ----------
         1  aaa
         2  bbb
         3  ccc
```

2. During the process of linking the dSource to the **Dinosaur** database, or in the dSource's **Configuration** tab after creating the link, enter `/work/extdata` in the **External Data Directory** field.

Provisioning a VDB

1. Provision **vdino** from **Dinosaur**.
2. Modify the directory **extdata** in **vdino**
   a. Log into the target environment **172.16.200.447**
   b. Set `SID` to `vdino`

```
$ export ORACLE_SID=vdino
```

   c. A query to `exttab` will fail.

```
$ sqlplus / as sysdba
    SQL> select * from exttab
select * from exttab
*
ERROR at line 1:
ORA-29913: error in executing ODCIEXTTABLEOPEN callout
ORA-29400: data cartridge error
KUP-04063: unable to open log file EXTTAB_23394.log
OS error No such file or directory
ORA-06512: at "SYS.ORACLE_LOADER", line 19
```

3. Modify directory to the new location.

```
SQL> create or replace directory extdata as '/mnt/provision/vdino/external';
```

4. Query `exttab` again.

```
SQL> select * from exttab;
        ID TEXT
```

```
---------- ----------
         1  aaa
         2  bbb
         3  ccc
```

Linking to Oracle dSources with RMAN compression or encryption enabled

This topic describes the behavior of the Delphix Engine when linking to a dSource with RMAN compression or encryption enabled.

In earlier versions of the Delphix Engine, the dSource linking process would fail if RMAN compression or encryption was enabled. In order for the linking process to complete, the administrator was required to ensure that compression was not enabled for device type **SBT_TAPE**, and that encryption was also not enabled.

With the Delphix Engine, linking a dSource succeeds if compression or encryption is enabled, but the RMAN backup that creates the dSource will not be compressed or encrypted. This is true in the case where the administrator has enabled compression for tape, and in the case where the administrator is using OSB and has enabled encryption for tape.

You can check the RMAN compression and encryption settings with the commands `show device type` and `show encryption for database`, respectively.

If compression is enabled on a source environment, then it may need to be enabled for the target environment. Compression should be enabled for target environments if all segments in a source environment utilize **NOCOMPRESS** or **COMPRESS="BASIC"**.

Upgrading dSources after an Oracle upgrade

This topic describes how to upgrade dSources after an Oracle database upgrade.

Prerequisites

Do not suspend LogSync on the Delphix Engine during an Oracle upgrade of the source environment. LogSync will detect the Oracle version change. It will then notify you to refresh the host and use the Update icon on the configuration tab for all the associated dSources and VDBs (see below). Follow all Oracle instructions and documentation.

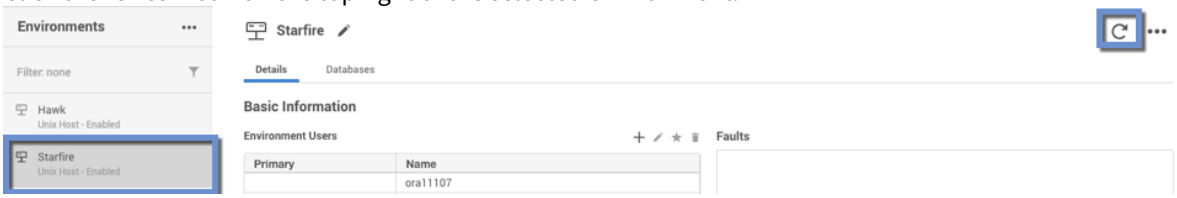
> ⚠ In order to upgrade your Oracle dSource from Non-Multi-Tenant (Non-MT) to Multi-Tenant(MT) database, please refer to Prepare and Upgrade a Non-MT Oracle dSource to MT section.

There are 2 ways to apply a PSU (Patch Set Update)/Oracle upgrade:

- Apply to existing ORACLE_HOME. Best if on Delphix v4.1.x or higher.
- Create a new ORACLE_HOME (or clone an existing one) and then apply the PSU to the new ORACLE_HOME

Applying to an existing ORACLE_HOME

1. Following Oracle documentation, patch the ORACLE_HOME and the database.
2. Click the refresh icon on the top right of the selected environment.



3. On the Refresh Environment pop-up window, click **Refresh**.
4. Under the **configuration** tab, go to the PDB and verify that the **Repository** and/or **Version** has been updated.

Creating a new ORACLE_HOME

1. Refresh the environment from the Delphix Management application.
2. In the **Databases** tab, verify that the new ORACLE_HOME appears as an ORACLE Installation.
3. Following Oracle documentation, patch the production database, etc.
4. Click **Manage**.
5. Select **Datasets**.
6. Expand the group(s) containing all the non-multitenant and multitenant dSources.
7. Select **dSource**. For upgrading a PDB or multitenant dSource, select the container CDB **dSource**.
8. Click the **Configuration** tab.
9. From the **Actions** menu (...), select **Upgrade** to switch the ORACLE_INSTALLATION to the new one.
10. Under the **configuration** tab, go to the PDB and verify that the **Repository** and/or **Version** has been updated.

> ⓘ **Updating the Oracle user after an upgrade**
> There may be cases when you upgrade the Oracle home and the Oracle User (the OS user who owns the Oracle binary) is a different user than the previous Oracle User. You will then need to update the Oracle User for each environment, and then re-connect each dSource and VDB to the upgraded Oracle home using the new Oracle User.
> The new Oracle User must be in the same OS group (for example, dba or oinstall) as the previous one.

1. Log into the Delphix Management application.
2. Click **Manage**.
3. Select **Environments**.
4. Select the **environment** where you want to add the user.
5. Next to **Environment Users**, click the **Pencil** icon to edit the new user.
6. Follow the procedure to upgrade the dSources and VDBs described in this topic.

Post-requisites

After upgrading the dSource to a new major release of Oracle (11.2.0.4 to 12.1 for example), you must re-run the createDelphixDBUser.sh script.

Oracle source continuity

Overview

In earlier versions of the Delphix Engine, when an Oracle database underwent a resetlogs operation, you were required to re-link the Oracle source. This meant that you had to completely back up the Oracle database and store it again on the Delphix Engine. If any virtual databases (VDBs) were provisioned from the dSource and needed to be saved, you had to rename and save the old dSource, resulting in a possible doubling of storage space consumed on the Delphix Engine. The old VDBs could not be refreshed to the relinked dSource.

Beginning with Delphix Engine version 4.1.1.0/4.0.6.0, the Oracle database no longer requires you to re-link sources after a resetlogs operation. The Delphix Engine will detect this condition, automatically take a new full backup, and create a new TimeFlow for the next SnapSync of the source. Benefits of the Oracle Source Continuity feature include:

- Lower storage costs and easier administration.
    - Only the changed blocks of the new SnapSync backup will be stored on the Delphix Engine. Because of the way the Delphix Engine handles duplicate blocks, the full backup likely has a storage requirement similar to an incremental backup.
- Existing VDBs provisioned from previous snapshots for the source will remain.
    - You can use and refresh those VDBs to the new snapshot

The improved user workflow replaces the old user workflow, which directed users to troubleshoot when SnapSync would fail. Begin Oracle Source Continuity in the following way:

1. The database undergoes a resetlogs operation.
2. If LogSync is enabled, it generates a fault and stops.
3. Start SnapSync. The SnapSync does a full restore of the database to a new TimeFlow, clears the fault, and restarts LogSync. If you created VDBs prior to the resetlogs operation, they will still exist after the SnapSync; you can refresh them from the new snapshot.

Creating a new timeFlow

When LogSync detects the restlogs operation, it will still stop and generate a fault. LogSync must stop because a new timeline has been created on the database. This usually happens because the database has been rewound to a past point. The transaction logs being generated on the new timeline are out of sync and conflict with logs from the old timeline. The data files are also out of sync with the data files on the Delphix Engine. You must create a corresponding new Timeflow on the Delphix Engine to store the new logs and new versions of the data files. This requires taking a new backup of the database.

Once LogSync detects the reset operation and throws the fault, no more changes will be retrieved from the database until you start a new SnapSync. This SnapSync will take a full backup, clear the fault, and restart LogSync. Only the new snapshot and Timeflow will be visible in the dSources view. Previous snapshots and Timeflow will still exist and be visible through the command-line (CLI) Capacity screen.

The following CLI output shows that the old and new Timeflow and snapshots are still available. The name of the original Timeflow for the database is "default." The name of the new Timeflow that was created during the SnapSync is "CLONE@2015-01-15T17:07:20."

```
delphix> /TimeFlow list display=name,container
```

```
NAME                          CONTAINER
```

```
'CLONE@2015-01-15T17:07:20'  dbdhcp1
```

```
default                       dbdhcp
```

```
delphix> /snapshot list display=name,container,TimeFlow
NAME                          CONTAINER  TimeFlow
'@2015-01-16T00:50:08.784Z' dbdhcp1 default
'@2015-01-16T00:52:13.685Z' dbdhcp1 default
'@2015-01-16T00:53:46.873Z' dbdhcp1 default
'@2015-01-16T00:55:18.079Z' dbdhcp1 default
'@2015-01-16T01:08:02.411Z' dbdhcp1 'CLONE@2015-01-15T17:07:20'
```

The old snapshots and Timeflow will still be subject to logfile and snapshot retention policies. You can also delete the snapshots manually. In addition, you can use the CLI to provision from the old TimeFlow.

Oracle liveSources

Oracle liveSources overview

Prior to Delphix Engine version 4.2, users ran reports against virtual databases (VDBs) that they created with the Delphix Engine. Although this workflow helped them offload the reporting load from production, the data in the VDBs was not updated asynchronously. If users wanted newer data, they had to stop their reporting applications, refresh their VDBs, and resume. In the current release, you can run reports against data that is constantly being updated. There is one live data feed for each source database that is linked as a dSource on the Delphix Engine. You can point your reporting applications to this live feed. Additionally, you will continue to have all existing Delphix functionality from the dSource, such as creating read/write VDBs.

Understanding Oracle liveSources

Oracle LiveSources leverage native Oracle Active Data Guard technology to keep a standby database up-to-date with changes happening on the source. The standby database is kept open for reads while it applies changes from the source. You can now connect to this standby database for real-time reporting needs. Using Delphix in conjunction with Active Data Guard gives you the ability to get both live up-to-date data and historical points in time from which you can provision virtual databases.

Understanding how to use Oracle liveSources

Oracle liveSources provide a read-only live data stream from Delphix

You can convert an Oracle dSource to a LiveSource, which is a real-time read-only feed of the linked source. You can access the LiveSource using a JDBC string. Internally, a LiveSource is a standby database instance tracking the Linked Source in real-time managed mode and opened in read-only mode.

Understanding Oracle liveSources with data age and threshold

One of the important utilities of a LiveSource is that it provides a real-time feed of the linked source. In some instances, due to slow networks or other reasons, the LiveSource might fall behind the linked source that it is tracking. When adding a LiveSource, you can specify a data age threshold. If the LiveSource falls behind the linked source by more than the data age threshold, the Delphix Engine will generate a fault and inform you.

The LiveSource information is displayed on the Status tab. Delphix continuously monitors the standby instance and notifies you of any abnormalities.

- Live Data Status - Indicates that the LiveSource standby database instance is actively receiving data from the source database
- Data Age - Displays the data age of the LiveSource

You can change the Data Age Threshold at any time by updating the threshold value located on the **Configuration > Data Management** tab.

Oracle liveSources quickly sync with consistent snapshots

Taking snapshots of a LiveSource is instantaneous since the standby database for the LiveSource is constantly receiving data from the source database and recovering it. Taking snapshots occurs instantaneously by taking a filesystem level snapshot of the data on the Delphix Engine without requiring an RMAN backup of the source database. All LiveSource snapshots are consistent; as a result, provisioning from LiveSource snapshots is fast, because no database recovery needs to happen.

Oracle liveSources use resync and apply

Resync is a way to refresh the LiveSource to the current point in the linked source. The following situations require a Resync to be performed:

- There are unresolvable gaps in the log sequence – for example, logs from the source database deleted before the primary database could ship them over to the LiveSource standby.
- The source database was taken through a point in time recovery/flashback, resulting in a changed incarnation.
- The source database contains non-logged changes. In this case, a Resync is needed only if you are interested in moving the non-logged data over to the LiveSource.
- The LiveSource is significantly behind the source database due to network communication issues or a large number of writes.

LiveSource Resync is a two-step operation of:

- **Start LiveSource Resync** – Start Resync performs an incremental backup of the source database to transfer the latest changes to the Delphix Engine. This operation does not affect the availability of the LiveSource
- **Apply LiveSource Resync –** Applying the Resync data will perform one more incremental backups from the source database to ensure up-to-date data, and recreate the LiveSource instance while preserving all the configurations. This operation requires downtime for the LiveSource.
- **Discard LiveSource Resync –** If the prepared resync data is no longer needed or resync has become obsolete (for example, another controlled change has been done on the source database), you can discard the current resync data with Discard LiveSource Resync. The next Resync will refetch data from the source database.

Pre-requisites: configuration and installation of staging environments to host a standby database

Oracle active data guard required

The LiveSource feature requires an Active Data Guard license. Delphix uses Active Data Guard to replicate changes from the source database that it creates on the staging environment.

Network requirements

LiveSource requires a Data Guard connection between the source and the standby database which utilizes TNS listeners associated with the databases.

Database requirements

LiveSource requires Enterprise Edition of Oracle Database.

LiveSource supports:	**LiveSource does not support:**
Oracle support matrix	
Oracle 11g and non-multi tenant Oracle 12c, 18c, and 19c	Oracle Standard Edition on the source and staging environments
Physical and standby source databases	LiveSources running on a RAC

Oracle liveSource user workflows

Please use the following documentation as a guide to identify and act on common Oracle liveSource User Workflows. The following table of contents includes steps for how to convert a dSource into a liveSource, a provision from a liveSource, sync a liveSource, convert a liveSource back to a dSource, and many other data procedures.

Converting to liveSource from a dSource

To get a live feed to the source database data through the Delphix Engine, you must first link the database to the Delphix Engine to create a dSource. You can then convert the dSource into a liveSource by following the steps outlined below:

1. In the left-hand panel, click the **dSource**.
2. From the Actions menu (...) select **Convert to liveSource**, as highlighted below. This launches the **Convert to liveSource** wizard.

Convert to liveSource database wizard

1. In the Overview tab select **Next**.
2. Enter a **DB Unique Name** for the liveSource.
3. Enter a **Database SID** for the liveSource.
4. Click **Next**.

> The liveSource database name must be the same as the database name of the primary database; therefore, this value is read-only.

Convert to liveSource, environment tab

Select the environment on which the liveSource will be created:

1. Select an **Environment User** for the liveSource instance.
2. Enter the **Mount Point** for the liveSource instance.
3. Select **Listeners** as needed. If you enable **Auto Select Listeners**, the Delphix Engine will pick the first available listener from the environment.
4. Click **Next**.

Convert to liveSource, configuration template tab

1. Select **VDB configuration templates** for the liveSource.
2. Enter additional **DB configuration parameters** for the liveSource.
3. Click **Next**.

Convert to liveSource, data management tab

1. Enter the **data age warning threshold** for the liveSource. If the data in liveSource lags behind the source database by more than this threshold, the Delphix Engine will raise a fault and notify you.
2. Click **Next**.

Convert to liveSource, hooks tab

1. Enter the **operations** to be performed on initial conversion. These operations are performed after the Delphix Engine has created the standby database for the liveSource.
2. Click **Next**.

These operations will also be performed when resyncing a liveSource.

Convert to liveSource, summary tab

1. Review the configuration summary.
2. Click **Convert** to begin the conversion.

Setting up log transport between a dSource or primary database and a liveSource or standby database

After adding a liveSource instance, you must configure the log transport between the dSource or primary database and the liveSource or standby database. For details on configuring a standby database, refer to the Oracle Data Guard Concepts and Administration guide.

At source/primary database

1. Configure the `LOG_ARCHIVE_CONFIG` parameter to enable the sending of redo logs to remote destinations and the receipt of remote redo logs (the liveSource instance). For example: `alter system set log_archive_config='DG_CONFIG=(sourcedb,liveSource)' scope=both;`
2. Configure the `LOG_ARCHIVE_DEST_n` parameter to point the redo logs to the liveSource instance. For example: `alter system set log_archive_dest_2='SERVICE="(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=liveSource.dcenter.delphix.com)(PORT=1521))(CONNECT_DATA=(SERVICE_NAME=liveSource)(SERVER=DEDICATED)))" ASYNC VALID_FOR=(ONLINE_LOGFILE,PRIMARY_ROLE) DB_UNIQUE_NAME=liveSource' scope=both;`
3. Create a **passwd file** for the liveSource into the target site.
4. Configure the corresponding `LOG_ARCHIVE_DEST_STATE_n` parameter to identify whether the log transport is enabled. For example: `alter system set log_archive_dest_state_2='ENABLE' scope=both;`
5. Configure the STANDBY_FILE_MANAGEMENT parameter to enable automatic standby file management. For example: `alter system set standby_file_management='AUTO' scope=both;`

RAC instance as standalone environment

For RAC instances that are discovered as Standalone type environments, please apply the following steps instead of the steps above.

1. `alter system set log_archive_config='DG_CONFIG=(sourcedb,liveSource)' sid='' scope=both;`
2. `alter system set log_archive_dest_2='SERVICE=liveSource ASYNC VALID_FOR=(ONLINE_LOGFILE,PRIMARY_ROLE) DB_UNIQUE_NAME=liveSource' sid='' scope=both;`
3. Set up **tnsnames.ora** in both source and target sites.
4. Make sure to have the same password file in all RAC instances and target site.
5. `alter system set log_archive_dest_state_2='ENABLE' sid='' scope=both;`

6. `alter system set standby_file_management='AUTO' sid='' scope=both;`

> The Delphix DB user must be updated to include grants for select on v$managed_standby and v$dataguard_stats in order to access MRP information and DataGuard statistics:

```
grant select on v_$managed_standby to <delphix DB username>;
create synonym <delphix DB username>.v$managed_standby for v_$managed_standby;
```

```
grant select on v_$dataguard_stats to <delphix DB username>;
create synonym <delphix DB username>.v$dataguard_stats for v_$dataguard_stats;
```

At the staging environment where the liveSource standby database environment is running

1. Configure the `FAL_SERVER` parameter to point to the primary database for proper fetch archive log function. For example:
   `ALTER system SET fal_server='service="(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=sourcedb.dcenter.delphix.com)(PORT=1521))(CONNECT_DATA=(SERVICE_NAME=sourcedb)(SERVER=DEDICATED)))"';`
2. If not already created, configure a password for Data Guard.

Removing a liveSource

1. In the **Datasets** panel, click the **liveSource**.
2. Click the **Configuration** tab.
3. From the Actions menu (...) select **Convert to dSource.**
4. Click **Convert to dSource** .

Taking a snapshot on a liveSource

To take a snapshot of a liveSource:

1. In the **Datasets** panel, select the **liveSource**.
2. In the upper right-hand corner, click the **camera** icon.
   liveSource snapshots are instantaneous, Quick Provision snapshots. They do not require an RMAN backup of the source database

Provisioning from a liveSource TimeFlow

Provisioning from a liveSource Timeflow is the same process as provisioning from a snapshot for dSource Timeflow. The only difference is that you will select a liveSource and a liveSource snapshot.

Enabling, disabling, and detaching a liveSource

A liveSource is **enabled** the same way as a regular dSource.

1. Login to the Delphix Management application as **admin** or another user with administrative privileges.
2. Click **Manage**.
3. Select **Datasets**.
4. Click the **liveSource** you want to enable.
5. From the Actions menu (...) select **Enable.**

6. Click **Enable** to confirm.

> When you enable the liveSource, the Delphix Engine will recreate the standby database on the staging environment.

A liveSource is **disabled** the same way as a regular dSource. Disabling a liveSource will stop further operations on the Delphix Engine related to the liveSource.

1. Login to the Delphix Management application as **admin** or another user with administrative privileges.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **liveSource** you want to disable.
5. In the upper right-hand corner, from the **Actions** menu (...) select **Disable.**
6. In the Disable dialog select **Disable**.

When you are ready to enable the liveSource again, from the Actions menu (...) select **Enable**, and the liveSource will continue to function as it did previously.

> Disabling a liveSource shuts down the standby database that Delphix manages on the staging environment.

You can detach a liveSource in the same way as detaching a regular dSource. Detaching a liveSource will implicitly convert the liveSource into a regular dSource. After a dSource is re-attached, you can convert it back to a liveSource.

Resyncing a liveSource and applying the resync

Resync is a way to refresh the liveSource to the current point in the linked source. Resync is a multi-phase operation comprised of the following:

Perform resync

1. Click **Manage**.
2. Select **Datasets**.
3. Select your liveSource.
4. From the Actions menu (...) select **Start liveSource Resync**. The liveSource can stay up while the Resync is in progress.

Discarding resync data

Prerequisites

- Resync is started and ready to apply

After Resync has finished, you can choose to not apply but rather discard the data that was brought over from the source database as part of Resync.

Procedure

To discard the data:

1. Click **Manage**.
2. Select **Datasets**.
3. Select your liveSource.
4. From the Actions menu (...) select **Discard liveSource Resync.**

Applying resync data

Prerequisites

- Resync started and ready to apply

Procedure

1. Click **Manage**.
2. Select **Datasets**.
3. Select your liveSource..
4. From the Actions menu (...) select **Apply liveSource Resync.**
5. If the apply resync data process fails, first investigate and resolve the cause of failure, such as a full disk. Then follow the procedure to start resync.

Migrating a liveSource

1. Click **Manage**.
2. Select **Datasets**.
3. Select your liveSource.
4. From the Actions menu (...) select **Disable**.
5. From the Actions menu (...) select **Migrate**.
6. Update the environment, user, and repository. From the **Configuration** tab select the **Source** sub-tab and click the Pencil (edit) icon next to the Database.
7. Enable the **dSource**.

After the liveSource is migrated to a different staging environment, you must ensure that the log transport between the source database and the liveSource instance on the new staging environment is set up correctly.

Oracle RAC dSource node addition or deletion

Managing Oracle RAC node changes in Delphix

The addition and removal of cluster nodes from an Oracle Cluster must be considered where Delphix has dSources attached to RAC databases running from the same cluster environment or RAC VDB's provisioned out to those cluster nodes. Each scenario needs to be considered in its own right as dSources and VDBs require specific changes be made to accommodate the removal of a cluster node.

The changes required by Delphix and made within Oracle that need to be considered when the Oracle cluster node configuration alters and the process for removing or adding an Oracle cluster node is best demonstrated through an example.

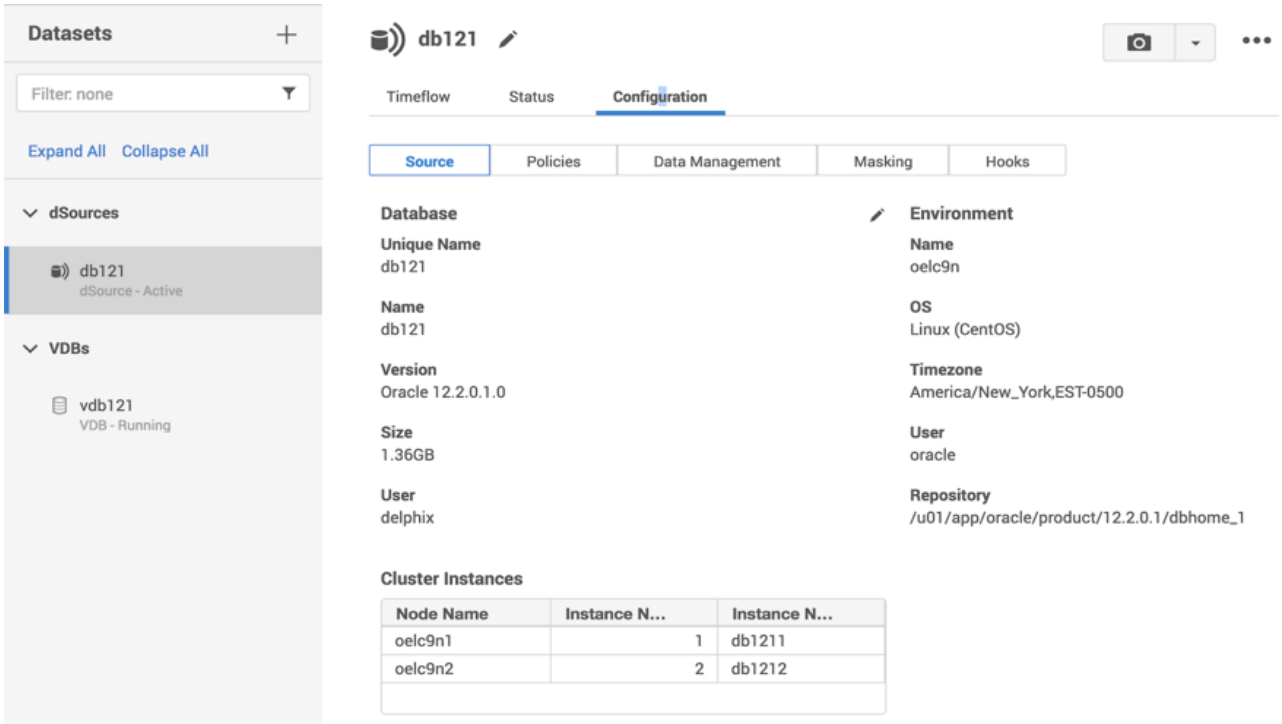
The oracle cluster is comprised of 2 cluster nodes as seen from olsnodes.

```
[oracle@oelc9n1 ~]$ su - grid
Password:
Last login: Thu Nov 30 17:15:36 AEDT 2017 on pts/1

[grid@oelc9n1 ~]$ . oraenv
ORACLE_SID = [grid] ? +ASM1
The Oracle base has been set to /u01/app/grid

[grid@oelc9n1 ~]$  olsnodes -s -t
oelc9n1 Active  Unpinned
oelc9n2 Active  Unpinned
```

Node 2 in the cluster oelc9n2 is the node targeted for removal from the Oracle cluster.

Delphix dSources and removing Oracle cluster nodes

In this example, Delphix has a 2 Node Oracle RAC database being ingested as a dSource. The sample database name is db121 comprised of 2 RAC instances db1211 and db1212. The instance impacted by the removal of node oelc1n2 is db1212. The dSource configuration appears as follows:



The cluster node, its associated listeners and instances must be deleted from the cluster through Oracles RAC RDBMS and Clusterware software using the processes described in the Oracle documentation.

This begins with the removal of the RAC instance using the process detailed in the Oracle RAC documentation. This example is operating against an Oracle 12.1 RAC and Cluster and the documentation link this specific release is the following:

Removing RAC instances and homes

https://docs.oracle.com/database/121/RACAD/GUID-CE995C7B-B420-4A4E-B4EA-E7A972277588.htm#RACAD7903

Removal of the RAC instance is typically executed through the database configuration assistant dbca and using the UI provided through dbca the instance db1212 is removed from node oelc9n2. After the instance has been deleted using dbca, Oracles srvctl utility will show one instance only (db1211) running from node oelc9n1.

```
[oracle@oelc9n1 ~]$ srvctl config database -d db121

Database unique name: db121
Database name: db121
Oracle home: /u01/app/oracle/12.1
Oracle user: oracle
Spfile: +DATA/DB121/PARAMETERFILE/spfile.289.961441955
Password file: +DATA/DB121/PASSWORD/pwddb121.277.961441737
Domain:
Start options: open
Stop options: immediate
Database role: PRIMARY
Management policy: AUTOMATIC
Server pools:
Disk Groups: FRA,DATA
Mount point paths:
```

```
Services:
Type: RAC
Start concurrency:
Stop concurrency:
OSDBA group: dba
OSOPER group:
Database instances: db1211
Configured nodes: oelc9n1
Database is administrator managed
```

Following the removal of the instance, the inventory on the node to be removed is updated to reflect the removal of the Oracle Home that was running this dSource instance on the node. This too must be performed according to the Oracle documentation. Oracles utility runInstaller is used for Oracle inventory maintenance. This must be performed using the operating system owner of the RDBMS Home (in this case oracle)

With the RAC instance and RAC Home removed the node itself can be removed using the process detailed in Oracles Clusterware Administration documentation,

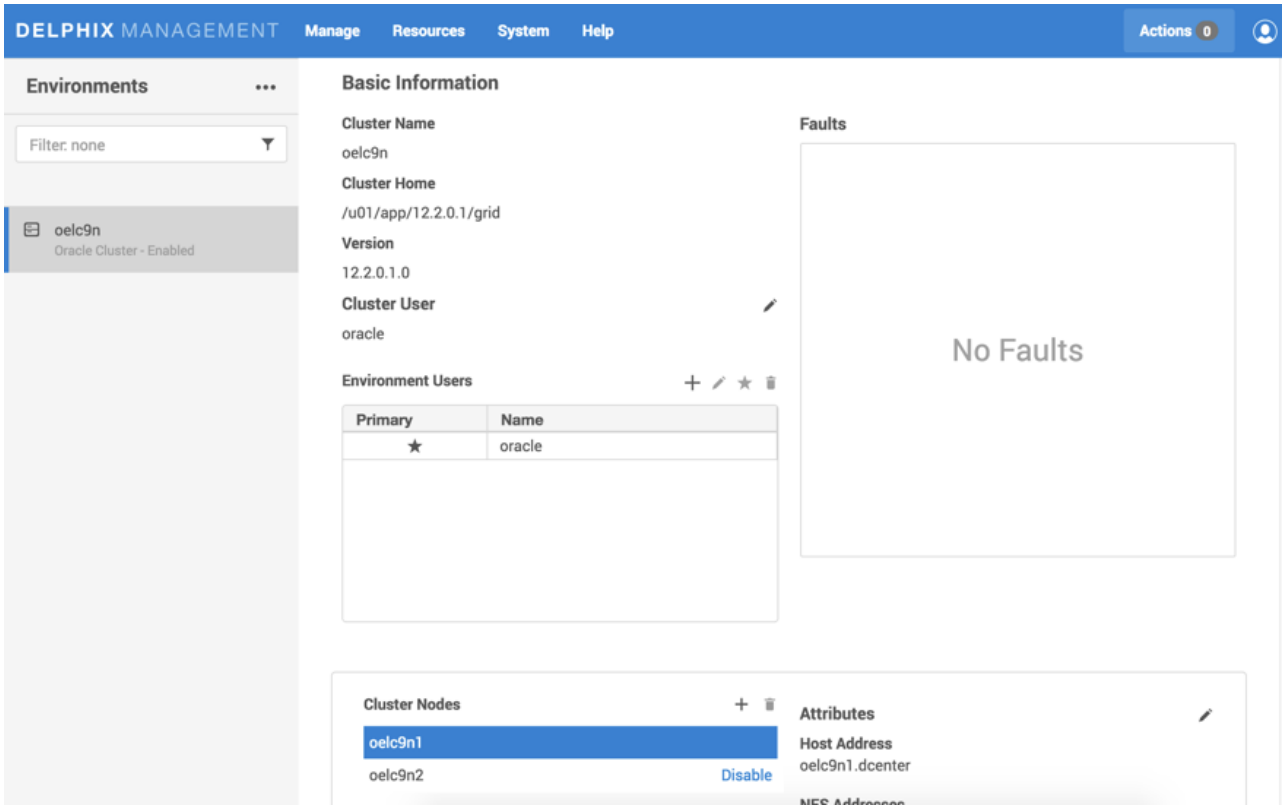
Removing Oracle cluster nodes

https://docs.oracle.com/database/121/CWADD/GUID-8ADA9667-EC27-4EF9-9F34-C8F65A757F2A.htm#CWADD90992

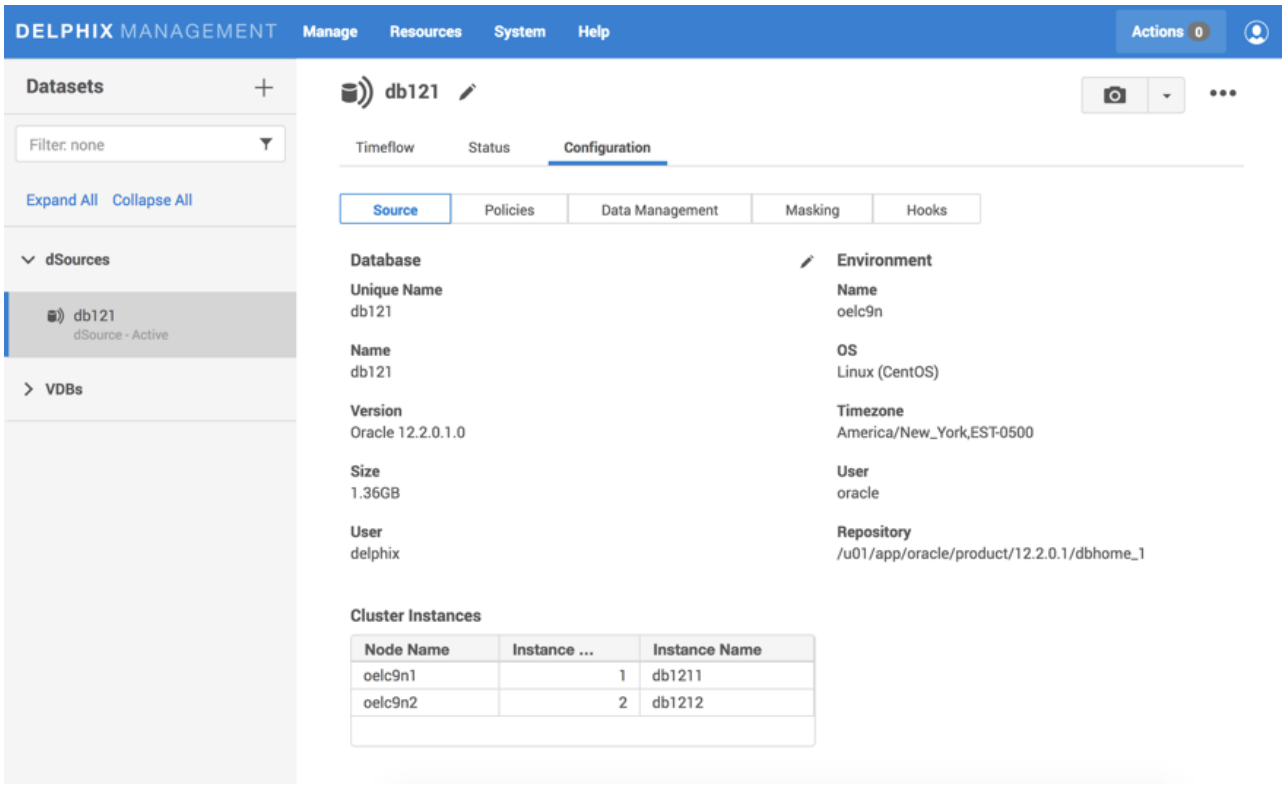
This must be performed as the Oracle Clusterware home operating system owner (in this case grid).

Removal of the node from within Oracles RDBMS and Clusterware software does not alter the fact that Delphix still sees the Oracle Cluster and RAC configuration as it was the last time an environment refresh was performed against the cluster.

The Environment in Delphix shows both cluster nodes are still present along with any node listeners that may have been discovered running on that node.



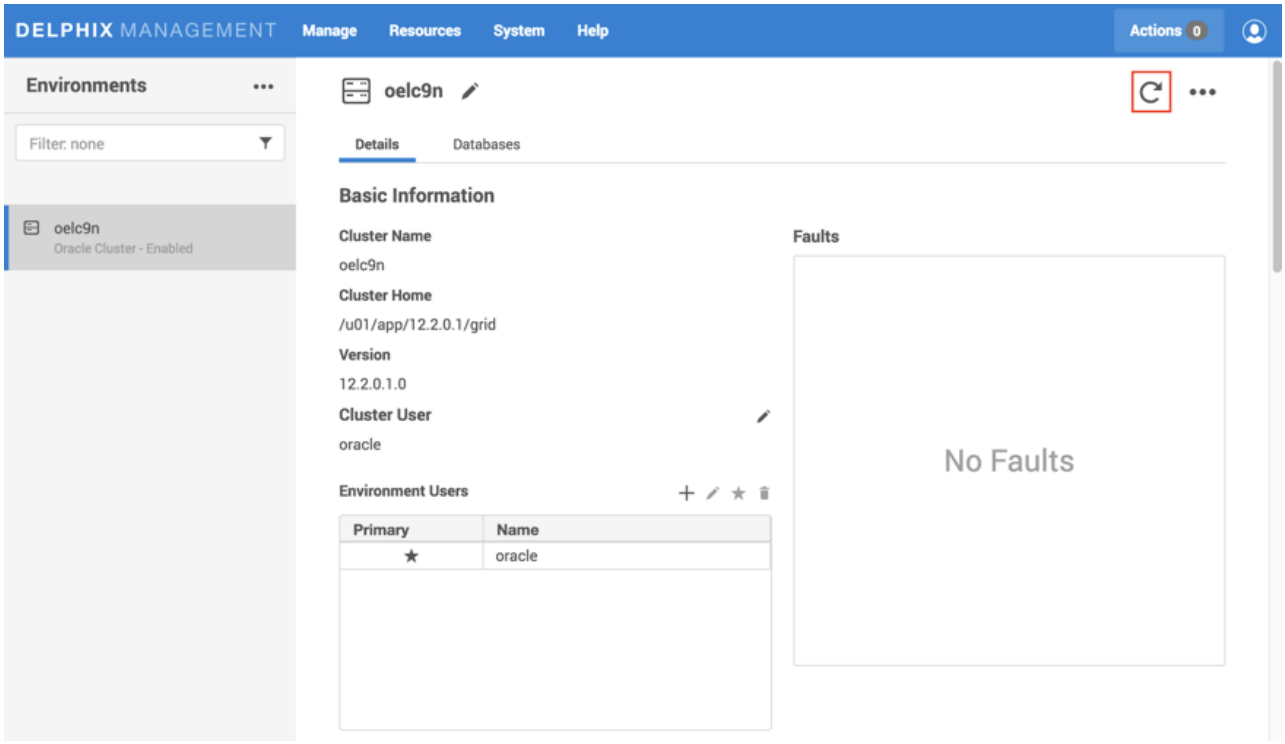
The dSource shows the node and both instances still present this needs to be cleaned up from a Delphix perspective.



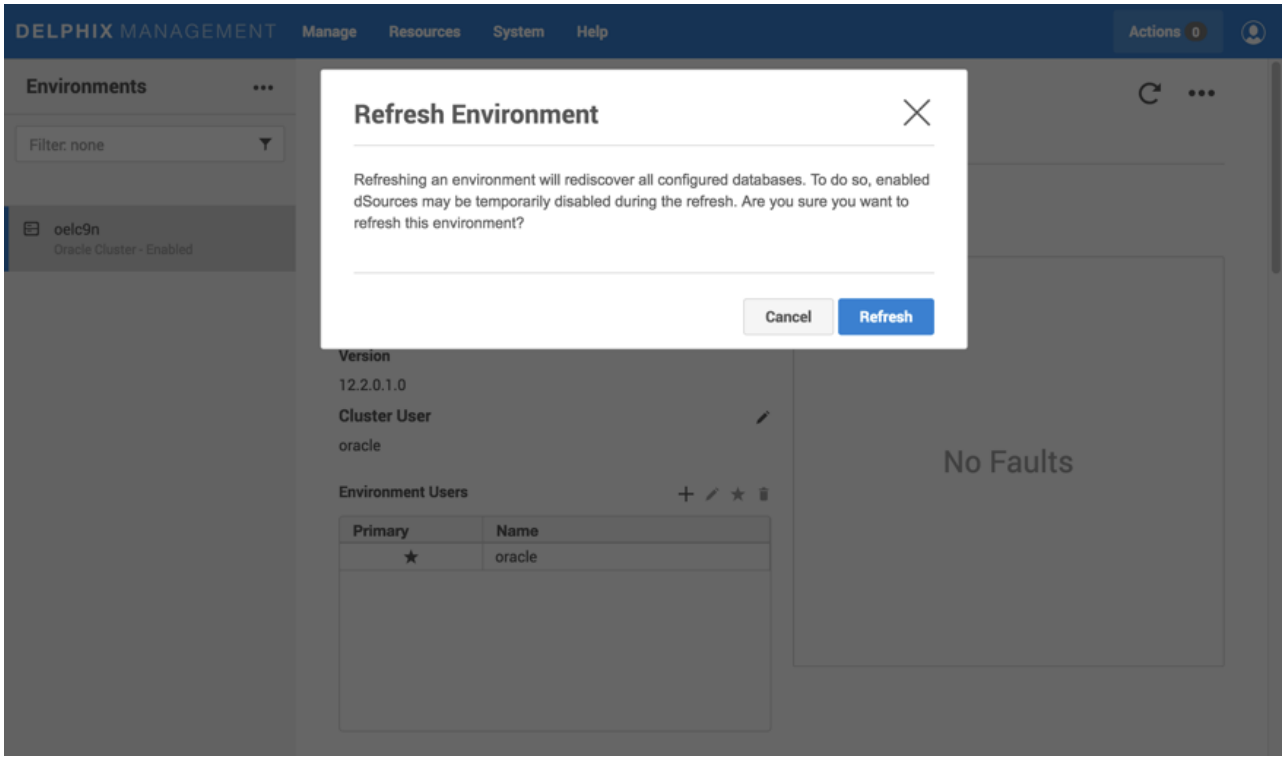
Aligning Delphix with the new Oracle cluster configuration

In order for Delphix to pick up on the latest changes to the environment (OELC9) in this case, the removal of an entire node from the Oracle Cluster an environment refresh must be performed.

This is performed through the environments management panel in the Delphix Management application and the refresh icon present on the top right-hand side of the panel (highlighted in red).



Clicking on the icon presents a prompt indicating that an environment refresh is about to be initiated.

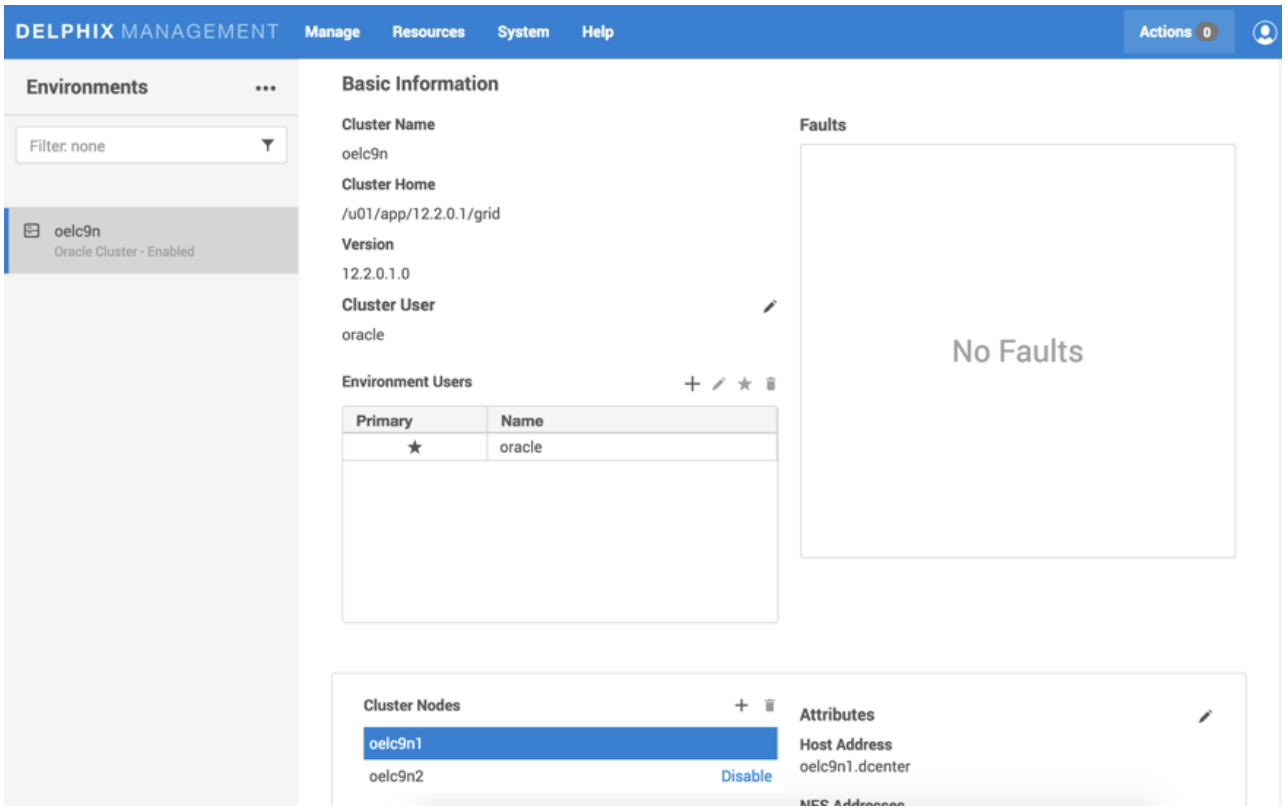


Click on refresh to kick off the environment refresh action.

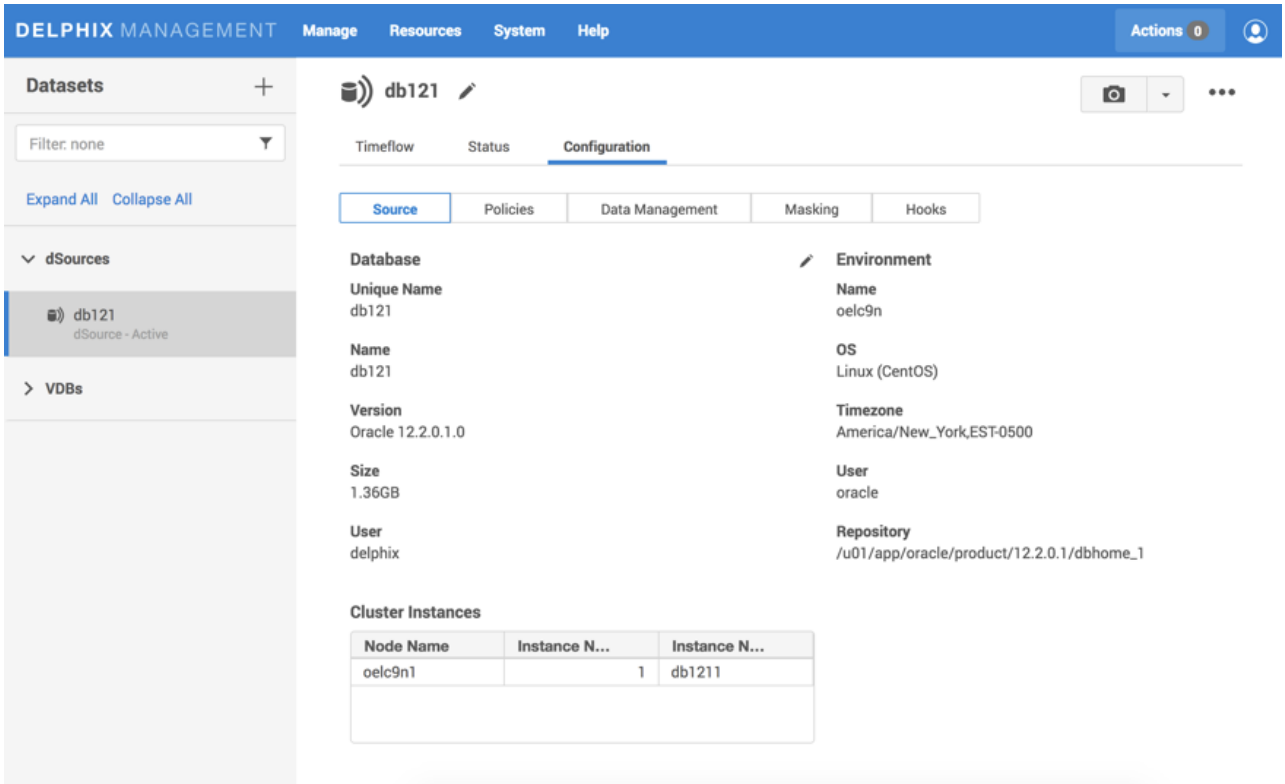
The environment refresh will force Delphix to re-examine the Oracle Cluster.

This will result in the environment and the dSource being updated based on the cluster's current node, instance and listener configuration where only one node oelc9n1 is present.

After the node removal, the cluster environment in Delphix will look as follows and only oelc9n1 appears in the cluster node list.

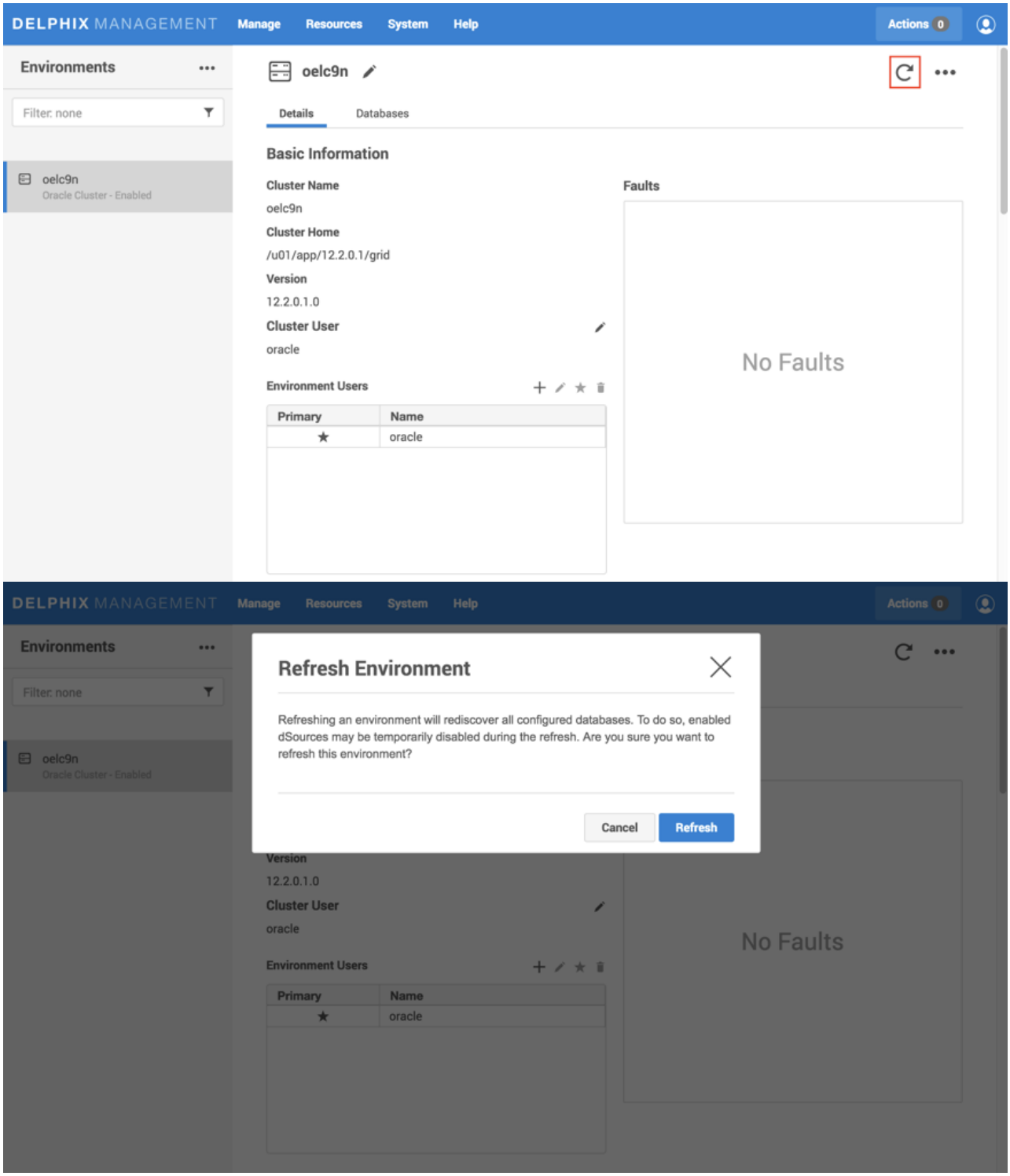


The dSource will now be comprised of only one instance db1211 running from node oelc9n1.



Adding Oracle cluster nodes

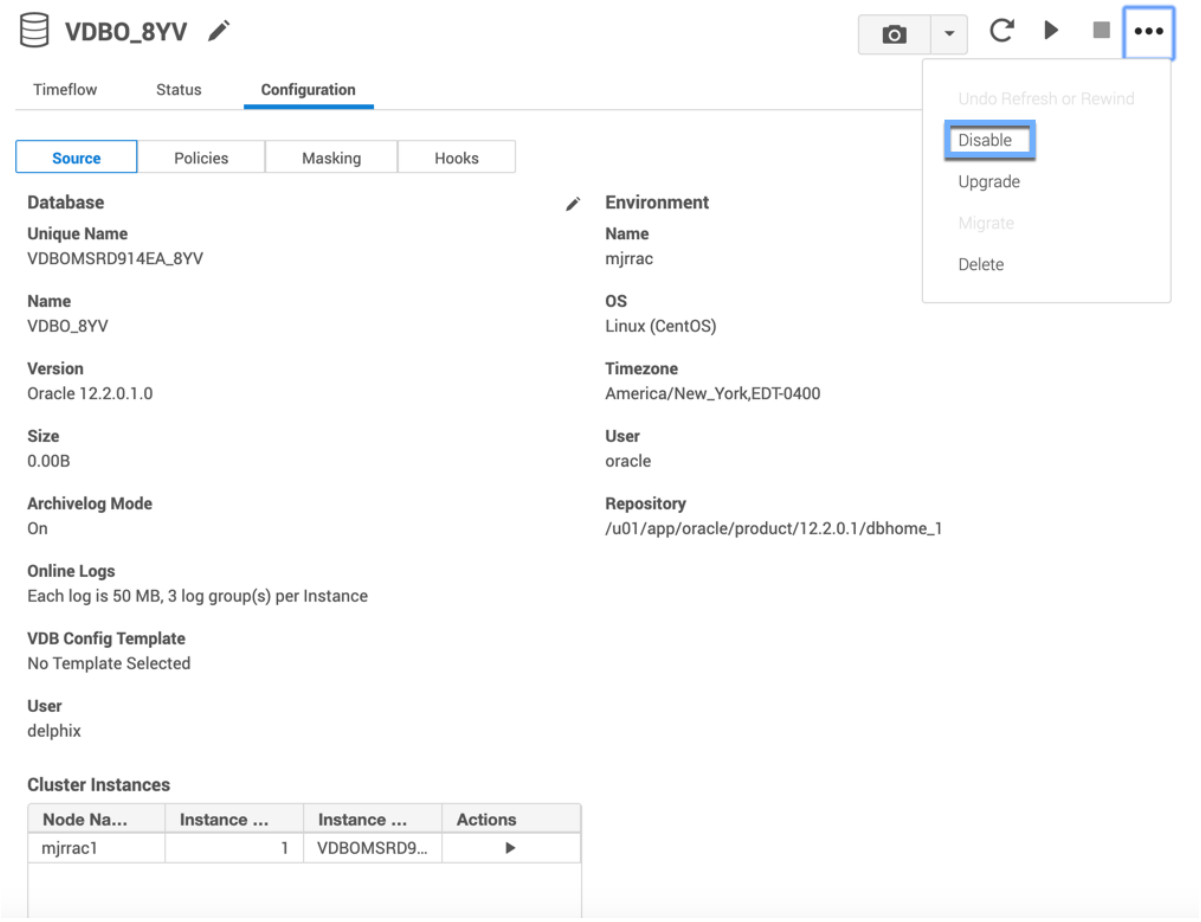
You can add a node through the **Environments** management panel in the Delphix Management application and the **Refresh** icon present on the top right-hand side of the panel. Clicking on the icon presents a prompt indicating that an environment refresh is about to be initiated.



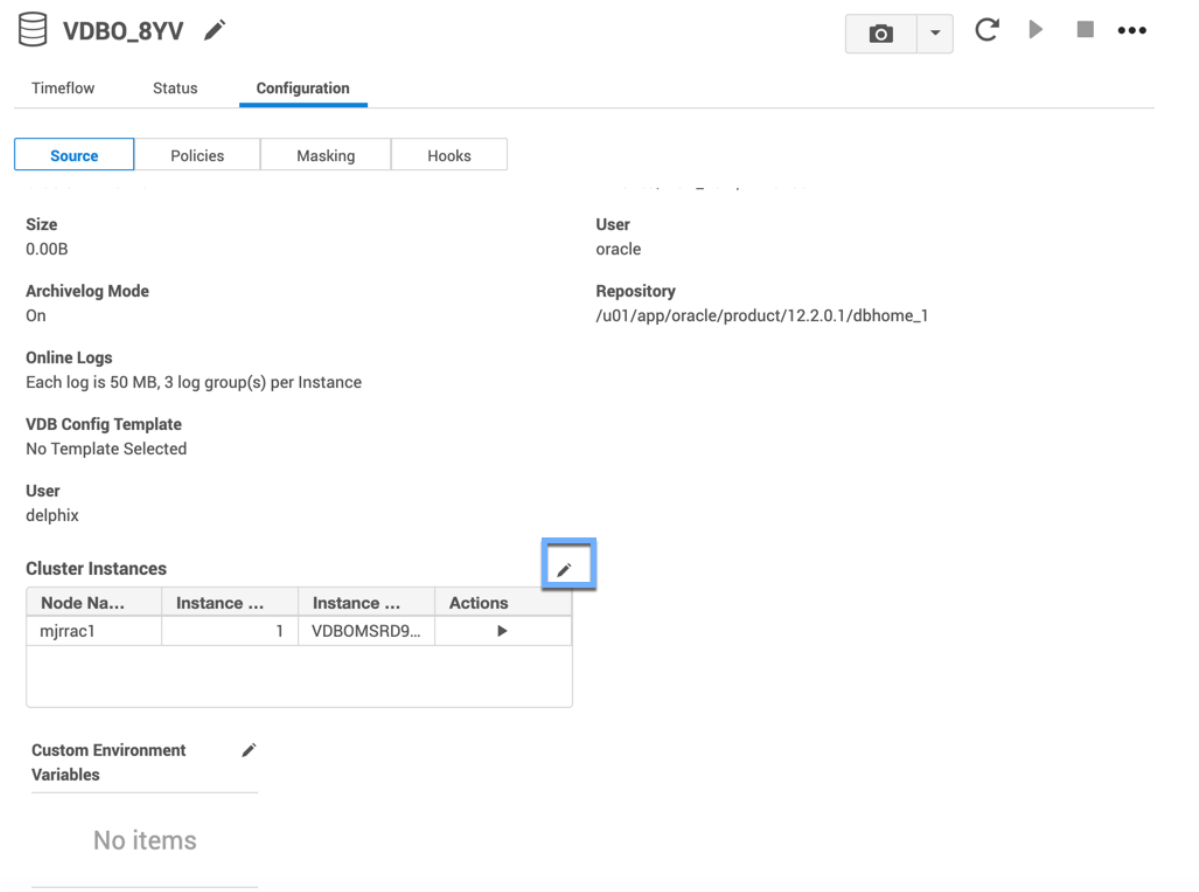
Click on the **Refresh** option to kick off the environment refresh action. The environment refresh will force Delphix to re-examine the Oracle Cluster. This will result in the environment and the dSource being updated based on the cluster's current node, instance, and listener configuration.

Delphix configuration steps to assign a VDB to a new cluster instance

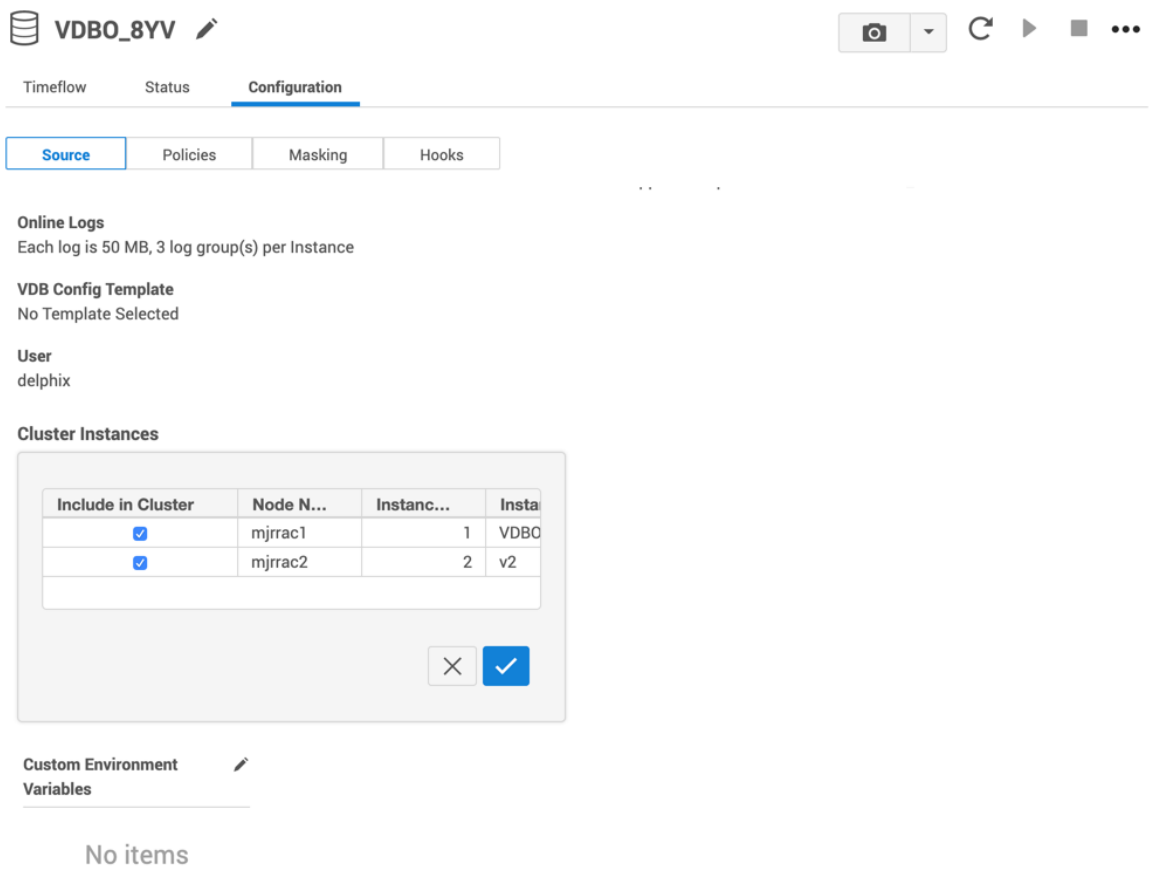
1. Login to the Delphix Management application.
2. In the Oracle RDBMS, add the required number of nodes using the process detailed in the Oracle RAC documentation
3. In the Delphix Management Engine, select the VDB you want to edit.
4. From the right-side pane select the **Configuration** tab.
5. Refresh the VDB to view new nodes.
6. In the upper right-hand corner, from the **Actions** menu (...) select **Disable.**



7. In the **Disable** dialog select **Disable.**
8. Click the **pencil** icon next to **Cluster Instances** to be able to add new nodes.



9. In the **Include in Cluster** column, check or uncheck the check-box to the left to add or remove a node for your RAC VDB.



10. Complete editing and then confirm using the **tick** icon.
11. From the Action menu (...), select **Enable.**

Using the double sync option for Oracle snapSync

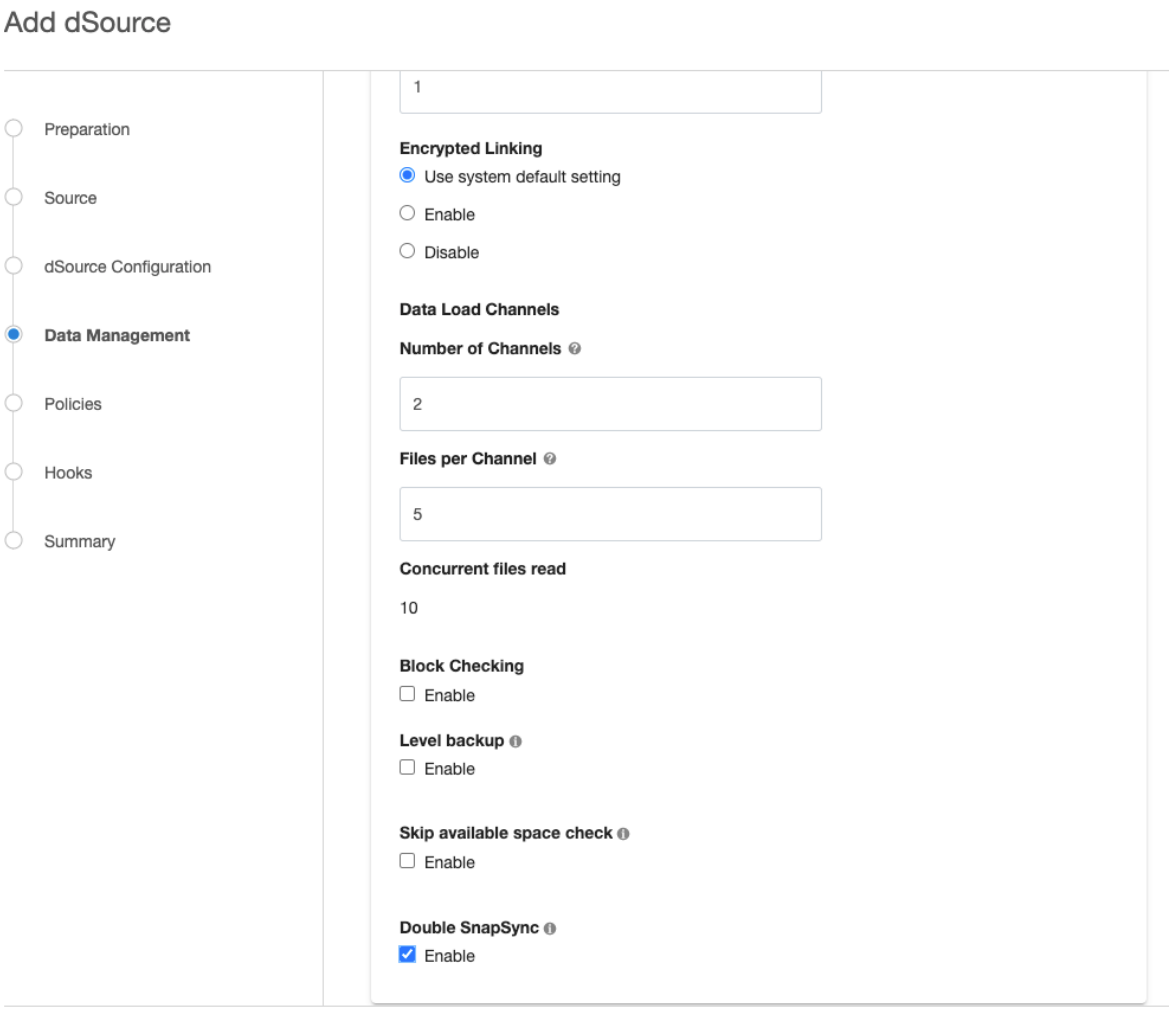
It is possible to request during Linking that a `doubleSync` is performed if `linkNow` is set to true. *(This is available in the GUI for linking starting in 5.0, and for manual Snapshots in 5.3.4.0, see notes below a CLI example).*

When the *doubleSync* option is selected, two SnapSyncs in a row are run, the first one will not even bother to get the archived logs to make the first Snapshot consistent. This means that LogSync will not run during the first SnapSync, or if LogSync is not enabled, the archived logs after the backup of the data will not be retrieved. Once the first snapSyncfinishes a second snapSyncis started, starting LogSync if enabled, or taking an archived log backup after the data files have been backed up.

The doubleSync option is also available for manually initiated SnapSync, just like `forceFullBackup` option. As a `forceFullBackup` is similar to an initial load, `doubleSync` makes a great companion to `forceFullBackup`, although it is not needed to use both at the same time.

Using the double snapSync option via the GUI

1. Login to the **Delphix Management** application.
2. Navigate to the Environment with a Data Source you want to link. Or, from the Datasets page, click the plus icon and select **Add dSource**.
3. In the Add dSource wizard, select the source database with the correct environment user-specified.
4. Enter your login credentials for the source database and click **Verify Credentials**. If you are linking a mounted standby, see the topics under Linking Oracle Physical Standby Databases for more information about how the Delphix Engine uses non-SYS login credentials. Click **Next**.
5. Enter a name for your dSource.
6. Select a **Database Group** for the dSource. Adding a dSource to a database group lets you set Delphix Domain user permissions for that database and its objects, such as snapshots. See the topics under Users and Groups for more information.
7. In the **Data Management** page select **Show Advanced** and then select **Enable Double SnapSync**. For more information, visit Data Management Settings for Oracle Data Sources



8. Assign existing policies to the new dSource. New policies can be created and associated later.
9. Enter any scripts that should be run using the Hooks page.
10. Review the dSource Configuration and Data Management information, and then click Submit.

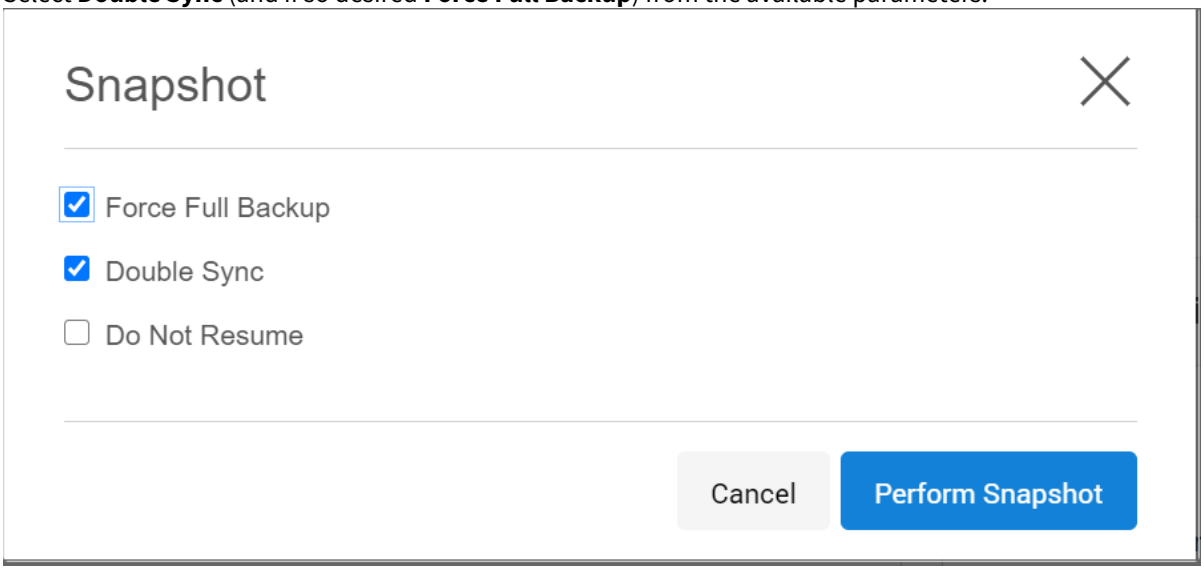
Linking with doubleSync via the CLI

```
ssh delphix_admin@your engine
delphix > database
delphix database > link
delphix database link*> edit source.operations
delphix database link*> edit postSync
delphix database link*> add
delphix database link*> set command=""
delphix database link*> back; back
delphix database link*> edit preSync
delphix database link*> add
delphix database link*> set command=""
delphix database link*> back; back; back
delphix database link*> set source.config=XXXX
```

```
delphix database link*> set container.name=XXXX
delphix database link*> set container.group=XXXX
delphix database link*> set container.sourcingPolicy.logsyncEnabled=true
delphix database link*> set container.sourcingPolicy.logsyncMode=ARCHIVE_REDO_MODE
delphix database link*> set linkNow=true
delphix database link*> set doubleSync=true
delphix database link*> set dbUser=XXXX
delphix database link*> set environmentUser=XXXX
delphix database link*> set dbCredentials.password=XXXX
delphix database link*>commit
```

Syncing with doubleSync via the GUI

1. Login to the **Delphix Management** application.
2. Navigate to the Datasets page, select the Dataset you want to SnapSync.
3. Click on the arrow next to the camera button.
4. Select **Snapshot with Params.**
5. Select **Double Sync** (and if so desired **Force Full Backup**) from the available parameters.



6. Select **Perform Snapshot.**

Syncing with doubleSync via CLI

```
ssh delphix_admin@yourengine
delphix > database
delphix database > select XXXX
delphix database "XXXX"> sync
delphix database "XXXX"sync *> set doubleSync=true
delphix database "XXXX"sync *>commit
```

Additional information

When two Double Sync snapshots are created, then the older snapshot will not be provisionable. Therefore, you should not attempt time flow repair operation as the logs required to make it provisionable, were not retrieved by design. Use the new snapshot for provisioning or refreshing operation.

DoubleSync is not available for multitenant databases.

Linking dSources from an encrypted Oracle database

This topic describes the behavior of the Delphix Engine when linking to a dSource based on an encrypted Oracle database.

> ⚠ This topic does not apply to vPDBs.

Beginning with version 10gR2, Oracle supports the encryption of permanent tablespaces using Transparent Data Encryption (TDE). You can link dSources from databases using TDE by following the basic procedure described in Linking an Oracle Data Source. However, in order to provision a VDB from a dSource that is linked to an encrypted database, you must copy wallet files from the physical database in the source environment to the target environment. See Provisioning a VDB from an Encrypted Oracle Database for more information.

Detaching and re-attaching Oracle dSources

Overview

Each dSource contains metadata that associates it with the source database, as well as the data it has ingested from the source database in the form of snapshots up to that point. It is possible to detach, or unlink, a dSource from its source database. This breaks the association with the source database without affecting the data within the Delphix Engine. Detached dSources and their source databases have these properties:

- A detached dSources can still be used to provision a virtual database (VDB).
- You can re-link the source database as a different dSource.

Prerequisites

Before detaching an Oracle dSource, you must capture the following RMAN configuration:

- Level or SCN based backups
- Data load channel settings: number of channels and files per channel

A dSource can only be attached to a new data source once it has been unlinked.The above RMAN configuration will be removed once the dSource is unlinked.

When attaching an Oracle dSource to a new data source, the new data source must be the same logical database satisfying the following constraints:

- Same dbid
- Same dbname
- Same creation time
- Same resetlogs SCN
- Same resetlogs time
- Same redo stream, where a log must exist with
    - Same sequence
    - Same thread
    - Same end SCN

For Oracle dSources, this procedure can be used to initially link from a standby server that is faster or less disruptive, unlink the dSource, and then attach it to the production server for subsequent incremental SnapSync operations. When you perform the attach operation, you will need the source config name of an unlinked database.

> ⚠ It is possible to link a Single Instance Oracle database, unlink from it, and link to a RAC cluster and vice versa as long as the instances are for the same database. For example, initially link a Single Instance standby database of a RAC primary database, unlink the dSource, and then link the RAC primary database itself or another RAC standby database to the same dSource.

Detaching a dSource

1. Login to the **Delphix Management** application as a user with **OWNER** privileges on the dSource, group, or domain.
2. Click **Manage**.
3. Select **Databases**.
4. Select the **database** you want to unlink or delete.
5. From the **Actions** menu (...) select **Unlink.** A warning message will appear.
6. Click **Unlink** to confirm.

To detach a dSource via CLI, see Detaching and Attaching an Oracle dSource via CLI

Rebuilding source databases and using VDBs
In situations where you want to rebuild a source database but retain the existing dSource, you will need to detach the original dSource and create a new one from the rebuilt data source.

1. Detach the dSource as described in the procedure on this page.
2. You cannot attach a dSource with the same name as a dSource that is already attached. If you intend to give the new dSource the same name as the original one, rename the detached dSource.
    a. At the top of the **Configuration** tab, next to the dSource's name, click the **Edit** (pencil) icon.
    b. After renaming the dSource, click the green **checkmark**.
3. Create the new dSource from the rebuilt database.

You will now be able to provision VDBs from both the detached dSource and the newly created one, but the detached dSource will only represent the state of the source database prior to being detached.

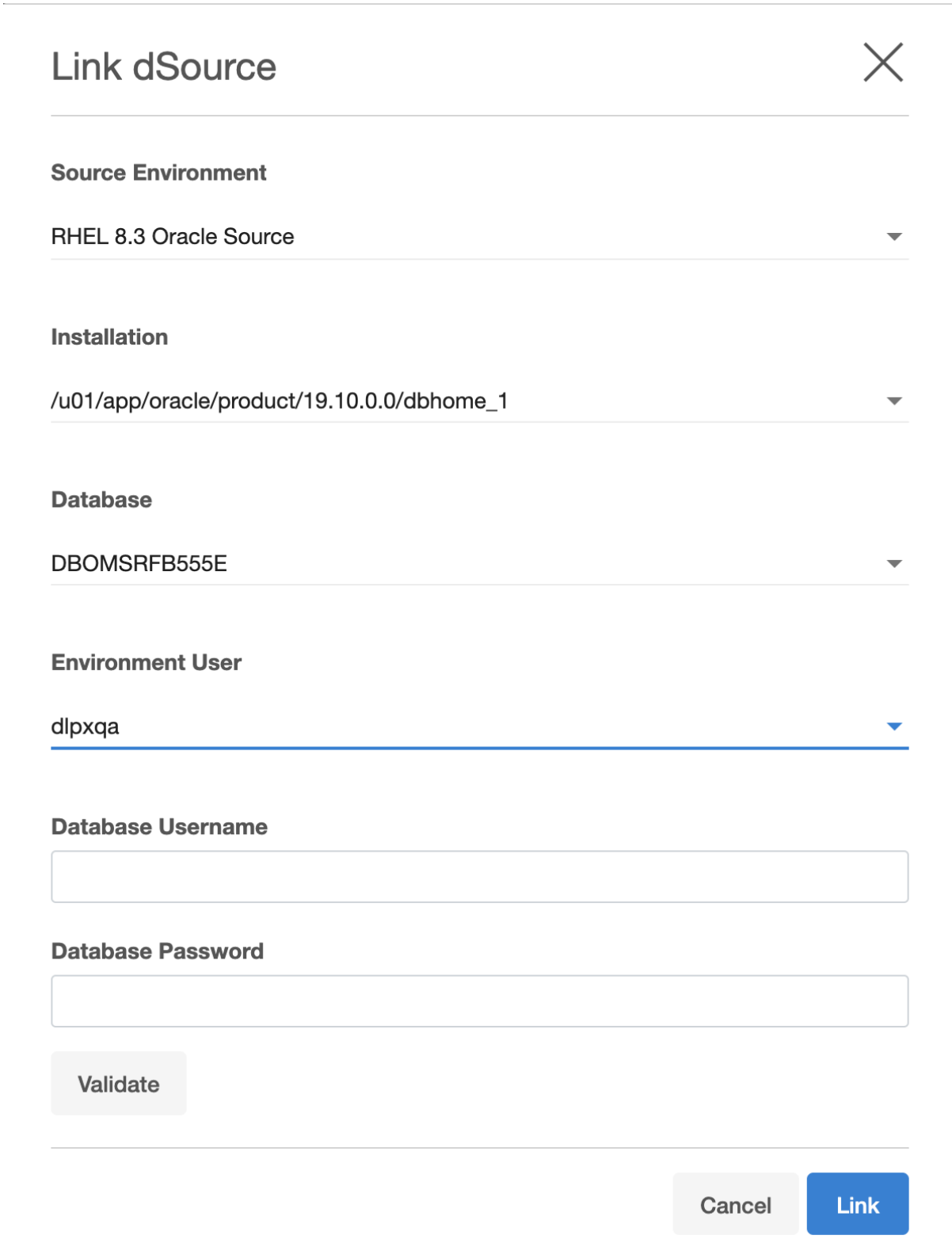
Attaching a previously detached dSource

You can only re-attach databases that represent the same physical database.

Via GUI

To attach a dSource back into the Delphix Engine via GUI, from the **Delphix Management** application:

1. Select **Manage > Datasets**.
2. Select your dSource.
3. From the **Actions** menu (...) select **Link dSource**.
4. In the Link dSource dialog, fill in the information corresponding to the new dSource as well as its new environment.



5. Click **Link**.

Via CLI

To attach a dSource back into the Delphix Engine via CLI:

1. Login to the **Delphix CLI** as delphix_admin or a user with OWNER privileges on the dSource, group, or domain.
2. Select the dSource by name using `database select`.
3. Run the `attachSource` command.
4. Set the source config you want to attach to, using `set attachData.config=`. Source configs are named by their database unique name.
5. Set any other source configuration operations as you would for a normal link operation.
6. Run the `commit` command.

Backup mode for sttaching Oracle dSources
For Oracle dSources, the SnapSync backup option should be set to **SCN Backup** mode. **Level Backup** mode is based on information stored in the database control file. If the control file of the newly attached database does not contain information about the previous backups, an initial backup will be created. In addition, Block Change Tracking will not be in sync, and the next SnapSync will need to read the entire database to determine which blocks have changed. See Advanced Data Management Settings for Oracle dSources for more information about Backup Mode.

Provisioning from a replicated Oracle dSource

This topic describes how to provision from a replicated dSource or virtual database (VDB).

The process for provisioning from replicated objects is the same as the typical VDB provisioning process, except that first you need to select the replica namespace containing the replicated object.

Prerequisites

- You must have a dSource or VDB that has been replicated from one Delphix Engine to another, as described in Replication Overview
- The Delphix Engine containing the replicated dSource or VDB must have a compatible target environment that it can use to provision a VDB from the replicated dSource or VDB.

Procedure

1. On the Delphix Engine containing the replicated dSource or VDB, login to the **Delphix Management** application.
2. In the top menu bar, click **Manage**.
3. Select **Datasets**.
4. From the list of replica namespaces, select the **replica namespace** that contains the dSource or VDB from which you want to provision.
5. The provisioning process is now identical to the process for provisioning standard objects.

Post-requisites

Once the provisioning job has started, the user interface will automatically display the new VDB in the live system.

Detaching and re-attaching a PDB dSource from a data guard site

If you want to move the PDB dSource from one Data Guard site to another, then you must detach the PDB from the current environment and attach it to the new environment.

Perform the following procedure to attach or detach a PDB dSource from a Data Guard site.

1. Detach PDB from CDB (a).
    a. Log into the **Delphix Management** application.
    b. Select **Manage > Datasets.**
    c. Select dSource.
    d. From the **Actions** menu (...), select **Unlink dSource**.
    e. Click **Unlink** to confirm.
2. If you are going to attach the PDB dSource from a different Data Guard site (Primary or Standby site), perform a fresh refresh of the existing environment to discover that host environment into Delphix. From the **Delphix Management** application:
    a. Go to **Manage > Environments**.
    b. If it is already a discovered environment, then from the **Actions(...)** menu, select **Refresh All**. Otherwise, select **Add Environment** to add the environment manually. For more information on adding an environment, see Adding an Oracle Single Instance or RAC Environment. The new database configuration will appear in the Delphix Engine.

> Do not add the discovered PDB as a dSource.

3. Detach the existing PDB dSource.

- Select the dSource.
- From the **Actions** menu (...), select **Unlink dSource**.
- Click **Unlink** to confirm.
- Using the Delphix CLI, detach the current CDB datasets associated with the same PDB dSource.

```
dlpx6010 database> select cdb19c1
dlpx6010 database 'cdb19c1'> detachSource
dlpx6010 database 'cdb19c1' detachSource *> ls
Properties
    type: DetachSourceParameters
    source: (required)
dlpx6010 database 'cdb19c1' detachSource *> set source=cdb19c1
dlpx6010 database 'cdb19c1' detachSource *> commit
    Dispatched job JOB-16
    DB_DETACH_SOURCE job started for "SugoPronto/cdb19c1".
    DB_DETACH_SOURCE job for "SugoPronto/cdb19c1" completed successfully.
```

- Using the Delphix CLI, attach the new CDB configuration from the new environment/Physical Standby site.

```
dlpx6010 *> database
dlpx6010 database *> select cdb19c1
dlpx6010 database 'cdb19c1'*> attachSource
dlpx6010 database 'cdb19c1' attachSource *> set
attachData.type=OracleAttachData
dlpx6010 database 'cdb19c1' attachSource *> set attachData.config=cdb19c1stb
```

```
dlpx6010 database 'cdb19c1' attachSource *> set
attachData.environmentUser=OEL7SITDE2/delphix
dlpx6010 database 'cdb19c1' attachSource *> commit;
    cdb19c1
    Dispatched job JOB-18
    DB_ATTACH_SOURCE job started for "SugoPronto/cdb19c1".
    Starting validation of attach parameters for database "cdb19c1".
    Validation finished for database "cdb19c1".
    Obtaining information from source database "SugoPronto/cdb19c1".
    The dSource "cdb19c1stb" was successfully linked from source database
"SugoPronto/cdb19c1".
    DB_ATTACH_SOURCE job for "SugoPronto/cdb19c1" completed successfully.
```

- Using the Delphix Engine user interface, attach the PDB.
    a. Go to **Manage > Datasets**.
    b. Select the PDB from the datasets list.
    c. From the **Actions(...)** menu, select **Link dSource**. The Link dSource page appears.
    d. Perform the linking of the PDB from the new environment and click **Next**.

Working with Oracle snapshots

This section lists the steps to take a snapshot and delete the same.

Taking a snapshot creates a new snapshot entry in the Oracle dSource's Timeflow. You can use either **snapshot (Default)** or **snapshot with Parameters** option for taking the snapshot.

snapshot (Default)

Perform the following steps to take a snapshot:

1. Login to the **Delphix Management** application.
2. Click **Manage** and select **Datasets** from the dropdown list.
3. Select the dSource you want to snapshot.
4. Click the **Camera** icon. Alternatively, click the arrow next to the Camera icon and select **snapshot (default)**.
5. From the snapshot dialog box, select **Yes**.
6. Navigate to the **Timeflow** tab and click **View: All snapshots** to verify the snapshot you just created.
7. To delete the snapshot, select the snapshot you just created, and from the Actions menu (...), select **Delete snapshot**.
8. From the **Delete snapshot** dialog box, select **Delete**.
9. Navigate to the **Timeflow** tab and click **View: All snapshots** to verify the snapshot you just deleted.

snapshot with parameters

Perform the following steps to take a snapshot:

1. Login to the **Delphix Management** application.
2. Click **Manage** and select **Datasets** from the dropdown list.
3. Select the dSource you want to snapshot.
4. Click the arrow next to the Camera icon and select the **snapshot with Params...** option.
5. From the snapshot dialog box, select one of the following:
    a. **Force full backup** - If you select this option, then the Delphix Engine will perform an incremental backup by default. You must select this option only when a full backup is required. Full and Incremental backups consume the same space on the Delphix Engine.
    b. **Double sync** - Selecting this option will perform a SnapSync operation as normal. After the first SnapSync is successful, the Engine will immediately perform a second SnapSync without waiting for the Log Files required for the first SnapSync to be made consistent. This is most useful when performing the initial SnapSync (or when "Force Full Backup" is selected) on a very large database that would lead to a large number of archive logs being required to make the SnapSync consistent. Provisioning from a SnapSync that requires excessive recovery is typically time-consuming.
    c. **Do not resume** - If a failure is encountered during the initial SnapSync, the Delphix Engine can resume the SnapSync at a later date. This option will cause the engine to not resume, but rather to start the initial SnapSync over again.
6. Navigate to the **Timeflow** tab and click **View: All snapshots** to verify the snapshot you just created.
7. To delete the snapshot, select the snapshot you just created, and from the Actions menu (...), select **Delete snapshot**.
8. From the **Delete snapshot** dialog box, select **Delete**.
9. Navigate to the **Timeflow** tab and click **View: All snapshots** to verify the snapshot you just deleted.

Moving the PDB to a new CDB

Perform the following procedure to move the dSource PDB to a new CDB.

1. Follow Oracle steps to unplug your dSource PDB and plug it into your chosen CDB (Refer to the Oracle webpage on Plugging an Unplugged Pluggable Database). After you have plugged your PDB dSource into the new CDB, discover the host environment that has the new CDB into Delphix, if it is not already discovered. Log into the **Delphix Management** application and do the following:
    a. Select **Manage > Environments**.
    b. Select your new CDB environment, then discover the new CDB where your PDB dSource is.
2. If you have already discovered the CDB, then:
    a. Select **Manage > Environments**.
    b. Select the dSource Environment where the PDB was plugged.
    c. **Select the Refresh** icon.
3. When you are ready to attach the PDB dSource, log into the **Delphix Management**application, and do the following:
    a. Select **Manage > Environments**.
    b. Select your original PDB dSource Environment.
    c. **Select the Refresh** icon.
4. To Attach the PDB dSource back into the Delphix Engine, do the following:
    a. Select **Manage > Datasets**.
    b. Select your PDB dSource.
    c. From the **Actions** menu (...), select **Link dSource**.
    d. In the Link dSource dialog, fill in the information corresponding to the new PDB dSource as well as its new environment:
    
    e. Click **link**.

Provisioning and managing virtual databases with Oracle

Each VDB has its own data management settings, found during the provisioning workflow as well as in the configuration page for that VDB. When you create a VDB, the Delphix Engine copies configuration settings from the dSource and uses them to create the VDB.

This section covers the following topics:

Overview of provisioning Oracle virtual databases

Virtual databases are a key data management concept for Delphix. In order to create or provision a virtual database, you will need a linked dSource from a source host and a compatible target environment, as described in the overview for Managing Environments and Hosts and Requirements for Oracle Hosts and Databases

> A Solaris x86 source host is compatible with a Linux x86 target host and vice versa.

From a dSource, you can select a snapshot or point in time to create a VDB. Oracle VDBs each have their own configuration settings as described in Configuration Settings for Oracle Virtual Databases

Procedure

1. In the Datasets panel on the left-hand side, click the group containing the dSource or VDB from which you want to provision.
2. From the **Timeflow** tab, select a snapshot or point in time to provision from.
3. Click to open the **Provision VDB wizard**, and select a compatible Target Environment for the new Oracle VDB
    - Select Provide Privileged Credentials if you want to use login credentials on the target environment that are different from those associated with the Environment User
4. In the **Advanced** section, you may customize the VDB's configuration settings, file mappings, or custom environment variables.
    - For more information, see Configuration Settings for Oracle Virtual Databases
5. Select a **Snapshot Policy** for the VDB.
6. Enable **Masked Provisioning** by selecting an option on the Masking page.
7. Enter any operations that should be run in the Hooks page.
8. Review the VDB Configuration and Summary, and then click **Submit**.

When provisioning starts, you can review the progress of the job by selecting the VDB and clicking on the Status tab, or by selecting System and viewing the Jobs page.

Alternatively, you could see this in the Actions Sidebar. When provisioning is complete, the VDB will be included in the group you designated and listed in the Datasets panel.

Provisioning by snapshot or logSync

When provisioning by snapshot, you can provision to the start of any particular snapshot, either by time or SCN.

Snapshot Options

Snapshot options	**Description**
Provision by Time	You can provision to the start of any snapshot by selecting that snapshot card from the TimeFlow view or by entering a value in the time entry fields below the snapshot cards. The values you enter will snap to the beginning of the nearest snapshot.

Snapshot options	Description
Provision by SCN	To open the SCN entry field select the Actions (...) menu for your selected snapshot and select View by SCN. Here, you can type or paste in the SCN to which you want to provision. After entering a value, it will "snap" to the start of the closest appropriate snapshot.

For more granularity, you can use the logSync options to provision to any point in time, or to any SCN, within a particular snapshot.

logSync Options

logSync options	Description
Provision by Time	Select Open logSync control to view the time range within that snapshot. Select the clock icon and then select Time to the point to the time from which you want to provision. You can also enter a date and time directly.
Provision by SCN	Select View SCN to view the range of SCNs within that snapshot. You can also type or paste in the specific SCN to which you want to provision. Note that if the SCN does not exist, you will see an error when you provision.

Configuration settings for Oracle virtual databases

Each VDB has its own data management settings, found during the provisioning workflow as well as in the configuration page for that VDB. When you create a VDB, the Delphix Engine copies configuration settings from the dSource and uses them to create the VDB.

The following settings are available for Oracle VDBs:

VDB setting	Explanation
Mount base	The directory to which Delphix will provide mounted storage. This is where the contents of this VDB will be located.
VDB configure parameters	These are customizable parameters of the VDB that you can set either in the provided table or in the text field. This concept is described further below.
Open database after provision	Open the VDB immediately after the provision completes. Enabling this option allows you to set the four parameters below: • Online Log Size (MB) • Number of Online Log Groups • Enable Archivelog Mode • Generate new DBID for VDB This setting is recommended to tune the Oracle database settings to match your requirements. Each of these four helps Delphix manage the Oracle database optimally. When provisioning an Oracle VDB with the "Open Database After Provision" option unchecked, the subsequent snapshot job fails because the database is not open. The UI will show that the provision failed, but this is not the case. A VDB provisioned this way will exist with the stop/start enable/disable function, but will need to be manually started on every refresh.
Online log size (MB)	Also known as the online redo log, this specifies the size of the VDB's redo log. We recommend increasing this value to 1024 MB for improved performance.
Number of online log groups	The number of online log groups to be assigned to this VDB. The number of groups will depend on the configuration of your VDB. For more information, please visit Creating redo log groups and members
Enable archivelog mode	When you enable this mode, redo logs will be archived instead of overwritten. This creates a backup of all transactions that have occurred in the database so that you can recover to any point in time. These archive logs are useful for recovering a database or updating a standby database.

VDB setting	Explanation
Generate new DBID for VDB	Create a new Oracle DBID value for this VDB. **Note: You can also toggle this option after VDB is created.** See Generate a new DBID for Oracle VDBs and Toggle new DBID generation upon refresh options for Oracle VDBs
Listeners	Select an Oracle listener from the list provided to use with this VDB.
Auto VDB restart	Enabling this option will automatically restart this VDB whenever its target host is rebooted. Auto VDB Restart can help you recover from host downtime automatically since the Delphix Engine will restart VDBs once they become available.
Configure File mapping	This option allows you to leverage Oracle's File Mapping feature with a VDB. Enabling this will add a step to the provisioning wizard to create file mappings.

Using Oracle Multitenant offers different configuration settings, as described in the table below:

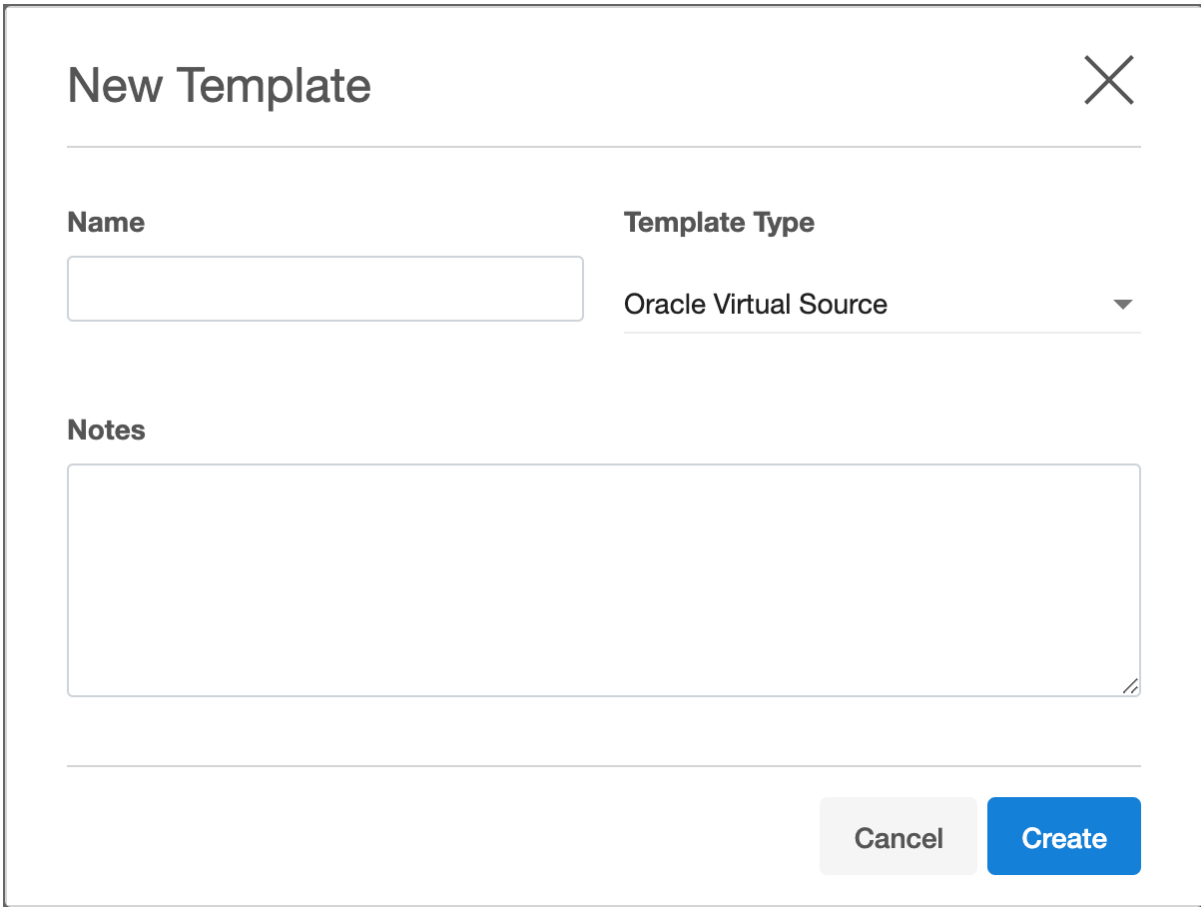
Multitenant setting	Explanation
vCDB database parameters	vCDB Database Parameters are Oracle configuration settings you can apply during the provisioning process. You can specify name-value pairs to apply these settings to the vCDB if you are creating one.
Auto vCDB restart	Enabling this setting will automatically restart this vCDB whenever the target host is rebooted. This will cause all vPDBs in the vCDB to restart as well.

VDB config templates

A VDB config template is a list of database configuration parameter names and values that you can save on the Delphix Engine to use at a later time.

Creating a VDB config template Via GUI

1. Log into the **Delphix Management** application as an engine administrator.
2. Click **Manage**.
3. Select **VDB Config Templates**.
4. Click the icon next to the **VDB Config Temp...** and select **New Template** to create a new template.
5. In the New Template dialog, enter the name for the new template, parameters that you want to provide, and select the source type from the available options.



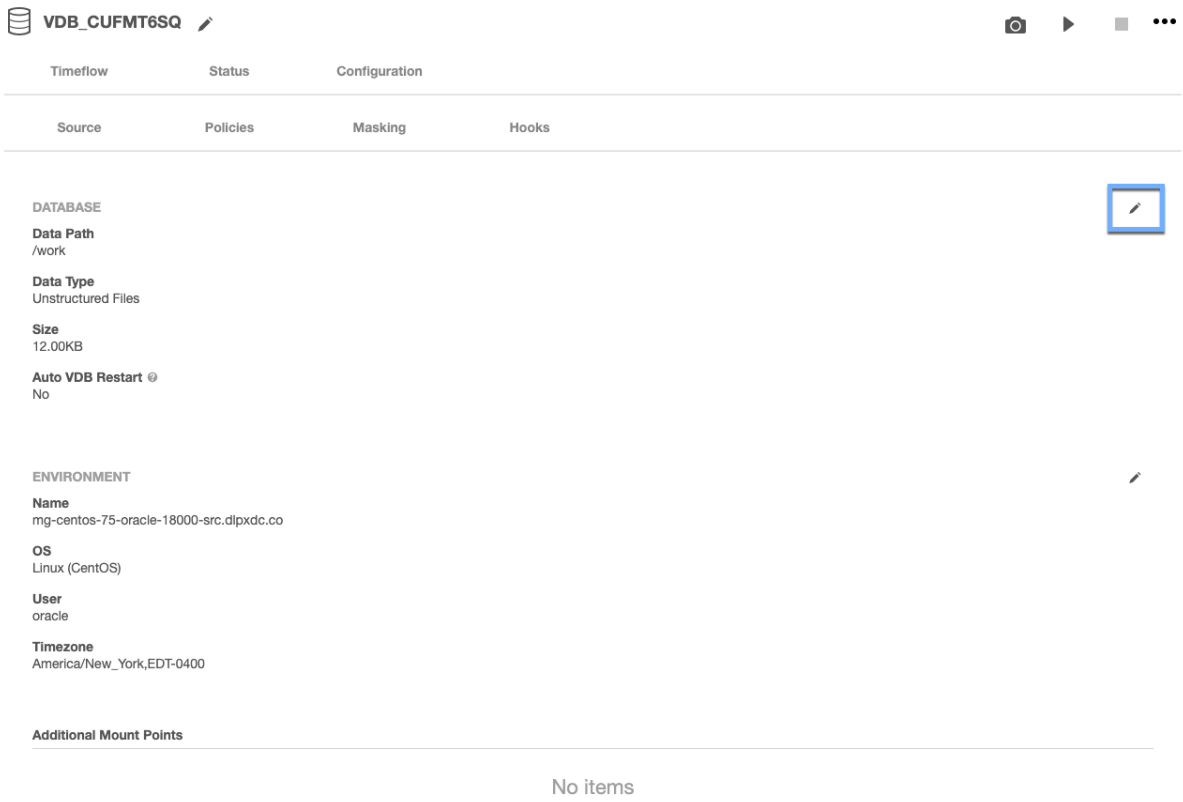
6. Click **Create**.

Updating a VDB config template via GUI

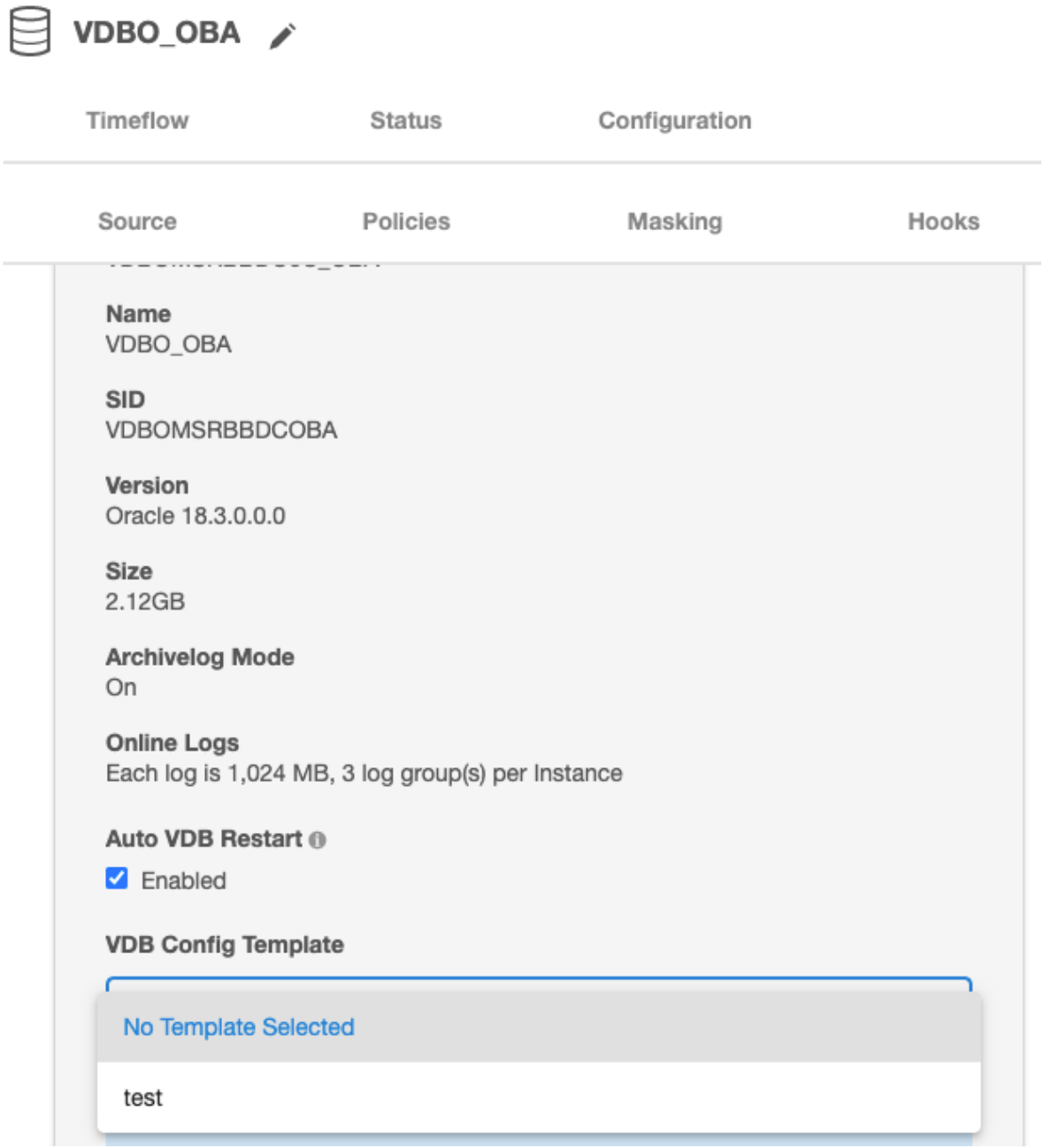
1. Log into the **Delphix Management** application as an engine administrator.
2. Click **Manage**.
3. Select **VDB Config Templates**.
4. Select the template from the left-side pane that you need to update.
5. Click on the **pencil** icons to edit an existing VDB template.
6. Click the button to save the changes or click the button to discard the changes that you made.

Apply/Configure a VDB config template to a VDB Via GUI

1. Login to the **Delphix Management** application as an engine administrator.
2. Click **Manage** > **Datasets.**
3. Select the VDB you want to edit.
4. From the right-side pane select the **Configuration** tab.
5. Click on the **pencil** icon next to 'DATABASE' to be able to edit the database.



6. To apply an already existing VDB template, select from the available template in the drop-down under VDB Config Template.Similarly, you can configure the VDB for an already associated template from the available templates.



7. Click the button to save the changes or click the button to discard the changes that you made.

You can apply a VDB Config Template to a VDB during the provisioning process, which copies the values from the template. Any subsequent changes to the template will be reflected in the VDB when that VDB is refreshed. During provisioning, you can specify configuration parameters directly or copy them from a VDB Configuration Template. Once set, the Delphix Engine will use these parameters whenever the VDB is refreshed, even if you change the original template. It is important to know, however, that some configuration parameters cannot be customized. In addition, some configuration parameters are stripped out during the provisioning process but are customizable. The list of restricted and customizable parameters can be found below.

Required parameters

The VDB configuration template must contain at least the following database parameters:

- **compatible** : Required for both Non-Multitenant and Multitenant sources. Must be set with the same value as in the dSource 'compatible' parameter.
- **enable_pluggable_database** : Required for Multitenant sources only. Must be set to TRUE.

Customizable VDB configure parameters

The default value for these parameters is cleared during the provisioning process. They are removed from the VDB configuration file unless you set values for them through a VDB Config Template.

- audit_file_dest
- audit_sys_operations
- audit_trail
- background_dump_dest
- core_dump_dest
- db_domain
- diagnostic_dest
- dispatchers
- fast_start_mttr_target
- log_archive_dest_n
    - n can be values between 1 - 31
- log_archive_dest_state_n
    - n can be values between 1 - 31
- remote_listener
- user_dump_dest

Restricted parameters

These parameters are restricted for use by the Delphix Engine. Attempting to customize these parameters through the use of a VDB Config Template will cause an error during the provisioning process.

- active_instance_count
- cluster_database
- cluster_database_instances
- cluster_interconnects
- control_files
- db_block_size
- db_create_file_dest
- db_create_online_log_dest_n
    - n can be values between 1 - 5
- db_file_name_convert
- db_name
- db_recovery_file_dest
- db_recovery_file_dest_size
- db_unique_name
- dg_broker_config_file1
- dg_broker_config_file2
- dg_broker_start
- fal_client
- fal_server
- instance_name
- instance_number
- local_listener
- log_archive_config
- log_archive_dest
- log_archive_duplex_dest

- log_file_name_convert
- spfile
- standby_archive_dest
- standby_file_management
- tde_configuration
- thread
- undo_tablespace
- wallet_root
- __db_cache_size
- __java_pool_size
- __large_pool_size
- __oracle_base
- __pga_aggregate
- __sga_target
- __shared_io_pool_size
- __shared_pool_size
- __streams_pool_size

Automatic VDB restart on target server after reboot

The Delphix Engine now automatically detects whether a target server has been rebooted, and proactively restarts any VBD on that server that was previously up and running. This happens independent of the data platform; it is done as if a target server was restarted and a start command was issued from the Delphix Engine. This feature is compatible with Delphix Self-Service (Jet Stream) ordering dependencies and is limited to non-clustered VDBs.

This feature is not supported for the following data source types:

- Oracle RAC VDBs
- Oracle 12.1 and older vPDBs that are provisioned into a non-virtual CDB.
- MSSQL cluster VDBs.

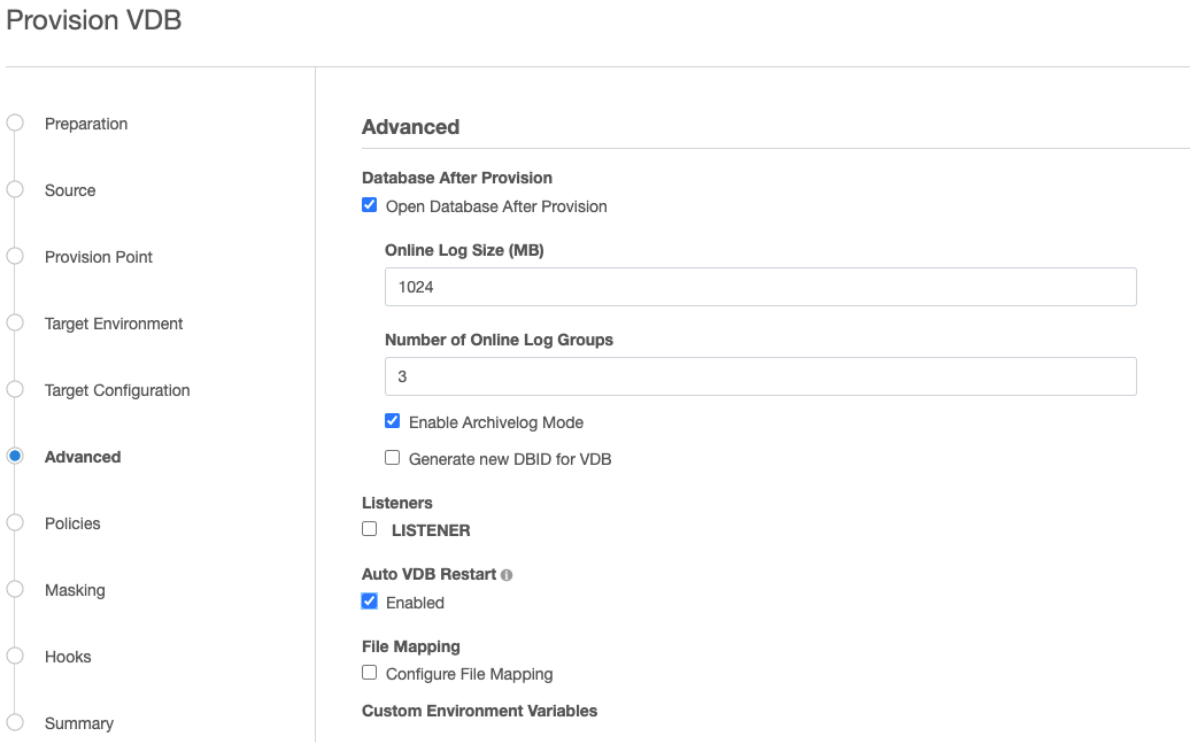
Automatic VDB Restart is supported for following Oracle data sources types:

- Non-Multi-Tenant (non-MT) VDBs
- Oracle 12c and later, vPDBs that are provisioned into a virtual CDB.
  - For Oracle 12.1.0.1, users can choose to enable or disable automatic restart at the virtual CDB level. There is no individual vPDB automatic restart setting.
  - For Oracle 12.1.0.2 and later versions, users can choose to enable or disable automatic restart for a virtual CDB or a virtual PDB.
- Oracle 12.2 and later vPDBs that are provisioned into a non-virtual CDB. Users can choose to enable or disable individual vPDB's automatic restart.

Please note, the scheduler checks every one minute if any VDB needs to be restarted. The Delphix Engine waits for five minutes after it detects a host has restarted before trying to restart VDBs. The engine will keep trying to restart the VDB for 30 minutes before giving up.

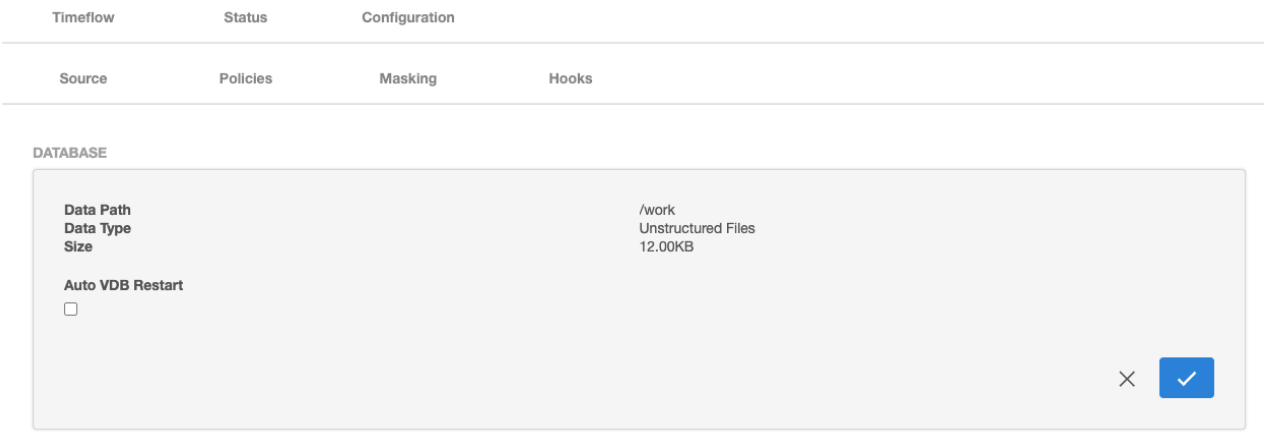
To enable automatic restart, complete the following steps:

1. When provisioning a new VDB in the VDB Provisioning wizard, check the **Auto VDB Restart** box.



2. Once the VDB has been provisioned, you will be able to turn **Automatic VDB Restart**on.

a. In the **Datasets** panel, select the **VDB**.
b. Select the **Configuration** tab.
c. Select **Source** sub-tab.
d. Select Database edit.



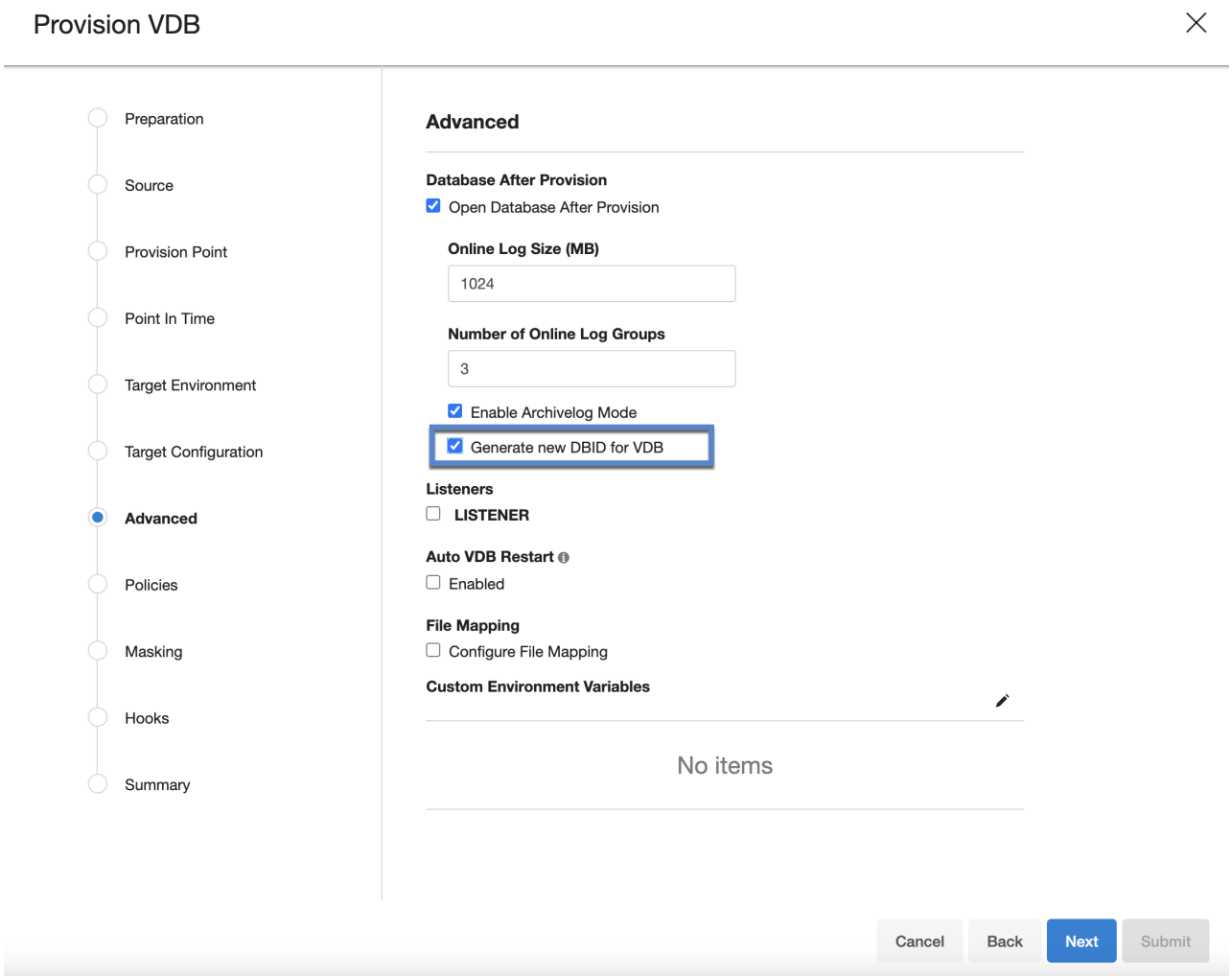
Generate a new DBID for Oracle VDBs

Delphix helps to generate a new Oracle internal database identifier (DBID) for VDBs. The DBID is an internal, unique identifier for an Oracle database.

> This feature is not supported for the following data source types:
> - Multitenant VDB
> - Oracle Live Source

Enable new DBID generation while provisioning a VDB

To enable generating a new DBID while provisioning a new VDB in the VDB Provisioning wizard, check the **Generate new DBID for VDB** option. This will create a new DBID for the provisioned VDB and subsequent refresh action performed to this VDB will get a different and new DBID.

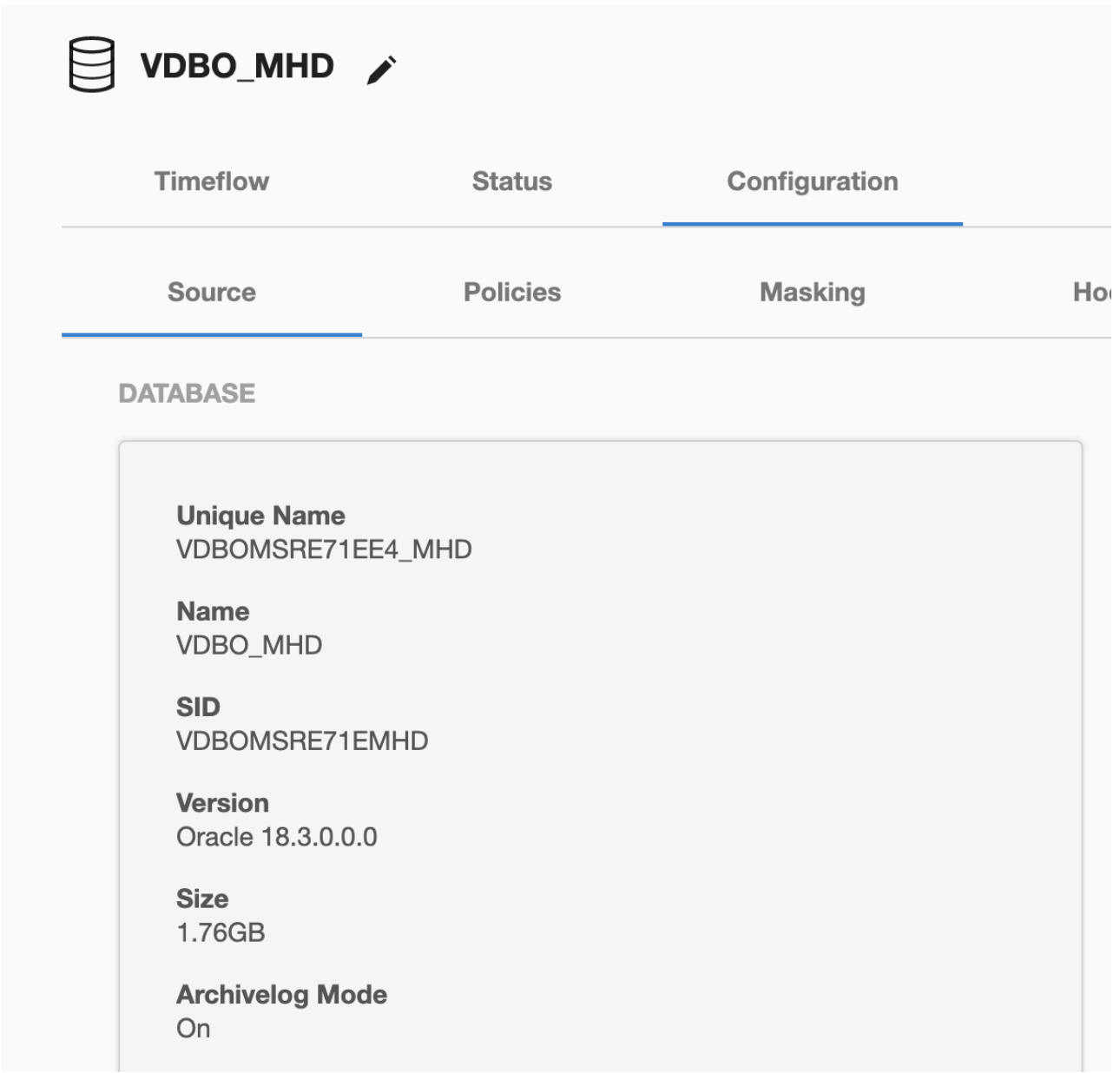


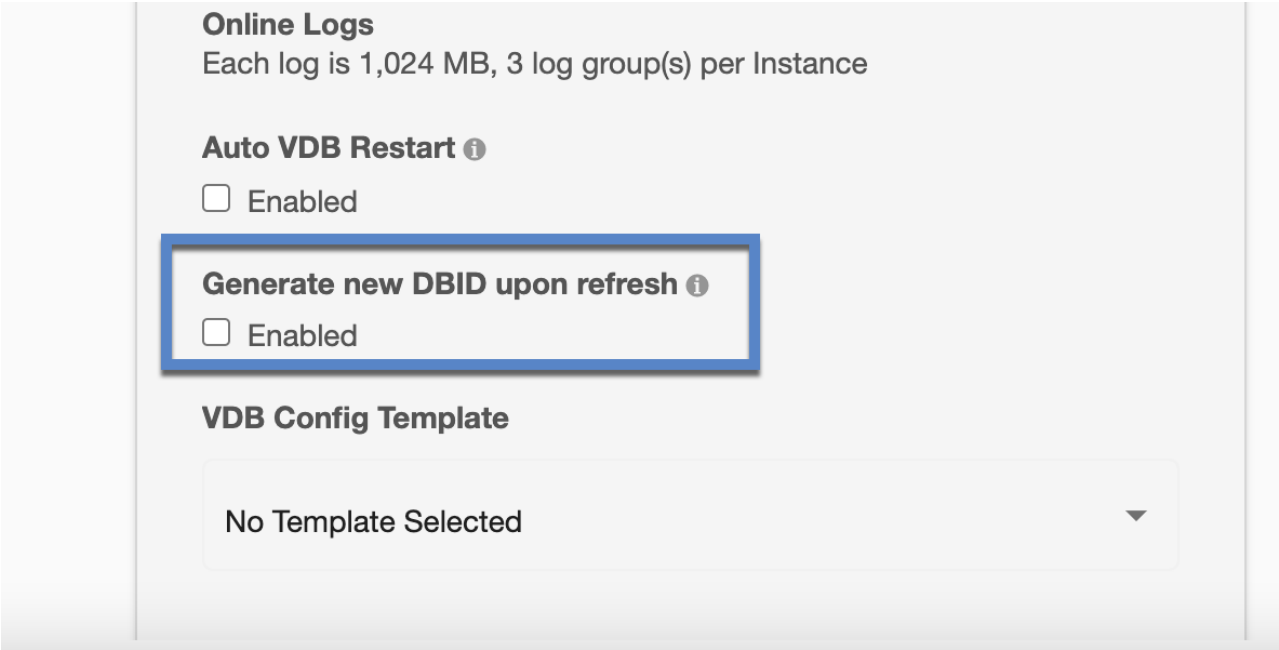
Enable new DBID generation for an already provisioned VDB

If originally the VDB had been provisioned without specifying the **generate new DBID for VDB** option, then you can enable it so that new refreshes will create VDBs with a new DBID (Note the current VDB or previous VDBs will still have the same DBID as the original database). To accomplish this follow the steps below.

1. In the **Datasets** panel, select the **VDB**.
2. Select the **Configuration** tab.
3. Select **Source** sub-tab.
4. Select Database and click the **pencil** icon to edit.
5. Check the **Generate new DBID upon refresh** option.
6. Refresh the VDB to see the changes.

If the **Generate new DBID upon refresh** option is enabled, then the Delphix engine will generate a new DBID upon refresh.





Disable new DBID generation for an already provisioned VDB

You can stop generating the new DBIDs for VDBs that were enabled for this at the time of provisioning or after provisioning. To accomplish this follow the steps below.

1. In the **Datasets** panel, select the **VDB**.
2. Select the **Configuration** tab.
3. Select **Source** sub-tab.
4. Select Database and click the **pencil** icon to edit.
5. Uncheck the **Generate new DBID upon refresh** option.
6. This will stop generating a new DBID for subsequent refreshes.
7. Refresh the VDB to reflect the changes.

ⓘ If the **Generate new DBID upon refresh** option is disabled, then the Delphix engine will use the same DBID of the parent upon refresh.

Customizing Oracle VDB environment variables

Overview

This topic describes how to customize the set of environment variables sourced prior to administering an Oracle virtual database (VDB). Certain Oracle database parameters are sensitive to the environment variables present when you start or administer the database. For this reason, the Delphix Engine allows you to dictate custom environment variables that will be set prior to any administrative action, such as provision, start, stop, rollback, or refresh.

You can specify environment variables by two different means:

- **Name-value pair** – A literal variable name and value to be set
- **Environment file** – An environment file to be sourced

Environment variables for Oracle RAC databases might vary in value between cluster nodes. Therefore, environment variable specifications for an Oracle RAC database must specify the cluster node to which they apply.

Setting custom environment variables

Prerequisites

If you are adding any environment variables that are environment files, these files must be accessible on the target environment.

Procedure

1. You can configure custom environment variables in the **Provision Wizard**.
    a. On the **Target Environment** tab, click **Advanced**. or
    b. You can also configure these variables on the **Configuration tab** when the VDB is disabled.
2. Click the **Plus** icon to add an environment variable.
3. Choose a **format** for the environment variable.
    a. Name-Value Pair
        i. Enter a **Name** to identify the variable.
        ii. Enter the variable's **Value**.
        iii. For Oracle RAC databases, you must also specify the cluster node to which this environment variable applies.
    b. Environment File
        i. Enter an absolute path to an environment file on the target environment. This path can be followed by parameters. Paths and parameters are separated by spaces.
        Escaping Spaces
        To specify literal spaces, escape them with a backslash ("hello\ world" -> "hello world"). To specify literal backslashes, escape them with a backslash ("foo\" -> "foo"). Any other character preceded by a backslash will retain both the backslash and the original character ("\b" -> "\b"). Escaping is done in order from left to right ("part1\ part2" -> "part1" "part2" will be two parameters).
        ii. For Oracle RAC databases, you must also specify the cluster node to which this environment variable applies.
4. Save the custom environment variables by completing provisioning, or clicking the **Confirm**. These environment variables will take effect when you start the Oracle VDB.

Environment variable denylist

The Delphix Engine denylists the following environment variables; they cannot be set by the user.

- `ORACLE_SID`

- `ORACLE_BASE`
- `ORACLE_HOME`
- `CRS_HOME`
- `ORACLE_UNQNAME`
- `ORAENV_ASK`
- `LOGON_STR`
- `DLPX_SHELL`
- `SQLPLUS_PLSQL_MODIFIERS`
- `SQLPLUS_DML_MODIFIERS`
- `SQLPLUS_DDL_MODIFIERS`

If a Name-Value pair has any one of these prohibited environment variables as the name, an error will be raised.

If an environment file sets one of these variables, the Delphix Engine will override this value when the Oracle VDB is started.

The following environment variables will be set before invoking the user-specified script, and thus can be accessed within the script.

- `ORACLE_SID`
- `ORACLE_BASE`
- `ORACLE_HOME`
- `CRS_HOME`
- `ORACLE_UNQNAME`

User-Input sanitation for environment variables

For security purposes, user-input provided through the custom environment variables feature retains its literal value when interpreted, including ', ", and undefinedORACLE_HOME

- `undefinedORACLE_SID`
- `undefinedORACLE_UNQNAME`
- `undefinedPATH`

Caveats

- Environment variables are sourced on provision, start, stop, rollback, and refresh. Custom environment variables are not applicable to V2P.
- Custom environment variables do not propagate to child VDBs and must be set again on provision.
- Custom environment variables do not persist after migration. On migration of a VDB with custom environment variables, an alert will be raised that the custom environment variables have been removed from the VDB. In order to view the alert, go to **System** > **Event Viewer**.
- Custom environment variables are *not* available within scripts executed by VDB Hook Operations.

Customizing VDB file mappings

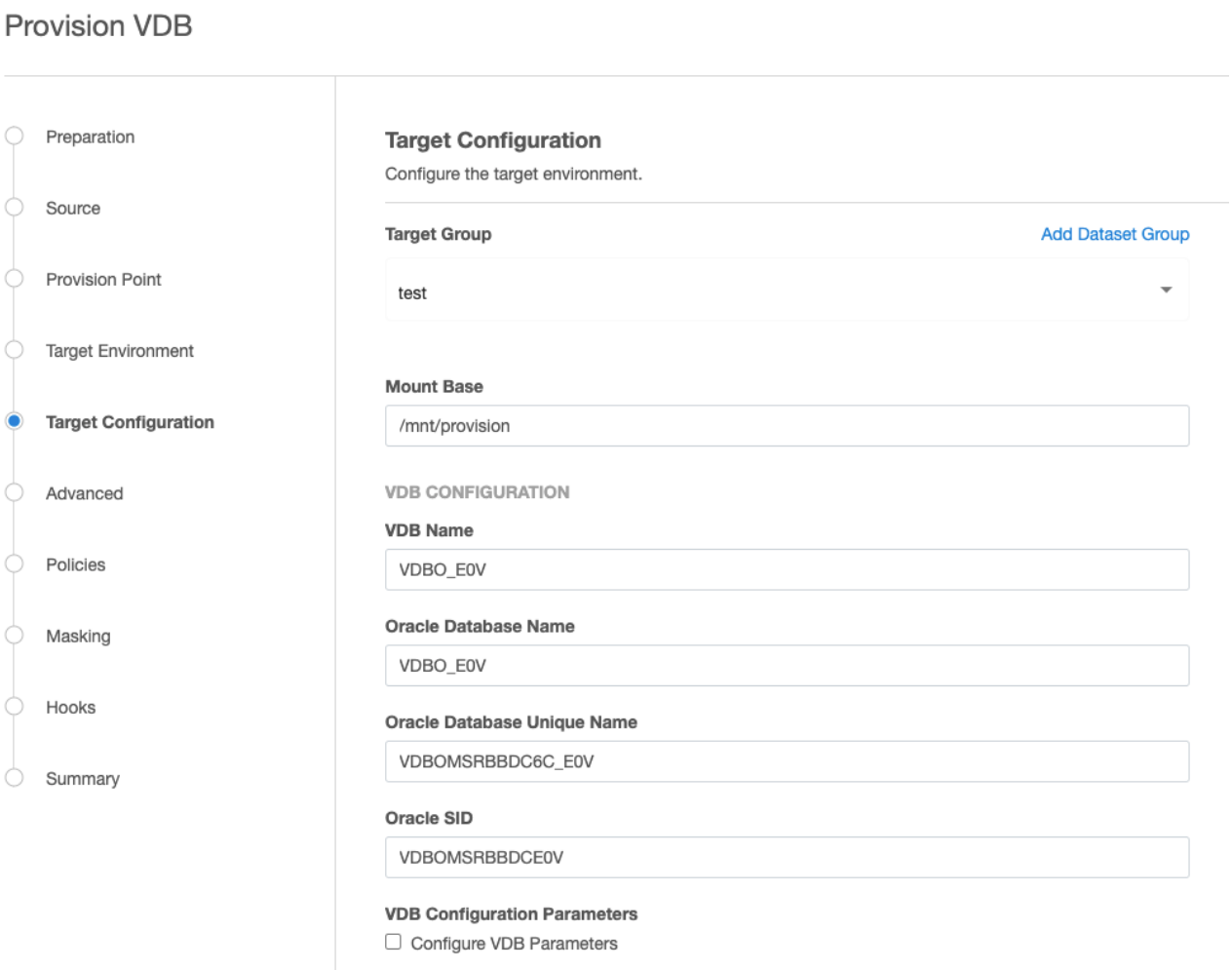
This topic describes how to customize file path mappings when provisioning a virtual database (VDB).

In the VDB provisioning process, it may be necessary to create mappings between files and directories that exist on the source, and files or file directories that exist on the target. An example of this is creating a copy in the target environment of a wallet file for an encrypted database that exists in the source environment.

During VDB provisioning, the archive, datafile, external, script and temp directories are mounted from the Delphix Engine to the unique="">>

You can specify the Mount Base and the Database Unique Name on the Provision VDB card at provision time. But the archive, datafile, external, script and temp mount points are fixed.

- The default Mount Base is set to /mnt/provision
- The default Database Unique Name starts with a capital V and includes the dSource name



Mount Base: /mnt/provision

Database Unique Name: Vdbhcp3_35F

Mounts: /mnt/provision/Vdbhcp3_35F/archive

/mnt/provision/Vdbhcp3_35F/datafile

/mnt/provision/Vdbhcp3_35F/external

/mnt/provision/Vdbhcp3_35F/script

/mnt/provision/Vdbhcp3_35F/temp

File Mappings affect everything that follows the mount point directories above. For datafiles, the datafile location from the dSource is used. You can determine the directory structure by querying the Source database:

```
SQL> select name from v$datafile;
```

```
NAME
```

```
--------------------------------------------------------------------------------
```

```
/datafile/dbdhcp3/oradata/dbdhcp3/system01.dbf
```

```
/datafile/dbdhcp3/oradata/dbdhcp3/sysaux01.dbf
```

```
/datafile/dbdhcp3/oradata/dbdhcp3/undotbs01.dbf
```

```
/datafile/dbdhcp3/oradata/dbdhcp3/users01.dbf
```

```
/u03/app/ora11202/product/11.2.0/dbhome_1/dbs/dbv_R2V4.dbf
```

Users have control over the datafiles that were returned by the select name from v$datafile and also the controlfile location.

Archive log files go directly into /mnt/provision/Vdbhcp3_35F/archive
Tempfiles go directly into /mnt/provision/Vdbhcp3_35F/temp

By default VDB File Mapping would append the above directory structure to the /mnt/provision/Vdbhcp3_35F/ datafile directory:

```
/mnt/provision/Vdbhcp3_35F/datafile/datafile/dbdhcp3/oradata/dbdhcp3/system01.dbf
/mnt/provision/Vdbhcp3_35F/datafile/datafile/dbdhcp3/oradata/dbdhcp3/sysaux01.dbf
/mnt/provision/Vdbhcp3_35F/datafile/datafile/dbdhcp3/oradata/dbdhcp3/undotbs01.dbf
/mnt/provision/Vdbhcp3_35F/datafile/datafile/dbdhcp3/oradata/dbdhcp3/users01.dbf
/mnt/provision/Vdbhcp3_35F/datafile/u03/app/ora11202/product/11.2.0/dbhome_1/dbs/
dbv_R2V4.dbf
```

Pattern matching example

You can use pattern matching rules to create full path names for data files and control files.

Pattern matching rules have the form **source-regex-expression-KEY : target-replacement-VALUE**. You can use multiple rules, which are applied successively. Multiple rules with the same source key are allowed.

File mapping options

Example 1

For this example, the ultimate goal is to create a VDB that has the following datafile File Mappings:

```
/u03/devvdb3/datafile/sys/system01.dbf
```

```
/u03/devvdb3/datafile/sys/sysaux01.dbf
```

```
/u03/devvdb3/datafile/ctrl/control01.ctl
```

```
/u03/devvdb3/datafile/undo/undotbs01.dbf
```

```
/u03/devvdb3/datafile/data/users01.dbf
```

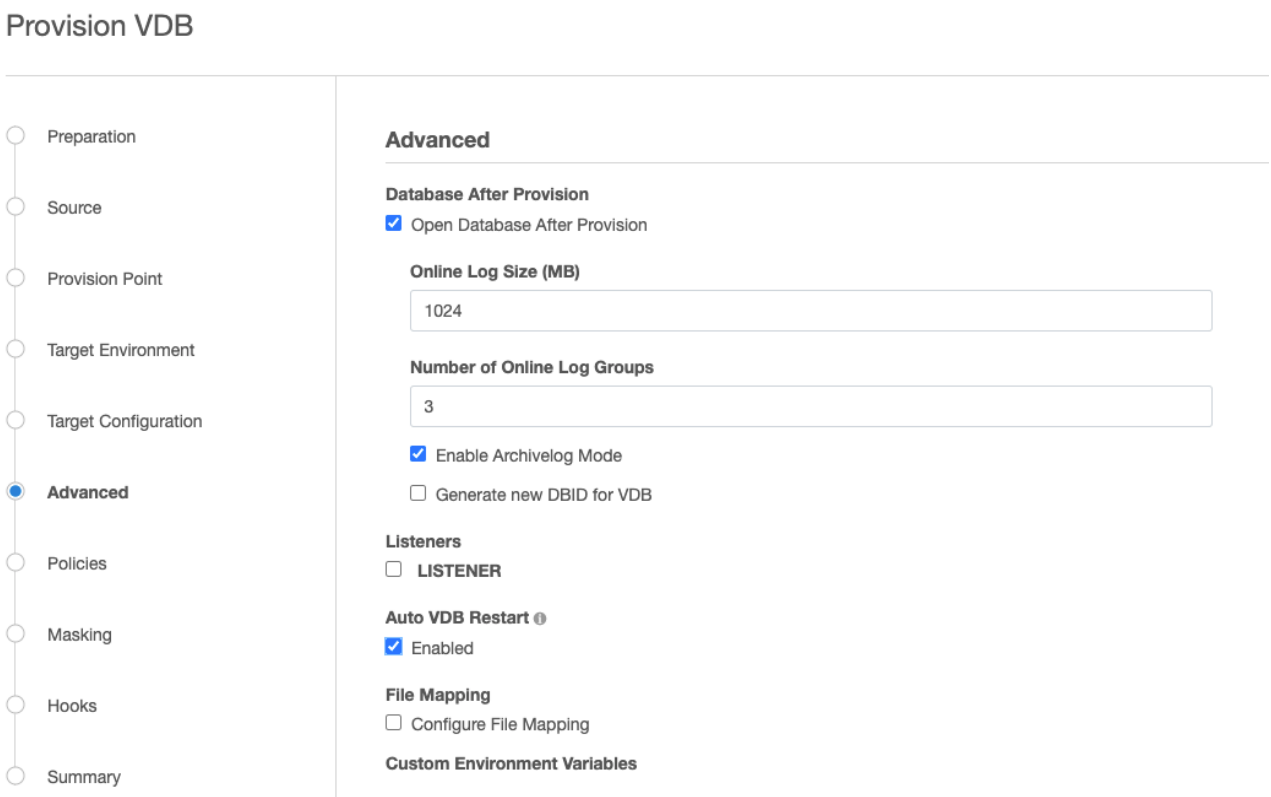
```
/u03/devvdb3/datafile/data/dbv_R2V4.dbf
```

You can change the default behavior in a few different ways.

1. Modify the Mount Base.
2. Modify the Database Unique Name.
3. Modify the Mount Base and the Database Unique Name.
4. Modify the name of the individual subdirectories within the Mount Base/Database Unique Name/datafile directory.

Numbers 1-3 can be changed by simply modifying the Mount Base and/or Database Unique Name values when provisioning a VDB:

The datafile locations can be changed in the **Advanced** tab of the **Provision VDB** wizard and scroll down to **File Mapping**:



When you are changing File Mapping, remember that you only need to refer to the part of the path after /Mount Base/Unique Database Name/datafile.

Start by changing all the datafile locations from /u03/devvdb3/datafile/datafile/dbdhcp3/oradata/dbdhcp3

to

/u03/devvdb3/datafile/data by selecting the **+** sign and setting **File Mapping Source File Match** and the **Replacement** and then select **Validate** to see result.

The result will show the new directory structure for all the datafiles and control file relative to the " /Mount Base/Unique Database Name/datafile" directory, but will only show the directories after the datafile directory.

```
Source File Match                  Replacement        Result
/datafile/dbdhcp3/oradata/dbdhcp3  /data              /data/users.dbf
```

Selecting Validate will show the result of the above mapping, for all the datafiles.

The File Mappings build upon one another, so the first mapping moves almost all the datafiles and the controlfile to /u03//devvdb3/datafile/data.

/u03/app/ora11202/product/11.2.0/dbhome_1/dbs/dbv_R2V4.dbf was not remapped, as it did not contain the directory structure defined in in Source File Match (/datafile/dbdhcp3/oradata/dbdhcp3)

If you wanted the remaining datafiles and controlfile to remain in /data, it would only require the two File Mappings.

The remaining datafiles and controlfile can be relocated by adding to the mapping (select "+" between each Source File Match/Replacement pair).

```
/datafile/dbdhcp3/oradata/dbdhcp3/system01.dbf                /sys/system01.dbf      /
sys/system01.dbf
```

```
/datafile/dbdhcp3/oradata/dbdhcp3/sysaux01.dbf                /sys/sysaux01.dbf      /
sys/sysaux01.dbf
```

```
/datafile/dbdhcp3/oradata/dbdhcp3/undotbs01.dbf               /undo/undotbs01.dbf    /
undo/undotbs01.dbf
```

```
/datafile/dbdhcp3/oradata/dbdhcp3/control01.ctl               /ctrl/control01.ctl    /
ctrl/control01.ctl
Select Validate between each new entry in order to verify that datafiles are being
mapped as expected.
```

Once all the files are located the way you want them, select **Next** to continue the provision process.

The **Summary** page will show the modifications to **Mount Base** and **Unique Database Name** and will show that **Customized File Mapping** was defined.

After provisioning completes, you can login to the Target server and verify that the datafiles were mapped correctly:

```
SQL> select name from v$datafile;
```

```
NAME
```

```
--------------------------------------------------------------------------------
```

```
/u03/devvdb3/datafile/sys/system01.dbf
```

```
/u03/devvdb3/datafile/sys/sysaux01.dbf
```

```
/u03/devvdb3/datafile/undo/undotbs01.dbf
```

```
/u03/devvdb3/datafile/data/users01.dbf
```

```
/u03/devvdb3/datafile/data/dbv_R2V4.dbf
```

Example 2

In this example, several rules are applied to the source file path **/app/oracle/oradata/system01.dbf**.

1. Applying the rule **ora:foo** results in: **/app/foocle/foodata/system01.dbf**
2. Applying the rule **foo:bar** results in: **/app/barcle/bardata/system01.dbf**
3. Applying the rule **ora:no** results in an error, because **ora** is no longer found in the pathname.
4. Applying the rule **bar:oranew** results in: **/app/oranewcle/oranewdata/system01.dbf**
5. Applying the rule **ora:yes** results in **/app/yesnewcle/yesnewdata/system01.dbf**

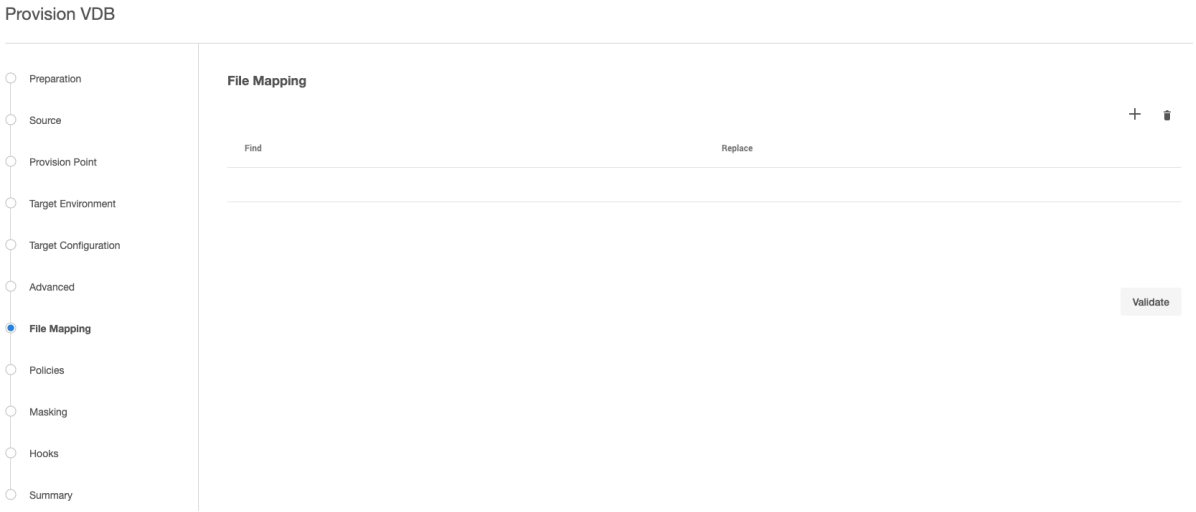
During the pattern matching process, two errors can be generated.

1. **No match for specified mapping rules** This is the result when no rules match a source file
2. **Invalid regex pattern specified for path mapping** This is the result of an invalid regex rule mapping

This topic on the java.regex.util class, hosted on docs.oracle.com, shows the regular expression syntax and constructs recognized by the Delphix Engine pattern-matching operations.

Applying VDB file mappings during the provisioning process

1. In the **Target Configuration** tab of the **Provision VDB** wizard, click **Advanced**.
2. Select **Configure File Mapping**.
3. Click **Next**.
4. Click the **Plus** icon to add a mapping rule.



5. Enter the mapping rule.
6. Click **Validate** to see the results of applying the rule. If not matches are found, you will see an error message.
7. Click **Next** to continue with the provisioning process.

Manually starting a VDB

When starting an Oracle VDB instance, an initialization file `$ORACLE_BASE_CONFIG/dbs` is used and by default, this directory does not have group write permissions. Previously to Delphix 6.0.10.0, this limitation forced target hosts to either use the instance owner for provisioning or modify the group permissions on this directory. Both options can be seen as a security risk and increase deployment complexity.

With Delphix Engine 6.0.10.0 onwards, a target host no longer requires write permission to the `$ORACLE_BASE_CONFIG/dbs` directory. Delphix Engine will only copy an initialization file *init*<> to this directory if write permissions exist.

All instance startup attempts from Delphix operations will specify an initialization file to use from the Delphix filesystem rather than the default Oracle location. If a VDB needs to be manually started and Delphix was unable to copy the initialization then it must be specified in the startup command. The instance init file is available in the VDB script directory.

You can use the following syntax to manually start the instance:

```
SQL> startup pfile='/mnt/provision/VDBOMSR66E005_61O/script/VDBOMSR66E061O/
initVDBOMSR66E061O.ora';
ORACLE instance started.
```

If there is no write permission in the `$ORACLE_BASE_CONFIG/dbs` directory then manually starting the instance will fail:

```
SQL> startup
ORA-01078: failure in processing system parameters
LRM-00109: could not open parameter file '/u01/app/oracle/product/11.2.0.4/dbhome_1/
dbs/initVDBOMSR66E061O.ora'
```

To manually start a VDB, complete the following steps:

1. Ensure that the VDB is enabled and NFS filesystems are mounted. This is required only when the NFS filesystems are offline. Follow the below steps via CLI.

   ```
   demo source> select VDBOMSR66E005_61O
   demo source 'VDBOMSR66E005_61O'> enable
   demo source 'VDBOMSR66E005_61O' enable *> set attemptStart=false
   demo source 'VDBOMSR66E005_61O' enable *> commit
       Dispatched job JOB-47
       SOURCE_ENABLE job started for "VDBOMSR66E005_61O".
       Enabling dataset "VDBOMSR66E005_61O".
       Exporting storage containers from the Delphix Engine.
       Mounting datasets.
       Dataset "VDBOMSR66E005_61O" enabled.
       SOURCE_ENABLE job for "VDBOMSR66E005_61O" completed successfully.
   ```

2. Login to the target host as a Delphix OS user.

3. Locate the VDB scripts directory. This is located at `/<toolkit_directory>/Delphix_<engineuuid>_<delphix_osuser_id>_<host|cluster>/databases/oracle/<vdb_uniq_name>/<vdb_instance_name>/.`

```
cd /work/Delphix_8e501f827dee_a7a9072fc4fe_2_host/databases/oracle/
VDBOMSR66E005_61O/VDBOMSR66E061O
```

4. Run "`setup-oraenv.sh`". This configures the Oracle environment variables such as ORACLE_SID and ORACLE_HOME.
5. Startup the instance by specifying the path to the instance init file in the scripts directory.

```
[oracle@mwrh74-ora11204-tgt VDBOMSR66E061O]$ sqlplus "/as sysdba"

SQL*Plus: Release 11.2.0.4.0 Production on Thu Jul 22 02:40:47 2021

Copyright (c) 1982, 2013, Oracle.  All rights reserved.

Connected to an idle instance.

SQL> startup pfile='/work/Delphix_8e501f827dee_a7a9072fc4fe_2_host/databases/
oracle/VDBOMSR66E005_61O/VDBOMSR66E061O/initVDBOMSR66E061O.ora';
ORACLE instance started.

Total System Global Area 1068937216 bytes
Fixed Size          2260088 bytes
Variable Size         616563592 bytes
Database Buffers      444596224 bytes
Redo Buffers            5517312 bytes
Database mounted.
Database opened.
SQL>
```

If Delphix Engine has permission to write to `$ORACLE_BASE_CONFIG/dbs` during provisioning, the instance can be started up with only the "startup" command.

Provisioning an Oracle VDB

This topic describes how to provision a virtual database (VDB) from a dSource or another VDB.

> Masked Provisioning
>
> Masked Provisioning is supported on Oracle RAC only when used with "script-based masking".

Prerequisites

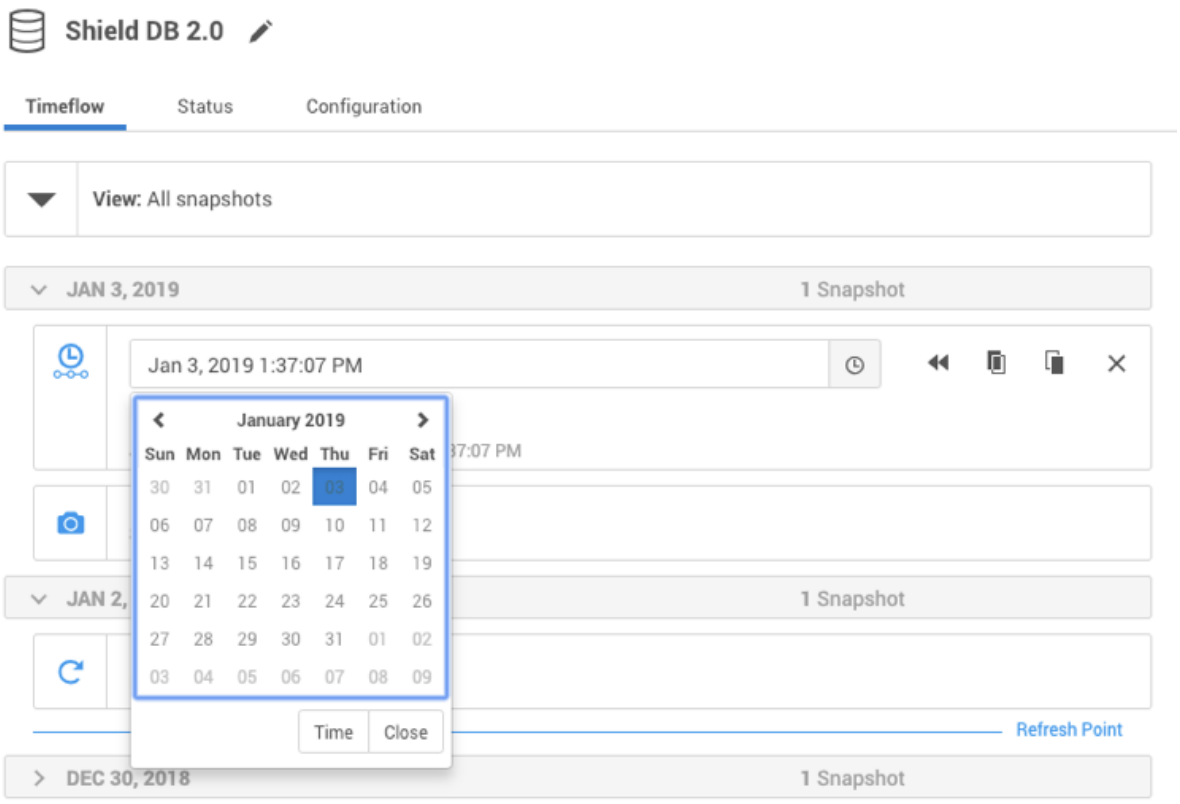
- You must have already done one of the following:
  - linked a dSource from a source database, as described in Linking an Oracle Data Source or
  - created a VDB from which you want to provision another VDB
- You will need to have the correct OS User privileges on the target environment, as described in Requirements for Oracle Target Hosts and Databases
- If you want to use customized database configuration settings, first create a VDB Config Template as described in Customizing Oracle VDB Configuration Settings
- If you are creating a VDB from a dSource linked to an encrypted database, make sure you have copied the wallet file to the target environment, as described in Provisioning a VDB from an Encrypted Oracle Database.

> **VDB configuration templates**
> It is recommended that you always create a VDB Configuration Template prior to provisioning a Virtual Database. This will allow you to customize your parameters for the initial provisioning, and ensure that subsequent changes are reflected on the VDB when refresh and rewind operations are run. See Customizing Oracle VDB Configuration Settings for more information.

> Once a VDB has been provisioned, archive logs, flashback logs, redo logs, and datafiles that are not known to be part of the database will be deleted when a SnapSync is performed on the VDB. As a result, make sure that no files are added to the `datafile` file system, as this is used by Delphix to store datafiles. Use the `external` filesystem for other files that are needed to be included in the snapshot that are not datafiles.
>
> For example, creating a second control file on the `datafile` filesystem is prohibited. Such a file will be removed after SnapSync, rendering the VDB unusable. Similarly, block change tracking must not be enabled on a VDB.

Procedure

1. Login to the **Delphix Management** application.
2. Select **Manage > Datasets**.
3. In the **Datasets** panel on the left-hand side, click the **group** containing the dSource or VDB from which you want to provision.
4. Click the **TimeFlow** tab.
5. Select a **snapshot**.
   For more information on provisioning options.
   Info : You can take a snapshot of the dSource from which to provision. To do so, click the **Camera** icon.
6. **Optional:** Select  to open LogSync timeline.



7. Select [ ] to provision from a point of time within a snapshot. You can select by date or time.
8. Click [ ] and the **Provision VDB**wizard will open:

- For **Oracle Single Instance** the fields **Installation Home**, **Database Unique Name**, **SID**, **Database Name**, **Mount Base**, and **Environment User** will auto-populate with information from the parent.
- For **Oracle RAC** the fields **Installation Home**, **Database Unique Name**, **SID**, **Database Name**, **Mount Base**, **Instance Number, Instance Name** and **Environment User** will auto-populate with information from the parent.

> Editable Fields in the VDB Provision Wizard
> The following fields are editable:
> Installation Home (need to have an additional compatible target)
> Database Unique Name
> SID
> Database Name
> Mount Base
> Instance Number (RAC Only)
> Instance Name (RAC Only)

9. If you need to add a new target environment for the VDB, click the green **Plus** icon next to the **Filter Target** field, and follow the instructions in Adding an Oracle Single Instance or RAC Environment
10. Review the information for **Installation Home**, **Database Unique Name**, **SID**, and **Database Name**. Edit as necessary.

11. Review the **Mount Base** and **Environment User**. Edit as necessary. The Environment User must have permission to write to the specified Mount Base, as described in Requirements for Oracle Target Hosts and Databases. You may also want to create a new writeable directory in the target environment with the correct permissions and use that as the Mount Base for the VDB.
12. Select **Provide Privileged Credentials** if you want to use login credentials on the target environment that are different from those associated with the **Environment User**.
13. Click **Advanced** to customize the VDB online log size and log groups, archivelog mode, local_listener parameter (TCP/IPC protocol addresses), additional VDB configuration settings or file mappings, or custom environment variables.If you are provisioning to a target environment that is running a Linux OS, you will need to compare the `SGA_TARGET` configuration parameter with the shared memory size in `/dev/shm`. The shared memory configured on the target host should match the SGA memory target. You can check the Linux OS shared memory size with the command df -k /dev/shm and the SGA_TARGET configuration parameter by opening the **Advanced** settings and then finding the value for `SGA_TARGET` under **VDB Configuration Templates**.
14. Click **Next**.
15. Select a **Target Group** for the VDB.
16. Enable Auto VDB Restart to allow the VDB to be automatically restarted when the target host reboot is detected by Delphix.

> The **Auto VDB Restart** feature is currently not available for Pluggable Databases, Clusters, or LiveSource.

17. Click Next.
18. Select a **Snapshot Policy** for the VDB.
19. Click **Next**.
20. Enter any operations that should be run at Hooks during the provisioning process.
21. Click **Next**.
22. Click **Submit**.

When provisioning starts, you can review the progress of the job by selecting the VDB and clicking on the **Status** tab, or by selecting **Manage/Dashboards** and viewing the **Job History** panel. Alternatively, you could see this in the **Actions Sidebar.** When provisioning is complete, the VDB will be included in the group you designated and listed in the **Datasets** panel. If you select the VDB in the **Datasets** panel and click the **Configuration** tab, you can view information about the database and its Data Management settings.

Provisioning by snapshot or LogSync

Refer to Overview of Provisioning Oracle Virtual Databases

Adding or removing RAC VDB cluster node after a VDB is provisioned

Prerequisites

- Disable the VDB before you edit the instance configuration.

Procedure

After provisioning:

1. Click the **group** containing the VDB.
2. Click the **VDB**.
3. Disable the VDB in order to make changes to instance configuration.
4. Under **Configuration > Source** click the **edit** button and edit the following:
    a. **Instance Number** for each corresponding instance
    b. **Instance Name**
    c. Check or uncheck the cluster nodes you want for this RAC VDB

5. Click the check button to save changes.
6. Enable the VDB to apply the instance configuration changes.

Provisioning an Oracle virtual pluggable database (vPDB)

In the Oracle multitenant architecture, there are two main database types: container databases (CDBs) and pluggable databases (PDBs).

Source CDBs containing Application Container PDBs are currently not supported.

The process of creating virtual pluggable databases is similar to creating non-multitenant virtual databases, with a few additional steps necessary in the multitenant architecture.

A Solaris x86 source host is compatible with a Linux x86 target host and vice versa.

Delphix supports provisioning Oracle Virtual Pluggable Databases (vPDBs) in two configurations:

1. **Linked container databases (Linked CDBs)**: Physical CDBs that have been previously provided by the Oracle DBA on the target environment to which Delphix may provision vPDBs. Physical CDBs must be configured and set up specifically for use by Delphix.
2. **Virtual container databases (vCDBs)**: vCDBs are created by Delphix during the provision workflow for vPDBs. Once created for Oracle versions 12.1.0.2 and later, it may be used to provision additional vPDBs.

Prerequisites for provisioning a vPDB to a linked CDB

There must be a target environment that has an Oracle installation compatible with the Oracle installation of the source CDB and the source PDB. The following database requirements are needed to provision the vPDB to a linked CDB. For more information, see Requirements for Oracle Hosts and Databases

- Recommend autodiscovery so that the linked CDB can be found. Otherwise, the linked CDB must be manually discovered before provisioning.
- Linked CDB must be running
- Linked CDB must be in ARCHIVELOG mode
- Linked CDB should be using Block Change Tracking (BCT)
- LogSync must be enabled for the Linked CDB.

Provisioning a vPDB to a linked CDB

For provisioning a vPDB into a Linked CDB, the source CDB and the target CDB must meet the following compatibility requirements:

- The value of the COMPATIBLE parameter in the source CDB must be less than or equal to the value of the COMPATIBLE parameter in the target CDB
- They must have the same endianness
- They must have compatible character sets and national character sets

Setting up auxillary CDB parameters

During a TDE-enabled vPDB provisioning into a Linked CDB or a vCDB, the Delphix Engine creates a temporary CDB instance (or Auxiliary CDB) on the target environment to recover the vPDB to a consistent state. This temporary CDB will be automatically deleted (in case of provisioning to a linked CDB or to an existing vCDB) or converted to a vCDB (in case of provisioning to a new vCDB) after the vPDB is provisioned successfully. By default, this temporary CDB is configured to have the same init parameters as the source database. To manage the configuration of the temporary CDB init parameters, you must set up Repository Templates for Oracle Databases prior to provisioning.

For more details on additional TDE-related processing that is performed in the Auxiliary CDB, see the "A closer look at TDE provisioning" section.

Provisioning a vPDB to a new or an existing vCDB

When provisioning a vPDB into a new vCDB:

- Delphix will provision a virtual CDB (vCDB) from the source CDB to host the vPDB you are about to create.
- There is no need to set up a repository template, as this provision workflow allows a user to directly specify a VDB configuration template, or set individual database parameters.

Procedure

1. In the **Datasets panel**, select an Oracle PDB dSource or a previously provisioned vPDB.
2. From the **Timeflow** tab, select a snapshot or point in time to provision from.
3. Once the Provision wizard is open, you can either provision with a:
    - Target Linked CDB: Select an existing container database as the provision target CDB from the **Container Database** drop-down menu of CDBs on that environment.
    - Existing vCDB: Select an existing vCDB as the provision target CDB from the **Container Database** drop-down menu of CDBs on that environment. (Supported only for Oracle versions 12.1.0.2 and later.)
    - New vCDB: Select the **Create a New Container Database** checkbox. This will create a new vCDB object in that environment with this new vPDB plugged into it.
4. Click **Next** to advance the left-hand pane to the **Target Configuration** tab, and edit as necessary.
5. The Environment User must have permission to write to the specified Mount Base, as described in Requirements for Oracle Environments and Data. You can also reuse the Delphix toolkit directory, which already exists as the Mount Base, or create a new writable directory in the target environment with the correct permissions and use that as the Mount Base.
6. Enter the vPDB Name, Target Group for the vPDB you are about to provision.
7. If you selected to create a new target vCDB, configure the vCDB:
    - Enter the vCDB Name, Database Unique Name, and Database Name for the vCDB you are about to provision.
    - Select the Configure vCDB Parameters checkbox if you want to use a VDB Configuration Template. See Customizing Oracle VDB Configuration Settings
8. Click **Next** to advance the left-hand pane to the Advanced tab.
9. The available options are Auto vCDB Restart, Auto vPDB Restart, File Mapping, and custom environment variables. For more information, see Customizing VDB File Mappings and Customizing Oracle VDB Environment Variables
10. Click **Next** to advance the left-hand pane to the Policies tab.
11. Select the VDB Snapshot policy to be applied to the vPDB. Select a Retention Policy for the vCDB, if you are provisioning a vCDB.
12. Click **Next** to advance the left-hand pane to the Masking tab. Select the Mask this vPDB checkbox if you want to mask, and select the masking job to be applied.
13. Click **Next** to advance the left-hand pane to the Hooks tab, and create any hooks if necessary. For more information, see Hook Scripts for Automation and Customization
14. Review the provisioning summary. Click **Submit** to proceed with provisioning the vPDB.

Provisioning a TDE-enabled vPDB

Overview

Refer to the Transparent Data Encryption and Delphix article for an overview of TDE and the Delphix implementation.

Provisioning a TDE-enabled Virtual Pluggable Database (vPDB) to a TDE-enabled target container requires specifying a few TDE provisioning parameters using the GUI or CLI, in addition to the vPDB parameters (such as the vPDB name and target container) and the snapshot to provision from. A TDE-enabled vPDB can be provisioned to either a Linked CDB or a vCDB. It is important to note that Delphix does not support provisioning a TDE-enabled vPDB from a source snapshot of a dSource or virtual database that is not encrypted at the time of linking.

Provisioning parameters for a TDE-enabled vPDB

To initiate the provision, Delphix needs the following pieces of information, all of which can be specified in the GUI or CLI:

Parameter	Description	CLI Parameter	Required?	Notes
Parent Database TDE Keystore Location	Path to a keystore on the target host that contains the keys used to encrypt the source database datafiles.	`source.parentTdeKeystorePath`	Yes	• The keystore must be accessible on the target host(s) in the specified path. • The wallet `ewallet.p12` must exist in the specified path and must be readable by the Oracle user. This is normally a copy of the source database wallet. • If the parent database TDE keystore has moved to a different location on the target host after provisioning the vPDB, this path must be updated via the GUI or CLI, as specified in the Updating the Parent Database TDE Keystore Location section, otherwise a subsequent refresh operation on that vPDB will fail. • If the master encryption keys of the source database have changed after the vPDB has been provisioned, you must ensure that a copy of the new encryption keys are present in the parent database TDE keystore location, usually by copying the updated parent database TDE keystore wallet from the source database to the parent database TDE keystore location on the target host. **Warning** Parent Database TDE keystore may be located on local storage, NFS or ACFS, but cannot be located on an ASM filesystem.
Parent Database TDE Keystore Password	Password for the parent database TDE keystore.	`source.parentTdeKeystorePassword`	Yes	This parameter must be updated if the password is changed.

Parameter	Description	CLI Parameter	Required?	Notes
TDE Secret for Exported Keys	Secret for the exported keys.	`source.tdeExportedKeyFileSecret`	Yes	• Oracle requires a password to be set when exporting keys to a keyfile from a keystore. The secret is an alphanumeric string that protects the keys in the file. • This parameter represents a new user-specified secret that is used by Delphix when exporting keys, and does not need to match any existing keystore password. • Once a vPDB is provisioned using this secret, it cannot be changed for the lifetime of the vPDB. • This secret is used by Delphix during provisioning and subsequent vPDB operations that require exporting the keys to a keyfile. **Warning** Make sure the TDE Secret for Exported Keys is stored in a secure location for your records. It is only known to you. In the rare event that keys need to be manually extracted from an exported keyfile, this password will be required. Delphix Support cannot assist with manually exporting keys without this password, therefore it should be known or recorded within your organization.
Target Keystore Password	Password for the target Linked CDB or existing vCDB keystore.	`sourceconfig.tdeKeystorePassword`	Yes for Linked CDB or existing vCDB targets. Not applicable to new vCDB targets.	• This parameter must be updated via the GUI or CLI in the Environments page, as specified in the Adding or Editing the Target Keystore Password section. • This is required when provisioning to an existing Linked CDB or existing vCDB, and must match the password used to open the Linked CDB or existing vCDB keystore.

Parameter	Description	CLI Parameter	Required?	Notes
TDE Keystores Root	Path to a directory on the target host under which all Delphix related TDE artifacts will be created.	`host.oracleParameters.tdeKeystoresRootPath`	Yes for cluster targets. Optional for single instance targets.	• This includes keystores used by the auxiliary CDB during provisioning and the artifact directories for TDE-enabled vPDBs. • This is an arbitrary path, which does not need to be referenced by `sqlnet.ora` or `wallet_root`. • When provisioning to a single instance target, this will default to `<toolkit path>/tde`. When provisioning to a cluster target, this path must be on shared storage and available to all cluster hosts. • The Environment User must have permission to write to this path.
Target vCDB TDE Keystore Location	Path of the location on the target host at which Delphix will create the keystore during new vCDB provisions.	`source.targetVcdbTdeKeystorePath`	Yes for new vCDB targets. Not applicable to linked CDB or existing vCDB targets.	• This path refers to a location on the target host. • This path must either not exist or can be an existing empty directory. • The Environment User must have permission to write to this location. • For Oracle 12.2, the path must match what is specified by `sqlnet.ora`. For higher versions, Delphix will set the `wallet_root` parameter to the provided location. • This parameter must be updated if the keystore location is changed or else future Delphix operations may fail.
Target vCDB TDE Keystore Password	Password for the new vCDB keystore.	`virtualCdb.sourceConfig.tdeKeystorePassword`	Yes for new vCDB targets. Not applicable to linked CDB or existing vCDB targets.	• This password is created during provisioning. It does not need to match any existing keystore password. • If this password is changed, it must be updated via the GUI or CLI in the Environments page, as specified in the Adding or Editing the Target Keystore Password section.

Prerequisites for Provisioning a TDE-enabled vPDB to a Linked CDB

The same prerequisites that apply for provisioning a regular vPDB to a Linked CDB, also apply to a TDE-enabled vPDB. Specifically, there must be a target environment that has an Oracle installation compatible with the Oracle installation of the source CDB and the source PDB. The following database requirements are needed to provision the vPDB to a linked CDB. For more information, seeRequirements for Oracle Hosts and Databases.

- Recommend autodiscovery so that the linked CDB can be found. Otherwise, the linked CDB must be manually discovered before provisioning.
- Linked CDB must be running
- Linked CDB must be in ARCHIVELOG mode
- Linked CDB should be using Block Change Tracking (BCT)
- LogSync must be enabled for the Linked CDB.

The additional prerequisites for provisioning a TDE-enabled vPDB are:

- The target keystore password must be entered on the Environments page when provisioning to a target linked CDB. Follow the procedure listed in the Adding or Editing the Target Keystore Password section.
- A copy of the source database wallet must be made available on the target host(s) and that path must be specified as the parent database TDE keystore location. This can be achieved either through copying the entire keystore (`ewallet.p12`) file from the source host, or by creating a new keystore and importing the required keys from the source keystore.
- The parent database TDE keystore location on the target host(s) must be accessible to the Environment User.
- If the target CDB is a new vCDB, the Environment User must have permission to write to the “Target vCDB TDE Keystore Location”.

Compatibility requirements for provisioning a TDE-enabled vPDB to a Linked CDB

For provisioning a vPDB into a Linked CDB, the source CDB and the target CDB must meet the following compatibility requirements:

- The value of the COMPATIBLE parameter in the source CDB must be less than or equal to the value of the COMPATIBLE parameter in the target CDB.
- They must have the same endianness.
- They must have compatible character sets and national character sets.
- If TDE is configured using `sqlnet.ora` , the encryption wallet location should be configured on the target host in the standard location as described in the Oracle documentation. If the `TNS_ADMIN` environment variable is being used to specify the directory location where the `sqlnet.ora` is located, `TNS_ADMIN` should point to the default location as indicated above.

Setting Up Auxiliary CDB Parameters

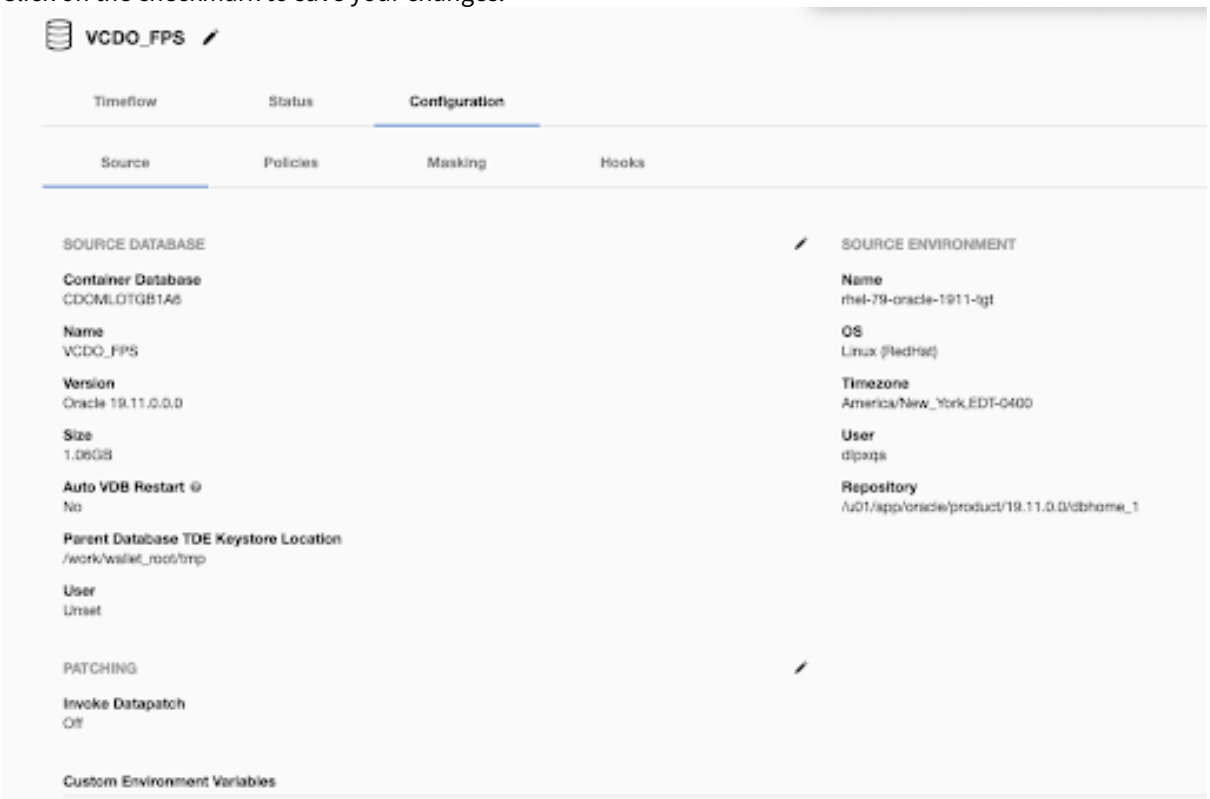
During a TDE-enabled vPDB provisioning into a Linked CDB or a vCDB, the Delphix Engine creates a temporary CDB instance (or Auxiliary CDB) on the target environment to recover the vPDB to a consistent state. This temporary CDB will be automatically deleted (in case of provisioning to a linked CDB or to an existing vCDB) or converted to a vCDB (in case of provisioning to a new vCDB) after the vPDB is provisioned successfully. By default, this temporary CDB is configured to have the same init parameters as the source database. To manage the configuration of the temporary CDB init parameters, you must set up Repository Templates for Oracle Databases prior to provisioning.

For more details on additional TDE-related processing that is performed in the Auxiliary CDB, see the A closer look at TDE provisioning section.

Updating the Parent Database TDE Keystore Location

The Parent Database TDE Keystore Location is specified in the GUI in the Configuration tab under the vPDB in the Source tab.

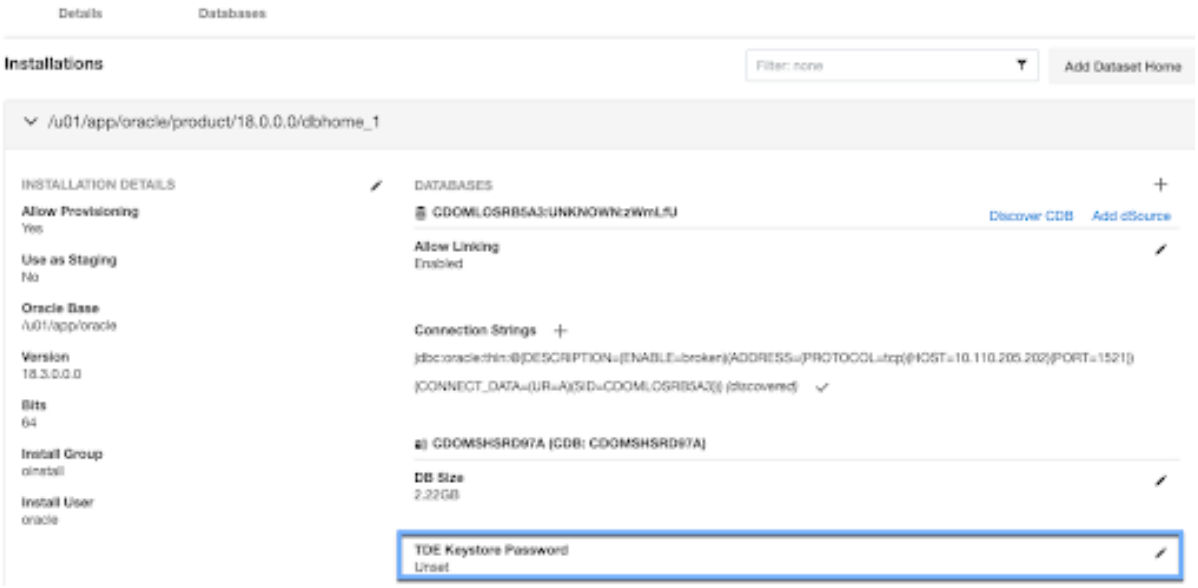
1. Login to the **Delphix Management** application.
2. Click **Manage > Datasets.**
3. Click on the vPDB.
4. Click the **Configuration** tab.
5. Click the **Source** tab.
6. Next to Source Database, click on the pencil icon to update the **Parent Database TDE Keystore Location**.
7. Click on the checkmark to save your changes.



Adding or Editing the Target Keystore Password

The target keystore password is specified in the GUI in the Databases tab under Environments:

1. Login to the **Delphix Management** application.
2. Click **Manage > Environments**.
3. Click the **Databases** tab for your Environment.
4. Next to **TDE Keystore Password** click on the pencil icon to set or update the password.

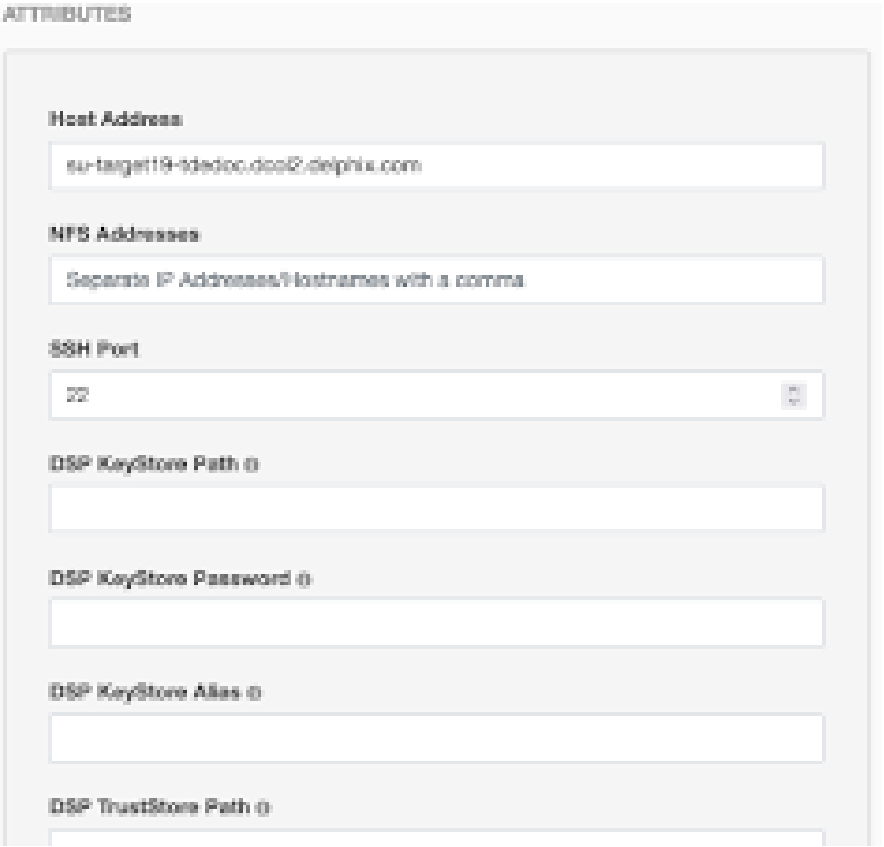


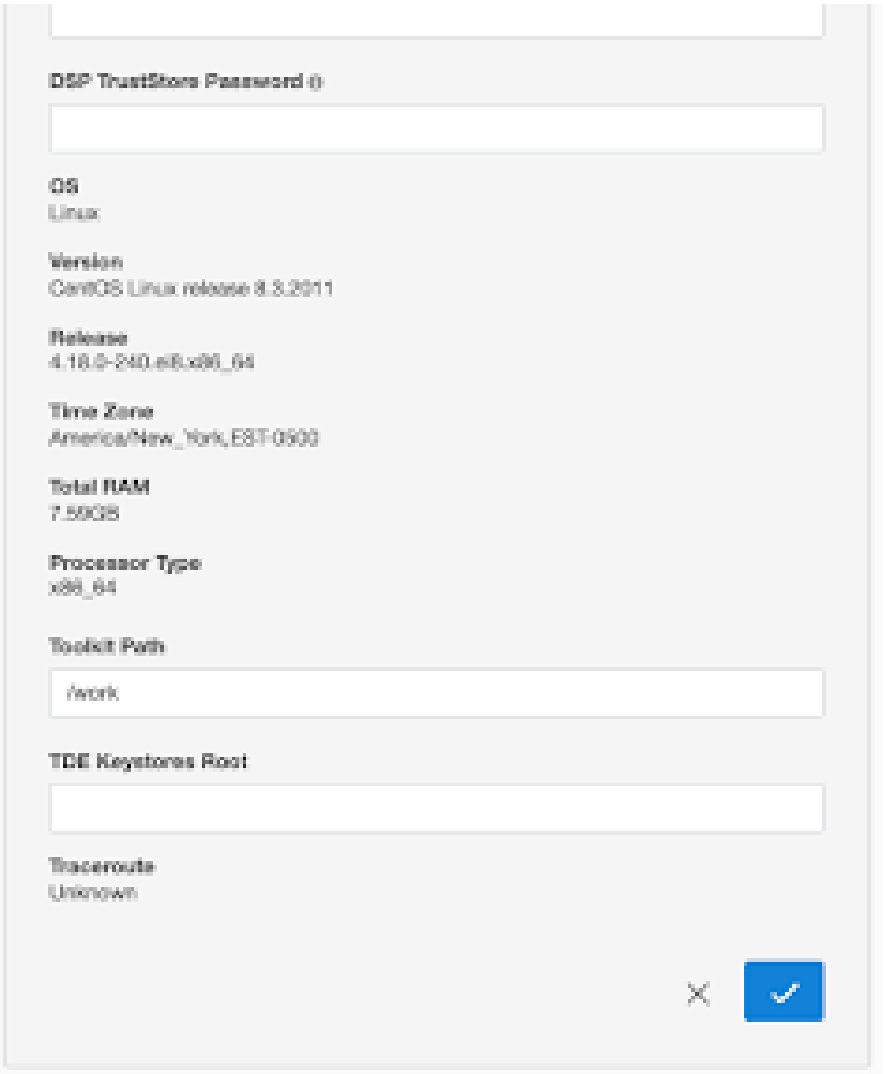
Adding or Editing the TDE Keystores Root

The TDE keystores root directory path is specified in the GUI in the Details tab under Environments:

1. Login to the **Delphix Management** application.
2. Click **Manage → Environments**.
3. Click the **Details** tab for your target environment.

Next to **Attributes** click on the pencil icon to set or update attributes, including the keystores root.





Provisioning a TDE-enabled vPDB

When provisioning a TDE-enabled vPDB into a Linked CDB or an existing vCDB:

- Delphix requires that the Linked CDB or vCDB already be present on the target host with all the encryption keys set up correctly.

When provisioning a TDE-enabled vPDB into a new vCDB:

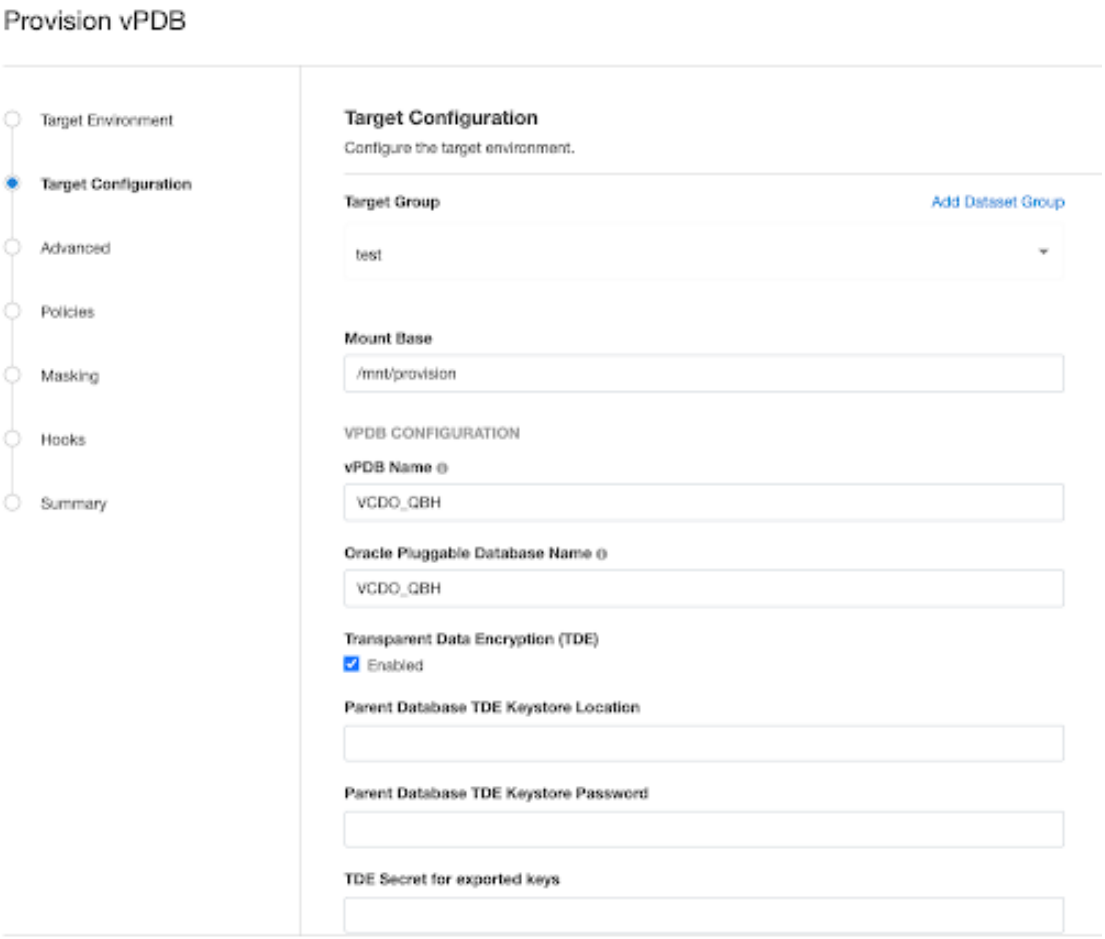
- Delphix will provision a vCDB from the source CDB to host the vPDB you are about to create. The newly-created vCDB will be configured with TDE enabled.
- There is no need to set up a repository template, as this provision workflow allows a user to directly specify a VDB configuration template, or set individual database parameters.

Procedure

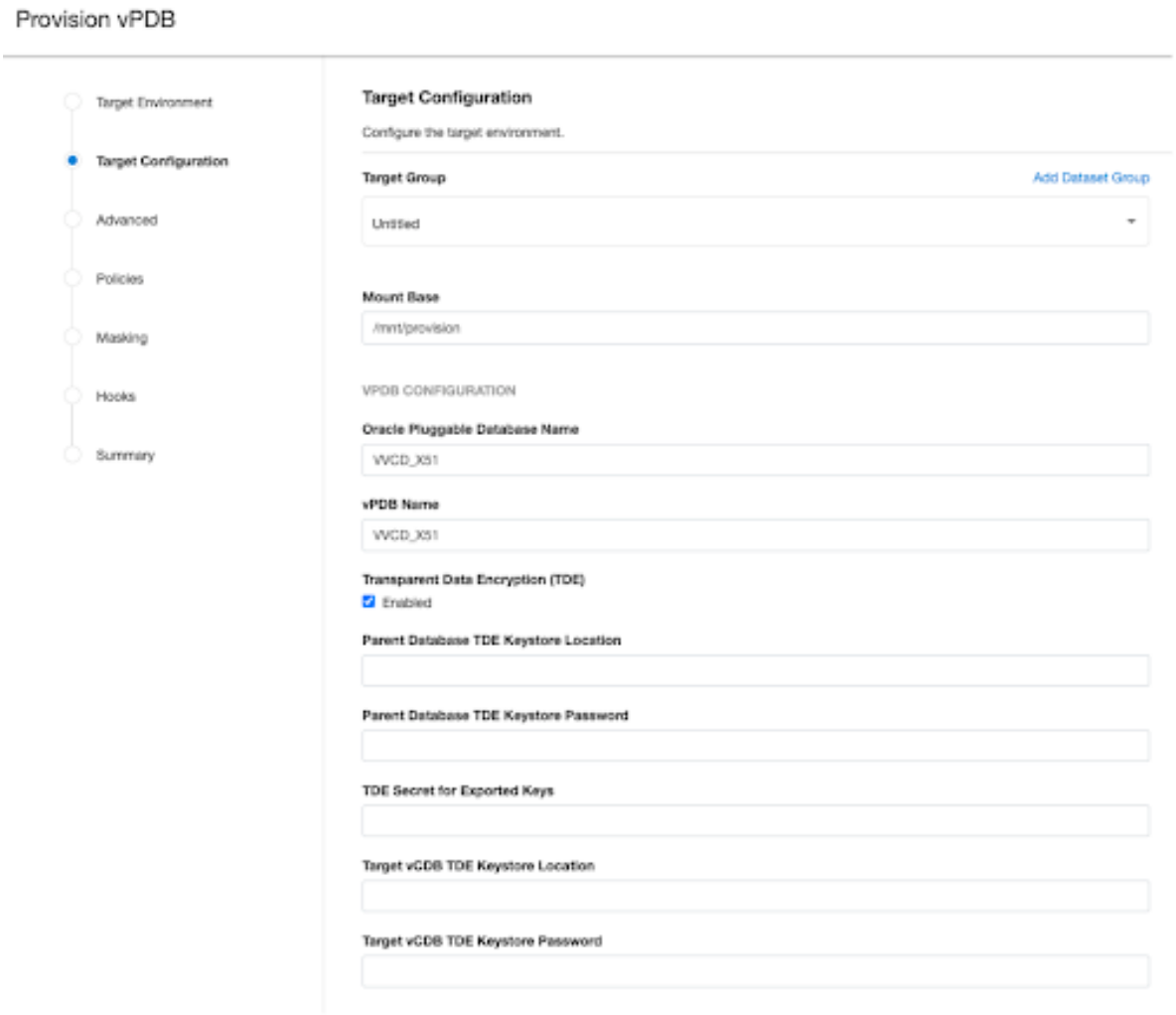
1. If you're provisioning to a Linked CDB for the first time, add the Target Keystore Password following the steps listed in the Adding or Editing the Target Keystore Password section.

> To validate that the parent database TDE keystore password is correct, refer to the Validating the Keystore Password section.

2. Ensure that the parent database TDE keystore location on the target host contains a keystore that includes the encryption keys from the source database and is readable by the Oracle User used for provisioning.
3. If provisioning to a cluster target, ensure that the keystores root directory path is set correctly following the steps listed in the Adding or Editing the TDE Keystores Root section.
4. In the **Datasets panel**, select an Oracle TDE-enabled PDB dSource or a previously provisioned TDE-enabled vPDB.
5. From the **Timeflow** tab, select a snapshot or point in time to provision from.
6. Once the Provision wizard is open, you can either provision with a:
    a. Target Linked CDB: Select an existing container database as the provision target CDB from the **Container Database** drop-down menu of CDBs on that environment.
    b. Existing vCDB: Select an existing vCDB as the provision target CDB from the **Container Database** drop-down menu of CDBs on that environment. (Supported only for Oracle versions 12.1.0.2 and later.)
    c. New vCDB: Select the **Create a New Container Database** checkbox. This will create a new vCDB object in that environment with this new vPDB plugged into it.
7. Click **Next** to advance the left-hand pane to the **Target Configuration** tab, and edit as necessary.
8. Enter the target Group for the vPDB you are about to provision.
9. The Environment User must have permission to write to the specified Mount Base, as described in Requirements for Oracle Environments and Data.
   You can also reuse the Delphix toolkit directory, which already exists as the Mount Base, or create a new writable directory in the target environment with the correct permissions and use that as the Mount Base.
10. Enter the vPDB Name and the Oracle Pluggable Database Name.
11. Click on the "**Transparent Data Encryption (TDE) Enabled"** checkbox. The following three fields need to be specified during the vPDB provision. Refer to the Provisioning parameters for a TDE-enabled vPDB section for important information on these three fields.
    a. Parent Database TDE Keystore Location - Specify the path to a keystore that contains the keys used to encrypt the source database datafiles. As noted earlier, the wallet `ewallet.p12` must exist in the specified path and must be readable by the Oracle user. **Warning:** Parent Database TDE keystore may be located on local storage, NFS or ACFS, but **cannot** be located on an ASM filesystem.
    b. Parent Database TDE Keystore Password - Specify the password for the parent database TDE keystore.
    c. TDE Secret for Exported Keys - Specify the password for the exported keys. **Warning:** Make sure the TDE Secret for Exported Keys is stored in a secure location for your records. It is only known to you. In the rare event that keys need to be manually extracted from an exported keyfile, this password will be required. Delphix Support cannot assist with manually exporting keys without this password, therefore it should be known or recorded within your organization.



12. If the target CDB is a vCDB, two additional necessary fields need to be specified - “Target vCDB TDE Keystore Location” and “Target vCDB TDE Keystore Password”. Please note that the Environment User must have permission to write to the “Target vCDB TDE Keystore Location” otherwise the provisioning will fail. Refer to the Provisioning parameters for a TDE-enabled vPDB section for important information on these two fields.



13. If you selected to create a new target vCDB, configure the vCDB:
    a. Enter the vCDB Name, Database Unique Name, and Database Name for the vCDB you are about to provision.
    b. Select the Configure vCDB Parameters checkbox if you want to use a VDB Configuration Template. See Customizing Oracle VDB Configuration Settings.
14. Click **Next** to advance the left-hand pane to the Advanced tab.
15. The available options are Auto vCDB Restart, Auto vPDB Restart, File Mapping, and custom environment variables.
    For more information, see Customizing VDB File Mappings and Customizing Oracle VDB Environment Variables.
16. Click **Next** to advance the left-hand pane to the Policies tab.
17. Select the VDB Snapshot policy to be applied to the vPDB.
    Select a Retention Policy for the vCDB, if you are provisioning a vCDB.
18. Click **Next** to advance the left-hand pane to the Masking tab.
    Select the Mask this vPDB checkbox if you want to mask, and select the masking job to be applied.
19. Click **Next** to advance the left-hand pane to the Hooks tab, and create any hooks if necessary. For more information, see Hook Scripts for Automation and Customization.
20. Review the provisioning summary. Confirm all the fields are correct. Click **Submit** to proceed with provisioning the vPDB.

TDE Keystores Root and Artifact Directory

The artifact directory on the target host stores the exported keyfiles used during the workflows for TDE-enabled vPDBs. It is located under the keystores root, in the directory `oracle_tde_keystores`. Each TDE-enabled vPDB will have its own directory within the `oracle_tde_keystores` directory, identified by the vPDB name, group name, and a unique identifier, separated by an underscore. If the keystores root directory is not specified, then it defaults to the toolkit directory path.

For example, if the keystores root directory is `/work` (or keystores root is not specified, and the toolkit directory is `/toolkit`), the artifact directory for the vPDB `tde_vpdb` in the group `Encrypted` could be

`/toolkit/oracle_tde_keystores/tde_vpdb_Encrypted_ce7a47e6-8860-4398-bab0-cf0233fc5e3c`

Within the artifact directory, there is a subdirectory `exported_keys` which contains within it the exported keyfiles for each timeflow associated with that vPDB. Each time an export is performed, a new exported keyfile is generated with a timestamp. The contents of the artifact directory may change in future releases, but the path to the artifact directory and the naming convention is not anticipated to change.

As the default keystores root directory is at the same level as the toolkit directory, it will not be overwritten if a host is refreshed through the Delphix Engine and the toolkit updated. It is the customer's responsibility to backup the keystores root directory and ensure that the contents are not lost, as a disk failure could prevent a TDE-enabled vPDB from being accessed. The Delphix Engine never keeps a copy of the keystores or exported keyfiles on Delphix storage. Thus it is recommended that the keystores root directory be on a disk which is regularly backed up.

The artifact directory is not removed when a TDE-enabled vPDB is deleted; the customer can remove it after confirming that the vPDB has been removed (including from any replicated Delphix Engines).

> **Warning**
> The Delphix engine does not keep a copy of the keystores or exported keyfiles on Delphix storage and thus a disk failure could prevent a TDE-enabled vPDB from being accessed. Therefore, Delphix highly recommends that the TDE keystores root directory be backed up at regular intervals.

Other operations on a TDE-enabled vPDB

Once a TDE-enabled vPDB is provisioned, it can be used the same as a non-TDE-enabled vPDB within Delphix, with the exception of migrate which is separately covered below. There are however a few caveats or behavioral differences as follows:

- Refreshing a TDE-enabled vPDB will use the parent keystore for the recovery. If the parent PDB's master keys are changed, the user will need to update the parent keystore with the new keys, for example by re-copying the parent PDB's `ewallet.p12` file to the parent keystore location on the target host. Similarly, if the location or password to the parent keystore has changed then they should be updated before the refresh.
- Rewinding a TDE-enabled vPDB will use the target keystore for the recovery. If the vPDB is rekeyed after it is provisioned, then the rekey will update the target keystore, so it does not need to be updated in Delphix.
- For a single vPDB in a vCDB, if the Target vCDB TDE Keystore Location is changed, the new path must be updated in Delphix before refresh or rewind.
- Disabling a TDE-enabled vPDB will result in the keys being exported to an exported keyfile in the artifact directory, to be used for a subsequent enable. Refresh and rewind operations will first disable the existing vPDB, so those will also result in a new exported keyfile in the artifact directory.
- Provisioning a second-generation vPDB (vvPDB) from a TDE-enabled vPDB is done in the same manner as a first-generation vPDB, by specifying the TDE parameters during provision. The parent database TDE keystore location for the vPDB can be specified as the parent database TDE keystore location for the vvPDB.

If a vPDB is moved to a different host (either through the migrate workflow or an enable after a failover, then the artifact directory will need to be copied to the new target host. See Migrating a TDE-enabled vPDB for details on the manual steps needed for migration.

A closer look at TDE provisioning

> The following is intended to provide a deeper understanding of the Delphix TDE implementation, and is not required for a general understanding of the general provisioning workflows.

Delphix Provisioning workflow with and without TDE

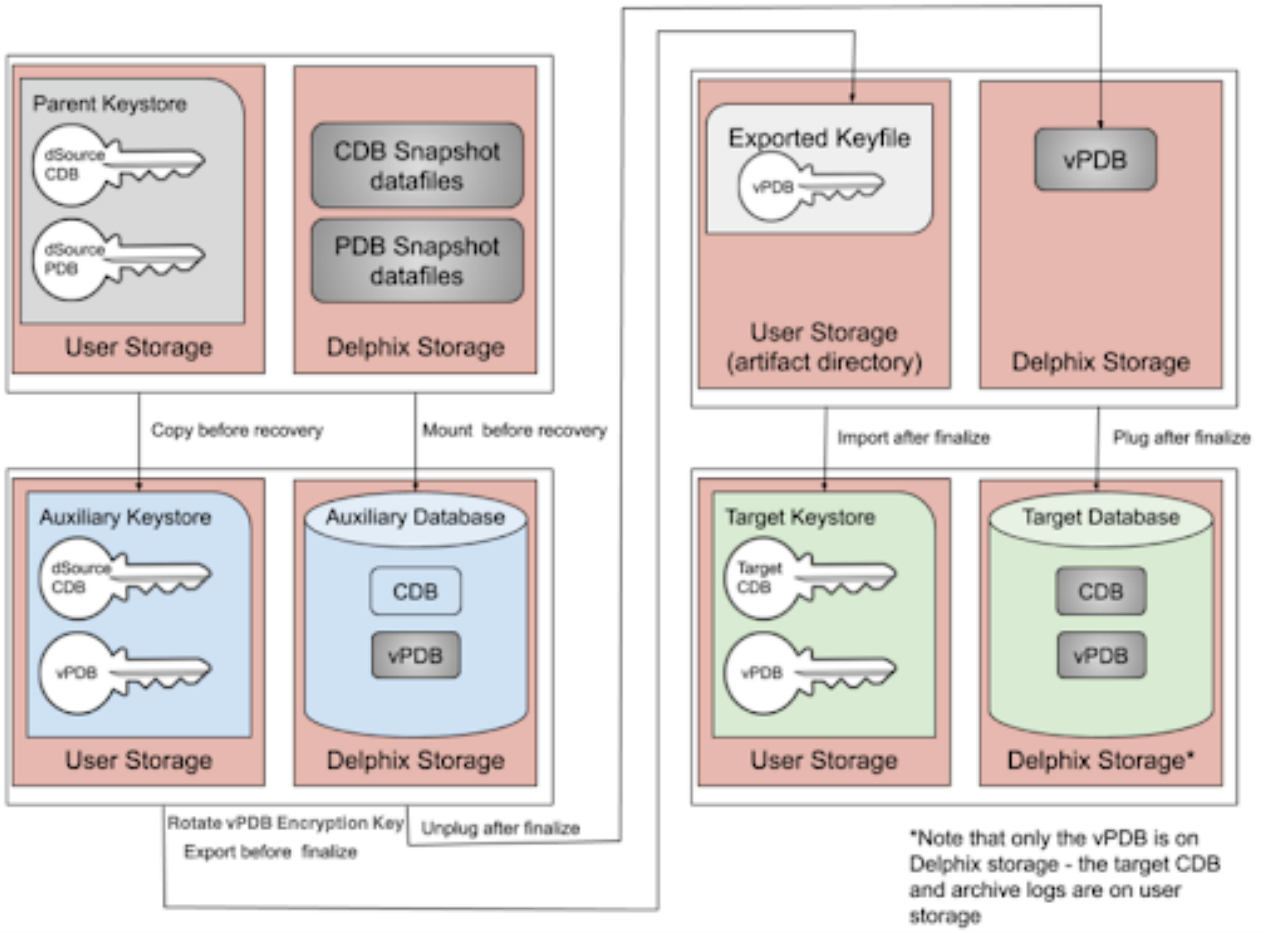
Provisioning a Virtual Pluggable Database (vPDB) first involves using the GUI or CLI to specify the vPDB parameters (such as the vPDB name and target container) along with the snapshot to provision from. Once the provision job is started with these parameters, the Delphix Engine does the following:

1. Mounts the snapshot files on the target host.
2. Creates and opens (in mount mode) the auxiliary container database on the target host, using the snapshot files. The auxiliary container database will have both the CDB and PDB data files from the dSource.
3. Completes recovery to bring the auxiliary container database into a consistent state.
4. Finalizes the state of the auxiliary database and unplugs the vPDB datafiles.
5. Plugs the vPDB into the target database, and opens it in read-write mode for general use.

If the dSource is TDE-enabled, then Delphix will need to perform additional operations to complete the provision of a TDE-enabled vPDB to a TDE-enabled target container (indicated in red):

1. Mounts the snapshot files on the target host.
2. Creates and opens (in mount mode) the auxiliary container database on the target host, using the snapshot files. The auxiliary container database will have both the CDB and PDB data files from the dSource.
3. Creates a keystore for the auxiliary container database with the necessary keys to apply encrypted archived log files.
4. Completes recovery to bring the auxiliary container database into a consistent state.
5. Rotates the vPDB and auxiliary CDB master encryption keys by generating new keys that are unique to the vPDB / auxiliary CDB and not associated with the source PDB or CDB.
6. Exports only the newly generated keys to an exported keyfile to enable unplug.
7. Finalizes the state of the auxiliary database and unplugs the vPDB datafiles.
8. Imports the keys from the exported keyfile into the target keystore.
9. When provisioning to a vCDB target, converts the auxiliary CDB into the final vCDB and creates the vCDB keystore from the auxiliary CDB keystore.
10. Plugs the vPDB into the target database, and opens it in read-write mode for general use.

The following diagram illustrates the provisioning steps.



At each stage of the provisioning process, the keys and exported keyfiles are always on user storage. The exported keyfile is located in the artifact directory, while the auxiliary and target keystores are in the auxiliary keystores directory. Both the artifact directory and auxiliary keystores directory are subdirectories of the TDE keystores root directory, which is either user specified, or if not specified defaults to the toolkit root directory. Similar to non-TDE-enabled vPDBs, the final vPDB (and vCDB, if applicable) is on Delphix storage while the target Linked CDB and its archive logs remain on user storage.

Validating the keystore password

To validate that the password for a given keystore is correct, Oracle provides the `mkstore` command-line utility. Navigate to the keystore folder on the target host where `ewallet.p12` exists and run the command `mkstore -wrl . list` as the Oracle user. If the password is incorrect, you will see output similar to the following:

```
$ mkstore -wrl . -list
Oracle Secret Store Tool Release 19.0.0.0.0 - Production
Version 19.4.0.0.0
Copyright (c) 2004, 2021, Oracle and/or its affiliates. All rights reserved.

Enter wallet password:   xxxxxxxx
oracle.security.crypto.core.CipherException: Invalid padding string (or incorrect
password)
```

If the password is correct, you will see output similar to the following:

```
$ mkstore -wrl . -list
Oracle Secret Store Tool Release 19.0.0.0.0 - Production
Version 19.4.0.0.0
Copyright (c) 2004, 2021, Oracle and/or its affiliates. All rights reserved.

Enter wallet password: xxxxxxxx
Oracle Secret Store entries:
ORACLE.SECURITY.DB.ENCRYPTION.AQtDPGje509PvxYeQuG7fmYAAAAAAAAAAAAAAAAAAAAAAAAAAAA
ORACLE.SECURITY.DB.ENCRYPTION.AVXDkpbWnE9rv8fzTPiHTXcAAAAAAAAAAAAAAAAAAAAAAAAAAAA
ORACLE.SECURITY.DB.ENCRYPTION.AWS1QwdQHk/jv0i5eJ3sb10AAAAAAAAAAAAAAAAAAAAAAAAAAAA
ORACLE.SECURITY.DB.ENCRYPTION.AZcpc0/QrU/Nv/Q54WRWRSMAAAAAAAAAAAAAAAAAAAAAAAAAAAA
ORACLE.SECURITY.DB.ENCRYPTION.MASTERKEY
ORACLE.SECURITY.DB.ENCRYPTION.MASTERKEY.C204B25A34B93EB7E055000000000001
ORACLE.SECURITY.DB.ENCRYPTION.MASTERKEY.C204B30EA4A03FABE055000000000001
ORACLE.SECURITY.DB.ENCRYPTION.MASTERKEY.C204B407D46A3FFDE055000000000001
ORACLE.SECURITY.ID.ENCRYPTION.
ORACLE.SECURITY.KB.ENCRYPTION.
ORACLE.SECURITY.KM.ENCRYPTION.AQtDPGje509PvxYeQuG7fmYAAAAAAAAAAAAAAAAAAAAAAAAAAAA
ORACLE.SECURITY.KM.ENCRYPTION.AVXDkpbWnE9rv8fzTPiHTXcAAAAAAAAAAAAAAAAAAAAAAAAAAAA
ORACLE.SECURITY.KM.ENCRYPTION.AWS1QwdQHk/jv0i5eJ3sb10AAAAAAAAAAAAAAAAAAAAAAAAAAAA
ORACLE.SECURITY.KM.ENCRYPTION.AZcpc0/QrU/Nv/Q54WRWRSMAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

Example Illustrating Movement of Encryption Keys

This example illustrates the movement of encryption keys that occur as a result of provisioning a TDE-enabled vPDB. Consider a vPDB `tde_vpdb` that is provisioned from a dSource `CDOMSHSR52CAPDB2` on the VM `tde-source18`, which is an Oracle database running version 18.11.0. Connecting to this database, we can query `v$encryption_keys` to determine the current keys in use by each PDB:

```
SQL> show pdbs
CON_ID CON_NAME                       OPEN MODE  RESTRICTED
------ ------------------------------ ---------- ----------
     2 PDB$SEED                       READ ONLY  NO
     3 CDOMSHSR52CAPDB1                 READ WRITE NO
     4 CDOMSHSR52CAPDB2                 READ WRITE NO
     5 CDOMSHSR52CAPDB3                 READ WRITE NO
SQL> select con_id, key_id from v$encryption_keys order by con_id;
CON_ID KEY_ID
------ ------------------------------------------------------------
     1 Ac9MY5kQwU8GvwlYMXImXmMAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
     3 AedrXL3aUk9zv+9t7J8ZsVYAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
     4 AdDdKibLKU9mv6PDAIvVvH0AAAAAAAAAAAAAAAAAAAAAAAAAAAAA
     5 AWdc3ZRaP09Pvw4+2FmLwHAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

The `v$encryption_keys` output for this environment shows that there are 3 PDBs within this CDB, all of which are TDE-enabled. In particular, the PDB used for the dSource has a `con_id` of 4, and an encryption `key_id` starting with **AdDdKibL**.

The vPDB `tde_vpdb` is provisioned to the CDB `CDOMSHTG93CF` on the VM `tde-target18`. Connecting to this database, we can again query `v$encryption_keys` to determine the keys in use by each PDB:

```
SQL> show pdbs
CON_ID CON_NAME         OPEN MODE  RESTRICTED
------ ---------------- ---------- ----------
     2 PDB$SEED         READ ONLY  NO
     3 CDOMSHTG93CFPDB1 READ WRITE NO
     4 CDOMSHTG93CFPDB2 READ WRITE NO
     5 CDOMSHTG93CFPDB3 READ WRITE NO
     6 TDE_VPDB         READ WRITE NO
SQL> select con_id, key_id from v$encryption_keys order by con_id;
CON_ID KEY_ID
------ ----------------------------------------------------------
     1 AZTc9eKqlk98v8GkQ8/AmaAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
     6 AdDdKibLKU9mv6PDAIvVvH0AAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

The key which was originally present in the wallet on the dSource - **AdDdKibL** - has been imported into the target keystore, and has a `con_id` of 6, which corresponds to the `con_id` of the vPDB. There are several things to note about the behavior of Oracle and the `v$encryption_keys` table:

1. Keys are never deleted from existing keystores by Oracle, only new keys are added. Therefore, if we were to disable the vPDB, which will unmount and unplug it from the CDB, `v$encryption_keys` will still show the key as present, with its original `con_id`, even though it has been unplugged:

   ```
   SQL> show pdbs
   CON_ID CON_NAME         OPEN MODE  RESTRICTED
   ------ ---------------- ---------- ----------
        2 PDB$SEED          READ ONLY  NO
        3 CDOMSHTG93CFPDB1  READ WRITE NO
        4 CDOMSHTG93CFPDB2  READ WRITE NO
        5 CDOMSHTG93CFPDB3  READ WRITE NO
   SQL> select con_id, key_id from v$encryption_keys order by con_id;
   CON_ID KEY_ID
   ------ ----------------------------------------------------------
        1 AZTc9eKqlk98v8GkQ8/AmaAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
        6 AdDdKibLKU9mv6PDAIvVvH0AAAAAAAAAAAAAAAAAAAAAAAAAAAAA
   ```

2. If the wallet is closed for a particular PDB, `v$encryption_keys` will not show any entries for that PDB. The wallet status can be determined by querying `v$encryption_wallet`.
3. Querying `v$encryption_wallet` while the session is attached to `CDB$ROOT` will return information about the entire CDB, otherwise, only the keys for the current PDB are returned.

Provisioning a vPDB from a non-multitenant source

Delphix supports provisioning a virtual pluggable database (vPDB) from a snapshot of a non-multitenant (non-MT) source database (VDB). This feature is only available through the API or command-line interface (CLI). This topic describes how to provision a vPDB from a snapshot of a non-MT VDB (also simply referred below as source VDB) using CLI.

The high-level workflow for the provisioning is as follows:

This feature has the following restrictions:

1. Transparent Data Encryption (TDE) is not supported.
2. The provision point must correspond to a snapshot. Provisioning from a point in time between snapshots is not supported.
3. The target CDB (where the new vPDB will be plugged into) must be a physical CDB. Virtual CDB targets are not supported.
4. For provisioning from a source VDB of Oracle 11g version, the VDB must be upgraded before provisioning (i.e. you must use the Upgrade Option 1 described below). Upgrade option 2 cannot be used in this case.

- Choose the upgrade option to use if the target CDB (where the new vPDB will be plugged into) version is newer than the source VDB. Refer to the prerequisites section below.
- Create the required hook scripts.
- Take source VDB snapshot.
- Provision the vPDB to the target CDB using the VDB snapshot.

Prerequisites

Environment requirements

Provisioning a vPDB from a non-MT source has the following environment requirements:

- Source host with a non-MT Oracle 11g or newer source database.
- VDB Target host for provisioning a non-MT VDB from the source database.
- CDB target host with a running Oracle target version CDB. The target CDB will be automatically linked if it is not already linked.

Select upgrade option

The target CDB can be a newer Oracle version than the source database (for example, the source is 12.2 and target is 19c). When an upgrade is also required, there are two options for upgrading the database:

- Upgrade Option 1: After source VDB is provisioned and before vPDB is provisioned via CLI. This upgrade is done manually by the user, before initiating the vPDB provisioning API/CLI. This option requires the ability to upgrade to the Oracle target version on the VDB target host.
- Upgrade Option 2: After plugging the newly provisioned vPDB into the target CDB database. This upgrade is performed by Delphix API/CLI using a hook script.
  If the source VDB's Oracle version is 11g, Upgrade Option 1 must be selected. For source VDB's with Oracle versions 12c and above, either option may be selected.

Prepare hook scripts

There are following three scripts used during this procedure and must be created manually before executing the vPDB provisioning command by CLI:

1. **Pre-snapshot Hook on source VDB**: This hook will open the database in "read only" mode and issue a call to the `dbms_pdb.describe` procedure to generate an XML file called delphix_plugin.xml, describing the VDB. The XML file will be used to plug the source VDB data files into the target CDB. The source VDB must be open read only during the subsequent snapshot so that the VDB data files do not require recovery when plugging them into the target CDB. This can be added using Delphix Management Application UI from the Datasets panel by selecting the source VDB and then going to Configuration -> Hooks tab.
2. **Post-snapshot Hook on source VDB**: This hook will return the VDB to "read write" mode. This is an optional script. The source VDB can also remain read only. This can be added using Delphix Management Application UI from the Datasets panel by selecting the source VDB and then going to Configuration -> Hooks tab.
3. **Post-plug Hook for vPDB**: This script will run as a hook present on the Linked CDB target host and will be executed after the newly provisioned vPDB is plugged into the target CDB.
   a. If no upgrade is required or using Upgrade Option 1, then this script will call the Oracle script `$ORACLE_HOME/rdbms/admin/noncdb_to_pdb.sql` to convert the VDB into a PDB. b. If using Upgrade Option 2, this script will upgrade the database and then call Oracle script `$ORACLE_HOME/rdbms/admin/noncdb_to_pdb.sql`.
   This script must be created manually and stored in the root folder of the Delphix Toolkit directory of the target CDB host. The name of the vPDB being provisioned/converted will be supplied by Delphix as the first parameter to the script when it invokes the script. The VDB data files will already be plugged into the Linked CDB target at the time the script is invoked.

Refer to the Sample Scripts section below for the content of scripts.

> Note about the sample scripts provided in this document:
>
> - These scripts are provided as-is, without warranty of any kind or commercial support through Delphix.
> - The scripts may need to be modified depending on the Oracle version or the SQL script package version being used.

Workflow

1. Link the non-MT source database as a dSource within Delphix.
2. Provision a non-MT Oracle VDB from the dSource onto the VDB target host. This will be referred to as the **Golden VDB**.
3. **If no upgrade is required:** a. Create a Pre-snapshot hook on the **Golden VDB.** b. (Optional) Create a Post-snapshot hook on the **Golden VDB**. c. Take a snapshot of the **Golden VDB**. d. Create a PDB conversion script named `dx-post-plug-hook.sh` in the root of the Delphix toolkit directory of the Linked CDB target host.
   If using Upgrade Option 1: a. Upgrade the **Golden VDB** to the Oracle target version: manually upgrade the database and point it to the new Oracle home. b. Create a Pre-snapshot hook on the **Golden VDB.** c. (Optional) Create a Post-snapshot hook on the **Golden VDB**. d. Take a snapshot of the **Golden VDB**. e. Create a PDB conversion script named `dx-post-plug-hook.sh` in the root of the Delphix toolkit directory of the Linked CDB target host.
   If using Upgrade Option 2: a. Create a Pre-snapshot hook on the **Golden VDB.** b. (Optional) Create a Post-snapshot hook on the **Golden VDB**. c. Take a snapshot of the **Golden VDB**. d. Create a PDB Upgrade and Conversion script named `dx-post-plug-hook.sh` in the root of the Delphix toolkit directory of the Linked CDB target host.
4. Select the **snapshot** on the **Golden VDB** created above and provision a vPDB to the Linked CDB target. The detailed steps for this are documented in the next section.

This step can be executed via the **API / CLI only**, and will not be allowed via the Delphix UI.

CLI procedure to provision a vPDB from a VDB

1. Log into the Delphix command-line interface using the admin user or a user with admin privileges.
   $ ssh admin@YOUR_ENGINE
2. Move to the database provisioning command line object.

```
delphix> database provision
```

3. Set the parameter type to `OracleMultitenantProvisionParameters`.

```
set type=OracleMultitenantProvisionParameters
```

4. (Optional) Set the login details for the provision and Delphix OS user who is to perform the provision.

```
delphix database provision *> set username=delphix
delphix database provision *> set credential.type=PasswordCredential
delphix database provision *> set credential.password=delphix
```

5. Give the dataset a name.

```
delphix database provision *> set container.name=vpdb
```

6. Place the new dataset in a Group that appears in the Delphix GUI, in this case, the Targets group.

```
delphix database provision *> set container.group=Targets
```

7. Set the destination mount point which Delphix NFS mounts are to be linked to under the virtual PDB. This folder must exist at a file system level on the Linked CDB target host. Do not use single quotes around the mount path.

```
delphix database provision *> set source.mountBase="/mnt/provision"
```

8. If automatically restarting the vPDB is not required after a reboot of the Linked CDB target host, set this to option to false. False is possibly a better option given the container database would need to be running prior to any attempt to pull up a vPDB.

```
delphix database provision *> set source.allowAutoVDBRestartOnHostReboot=false
```

9. Supply the destination container database name. The container database should already be discovered. This will be where the vPDB will ultimately be placed.

```
delphix database provision *> set sourceConfig.cdbConfig=CDBSTAGE
```

10. Name the vPDB. This is what it will appear as in the destination container database.

```
delphix database provision *> set sourceConfig.databaseName=vpdb
```

11. Supply the source **Golden VDB** details. In this example, the provision will use the latest snapshot available from the **Golden VDB** as the point in time from which to provision the vPDB. A specific snapshot can also be picked, but an arbitrary point in time is not supported.
delphix database provision *> set timeflowPointParameters.type=TimeflowPointSemantic delphix database provision *> set timeflowPointParameters.container=gold_vdb delphix database provision *> set timeflowPointParameters.location=LATEST_SNAPSHOT
12. Check that all the settings you require are in place using the "ls" command.

```
delphix database provision *> ls
Properties
  type: OracleMultitenantProvisionParameters
  container:
    type: OracleDatabaseContainer
    name: vpdb (*)
    description: (unset)
    diagnoseNoLoggingFaults: true
    group: Targets (*)
    performanceMode: DISABLED
    preProvisioningEnabled: false
    sourcingPolicy: (unset)
  credential:
    type: PasswordCredential (*)
    password: ******** (*)
  masked: (unset)
  maskingJob: (unset)
  source:
    type: OracleVirtualPdbSource (*)
    name: (unset)
    allowAutoVDBRestartOnHostReboot: false (*)
    config: (unset)
    customEnvVars: (unset)
    fileMappingRules: (unset)
    LogCollectionEnabled: false
    mountBase: /mnt/provision (*)
    operations: (unset)
    parentTdeKeystorePassword: (unset)
    parentTdeKeystorePath: (unset)
    tdeExportedKeyFileSecret: (unset)
  sourceConfig:
    type: OraclePDBConfig
    cdbConfig: CDBSTAGE (*)
    databaseName: vpdb (*)
    environmentUser: (unset)
    linkingEnabled: true
    nonSysCredentials: (unset)
    nonSysUser: (unset)
    repository: (unset)
    services: (unset)
  timeflowPointParameters:
```

```
    type: TimeflowPointSemantic
    container: gold_vdb (*)
    location: LATEST_SNAPSHOT (*)
  username: delphix (*)
  VirtualCdb: (unset)

Operations
defaults
```

13. Initiate the provision by committing the operation in the CLI.

```
delphix database provision *> commit
  vpdb
  Dispatched job JOB-333
  DB_PROVISION job started for "Targets/vpdb".
  Starting provision of virtual PDB database "vpdb" converted from a single
tenant database.
  Preparing multitenant container database "CDBSTAGE".
  Creating new TimeFlow.
  Generating recovery scripts.
  Exporting storage.
  Preparing XML manifest file prior to plugin.
  Plugging in Oracle pluggable database.
  Running user-defined post plug hook.
  Opening Oracle pluggable database.
  Setting OMF destination for Oracle pluggable database.
  Creating PDB tempfiles.
  Checking Oracle pluggable database plugin violations.
  DB_PROVISION job for "Targets/vpdb" completed successfully.
```

To refresh the data in the vPDB from production, first, refresh the **Golden VDB** from the dSource, then refresh the vPDB from the new snapshot in the **Golden VDB**.

There are some workflow customizations required for RAC databases:

1. The PDB conversion script must be in the root of the Delphix toolkit directory for all the target CDB RAC instances.
2. The **Golden VDB** Pre-Snapshot hook, as provided below, will not work in a clustered (RAC) environment with more than one active instance because it only shuts down the local instance. `dbms_pdb.describe` will not execute while an instance is open read-write. The workarounds are:
    a. Provision the **Golden VDB** as single-instance, either by provisioning to a non-RAC target or by provisioning to a RAC target with only one active instance. The sample hook will work in this case.
    b. Write a customized pre-snapshot hook that shuts down all instances, restarts only one instance in read-only mode, and runs `dbms_pdb.describe`.
    c. Manually perform the actions of the hook: shutdown the **Golden VDB**, restart one of the instances in read-only mode and then run `dbms_pdb.describe`.

Sample scripts

Golden VDB pre-snapshot hook

Restarts the source/Golden VDB in read only mode and runs `dbms_pdb.describe` to generate an XML file describing the VDB. The XML file will be used to plug the VDB into the Linked CDB target. The target for the XML file must be $DELPHIX_MOUNT_PATH/$DELPHIX_DATABASE_UNIQUE_NAME/datafile/delphix_plugin.xml.

```
#!/bin/sh

sqlplus "/ AS SYSDBA" <<-EOF
  whenever sqlerror exit 2;
  spool $DELPHIX_MOUNT_PATH/$DELPHIX_DATABASE_UNIQUE_NAME/datafile/presnapshot.out
replace
  shutdown immediate
  startup mount
  alter database open read only;
  exec dbms_pdb.describe(pdb_descr_file=>'$DELPHIX_MOUNT_PATH/
$DELPHIX_DATABASE_UNIQUE_NAME/datafile/delphix_plugin.xml');
  exit;
EOF
```

Golden VDB post-snapshot hook

This is only necessary if the source VDB should not be left in read-only mode after the snapshot.

```
#!/bin/sh

sqlplus "/ AS SYSDBA" <<-EOF
  whenever sqlerror exit 2;
  spool $DELPHIX_MOUNT_PATH/$DELPHIX_DATABASE_UNIQUE_NAME/datafile/postsnapshot.out
replace
  shutdown immediate
  startup
  exit;
EOF
```

PDB conversion script

This script converts the source/Golden VDB datafiles into PDB datafiles. The script should be named `dx-post-plug-hook.sh` and reside in the root of the Delphix toolkit directory of the Linked CDB target host. Delphix will supply the name of the PDB being provisioned/converted as the first parameter.

The VDB datafiles will have already been plugged into the target CDB at the time the script is invoked and the virtual PDB will be in the mounted (not open) state. The PDB conversion script should return with the virtual PDB in either the mounted or open (not restricted) state. Delphix does not enforce a time-out for the script.

```
#!/bin/sh
```

```
DELPHIX_PDB_NAME=$1
SCRIPT_DIR="$( cd "$( dirname "$0" )" && pwd )"
CONVERT_LOGFILE=$SCRIPT_DIR/$DELPHIX_PDB_NAME-pdbconvert.log

sqlplus "/ AS SYSDBA" <<-EOF
  whenever sqlerror exit 2;
  spool $CONVERT_LOGFILE replace
  alter session set container=$DELPHIX_PDB_NAME;
  @?/rdbms/admin/noncdb_to_pdb.sql
  exit;
EOF
```

PDB upgrade and conversion script

This script upgrades the newly provisioned vPDB to the target CDB version and then converts the source/Golden VDB datafiles into PDB datafiles. The script should be named `dx-post-plug-hook.sh` and reside in the root of the Delphix toolkit directory of the Linked CDB target host. Delphix will supply the name of the PDB being provisioned/converted as the first parameter. The VDB datafiles will have already been plugged into the target CDB at the time the script is invoked and the virtual PDB will be in the mounted (not open) state. The PDB conversion script should return with the virtual PDB in either the mounted or open (not restricted) state. Delphix does not enforce a time-out for the script.

```
#!/bin/sh

DELPHIX_PDB_NAME=$1
SCRIPT_DIR="$( cd "$( dirname "$0" )" && pwd )"
UPGRADE_LOGFILE=$SCRIPT_DIR/$DELPHIX_PDB_NAME-dx-post-plug-upgrade.log
UPGRADE_LOGDIR=$SCRIPT_DIR/$DELPHIX_PDB_NAME-upgrade

mkdir $UPGRADE_LOGDIR
cd $ORACLE_HOME/rdbms/admin
switches="-c '$DELPHIX_PDB_NAME' -l $UPGRADE_LOGDIR"
$ORACLE_HOME/perl/bin/perl catctl.pl $switches catupgrd.sql &>> $UPGRADE_LOGFILE

CONVERT_LOGFILE=$SCRIPT_DIR/$DELPHIX_PDB_NAME-pdbconvert.log

sqlplus "/ AS SYSDBA" <<-EOF
  whenever sqlerror exit 2;
  spool $CONVERT_LOGFILE replace
  alter session set container=$DELPHIX_PDB_NAME;
  @?/rdbms/admin/noncdb_to_pdb.sql
  exit;
EOF
```

Provisioning a VDB from an encrypted Oracle database

Overview

This topic describes how to provision a VDB from an encrypted database. The Delphix Engine supports provisioning from a dSource linked to a physical database that has been encrypted with Oracle's Transparent Database Encryption (TDE), which can be used to encrypt columns or tablespaces.

> The Delphix engine does not support provisioning from a dSource with an encrypted system tablespace in a non-multitenant configuration.

Provisioning a VDB from an encrypted dSource requires an auto-open wallet setup in the target environment, because the provisioning process requires the master key to be stored in the wallet file. On the dSource, export the keys and copy the export file (both `ewallet.p12` and `cwallet.sso)` to the VDB server. On the VDB server, create an empty wallet and import the keys into the wallet, then set the key to be used.

When provisioning a VDB from an encrypted dSource, if the target environment has other databases that also use TDE, each database should use a different wallet. This also includes a scenario where the VDB has been provisioned back to the same environment as the encrypted dSource. Please check Oracle documentation on how to set up different wallet locations for different databases. For example, use `$ORACLE_SID` in the `DIRECTORY` clause of the `ENCRYPTION_WALLET_LOCATION` parameter in `sqlnet.ora`.

```
ENCRYPTION_WALLET_LOCATION=(SOURCE=(METHOD=FILE)(METHOD_DATA=(DIRECTORY=/opt/oracle/
wallets/$ORACLE_SID)))
```

Procedure

1. Check for any encrypted columns or tablespaces on the source database by using these commands:

   ```
   SELECT t.name name, e.encryptionalg algorithm FROM v$tablespace t,
   v$encrypted_tablespaces e
   WHERE t.ts# = e.ts# and upper(e.encryptedts) = 'YES';
   ```

2. Copy wallet files from the source database to the target environment, and then configure the `sqlnet.ora` file on the target to point to the directory where the wallet is located.

   ```
   $ more sqlnet.ora
   ENCRYPTION_WALLET_LOCATION=(SOURCE(METHOD=file) (METHOD_DATA=(DIRECTORY=/opt/
   oracle/oradata/nf/wallet)))
   ```

3. If the source database does not use the auto-open wallet, create the auto-open wallet at the target environment.

   ```
   $ orapki wallet create -wallet /opt/oracle/oradata/nf/wallet -auto_login [-pwd
   password]
   ```

4. Proceed with provisioning the VDB as described in Provisioning an Oracle VDB

Managing multiple virtual PDBs in a virtual CDB

Overview

Earlier, only one virtual PDB was supported in a virtual CDB. Starting with Delphix Engine version 8.0.0.0, you can provision multiple vPDBs in a vCDB. This enables you to manage vCDBs and vPDBs independently, when there are more than one vPDB in a vCDB.

This new feature is supported for Oracle versions 12.1.0.2 and later only.

> To support multiple vPDBs in a vCDB, Oracle 12.1.0.2 needs an Oracle patch for bug 18967466. Once the patch or update (containing the patch) is installed,
>
> make sure that the `appliedPatches` property of the Oracle repository is also updated to include the bug number. Refer to Updating repository for applied patches with the Command Line Interface for instructions.

Managing vCDBs

You can enable/disable, start/stop or delete a vCDB using **Actions (...)** menu from the **Delphix Management** application.

Prerequisites

- To disable a vCDB, all of its vPDBs must already be disabled using Delphix Management application.
- To stop a vCDB, all of its vPDBs must already be stopped or disabled using Delphix Management application.
- To delete a vCDB, all of its vPDBs must already be deleted using Delphix Management application.

Procedure

1. Login to the **Delphix Management** application.
2. Navigate to **Manage > Datasets.**
3. Select the vCDB you want to manage.
4. Click **Actions (...)** menu
5. Select the desired action for the vCDB.

Managing vPDBs

You can refresh/rewind(undo refresh), enable/disable, start/stop or delete a vPDB using **Actions (...)** menu from the **Delphix Management** application.

Prerequisites

- To disable a vPDB, its vCDB must be in a running state. You cannot disable a vPDB if its vCDB is in a stopped/disabled state. This is required in the current Delphix Engine version, because in previous versions disable succeeds even if the vCDB is in a stopped/disabled state.
- To enable/start a vPDB, if there are multiple vPDBs in its vCDB, then vCDB must have been enabled/started and open in RW mode already. If the vCDB is not enabled/started, enable/start it using **Delphix Management** application's **Action (...)** menu as mentioned in above.

Procedure

1. Login to the **Delphix Management** application.
2. Navigate to **Manage > Datasets.**

3. Select the vPDB you want to manage.
4. Click **Actions (...)**
5. Select the desired action for the vPDB.

Make sure to follow the below points, once the desired action is selected.

- Refresh/Rewind:
    - Only one vPDB in a vCDB with source-level parent-child relationship: Both the vPDB and its vCDB will be refreshed/rewound.

> The vPDB and its vCDB has a source-level parent-child relationship if the dSource CDB timeflow, from which vCDB timeflow is derived, is the same as the parent CDB timeflow of the dSource PDB timeflow from which vPDB is derived. This was always the case when there was only one vPDB in a vCDB. But with multiple vPDBs support, this may not be always the case, since later vPDB (provisioned to the same vCDB) may be provisioned from a dSource PDB of a different dSource CDB.

Consider the following scenarios:

- Provisioning a vPDB (VPDB1) from a dSource PDB (PDB1 in a source CDB1) to a new vCDB (VCDB1) will create a vPDB-vCDB virtual databases with source-level parent-child relationship. If such a vPDB is refreshed/rewound, then both vPDB and its vCDB will be refreshed/rewound.
- Provisioning a 2nd vPDB (VPDB2) from another dSource PDB (PDB2 in another source container CDB2) to the same vCDB (VCDB1) will create vPDB-vCDB databases (i.e. VPDB2-VCDB1) that does not have such a relationship. So, if such a vPDB is the only vPDB in the vCDB, then only the vPDB will be refreshed/rewound and not its vCDB.
- Only one vPDB in a vCDB with no source-level parent-child relationship: Only the vPDB, on which the action is performed, will be refreshed/rewound. Its vCDB will not be refreshed/rewound. (Refer to the example scenario 2 above.)
- Multiple vPDBs in a vCDB: Only the vPDB, on which the action is performed, will be refreshed/rewound.
- Disable/Stop:
    - If there are multiple vPDBs in a vCDB, then the action will be performed only for the vPDB.
    - If there is only one vPDB in the vCDB, then its vCDB will be also be disabled/stopped after the vPDB is disabled/stopped.

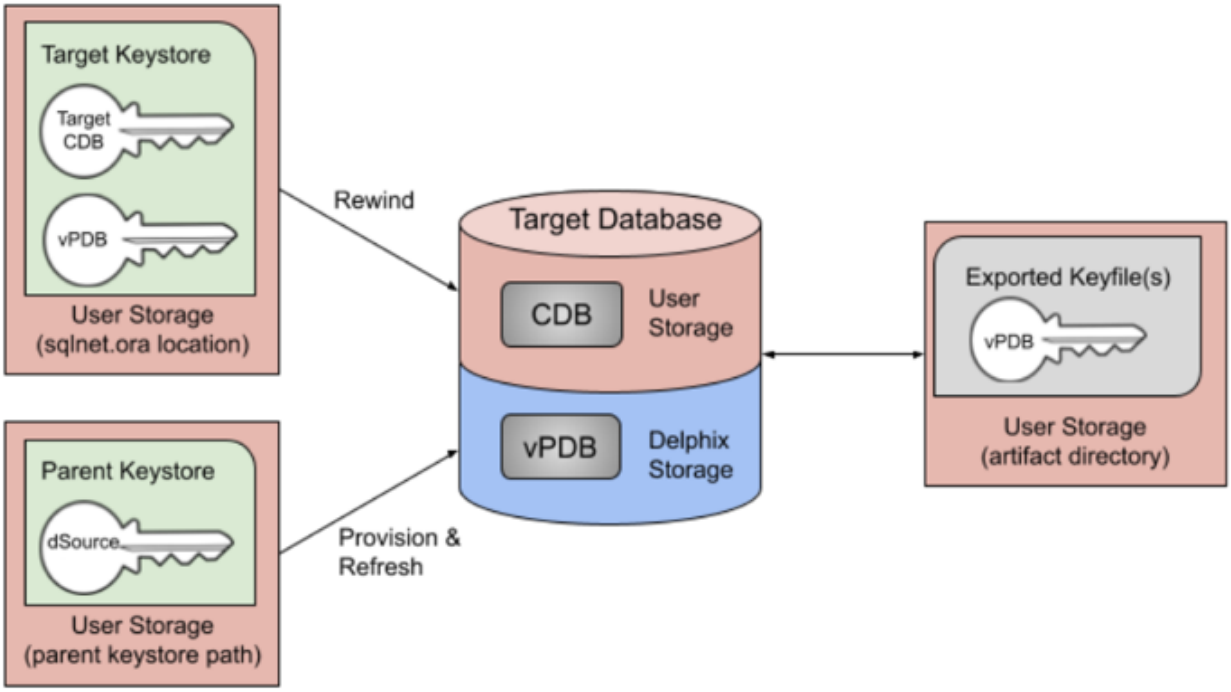
> This differs from the previous Delphix Engine(version above 8.0.0.0) behavior which always disabled/stopped the vCDB as well.

- Enable/start:
    - If there are multiple vPDBs in a vCDB, then the action will enable/start the vPDB only.
    - If there is only one vPDB in the vCDB, then the vCDB will be also be enabled/started if required before enabling/starting the vPDB itself.
- Delete
    - If this is the last vPDB in a vCDB, then the vCDB is also deleted.
    - If you are deleting a vPDB using the force option, make sure to follow the steps below to clean up the vPDB from the target host:
      1. Drop the virtual PDB from the virtual CDB manually after it is deleted from the Delphix Management UI.
      2. Remove the virtual PDB mount point from the target host (residing in the mount base directory) to avoid stale mounts.

If a physical PDB is added to a virtual CDB or a virtual PDB from a virtual CDB is deleted using the force option, LogSync will fail with the error "`Virtual container database <vCDB Name> has pluggable databases that are not managed by Delphix: <PDB names>.`" and/or SnapSync will fail with the error code "`exception.oracle.vdb.foreign.pdbs.found.in.vcdb`". To resolve the errors, unplug the physical PDBs from the virtual CDB manually or drop the virtual PDBs (that were deleted from Delphix using the force option) from the virtual CDB manually.

Refreshing and rewinding a TDE-enabled vPDB

Just like a non-TDE-enabled vPDB, a TDE-enabled vPDB can be refreshed from the dSource or rewound to a previous snapshot or point in time. In each case, no additional manual steps or input from the user is required. The first step of a refresh or rewind operation is to disable the existing vPDB, which will result in a new keyfile exported to the artifact directory. The appropriate snapshot files are then mounted for the auxiliary database so that it can be recovered and brought to a consistent state. Since the vPDB is TDE-enabled, a keystore is needed for the recover operation. For a refresh, the Delphix Engine will use the parent keystore, and for a rewind, the Delphix Engine will use the target keystore, as shown below.



Overview of Key Rotation

Some customers have strict security compliance standards that mandate that production master keys cannot be shared into non-production zones. Delphix supports the ability to perform automated keystore sanitization of a vPDB. In simpler terms, Delphix allows provisioning a vPDB that has no previous production keys associated with it. A freshly provisioned vPDB will thus contain one and only one newly-set master encryption key that can be imported into the target CDB keystore to resolve TDE-plugin violations at the end of a provision job. Note that the tablespace encryption keys, which are themselves encrypted by the PDB key, are not rotated. In such a scenario, this new encryption key is expected to be the only key imported by the target container database (CDB) at the end of the provision job. It is important to note that Delphix does not re-encrypt the actual data files when the production master key is rotated.

There are two potential places for keys to be rotated in a vPDB environment:

1. dSource: If the dSource keys are rotated and a new snapshot taken with the new key, the customer is responsible for updating the parent keystore before refreshing from the later snapshot encrypted with the new key. The parent keystore would then contain both the new key and the original keys.
2. Target: If the target CDB keys are rotated, the target keystore will be updated. This is why the Delphix Engine uses the target keystore for rewind operations.

In either scenario, the keystore used for recovery will contain the current and all prior keys used to encrypt the datafiles and archive logs, for both the vPDB and CDB used in the auxiliary container.

vPDB encryption key management

During the provisioning process of a TDE-enabled vPDB, Delphix generates a unique encryption key for the vPDB. This unique key is not associated with the parent keystore to ensure that no keys from the parent are imported by the target. During refresh and rewind operations, Delphix reuses that key after recovery has finished. It is possible to customize the key that is used by updating the `tdeKeyIdentifier` parameter of the source via the CLI. If a valid `key_id` is entered for a key that is already present in the keystore, that key will be used as the active encryption key for the vPDB at the end of refresh/rewind. If the field is unset, Delphix will generate a new encryption key for the vPDB to be used from that point onward. This procedure is the same when using a vCDB, in which case Delphix will also generate a new unique encryption key for the vCDB that is reused for refresh and rewind, and which can be customized by updating the `tdeKeyIdentifier` parameter of the CDB source. See the CLI steps for Locating and Updating the Value of tdeEncryptionKey.

Timeflows for RAC provisioning of VDBs

Overview

This topic describes special considerations when provisioning by timestamp from a RAC time flow.

Timestamps in Oracle RAC time flows can be imprecise because of time skew among the hosts in a RAC configuration. The time stamps will generally track the host with the fastest clock. For this reason, provisioning by a timestamp may not leave the VDB provisioned at the exact time desired. The provision by SCN should be used if more fine-grained control is required when provisioning.

Introduction

The Delphix Engine provides the ability to link to an external database by creating a dSource within the Delphix system. Once linked, the Delphix Engine maintains a complete history of the database as part of a Timeflow, limited by the retention policies configured by the administrator. From any time within that Timeflow, you can provision a virtual database (VDB) from the Delphix Engine. This Timeflow is maintained through the use of SnapSync and LogSync.

The SnapSync operation pulls over the complete data set of the external database during the initial load. Subsequent SnapSync operations pull and store only incremental changes. At the end of each SnapSync operation, the Delphix Engine creates a snapshot that serves as the base point for provisioning operations. In addition, LogSync periodically connects to the host(s) running the source database and pulls over any log files associated with the database. These log files are stored separately from the SnapSync data and are used to provision from points in between SnapSync snapshots. Usually, SnapSync operates against a live database with changes actively being made to it. Hence the data that it pulls over is “fuzzy” and logs must be applied to the data to make it consistent and provisionable. If LogSync is enabled, SnapSync relies on it to copy the logs over. If LogSync is not enabled, SnapSync copies the logs itself. Occasionally, LogSync or SnapSync is not able to retrieve one or more log files from the database. This creates a break in the Timeflow or can prevent a snapshot from being provisioned. To remedy this situation, the Delphix Engine has tools to repair, or patch, a snapshot and the Timeflow.

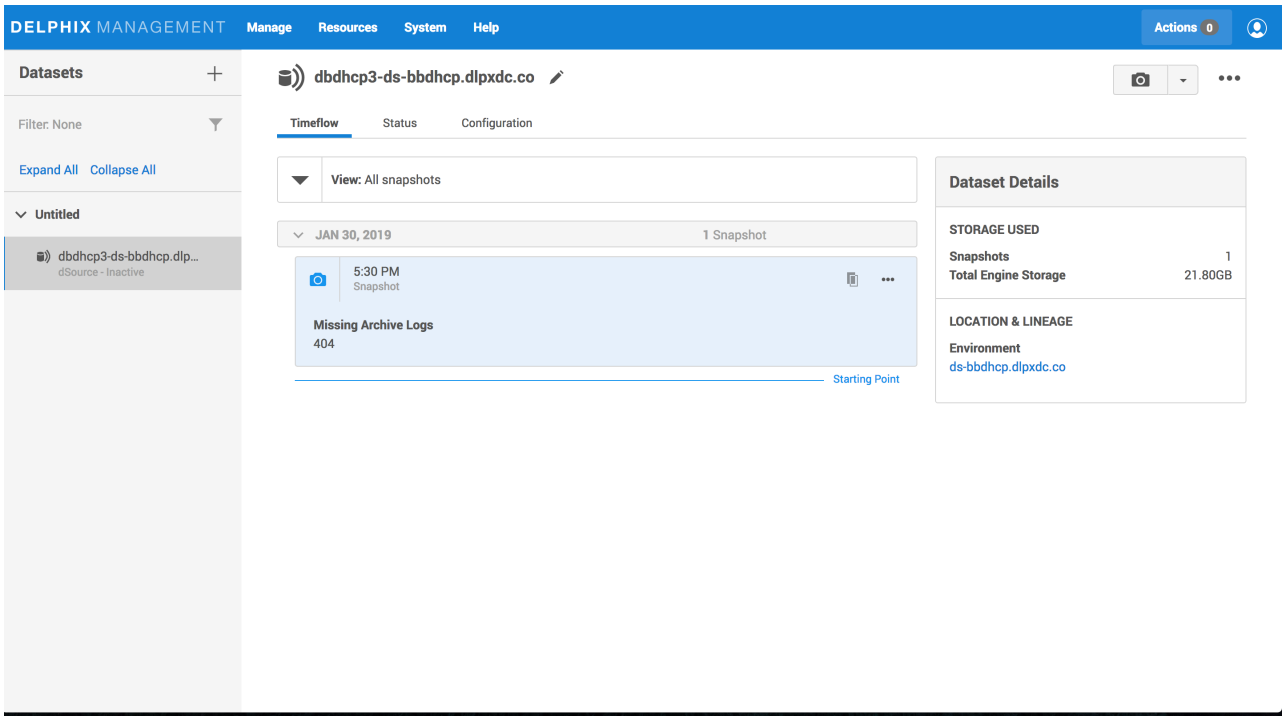
Snapshot repair

> **ASM**
>
> The steps below do not apply if your archive logs are stored on ASM. If they are stored on ASM, you must move the archived logs to a supported filesystem directory.

When missing log files prevent the Delphix Engine from provisioning a snapshot, you can use the Delphix Management application to identify the missing logs and repair the snapshot. The Delphix Engine will generate a fault whenever missing logs prevent a snapshot from being provisionable. The fault will likely have the title "Cannot provision database from snapshot" and will contain a description of the cause. The common causes are:

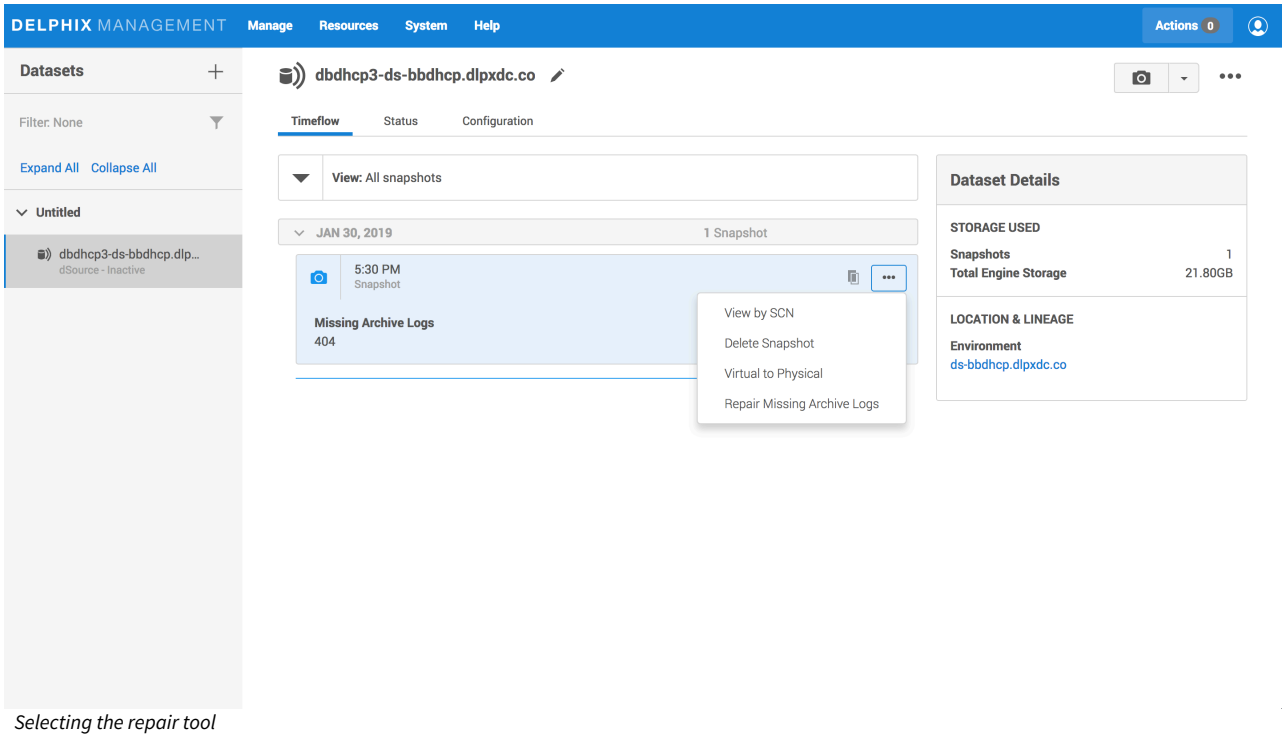
- Logs were deleted/moved/archived from the database before the Delphix Engine could retrieve them. In this case, the archive log retention policy on the source database may be too aggressive. Use the GUI snapshot repair tool to fetch the logs.
- LogSync is still fetching the logs. SnapSync is relying on LogSync to fetch the logs needed to make the snapshot consistent. SnapSync normally will wait up to 15 minutes for LogSync to fetch the logs. If LogSync has not fetched the logs by then, SnapSync will generate a fault and finish. The best course of action, in this case, maybe to wait for LogSync to fetch the logs.
- The source database is a physical standby in real-time apply mode. The changes described in the current online log of the database are needed to make the snapshot consistent. LogSync cannot retrieve the log until it is archived, and SnapSync cannot force the log to be archived because the source database is a physical standby. Force a log switch on the primary database or wait until the log is naturally archived.

Below is a screenshot of a snapshot with missing logs. Clicking on the snapshot causes the list of missing log(s) to appear. In this example, log sequence 404 is missing.

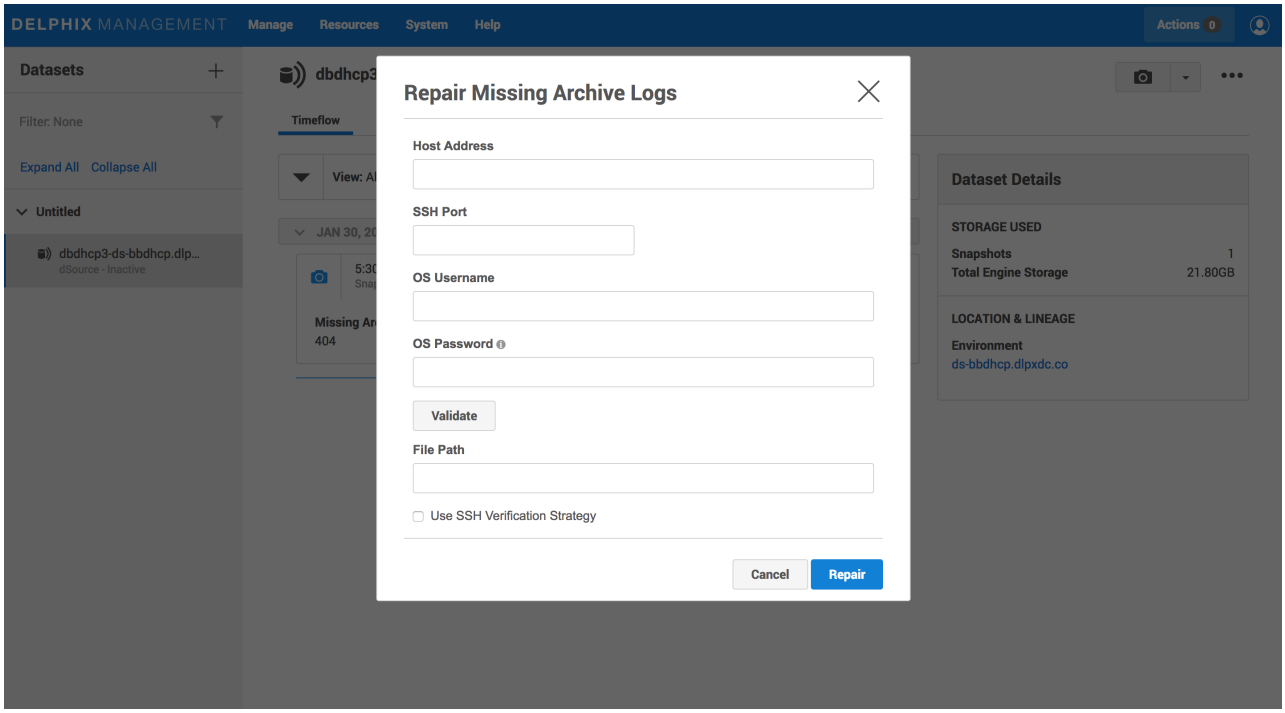


Snapshot with missing logs

If the snapshot can be repaired by fetching the logs from the source database, then you can use the GUI snapshot repair tool to fetch the logs. Hovering over the snapshot exposes more options ("**...**") which can then be clicked to show the **Repair Missing Archive Logs** option. Clicking on the option starts the repair tool.



Selecting the repair tool



Repair tool

To use the snapshot repair tool, as seen above:

1. Enter a **Hostname**. This should be the host from which to retrieve the log(s).
2. Enter a **Username** and **Password**. These should be the credentials for a user who can read the archived log file(s). The user credentials are optional if the host and user credentials have already been added to the Delphix Engine.
3. Enter a **File Path**. This should be the name of the directory containing the missing log(s).

If more than one file is missing, they should all exist in the directory specified by **File Path**. The tool will read every file in the **File Path** directory, so it is best that it only contains the files that are to be retrieved.

Timeflow patching

When missing log files cause a break in the Timeflow, you can use the command-line interface (CLI) to identify the missing logs and patch the Timeflow. The Delphix Engine will generate a fault whenever there are missing logs on a portion of the Timeflow. The fault will likely have the title “Cannot provision a database from a portion of Timeflow” and will contain a description of the cause. The most common cause is an overly aggressive archive log retention policy on the source database causing a log to be deleted before LogSync can fetch it. Other faults can also be generated by describing the specific errors encountered when fetching the log(s).

You can use the CLI to list the missing logs and patch the Timeflow. The following CLI Cookbook entry demonstrates how to do this: CLI Cookbook: Repairing a Timeflow

If you delete or move archivelogs from the source database that are not needed for a snapshot, you still may need to repair the TimeFlow to provision using LogSync. In this case, an icon will not be visible on the TimeFlow tabs. This means you cannot repair the TimeFlow in the GUI. However, you can still repair it using the CLI.

Upgrading an Oracle VDB, linked CDB, or a vCDB

Prerequisites

Upgrading a VDB involves Oracle level activity and Delphix GUI/CLI level activity. The Oracle activity, to be performed first, involves upgrading/patching current oracle homes, installing new oracle homes, and managing init.ora/spfile.ora files. The Delphix GUI/CLI activity, to be performed second, involves rediscovering new oracle installations, giving a VDB a new oracle home, and updating the oracle grid home. These are discussed in detail below.

During an Oracle upgrade, refreshing an Environment will generally discover new Oracle installations, allowing for a virtual database (VDB) or dSource upgrade to be handled through the UI. However, if the Oracle Grid home was changed, due to an Oracle upgrade, the crsClusterHome parameter will need to be updated manually through the command line (CLI).

Limitations

Currently, it is not possible to convert an existing VDB into a vPDB.

PSU/Oracle upgrade procedure

Normally a PSU or Oracle upgrade will have both binary changes and some scripts to run on the database side as well.

There are 3 ways to apply a PSU/Oracle upgrade:

1. Apply to existing ORACLE_HOME. You must be on Delphix version 4.1.x or higher to do this.
2. Create a new ORACLE_HOME (could clone the existing one) and then apply the PSU to the new ORACLE_HOME
3. After a dSource is upgraded, use refresh on the Timeflow tab to upgrade the VDB

Follow Oracle documentation and run the appropriate script(s) and/or steps on the databases using those ORACLE_HOMEs. In option B, stop the instance using the old ORACLE_HOME, then restart the instance with the new ORACLE_HOME from the command line as normal.

Applying to an existing ORACLE_HOME

1. Following Oracle documentation, patch the ORACLE_HOME, then the database for the VDB(s).
2. Refresh the environment the VDBs are on in the Delphix Management application.

Creating a new ORACLE_HOME

1. Refresh the environment from the Delphix Management application. Verify that the new ORACLE_HOME is picked up and displayed in the **Databases** tab as an ORACLE Installation.
2. Stop the VDB instance (old ORACLE_HOME) using Oracle tools. Do not use the Delphix VDB **stop** operation as the VDB should be stopped outside of Delphix.
3. Export ORACLE_HOME=(newORACLE_HOME). Follow Oracle documentation to patch the database.
4. Copy the init.ora for that VDB in this new $ORACLE_HOME/dbs directory. The delphix_os user will need the write permissions to this directory.
5. If there are any database parameter changes, update the spfile located on the Delphix mount base with those values.
6. Navigate to the **Datasets** view
7. Expand the group(s) containing all non-multitenant and multitenant VDBs.
8. Click the **Configuration** tab.

9. From the Actions menu (...) select **Upgrade** to switch the ORACLE_INSTALLATION to the new one.
10. If the database is a linked CDB or vCDB, go to any child vPDBs and verify that the **Repository** and/or **Version** has been updated under the **configuration** tab.

Using refresh

1. Refresh the environment from the Delphix Management application.
2. Verify that the new ORACLE_HOME is picked up and displayed in the **Databases** tab of the **Environments** screen as an ORACLE Installation.
3. On the **VDB Configuration tab,** click the **stop** icon to stop the VDB.
4. From the Actions menu (...) select **upgrade** to switch the ORACLE_INSTALLATION to the new upgrade version same as the dSource.
5. Navigate to the **Datasets** view, select the VDB, and then select the **Timeflow** tab.
6. Click the **Refresh** button.
7. Select a new **snapshot** from the dSource that was taken after the dSource was upgraded. (The database version is on the snapshot card.)

Updating the Oracle user after an upgrade
There may be cases when you upgrade the Oracle home and the Oracle User (who owns the binary) is a different user than the previous Oracle User. You will then need to update the Oracle User for each environment, and then re-connect each VDB to the upgraded Oracle home using the new Oracle User. The new Oracle User must be in the same OS group (for example, dba or oinstall) as the previous one.

1. Login to the Delphix Management application using delphix_admin credentials.
2. Click **Manage**.
3. Select **Environments**.
4. Select the **environment** where you want to add the user.
5. Next to **Environment Users**, click the **Pencil** icon to add the new user.
6. Set the new user as the **default** user.
7. Follow the procedure to upgrade VDBs described in this topic.

Linked CDB upgrade procedure

For vPDBs in the linked CDBs, there are two ways to upgrade the linked CDB:

A. Perform the Oracle upgrade of the current target CDB.

B. Create an entirely new target CDB of the higher version.

Performing the Oracle upgrade of the current target CDB

1. Perform the Oracle upgrade of the current target CDB.
2. Login to the Delphix Management application.
3. Navigate to **Manage** > **Datasets** and select the target CDB.
4. From the **Actions** menu (...), select **Upgrade** to switch the ORACLE_INSTALLATION to the new one.
5. Click Upgrade.
6. Under the **Configuration** tab, verify that the **Repository** and/or **Version** has been updated.

Creating a new target CDB

You can first disable the vPDB, then use the migrate vPDB feature to select a new container database.

Procedure

1. Login to the Delphix Management application.
2. Navigate to **Manage** > **Datasets**.

3. Select the vPDB you want to migrate.
4. From the **Actions** menu (...) select **Disable**.
5. Click **Disable** to confirm.
6. From the **Actions** menu (...) select **Migrate**.
7. Select the new container database for the vPDB, the user for that environment, and the database installation where the container database of the vPDB will reside.
8. Click the **Migrate** option to confirm your selections.
9. Manually copy the exported keys to the target host's toolkit directory.
10. From the **Actions** menu (...) select **Enable**.
11. Click **Enable** to confirm. Your vPDB will re-start in the new environment, and you can continue to work with it as you would any other vPDB.

Customizing RAC instances after provision

Provisioning a RAC VDB requires users to select which RAC node to be included for this VDB, as well as the instance number and instance name of each RAC node that runs the RAC VDB. At any time after the provision, there may be a need to downsize or increase the number of instances of the RAC VDB or reassign instance number and instance names to individual VDB RAC node.

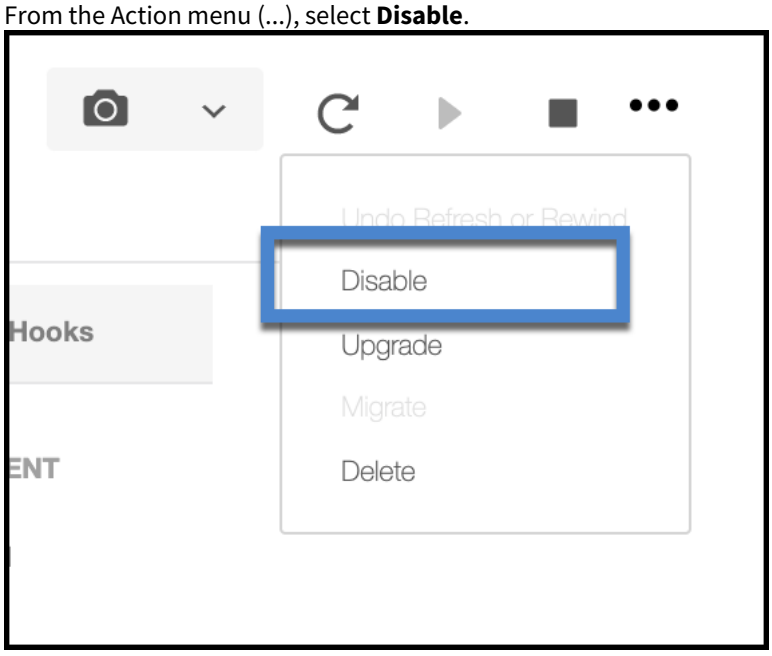
Oracle pluggable database
The current release does not support the customization of RAC vPDB instances after provision.

If you make any changes to the VDB's configuration on the engine either by changing the VDB's configParams using CLI or VDB configuration template that was used to provision it, the Delphix engine will push out these changes when the VDB is next enabled after the RAC instance changes are made.

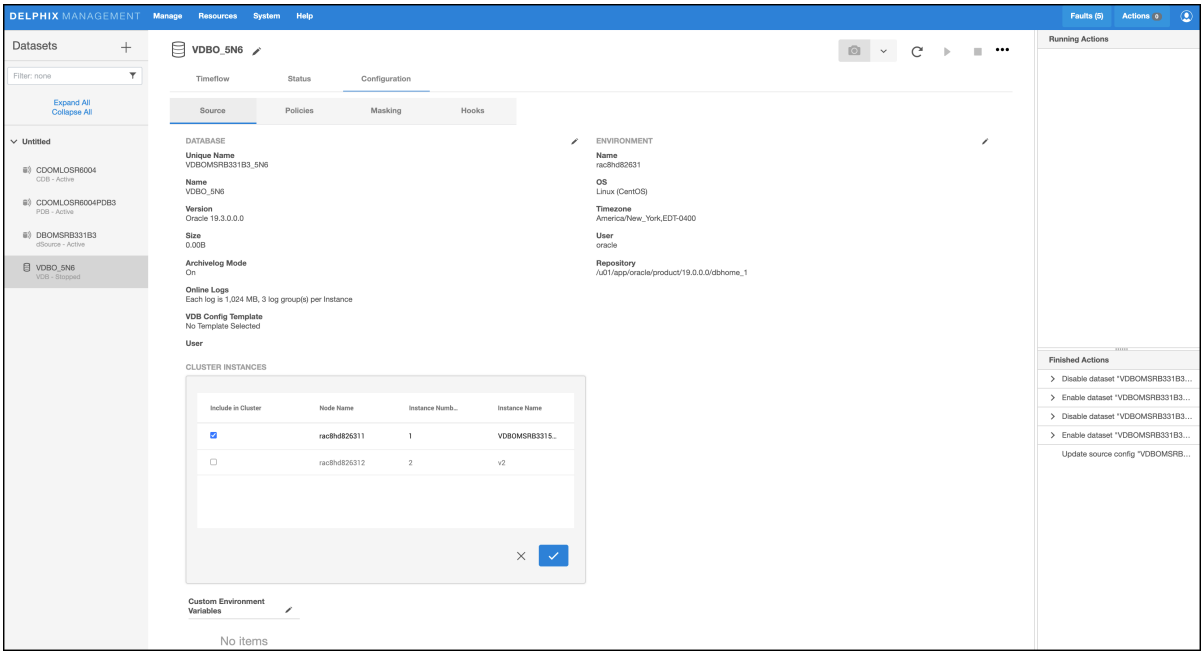
Procedure

The following steps illustrate how you can customize your RAC VDB instance configuration after provision:

1. Login to your Delphix Management application.
2. Select the VDB you want to edit.
3. From the right-side pane select the **Configuration** tab.
4. From the Action menu (...), select **Disable**.



5. Click on the **pencil** icon next to 'CLUSTER INSTANCES' to be able to edit your instance names.
6. Click on the name of the instance you want to alter and edit it to suit.



7. Check or uncheck the check-box to the left to indicate if you want to add or remove a cluster node for your RAC VDB.
8. Complete editing and then confirm using the tick icon.
9. From the Action menu (...), select **Enable**.

> Disabling a VDB will result in the VDB Instances being shut down and the NFS mounts for that VDB presented from Delphix being dismounted.

Migrating an Oracle VDB

This topic describes how to migrate a Virtual Database (VDB) from one target environment to another.

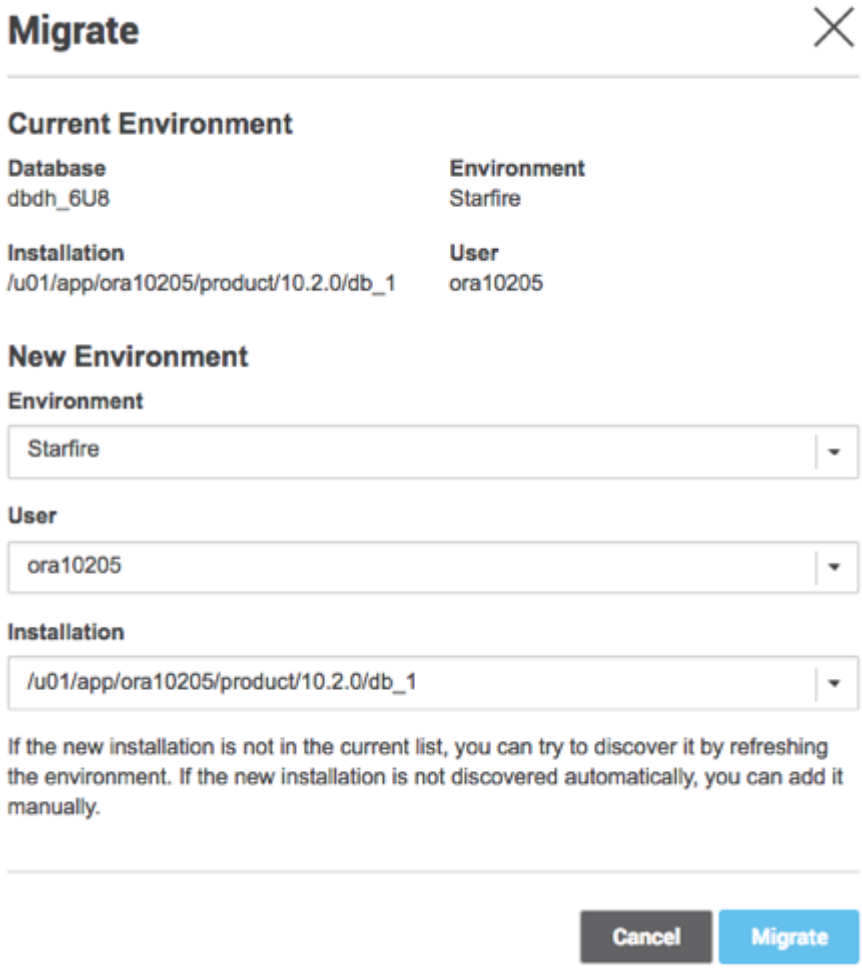
There may be situations in which you want to migrate a virtual database to a new target environment, for example when upgrading the host on which the VDB resides, or as part of a general data center migration. This is easily accomplished by first disabling the database, then using the Migrate VDB feature to select a new target environment.

Prerequisites

- You should have already set up a new target environment that is compatible with the VDB that you want to migrate.
- You cannot migrate a single instance of Oracle VDB to a RAC environment and vice versa. An additional reconfiguration is needed when converting a single instance to RAC that is only performed during a VDB provision. Instead, you should provision a new VDB.

Procedure

1. Login to your Delphix Management application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **VDB** you want to migrate.
5. From the Actions menu (...) select **Disable.**
6. Click **Disable** to confirm. When the VDB is disabled, its icon will turn grey.
7. From the Actions menu (...) select **Migrate**.
8. Select the new target environment for the VDB, the user for that environment, and the database installation where the VDB will reside.



9. Select **Migrate** to confirm your selections.
10. From the Actions menu (...) select the **Enable**
11. Click **Enable** to confirm.

Within a few minutes, your VDB will re-start in the new environment, and you can continue to work with it as you would any other VDB.

Migrating a vPDB

There may be situations in which you want to migrate a virtual pluggable database (vPDB) to a new container database on the same or a different target environment, for example when upgrading the host on which the vPDB resides, or as part of a general data center migration. This is easily accomplished by first disabling the vPDB, then using the Migrate vPDB feature to select a new container database.

> The current release only supports migrating from a vPDB in a virtual CDB, to another linked or virtual CDB using the CLI.

Pre-requisites

- You should already set up and have Delphix discover a container database in the same environment as the vPDB currently is or from an environment to which the vPDB will be migrated to.
- The virtual CDB from/to which vPDB is to be migrated must be already running and open in a read-write mode.

Procedure

Login to your Delphix Management application.

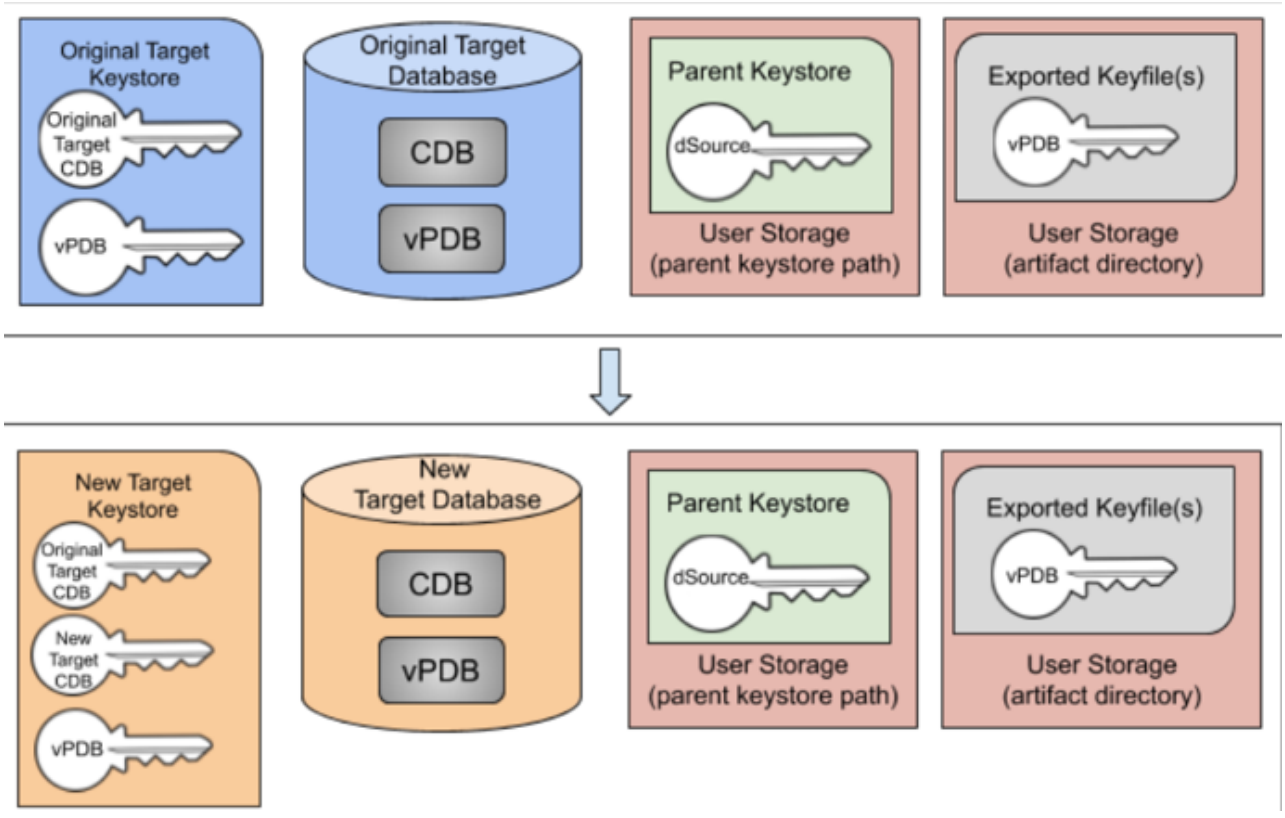
1. Click **Manage**.
2. Select **Datasets**.
3. Select the **vPDB** you want to migrate.
4. From the Actions menu (...) select **Disable**.
5. Click **Disable** to confirm.
6. From the Actions menu (...) select **Migrate**.
7. Select the new **container database** for the vPDB, the **user** for that environment, and the **database installation** where the container database of the vPDB will reside.
8. Click the **Migrate** to confirm your selections.
9. Manually copy the exported keys to the target hosts toolkit directory. When migrating a TDE enabled vPDB, the disable step results in the vPDB keys being exported to the toolkit. Enabling the vPDB in a different target host will result in plugin violations, therefore when migrating to a new host, the user must manually copy the exported keys to the toolkit directory on the target host.
10. From the Actions menu (...) select **Enable**.
11. Click **Enable** to confirm. Your vPDB will re-start in the new environment, and you can continue to work with it as you would any other vPDB.

Migrating a TDE-enabled vPDB

Migrating a vPDB from one container database to another container database on a different host involves the following steps:

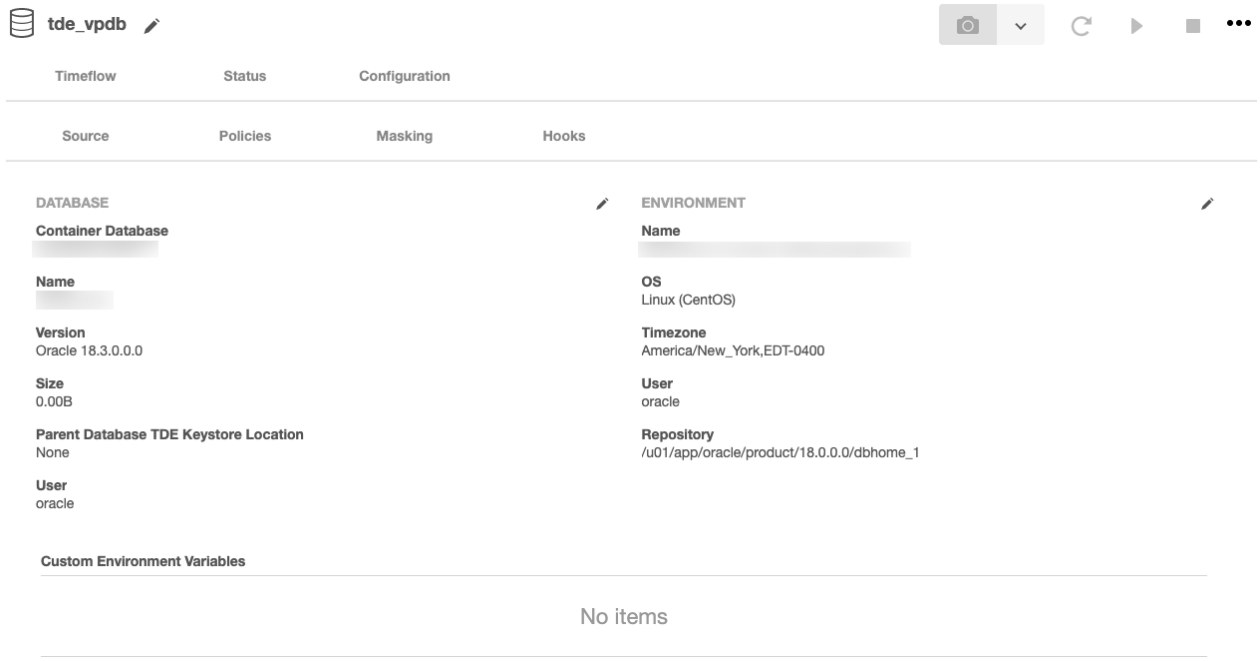
1. Disable the vPDB.
2. Migrate.
3. For a TDE-enabled vPDB, there are additional steps that are required before the enable. The steps are necessary because the keystore and exported keyfiles are not present on Delphix storage. These steps are:
    a. Copy the parent TDE keystore from the original target host to the new host. If the vPDB is not being migrated to a new host, then this step is not needed. Similarly, if the new host has the parent keystore in a different location, then the parent keystore path for the vPDB needs to be updated.
    b. Copy the artifact directory for the vPDB from the original target host to the new host. If the vPDB is not being migrated to a new host, then this step is not needed.
    c. Ensure that the new target container has the TDE keystore password set.
    d. Merge the original target keystore into the new target keystore. This is required to support a rewind operation to a snapshot taken before the migrate.
4. Enable.

The diagram below illustrates the scenario of a migrated TDE-enabled vPDB.



Example

Consider the following TDE-enabled vPDB:



This vPDB is currently provisioned to the container database `CDOMLOTG9620` on the host `tde-target18`. Connecting to that database, we can see the keys for the target container and the vPDB:

```
SQL> show pdbs
CON_ID CON_NAME            OPEN MODE  RESTRICTED
------ --------------------- ---------- ----------
     2 PDB$SEED            READ ONLY  NO
     3 CDOMLOTG9620PDB1        READ WRITE NO
     4 CDOMLOTG9620PDB2        READ WRITE NO
     5 CDOMLOTG9620PDB3        READ WRITE NO
     6 TDE_VPDB            READ WRITE YES
SQL> select con_id, key_id from v$encryption_keys;
CON_ID KEY_ID
------ ----------------------------------------------------------
     1 ASFwmcfaMk8vv1LvzV0H8BEAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
     6 AdDdKibLKU9mv6PDAIvVvH0AAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

The key for the target container starts with **ASFwmcfa** and the vPDB key starts with **AdDdKibL**. We also have an artifact directory for this vPDB found in `<toolkit_directory>/oracle_tde_keystores`:

```
$ ls /toolkit/oracle_tde_keystores/
tde_vpdb_Encrypted_e9d2befb-9849-43c8-85f5-5ee8b760e334
```

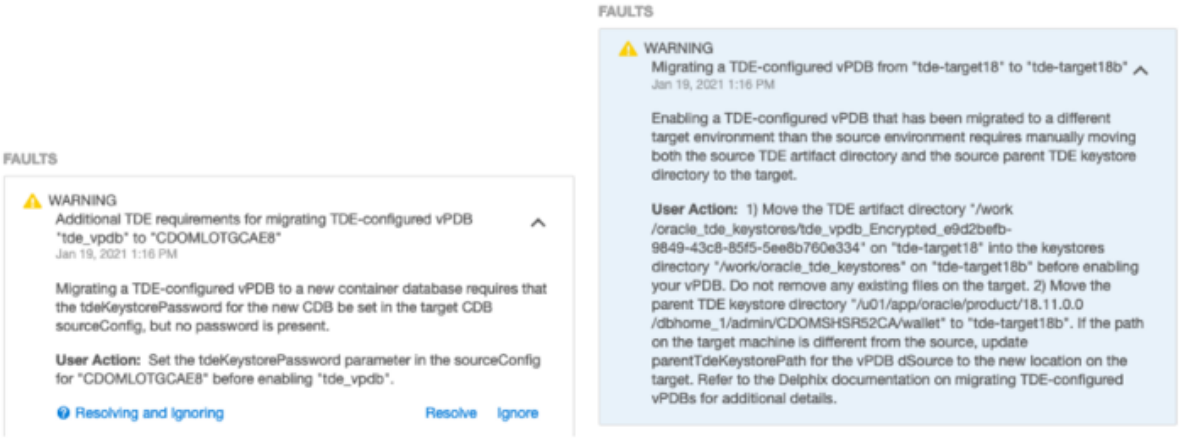
We are going to migrate this vPDB to the container database `CDOMLOTGCAE8` located on the host `tde-target18b`. Currently, that container has the following configuration:

```
SQL> show pdbs
```

```
CON_ID CON_NAME            OPEN MODE  RESTRICTED
------ -------------------- ---------- ----------
     2 PDB$SEED             READ ONLY  NO
     3 CDOMLOTGCAE8PDB1         READ WRITE NO
     4 CDOMLOTGCAE8PDB2         READ WRITE NO
     5 CDOMLOTGCAE8PDB3         READ WRITE NO
SQL> select con_id, key_id from v$encryption_keys order by con_id;
CON_ID KEY_ID
------ ----------------------------------------------------
     1 Ad344VSUDk/jv1VDb1QNBHMAAAAAAAAAAAAAAAAAAAAAAAAAAA
     3 AetaL+IKpE+ov5avXouApwUAAAAAAAAAAAAAAAAAAAAAAAAAAAA
     4 AXZTgaBbxk+6v6x2yHQ4ArgAAAAAAAAAAAAAAAAAAAAAAAAAAAA
     5 AayTgbjkJE/3v82/hBBBAWEAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

The new target container has a CDB key starting with **Ad344VSU**. The migration steps are as follows:

1. Disable the vPDB in the original container on the host `tde-target18a`.
2. Migrate the vPDB to point to the new container on the new target host `tde-target18b`. The UI will report the following warnings during the migrate operation, which indicate the manual steps which need to be done:



3. Set the TDE keystore password for the new target container `CDOMLOTGCAE8` on the new target host `tde-target18b`.
4. Copy the parent keystore to the new target host `tde-target18b`. In this case, we will keep the existing location `/u01/app/oracle/product/18.11.0.0/dbhome_1/admin/CDOMSHSR52CA/wallet` on both hosts, so the configuration does not need to be updated.
5. Copy the TDE artifact directory `/work/oracle_tde_keystores/tde_vpdb_Encrypted_e9d2befb-9849-43c8-85f5-5ee8b760e334` to the new target host `tde-target18b`. If there are no TDE-enabled vPDBs on the new target host, the `oracle_tde_keystores` directory may need to be created first.
6. Merge the original target keystore into the new target keystore. Oracle provides the `ADMINISTER KEY MANAGEMENT MERGE KEYSTORE` command to facilitate this. Note that you cannot successfully merge into a keystore that is currently in use by the database, so the recommended process is to first copy the original target keystore to the new host, merge the two keystores into a new keystore, replace the existing new target keystore and finally bounce the CDB to start using the merged keystore:

a. Copy the original keystore to `/tmp/tde-target18a` on `tde-target18b`.
b. Create a temporary directory `/tmp/merged` on `tde-target18b.`
c. While connected to the new container on `tde-target18b`, issue the merge keystore command:

```
SQL> administer key management merge keystore '/tmp/tde-target18a'
 identified by
  *** and keystore '/u01/app/oracle/product/18.11.0.0/dbhome_1/admin/
CDOMLOTGCAE8/wallet'
  identified by *** into new keystore '/tmp/merged' identified by ***;
```

d. Copy the new merged keystore `/tmp/merged/ewallet.p12` into the existing keystore location on `tde-target18b` (backing up the existing keystore first).
e. If there is an autologin wallet configured for the container, it must be recreated:

```
SQL> administer key management create auto_login keystore from keystore
 '/tmp/merged' identified by ***;
```

Copy the new merged autologin keystore `/tmp/merged/cwallet.sso` into the existing keystore location on `tde-target18b` (backing up the existing autologin keystore first).
f. Shutdown and restart the new target container database on the host `tde-target18b`.

7. Confirm that the keystore now contains both the original key and the new key:

```
SQL> select key_id from v$encryption_keys where con_id = 1;
KEY_ID
------------------------------------------------
ASFwmcfaMk8vv1LvzV0H8BEAAAAAAAAAAAAAAAAAAAAAAAAAAAA
Ad344VSUDk/jv1VDb1QNBHMAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

Note that we have the key starting with **ASFwmcfa** from the original container, and the key starting with **Ad344VSU** from the new container.

8. Enable the vPDB to complete the migration operation. We can now see that the vPDB is successfully started in the new container, with its key starting with **AdDdKibL**:

```
SQL> show pdbs
CON_ID CON_NAME            OPEN MODE  RESTRICTED
------ ------------------------------ ---------- ----------
     2 PDB$SEED            READ ONLY  NO
     3 CDOMLOTGCAE8PDB1        READ WRITE NO
     4 CDOMLOTGCAE8PDB2        READ WRITE NO
     5 CDOMLOTGCAE8PDB3        READ WRITE NO
     6 TDE_VPDB            READ WRITE NO
SQL> select con_id, key_id from v$encryption_keys order by con_id;
CON_ID KEY_ID
------ ------------------------------------------
     1 Ad344VSUDk/jv1VDb1QNBHMAAAAAAAAAAAAAAAAAAAAAAAAAAAA
     1 ASFwmcfaMk8vv1LvzV0H8BEAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

```
     3 AetaL+IKpE+ov5avXouApwUAAAAAAAAAAAAAAAAAAAAAAAAAAA
     4 AXZTgaBbxk+6v6x2yHQ4ArgAAAAAAAAAAAAAAAAAAAAAAAAAAA
     5 AayTgbjkJE//v82/hBBBAWEAAAAAAAAAAAAAAAAAAAAAAAAAAA
     6 AdDdKibLKU9mv6PDAIvVvH0AAAAAAAAAAAAAAAAAAAAAAAAAAA
6 rows selected.
```

Migrating a vPDB (from a linked CDB) to a higher Oracle version (linked CDB)

Given a vPDB in a linked CDB (CDBold for hereon) the following steps can be used to migrate it to a higher Oracle version linked CDB (CDBnew from hereon):

- It might be necessary to run pre-upgrade Oracle scripts before doing the upgrade, consult Oracle documentation.
- Disable the vPDB, using the GUI or the CLI.
- Using the CLI perform manual migration. If the CDBnew is in the same host as the CDBold, then only cdbConfig needs to be specified, otherwise the repository and the environmentUser also need to be specified:

```
> cd /sourceconfig
sourceconfig> select vpdb
sourceconfig 'vpdb'> update
sourceconfig 'vpdb' update *> set cdbConfig=CDBnew
sourceconfig 'vpdb' update *> set environmentUser=<new oracle user>'
sourceconfig 'vpdb' update *> set repository=''
sourceconfig 'vpdb' update *> commit
```

After enabling the vPDB without starting, the environment of the linked CDB should not be refreshed until the vPDB has been created and upgraded.

- Enable the vPDB without starting it:

```
> cd /source
source> select vpdb
source 'vpdb'> enable
source 'vpdb' enable *> set attemptStart=false
source 'vpdb' enable *> commit
```

- Add the vPDB to CDBnew using sqlplus, which has two options:
  - The vPDB can be plugged as a clone (to get a new DBID/GUID):

```
SQL> create pluggable database vpdb as clone using'/mnt/provision/vpdb-
CDOMLOSRA92E/datafile/delphix_group_writable/VPDB.xml' nocopy tempfile
reuse;
```

  - The vPDB can be plugged without the clone option (to retain the same DBID/GUID):

```
SQL> create pluggable database vpdb using'/mnt/provision/vpdb-
CDOMLOSRA92E/datafile/delphix_group_writable/VPDB.xml' nocopy tempfile
reuse;
```

- Open the vPDB in upgrade mode:

```
SQL> alter pluggable database vpdb open upgrade;
```

- Upgrade the vPDB following the procedures documented by Oracle.
- Take a new snapshot with the new version upgraded and delete the previous snapshot containing the previous version metadata.

All Delphix operations on the upgraded vPDB are possible, which include:

- Start
- Stop
- Enable
- Disable
- Taking a snapshot
- Rewind (to an upgraded snapshot)
- Refresh (requires additional steps)
    - Disable the vPDB
    - Using the CLI to migrate the vPDB back to CDBold:

```
> cd /sourceconfig
sourceconfig> select vpdb
sourceconfig 'vpdb'> update
sourceconfig 'vpdb' update *> set cdbConfig=CDBold
sourceconfig 'vpdb' update *> set environmentUser=<old oracleuser>’
sourceconfig 'vpdb' update *> set repository=''
sourceconfig 'vpdb' update *> commit
```

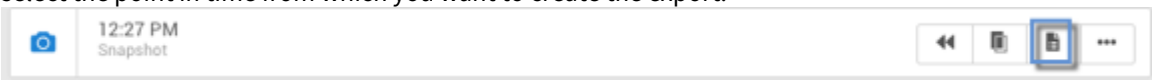
    - Refresh the vPDB
    - Repeat the procedure for migration as described before

V2P with an Oracle VDB

This topic describes the procedure for exporting a virtual database (VDB) to a physical one, also known as V2P.

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage** and select **Datasets**.
3. Select the dSource or VDB you want to export.
4. Select the snapshot of the dSource or VDB state you want to export.
5. If you want to export the state of the database from a specific point in time, select the LogSync icon and then select the point in time from which you want to create the export.



6. From the actions menu (...) select **Virtual to Physical**.
7. Select the **target environment**. The target environment should have been added to Delphix previously and should meet all target host requirements, see Requirements for Oracle hosts and databases
8. Enter the Target Directory for the export.
   The target directory you enter must exist in the target environment, and the Delphix operating system user listed under the environment must have permission to write to it. The target directory should be empty.
9. Select whether or not to **Open database after recovery**.
   If you do not select this option, the Oracle database will not undergo open resetlogs, and the database will not be available for read/write access. This can be useful if the files are to be used to restore an existing data file for recovery purposes. You can use the scripts that are created in the target environment to complete the database open process at a later time. For more information, see Manually recovering a database after V2P
10. Click **Advanced** to customize **data transfer settings**, customize the **target directory layout**, enter any **database configuration** parameters or enter **file mappings** from the source environment to the target. For more information, see Customizing target directory structure for database export, Configuration settings for Oracle virtual databases and Customizing VDB file mappings. The data transfer settings are described below:
    a. **Compression** – Enable compression of data sent over the network. Default is **Off**.
    b. **Encryption** – Enable encryption of data sent over the network. Default is **Off**.
    c. **Bandwidth Limit** – Select the network bandwidth limit in units of megabytes per second (MB/s) between the Delphix Engine and the target environment. The default is **0**, which means no bandwidth limit is enforced.
    d. **Number of Connections** – Select the number of transmission control protocol (TCP) connections to use between the Delphix Engine and the target environment. Multiple connections may improve network throughput, especially over long-latency and highly-congested networks. The default is **1**.
    e. **Number of Files to Stream Concurrently** – Select the number of files that V2P should stream concurrently from the Delphix Engine to the target environment. The default is **3**.
11. Click **Next**.
12. Select whether you want to have an email sent to you when the export process completes.
13. Click **Submit**.

Post-requisites

If you did not select **Open Database After Recovery**, follow the instructions in Manually Recovering a Database after V2P to complete the database open process.

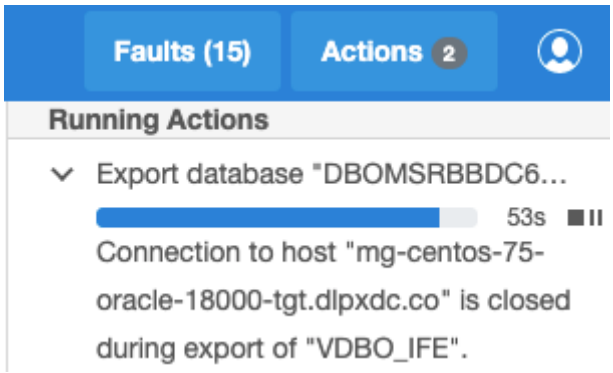
Resumable V2P

Resumable V2P is a capability that allows you to suspend a V2P operation and then resume it at a later time, without redoing any of the work already completed. For example, any portion of a file that has already been

transferred to the target environment is not re-sent. For an entire file that has already been transferred, no part is re-sent.

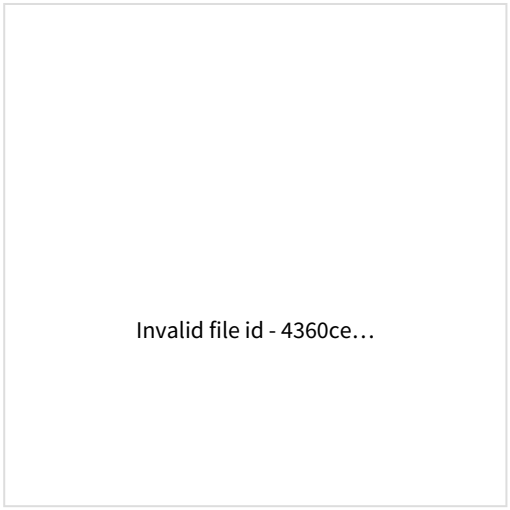
The image below presents a progress bar, a stop button, and a pause button while a V2P is running. To manually suspend a V2P operation:

1. Click the pause button.



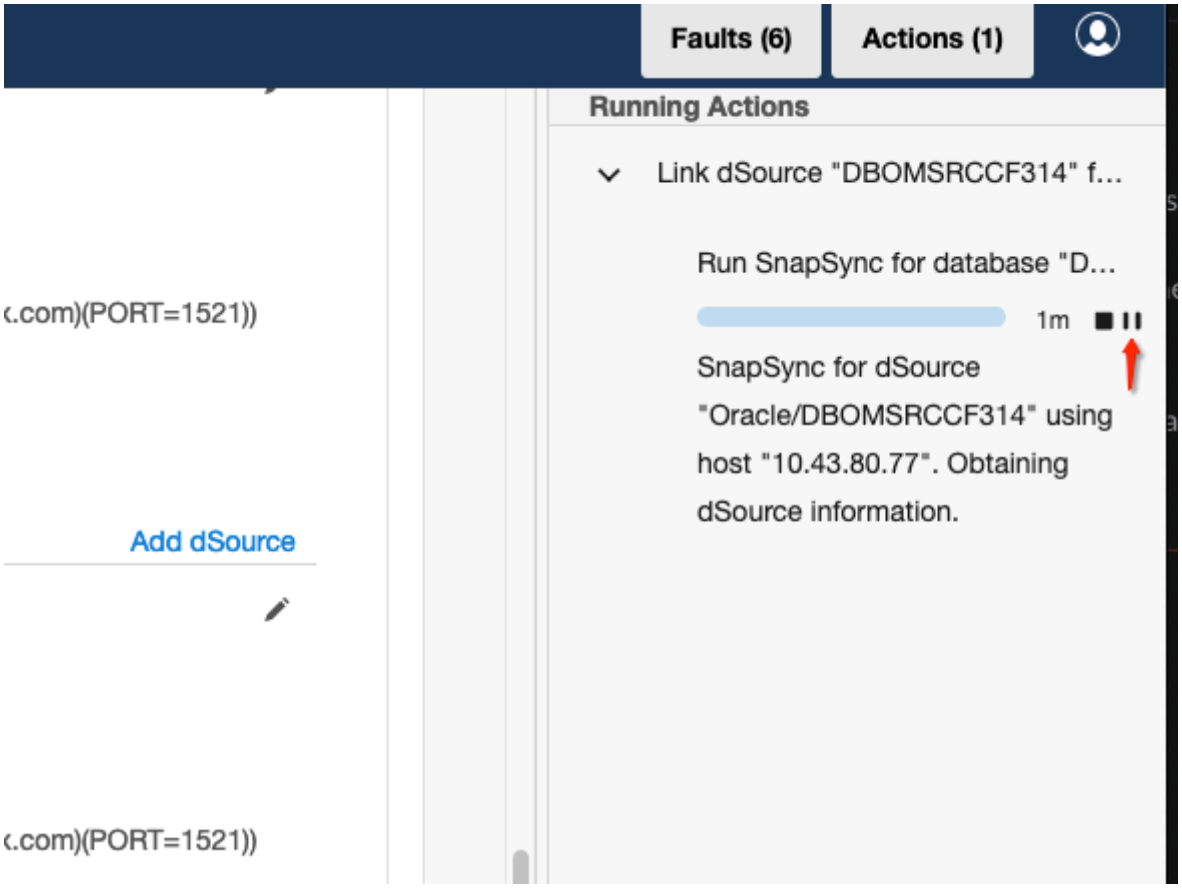
The next image represents a message alert generated after a V2P job has been suspended. To manually resume the job:

1. Click the play button.



The below image indicates that you can also pause and resume the initial snapsync from the **Actions** panel.

1. Click the resume button



Recoverable errors

Broadly speaking, a "recoverable error" is an error condition caused by a disruption in the environment or on the target host, not errors in the actual V2P operation. Examples of recoverable errors include:

- A timeout due to a network outage
- Running out of disk space on the target environment
- An inability to create directories or files on the target environment

You can often address recoverable errors by taking some action to fix the problem, such as freeing up space on the target environment.

Auto-suspend

A V2P operation that encounters a recoverable error is auto-suspended: it appears as a suspended job in the user interface (UI), with a message detailing the error condition. Once you have fixed the error, you can simply resume the job. Alternatively, you can cancel the job. Just as when you manually suspend and resume a job, any portion of a file that has been transferred, including possibly the entire file itself, is not re-sent when the job resumes.

Export an Oracle VDB or a vPDB to a physical ASM or exadata database

Overview

This article describes how to perform an export or an in-place conversion of a virtual database (VDB) or a virtual pluggable database (vPDB) into a physical database stored on Oracle Automatic Storage Management (ASM) disk groups. No intermediate storage is needed; the database files are moved directly from Delphix into the ASM diskgroup(s).

This procedure can be used to export an Oracle VDB, or a vPDB in a Linked CDB, to ASM disk group(s), including disk group(s) residing in an Oracle Exadata machine. The procedure follows Oracle's recommended best practice of using a single disk group for data files. A separate disk group can be specified for redo log files.

> This procedure can be performed using the CLI only and applies to all Oracle RDBMS Versions supported b y Delphix. For CLI commands, refer to the CLI cookbook: export a non-multitenant virtual Oracle database to ASM article.
> Furthermore, it is also fully supported with TDE- enabled virtual databases provisioned through Delphix.

Prerequisites

1. Oracle Managed Files (OMF) must be enabled on the VDB or vPDB before export can be performed. OMF eliminates the need for the DBA to directly manage the operating system files that comprise an Oracle Database. As a result of this OMF requirement, it is expected that all database file names of the exported physical database would change.
2. The following conditions must be met prior to the export operation. Export will fail if any of these conditions are not met:
    a. Sufficient storage space must be available in the target ASM diskgroup for datafiles and the target ASM diskgroup for online logs if specified. The validation of the target ASM diskgroup for online logs is only applicable in the case of VDBs and not vPDBs.
    b. While exporting a vPDB, if a new PDB name is specified:
        i. it must meet all the naming constraints as defined in the Oracle documentation.
        ii. it must be different from the existing PDBs in the target CDB.
    c. While exporting a VDB, if a new database unique name is specified:
        i. it must meet all the naming constraints as defined in the Oracle documentation.
        ii. it must be different from the database unique name of any other database on the same host.
        iii. it must be different from the database unique name of any RAC database which has a node on this host even if the node is down.
    d. No offline datafiles or tablespaces must exist in the VDB or vPDB.
    e. The VDB or vPDB on which the export is initiated must be Open.
    f. Export of a VDB in a RAC environment must be performed as the Oracle user, otherwise the export will fail. This is required because Delphix issues srvctl commands to configure the resulting physical database in RAC and these commands can only be run with Oracle user privileges.
    g. No other job must be running on the virtual database or its associated environment.
    h. When running export of a VDB or vPDB in a RAC environment, ensure that the states of all the cluster nodes are displayed as 'Enabled' in the Delphix management GUI. If one of the cluster nodes is disabled, the export operation will fail, although the physical copy has been completed and it is usable, however the VDB or vPDB will need to be force disabled manually.
3. If you want to export a vPDB in a vCDB, the vPDB must be migrated to a Linked CDB and then enabled, or alternatively, provision a child vPDB from this vPDB snapshot to a linked CDB and then perform the export.

Procedure

1. Delphix takes a new snapshot of the virtual database/pluggable database before starting export. Also, Delphix retains the timeflow and all its snapshots after the virtual source is exported. As such, after export, the virtual source can be easily migrated and rewinded to the previously created snapshots.
2. It is recommended that the export be performed on a newly provisioned VDB or vPDB, although it can still be performed on your existing VDB or vPDB. The reason is it simplifies the post export process to re-enable the existing VDB or vPDB.
3. Please refer to the appropriate CLI cookbook sections for the procedure to export a VDB or export a vPDB to a physical ASM or Exadata database.

Performance considerations before running the export

1. When deciding the number of RMAN channels to use, there are tradeoffs between speed and resource consumption on the host.
2. The number of RMAN channels should not be more than the number of datafiles.
3. Similar to selecting the number of RMAN channels to perform backup, if impact to other databases is not a concern, then setting the number of channels should be increased to the point of diminished returns. Otherwise it is a compromise between what the system can handle and how fast we want the export to finish.
4. By default it is set to 8, but this value might be too large for some environments and should be adjusted down appropriately.

Considerations after successful export of a VDB or vPDB to ASM

After the export is successful, no Delphix operations are allowed on the VDB/vPDB except migrate or a fresh provision of a new child VDB or vPDB. However, there may be situations where you may need to re-enable the VDB or vPDB, please perform a migrate of the VDB/vPDB to a different host and/or CDB and then rewind the VDB/vPDB to it's latest snapshot

1. After successfully performing export of a VDB to replace a linked physical database that is damaged and is unusable with new physical database having the same name, same unique name and same SID as the original damaged database, the new physical database can be linked back to Delphix engine using the following steps:
    a. Delete or Migrate the VDB that was used to create a new physical database to a different environment.
    b. Refresh the environment where the new physical database is created.
    c. Detach the dSource from the original source database.
    d. Force attach the dSource to the newly created physical database.
2. After successfully performing export of a vPDB with the new physical PDB name that is same as the vPDB name, the new physical PDB can be linked back to Delphix engine using the following steps:
    a. Delete or migrate the vPDB to a different CDB.
    b. Refresh the environment where the new physical PDB is created.
    c. Detach the dSource PDB from the original source PDB.
    d. Force attach the dSource to the newly created physical PDB.

Move an Oracle VDB to a physical ASM or exadata database

> ⚠ Deprecation Notice
> **Warning: T**his procedure is deprecated as of 9.0.0.0. Delphix recommends using the new export feature in 9.0.0.0 that is documented in the Export an Oracle VDB or vPDB to a Physical ASM or Exadata Database article. This procedure will be eventually removed in future versions of the Delphix Continuous Data Engine.

Overview

This article describes how to move a virtual database (VDB) into a physical database stored on Oracle Automatic Storage Management (ASM) disk groups. This is a scripted procedure that is assisted (but not fully automated) by Delphix. No intermediate storage is needed; the database files are moved directly from Delphix into the ASM diskgroup(s).

This procedure can be used to export a VDB to ASM disk group(s), including disk group(s) residing in an Oracle Exadata machine. The `move-to-asm.sh` script follows Oracle's recommended best practice of using a single disk group for data files. A separate disk group can be specified for redo log files. Professional Services can be engaged if customization is needed.

> ⓘ This procedure applies to **all** Oracle RDBMS Versions supported by Delphix; in addition, it is not supported by PDB/Multi-tenant type databases.

Procedure

1. Create an ASM disk group that will contain all the database files. Optionally, create a separate disk group for redo log files.
2. Provision a VDB on the target machine that is running Oracle ASM or Exadata in order to do the export; this VDB will be deleted after the export is complete. This temporary VDB can be provisioned from an existing VDB. You can also provision it from a dSource, which allows you to achieve a full restore of the original source ASM database.
3. Login to the target machine (or a node of the target RAC cluster) where the VDB instance and ASM instance are running. The directory will be in the form:

   ```
   <toolkit_path>/Delphix_<engine_id><user><host|cluster>/databases/oracle/
   <db_unique_name>/<database_sid>/
   ```

4. For a RAC database, be sure to use the directory path containing the SID of the instance running on the node where you have logged in.
5. Ensure that Oracle environment variables ORACLE_HOME,ORACLE_SID, and CRS_HOME (RAC only) are correctly set for the VDB that needs to be moved.
6. Execute the script `move-to-asm.sh` as the **Environment User** who provisioned the single instance VDB. For a RAC VDB, the **Environment User** selected to execute the `move-to-asm.sh` *must* be the Oracle installation owner. This is due to an Oracle restriction that only the installation owner can invoke `srvctl` to add or remove database configurations. Alternatively, enter `move-to-asm.sh` as a **Post-Provision Hook Script** when provisioning the VDB. This will provision and move the VDB into ASM disk groups in a

single flow. You must specify the `-noask` option to execute in non-interactive mode. For more information about hook scripts, visit the Hook Scripts for Automation and Customization article.

Command syntax: `move-to-asm.sh [-noask] [-parallel #] [-dbunique db_unique_name] <data_diskgroup> [<redo_diskgroup>]`

Arguments	Description
`-noask` [optional]	Do not prompt for confirmation before moving the VDB. The default is to prompt.
`-parallel #` [optional]	The number of RMAN channels used to move the VDB to ASM. The default is 8.
`-dbunique <>` [optional]	Unique database name for the resulting physical database. The default is the unique name of the VDB. The `-dbunique` argument only works if a database with that name that you select already exists.
`<>` [required]	Target ASM disk group for data, server parameter and control files.
`<>` [optional]	Target ASM disk group for redo log files. Default is `<>`.

move-to-asm script

The *move-to-asm.sh* script generates several output files. These files are all written to the working directory of the script:

```
init<oracle_sid>_run<process-id>_moveasm.ora
```

The log file for the operation:

```
move-to-asm.sh_<oracle_sid>_run<process-id>.log
```

The init.ora parameter file created for the ASM DB instance:

```
source_init<oracle_sid>.ora
```

The init.ora for the source database (from which the VDB was created).

Finishing steps

The final steps to manually execute are shown when the script completes and included in the script output.

1. For Single Instance only: copy generated init<$ORACLESID>.ora parameter file to the default $ORACLE_HOME/dbs/init<$ORACLE_SID>.ora

2. Modify initialization parameters to match the original source database parameters, if necessary.

As a convenience to assist with this step, the source database parameters are restored as source_init<$ORACLE_SID>.ora in the execution directory. Be sure to copy this file to a save place *before* deleting the temporary VDB.

The *move-to-asm.sh* script generates several output files. These files are all written to the working directory of the script and should also be copied to a safe place if needed:

```
move-to-asm.sh_<oracle_sid>_run<process-id>.log
```

The log file for the operation:

```
init<oracle_sid>_run<process-id>_moveasm.ora
```

The init.ora parameter file created for the ASM DB instance:

```
source_init<oracle_sid>.ora
```

The init.ora for the source database (from which the VDB was created).

3. Delete the Delphix VDB that was moved.
4. Start up the physical database that will now run on ASM.
   (A RAC database is automatically started up by the `move-to-asm.sh` script using `srvctl` but may have been stopped when the VDB was deleted, if it has the same name.)

Sample execution

```
$ cd toolkit/Delphix_564dc816_d457_6224_1327_4d43fd1b3d97_oracle_cluster/databases/
oracle/RAC2ASM/RAC2ASM1
$ export ORACLE_HOME=/u01/app/oracle/product/12.2.0.1/dbhome_1
$ export ORACLE_SID=RAC2ASM1
$ sh move-to-asm.sh -noask +DATA +REDO
END_OF_SETUP
============================================================
Virtual-to-ASM script (move-to-asm.sh v5.2) for Delphix
Copyright (c) 2013, 2018 by Delphix. All rights reserved.
============================================================

Moving database RAC2ASM to ASM: started at Wed Feb 19 16:08:43 EST 2020
db_unique_name => RAC2ASM
ORACLE_SID => RAC2ASM1
ORACLE_HOME => /u01/app/oracle/product/12.2.0.1/dbhome_1
Datafile diskgroup => +DATA
Online redo diskgroup => +REDO
RMAN Channels => 8

Create cluster database RAC2ASM
Generate script to move tempfiles to ASM
Generate script to drop old tempfiles
Generate script to drop offline tablespaces
```

```
Generate script to make read-only tablespaces read-write
Generate script to make tablespaces read-only again
Make read-only tablespaces read-write
Remove offline tablespaces
Updating server parameter file with ASM locations
Creating temporary initialization parameter file
Move spfile to ASM
Move datafiles to ASM: started at Wed Feb 19 16:09:41 EST 2020
Move datafiles to ASM: completed at Wed Feb 19 16:10:31 EST 2020
Stop cluster database RAC2ASM
Restart cluster database RAC2ASM
Remove old tempfiles
Move tempfiles into ASM
Move Online logs
Restore any read-only tablespaces
Shutdown database
Starting up RAC database RAC2ASM...
Database RAC2ASM moved to ASM: completed at Wed Feb 19 16:12:45 EST 2020

Final Steps to complete the move to ASM:
Source parameters are restored at /home/oracle/toolkit/
Delphix_564dc816_d457_6224_1327_4d43fd1b3d97_oracle_cluster/databases/oracle/RAC2ASM/
RAC2ASM1/source_initRAC2ASM1.ora
*.audit_file_dest='/u01/app/oracle/admin/DBOMSRD914EA/adump'
*.audit_trail='DB'
*.cluster_database=TRUE
*.compatible='12.2.0'
*.control_files='+DATA/DBOMSRD914EA/CONTROLFILE/current.297.1027776391','+REDO/
DBOMSRD914EA/CONTROLFILE/current.264.1027776391'
*.db_block_size=8192
*.db_create_file_dest='+DATA'
*.db_create_online_log_dest_1='+DATA'
*.db_create_online_log_dest_2='+REDO'
*.db_name='DBOMSRD9'
*.db_recovery_file_dest='+REDO'
*.db_recovery_file_dest_size=6989807616
*.db_unique_name='DBOMSRD914EA'
*.diagnostic_dest='/u01/app/oracle'
*.dispatchers='(PROTOCOL=TCP) (SERVICE=DBOMSRD914EAXDB)'
*.dlpx_recovery_log_archive_format='arch_%t_%s_%r.log'
*.instance_number=1
*.listener_networks=''
*.local_listener='(ADDRESS=(PROTOCOL=TCP)(HOST=10.43.5.243)(PORT=1521))'
*.log_archive_format='%t_%s_%r.dbf'
*.memory_max_target=1073741824
*.memory_target=1073741824
*.nls_language='AMERICAN'
*.nls_territory='AMERICA'
*.open_cursors=300
*.processes=300
*.remote_listener='node1.sample.delphix.com:1521'
*.remote_login_passwordfile='EXCLUSIVE'
```

```
*.thread=1
*.undo_tablespace='UNDOTBS1'

Copy this file to a safe location before removing the VDB

1) Delete VDB on Delphix.
2) Modify initialization parameters to match source and restart.
```

Refresh/Rewind an Oracle virtual database exported to physical ASM database

The "Export an Oracle VDB or a vPDB to a Physical ASM or Exadata Database" feature introduced in Delphix 9.0.0.0, allowed the export or in-place conversion of a virtual database to ASM, but the refresh or rewind operation was not allowed after the virtual database was exported to ASM.

Starting with the 12.0.0.0 release, Delphix allows a refresh or rewind operation to be performed on a virtual database (VDB or vPDB) after the V2ASM export job completes.

How to allow refresh/rewind on a virtual database after it has been exported to ASM?

A new parameter `allowRefreshRewindPostV2P` under `operationsPostV2P` has been introduced to allow refresh/rewind on the virtual database after it has been exported to ASM.

```
ip-10-110-247-210 database export *> ls
Properties
    type: OracleDBExportParameters (*)
    storageStrategy:
        type: OracleExportASMStorageStrategy (*)
        asmLayout:
            type: OracleASMLayout (*)
            defaultDataDiskgroup: +DATA (*)
            redoDiskgroup: (unset)
    transferStrategy:
        type: OracleExportDBInPlaceTransferStrategy (*)
        dbUniqueName: (unset)
        operationsPostV2P: (unset)
        rmanChannels: 8 (*)
        rmanFileSectionSizeInGb: 0 (*)
        virtualSource: VDBOMSRASF1B4_25I (*)
```

To allow the refresh or rewind operation on the virtual database after the export operation, set the parameter `allowRefreshRewindPostV2P` under `operationsPostV2P` as true.

On CLI, run the following command to set the same.

```
set transferStrategy.operationsPostV2P.allowRefreshRewindPostV2P=true
```

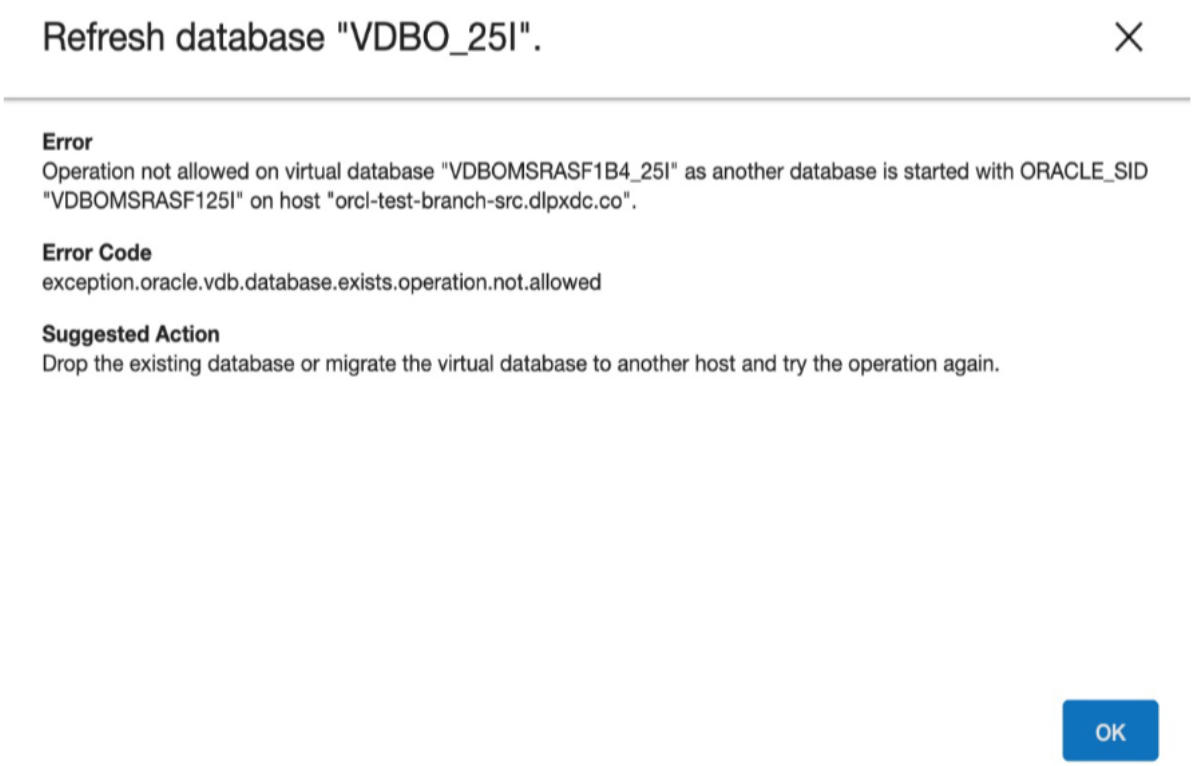
```
ip-10-110-247-210 database export *> ls
Properties
    type: OracleDBExportParameters (*)
    storageStrategy:
        type: OracleExportASMStorageStrategy (*)
        asmLayout:
            type: OracleASMLayout (*)
            defaultDataDiskgroup: +DATA (*)
            redoDiskgroup: (unset)
    transferStrategy:
        type: OracleExportDBInPlaceTransferStrategy (*)
        dbUniqueName: (unset)
```

```
        operationsPostV2P:
            type: OracleExportOperationsPostV2P (*)
            allowRefreshRewindPostV2P: true (*)
        rmanChannels: 8 (*)
        rmanFileSectionSizeInGb: 0 (*)
        virtualSource: VDBOMSRASF1B4_25I (*)
```

After the export job, a physical database will be running on Oracle Host.

What happens if a refresh or rewind operation is attempted on a virtual database after export?

On refresh or rewind operation, the user will get an exception.



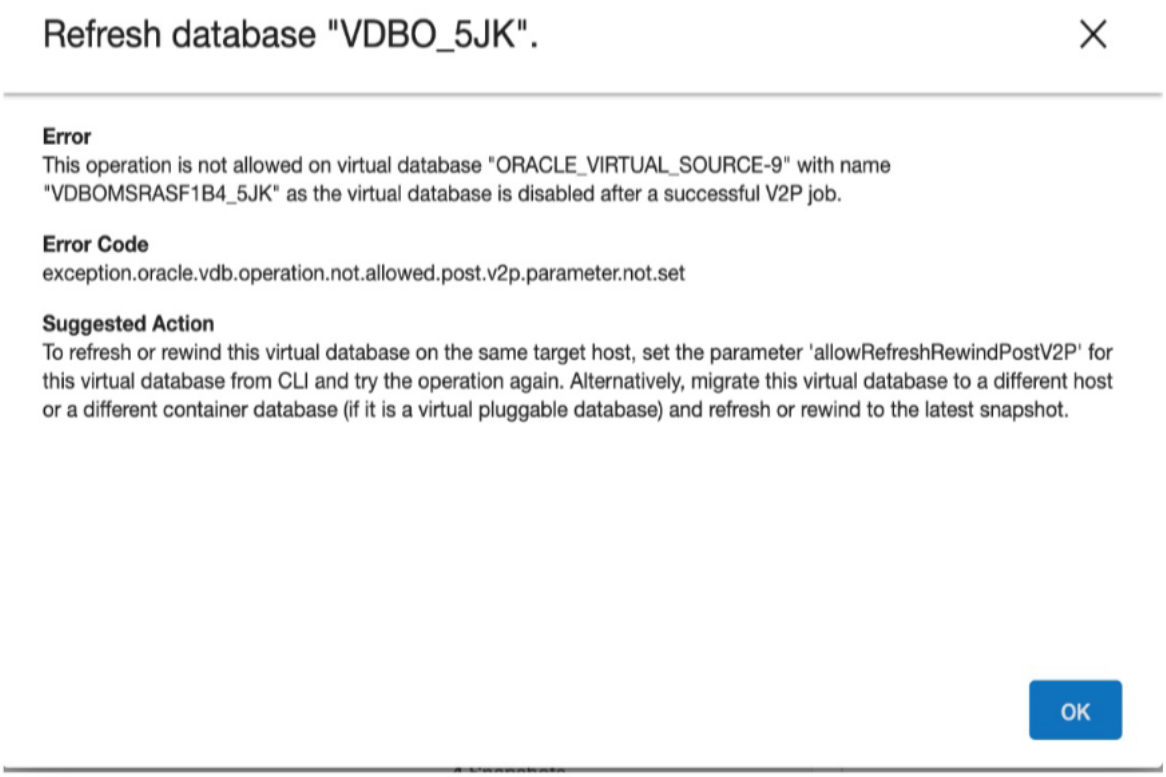
The refresh or rewind operation is not allowed because there is another database running on the Oracle host with the same SID. To go ahead with the refresh or rewind operation, shut down the physical database and then try doing the refresh operation again. The refresh or rewind operation will be successful now.

What if a user forgets to set the operationsPostV2P during export?

```
ip-10-110-247-210 database export *> ls
Properties
    type: OracleDBExportParameters (*)
    storageStrategy:
        type: OracleExportASMStorageStrategy (*)
        asmLayout:
            type: OracleASMLayout (*)
            defaultDataDiskgroup: +DATA (*)
```

```
        redoDiskgroup: (unset)
    transferStrategy:
        type: OracleExportDBInPlaceTransferStrategy (*)
        dbUniqueName: (unset)
        operationsPostV2P: (unset)
        rmanChannels: 8 (*)
        rmanFileSectionSizeInGb: 0 (*)
        virtualSource: VDBOMSRASF1B4_5JK (*)
```

After the export job, if the user tries refresh or rewind operation on a virtual database, an exception will be thrown because the parameter `allowRefreshRewindPostV2P` is not set.



How can this parameter be set from the Delphix Engine CLI using source update API?

After the export operation, the Delphix source update API does not allow updating the parameters for a virtual database that has been disabled after an export operation. The only exception is the `allowRefreshRewindPostV2P` parameter, which can be set as shown below:

```
ip-10-110-247-210> source
ip-10-110-247-210 source> select VDBOMSRASF1B4_5JK
ip-10-110-247-210 source 'VDBOMSRASF1B4_5JK'> update
ip-10-110-247-210 source 'VDBOMSRASF1B4_5JK' update *> set allowRefreshRewindPostV2P=t
rue
ip-10-110-247-210 source 'VDBOMSRASF1B4_5JK' update *> commit;
    Dispatched job JOB-116
    SOURCE_UPDATE job started for "VDBOMSRASF1B4_5JK".
```

```
    SOURCE_UPDATE job for "VDBOMSRASF1B4_5JK" completed successfully.
```

An attempt to update any other virtual database parameter will fail with the following error:

```
ip-10-110-247-210 source 'VDBOMSRASF1B4_5JK' update *> set
allowAutoVDBRestartOnHostReboot=true;
ip-10-110-247-210 source 'VDBOMSRASF1B4_5JK' update *> commit;
    Dispatched job JOB-117
    SOURCE_UPDATE job started for "VDBOMSRASF1B4_5JK".
    This operation is not allowed on source 'ORACLE_VIRTUAL_SOURCE-9' with name
'VDBOMSRASF1B4_5JK' as the source is disabled after a successful V2P job.
  Action: To re-enable, migrate this virtual source to a different host or a
different CDB (if it is a virtual pluggable database) first and then rewind to the
latest snapshot. Delete the virtual source if it is no longer needed. This does not
impact the newly created physical database.
ip-10-110-247-210 source 'VDBOMSRASF1B4_5JK'>
```

After setting the parameter `allowRefreshRewindPostV2P` to true, the refresh or rewind operation will be successful from GUI.

Provisioning from a replicated Oracle VDB

This topic describes how to provision from a replicated dSource or virtual database (VDB).

The process for provisioning from replicated objects is the same as the typical VDB provisioning process, except that first you need to select the replica namespace containing the replicated object.

Prerequisites

- You must have a dSource or VDB that has been replicated from one Delphix Engine to another, as described in Replication Overview
- The Delphix Engine containing the replicated dSource or VDB must have a compatible target environment that it can use to provision a VDB from the replicated dSource or VDB.

Procedure

1. On the Delphix Engine containing the replicated dSource or VDB, login to the **Delphix Management** application.
2. In the top menu bar, click **Manage**.
3. Select **Datasets**.
4. From the list of replica namespaces, select the **replica namespace** that contains the dSource or VDB from which you want to provision.
5. The provisioning process is now identical to the process for provisioning standard objects.

Post-requisites

Once the provisioning job has started, the user interface will automatically display the new VDB in the live system.

Provisioning a TDE-enabled vPDB to a cluster target

Overview

Provisioning a Virtual Pluggable Database (vPDB) to a linked RAC container database first involves using the GUI or CLI to specify the vPDB parameters (such as the vPDB name and target container) along with the snapshot to provision from. Once the provision job is started with these parameters, the Delphix Engine does the following:

1. Chooses an available instance from the target cluster and mounts the snapshot files on that instance.
2. Creates and opens (in mount mode) the auxiliary container database on the target instance, using the snapshot files. The auxiliary container database will have both the CDB and PDB data files from the dSource.
3. Completes recovery to bring the auxiliary container database into a consistent state.
4. Finalizes the state of the auxiliary database and unplugs the vPDB datafiles.
5. Plugs the vPDB into the target database, and opens it in read-write mode on the same target instance used for recovery.
6. Spawns start jobs to open the vPDB in read-write mode on the remaining target instances in parallel.

If the dSource is TDE-enabled, then Delphix will need to perform additional operations to complete the provision of a TDE-enabled vPDB to a TDE-enabled cluster target (indicated in red):

1. Chooses an available instance from the target cluster and mounts the snapshot files on that instance.
2. Creates a keystore with the necessary keys to apply encrypted archived log files on the target instance.
3. Creates and opens (in mount mode) the auxiliary container database on the target instance, using the snapshot files. The auxiliary container database will have both the CDB and PDB data files from the dSource.
4. Completes recovery to bring the auxiliary container database into a consistent state.
5. Rotates the vPDB and auxiliary CDB master encryption keys by generating new keys that are unique to the vPDB/auxiliary CDB and not associated with the source PDB or CDB.
6. Exports only the newly generated keys to an exported keyfile to enable unplug.
7. Finalizes the state of the auxiliary database and unplugs the vPDB datafiles.
8. Imports the vPDB key from the exported keyfile into the target keystore.
9. When provisioning to a vCDB target, converts the auxiliary CDB into the final vCDB and creates the vCDB keystore from the auxCDB keystore.
10. Opens the Keystore on each node.
11. Plugs the vPDB into the target database, and opens it in read-write mode on the same target instance used for recovery.
12. Spawns start jobs to open the vPDB in read-write mode on the remaining target instances.

All the same information needed for a single instance TDE-enabled vPDB provision is also required for a cluster TDE-enabled vPDB provision, specifically the target keystore password, parent keystore path and password, and encryption secret. The keystores root path is required for a cluster provision.

Shared storage for keystores

In a cluster database, the database files are on shared storage, which is accessible from all instances in the cluster. If the database is encrypted, then the keystore file itself is also located on the operating system. Oracle recommends that the keystore also be on shared storage, on a different disk from the database files. If the keystore is not on shared storage, then it must be copied to all instances in the cluster after any changes, such as importing a key or generating a new key. Similarly, Delphix recommends that the parent keystore specified for the provision also be on shared storage. If not, then the same file must be copied to all of the instances before the vPDB is first provisioned, and any updates to the parent keystore must also be copied to all of the instances before any vPDB refresh or rewind.

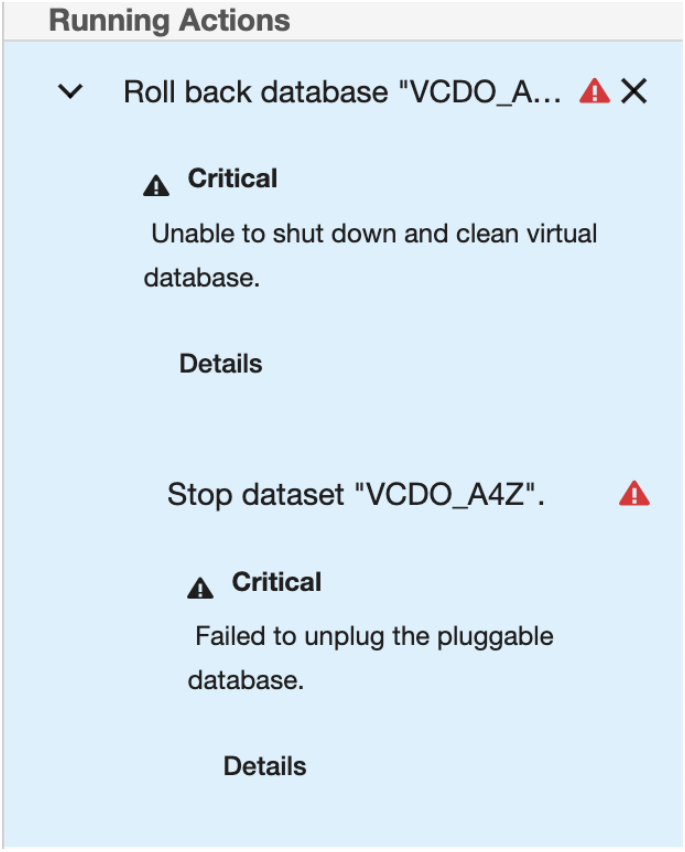
As the autologin wallet is located in the same location as the password-based keystore, it should also be on shared storage. For this reason, `local_autologin` wallets will not work properly, as they will be accessed from multiple nodes in the cluster.

Keystores root path requirements

The artifact directory for a given vPDB is created under the keystores root. As a subsequent operation on the vPDB may choose a different instance than the one used for the initial provision, the artifact directory needs to be accessible from all instances in the cluster. Delphix requires that the keystores root be specified for a TDE-enabled vPDB provision to a cluster target, and furthermore that it be located on shared storage. The engine will validate that this is the case before proceeding with the initial provision.

Refreshing or rewinding a broken/unusable virtual PDB

A virtual pluggable database could be broken/unusable if it cannot be operated upon from the Delphix Management such that all Delphix operations involving opening or unplugging of the virtual pluggable database from its container database like Disable, Enable, Refresh and Rewind fail.



A virtual PDB can get into a such state because of (but not limited to):

- Datafile is deleted or gets corrupted
- Datafile is renamed or moved
- Oracle DB instance crashed and archive logs needed to perform recovery are lost/deleted/unavailable.

Using force option

A virtual PDB can be brought out of the bad state by Refreshing or Rewinding it with a `force` option. The force option is only available in the CLI. The `force` parameter is part of `OracleRefreshParameters` and `OracleRollbackParameters` for the Refresh and Rewind operations respectively. The default value for the `force` parameter is `false`. For details on how to use the force option, refer to CLI cookbook: Force refresh/ rewind a virtual PDB

Implications of `force` option

Using the `force` option with Refresh/Rewind has the following implications:

- A forced Refresh/Rewind cannot be subsequently undone using the ‘Undo Refresh or Rewind’ functionality.A user cannot switch to the timeflow prior to the force Refresh/Rewind using switchTimeflow API
- Timeflow before the force Refresh/Rewind will remain in the broken state, however, any snapshots in that timeflow will continue to remain provisionable.
- In the case of Rewind, Delphix may not be able to identify the 'last good point in timeflow' for the vPDB.

Oracle hook operations

Oracle RAC

When linking from, or provisioning to Oracle RAC environments, hook operations will not run once on each node in the cluster. Instead, the Delphix Engine picks a node in the cluster at random and guarantees all operations within any single hook will execute serially on this node.

Note that the Delphix Engine does not guarantee the same node is chosen for the execution of every hook but does guarantee that Pre-/Post- hook pairs (such as Pre-Sync and Post-Sync) will execute on the same node.

Shell operations

RunCommand operation

The RunCommand operation runs a shell command on a Unix environment using whatever binary is available at `/bin/sh`. The environment user runs this shell command from their home directory. The Delphix Engine captures and logs all output from this command. If the script fails, the output is displayed in the Delphix Management application and command-line interface (CLI) to aid in debugging.

If successful, the shell command must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Examples of RunCommand operations

You can input the full command contents into the RunCommand operation.

```
remove_dir="$DIRECTORY_TO_REMOVE_ENVIRONMENT_VARIABLE"

if test -d "$remove_dir"; then
    rm -rf "$remove_dir" || exit 1
fi

exit 0
```

If a script already exists on the remote environment and is executable by the environment user, the RunCommand operation can execute this script directly.

```
/opt/app/oracle/product/10.2.0.5/db_1/dbs/myscript.sh "$ARG_ENVIRONMENT_VARIABLE"
"second argument in double quotes" 'third argument in single quotes'
```

RunBash operation

The RunBash operation runs a Bash command on a Unix environment using a `bash` binary provided by the Delphix Engine. The environment user runs this Bash command from their home directory. The Delphix Engine captures and logs all output from this command. If the script fails, the output is displayed in the Delphix Management application and command-line interface (CLI) to aid in debugging.

If successful, the Bash command must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Example of RunBash operations

You can input the full command contents into the RunBash operation.

```
remove_dir="$DIRECTORY_TO_REMOVE_ENVIRONMENT_VARIABLE"

# Bashisms are safe here!
if [[ -d "$remove_dir" ]]; then
    rm -rf "$remove_dir" || exit 1
fi

exit 0
```

Shell operation tips

Using `nohup`

You can use the `nohup` command and process backgrounding from resource in order to "detach" a process from the Delphix Engine. However, if you use `nohup` and process backgrounding, you MUST redirect `stdout` and `stderr.`

Unless you explicitly tell the shell to redirect `stdout` and `stderr` in your command or script, the Delphix Engine will keep its connection to the remote environment open while the process is writing to either `stdout` or `stderr` . Redirection ensures that the Delphix Engine will see no more output and thus not block waiting for the process to finish.

For example, imagine having your `RunCommand` operation background a long-running Python process. Below are the bad and good ways to do this.

Bad Examples

- `nohup python file.py & # no redirection`
- `nohup python file.py 2>&1 & # stdout is not redirected`
- `nohup python file.py 1>/dev/null & # stderr is not redirected`
- `nohup python file.py 2>/dev/null & # stdout is not redirected`

Good Examples

- `nohup python file.py 1>/dev/null 2>&1 & # both stdout and stderr redirected, Delphix Engine will not block`

Other operations

RunExpect operation

The RunExpect operation executes an Expect script on a Unix environment. The Expect utility provides a scripting language that makes it easy to automate interactions with programs which normally can only be used interactively, such as `ssh`. The Delphix Engine includes a platform-independent implementation of a subset of the full Expect functionality.

The script is run on the remote environment as the environment user from their home directory. The Delphix Engine captures and logs all output of the script. If the operation fails, the output is displayed in the Delphix Management application and CLI to aid in debugging.

If successful, the script must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Example of a RunExpect operation

Start an `ssh` session while interactively providing the user's password.

```
spawn ssh user@delphix.com
expect {
    -re {Password: } {
        send "${env(PASSWORD_ENVIRONMENT_VARIABLE)}\n"
    }
    timeout {
        puts "Timed out waiting for password prompt."
        exit 1
    }
}
exit 0
```

Oracle environment variables

to access the dSource or VDB.

dSource environment variables

Environment variable	Description
`ORACLE_SID`	The SID of the dSource
`ORACLE_BASE`	The home directory of the Oracle software hosting the dSource
`ORACLE_HOME`	The Oracle Home for the dSource
`CRS_HOME` (only set for RAC dSources)	The home directory for cluster services hosting the dSource

Environment variable	Description
`ORAENV_ASK`	Always set to NO
`DELPHIX_DATABASE_NAME`	The database name reported by Oracle
`DELPHIX_DATABASE_UNIQUE_NAME`	The database unique name reported by Oracle
`DELPHIX_PDB_NAME` (only set for PDBs)	The PDB name reported by Oracle

VDB environment variables

Environment variable	Description
`ORACLE_SID`	The SID for the VDB
`ORACLE_BASE`	The home directory for the Oracle software hosting the VDB
`ORACLE_HOME`	The Oracle Home for the VDB
`CRS_HOME` (only set for RAC VDBs)	The home directory for cluster services hosting the RAC VDB
`ORAENV_ASK`	Always set to NO
`DELPHIX_DATABASE_NAME`	The database name reported by Oracle
`DELPHIX_DATABASE_UNIQUE_NAME`	The database unique name reported by Oracle
`DELPHIX_PDB_NAME` (only set for PDBs)	The PDB name reported by Oracle
`DELPHIX_MOUNT_PATH`	The root of the NFS mount hosting the VDB data
`MASKING_CONNECTOR_HOST (only set for masked provisioning)`	The host that DMSuite will use for the connector

Environment variable	Description
`MASKING_CONNECTOR_PORT (only set for masked provisioning)`	The port that DMSuite will use for the connector

PATH and LD_LIBRARY_PATH configuration

`PATH` is configured by appending the `bin` directory in the Oracle home for the dSource or VDB.

`LD_LIBRARY_PATH` is configured by appending the `lib` directory in the Oracle home for the dSource or VDB.

Oracle E-Business Suite (EBS) environments and data sources

This section contains the following topics:

To view the EBS support matrix, see Oracle E-Business Suite (EBS) matrix.

Oracle E-Business Suite and Delphix conceptual overview

This topic provides a conceptual overview of the integration between the Delphix Engine and Oracle E-Business Suite (EBS).

Why use the Delphix Engine with E-Business suite?

Oracle E-Business Suite offers a multitude of enterprise resource planning (ERP), supply chain management (SCM), and customer relationship management (CRM) applications useful for running a business. Typical EBS instances are consumed horizontally across an organization but managed on-site by an IT organization. Most businesses hire consultants or in-house developers to customize these instances to fit business needs.

Production EBS instances are critical to many business processes; EBS is expected to be always available and functioning correctly. An IT organization's EBS team is responsible for maintaining this production instance: this team monitors the instance and manages how software modifications are integrated. Common EBS software modifications include patches provided by Oracle or new application logic provided by consultants or in-house developers.

However, before making a software modification to their production EBS instance, most EBS teams vet the modification. These teams typically maintain a series of non-production copies of their production EBS instance to use for the vetting process. To vet a modification, the EBS team will modify a non-production instance in an attempt to identify bugs in the modification that might lead to incorrect functionality, performance regressions, or downtime. Once a modification has been installed and tested successfully, it is considered vetted and the EBS team make the same modification to the production instance.

The problem with this process is that maintaining non-production copies of EBS is costly. EBS teams not only need to provision and maintain resources for each copy, they also must update each copy to match production every time production itself is modified. This process takes two hours at a minimum and can take up to a week or more.

The Delphix Engine can alleviate much of the pain around managing non-production copies of Oracle E-Business Suite. You can link a production EBS instance to the Delphix Engine and use the engine to provision virtual copies. These virtual copies lower storage costs associated with non-production. They also drastically reduce the amount of active attention required from the EBS team. When a change is made to the production EBS instance, the Delphix Engine can synchronize its copy; all virtual EBS instances can then be refreshed to match this new version of the production. The process of refreshing a virtual EBS instance is fully automated, so an EBS team no longer needs to invest copious amounts of time refreshing non-production instances manually.

In addition, the Delphix Engine can take snapshots of virtual EBS instances in known healthy states so that you can easily roll back risky modifications while vetting them. This functionality provides modern version control semantics for a process that previously had none.

Overview of linking and provisioning EBS

Oracle E-Business Suite is primarily comprised of a database service and a plethora of application services. The Delphix Engine supports the linking and provisioning of all EBS datasets, including the database technology stack (dbTechStack), database, and application files (appsTier). Note that the Delphix Engine can also manage custom extensions and plug-ins.

The process of linking EBS data involves creating multiple dSources:

- a dSource for the dbTechStack
- a dSource for the Oracle database used by EBS
- a dSource for the appsTier

These dSources are collectively referred to as EBS dSources. They are also the source datasets from which you can provision virtual EBS instances.

The process of provisioning a virtual EBS instance involves provisioning from each of these dSources separately to the proper environments in sequence. You can add custom configuration logic to the Delphix Engine for each EBS instance such that the linking, provisioning, and refresh processes are fully automated.

> The plugin versions for AppsTier and DB Tech Stack need to be the same, at installation and through the upgrade. Installation and usage of different versions of these plugins are not supported and may encounter operational issues.

dbTechStack dataset

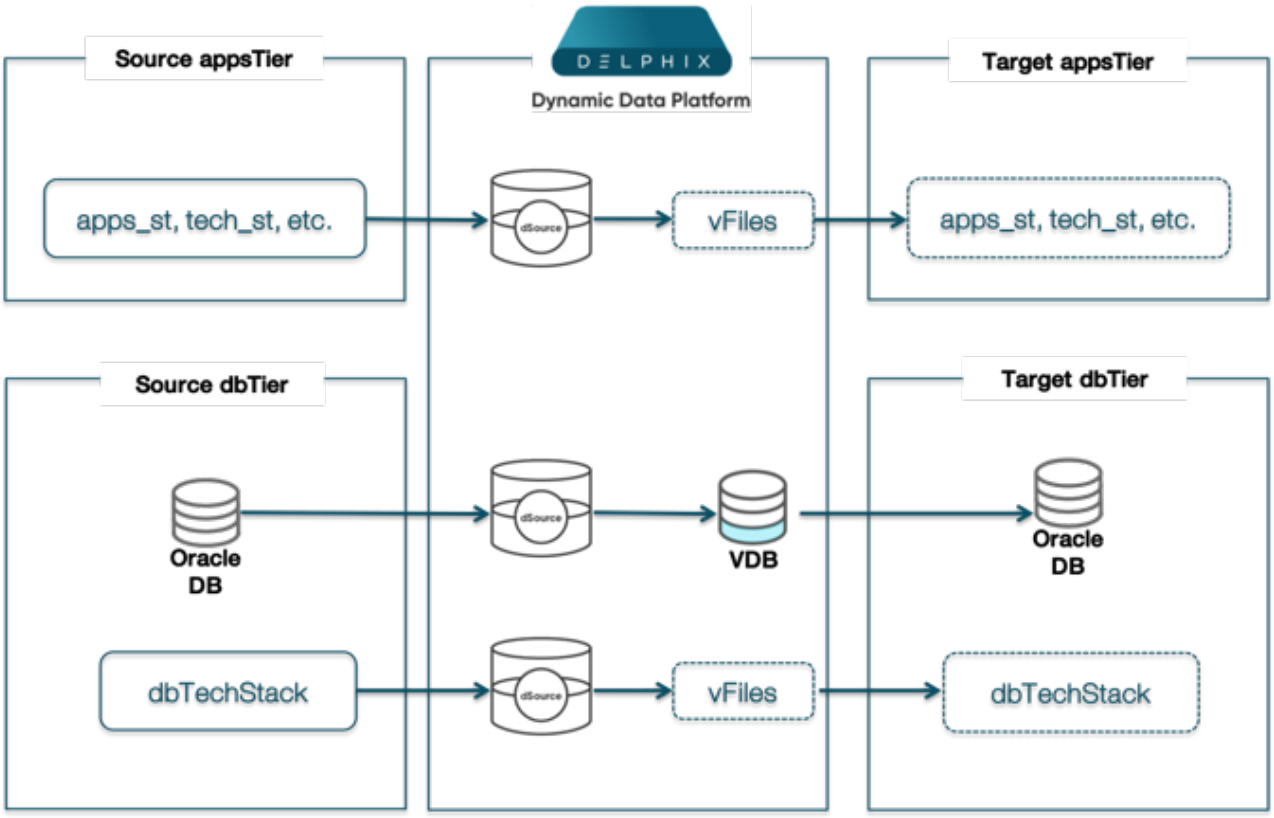
The source dbTechStack is linked using the Delphix Engine's EBS support: the linking process automatically runs pre-clone logic to ensure EBS configuration is always appropriately staged at the time of data capture. When you provision EBS, the Delphix Engine automates post-clone configuration such that a copy of the dbTechStack is available for use on the target dbTier server with no additional effort. You can add this copy of the dbTechStack to the Delphix Engine as an Oracle installation home and use it to host an EBS virtual database (VDB).

Database dataset

The database dSource is linked using the Delphix Engine's support for Oracle databases. This dSource contains database data files that EBS is currently using. For more information about managing Oracle databases. When you provision EBS, you will use the Delphix Engine to set up a copy of the EBS database on the target dbTier server. This copy of the database will be used to back virtual EBS instance’s appsTier.

appsTier dataset

The appsTier is linked using the Delphix Engine's EBS support: the linking process automatically runs pre-clone logic to ensure EBS configuration is always appropriately staged at the time of data capture. When you provision EBS, the Delphix Engine will automate post-clone configuration such that a copy of the appsTier is available for use on the target appsTier server. This virtual copy of the appsTier will connect to the provisioned EBS virtual database (VDB).



How does it work?

The Delphix Engine's linking and provisioning logic follows the Oracle-recommended cloning procedures outlined in the following Oracle Support documents found at http://docs.oracle.com:

- Cloning Oracle E-Business Suite Release 12.2 with Rapid Clone (Doc ID 1383621.1)
- Cloning Oracle Applications Release 12 with Rapid Clone (Doc ID 406982.1)

EBS HealthChecker for validating source and target environments

Overview

The Oracle E-Business Suite (EBS) HealthChecker utility performs various checks to ensure Delphix's EBS Source and Target environments (hosts) are configured correctly. We recommend using this utility to validate a fresh environment or following an environment's database upgrade.

⚠
- The scripts provided as part of this automation package are provided as-is and are intended to help you easily check the feasibility of linking and provisioning operations.
- Although the scripts have been tested by Delphix Engineering, they are not officially supported by Delphix. This means that Delphix is not liable for any bugs or defects in this implementation.
- You can create or add your own scripts to support implementing the HealthChecker operations using the EBS utilities of your choice.

Structure

The HealthChecker utility is structured as follows:

- The utility is designed to be easily extensible, with new checks being added as required.
- The utility is designed to be easily customizable, with parameters that can be changed to suit specific requirements.

Workflows

The HealthChecker utility provides four specific environment workflows:

- AppsTier - Source
- AppsTier - Target
- DbTier - Source
- DBTier - Target

Prerequisites

- Linux or Solaris environment

ⓘ **Exclusions For Solaris OS**

1. For Solaris, Delphix will not check the sudoers entries. The utility won't ask if you are running privilege elevation.
   If you are running privilege elevation on Solaris, see the Host requirements section on Preparing a source EBS R12.2 instance for linking for required sudoers entries.
2. Kernel `shmmax` and `shmall` values are not being checked for Solaris, see /etc/hosts section on Preparing target EBS R12.2 environments for provisioning for required values.

Instructions

Follow the steps below to run the EBS HealthChecker utility. It runs in interactive mode through your command terminal.

1. Download the EBS HealthChecker utility from the Delphix Download Portal.
    a. Find it beside the standard EBS connector download.
2. Manually copy it to the Source or Target environment you intend to validate.
3. Extract the zip file and view its contents.
4. Grant the `main.sh` file execution permissions using the command: `chmod +x main.sh`
5. Run the HealthChecker tool using the command: `./main.sh`
6. Follow the terminal prompts to validate your chosen environment.

The following is a sample runtime log from an AppsTier Target Environment:

```
[applvis@e27a171at-app ebs-healthchecker]$ ./main.sh
Select the host type:
1. appstier
2. dbtier
Enter your choice (1 or 2): 1
Select the appstier host type:
1. source
2. target
Enter your choice (1 or 2): 2
Executing checks for appstier target host..
Enter the toolkit directory path: /var/tmp
Enter appstier target hostname(fqdn): e27a171at-app.dlpxdc.co
Enter dbtier target hostname(fqdn): e27a171at-db.dlpxdc.co
Do you want to use privilege elevation? [y/n]: y
Enter the 'high' privilege user used for privilege elevation in sudoers entries:
applvis
Enter the 'low' privilege user used for privilege elevation in sudoers entries:
delphix_os
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++
Current CPU utilization: 2 percent
CPU utilization is below threshold i.e. 80.
          ###Printing server results###
Server_Configuration                        | Current_Value                 |
Required_Value
------------------------------------------------------------------------------------
-----------------
Architecture                                | x86_64 Linux                  |
Kernel version                              | 4.18.0-240.el8.x86_64         |
Total CPU cores on server                   | 8 cores                       | Minimum 2
 cores
Total memory on server                      | 32149 MB                      | Minimum
8GB
Free memory on server                       | 31805 MB                      | Minimum
8GB
kernel.shmmax value set on server           | 31814338025095168 bytes       |
30340517068 bytes
kernel.shmall value set on server           | 8291456 Pages                 | 8230392
 pages
------------------------------------------------------------------------------------
-----------------
```

```
For EBS sizing Refer- https://docs.oracle.com/cd/E26401_01/doc.122/e22950/
T422699i4773.htm#ebs_sizing
-------------------------------------------------------------------------------------------
-----------------
Memory utilization OK.
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++
Checking for expect
found expect utility at  - /usr/bin/expect
Checking for ss
found ss utility at - /usr/sbin/ss
Checking for netstat
found netstat at - /usr/bin/netstat
Checking for make
found make at - /usr/bin/make
Checking for ld
found ld at - /usr/bin/ld
Checking for gcc
found gcc at - /usr/bin/gcc
Checking for g++
found g++ at - /usr/bin/g++
Checking for ar
found ar at - /usr/bin/ar
Utility check completed successfully
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++
Available disk space in toolkit directory: 511817 MB
There is enough disk space in toolkit directory.
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++
Ping to e27a171at-app.dlpxdc.co succeeded. IP address: 10.110.237.75
Entry for e27a171at-app.dlpxdc.co with IP address 10.110.237.75 exists in /etc/hosts.
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++
Ping to e27a171at-db.dlpxdc.co succeeded. IP address: 10.110.241.161
Entry for e27a171at-db.dlpxdc.co with IP address 10.110.241.161 exists in /etc/hosts.
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++
Checking sudoers file entries. Using : appstier_tgt_sudo.sh
Sudoers entry check completed successfully!
Following entries not found in /etc/sudoers:
 /bin/su - applvis -c rm -f *updateDSpwd.py,
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++
Central Inventory location is /u01/oracle/VIS12210/oraInventory
Make sure Oracle installation owners you intend to use have the Oracle Inventory
group oinstall as their primary group
Printing members of oinstall group, group id 9999 using /etc/group file
applvis
delphix_os
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++
           ### Beginning ulimit check ###
```

```
### Oracle Software Installation Owner Resource Limit Recommended Ranges for Standard
Installations ###
-----------------------------------------------------------------------------------------
----------------------------------------
Resource Shell Limit                            | Resource       | Soft Limit       |
Hard Limit
-----------------------------------------------------------------------------------------
----------------------------------------
Open file descriptors                           | nofile         | at least 1024    |
at least 65536
Number of processes available to a single user | nproc          | at least 2047    |
at least 16384
Size of the stack segment of the process       | stack          | at least 10240 KB|
at least 10240 KB, and at most 32768 KB
Maximum locked memory limit                     | memlock        | unlimited        |
unlimited
-----------------------------------------------------------------------------------------
----------------------------------------
For EBS sizing, refer to:
Checking Resource Limits for Oracle Software Installation Users
https://docs.oracle.com/en/database/oracle/oracle-database/21/ladbi/checking-
resource-limits-for-oracle-software-installation-users.html#GUID-293874BD-8069-470F-
BEBF-
A77C06618D5A:~:text=Table%205%2D1%20Oracle%20Software%20Installation%20Owner%20Resour
ce%20Limit%20Recommended%20Ranges%20for%20Standard%20Installations
-----------------------------------------------------------------------------------------
----------------------------------------
Current system settings:
-----------------------------------------------------------------------------------------
----------------------------------------
Resource Shell Limit                            | Resource       | Soft Limit       |
Hard Limit
-----------------------------------------------------------------------------------------
----------------------------------------
Open file descriptors                           | nofile         | 4096             |
65536
Number of processes available to a single user | nproc          | 2047             |
16384
Size of the stack segment of the process       | stack          | 10240            |
16384
-----------------------------------------------------------------------------------------
----------------------------------------
Other ulimit settings                                                |
---------------------------------------------------------------------
Resource Shell                                  | Limit          |
---------------------------------------------------------------------
max locked memory                               | 64 in KB       |
max memory size                                 | unlimited in KB|
virtual memory                                  | unlimited in KB|
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++
-----------------------------------------------------------------------------------------
----------------------------------------
```

```
Health checker warnings!
-------------------------------------------------------------------------------------------
-----------------------------------------
Following entries not found in /etc/sudoers:
 /bin/su - applvis -c rm -f *updateDSpwd.py,
-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
-----------------------------------------------------------------
```

EBS plugin installation

The EBS Plugin is provided as a zip file on the Delphix download site. Follow the steps below to download the plugin:

1. In the web browser, go to the Delphix download site
2. Login to the download site using email and password credentials.
3. Navigate to the **Delphix Product Releases** folder and select the required Delphix Engine version.
4. Go to `Plugins> EBS> EBS12.2> EBS_Toolkit_<version_number>_for_EBS12.2.zip folder`, then download the zip file, unzip it, and copy both the .json files at your preferred location.

   Note:
   - For a particular version of the EBS plugin, there are two .json files that will be downloaded and the same would be uploaded while installing or upgrading the plugin. Also, you need to upload these .json files in sequential order. For example, for EBS12.2 2.0.1 plugin version you need to upload ebs122-db2.0.1.json first and then ebs122-app2.0.1.json.
   - The plugin versions for AppsTier and DB Tech Stack need to be the same, at installation and through the upgrade. Installation and usage of different versions of these plugins are not supported and may encounter operational issues.
5. Existing users who are planning to upgrade from Plugin version 3.0.x to 4.0.0 need to consider below points:
   a. Follow the upgrade path available at EBS Release notes
   b. For low-privilege users, old sudoers entries need to be updated with the newly provided sudoers entries in both source and target environments prior to upgrade.
   c. Update the Database VDB Hooks with the newly provided hooks.
   d. For plugin upgrade and installation steps, refer to Delphix Engine plugin management. Also, refer to EBS support matrix for upgrade paths.
6. If you are planning to deploy the EBS 12.2 4.0.0 for the first time in the Delphix Engine, you just need to upload the new plugin and proceed with the Delphix Engine operations.

Upgrade EBS objects to 6.0

This section focuses on the steps to be taken post-upgrade to Delphix 6.0 release for both EBS 12.1 and EBS 12.2 for all dSources and VDBs present on the Delphix Engine.

Uploading a plugin

After upgrading to the required DE version 6.0.3 or later, if you need to upgrade or install a new plugin, follow the steps mentioned in EBS Plugin Installation.

Post upgrading to 6.0.3.0 or later

EBS 12.2 plugin

This section focuses on the steps taken after upgrading to the Delphix 6.0.3.0 release or later, for all the users who intend to use plugin version 2.0.0 or 2.0.1 for EBS 12.2.

Pre-requisites for the existing users:

To successfully upgrade to Delphix Engine 6.0.3.0 or later with EBS 12.2 plugin version 3.0.0, the following conditions must be fulfilled:

- The user must be on or upgrade to the 2.0.0/2.0.1 version before moving to 3.0.0 on the Delphix Engine. Refer Plugin Upgrade Path for EBS 12.2 Customers for all the recommended upgrade paths.
- All datasets on Delphix Engine must be in a disabled state.

Post upgrade steps:

1. Verify the plugin version under **Manage** > **Plugins** section. It should show as 3.0.0.
2. Refresh all the environments on the Delphix Engine.
3. Users who intend to use low-privileged user, (for example, "delphix_os") should follow the environment configuration mentioned here and steps 4 and 5 below.
4. All the EBS dSources and VDB/vFiles should edit the Environment User under the Configuration → Source to "delphix_os".
5. Users who intend to use low-privileged user, must update the "Privileged OS User" field on the dSource/VDB provision wizard to high privileged user (user who should own the file system).
6. Enable all the EBS dSources - dbTechStack, Database, and appsTier.
7. Take new snapshots for all the enabled EBS dSources.
8. Enable all the EBS VDB/vFiles as per the following sequence: (1) dbTechStack (2) Database (3) appsTier
9. Under **Custom** > **Configuration** tab for dbTechStack vFiles, update the value for the "Target APPS Password" field to the value similar to the "Target APPS Password" field as provided under appsTier vFiles.
10. Modify the database hooks. Follow the steps under Provisioning a Virtual EBS R12.2 instance to configure the hooks.
11. Take new snapshots for all the enabled EBS VDB's.
12. To refresh the VDB's, follow the steps and instructions for Refreshing a virtual EBS instance.

Upgrade objects to 5.3

Post upgrade to 5.3.2.0

This section focuses on steps to be taken post-upgrade to the Delphix 5.3.2.0 release for all customers who intend to use WebLogic password change feature for EBS 12.2 release.

1. Disable all the virtual datasets created on Delphix Engine in the following order:
    a. appsTier vFiles
    b. VDB utilized by your EBS instance
    c. dbTechStack vFiles
2. Disable all the EBS dSource created on Delphix Engine once you have disabled all three EBS virtual datasets in the above step.
3. Upgrade Delphix Engine to 5.3.2.0 release and upload EBS 1.4.0 plugin release.
    a. Go to Manage → Plugins
        i. Upload a plugin file to upgrade existing toolkit by clicking plus (+) symbol.
4. Refresh all host environments added on the Delphix Engine
    a. Go to Manage → Environments
        i. From the symbol (...) located in the upper left-hand corner, select Refresh All.
5. Change the Weblogic admin password on EBS appsTier vFile.
    - Select the appsTier vFiles hosting your virtual EBS database.
    - Go to Configurations → Custom
    - Update "**Target Weblogic AdminServer Password**" field with the password you would like to set for EBS appsTier.
6. Enable all the EBS dSource created on Delphix Engine.
7. Enable all the virtual datasets created on Delphix Engine in the following order:
    a. dbTechStack vFiles
    b. VDB utilized by your EBS instance
    c. appsTier vFiles
8. Verify the changed WebLogic password by logging into the WebLogic console.

Post upgrade to 5.3.1.0

This section focuses on steps to be taken post-upgrade to the Delphix 5.3.1.0 release for all customers who intend to use sudo privileges for EBS 12.1 release.

1. Create a low-privileged OS account (delphix_os) in source and target environments.
2. Steps and configuration required for a low-privileged account to perform delphix privileged operations are outlined in Requirements for Unix environments.
3. Push the privilege elevation script "DLPX_DB_EXEC" to both and target environments as per the steps outlined in Privilege script.
4. For all 12.1 EBS appsTier dSource :
    a. Go to Configuration >Source
        i. Update Environment User value to low privileged account (delphix_os).
    b. Go to Configuration>Custom
        i. Update Privileged OS Account with the value for high privileged user (applmgr).
5. For all 12.1 EBS appsTier vFile:
    a. Go to Configuration>Source
        i. Update Environment User value to low privileged account (delphix_os).
    b. Go to Configuration>Custom
        i. Update Privileged OS Account with the value for high privileged user (applmgr).
6. For both source and target environment:

a. Go to Manage>Environments
    i. Add low privileged OS user (delphix_os) in Environments Users section by clicking plus (+) symbol.
    ii. Mark low-privileged OS user (delphix_os) as the primary user.
    iii. Remove the high privileged OS user (applmgr) from the Environments Users section.

Upgrade EBS objects to 5.2

This section focuses on steps to be taken post-upgrade to the Delphix 5.2 release for all dSources and VDBs present on the Delphix Engine.

1. **For all 12.1 EBS apps Tier Dsource** :
   Go to Configuration → Custom
   - Update context_name with the Context value of the application tier
   - Update apps password with the value of the dsource.
   - Take a new snapshot after updating the above parameters
2. **For all 12.1 and 12.2 EBS Database VDB**:
   - Update the pre snapshot hook as follows

```
# NOTE: Ensure the below environment variables will be set up correctly by
the shell. If not, hardcode or generate the values below.
# CONTEXT_NAME=${ORACLE_SID}_$(hostname -s)
# APPS_PASSWD=<source apps passwd>
# TARGET_APPS_PASSWD=<new apps password>

timeout=3600
waittime=0
. ${ORACLE_HOME}/${CONTEXT_NAME}.env

testAppsPassword() {
    local passwordInQuestion=$1
    ERROR=`sqlplus "apps/${passwordInQuestion}" <<< "exit;"`

    grep ORA-01017 <<< ${ERROR} >/dev/null && return 1
    return 0

}

testAppsPassword ${SOURCE_APPS_PASSWD}
testResult=$?

if [[ ${testResult} == 0 ]]; then
    APPS_PASSWD=${SOURCE_APPS_PASSWD}
else
    APPS_PASSWD=${TARGET_APPS_PASSWD}
fi

while [[ -f "${ORACLE_HOME}/.delphix_adpreclone.lck" || "$(ps -Ao args |
grep "^${ORACLE_HOME}" | grep -v grep |  grep "adpreclone")" ]] ; do
if [[ $waittime -gt $timeout  ]]; then
        echo "Another adpreclone process is still running from last 60
mins.Delphix cannot proceed until it is complete. Exiting Now."
        exit 1
    fi
    echo  " Another adpreclone process is running. waiting for the process
to complete before starting adpreclone of the database...."
    (( timeout += 10 ))
```

```
    sleep 10
done
echo "No other adpreclone process is running on database, proceeding ...."
. ${ORACLE_HOME}/${CONTEXT_NAME}.env
${ORACLE_HOME}/perl/bin/perl ${ORACLE_HOME}/appsutil/scripts/$
{CONTEXT_NAME}/adpreclone.pl database <<-EOF
${APPS_PASSWD}
EOF
```

3. **For all 12.1 EBS apps Tier VDB** :
   Go to Configuration → Custom
   - Update system password with the system password corresponding to the database.
   - Take a new snapshot after updating the above parameters

Oracle EBS R12.2

This section contains the following topics:

Source EBS R12.2 instance requirements

The Delphix Engine supports linking a variety of versions and configurations of Oracle E-Business Suite R12.2. Below, detailed compatibility notes are outlined.
Note that minor releases of EBS are not certified for Delphix Engine compatibility individually; major release support implies support for any minor release.

To review the requirements for provisioning a virtual EBS R12.2 instance from the Delphix Engine, see Virtual EBS R12.2 instance requirements.

Operating systems

- Linux – SLES 9, 10, 11; RHEL 5, 6, 7, OL 5, 6
- Solaris - SPARC 9, 10, and 11

AIX and HP-UX not supported
The Delphix Engine does not support linking EBS instances running on AIX or HP-UX.

dbTier requirements

Supported topologies

- Oracle single-instance (SI) dbTechStack and Database
  - Oracle11gR2 v11.2.0.4
  - Oracle12cR1 v12.1.0.2
  - Oracle19c MT v19.3.0.0
- Oracle RAC dbTechStack and Database
  - Oracle11gR2 v11.2.0.4
  - Oracle12cR1 v12.1.0.2

Caveats

The Delphix Engine does not provide support for linking an EBS R12.2 instance utilizing custom context variables maintained in the EBS context file.

Virtual EBS R12.2 instance requirements

The Delphix Engine supports provisioning a variety of versions and configurations of Oracle E-Business Suite R12.2. Below, detailed compatibility notes are outlined.

Note that minor releases of EBS are not certified for Delphix Engine compatibility individually: major release support implies support for any minor release.

To review the requirements for linking an EBS R12.2 instance to the Delphix Engine, see Source EBS R12.2 instance requirements.

Operating systems

- Linux - SLES 9, 10, 11; RHEL 5, 6, 7, OL 5, 6
- Solaris - SPARC 9, 10, and 11

AIX and HP-UX not supported
The Delphix Engine does not support provisioning EBS instances to AIX or HP-UX.

dbTier requirements

Supported topologies

- Oracle SI dbTechStack and Database
  - Oracle11gR2 v11.2.0.4
  - Oracle12cR1 v12.1.0.2
  - Oracle19c MT v19.3.0.0

Oracle RAC not supported
The Delphix Engine does not support provisioning Oracle RAC dbTechStack for use with Oracle E-Business Suite. However, you may provision an Oracle SI dbTier from a linked EBS RAC dbTier instance. During the provisioning process, the Delphix Engine will relink the dbTechStack for use with an Oracle SI database and scale down the Oracle RAC database to an Oracle SI database.

appsTier requirements

Supported topologies

- Single-node appsTier
- Multi-node appsTier with a shared APPL_TOP

Non-shared APPL_TOP not supported
The Delphix Engine does not provide support for provisioning a multi-node appsTier where the APPL_TOP is not shared between nodes.

appsTier topology Is configurable
The appsTier topology of the virtual EBS instance does NOT need to match the appsTier topology of the source EBS instance. The Delphix Engine will automate scale up or scale down logic as required during provisioning.

Caveats

The Delphix Engine will only register the default WLS-managed server for each server type during multi-node appsTier provisioning. Additional WLS managed servers need to be registered manually or as a **Configure clone** hook.

Remote logging

Remote Logging for EBS 12.2 Plugin adds the following options to provide enhanced logging:

- set debug log levels
- set active log file size
- set log retention levels

You can modify the parameters defined in the `ebs_plugin.conf` file located in remote target host at `<Toolkit directory>/<Delphix_COMMON>/plugin/ebs122-vsdk-db_9f7f5a63-1da7-495c-80df-e0fa61941342/ebs_plugin.conf` for DbTechStack and `<Toolkit directory>/<Delphix_COMMON>/plugin/ebs122-vsdk-app_403e7034-0028-4200-ad0b-15031f0f8871/ebs_plugin.conf` for appsTier to configure the logging mechanism.

Setting log levels (level)

You can set log levels for plugin-generated logs by setting the parameter level in the `ebs_plugin.conf` file. Higher logging levels will help to expedite debugging issues. There are two levels of logging: info and debug.

Log level description

- level=INFO: This level will print only informational logs. This is the default log level.
- level=DEBUG: This level will print informational logs and debug statements.

Setting maximum size of active log file (logFileSize)

You can set this parameter inside the `ebs_plugin.conf` file to set the maximum size of the active log file in MB. Once this limit is reached, the plugin rotates the log as per the retention property defined below. The minimum value of this parameter is 1 MB and the maximum value is 10 MB. This parameter only takes in a positive integer value. The default value is 1 MB.

Setting log retention levels (retention)

You can set a retention level for diag logs (diag.log) using parameter retention in the `ebs_plugin.conf` file. As per this parameter, the log files are moved (archived), renamed, or deleted once they reach the value set in the `logFileSize` parameter. New incoming log data is directed into a new fresh file (at the same location).

By default, this value will be set to a minimum value of 2. The user can change this value and set its value within the range 2 and 50. For example; if retention is set to 4, the plugin will have the following log files: diag.log, diag.log.1, diag.log.2, diag.log.3, diag.log.4.

- File diag.log is the active log file.
- File diag.log.1 is the most recent archive log file
- File diag.log.4 is the oldest one

ebs_plugin.conf file

```
#
# This flag will set the debug level of plugin logs on the remote server. There are
two levels:
# Info
# Debug
```

```
# The above are the only valid values that can be assigned. If any other value is
assigned, the plugin will set the
# default level as Info. the string is case insensitive so info, Info and INFO are
acceptable.
#

level=INFO

#
# This parameter will set the maximum size of the active log file in MB. Once this
 limit is reached, the plugin will
# rotate the log as per the retention property defined below. The minimum value of
this parameter is 1 MB and maximum
# value is 10 MB. This parameter only takes in a positive integer value. The default
 value is 1 MB
# logFileSize=<positive integer>
#

logFileSize=1

#
# Whenever the size of diag.log file exceeds the value provided by logFileSize
parameter then the plugin will
# rename the active log file to diag.log.<number> and a new log file with name
diag.log will be
# generated. For example, if LogRetention is set to 4, the plugin will have the
following log files: diag.log,
# diag.log.1, diag.log.2, diag.log.3, diag.log.4.
# File diag.log.4 will be the oldest one.
# File diag.log.1 will be the most recent archive log file.
# File diag.log will be the active log file.
# The minimum and default value for retention flag is 2.
# retention=<positive integer>
#

retention=2
```

Preparing a source EBS R12.2 instance for linking

This topic outlines the prerequisites for linking an EBS R12.2 instance to the Delphix Engine.

Ensure your EBS R12.2 instance is supported

See Source EBS R12.2 instance requirements to ensure you can link your EBS R12.2 instance to the Delphix Engine.

Ensure your EBS 12.2 environments comply with Oracle's documentation

Your environments must comply with Oracle's requirements for installing EBS. These requirements are outlined on **Oracle E-Business Suite Release 12 Installation Guidelines (Doc ID 405565.1)** found at https://support.oracle.com

Prepare the dbTier for linking

Delphix engine's Unix environment requirements

The dbTier must meet the source requirements outlined in Requirements for Unix environments. These requirements are generic to all source Unix environments added to the Delphix Engine.

`oracle` user

The Delphix Engine must have access to an `oracle` user on the dbTier.

- This user should be a member of both the EBS `dba` and `oinstall` groups.
- The user should own the stage directory for DBTechStack and database files that will be cloned.
- Ensure that the "expect" utility exists on the remote host.

dbTechStack binary permissions

Verify that the `oracle` user described above has read permissions at the group level for:

- `$ORACLE_HOME/bin/nmb`
- `$ORACLE_HOME/bin/nmhs`
- `$ORACLE_HOME/bin/nmo`

The Delphix Engine's Oracle database requirements

The dbTier must meet the source requirements outlined in Oracle support and requirements. These requirements are generic to all Unix environments containing an Oracle database to be linked.

Prepare the appsTier for linking

Delphix Engine's Unix environment requirements

The appsTier must meet the source requirements outlined in Requirements for unix environments. These requirements are generic to all source Unix environments added to the Delphix Engine.

`applmgr` user

The Delphix Engine must have access to an `applmgr` user on the appsTier.

- This user should be a member of the EBS `oinstall` group.
- The user should own the stage directory for Appstier files that will be cloned.

delphix_os OS user account

In order to separate authentication and perform privileged operations with a non-privileged OS account, first create an OS user account (i.e. "delphix_os") on the EBS DB Tier and appsTier node to be used as a source.
This user is easily created by the createDelphixOSUser.sh (located below on this page) script.

- The primary OS group of the Delphix Engine software owner account's (i.e. delphix_os) should be the same as the EBS Database/AppsTier software owner account (i.e. oravis or applmgr).
- Primary group = Oracle Install Group (typically oinstall), secondary group = OSDB Group (typically dba). There are lots of cases where the OS group named dba fills this role, so be sure to check the group membership of the EBS Database and AppsTier software owner account.
- Please note, the non-privileged OS account must have the **same** group as assigned to EBS Database or AppsTier privileged account (like oravis or applmgr).

Host requirements:

To accomplish necessary tasks on the EBS Database and appsTier source hosts, the Delphix OS user account (henceforth referred to as "delphix_os") requires privilege elevation specifications.

Here is an example specification for the "sudo" privilege elevation utility, using the "visudo" to edit the "sudoers" configuration file. This specification makes the following assumptions:

- OS = Linux
  - OS account owning Oracle EBS Database Tier is named **oravis**
  - OS account owning Oracle EBS appsTier is named **applvis**
- OS = Solaris
  - OS account owning Oracle EBS Database Tier is named **oravis**
  - OS account owning Oracle EBS appsTier is named **oravis**
- EBS Database and appsTier system base directory is **/u01/oracle/VIS.** This can be prefixed before all the below-mentioned commands to ensure the same are being executed from relevant paths.

The following sudoers entry is only for template purpose. Modify the path in the below sudoers entry with the appropriate binary paths of your environment.

Sudoers entry for Linux -

Entries required for linking via low privileged user (delphix_os):	
EBS DB Tier	`Defaults:delphix_os !requiretty` `delphix_os ALL=NOPASSWD: /bin/su - oravis -c echo *, /bin/su - oravis -c rm -f */dlpx_force_autoflush*, /bin/su - oravis -c cp -f */dlpx_force_autoflush*; chmod 755 */dlpx_force_autoflush*, /bin/su - oravis -c rm -rf */appsutil/clone/dbts*, /bin/su - oravis -c export PERL5LIB*perl -mdlpx_force_autoflush */adpreclone.pl dbTier*, /bin/su - oravis -c */rsync*, /bin/su - oravis -c test*mkdir*, /bin/su - oravis -c test*touch*chmod 750*cat*, /bin/su - oravis -c */EBS_kill/kill_script.sh*, /bin/su - oravis -c rm -f */test_status.tmp*, /bin/ps, /bin/su - oravis -c cp *dlpx_force_autoflush.pm*, /bin/su - oravis -c chmod 755 */dlpx_force_autoflush*`

Entries required for linking via low privileged user (delphix_os):	
EBS appsTier	`Defaults:delphix_os !requiretty` `delphix_os ALL=NOPASSWD: /bin/su - applvis -c echo *, /bin/su - applvis -c rm *.dlpx_run_edition*, /bin/su - applvis -c rm -f */dlpx_force_autoflush*, /bin/su - applvis -c cd *echo *dlpx_force_autoflush*, /bin/su - applvis -c export PATH* export PERL5LIB* cd *perl -mdlpx_force_autoflush ./adpreclone.pl appsTier*, /bin/su - applvis -c */rsync*, /bin/su - applvis -c test*mkdir*, /bin/su - applvis -c test*touch*chmod 750*cat*, /bin/su - applvis -c */EBS_kill/kill_script.sh*, /bin/su - applvis -c rm -f */test_status.tmp*, /bin/su - applvis -c *.env* sqlplus -s apps*, /bin/su applvis -c *.env* sqlplus -s apps*, /bin/su - applvis -c */*.env*echo*adadminsrvctl.sh status -nopromptmsg*, /bin/ps, /bin/su - applvis -c cp *dlpx_force_autoflush.pm*, /bin/su - applvis -c export PERL5LIB* cd *perl -mdlpx_force_autoflush */adpreclone.pl appsTier*`

Sudoers entry for Solaris -

Entries required for linking via low privileged user (delphix_os):	
EBS DB Tier	`Defaults:delphix_os !requiretty delphix_os ALL=NOPASSWD: /usr/bin/su - oravis -c echo *, /usr/bin/su - oravis -c rm -f */dlpx_force_autoflush*, /usr/bin/su - oravis -c cp -f */dlpx_force_autoflush*; chmod 755 */dlpx_force_autoflush*, /usr/bin/su - oravis -c rm -rf */appsutil/clone/dbts*, /usr/bin/su - oravis -c export PERL5LIB*perl -mdlpx_force_autoflush */adpreclone.pl dbTier*, /usr/bin/su - oravis -c */rsync*, /usr/bin/su - oravis -c test*mkdir*, /usr/bin/su - oravis -c test*touch*chmod 750*cat*, /usr/bin/su - oravis -c */EBS_kill/kill_script.sh*, /usr/bin/su - oravis -c rm -f */test_status.tmp*, /bin/ps, /usr/bin/su - oravis -c cp *dlpx_force_autoflush.pm*, /usr/bin/su - oravis -c chmod 755 */dlpx_force_autoflush*`

Entries required for linking via low privileged user (delphix_os):	
EBS appsTier	`Defaults:delphix_os !requiretty delphix_os ALL=NOPASSWD: /usr/bin/su - oravis -c echo *, /usr/bin/su - oravis -c rm *.dlpx_run_edition*, /usr/bin/su - oravis -c rm -f */dlpx_force_autoflush*, /usr/bin/su - oravis -c cd *echo *dlpx_force_autoflush*, /usr/bin/su - oravis -c export PATH* export PERL5LIB* cd *perl -mdlpx_force_autoflush ./adpreclone.pl appsTier*, /usr/bin/su - oravis -c */rsync*, /usr/bin/su - oravis -c test*mkdir*, /usr/bin/su - oravis -c test*touch*chmod 750*cat*, /usr/bin/su - oravis -c */EBS_kill/kill_script.sh*, /usr/bin/su - oravis -c rm -f */test_status.tmp*, /usr/bin/su - oravis -c *.env* sqlplus -s apps*, /usr/bin/su oravis -c *.env* sqlplus -s apps*, /usr/bin/su - oravis -c */*.env*echo*adadminsrvctl.sh status -nopromptmsg*, /bin/ps, /usr/bin/su - oravis -c cp *dlpx_force_autoflush.pm*, /usr/bin/su - oravis -c export PERL5LIB* cd *perl -mdlpx_force_autoflush */adpreclone.pl appsTier*`

Requirement for privilege elevation script: DLPX_DB_EXEC

In order to elevate privileges from a non-privileged OS account (like delphix_os) to a privileged OS account (like applmgr), we need to push a privilege elevation script (***dlpx_db_exec***) up into the Delphix virtualization engine to become part of the Delphix common plugin.

Why we need DLPX_DB_EXEC

Some customers want to use low privilege users to perform delphix operations like linking and provisioning. It means their low privilege user should have sudo permissions to execute EBS application and DB related commands. The privilege elevation profile script ***dlpx_db_exec*** allows them to execute commands that require superuser privileges on customer source and target machines.

The privilege elevation script ***dlpx_db_exec*** can be created or pushed to Delphix Engine using Web API calls, CURL or dxtoolkit.

For steps on creating a Privilege Elevation Profile please refer to CLI Cookbook: How to create or edit a privilege elevation profiles and profile scripts

Content of DLPX_DB_EXEC Privilege Elevation Profile:

```
#!/bin/sh
#
# Copyright (c) 2018 by Delphix. All rights reserved.
#
# This script allows customization of command execution with an alternate user
# account.
# Arg $1 contains "-u<optional user account>" for the desired user under
# which database commands will be executed.
# By default this argument is ignored and the script is executed as the default
# account.
#
if [[ $1 != -u* ]]; then
   echo "Incorrect command line parameters, -u<optional user account> is required as
the first parameter"
   exit 1
fi
user_id=`echo $1 | sed -e "s/^-u//"`
shift 1
if [[ $user_id != "delphix_os" ]]; then
    command=$(printf "%s " "$@")
    sudo su - $user_id -c "$command"
else
    $@
fi
```

Below is an example of how we can push privilege elevation script “dlpx_db_exec” on a customer Delphix Engine:

- Create a session to Delphix Engine as Delphix os user:

```
curl -i -c cookies.txt -X POST -H "Content-Type:application/json" http://
<Delphix-Engine>/resources/json/delphix/session -d '{
    "version":{
        "minor":11,
        "major":1,
        "micro": 5,
        "type":"APIVersion"
        },
        "type":"APISession"}'
```

Note: The API Version needs to be identified as per the Delphix Engine installed at the customer site.

- Login to Delphix Engine as Delphix OS User:

```
curl -i -c cookies.txt -b cookies.txt -X POST -H "Content-Type:application/
json" http://<Delphix-Engine>/resources/json/delphix/login -d '{
    "password":"delphix",
    "type":"LoginRequest",
    "target":"DOMAIN",
    "username":"delphix_admin"
    }'
```

- Push DLPX_DB_EXEC contents to Delphix Engine:

```
curl -i -b cookies.txt -X POST -H "Content-Type:application/json" http://
<Delphix-Engine>/resources/json/delphix/host/privilegeElevation/profileScript/
HOST_PRIVILEGE_ELEVATION_PROFILE_SCRIPT-7 -d '{
    "type": "HostPrivilegeElevationProfileScript",
    "contents": "#\n# Copyright (c) 2018 by Delphix. All rights reserved.
\n#\n\n#\n# This script allows customization of command execution with an
alternate user\n# account.\nif [[ $1 != -u* ]]; then\n    echo \"Incorrect
command line parameters, -u<optional user account> is required as the first
parameter\"\n exit 1\nfi\nuser_id=`echo $1 | sed -e \"s\/^-u\/\/\"`\n\nshift
1\nif [[ $user_id != \"delphix_os\" ]]; then\ncommand=$(printf \"%s \" \"$@\")
\nsudo su - $user_id -c \"$command\"\nelse\n$@\nfi\n"
    }'
```

Script

createDelphixOSUser.sh

Linking a source EBS R12.2 instance

This topic describes the process of linking an EBS R12.2 instance and creating the necessary dSources.

Prerequisite

Prepare your source EBS R12.2 instance for linking by following the outline in Preparing a Source EBS R12.2 instance for linking

Procedure

Link the Oracle database

1. Link the Oracle database used by EBS, as outlined in Linking an Oracle data source
2. Link the Oracle Pluggable database for 19c MT, as outlined in Linking an Oracle pluggable database

Link the EBS dbTechStack

Prerequisite

Upload the EBS 12.2 dbTechStack plugin and refresh the environments during the preparation of the source environment. For more information, see Upgrade EBS objects to 6.0

For successful linking of DBTECHSTACK from the standby site, follow any one of the procedures below:

- Place the physical standby database in READ ONLY/ READ ONLY WITH APPLY mode and run linking of source DBTECHSTACK. Even during subsequent snapsync operations, you have to make sure that the standby database container & its PDB is in READ ONLY / READ ONLY WITH APPLY mode.
  Note: Make sure that the PDB is in READ ONLY mode, also the two default EBS PDB services `ebs_` and should be in READY STATE. Linking will fail if the PDB is not in READ ONLY mode.
- Perform the linking of dbTechStack(ORACLE_HOME) from the primary source site which is in READ_WRITE mode.

- Before DBTechStack linking, to ensure that the hook operation succeeds during database provision, the ETCC patch level should be the latest. In the configure clone 1 hook adcfgclone dbconfig will perform the ETCC patch level check and will fail if it's not the latest.
- While linking using the EBS plugin, the plugin runs `adpreclone.pl` dbTier in the background, which asks for apps database user password and connects with standby DB. If database is in MOUNT state then linking fails, as apps schema will not be able to login into standby DB.

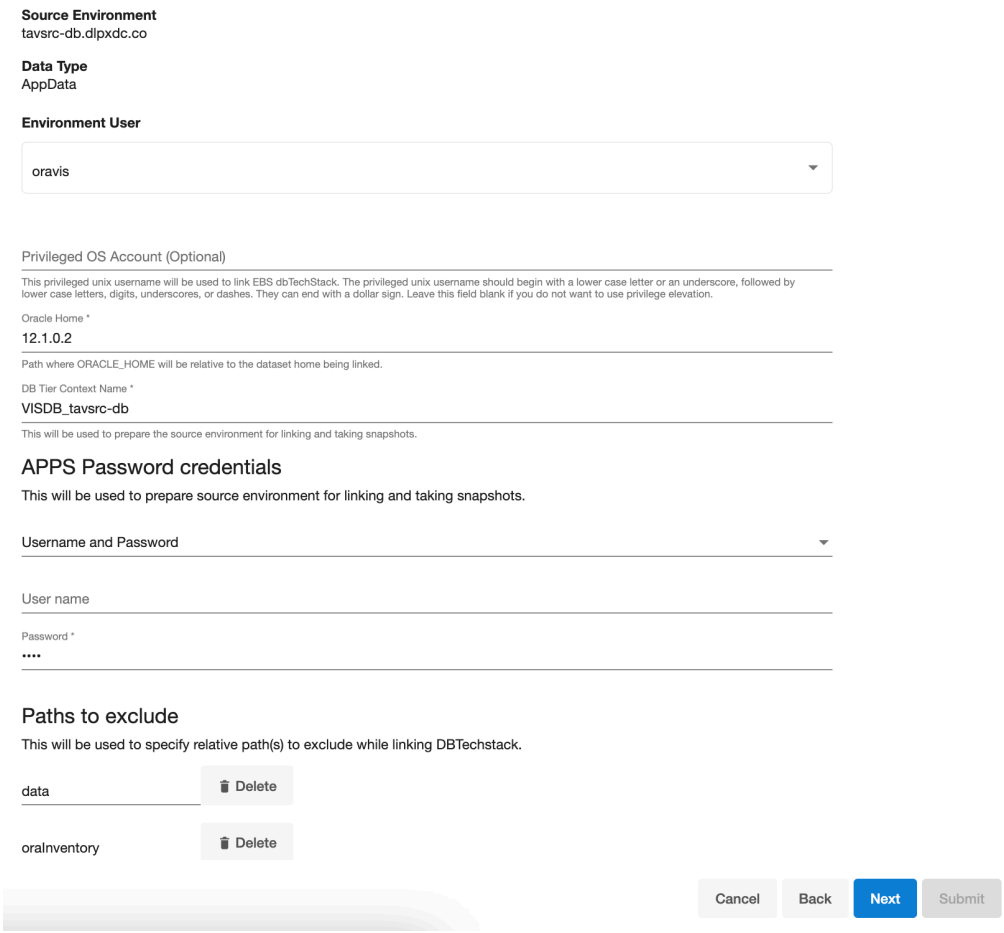
1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Select the **source dbTier environment** containing the source dbTechStack.
   Note: If you are linking from a RAC dbTier, select the environment for a single running node of the RAC cluster.
5. Click the **Environment Details** tab.
6. If the `oracle` environment user described in Preparing a source EBS R12.2 instance for linking is not already added to the Delphix Engine, add the user.
7. Click the **Databases** tab.
8. Click the **Plus** icon within **E-Business Suite R12.2 dbTechStack** section under installations.
9. **Add Database** dialog box pops to add source config for dbTechStack.

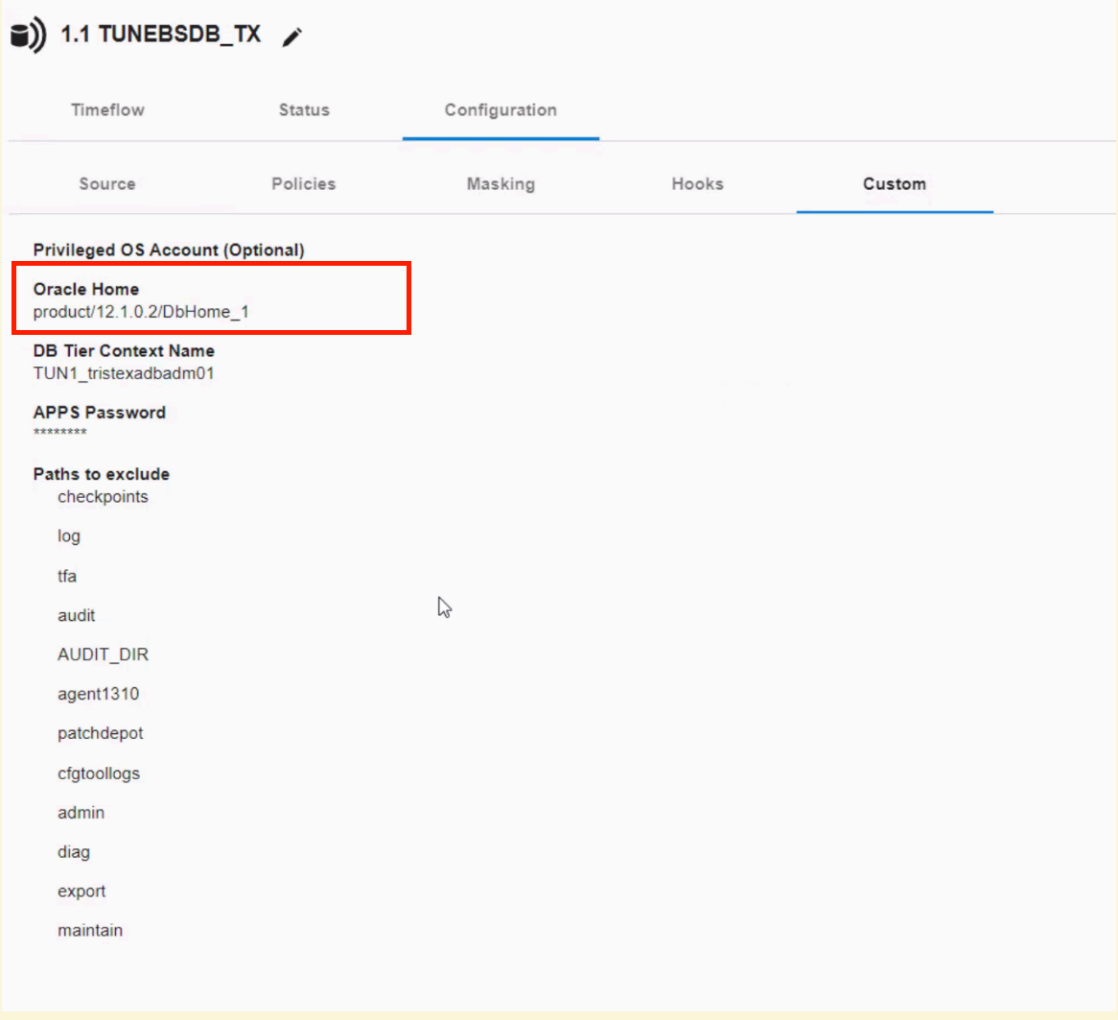
10. Provide a user-defined name to the `Database Name` field and Oracle base install directory in the `Path` field.
11. Click on the **Add** button to successfully add the source config.
12. Re-check the **E-Business Suite R12.2 dbTechStack** section under installations for the `Add dSource` option mapped to the source config that is added in the above step.
13. Click **Manage > Datasets.**
14. Click the **Plus** icon next to Datasets and select **Add dSource**.
15. In the **Add dSource wizard**, select the Linked dSource as the type.
16. Enter the **EBS-specific parameters** for your dbTechStack.
17. These parameter values will be used when `adpreclone.pl` is run. Ensure that the **DB Tier Context Name uses** the short hostname.
18. Select an Environment user.
19. **Privileged OS Account (Optional)** field should contain a high privileged user when the low privileged user is being used for linking.
20. Provide source oracle home directory name relative to oracle base installation directory in Oracle Home input field.
21. In the **DB Tier** Context Name field, provide the source context name; To get the DB Tier context name on source side:
    - For Oracle 12.1.0.2 version,

```
cd $ORACLE_HOME
source SID_hostname.env file
echo $CONTEXT_NAME
```

Below is an example, Consider VISDB as the database SID and tavsrc-db as the short hostname.

```
[oravis@tavsrc-db VIS]$ source /u01/oracle/VIS/12.1.0.2/VISDB_tavsrc-
db.env
[oravis@tavsrc-db VIS]$ echo $CONTEXT_NAME
VISDB_tavsrc-db
```


Source Environment
tavsrc-db.dlpxdc.co
Data Type
AppData
Environment User
oravis
Privileged OS Account (Optional)
This privileged unix username will be used to link EBS dbTechStack. The privileged unix username should begin with a lower case letter or an underscore, followed by lower case letters, digits, underscores, or dashes. They can end with a dollar sign. Leave this field blank if you do not want to use privilege elevation.
Oracle Home *
12.1.0.2
Path where ORACLE_HOME will be relative to the dataset home being linked.
DB Tier Context Name *
VISDB_tavsrc-db
This will be used to prepare the source environment for linking and taking snapshots.
APPS Password credentials
This will be used to prepare source environment for linking and taking snapshots.
Username and Password
User name
Password *
Paths to exclude
This will be used to specify relative path(s) to exclude while linking DBTechstack.
data
Delete
oraInventory
Delete
Cancel
Back
Next
Submit



- For multi-tenant database, CONTEXT_NAME will be (<PDB_NAME>_<short hostname>)
- For RAC, CONTEXT_NAME will be <INSTANCE_SID>_<HOSTNAME>
  For example: VIS2_exaukdb02
  Note: When preparing the source Oracle RAC system for cloning, make sure that the linking is performed against the primary database node.

22. Provide apps schema password in the **Apps Password Credentials** section by selecting the **Username** and **Password** from the dropdown, leaving the username field empty.
23. Exclude the EBS database's data files if they are stored underneath the Oracle base install directory.
24. These data files will be linked with the database instead of with the dbTechStack. Add the "relative path" to the data files to the **Paths to Exclude** list. If the audit files (located under $ORACLE_HOME/rdbms/audit) are large in size, the path of these files could also be excluded so as to avoid dbTechStack timeout issues during provisioning.
    Note: Ensure that you add the relative paths and NOT the absolute paths. For example, if you want to exclude the */u01/oracle/VIS/data* path, then add "*data*" only under the **Paths to Exclude** list. Here, */u01/oracle/VIS/data* is the absolute path and "*data*" is the relative path to be added.
25. Click **Next**.
26. Enter a **dSource Name**.
27. Select a **Database Group** for the dSource.
28. Click **Next**. Adding a dSource to a database group enables you to set Delphix Domain user permissions for that dSource's objects, such as snapshots.
29. Select a **SnapSync** policy.

30. Click **Next**.
31. Enter any **custom pre or post sync logic** as Pre-Sync or Post-Sync hook operations. Remember that `adpreclone.pl dbTier` is already run prior to every Snapshot of the dbTechStack. The Pre-Sync hook operations will be run prior to running the `adpreclone.pl` tool.
32. Click **Next**.
33. Review the **dSource Configuration** and **Data Management** information, and then click **Finish**. The Delphix Engine will initiate two jobs to create the dSource, **DB_Link** and **DB_Sync** . You can monitor these jobs by clicking **Active Jobs** in the top menu bar, or by selecting **System** > **Event Viewer**. When the jobs have completed successfully, the **files** icon will change to a **dSource** icon on the **Environments** > **Databases** screen, and the dSource will be added to the list of **Datasets** under its assigned group.

Intelligent AppsTier Linking

Prerequisite

Upload the EBS 12.2(`ebs122app-4.3.0.json`) AppsTier plugin and refresh the environments during the preparation of source environment.

Procedure

1. Login to the **Delphix Management** application using **Delphix Admin** credentials.
2. Click **Manage**.
3. Select **Environments**.
4. Select the **source appsTier environment**. Linking from multi-node appsTier
   Note - If you are linking from a multi-node appsTier, select the environment for the node on which EBS admin WLS services reside.
5. Click the **Environment Details** tab.
6. If the `applmgr` environment user described in Preparing a Source EBS R12.2 Instance for Linking is not already added to the Delphix Engine, add the user.
   Note - There is no actual requirement that the account be named applmgr on source environment. You can use the account that can run adpreclone.pl successfully and ingest RUN filesystem directory into Delphix engine.
7. Click the **Databases** tab.
8. Click the **Plus** icon within **E-Business Suite R12.2 appsTier** section under installations.
9. **Add Database** dialog box pops to add source config for dbTechStack.
10. Provide a user-defined name to the 'Database Name' field and $APPS_BASE directory in the 'Path' field. $APPS_BASE path is where EBSapps.env file is located.
11. Click on the **Add** button to successfully add the source config.
12. Re-check the **E-Business Suite R12.2 appsTier** section under installations for the **Add dSource** option mapped to the source config that is added in the above step.
13. Click **Manage > Datasets.**
14. Click the **Plus** icon next to Datasets and select **Add dSource**.
15. In the **Add dSource wizard**, select the Linked dSource as the type.
16. In the **Add dSource wizard**, select the **appsTier files source** you just created.
17. Enter the **EBS-specific parameters** for your appsTier.
    These parameter values will be used when `adpreclone.pl` is run.
18. Select an Environment User.
19. **Privileged OS Account (Optional)** field should contain a high privileged user when the low privileged user is being used for linking.
20. Provide the apps schema password in the **Apps Password** section by choosing the **Username** and **Password** from the dropdown, leaving the username field empty.

21. Provide the weblogic password in the **Weblogic AdminServer Password** section by choosing the **Username** and **Password** from the dropdown, leaving the username field empty.
22. With 4.3.0 or later We **do not need** to specify the "relative paths" of files to exclude in the **Paths to Exclude** list.
    Note: Delphix Engine will automatically detect the RUN FS.
23. Click **Next**.
24. Enter a **dSource Name**.
25. Select a **Database Group** for the dSource.
26. Click **Next**. Adding a dSource to a database group enables you to set Delphix Domain user permissions for that dSource's objects, such as snapshots.
27. Click **Next**.
28. Select a **SnapSync** policy.
29. Click **Next**.
30. Enter any **custom pre or post-sync logic** as Pre-Sync or Post-Sync hook operations.

    Remember that `adpreclone.pl appsTier` is already run prior to every SnapSync of the appsTier.

    The Pre-Sync hook operations will be run prior to running the `adpreclone.pl` tool.
31. Click **Next**.
32. Review the **dSource Configuration** and **Data Management** information, and then click **Finish**.
    The Delphix Engine will initiate two jobs to create the dSource, **DB_Link,** and **DB_Sync**. You can monitor these jobs by clicking **Active Jobs** in the top menu bar, or by selecting **System** > **Event Viewer**. When the jobs have completed successfully, the **files** icon will change to a **dSource** icon on the **Environments** > **Databases** screen, and the dSource will be added to the list of **Datasets** under its assigned group.

Preparing target EBS R12.2 environments for provisioning

This topic outlines the prerequisites for provisioning a virtual EBS R12.2 instance to target environments.

Ensure your target EBS R12.2 instance is supported

See Virtual EBS R12.2 instance requirements to ensure you can provision your EBS R12.2 instance.

Ensure your EBS 12.2 environments comply with Oracle's documentation

Your environments must comply with Oracle's requirements for installing EBS. These requirements are outlined on **Oracle E-Business Suite Release Notes, Release 12.2 (Doc ID 1320300.1)** found at https://support.oracle.com

> Oracle has released an E-Business Suite Pre-Install RPM (available on ULN and public yum) that includes all required RPMs for both the appsTier and dbTier of an R12.2 installation. Details can be found in **Oracle E-Business Suite Installation and Upgrade Notes Release 12 (12.2) for Linux x86-64 (Doc ID 1330701.1)** found at https://support.oracle.com .

Prepare the dbTier for provisioning

Delphix Engine's Unix environment requirements

The dbTier must meet the target requirements outlined in Requirements for Unix Environments. These requirements are generic to all target Unix environments you add to the Delphix Engine.

`oracle` User

The Delphix Engine must have access to an `oracle` user on the dbTier.

- This user should be a member of both the EBS `dba` and `oinstall` groups.
- This user will be given proper permissions to manage the dbTechStack and database.
- Ensure that the "expect" utility exists on the remote host.

oraInst.loc

An oraInst.loc file must exist on the dbTier prior to provisioning. This file will specify where the oraInventory directories live or where they should be created if they do not already exist.

The oraInst.loc file is typically located at `/etc/oraInst.loc` on Linux or `/var/opt/oracle/oraInst.loc` on Solaris. Ensure that the oraInventory to which this file points is writable by the `oracle` user.

Consult Oracle EBS documentation for more information about where to place this file on your dbTier and what this file should contain.

> **Oracle Central Inventory**
> The Oracle Central Inventory is a repository for all Oracle products (software) installed on a system. Since the Central Inventory consists of system-specific information, it is required that the Central Inventory be saved on a local non-shared directory on the system. While software can be shared across nodes, the inventory should be local to each system. The global inventory/central inventory(oraInventory) for each node should be on a local file system other than Delphix mount base path.
> For more info, see the following notes

- **Note 360079.1- Global and Local Inventory explained.**
- **Note 564192.1- FAQs on Central Inventory and Oracle Home Inventory (Local Inventory) in Oracle RDBMS.**
- **Note 834894.1- How to Create a Clean oraInventory in Release 12?**
- **Note 742477.1 - How to create, update or rebuild the Central Inventory for Applications R12?**
- **Note 295185.1- How to Recreate the Global oraInventory?**

/etc/hosts

Use the following to verify hostname settings:
For Oracle Linux 5, 6, and 7, and Red Hat Enterprise Linux 5, 6, and 7:

- Verify that the /etc/hosts file is formatted as follows:

```
127.0.0.1 localhost.localdomain localhost
[ip_address] [node_name].[domain_name] [node_name]
```

Hostname and its IP address of Database host and Appstier host(s) (primary and secondary hosts for multinode) should be added in the `/etc/hosts` file of all the Database, host, Appstier host(s) (primary and secondary hosts for Multinode).

These correct entries are required to avoid adcfgclone timeout issue on the prompt for "**Target System Domain Name**".

Kernel shmmax and shmall values

- The minimum required kernel.shmmax value on target hosts should be atleast half the size of physical memory in bytes.
- The minimum required kernel.shmall value on target hosts should be greater than or equal to the value of shmmax, in pages.
  For more information refer to Oracle documentation: https://docs.oracle.com/en/database/oracle/oracle-database/19/ladbi/minimum-parameter-settings-for-installation.html#GUID-CDEB89D1-4D48-41D9-9AC2-6AD9B0E944E3

Clean up before provisioning option

If you plan to utilize the Cleanup Before Provision option available during dbTechStack provisioning, the Delphix Engine requires the Database Oracle Home to be patched with Oracle Universal Installer (OUI) version 10.2 or above. You can read more about the Clean Up Provisioning option in Provisioning a Virtual EBS R12.2 Instance. Note that provisioning is still possible without this option specified, but you will need to manage the target dbTier's Oracle Inventory manually to ensure that conflicting entries do not cause provisions to fail.

The Delphix Engine's Oracle database requirements

The dbTier must meet the target requirements outlined in Oracle support and requirements. These requirements are generic to all target Unix environments expected to host a virtual Oracle database.

Prepare the appsTier for provisioning

Delphix Engine's Unix environment requirements

The appsTier must meet the target requirements outlined in Requirements for Unix environments. These requirements are generic to all target Unix environments added to the Delphix Engine.

`applmgr` user

The Delphix Engine must have access to an `applmgr` user on the appsTier.

- This user should be a member of the EBS `oinstall` group.
- This user will be given proper permissions to manage the appsTier.

delphix_os OS user account

In order to separate authentication and perform privileged operations with a non-privileged OS account, first create an OS user account (i.e. "delphix_os") on the EBS DB Tier and appsTier node to be used as a target. This user is easily created by the createDelphixOSUser.sh (located below on this page) script.

- The primary OS group of the Delphix Engine software owner account's (i.e. delphix_os) should be the same as the EBS Database/AppsTier software owner account (i.e. oravis or applmgr).
- Primary group = Oracle Install Group (typically oinstall), secondary group = OSDBA Group (typically dba). There are lots of cases where the OS group named dba fills this role, so be sure to check the group membership of the EBS Database and AppsTier software owner account.
- Please note, the non-privileged OS account must have the **same** group as assigned to EBS Database or appsTier privileged account (like oravis or applmgr).
- Ensure that "expect", "wget", and "lsof" utility exist on the remote host.

Host requirements for multinode:

- The primary host and the secondary host user name and its UID, Group name and GID should be same to avoid the permission issue during the provisioning operation.
- Check the high and low privilege user id and GID in the primary and secondary node and change it accordingly using groupmod with the following commands:

```
$ id oravis
uid=509(oravis) gid=502(oinstall)
$ id delphix_os
uid=510(delphix_os) gid=502(oinstall)
Change the group id as root using,
```
```
# groupmod -g 502 oinstall
# usermod -g oinstall oravis
# usermod -g oinstall delphix_os
```

Primary and secondary appstier host need to be configured with passwordless ssh if the target WLS password is going to be different from the source WLS password. Passwordless ssh configuration on the Appstier multinode host requirement is required for configuring WLS password different from source.

Host requirements:

To accomplish necessary tasks on the EBS Database and appsTier target hosts, the Delphix OS user account (henceforth referred to as "delphix_os") requires privilege elevation specifications. Ensure that "wget" and "lsof" utilities exist on the remote host.

Here is an example specification for the "sudo" privilege elevation utility, using the "visudo" to edit the "sudoers" configuration file. This specification makes the following assumptions:

- OS = Linux
  - OS account owning Oracle EBS Database Tier is named **oravis**
  - OS account owning Oracle EBS appsTier is named **applvis**
- OS = Solaris
  - OS account owning Oracle EBS Database Tier is named **oravis**
  - OS account owning Oracle EBS appsTier is named **oravis**
- EBS Database and appsTier system base directory is */u01/oracle/VIS***.**This can be prefixed before all the below-mentioned commands to ensure the same is being executed from relevant paths.

The following sudoers entry is only for template purposes. Modify the path in the below sudoers entry with the appropriate binary paths of your environment.

Sudoers entry for Linux

Entries required for provisioning via low privileged user (delphix_os):	

EBS DB Tier	`Defaults:delphix_os !requiretty`
	
	`delphix_os ALL=NOPASSWD: /bin/su - oravis -c /bin/mount, /bin/su - oravis -c /bin/umount, /bin/su - oravis -c echo *, /bin/su - oravis -c export *, /bin/su - oravis -c test*mkdir*, /bin/su - oravis -c test*touch*chmod 750*cat*, /bin/su - oravis -c */EBS_kill/kill_script.sh*, /bin/su - oravis -c rm -f */test_status.tmp*, /bin/su - oravis -c cp -f */dlpx_force_autoflush*; chmod 755 */dlpx_force_autoflush*, /bin/su - oravis -c rm -f */dlpx_force_autoflush*, /bin/su - oravis -c */*.env*bin/lsnrctl status*, /bin/su - oravis -c */*.env*/addlnctl.sh*, /bin/su - oravis -c rm -f /u01/oracle/VIS/*/.delphix_adpreclone.lck*, /bin/su - oravis -c rm -rf */appsutil/clone/dbts*, /bin/su - oravis -c touch /u01/oracle/VIS/*/.delphix_adpreclone.lck*, /bin/su - oravis -c */bin/runInstaller -silent -detachHome*, /bin/su - oravis -c export PATH* export PERL5LIB*perl -mdlpx_force_autoflush */adcfgclone.pl dbTechStack*, /bin/su - oravis -c export PERL5LIB*perl -mdlpx_force_autoflush */adpreclone.pl dbTechStack*, /bin/su - oravis -c cd *; make -f`

	`ins_rdbms.mk*ioracle*, /bin/su - oravis -c cd *; make -f ins_rdbms.mk dnfs_off*, /bin/su - oravis -c rm -f *bak, /bin/su - oravis -c mv *, /bin/su - oravis -c */*.env*sqlplus* as sysdba*, /bin/su - oravis -c */*.env*perl */appsutil/clone/bin/adcfgclone.pl dbconfig*, /bin/su - oravis -c */*.env*perl */appsutil/scripts/*/adpreclone.pl database*, /bin/su - oravis -c sed*sqlnet.ora*, /bin/su - oravis -c */*.env*sqlplus apps*, /bin/su - oravis -c */*.env*; make -f *rdbms/lib/ins_rdbms.mk dnfs_off*, /bin/su - oravis -c */*.env*; ln -s *, /bin/mount, /bin/umount, /bin/ps, /bin/mkdir, /bin/su - oravis -c source* -dboraclehome* perl*txkGenCDBTnsAdmin.pl*, /bin/su - oravis -c */*.env*perl */appsutil/bin/txkCfgUtlfileDir.pl -contextfile*, /bin/su - oravis -c */*.env* mkdir -p*, /bin/su - oravis -c chmod 775 *dbs*, /bin/su - oravis -c chmod 6751 */bin/oracle*, /bin/su - oravis -c cp *dlpx_force_autoflush.pm*, /bin/su - oravis -c chmod 755 */dlpx_force_autoflush*, /bin/su - oravis -c umask*touch *source_apps_file.txt, /bin/su - oravis -c mkdir -p*, /bin/su - oravis -c cp *pairsfile*, /bin/su - oravis -c *perl -mdlpx_force_autoflush */adclonectx.pl*, /bin/su - oravis -c *perl -mdlpx_force_autoflush */adcfgclone.pl dbTechStack*, /bin/su - oravis -c touch */.delphix_adpreclone.lck*, /bin/su - oravis -c rm -f */.delphix_adpreclone.lck*`

EBS appsTier	`Defaults:delphix_os !requiretty`
	
	`delphix_os ALL=NOPASSWD: /bin/su - applvis -c echo *, /bin/su - applvis -c rm *.dlpx_run_edition*, /bin/su - applvis -c rm -f *dlpx_force_autoflush*, /bin/su - applvis -c cd *echo *dlpx_force_autoflush*, /bin/su - applvis -c export PATH* export PERL5LIB* cd *perl -mdlpx_force_autoflush ./adpreclone.pl appsTier*, /bin/su - applvis -c */rsync*, /bin/su - applvis -c test*mkdir*, /bin/su - applvis -c test*touch*chmod 750*cat*, /bin/su - applvis -c */EBS_kill/kill_script.sh*, /bin/su - applvis -c rm -f */test_status.tmp*, /bin/su - applvis -c */*.env* sqlplus -s *apps*, /bin/su - applvis -c */adstrtal.sh*, /bin/su - applvis -c */adstpall.sh*, /bin/su - applvis -c */*.env* cd *adautocfg.sh*, /bin/su - applvis -c`

```
export PATH* export PERL5LIB* cd *perl
-mdlpx_force_autoflush ./adcfgclone.pl appsTier*, /bin/su -
applvis -c *perl -mdlpx_force_autoflush */adclonectx.pl addnode
contextfile*pairsfile*outfile*, /bin/su - applvis -c
*adadminsrvctl.sh*, /bin/su - applvis -c
*admanagedsrvctl.sh*, /bin/su - applvis -c *adnodemgrctl.sh*, /
bin/su - applvis -c */*.env* cd *perl -mdlpx_force_autoflush ./
txkUpdateEBSDomain.pl*contextfile*action*updateAdminPassword*,
/bin/su - applvis -c */bin/runInstaller -silent -detachHome*, /
bin/su - applvis -c rm -rf */inst/apps/*, /bin/su - applvis -c
rm -rf *FMW_Home*, /bin/su - applvis -c rm -rf *fs*, /bin/su -
applvis -c find*exec rm -rf *, /bin/su - applvis -c cp */inst/
apps/*appl/admin* */inst/apps/*, /bin/su - applvis -c */
*EBSapps.env*perl */patch/115/bin/
txkSetAppsConf.pl*contextfile*configoption*oacore*oafm*forms*fo
rmsc4ws*, /bin/su - applvis -c rsync -aH --delete --ignore-
errors */EBSapps/ */EBSapps/*, /bin/su - applvis -c rm */
serviceStartfile.tmp*, /bin/su - applvis -c rm -rf */
change_apps_password*, /bin/su - applvis -c mkdir -p */
change_apps_password*, /bin/su - applvis -c */*.env* run; cd */
change_apps_password*/fnd/12.0.0/bin/
FNDCPASS*apps*system*SYSTEM APPLSYS*, /bin/su - applvis -c */
*.env* sqlplus *apps*, /bin/su - applvis -c */*.env* rm -f */
updateDSpwd.py*cat */updateDSpwd.py*, /bin/su - applvis -c */
*.env*/adautocfg.sh*, /bin/su - applvis -c */*.env* rm -f */
serverStateAll.py*cat*serverStateAll.py*, /bin/su - applvis -c
*/*.env* run; */wlserver_10.3/common/bin/wlst.sh */
updateDSpwd.py*, /bin/su - applvis -c */*.env*; */
wlserver_10.3/common/bin/wlst.sh */serverStateAll.py*, /bin/su
- applvis -c *lsof*, /bin/su - applvis -c *sed *, /bin/su -
applvis -c */*.env*;*adapcctl.sh status*, /bin/mount, /bin/
umount, /bin/ps, /bin/su - applvis -c find*, /bin/su - applvis
-c cp *dlpx_force_autoflush.pm*, /bin/su - applvis -c *perl
-mdlpx_force_autoflush ./adcfgclone.pl appsTier*, /bin/su -
applvis -c cat*serverStateAll.py*, /bin/su - applvis -c
cat*updateDSpwd.py*, /bin/su - applvis -c mkdir -p */
pairsdir*, /bin/su - applvis -c export PERL5LIB* cd *perl
```

	`-mdlpx_force_autoflush *adpreclone.pl appsTier*, /bin/su - applvis -c mv *scratch_file* *, /bin/su - applvis -c pmap -r*, /bin/su - applvis -c */*.env* patch; */wlserver_10.3/common/bin/wlst.sh */updateDSpwd.py*`

Sudoers entry for Solaris

Entries required for provisioning via low privileged user (delphix_os):	
EBS DB Tier	`Defaults:delphix_os !requiretty` `delphix_os ALL=NOPASSWD: /usr/bin/su - oravis -c /usr/sbin/mount, /usr/bin/su - oravis -c /usr/sbin/umount, /usr/bin/su - oravis -c echo *, /usr/bin/su - oravis -c export *, /usr/bin/su - oravis -c test*mkdir*, /usr/bin/su - oravis -c test*touch*chmod 750*cat*, /usr/bin/su - oravis -c */EBS_kill/kill_script.sh*, /usr/bin/su - oravis -c rm -f */test_status.tmp*, /usr/bin/su - oravis -c cp -f */dlpx_force_autoflush*; chmod 755 */dlpx_force_autoflush*, /usr/bin/su - oravis -c rm -f */dlpx_force_autoflush*, /usr/bin/su - oravis -c */*.env*bin/lsnrctl status*, /usr/bin/su - oravis -c */*.env*/addlnctl.sh*, /usr/bin/su - oravis -c rm -f */.delphix_adpreclone.lck*, /usr/bin/su - oravis -c rm -rf */appsutil/clone/dbts*, /usr/bin/su - oravis -c touch */.delphix_adpreclone.lck*, /usr/bin/su - oravis -c */bin/runInstaller -silent -detachHome*, /usr/bin/su - oravis -c export PATH*`

```
export PERL5LIB*perl
-mdlpx_force_autoflush */
adcfgclone.pl dbTechStack*, /usr/bin/
su - oravis -c export PERL5LIB*perl
-mdlpx_force_autoflush */
adpreclone.pl dbTechStack*, /usr/bin/
su - oravis -c cd *; make -f
ins_rdbms.mk*ioracle*, /usr/bin/su -
oravis -c cd *; make -f ins_rdbms.mk
dnfs_off*, /usr/bin/su - oravis -c rm
-f *bak, /usr/bin/su - oravis -c mv
*, /usr/bin/su - oravis -c */
*.env*sqlplus* as sysdba*, /usr/bin/
su - oravis -c */*.env*perl */
appsutil/clone/bin/adcfgclone.pl
dbconfig*, /usr/bin/su - oravis -c */
*.env*perl */appsutil/scripts/*/
adpreclone.pl database*, /usr/bin/su
- oravis -c sed*sqlnet.ora*, /usr/
bin/su - oravis -c */*.env*sqlplus
apps*, /usr/bin/su - oravis -c */
*.env*; make -f *rdbms/lib/
ins_rdbms.mk dnfs_off*, /usr/bin/su -
oravis -c */*.env*; ln -s *, /usr/
sbin/mount, /usr/sbin/umount, /usr/
bin/ps, /usr/bin/mkdir, /usr/bin/su -
oravis -c source* -dboraclehome*
perl*txkGenCDBTnsAdmin.pl*, /usr/bin/
su - oravis -c */*.env*perl */
appsutil/bin/txkCfgUtlfileDir.pl
-contextfile*, /usr/bin/su - oravis
-c */*.env* mkdir -p*, /usr/bin/su -
oravis -c chmod 775 *dbs*, /usr/bin/
su - oravis -c chmod 6751 */bin/
oracle*, /usr/bin/su - oravis -c cp
*dlpx_force_autoflush.pm*, /usr/bin/
su - oravis -c chmod 755 */
```

	`dlpx_force_autoflush*, /usr/bin/su - oravis -c umask*touch *source_apps_file.txt, /usr/bin/su - oravis -c mkdir -p*, /usr/bin/su - oravis -c cp *pairsfile*, /usr/bin/su - oravis -c *perl -mdlpx_force_autoflush */adclonectx.pl*, /usr/bin/su - oravis -c *perl -mdlpx_force_autoflush */adcfgclone.pl dbTechStack*, /usr/bin/su - oravis -c touch */.delphix_adpreclone.lck*, /usr/bin/su - oravis -c rm -f */.delphix_adpreclone.lck*, /usr/bin/su - oravis -c chmod 755 *hooksUtil*, /usr/bin/su - oravis -c set -o *hooksUtil*, /usr/bin/su - oravis -c mkdir *hooksUtil*, /usr/bin/mkdir, /usr/bin/rmdir, /usr/sbin/mount, /usr/sbin/umount, /usr/bin/pargs, /usr/bin/ps, /usr/bin/netstat`
EBS appsTier	`Defaults:delphix_os !requiretty delphix_os ALL=NOPASSWD: /usr/bin/su - oravis -c echo *, /usr/bin/su - oravis -c rm *.dlpx_run_edition*, /usr/bin/su - oravis -c rm -f *dlpx_force_autoflush*, /usr/bin/su - oravis -c cd *echo *dlpx_force_autoflush*, /usr/bin/su - oravis -c export PATH* export PERL5LIB* cd *perl -mdlpx_force_autoflush ./adpreclone.pl appsTier*, /usr/bin/su - oravis -c */rsync*, /usr/bin/su - oravis -c test*mkdir*, /usr/bin/su - oravis -c test*touch*chmod`

```
750*cat*, /usr/bin/su - oravis -c */
EBS_kill/kill_script.sh*, /usr/bin/su
- oravis -c rm -f */
test_status.tmp*, /usr/bin/su -
oravis -c */*.env* sqlplus -s
*apps*, /usr/bin/su - oravis -c */
adstrtal.sh*, /usr/bin/su - oravis -c
*/adstpall.sh*, /usr/bin/su - oravis
-c */*.env* cd *adautocfg.sh*, /usr/
bin/su - oravis -c export PATH*
export PERL5LIB* cd *perl
-mdlpx_force_autoflush ./
adcfgclone.pl appsTier*, /usr/bin/su
- oravis -c *perl
-mdlpx_force_autoflush */
adclonectx.pl addnode
contextfile*pairsfile*outfile*, /usr/
bin/su - oravis -c
*adadminsrvctl.sh*, /usr/bin/su -
oravis -c *admanagedsrvctl.sh*, /usr/
bin/su - oravis -c
*adnodemgrctl.sh*, /usr/bin/su -
oravis -c */*.env* cd *perl
-mdlpx_force_autoflush ./
txkUpdateEBSDomain.pl*contextfile*act
ion*updateAdminPassword*, /usr/bin/su
- oravis -c */bin/runInstaller
-silent -detachHome*, /usr/bin/su -
oravis -c rm -rf */inst/apps/*, /usr/
bin/su - oravis -c rm -rf
*FMW_Home*, /usr/bin/su - oravis -c
rm -rf *fs*, /usr/bin/su - oravis -c
find*exec rm -rf *, /usr/bin/su -
oravis -c cp */inst/apps/*appl/admin*
*/inst/apps/*, /usr/bin/su - oravis
-c */*EBSapps.env*perl */patch/115/
bin/
```

```
txkSetAppsConf.pl*contextfile*configoption*oacore*oafm*forms*formsc4ws*, /usr/bin/su - oravis -c rsync -aH --delete --ignore-errors */EBSapps/ */EBSapps/*, /usr/bin/su - oravis -c rm */serviceStartfile.tmp*, /usr/bin/su - oravis -c rm -rf */change_apps_password*, /usr/bin/su - oravis -c mkdir -p */change_apps_password*, /usr/bin/su - oravis -c */*.env* run; cd */change_apps_password*/fnd/12.0.0/bin/FNDCPASS*apps*system*SYSTEM APPLSYS*, /usr/bin/su - oravis -c */*.env* sqlplus *apps*, /usr/bin/su - oravis -c */*.env* rm -f */updateDSpwd.py*cat */updateDSpwd.py*, /usr/bin/su - oravis -c */*.env*/adautocfg.sh*, /usr/bin/su - oravis -c */*.env* rm -f */serverStateAll.py*cat*serverStateAll.py*, /usr/bin/su - oravis -c */*.env* run; */wlserver_10.3/common/bin/wlst.sh */updateDSpwd.py*, /usr/bin/su - oravis -c */*.env*; */wlserver_10.3/common/bin/wlst.sh */serverStateAll.py*, /usr/bin/su - oravis -c *lsof*, /usr/bin/su - oravis -c *sed *, /usr/bin/su - oravis -c */*.env*;*adapcctl.sh status*, /usr/sbin/mount, /usr/sbin/umount, /usr/bin/ps, /usr/bin/su - oravis -c find*, /usr/bin/su - oravis -c cp *dlpx_force_autoflush.pm*, /usr/bin/su - oravis -c *perl -mdlpx_force_autoflush ./adcfgclone.pl appsTier*, /usr/bin/su
```

	`- oravis -c cat*serverStateAll.py*, /usr/bin/su - oravis -c cat*updateDSpwd.py*, /usr/bin/su - oravis -c mkdir -p */pairsdir*, /usr/bin/su - oravis -c export PERL5LIB* cd *perl -mdlpx_force_autoflush *adpreclone.pl appsTier*, /usr/bin/su - oravis -c mv *scratch_file* *, /usr/bin/su - oravis -c pmap -r*, /usr/bin/su - oravis -c */*.env* patch; */wlserver_10.3/common/bin/wlst.sh */updateDSpwd.py*, /usr/bin/su - oravis -c */*.env* run; */adcmctl.sh status*, /usr/bin/su - oravis -c set -o pipefail; cat*tee */.*_pairs.txt*, /usr/bin/su - oravis -c chmod 755 */.*_pairs.txt*, /usr/bin/su - oravis -c *perl -mdlpx_force_autoflush */adcfgclone.pl*appsTier*, /usr/bin/su - oravis -c /usr/bin/netstat -an *, /usr/bin/mkdir, /usr/bin/rmdir, /usr/sbin/mount, /usr/sbin/umount, /usr/bin/pargs, /usr/bin/ps, /usr/bin/netstat`

oraInst.loc

An oraInst.loc file must exist on every appsTier node prior to provisioning. This file will specify where the oraInventory directories live or where they should be created if they do not already exist.

The oraInst.loc file is typically located at `/etc/oraInst.loc` on Linux or `/var/opt/oracle/oraInst.loc` on Solaris. Ensure that the oraInventory to which this file points is writable by the `applmgr` user or application tier OS user.

If you are provisioning a single-node appsTier,

- [Required] The global inventory/central inventory(oraInventory) for each node should be on a local file system other than Delphix mount base path.

If you are provisioning a multi-node appsTier,

- The following recommendations are mandatory;

    - While configuring SHARED APPL_TOP application, Delphix expects the oraInventory directory path should be beneath the Delphix mount base path.
    - The `inventory_loc` variable defined in `file /etc/oraInst.loc` on all application servers should point to same shared oraInventory directory location on shared Delphix mount base path.
    - The Delphix Engine's cloning workflow requires that all nodes in the appsTier have access to the shared oraInventory directories via Delphix-provided storage.
- [Required] The Delphix Engine's automation requires that all nodes in the appsTier have access to the oraInventory directories via Delphix-provided storage.

> Objective of placing oraInventory on Shared Delphix mount storage
> While configuring shared APPL_TOP application, instead of storing oraInventory on the local directory of each server, all application servers should point to the same oraInventory directory on shared Delphix mount storage, so that you need to patch only on a single server and the oraInventory gets in-sync for all application servers thus preventing the oraInventory of remaining servers becoming invalid.

Details can be found in **Sharing The Application Tier File System in Oracle E-Business Suite Release 12 (Doc ID 384248.1)** found at https://support.oracle.com . You can also consult Oracle EBS documentation for more information about where to place this file on your appsTier and what this file should contain.

Clean up before provisioning option

If you plan to utilize the Cleanup Before Provision option available during appsTier provisioning, the Delphix Engine requires the Tools Oracle Home to be patched with Oracle Universal Installer (OUI) version 10.2 or above. You can read more about this provisioning option in Provisioning a Virtual EBS R12.2 Instance. Note that provisioning is still possible without this option specified, but you will need to manage the target appTier's Oracle Inventory manually to ensure that conflicting entries do not cause provisions to fail.

Script

createDelphixOSUser.sh

Provisioning a virtual EBS R12.2 instance

This topic describes the process of provisioning a virtual EBS R12.2 instance.

Maintain the integrity in names(CDB name, PDB name, DB hostname, /etc/hosts file format nomenclature). During dbTechStack, vPDB or vDB provisioning, maintain the same lower case and use the same name vPDB during virtual Appstier provision as well. There is no restriction on using UPPER or LOWER case characters. However, it is recommended to maintain consistency.

Prerequisites

- You must have already linked a source instance of EBS R12.2. For more information, see Linking a source EBS R12.2 instance
- Prepare your target EBS R12.2 environments for provisioning by following the outline in Preparing target EBS R12.2 environments for provisioning

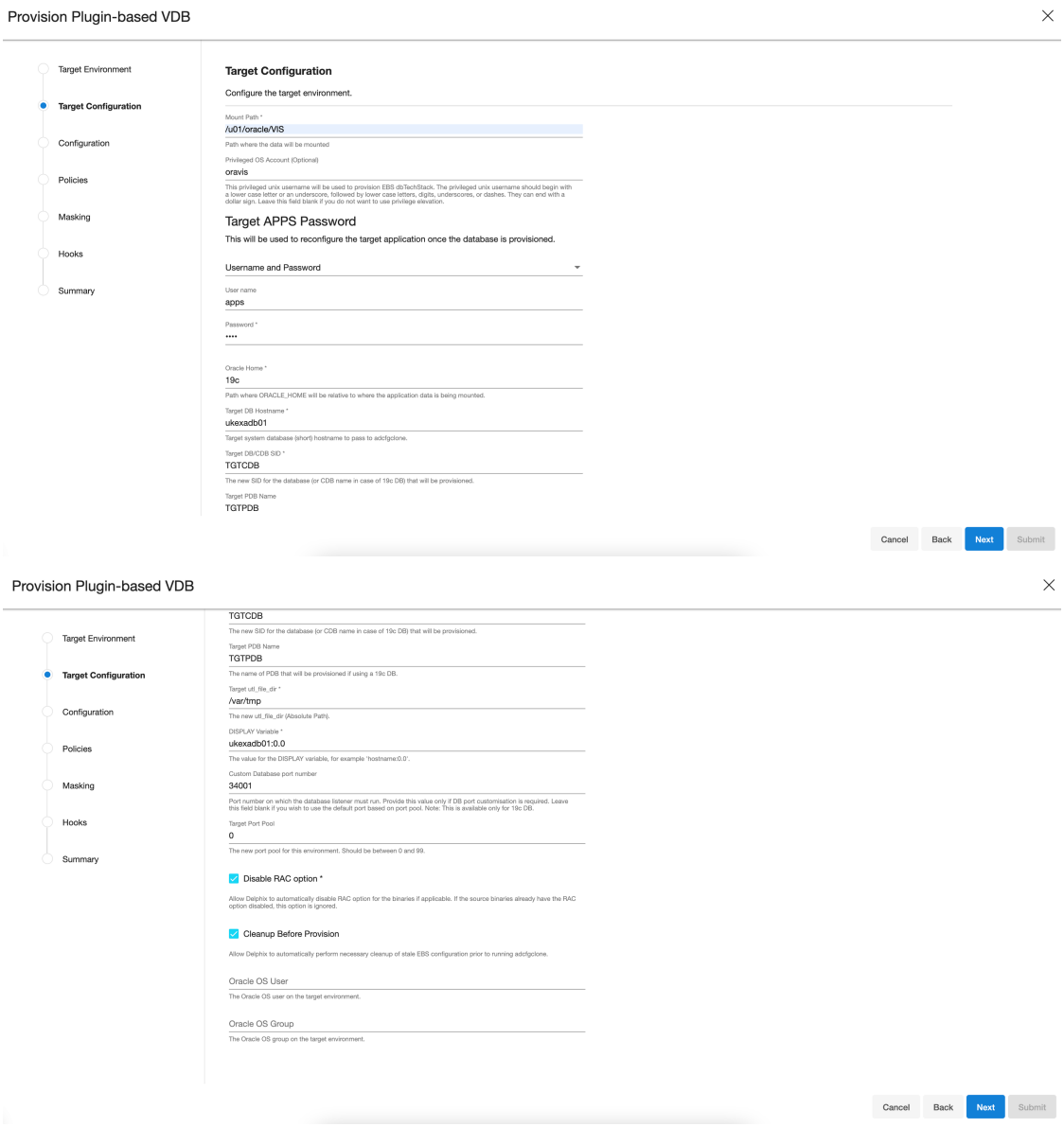
Snapshot coordination
Changes applied to EBS and picked up only in certain dSource snapshots may make certain combinations of snapshots across the appsTier and dbTier incompatible. When provisioning, refreshing or rewinding a virtual EBS instance, be sure the points in time you select for each dataset are compatible with each other.

The source DB host may have multiple DATA_TOPs but the target DB host will be provisioned with a single DATA_TOP that resides on the target Mount Base path provided during the database provisioning.

Provisioning the EBS dbTechStack

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **dbTechStack dSource**.
5. Click the **TimeFlow** tab.
6. Select a dSource **snapshot**.
7. Click **Provision**.
   The **Provision VDB** wizard will open.
8. Select an **Environment.**
   This environment will host the virtual dbTechStack and be used to execute hook operations specified in step 15 in the **s**section.
9. Select **E-Business Suite R12.2 dbTechStack** from the **Installation** dropdown.
10. Select an **Environment User**.
11. This user should be the `oracle` user-outlined in Preparing target EBS R12.2 environments for provisioning
12. Enter a **Mount Path** for the virtual dbTechStack files.
13. Enter the **EBS-specific parameters** for the virtual dbTechStack. A subset of these parameters is discussed in more detail below.
    a. The **Privileged OS Account (Optional)** field should contain the high privileged user when the low privileged user is being used for provisioning.
    b. The **Target APPS Password** is the new apps password that is required to configure virtual dbTechStack. This password is encrypted when stored within the Delphix Engine and is available as an environment variable to the `adcfgclone` process.

c. Provide source oracle home directory name relative to Oracle base installation directory that was provided during linking of dbTechStack in Oracle Home input field.
d. Ensure that the **Target DB Hostname** value is the short hostname, not the fully-qualified hostname.
e. The **Target DB/CDB SID** is the new database SID (CDB SID in case of 19c) that is required to configure virtual dbTechStack.
f. The **Target PDB Name** field is added for the pluggable database to be configured for Oracle 19c database. Please leave this field empty when using Oracle 11g/12c database.
g. Provide **Target utl_file_dir** , the default value for this field will be “/var/tmp".
The ***utl_file_dir*** initialization parameter is deprecated in Oracle Database 12c Release 2 (12.2.0.1), and maybe unsupported in a future release.
h. Provide **DISPLAY Variable**, the default value is hostname:0.0
i. [Optional] Provide the **Custom Database Port Number** on which database listener must run.
Info:
    i. Enter a numeric port value in this field only if you want to use DB port customisation. Leaving this field blank automatically uses the default port based on port pool. The custom database port number is applicable only for Oracle 19c database.
    ii. To run the dbTechStack and listener on custom DB port, it is recommended to provide **Target Port Pool** value either as default or 0; also the **Target Port Pool** value in dbTechStack is not related with **Run/Patch Edition Port Pool** value in Appstier.
j. Provide **Target Port Pool**, please provide a port pool that is available. By default, the value is 1. The range for this value is 0 to 99.
k. Enable the **Disable RAC** option if you want to permit the Delphix Engine to automatically disable the RAC option for the binaries when applicable.
This option is necessary if provisioning from a dSource with RAC dbTier because the binaries are relinked with the `rac_on` option even after running `adcfgclone`. If the source binaries already have the RAC option disabled (also the case for SI dbTier), the Delphix Engine ignores this option.
l. Enable the **Cleanup Before Provision**option if you want to permit the Delphix Engine to automatically clean up stale EBS configuration during a refresh. This option is recommended, but only available if your Oracle Home is patched with Oracle Universal Installer (OUI) version 10.2 or above.
    i. With this option enabled, the Delphix Engine will inspect the target environment's oraInventory prior to refreshing this virtual dbTechStack. If any Oracle Homes are already registered within the specified **Mount Path**, the Delphix Engine will detach them from the inventory prior to running adcfgclone. These homes must be detached prior to running post-clone configuration. If they are not detached, `adcfgclone` will fail, citing conflicting oraInventory entries as an issue.
    ii. Without this option enabled, Oracle Homes that conflict with the specified **Mount Path** will be reported in an error instead of automatically detached. For refresh to succeed, you must manually detach conflicting Oracle Homes prior to refresh.
m. Provide **Oracle OS User** and **Group**, it’s optional.



14. Click **Next**.
15. Enter a **VDB Name**.
16. Select a **Target Group** for the VDB.
    If necessary, click the green **Plus** icon to add a new group.
17. Select a **Snapshot Policy** for the VDB.
    If necessary, click the **Plus** icon to create a new policy.
    Warning: Snapshot conflicts when Snapshot is running against the dbTechStack, database, or appsTier, the Delphix Engine also executes pre-clone logic to ensure the latest configuration is staged in the captured snapshots. Unfortunately, if multiple Snapshots are running against the same EBS instance concurrently, this pre-clone logic may fail and produce bad snapshots.
    To avoid SnapSync conflicts, spread out your SnapSync policies for an EBS instance by one hour or more.
18. Click **Next**.
19. Enter any **custom hook operations** that are needed to help correctly manage the virtual dbTechStack files. For more information about these hooks, when they are run, and how operations are written.

The Configure Clone hook will be run after the `adcfgclone.pl` tool has both mounted and configured the dbTechStack.

20. Click **Next**.
21. Click **Submit**.

When provisioning starts, you can review the progress of the job in the **Datasets** panel, or in the **Job History** panel of the **Dashboard**. When provisioning is complete, the dbTechStack VDB will be included in the group you designated and listed in the **Datasets** panel. If you select the dbTechStack VDB in the **Datasets** panel and click the **Configuration** tab, you can view information about the virtual files and its Data Management settings.

For tips on monitoring the progress of dbTechStack provisioning, see Monitoring EBS R12.2 dbTechStack Provisioning Progress

Provisioning the Oracle 19C database

1. Refresh the Target DB environment from **Manage > Environments** so that the Listener brought up during dbTechStack provision gets discovered and the Installation Home to be listed in dropdown during Database provision. If Target DB environment is not refreshed, you will see `This environment has no compatible Oracle Installation Homes` in the Installation Home dropdown in the **Database provision**.
2. Provision the EBS database to the target dbTier environment by following the steps outlined in Provisioning an Oracle VDB. For Oracle EBS database for 19c MT, provision the database by following the steps outlined in Provisioning an Oracle Virtual Pluggable Database (vPDB)
   Note: Snapshot conflicts: when Snapshot is running against the dbTechStack, database, or appsTier, the Delphix Engine also executes pre-clone logic to ensure the latest configuration is staged in the captured snapshots. Unfortunately, if multiple Snapshots are running against the same EBS instance concurrently, this pre-clone logic may fail and produce bad snapshots.
   To avoid SnapSync conflicts, spread out your SnapSync policies for an EBS instance by one hour or more.
3. Select the correct **Installation Home**.
   This should be the virtual dbTechStack you just added to the Delphix Engine.
4. Select an Environment User.
5. This user should be the oracle user-outlined in Preparing Target EBS R12.2 Environments for Provisioning.
6. For the **Container Database** option, select the checkbox next to **Create a New Container Database**, and click **Next**.
7. Select a Target Group for the VDB.
8. In the vPDB **Configuration** section, provide the Target PDB name given during dbTechStack provision in the **Oracle Pluggable Database Name** field and provide any unique user-defined name in **vPDB Nam**e to be displayed in the Delphix Engine.
9. In vCDB **CONFIGURATION** section, provide the Target DB/CDB SID name given during dbTechStack provision in the **Database Name** field and SID and provide any unique user-defined name in the **vCDB Name** and **Database Unique Name** field, and click **Next**.
10. Click **Advanced**.
11. Select the correct **Oracle Node Listeners** value.
    This should be the listener corresponding to the virtual dbTechStack you just added to the Delphix Engine.
12. Add the EBS R12.2 dbTier environment file as a **Custom Environment Variables** entry.
13. This file can be specified as an Environment File with Path Parameters of `$ORACLE_HOME/<CONTEXT_NAME>` .

    Replace `<CONTEXT_NAME>` with the virtual EBS instance's context name. The Delphix Engine will expand the `$ORACLE_HOME` variable at runtime.

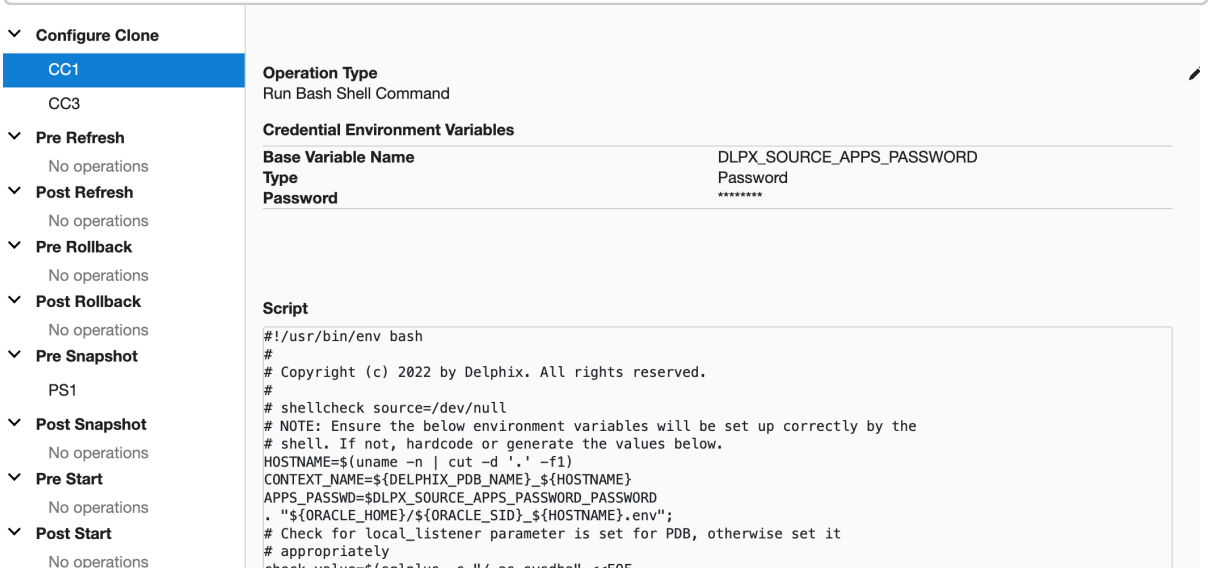
    For more information, see Customizing Oracle VDB environment variables
    Note: To complete the DB Provisioning operation, you must provide content to configure clones and pre-snapshot hooks. You must either follow step 14 to use the hooks utility feature OR proceed with the conventional way by following steps 15-21.

14. To complete the DB Provisioning operation, you must provide content to configure clones and pre-snapshot hooks. The following are the content for both the hooks:
    a. Configure Clone hook will be `<DBTechStack Mount point>/hooksUtil/hooksRunner --operation configure`. You must set a hooks environment variable `DLPX_SOURCE_APPS_PASSWORD` for providing the source apps schema password.
    b. Pre-Snapshot hook : `<DBTechStack Mount point>/hooksUtil/hooksRunner --operation pre-snapshot`. You must set a hooks environment variable `DLPX_SOURCE_APPS_PASSWORD` for providing the source apps schema password. In case the apps password change is required in target VDB, you can set hook environment variables `DLPX_TARGET_APPS_PASSWORD` and `DLPX_SYS_PASSWORD` for providing the target apps schema and system schema passwords.
    Note:
    For versions 19.3 or later, follow the below guidelines if you see any of the following error:
        i. ORA-01017: invalid username/password logon denied
        ii. ORA-28040 During cloning: No Matching Authentication Protocol
    EBS application database user accounts created in the earlier release uses a case-insensitive password version from an earlier release authentication protocol, such as the 10G password version. If your release user account passwords have not been reset and the following conditions persists, then take the action as described below:
    On source, the database server has been configured with SEC_CASE_SENSITIVE_LOGON set to FALSE, so that it can only authenticate users who have a 10G case-insensitive password version. Then, on target container VDB, it becomes necessary to set the SEC_CASE_SENSITIVE_LOGON to FALSE, to make authentication of users successful.
    To take advantage of the password protections introduced in Oracle Database 19c, users must change their passwords.
    Occasionally, it is necessary to restart the database to resolve connectivity to the database. This is atypical but may be necessary.
    To set-up target:
    On target container database, run the following command:
    alter system set SEC_CASE_SENSITIVE_LOGON=FALSE scope=both;
15. Add a **Run Bash Shell Command** operation to the Configure Clone hook to ensure that `adcfgclone` is run against the newly provisioned database for a **high privileged user**, as shown in the script below.
    Note: You can pass the credentials securely to Hook Operations by setting up the base variables. See Passing credentials securely to hook operations for more details.
    Note: It is mandatory to set a hooks environment variable `DLPX_SOURCE_APPS_PASSWORD` for providing the source apps schema password. In case the apps password change is required in target VDB, you can set hook environment variables `DLPX_TARGET_APPS_PASSWORD` and `DLPX_SYS_PASSWORD` for providing the target apps schema and system schema passwords in respective hooks. The variables will be declared in **Credential Environment Variables** hook section.
    The hooks environment variable will be declared as, DLPX_SOURCE_APPS_PASSWORD and then its password value will be the same variable used in hooks, along with the PASSWORD keyword appended at the end so that it will be like DLPX_SOURCE_APPS_PASSWORD_PASSWORD
    We assign the value of $DLPX_SOURCE_APPS_PASSWRD to APPS_PASSWD.
    In hooks, it will be written as

    ```
    APPS_PASSWD=$DLPX_SOURCE_APPS_PASSWORD_PASSWORD
    ```

    Same goes for,

```
SYSTEM_PASSWD=$DLPX_SYS_PASSWORD_PASSWORD
```



Configure clone first hook for a 19c multi-tenant high privileged user

```
#!/usr/bin/env bash
#
# Copyright (c) 2022 by Delphix. All rights reserved.
#
# shellcheck source=/dev/null
# NOTE: Ensure the below environment variables will be set up correctly by the
# shell. If not, hardcode or generate the values below.
HOSTNAME=$(uname -n | cut -d '.' -f1)
CONTEXT_NAME=${DELPHIX_PDB_NAME}_${HOSTNAME}
APPS_PASSWD=$DLPX_SOURCE_APPS_PASSWORD_PASSWORD
. "${ORACLE_HOME}/${ORACLE_SID}_${HOSTNAME}.env";
# Check for local_listener parameter is set for PDB, otherwise set it
# appropriately
check_value=$(sqlplus -s "/ as sysdba" <<EOF
alter session set container="${DELPHIX_PDB_NAME}";
show parameter local_listener;
EOF
)
local_listener=$(echo "$check_value" | awk '{print $11}')
value=("${local_listener//:/ }")
host="${value[0]}"
port="${value[1]}"
curr_port=$(grep PORT < "${ORACLE_HOME}/network/admin/listener.ora" | awk
'{print $9}' | sed 's/)//g')
if [[ $port != "$curr_port" || $host != "${HOSTNAME}" ]]; then
sqlplus -s "/ as sysdba" <<EOF
alter session set container="${DELPHIX_PDB_NAME}";
```

```
alter system set local_listener='${HOSTNAME}:${curr_port}';
alter system register;
EOF
fi

#For compatibility with 6.0.16.0 Delphix engine version
. ${ORACLE_HOME}/${ORACLE_SID}_${HOSTNAME}.env; sqlplus -s "/ as sysdba" <<EOF
shutdown immediate;
startup;
ALTER PLUGGABLE DATABASE ALL OPEN read write services=all;
alter pluggable database ${DELPHIX_PDB_NAME} open read write services=all;
alter pluggable database all save state instances=all;
EOF
sqlplus "/ as sysdba" <<EOF
@${ORACLE_HOME}/appsutil/install/${CONTEXT_NAME}/adupdlib.sql so
EOF
. "${ORACLE_HOME}/${CONTEXT_NAME}.env";
perl "${ORACLE_HOME}/appsutil/clone/bin/adcfgclone.pl" dbconfig "$
{ORACLE_HOME}/appsutil/${CONTEXT_NAME}.xml" <<EOF # noqa
${APPS_PASSWD}
EOF
```

16. Configure Clone hook to ensure that `adcfgclone` is run against the newly provisioned database for a **low privileged user**, as shown in the script below.
Note:
Customers have to identify the DLPX_DB_EXEC_SCRIPT value from their remote environment, this would always be inside the toolkit directory provided while adding the environment on Delphix Engine.
Configure clone first hook for a 19c multi-tenant low privileged user

```
#!/usr/bin/env bash
#
# Copyright (c) 2022 by Delphix. All rights reserved.
#
# shellcheck source=/dev/null
# NOTE: Ensure the below environment variables will be set up correctly by the
# shell. If not, hardcode or generate the values below.
DLPX_DB_EXEC_SCRIPT="<Remote BIN location till dlpx_db_exec script>"
DLPX_PRIV_USER=oravis
HOSTNAME=$(uname -n | cut -d '.' -f1)
APPS_PASSWD=$DLPX_SOURCE_APPS_PASSWORD_PASSWORD
CONTEXT_NAME=${DELPHIX_PDB_NAME}_${HOSTNAME}

"${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/${ORACLE_SID}
_${HOSTNAME}.env;"

# Check for local_listener parameter is set for PDB, otherwise set it
appropriately
check_value=$("${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/
${CONTEXT_NAME}.env; sqlplus -s \"/ as sysdba\" <<-EOF
alter session set container=${DELPHIX_PDB_NAME};
show parameter local_listener;
```

```
EOF
")

local_listener=$(echo "$check_value" | awk '{print $11}')
value=("${local_listener//:/ }")
host="${value[0]}"
port="${value[1]}"

curr_port=$(grep PORT < "${ORACLE_HOME}/network/admin/listener.ora" | awk
'{print $9}' | sed 's/)//g')

if [[ $port != "$curr_port" || $host != "${HOSTNAME}" ]]; then
    "${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/$
{ORACLE_SID}_${HOSTNAME}.env; sqlplus -s \"/ as sysdba\" <<EOF
    alter session set container=${DELPHIX_PDB_NAME};
    alter system set local_listener='${HOSTNAME}:${curr_port}';
    alter system register;
EOF
"
fi

#For compatibility with 6.0.16.0 Delphix engine version
"${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/${ORACLE_SID}
_${HOSTNAME}.env; sqlplus -s "/ as sysdba" <<EOF
shutdown immediate;
startup;
ALTER PLUGGABLE DATABASE ALL OPEN read write services=all;
alter pluggable database ${DELPHIX_PDB_NAME} open read write services=all;
alter pluggable database all save state instances=all;
EOF
"
"${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/${ORACLE_SID}
_${HOSTNAME}.env; sqlplus \"/ as sysdba\" <<-EOF
@${ORACLE_HOME}/appsutil/install/${CONTEXT_NAME}/adupdlib.sql so
EOF
"
"${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/$
{CONTEXT_NAME}.env; perl ${ORACLE_HOME}/appsutil/clone/bin/adcfgclone.pl
dbconfig ${ORACLE_HOME}/appsutil/${CONTEXT_NAME}.xml <<-EOF1
${APPS_PASSWD}
EOF1
"
```

17. For the EBS database Oracle 12.2, add a **Run Bash Shell Command** operation to the Configure Clone hook to ensure that `sqlnet.ora` or `sqlnet_ifile.ora` specify a value for `SQLNET.ALLOWED_LOGON_VERSION_SERVER` for a **high privileged user**, as shown in the script below.
    This requirement is outlined in **Cloning Oracle E-Business Suite Release 12.2 with Rapid Clone (Doc ID 1383621.1)** found at http://docs.oracle.com
18. When prompted, if your environment is on R12.TXK.C.Delta.13 or later, enter the password for the EBS_SYSTEM user, if your environment is on R12.TXK.C.Delta.12 or earlier, enter the password for the SYSTEM user, or if your source EBS setup is running on AD-TXK Delta 13 then you need to supply the

EBS_SYSTEM schema password in the below second hook script.
This requirement is outlined in the FAQ: Oracle E-Business Suite and System Schema Migration (DocID 2758999.1) Using UTL_FILE_DIR or Database Directories for PL/SQL File I/O in Oracle E-Business Suite Releases 12.1 and 12.2 (DocID 2525754.1) found at http://docs.oracle.com
Configure clone second hook for a 19c multi-tenant high privileged user

```
#!/usr/bin/env bash
#
# Copyright (c) 2022 by Delphix. All rights reserved.
#
# shellcheck source=/dev/null
# Set the target UTL_FILE_DIR values in the Oracle 19c database in the below
# variable UTL_FILE_DIR_PATH separated by commas
# Example UTL_FILE_DIR_PATH=/u01/oracle/VIS/temp/VISPDB,\
# /u01/oracle/VIS/19.3.0/appsutil/outbound/UTL,/u01/tmp

# UTL_FILE_DIR_PATH=<provide utl_file_dir locations separated by commas>
# If your environment is on R12.TXK.C.Delta.13 or later, enter the password
# for the EBS_SYSTEM user. If your environment is on R12.TXK.C.Delta.12 or
# earlier, enter the password for the SYSTEM user.
SYSTEM_PASSWD=$DLPX_SYS_PASSWORD_PASSWORD
DLPX_HOSTNAME=$(uname -n | cut -d '.' -f1)
TIMESTAMP=$(date +%d-%m-%Y_%H-%M-%S)
CONTEXT_NAME=${DELPHIX_PDB_NAME}_${DLPX_HOSTNAME}
APPS_PASSWD=$DLPX_SOURCE_APPS_PASSWORD_PASSWORD
DIR_OBJ_PATH="${ORACLE_HOME}/appsutil/outbound/${DELPHIX_PDB_NAME}_$
{DLPX_HOSTNAME}"
checkExists() {
if [[ ! -f $1 ]]; then
    echo "$1 does not exist."
    exit 1
fi
}
checkExists "${ORACLE_HOME}/${CONTEXT_NAME}.env"
checkExists "${ORACLE_HOME}/appsutil/bin/txkCfgUtlfileDir.pl"
. "${ORACLE_HOME}/${CONTEXT_NAME}.env";
echo "${APPS_PASSWD}" | perl "${ORACLE_HOME}/appsutil/bin/txkCfgUtlfileDir.pl"
 -contextfile="${ORACLE_HOME}/appsutil/${DELPHIX_PDB_NAME}_$
{DLPX_HOSTNAME}.xml" -oraclehome="${ORACLE_HOME}" -outdir="${ORACLE_HOME}/
appsutil/log" -mode=getUtlFileDir
checkExists "${ORACLE_HOME}/dbs/${DELPHIX_PDB_NAME}_utlfiledir.txt"
cp "${ORACLE_HOME}/dbs/${DELPHIX_PDB_NAME}_utlfiledir.txt" "${ORACLE_HOME}/dbs/
${DELPHIX_PDB_NAME}_utlfiledir.txt${TIMESTAMP}"
cat /dev/null > "${ORACLE_HOME}/dbs/${DELPHIX_PDB_NAME}_utlfiledir.txt"
if [ -z "$UTL_FILE_DIR_PATH" ]; then
    mkdir -p "${ORACLE_HOME}/appsutil/outbound/${DELPHIX_PDB_NAME}_$
{DLPX_HOSTNAME}"
    echo "${ORACLE_HOME}/appsutil/outbound/${DELPHIX_PDB_NAME}_$
{DLPX_HOSTNAME}" > "${ORACLE_HOME}/dbs/${DELPHIX_PDB_NAME}_utlfiledir.txt"
    mkdir -p "${ORACLE_BASE}/temp/${DELPHIX_PDB_NAME}"
```

```
    echo "${ORACLE_BASE}/temp/${DELPHIX_PDB_NAME}" >> "${ORACLE_HOME}/dbs/$
{DELPHIX_PDB_NAME}_utlfiledir.txt"
else
    echo "$UTL_FILE_DIR_PATH" | awk -F, '{ for (i = 1; i < NF+1; ++i ) print $i
>> "${ORACLE_HOME}/dbs/${DELPHIX_PDB_NAME}_utlfiledir.txt" }'
fi
{ echo "${APPS_PASSWD}"; echo "${SYSTEM_PASSWD}"; } | perl "${ORACLE_HOME}/
appsutil/bin/txkCfgUtlfileDir.pl" -contextfile="${ORACLE_HOME}/appsutil/$
{DELPHIX_PDB_NAME}_${DLPX_HOSTNAME}.xml" -oraclehome="${ORACLE_HOME}" -outdir="
${ORACLE_HOME}/appsutil/log" -mode=setUtlFileDir
{ echo "${APPS_PASSWD}"; echo "${SYSTEM_PASSWD}"; echo "${DIR_OBJ_PATH}"; } |
perl "${ORACLE_HOME}/appsutil/bin/txkCfgUtlfileDir.pl" -contextfile="$
{ORACLE_HOME}/appsutil/${DELPHIX_PDB_NAME}_${DLPX_HOSTNAME}.xml" -oraclehome="$
{ORACLE_HOME}" -outdir="${ORACLE_HOME}/appsutil/log" -mode=createDirObject
echo "${APPS_PASSWD}" | perl "${ORACLE_HOME}/appsutil/bin/txkCfgUtlfileDir.pl"
 -contextfile="${ORACLE_HOME}/appsutil/${DELPHIX_PDB_NAME}_$
{DLPX_HOSTNAME}.xml" -oraclehome="${ORACLE_HOME}" -outdir="${ORACLE_HOME}/
appsutil/log" -mode=syncUtlFileDir -skipautoconfig=yes
```

19. Configure Clone hook to ensure that `sqlnet.ora` or `sqlnet_ifile.ora` specify a value for `SQLNET.ALLOWED_LOGON_VERSION_SERVER` for a **low privileged user**, as shown in the script below.
When prompted, if your environment is on R12.TXK.C.Delta.13 or later, enter the password for the EBS_SYSTEM user, if your environment is on R12.TXK.C.Delta.12 or earlier, enter the password for the SYSTEM user, or if your source EBS setup is running on AD-TXK Delta 13 then you need to supply the EBS_SYSTEM schema password in the below second hook script.
This requirement is outlined in the FAQ: Oracle E-Business Suite and System Schema Migration (DocID 2758999.1) Using UTL_FILE_DIR or Database Directories for PL/SQL File I/O in Oracle E-Business Suite Releases 12.1 and 12.2 (DocID 2525754.1) found at http://docs.oracle.com
Configure clone second hook for a 19c multi-tenant low privileged user

```
#!/usr/bin/env bash
#
# Copyright (c) 2022 by Delphix. All rights reserved.
#
# shellcheck source=/dev/null
# Set the target UTL_FILE_DIR values in the Oracle 19c database
# Example UTL_FILE_DIR_PATH=/u01/oracle/VIS/temp/VISPDB,\
# /u01/oracle/VIS/19.3.0/appsutil/outbound/UTL

# UTL_FILE_DIR_PATH=<provide utl_file_dir locations separated by commas>
# If your environment is on R12.TXK.C.Delta.13 or later, enter the password
# for the EBS_SYSTEM user. If your environment is on R12.TXK.C.Delta.12 or
# earlier, enter the password for the SYSTEM user.
SYSTEM_PASSWD=$DLPX_SYS_PASSWORD_PASSWORD
DLPX_DB_EXEC_SCRIPT="<Remote BIN location till dlpx_db_exec script>"
DLPX_PRIV_USER=oravis
APPS_PASSWD=$DLPX_SOURCE_APPS_PASSWORD_PASSWORD
HOSTNAME=$(uname -n | cut -d '.' -f1)
CONTEXT_NAME=${DELPHIX_PDB_NAME}_${HOSTNAME}
DIR_OBJ_PATH="${ORACLE_HOME}/appsutil/outbound/${CONTEXT_NAME}"
TIMESTAMP=$(date +%d-%m-%Y_%H-%M-%S)
```

```
checkExists() {
if [[ ! -f $1 ]]; then
    echo "$1 does not exist."
    exit 1
fi
}
checkExists "${ORACLE_HOME}/${CONTEXT_NAME}.env"
checkExists "${ORACLE_HOME}/appsutil/bin/txkCfgUtlfileDir.pl"
. "${ORACLE_HOME}/${CONTEXT_NAME}.env";

"${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/$
{CONTEXT_NAME}.env; perl ${ORACLE_HOME}/appsutil/bin/txkCfgUtlfileDir.pl
-contextfile=${ORACLE_HOME}/appsutil/${CONTEXT_NAME}.xml -oraclehome=$
{ORACLE_HOME} -outdir=${ORACLE_HOME}/appsutil/log -mode=getUtlFileDir <<-EOF
${APPS_PASSWD}
EOF
"

checkExists "${ORACLE_HOME}/dbs/${DELPHIX_PDB_NAME}_utlfiledir.txt"
"${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" "cp ${ORACLE_HOME}/dbs/$
{DELPHIX_PDB_NAME}_utlfiledir.txt" "${ORACLE_HOME}/dbs/${DELPHIX_PDB_NAME}
_utlfiledir.txt${TIMESTAMP}"
"${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" "echo ""  > \"${ORACLE_HOME}/
dbs/${DELPHIX_PDB_NAME}_utlfiledir.txt\""

# Initialise for shell check to succeed
i=0
if [ -z "$UTL_FILE_DIR_PATH" ]; then
    "${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" "mkdir -p ${ORACLE_HOME}/
appsutil/outbound/${CONTEXT_NAME}"
    "${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" "echo ${ORACLE_HOME}/
appsutil/outbound/${CONTEXT_NAME} > ${ORACLE_HOME}/dbs/${DELPHIX_PDB_NAME}
_utlfiledir.txt"
    "${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" "mkdir -p ${ORACLE_BASE}/
temp/${DELPHIX_PDB_NAME}"
    "${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" "echo ${ORACLE_BASE}/temp/$
{DELPHIX_PDB_NAME} >> ${ORACLE_HOME}/dbs/${DELPHIX_PDB_NAME}_utlfiledir.txt"
else
    "${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" "echo $UTL_FILE_DIR_PATH |
awk -F, '{ for (i = 1; i < NF+1; ++i ) print $i }' > ${ORACLE_HOME}/dbs/$
{DELPHIX_PDB_NAME}_utlfiledir.txt"
fi

"${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/$
{CONTEXT_NAME}.env; perl ${ORACLE_HOME}/appsutil/bin/txkCfgUtlfileDir.pl
-contextfile=${ORACLE_HOME}/appsutil/${CONTEXT_NAME}.xml -oraclehome=$
{ORACLE_HOME} -outdir=${ORACLE_HOME}/appsutil/log -mode=setUtlFileDir <<-EOF1
${APPS_PASSWD}
${SYSTEM_PASSWD}
EOF1
"
```

```
"${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/$
{CONTEXT_NAME}.env; perl ${ORACLE_HOME}/appsutil/bin/txkCfgUtlfileDir.pl
-contextfile=${ORACLE_HOME}/appsutil/${CONTEXT_NAME}.xml -oraclehome=$
{ORACLE_HOME} -outdir=${ORACLE_HOME}/appsutil/log -mode=createDirObject <<-EOF2
${APPS_PASSWD}
${SYSTEM_PASSWD}
${DIR_OBJ_PATH}
EOF2
"

"${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/$
{CONTEXT_NAME}.env; perl ${ORACLE_HOME}/appsutil/bin/txkCfgUtlfileDir.pl
-contextfile=${ORACLE_HOME}/appsutil/${CONTEXT_NAME}.xml -oraclehome=$
{ORACLE_HOME} -outdir=${ORACLE_HOME}/appsutil/log -mode=syncUtlFileDir
-skipautoconfig=yes <<-EOF3
${APPS_PASSWD}
EOF3
"
```

20. Configure Clone hook to ensure that the ebs_<*PDB_SID*> listener services are not disappeared at the stop or start of the VDB,as shown in the script below
    Configure clone third hook for 19c multi-tenant high privileged and low privileged users

```
#!/usr/bin/env bash
#
# Copyright (c) 2022 by Delphix. All rights reserved.
#
# shellcheck source=/dev/null
HOSTNAME=$(uname -n | cut -d '.' -f1)
. "${ORACLE_HOME}/${ORACLE_SID}_${HOSTNAME}.env";
sqlplus "/ as sysdba" <<-EOF
alter pluggable database ${DELPHIX_PDB_NAME} close;
alter pluggable database ${DELPHIX_PDB_NAME} open read write services=all;
alter pluggable database ${DELPHIX_PDB_NAME} save state;
EOF
```

21. Set up a Pre-Snapshot hook **Run Bash Shell Command** operation to run any pre-clone steps necessary and specific to your EBS database for a **high privileged user**, as shown in the script below.
    Normally, these steps will include running Oracle's `adpreclone` tool.
    Pre-snapshot hook for a 19c multi-tenant high privileged user

```
#!/usr/bin/env bash
#
# Copyright (c) 2022 by Delphix. All rights reserved.
#
# shellcheck source=/dev/null
# As grep has to check all the matches in making decision on L58
# shellcheck disable=SC2143
# NOTE: Ensure the below environment variables will be set up correctly by
# the shell. If not, hardcode or generate the values below.
```

```
HOSTNAME=$(uname -n | cut -d '.' -f1)
CONTEXT_NAME=${DELPHIX_PDB_NAME}_${HOSTNAME}
SOURCE_APPS_PASSWD=$DLPX_SOURCE_APPS_PASSWORD_PASSWORD
TARGET_APPS_PASSWD=$DLPX_TARGET_APPS_PASSWORD_PASSWORD
timeout=3600
waittime=0
ORACLE_USER="oravis"

# Query active PDB services
. "${ORACLE_HOME}/${ORACLE_SID}_${HOSTNAME}.env";
check_value=$(sqlplus -s "/ as sysdba" <<EOF
SELECT NAME FROM v\$active_services;
exit;
EOF
)

checkService() {
if ! echo "$1" | grep -q "$2"; then
. "${ORACLE_HOME}/${ORACLE_SID}_${HOSTNAME}.env";
sqlplus -s "/ as sysdba" <<EOF
alter session set container="${DELPHIX_PDB_NAME}";
BEGIN DBMS_SERVICE.START_SERVICE( service_name => '$2');
end;
/
EOF
fi
}

# Check for ebs_PDB service
checkService "$check_value" "ebs_${DELPHIX_PDB_NAME}"
# Check for PDB_ebs_patch service
checkService "$check_value" "${DELPHIX_PDB_NAME}_ebs_patch"

. "${ORACLE_HOME}/${CONTEXT_NAME}.env";

testAppsPassword() {
    local passwordInQuestion=$1
    ERROR=$(sqlplus "apps/${passwordInQuestion}@${DELPHIX_PDB_NAME}" <<<
"exit;")
    grep ORA-01017 <<< "${ERROR}" >/dev/null && return 1
    return 0
}

testAppsPassword "${SOURCE_APPS_PASSWD}"
testResult=$?

if [[ ${testResult} == 0 ]]; then
    APPS_PASSWD="${SOURCE_APPS_PASSWD}"
else
    APPS_PASSWD="${TARGET_APPS_PASSWD}"
fi
```

```
while [[ -f "${ORACLE_HOME}/.delphix_adpreclone.lck" || $(pgrep -au "$
{ORACLE_USER}" | grep "^${ORACLE_HOME}" | grep -v grep | grep "adpreclone") ]]
; do
    if [[ $waittime -gt $timeout ]]; then
        echo "Another adpreclone process is still running from last 60
mins.Delphix cannot proceed until it is complete. Exiting Now."
        exit 1
    fi
    echo " Another adpreclone process is running. waiting for the process to
complete before starting adpreclone of the database...."
    (( timeout += 10 ))
    sleep 10
done
echo "No other adpreclone process is running on database, proceeding ...."
"${ORACLE_HOME}"/perl/bin/perl "${ORACLE_HOME}/appsutil/scripts/$
{CONTEXT_NAME}/adpreclone.pl" dbTier <<-EOF
${APPS_PASSWD}
EOF
```

Pre-Snapshot hook **Run Bash Shell Command** operation to run any pre-clone steps necessary and specific to your EBS database for a **low privileged user**, as shown in the script below.

${ORACLE_SID}_source_apps_file.txt will be created automatically inside the /var/tmp/ location of the target database host once the DBTechStack provisioning is completed.

Pre-snapshot hook for a 19c multi-tenant low privileged user

```
#!/usr/bin/env bash
#
# Copyright (c) 2022 by Delphix. All rights reserved.
#
# shellcheck source=/dev/null
# As grep has to check all the matches in making decision on L58
# shellcheck disable=SC2143
# NOTE: Ensure the below environment variables will be set up correctly by the
# shell. If not, hardcode or generate the values below.
DLPX_DB_EXEC_SCRIPT="<Remote BIN location till dlpx_db_exec script>"
DLPX_PRIV_USER=oravis
HOSTNAME=$(uname -n | cut -d '.' -f1)
CONTEXT_NAME=${DELPHIX_PDB_NAME}_${HOSTNAME}
SOURCE_APPS_PASSWD=$DLPX_SOURCE_APPS_PASSWORD_PASSWORD
TARGET_APPS_PASSWD=$DLPX_TARGET_APPS_PASSWORD_PASSWORD
timeout=3600
waittime=0

# Query active PDB services
. "${ORACLE_HOME}/${ORACLE_SID}_${HOSTNAME}.env";
check_value=$(sqlplus -s "/ as sysdba" <<EOF
SELECT NAME FROM v\$active_services;
exit;
EOF
)
```

```
checkService() {
if ! echo "$1" | grep -q "$2"; then
. "${ORACLE_HOME}/${ORACLE_SID}_${HOSTNAME}.env";
sqlplus -s "/ as sysdba" <<EOF
alter session set container="${DELPHIX_PDB_NAME}";
BEGIN DBMS_SERVICE.START_SERVICE( service_name => '$2');
end;
/
EOF
fi
}

# Check for ebs_PDB service
checkService "$check_value" "ebs_${DELPHIX_PDB_NAME}"
# Check for PDB_ebs_patch service
checkService "$check_value" "${DELPHIX_PDB_NAME}_ebs_patch"

. "${ORACLE_HOME}/${CONTEXT_NAME}.env";
testAppsPassword() {
    local passwordInQuestion=$1
    ERROR=$("${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/$
{CONTEXT_NAME}.env; sqlplus apps/\"${passwordInQuestion}\"@${DELPHIX_PDB_NAME} <<<
\"exit;\"")
    grep ORA-01017 <<< "${ERROR}" >/dev/null && return 1
    return 0
}
testAppsPassword "${SOURCE_APPS_PASSWD}"
testResult=$?

if [[ ${testResult} == 0 ]]; then
    APPS_PASSWD=${SOURCE_APPS_PASSWD}
else
    APPS_PASSWD=${TARGET_APPS_PASSWD}
fi
while [[ -f "${ORACLE_HOME}/.delphix_adpreclone.lck" || $(pgrep -au "$
{DLPX_PRIV_USER}" | grep "^${ORACLE_HOME}" | grep -v grep | grep "adpreclone") ]] ;
do
    if [[ $waittime -gt $timeout ]]; then
        echo "Another adpreclone process is still running from last 60 mins.Delphix
cannot proceed until it is complete. Exiting Now."
        exit 1
    fi
    echo " Another adpreclone process is running. waiting for the process to complete
before starting adpreclone of the database...."
    (( timeout += 10 ))
    sleep 10
done
echo "No other adpreclone process is running on database, proceeding ...."
${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/${CONTEXT_NAME}.env;
${ORACLE_HOME}/perl/bin/perl ${ORACLE_HOME}/appsutil/scripts/${CONTEXT_NAME}/
adpreclone.pl dbTier <<-EOF
```

```
${APPS_PASSWD}
EOF
"
```

Provisioning the Oracle 12C database

1. Refresh the Target DB environment from **Manage > Environments** so that the Listener brought up during dbTechStack provision gets discovered and the Installation Home to be listed in dropdown during Database provision. If Target DB environment is not refreshed, you will see `This environment has no compatible Oracle Installation Homes` in the Installation Home dropdown in the **Database provision**.
2. Provision the EBS database to the target dbTier environment by following the steps outlined in Provisioning an Oracle VDB. For Oracle EBS database for 19c MT, provision the database by following the steps outlined in Provisioning an Oracle Virtual Pluggable Database (vPDB)
   Note: When Snapshot is running against the dbTechStack, database, or appsTier, the Delphix Engine also executes pre-clone logic to ensure the latest configuration is staged in the captured snapshots. Unfortunately, if multiple Snapshots are running against the same EBS instance concurrently, this pre-clone logic may fail and produce bad snapshots.
   To avoid SnapSync conflicts, spread out your SnapSync policies for an EBS instance by one hour or more.
3. Select the correct **Installation Home**.
   This should be the virtual dbTechStack you just added to the Delphix Engine.
4. Select an Environment User.
5. This user should be the oracle user-outlined in Preparing Target EBS R12.2 Environments for Provisioning.
6. Select a Target Group for the VDB.
7. In the VDB **Configuration** section, provide Target DB/CDB SID name provided during dbTechStack provisioning in the **Oracle Database Name** and **Oracle SID** field. Provide any unique user-defined name in **VDB Name** and **Oracle Database Unique Name** and click **Next**.
8. Click **Advanced**.
9. Select the correct **Oracle Node Listeners** value.
   This should be the listener corresponding to the virtual dbTechStack you just added to the Delphix Engine.
10. Add the EBS R12.2 dbTier environment file as a **Custom Environment Variables** entry.
11. This file can be specified as an Environment File with Path Parameters of `$ORACLE_HOME/<CONTEXT_NAME>` .
    Replace `<CONTEXT_NAME>` with the virtual EBS instance's context name. The Delphix Engine will expand the `$ORACLE_HOME` variable at runtime.
    For more information, see Customizing Oracle VDB Environment Variables. **Note:** To complete the DB Provisioning operation, you must provide content to configure clones and pre-snapshot hooks. You must either follow step 12 to use the hooks utility feature OR proceed with the conventional way by following steps 13-20.
12. To complete the DB Provisioning operation, you must provide content to configure clones and pre-snapshot hooks. The following are the content for both the hooks:
    a. Configure Clone hook will be `<DBTechStack Mount point>/hooksUtil/hooksRunner --operation configure` . You can set a hooks environment variable `DLPX_SOURCE_APPS_PASSWORD` for providing the source apps schema password. In case, you are not sure about the source apps schema password, then you should skip the entire step12.
    b. Pre-Snapshot hook : `<DBTechStack Mount point>/hooksUtil/hooksRunner --operation pre-snapshot` . You can set a hooks environment variable `DLPX_SOURCE_APPS_PASSWORD` for providing the source apps schema password. In case of apps password change in target VDB, you can set hook environment variables

`DLPX_TARGET_APPS_PASSWORD` and `DLPX_SYS_PASSWORD` for providing the target apps schema and system schema passwords.

13. Add a **Run Bash Shell Command** operation to the Configure Clone hook to ensure that `adcfgclone` is run against the newly provisioned database for a **high privileged user**, as shown in the script below.
Note: You must set a hooks environment variable `DLPX_SOURCE_APPS_PASSWORD` for providing the source apps schema password. In case the apps password change is required in target VDB, you can set hook environment variables `DLPX_TARGET_APPS_PASSWORD` and `DLPX_SYS_PASSWORD` for providing the target apps schema and system schema passwords.
Configure clone first hook for a high privileged user

```
#!/usr/bin/env bash
#
# Copyright (c) 2022 by Delphix. All rights reserved.
#
# shellcheck source=/dev/null
# NOTE: Ensure the below environment variables will be set up correctly by
# the shell. If not, hardcode or generate the values below.
HOSTNAME=$(uname -n | cut -d '.' -f1)
CONTEXT_NAME=${ORACLE_SID}_${HOSTNAME}
APPS_PASSWD=$DLPX_SOURCE_APPS_PASSWORD_PASSWORD

. "${ORACLE_HOME}/${CONTEXT_NAME}.env";
# Check for local_listener parameter is set for SID, otherwise set it
# appropriately
check_value=$(sqlplus -s "/ as sysdba" <<EOF
show parameter local_listener;
EOF
)
local_listener=$(echo "$check_value" | awk '{print $11}')
value=("${local_listener//:/ }")
host="${value[0]}"
port="${value[1]}"
curr_port=$(grep PORT < "${TNS_ADMIN}/listener.ora" | awk '{print $9}' | sed
's)//g')

if [[ $port != "$curr_port" || $host != "${HOSTNAME}" ]]; then
sqlplus -s "/ as sysdba" <<EOF
alter system set local_listener='${HOSTNAME}:${curr_port}';
alter system register;
EOF
fi

sqlplus "/ as sysdba" <<EOF
@${ORACLE_HOME}/appsutil/install/${CONTEXT_NAME}/adupdlib.sql so
EOF

perl "${ORACLE_HOME}/appsutil/clone/bin/adcfgclone.pl" dbconfig "$
{ORACLE_HOME}/appsutil/${CONTEXT_NAME}.xml" <<EOF
${APPS_PASSWD}
EOF
```

14. Configure Clone hook to ensure that `adcfgclone` is run against the newly provisioned database for a **low privileged user**, as shown in the script below.Customers have to identify the DLPX_DB_EXEC_SCRIPT value from their remote environment, this would always be inside the toolkit directory provided while adding the environment on Delphix Engine.
Configure clone first hook for a low privileged user

```
#!/usr/bin/env bash
#
# Copyright (c) 2022 by Delphix. All rights reserved.
#
# shellcheck source=/dev/null
# NOTE: Ensure the below environment variables will be set up correctly by the
# shell. If not, hardcode or generate the values below.

DLPX_DB_EXEC_SCRIPT="<Remote BIN location for dlpx_db_exec script>"
DLPX_PRIV_USER=oravis
APPS_PASSWD=$DLPX_SOURCE_APPS_PASSWORD_PASSWORD
HOSTNAME=$(uname -n | cut -d '.' -f1)
CONTEXT_NAME=${ORACLE_SID}_${HOSTNAME}

. "${ORACLE_HOME}/${CONTEXT_NAME}.env";
# Check for local_listener parameter is set for SID, otherwise set it
# appropriately
check_value=$(sqlplus -s "/ as sysdba" <<EOF
show parameter local_listener;
EOF
)
local_listener=$(echo "$check_value" | awk '{print $11}')
value=("${local_listener//:/ }")
host="${value[0]}"
port="${value[1]}"
curr_port=$(grep PORT < "${TNS_ADMIN}/listener.ora" | awk '{print $9}' | sed
's/)//g')

if [[ $port != "$curr_port" || $host != "${HOSTNAME}" ]]; then
sqlplus -s "/ as sysdba" <<EOF
alter system set local_listener='${HOSTNAME}:${curr_port}';
alter system register;
EOF
fi

"${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/$
{CONTEXT_NAME}.env; sqlplus \"/ as sysdba\" <<-EOF
@${ORACLE_HOME}/appsutil/install/${CONTEXT_NAME}/adupdlib.sql so
EOF
"

"${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/$
{CONTEXT_NAME}.env; perl ${ORACLE_HOME}/appsutil/clone/bin/adcfgclone.pl
dbconfig ${ORACLE_HOME}/appsutil/${CONTEXT_NAME}.xml <<-EOF1
${APPS_PASSWD}
```

```
EOF1
"
```

15. For the EBS database Oracle 12.2, add a **Run Bash Shell Command** operation to the Configure Clone hook to ensure that `sqlnet.ora` or `sqlnet_ifile.ora` specify a value for `SQLNET.ALLOWED_LOGON_VERSION_SERVER` for a **high privileged user**, as shown in the script below. This requirement is outlined in **Cloning Oracle E-Business Suite Release 12.2 with Rapid Clone (Doc ID 1383621.1)** found at http://docs.oracle.com
Configure clone second hook for a high privileged user

```
#!/usr/bin/env bash
# Copyright (c) 2023 by Delphix. All rights reserved.
#
# NOTE: Ensure the below environment variables will be set up correctly by
# the shell. If not, hardcode or generate the values below.
# If you are using sqlnet_ifile.ora, change the script below to reflect
# sqlnet_ifile.ora Initialize dummy variable `parameter` to avoid shellcheck
# to fail
parameter=""
CONTEXT_NAME=${ORACLE_SID}_$(uname -n | cut -d '.' -f1)
. "${ORACLE_HOME}/${CONTEXT_NAME}.env";
TNS_ADMIN=${ORACLE_HOME}/network/admin/${CONTEXT_NAME}
FILE=${TNS_ADMIN}/sqlnet.ora

if [[ ! -f ${FILE} ]];  then
     {
    echo "#Creating sqlnet.ora for Delphix hooks"
    echo "################################################################"
    echo "NAMES.DIRECTORY_PATH=(TNSNAMES, ONAMES, HOSTNAME)"
    echo "SQLNET.EXPIRE_TIME= 10"
    echo "SQLNET.INBOUND_CONNECT_TIMEOUT =60"
    echo "SQLNET.ALLOWED_LOGON_VERSION_SERVER=8"
    } >> "${TNS_ADMIN}/sqlnet.ora"
fi
echo "sqlnet.ora is created in location ${TNS_ADMIN}"

check_value=$(sqlplus -s "/ as sysdba" <<EOF
set head off termout off feedback off wrap off
select DISPLAY_VALUE from v\\$parameter where NAME='sec_case_sensitive_logon';
EOF
)

if [[ ${check_value} = "FALSE" ]]; then
        sed 's/^SQLNET.ALLOWED_LOGON_VERSION_SERVER=.*/
SQLNET.ALLOWED_LOGON_VERSION_SERVER=8/g' "${TNS_ADMIN}/sqlnet.ora" > /var/tmp/
temp.ora && mv /var/tmp/temp.ora "${TNS_ADMIN}/sqlnet.ora"
elif [[ ${check_value} = "TRUE" ]]; then
        sed 's/^SQLNET.ALLOWED_LOGON_VERSION_SERVER=.*/
SQLNET.ALLOWED_LOGON_VERSION_SERVER=10/g' "${TNS_ADMIN}/sqlnet.ora" > /var/tmp/
temp.ora && mv /var/tmp/temp.ora "${TNS_ADMIN}/sqlnet.ora"
else
```

```
        echo "sec_case_sensitive_logon parameter is not set in the database. So
the sqlnet.ora has not been updated."
fi
```

Copy

16. When prompted, if your environment is on R12.TXK.C.Delta.13 or later, enter the password for the EBS_SYSTEM user, if your environment is on R12.TXK.C.Delta.12 or earlier, enter the password for the SYSTEM user, or if your source EBS setup is running on AD-TXK Delta 13 then you need to supply the EBS_SYSTEM schema password in the below second hook script.
This requirement is outlined in the FAQ: Oracle E-Business Suite and System Schema Migration (DocID 2758999.1) Using UTL_FILE_DIR or Database Directories for PL/SQL File I/O in Oracle E-Business Suite Releases 12.1 and 12.2 (DocID 2525754.1) found at http://docs.oracle.com
17. Configure Clone hook to ensure that `sqlnet.ora` or `sqlnet_ifile.ora` specify a value for `SQLNET.ALLOWED_LOGON_VERSION_SERVER` for a **low privileged user**, as shown in the script below.
Configure clone second hook for a low privileged user

```
#!/usr/bin/env bash
#
# Copyright (c) 2022 by Delphix. All rights reserved.
#
# NOTE: Ensure the below environment variables will be set up correctly by the
# shell. If not, hardcode or generate the values below.
# If you are using sqlnet_ifile.ora, change the script below to reflect
# sqlnet_ifile.ora.
# Initialize dummy variable `parameter` to avoid shellcheck to fail
parameter=""
DLPX_DB_EXEC_SCRIPT="<Remote BIN location for dlpx_db_exec script>"
DLPX_PRIV_USER=oravis
CONTEXT_NAME=${ORACLE_SID}_$(uname -n | cut -d '.' -f1)
TNS_ADMIN=${ORACLE_HOME}/network/admin/${CONTEXT_NAME}

check_value=$("${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/
${CONTEXT_NAME}.env; sqlplus \"/ as sysdba\" <<-EOF
set head off termout off feedback off wrap off
select DISPLAY_VALUE from v\\$parameter where NAME='sec_case_sensitive_logon';
EOF
")

if [[ ${check_value} = "FALSE" ]]; then
    "${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" "sed 's/
^SQLNET.ALLOWED_LOGON_VERSION_SERVER=.*/SQLNET.ALLOWED_LOGON_VERSION_SERVER=8/
g' \"${TNS_ADMIN}/sqlnet.ora\" > /var/tmp/temp.ora && mv /var/tmp/temp.ora \"$
{TNS_ADMIN}/sqlnet.ora\""
elif [[ ${check_value} = "TRUE" ]]; then
    "${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" "sed 's/
^SQLNET.ALLOWED_LOGON_VERSION_SERVER=.*/SQLNET.ALLOWED_LOGON_VERSION_SERVER=10/
g' \"${TNS_ADMIN}/sqlnet.ora\" > /var/tmp/temp.ora && mv /var/tmp/temp.ora \"$
{TNS_ADMIN}/sqlnet.ora\""
else
    echo "sec_case_sensitive_logon parameter is not set in the database. So the
sqlnet.ora has not been updated."
```

```
fi
```

When prompted, if your environment is on R12.TXK.C.Delta.13 or later, enter the password for the EBS_SYSTEM user, if your environment is on R12.TXK.C.Delta.12 or earlier, enter the password for the SYSTEM user, or if your source EBS setup is running on AD-TXK Delta 13 then you need to supply the EBS_SYSTEM schema password in the below second hook script.
This requirement is outlined in the FAQ: Oracle E-Business Suite and System Schema Migration (DocID 2758999.1) Using UTL_FILE_DIR or Database Directories for PL/SQL File I/O in Oracle E-Business Suite Releases 12.1 and 12.2 (DocID 2525754.1) found at http://docs.oracle.com

18. Configure Clone hook to ensure that the ebs_*<PDB_SID>* listener services are not disappeared at the stop or start of the VDB, as shown in the script below.
19. Set up a Pre-Snapshot hook **Run Bash Shell Command** operation to run any pre-clone steps necessary and specific to your EBS database for a **high privileged user**, as shown in the script below.

    Normally, these steps will include running Oracle's `adpreclone` tool.

    Pre-snapshot hook for a high privileged user

```
#!/usr/bin/env bash
#
# Copyright (c) 2023 by Delphix. All rights reserved.
#
# shellcheck source=/dev/null
# As grep has to check all the matches in making decision on L58
# shellcheck disable=SC2143
# NOTE: Ensure the below environment variables will be set up correctly by
# the shell. If not, hardcode or generate the values below.
HOSTNAME=$(uname -n | cut -d '.' -f1)
CONTEXT_NAME=${ORACLE_SID}_${HOSTNAME}
SOURCE_APPS_PASSWD=$(cat "/var/tmp/${ORACLE_SID}_source_apps_file.txt")
TARGET_APPS_PASSWD="<new apps password of the target DB node>"
ORACLE_USER="oravis"

timeout=3600
waittime=0
. "${ORACLE_HOME}/${CONTEXT_NAME}.env";

testAppsPassword() {
    local passwordInQuestion=$1
    ERROR=$(sqlplus "apps/${passwordInQuestion}" <<< "exit;")

    grep ORA-01017 <<< "${ERROR}" >/dev/null && return 1
    return 0

}

testAppsPassword "${SOURCE_APPS_PASSWD}"
testResult=$?

if [[ ${testResult} == 0 ]]; then
    APPS_PASSWD=${SOURCE_APPS_PASSWD}
else
    APPS_PASSWD=${TARGET_APPS_PASSWD}
```

```
fi

# Query active SID services
check_value=$(sqlplus -s "/ as sysdba" <<EOF
SELECT NAME FROM v\$active_services;
exit;
EOF
)
checkService() {
if ! echo "$1" | grep -q "$2"; then
. "${ORACLE_HOME}/${CONTEXT_NAME}.env";
sqlplus -s "/ as sysdba" <<EOF
BEGIN DBMS_SERVICE.START_SERVICE( service_name => '$2');
end;
/
EOF
fi
}
# Check for SID_ebs_patch or ebs_patch service based on the AD-TXK codelevel
# version. In AD-TXK Delta 8 and earlier, the service name for connections
# to the patch edition of the database was ebs_patch. In AD-TXK Delta 9, the
# service name to connect to the patch edition has been changed to
# <instance_name>_ebs_patch
patch_level=$(sqlplus -s "apps/${APPS_PASSWD}" <<EOF
select PATCH_LEVEL from apps.fnd_product_installations where PATCH_LEVEL like
'R12.AD.%';
EOF
)
patch_service="${ORACLE_SID}_ebs_patch"
if [ "${patch_level: -1}" -le 8 ]; then
    patch_service="ebs_patch"
fi
checkService "$check_value" "${patch_service}"


while [[ -f "${ORACLE_HOME}/.delphix_adpreclone.lck" || $(pgrep -au "$
{ORACLE_USER}" | grep "^${ORACLE_HOME}" | grep -v grep |  grep "adpreclone") ]]
 ; do
    if [[ $waittime -gt $timeout  ]]; then
        echo "Another adpreclone process is still running from last 60
mins.Delphix cannot proceed until it is complete. Exiting Now."
        exit 1
    fi
    echo  " Another adpreclone process is running. waiting for the process to
complete before starting adpreclone of the database...."
    (( timeout += 10 ))
    sleep 10
done
echo "No other adpreclone process is running on database, proceeding ...."
"${ORACLE_HOME}"/perl/bin/perl "${ORACLE_HOME}/appsutil/scripts/$
{CONTEXT_NAME}/adpreclone.pl" database <<-EOF
${APPS_PASSWD}
```

```
EOF
```

Copy

20. Pre-Snapshot hook **Run Bash Shell Command** operation to run any pre-clone steps necessary and specific to your EBS database for a **low privileged user**, as shown in the script below.
Pre-snapshot hook for a low privileged user

```
#!/usr/bin/env bash
#
# Copyright (c) 2022 by Delphix. All rights reserved.
#
# shellcheck source=/dev/null
# NOTE: Ensure the below environment variables will be set up correctly by the
# shell. If not, hardcode or generate the values below.
# As grep has to check all the matches in making decision on L58
# shellcheck disable=SC2143
DLPX_DB_EXEC_SCRIPT="<Remote BIN location for dlpx_db_exec script>"
DLPX_PRIV_USER=oravis
HOSTNAME=$(uname -n | cut -d '.' -f1)
CONTEXT_NAME=${ORACLE_SID}_${HOSTNAME}
SOURCE_APPS_PASSWD=$DLPX_SOURCE_APPS_PASSWORD_PASSWORD
TARGET_APPS_PASSWD=$DLPX_TARGET_APPS_PASSWORD_PASSWORD

timeout=3600
waittime=0

testAppsPassword() {
    local passwordInQuestion=$1
    ERROR=$("${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/$
{CONTEXT_NAME}.env; sqlplus apps/\"${passwordInQuestion}\" <<< \"exit;\"")

    grep ORA-01017 <<< "${ERROR}" >/dev/null && return 1
    return 0
}

testAppsPassword "${SOURCE_APPS_PASSWD}"
testResult=$?

if [[ ${testResult} == 0 ]]; then
    APPS_PASSWD=${SOURCE_APPS_PASSWD}
else
    APPS_PASSWD=${TARGET_APPS_PASSWD}
fi

# Query active SID services
. "${ORACLE_HOME}/${CONTEXT_NAME}.env";
check_value=$(sqlplus -s "/ as sysdba" <<EOF
SELECT NAME FROM v\$active_services;
exit;
EOF
)
checkService() {
```

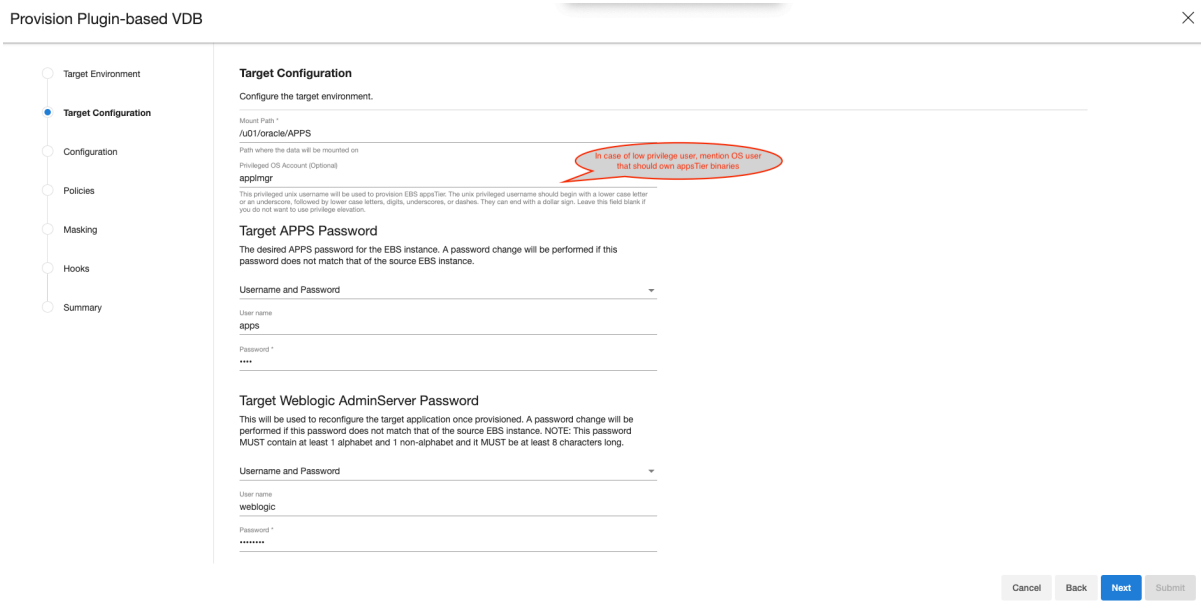
```
if ! echo "$1" | grep -q "$2"; then
. "${ORACLE_HOME}/${CONTEXT_NAME}.env";
sqlplus -s "/ as sysdba" <<EOF
BEGIN DBMS_SERVICE.START_SERVICE( service_name => '$2');
end;
/
EOF
fi
}
# Check for SID_ebs_patch or ebs_patch service based on the AD-TXK codelevel
# version. In AD-TXK Delta 8 and earlier, the service name for connections
# to the patch edition of the database was ebs_patch. In AD-TXK Delta 9, the
# service name to connect to the patch edition has been changed to
# <instance_name>_ebs_patch
patch_level=$(sqlplus -s "apps/${APPS_PASSWD}" <<EOF
select PATCH_LEVEL from apps.fnd_product_installations where PATCH_LEVEL like
'R12.AD.%';
EOF
)
patch_service="${ORACLE_SID}_ebs_patch"
if [ "${patch_level: -1}" -le 8 ]; then
    patch_service="ebs_patch"
fi
checkService "$check_value" "${patch_service}"

while [[ -f "${ORACLE_HOME}/.delphix_adpreclone.lck" || $(pgrep -au "$
{DLPX_PRIV_USER}" | grep "^${ORACLE_HOME}" | grep -v grep |  grep "adpreclone")
 ]] ; do
    if [[ $waittime -gt $timeout  ]]; then
        echo "Another adpreclone process is still running from last 60
mins.Delphix cannot proceed until it is complete. Exiting Now."
        exit 1
    fi
    echo  " Another adpreclone process is running. waiting for the process to
complete before starting adpreclone of the database...."
    (( timeout += 10 ))
    sleep 10
done
echo "No other adpreclone process is running on database, proceeding ...."
"${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/$
{CONTEXT_NAME}.env; ${ORACLE_HOME}/perl/bin/perl ${ORACLE_HOME}/appsutil/
scripts/${CONTEXT_NAME}/adpreclone.pl database <<-EOF
${APPS_PASSWD}
EOF
"
```

${ORACLE_SID}_source_apps_file.txt will be created automatically inside the /var/tmp/ location of the target database host once the DBTechStack provisioning is completed.

Provisioning the EBS appsTier

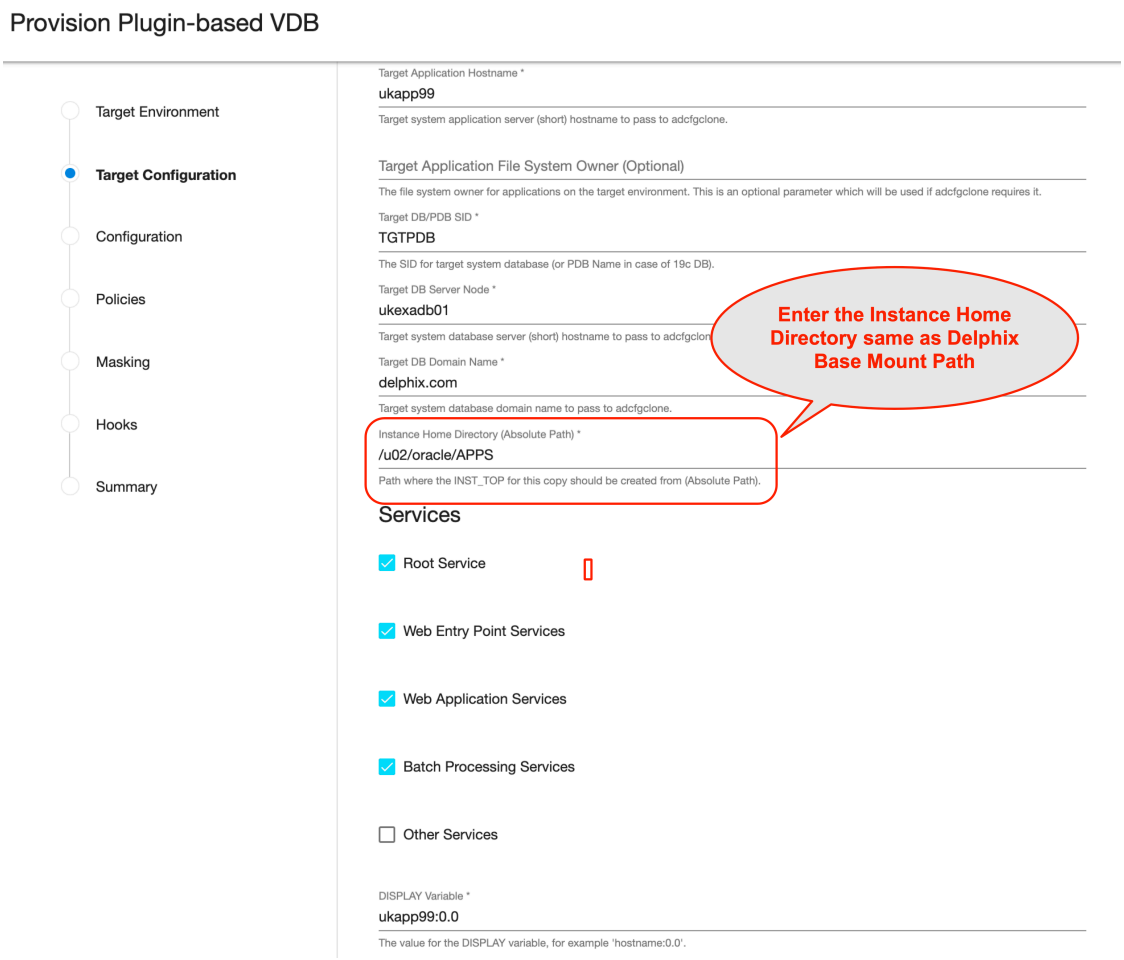
1. Login to the **Delphix Management** application using **Admin** credentials.
2. Click **Manage**.
3. Select **Datasets**.

4. Select the **appsTier dSource**.
5. Select a dSource **snapshot**.
   All snapshots will have staged configuration prepared by `adpreclone.pl` and any hook operations placed on the dSource.
6. Click **Provision**.
   The **Provision VDB** wizard will open.
7. Select an **Environment.**
   This environment will host the virtual appsTier and be used to execute hook operations specified in a few steps. This environment will also run the WebLogic Admin server (Web Administration service) for the virtual appsTier.
   If you are provisioning a multi-node appsTier, you will be able to specify additional environments to host the virtual appsTier in a few steps.
8. Select E-Business Suite R12.2 appsTier from the **Installation** dropdown.
9. Select an Environment User.
   This user should be the `applmgr` user-outlined in Preparing Target EBS R12.2 Environments for Provisioning
10. Enter a **Mount Path** for the virtual appsTier VDB.If you are provisioning a multi-node appsTier, this mount path will be used across all target environments.
11. Enter the **EBS-specific parameters** for the virtual appsTier. A subset of these parameters are discussed in more detail below.



   a. **Privileged OS Account (Optional)** field should contain high privileged user when low privileged user is being used for provisioning.
   b. The **Target APPS Password** is the new apps password that is required to configure and manage the virtual appsTier.
      This password is encrypted when stored within the Delphix Engine and is available as an environment variable to the `adcfgclone`, `adstrtal`, and `adstpall` processes.
   c. The **Weblogic AdminServer Password** is the new WebLogic password required to configure the virtual appsTier. A password change will be performed if this password does not match that of the source EBS instance. This password is encrypted when stored within the Delphix Engine and is available as an environment variable to the adcfgclone, adstrtal, and adstpall processes.
   d. The **SYSTEM Password** is required to configure the virtual appsTier with the new apps password. This password is not required if the Target Apps Password is the same as the source.

e. Provide the Target Application hostname.
f. The **Target DB/PDB SID** is the new database SID (PDB SID in case of 19c) that is required to configure and manage the virtual appsTier.
g. Provide the Target DB server node hostname.
h. Provide **Target DB Domain Name**, this will be provided to the ***adcfgclone.***
i. Delphix recommends specifying an **Instance Home Directory** under the **Mount Path** so that instance-specific EBS files live on Delphix-provided storage.

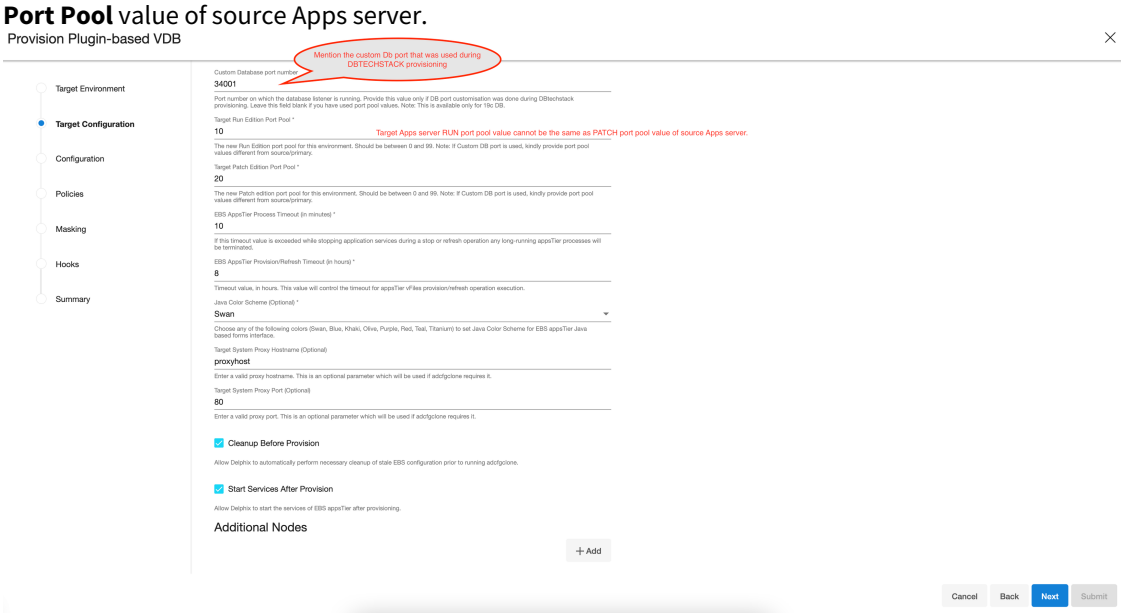


For example, if the provided **Mount Path** is `/u01/oracle/VIS`, then providing an **Instance Home Directory** of `/u01/oracle/VIS` would allow EBS to generate virtual application INST_TOP in `/u01/oracle/VIS/fs1/inst/apps/<CONTEXT_NAME>` and `/u01/oracle/VIS/fs2/inst/apps/<CONTEXT_NAME>`.

   i. If you are provisioning a single-node appsTier, this recommendation is OPTIONAL; putting instance-specific EBS files on Delphix-provided storage merely eases the administration of the virtual EBS instance.
   ii. If you are provisioning a multi-node appsTier, this recommendation is REQUIRED; the Delphix Engine's automation requires that all nodes in the appsTier have access to instance-specific files via Delphix-provided storage.

j. Ensure that the **Target Application Hostname** and **Target DB Server Node** values are the short hostnames, not the fully-qualified hostnames.

k. Select **Services**, by default all the ***Root***, ***Web Entry Point***, ***Web Application*** and ***Batch Processing*** would be checked. In case you are using multinode environment and want to start a few services on a secondary node then this field is going to be very useful for you.
l. Provide **DISPLAY Variable**, the default value is hostname:0.0
m. [Optional] Provide the **Custom Database Port Number** on which database listener must run. **Note:** Enter the numeric port value in this field only if you want to use DB port customisation, and this value should match with the custom database port number given during the DBTechStack provisioning. Leaving this field blank automatically uses the default port based on port pool. The custom database port number is applicable only for Oracle 19c database.
n. Provide **Target Run Edition Port Pool**, the value of this field should be 0 to 99. This should match with the port pool value given during the DBTechStack provisioning.
o. Provide **Target Patch Edition Port Pool**, the value of this field should be 0 to 99. This value should be Target Run Edition Port Pool plus 1. For example, if Run Edition Port Pool is 9 then the value should be 10. **Note:** Target Apps server **RUN Edition Port Pool** value cannot be the same as **PATCH Edition Port Pool** value of source Apps server.



p. The **EBS AppsTier Timeout** is required to terminate all the long-running appsTier processes which have exceeded the timeout value when stopping the applications as part of a refresh. This timeout will be calculated in minutes. For example, if set to 30, then we run adstpall at the start of the refresh, and if after 30 minutes the application has not stopped, then the processes will be terminated to allow the refresh to continue.
q. The **EBS AppsTier Provision/Refresh Timeout** is required to terminate configure and cloning via `adcfgclone` process which have exceeded the timeout as part of a provision/refresh. This timeout will be calculated in hours and the default value will be 8 hours. For example, if set to 9, then plugin will detect a timeout after 9 hours instead of getting in a hung state and UI error for the same will be prompted.
r. The **Java Color Scheme** is an **optional** parameter and is used to change the JAVA color for provisioned Oracle EBS JAVA based form interface. Please choose any of the following colors (Swan, Blue, Khaki, Olive, Purple, Red, Teal, Titanium) to set Java Color Scheme for EBS appsTier java-based forms interface.
s. Provide **Target System Proxy Hostname (Optional)**, This is an optional parameter that will be used if ***adcfgclone*** requires it.
t. Provide **Target System Proxy Port (Optional)**, This is an optional parameter that will be used if ***adcfgclone*** requires it.

u. Enable the **Cleanup Before Provision**option if you want to permit the Delphix Engine to automatically cleanup stale EBS configuration during a refresh. This option is recommended, but only available if your Oracle Home is patched with Oracle Universal Installer (OUI) version 10.2 or above.
   i. With this option enabled, the Delphix Engine will inspect the target environment's oraInventory prior to refreshing this virtual appsTier. If any Oracle Homes are already registered within the specified **Mount Path**, the Delphix Engine will detach them from the inventory prior to running `adcfgclone`. These homes must be detached prior to running post-clone configuration. If they are not detached, `adcfgclone` will fail, citing conflicting oraInventory entries as an issue. The Delphix Engine will also remove any conflicting INST_TOP directories left on the environment. Non-conflicting INST_TOP directories will not be modified.
   ii. Without this option enabled, Oracle Homes or INST_TOP directories that conflict with the specified **Mount Path** or desired INST_TOP location will be reported in errors instead of automatically cleaned up. For refresh to succeed, you must manually detach conflicting Oracle Homes and manually remove conflicting INST_TOP directories prior to refresh.

v. Enable the **Start Services After Provision** option if you want to permit the Delphix Engine to automatically start the services for EBS appsTier after provisioning.
   i. With this option enabled, the Delphix Engine will start all the services of EBS appsTier after provisioning.
   ii. Without this option enabled, the Delphix Engine will **NOT** start the services of EBS appsTier after provisioning. This will allow customers to prevent EBS services from being started at the conclusion of provisioning or refreshing. That way, customers can perform post-clone processing automation using a configure-clone hook without having to stop services first.

w. If you are provisioning a multi-node appsTier, enter additional appsTier nodes as **Additional Nodes**.
   i. Select the Environment for the secondary node.
   ii. The **Environment User** for each node should be the `applmgr` user-outlined in Preparing Target EBS R12.2 Environments for Provisioning
   iii. Ensure that the **Hostname** value for each node is the short hostname, not the fully-qualified hostname.
   iv. Provide DISPLAY Variable, the default value is hostname:0.0.
   v. The **Mount Path** is not configurable for each node individually. The **Mount Path** provided for the primary environment will be used for each additional node.

12. Click **Next**.
13. Enter a **VDB Name**.
14. Select a **Target Group** for the VDB.
    If necessary, click the **Plus** icon to add a new group.
15. Select a **Snapshot Policy** for the VDB.
    If necessary, click the **Plus** icon to create a new policy.
    Note:
    When Snapshot is running against the dbTechStack, database, or appsTier, the Delphix Engine also executes pre-clone logic to ensure the latest configuration is staged in the captured snapshots. Unfortunately, if multiple Snapshots are running against the same EBS instance concurrently, this pre-clone logic may fail and produce bad snapshots.
    To avoid SnapSync conflicts, spread out your SnapSync policies for an EBS instance by one hour or more.
    Click **Next**.
16. Enter any **custom hook operations** that are needed to help correctly manage the virtual appsTier.
    The Configure Clone hook will be run after the `adcfgclone.pl` tool has both mounted and configured the appsTier.
    All hook operations run against the environment specified for provision. For a multi-node appsTier, hook operations never run against additional nodes specified.

17. Click **Next.**
18. Click **Submit**.
    Warning: dbTier must be accessible during appsTier provisioning
    Info: Post-clone configuration will fail if the appsTier cannot connect to the database. Ensure the target dbTier is accessible to the appsTier during the provisioning process. Ensure both the virtual database and database listener are running.

    Note: If your source setup is EBS ISG enabled, then during virtual AppsTier creation on the target server, perform post cloning steps as described in Installing Oracle E-Business Suite Integrated SOA Gateway, Release 12.2 (Doc ID 1311068.1).
    When provisioning starts, you can review the progress of the job in the **Datasets** panel, or in the **Job History** panel of the **Dashboard**. When provisioning is complete, the appsTier VDB will be included in the group you designated and listed in the **Datasets** panel. If you select the appsTier VDB in the **Datasets** panel and click the **Configuration** tab, you can view information about the virtual files and its Data Management settings.
19. For tips on monitoring the progress of appsTier provisioning, see Monitoring EBS R12.2 appsTier Provisioning Progress
    Once all three EBS virtual datasets have been provisioned successfully, your virtual EBS instance should be running and accessible.

Rewind operation on snapshot
Perform the rewind operation on the snapshot after Appstier provisioning in the following sequence.

1. Take a snapshot of the Database VDB after Appstier provision.
2. Stop the appsTier, DB, and dbTechStack one by one.
3. Rewind dbTechStack, DB, and appsTier one by one.

If the DB snapshot was taken before appsTier provisioning, rewinding the DB will cause loss of new data updated by appsTier during provisioning and the next operation of appsTier rewind will fail. So, to ensure that the Rewind operation of appsTier succeeds, take a snapshot of DB after Appstier provision.

Registering the EBS dbTechStack

Register the freshly-provisioned dbTechStack with the Delphix Engine.

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Select the **target dbTier environment**.
5. Click the **Databases** tab.
6. Click the **Plus** icon next to **Add Dataset Home**.
7. Select **Oracle** as your **Dataset Home Type**.
8. Enter an **Installation Home**.
   This path should be the value of `$ORACLE_HOME` on your target dbTier; this path will live under the mount path specified when you provisioned the virtual dbTechStack.
   Commonly, this path looks like `/u01/oracle/VIS/11.2.0`.
9. Click the **checkmark** to save your dataset home.
   If necessary, scroll down the list of dataset homes to view and edit this dataset home.
10. Refresh the dbTier environment.
    Refreshing the environment will discover an EBS database listener and ensure it is available for selection when provisioning the EBS database.
    a. Click the **Refresh** button in the top right-hand corner of the environment card.

Monitoring EBS R12.2 appsTier provisioning progress

This topic describes how to monitor the progress of EBS R12.2 appsTier provisioning.

The Delphix Engine automates the configuration of the appsTier during provisioning. It spends the majority of provisioning time running RapidClone utilities such as `adcfgclone`. The Delphix Engine does not report the progress of RapidClone utilities through the progress of the **DB_Provision** job. To track provisioning progress more accurately, Delphix recommends monitoring EBS log files generated on the target appsTier environment.

Procedure

1. Connect to the target appsTier environment using SSH or an alternative utility.
2. Change directories to the < `INST_TOP>/admin/log/` .
    - Replace `<INST_TOP>` with the value of INST_TOP on the virtual EBS instance.
3. After `adcfgclone` has begun running, a file matching `ApplyAppsTier_*.log` will exist. Identify this log file and use `tail` or an equivalent utility to monitor it.

> **Leave the Mount Point Prior to Refresh**
> If provisioning fails, you will need to fix the cause of the failure and perform a refresh of the dataset before you can attempt configuration again. Prior to refreshing or disabling the dataset, be sure to change directories to a location outside of the mount path on the target environment. If you leave a shell session with a current working directory inside the mount path, the dataset will fail to unmount cleanly during a refresh or disable.

Monitoring EBS R12.2 dbTechStack provisioning progress

This topic describes how to monitor the progress of EBS R12.2 dbTechStack provisioning.

The Delphix Engine automates the configuration of the dbTechStack during provisioning. It spends the majority of provisioning time running RapidClone utilities such as `adcfgclone`. The Delphix Engine does not report the progress of RapidClone utilities through the progress of the **DB_Provision** job. To track provisioning progress more accurately, Delphix recommends monitoring EBS log files generated on the target dbTier environment.

Procedure

1. Connect to the target dbTier environment using SSH or an alternative utility.
2. Change directories to the < `ORACLE_HOME>/appsutil/log/<CONTEXT_NAME>/`
    - Replace `<ORACLE_HOME>` with the path to the dbTechStack's Oracle Home. This path will be under the mount path specified during provisioning.
    - Replace `<CONTEXT_NAME>` with the virtual EBS instance's context name.
3. After `adcfgclone` has begun running, a file matching `ApplyDBTechStack_*.log` will exist. Identify this log file and use `tail` or an equivalent utility to monitor it.

> **Leave the mount point prior to refresh**
> If provisioning fails, you will need to fix the cause of the failure and perform a refresh of the dataset before you can attempt configuration again. Prior to refreshing or disabling the dataset, be sure to change directories to a location outside of the mount path on the target environment. If you leave a shell session with a current working directory inside the mount path, the dataset will fail to unmount cleanly during a refresh or disable.

Linking & provisioning TDE Enabled (19c) Oracle EBS PDB

Pre-requisites

1. **Oracle: Multi-Tenant TDE for VCDBs**
   Make sure to use transparent data encryption (TDE) with virtual container databases to secure data-at-rest. Existing TDE keys can be used or data can be rekeyed upon deployment. Automated KeyStore sanitization ensures production TDE keys are not shared with non-production environments. This workflow allows you to automate the Oracle data lifecycle end-to-end, which results in developer productivity enhancements.
2. For a cluster target, it is recommended to place auto-login keystore on the ACFS storage instead of the Delphix mounted storage.
3. The Oracle EBS database version must be 19c (12c, 11g is not supported).
4. With 7.0.0.0 release, existing dSources can now have encrypted system tablespaces, and thus the resulting vPDBs will also have encrypted system tablespaces. Plugin doesn't support encrypting an existing unencrypted vPDB - the encryption comes from the dSource.
   Prior to 6.0.15.0, If the source has encrypted system tablespaces(SYSTEM, SYSAUX, UNDOTBS*), linking or ingestion of the source in Delphix Engine would fail.
5. Source database keystores must not be on ASM storage.
6. Only software keystores on the same host as the database files are supported. In particular, Oracle Key Vault is not supported.
7. The dSource from which the initial provision is done must be encrypted when it is linked. Existing dSources cannot be encrypted without unlinking and creating a new dSource.

Points to pay attention to:

- A CDB with one PDB (single tenant) is currently the only certified deployment for Oracle E-Business Suite with Database 19c.
- A CDB with multiple PDBs (multi-tenant) is not currently certified for Oracle E-Business Suite.
- A non-CDB architecture is not planned to be certified or supported for EBS with Database 19c.

High-level steps:

1. Create virtual dbTechStack using plugin, Make sure to remember the names of CDB, PDB that end user input in Delphix Engine UI wizard, as same will be used during vCDB and vPDB
2. Refresh target host environment in DE UI
3. Copy source DB keystore (WALLET_ROOT) files `cwallet.sso` , `ewallet.p12` to target server.
4. During VDB provision, use **Create a new container database option** on Delphix Engine UI to create vCDB & vPDB.
5. Create new EBS services in vPDB. To avoid confusion, delete the EBS services that have drifted downstream from dSource to target vPDB.
6. Create virtual AppsTier using EBS plugin.

Overview

Provisioning a Virtual Pluggable Database (vPDB) first involves using the GUI or CLI to specify the vPDB parameters (such as the vPDB name and target container) along with the snapshot to provision from. Once the provisioning is started with these parameters, the Delphix Engine does the following:

1. Mounts the snapshot files on the target host.
2. Creates and opens (in mount mode) the auxiliary container database on the target host, using the snapshot files. The auxiliary container database will have both the CDB and PDB data files from the dSource.

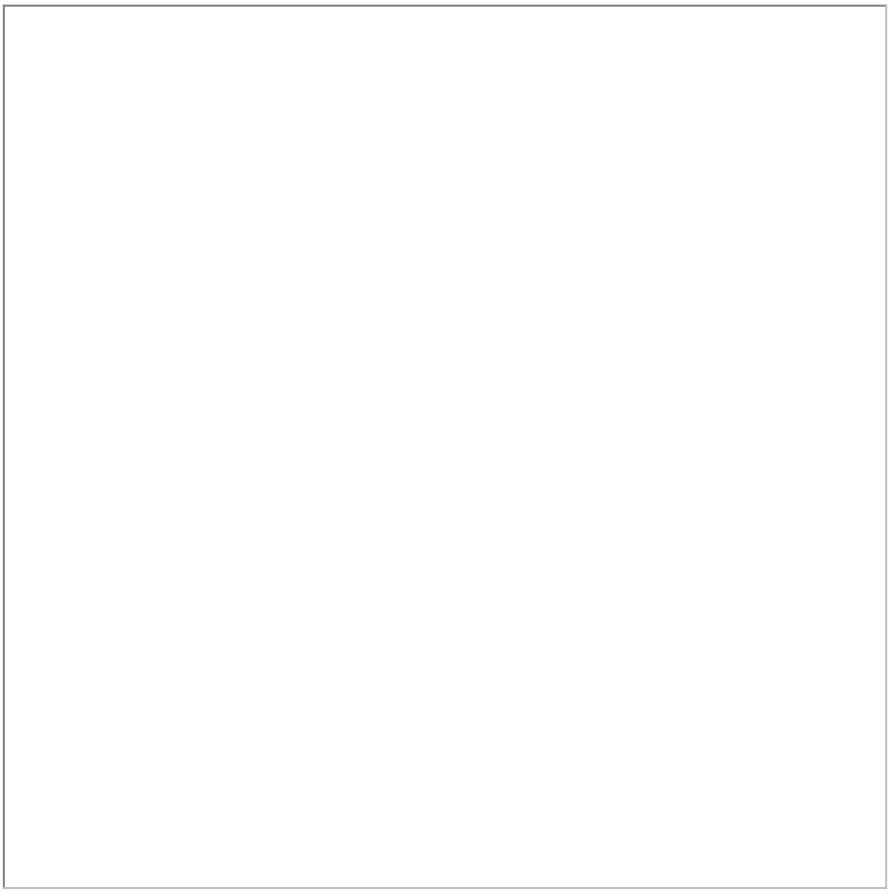
3. Completes recovery to bring the auxiliary container database into a consistent state.
4. Finalizes the state of the auxiliary database and unplugs the vPDB datafiles.
5. Plugs the vPDB into the target database, and opens it in read-write mode for general use.

If the dSource is TDE-enabled, then Delphix Engine will need to perform the following additional operations to complete the provision of a TDE-enabled vPDB to a TDE-enabled target container.

1. Mounts the snapshot files on the target host.
2. Creates a keystore with the necessary keys to apply encrypted archived log files.
3. Creates and opens (in mount mode) the auxiliary container database on the target host, using the snapshot files. The auxiliary container database will have both the CDB and PDB data files from the dSource.
4. Completes recovery to bring the auxiliary container database into a consistent state.
5. Rotates the vPDB and auxiliary CDB master encryption keys by generating new keys that are unique to the vPDB / auxiliary CDB and not associated with the source PDB or CDB.
6. Exports only the newly generated keys to an exported keyfile to enable unplug.
7. Finalizes the state of the auxiliary database and unplugs the vPDB datafiles.
8. Imports the keys from the exported keyfile into the target keystore.
9. When provisioning to a vCDB target, converts the auxiliary CDB into the final vCDB and creates the vCDB keystore from the auxiliary CDB keystore.
10. Plugs the vPDB into the target database, and opens it in read-write mode for general use.

If the initial provision of a TDE-enabled vPDB to a vCDB has either failed or been cancelled before step 9, which creates the vCDB keystore, refreshing that vPDB will fail because there is no target keystore to merge into the auxiliary CDB keystore for use during recovery. In this case, it will be necessary to delete the failed/cancelled vPDB and provision again. This does not apply when provisioning to a linked target CDB, which will already have its own keystore configured.

The following diagram illustrates the provisioning steps.



At each stage of provisioning, the keys and exported keyfiles are always on user storage. The exported keyfile is located in the artifact directory, while the auxiliary and target keystores are in the auxiliary keystores directory. Both the artifact directory and auxiliary keystores directory are subdirectories of the TDE keystores root directory, which is either user specified, or if not specified defaults to the toolkit root directory. As for non-TDE-enabled vPDBs, the final vPDB (and vCDB, if applicable) is on Delphix Engine storage while the target linked CDB and its archive logs remain on user storage.

Provisioning a TDE-enabled vPDB

To initiate the provision, Delphix needs the following pieces of information, all of which can be specified in the GUI or CLI:

Parameter	Description	CLI Parameter	Required or not
Parent keystore path	Path to a keystore that contains the keys used to encrypt the dSource datafiles. This keystore must be located on the target system. It does not have to be in the location specified by `sqlnet.ora` or `wallet_root`, but can be for consistency. It cannot be located on an ASM filesystem. This parameter can be updated if the path is changed.	source.parentTdeKeystorePath	Yes

Parameter	Description	CLI Parameter	Required or not
Parent keystore password	Password for the parent keystore. This parameter can be updated if the password is changed.	source.tdeExportedKeyFileSecret	Yes
Exported keyfile secret	When exporting keys to a keyfile from a keystore, Oracle requires a password to be set. Once specified, this cannot be changed for the life of the vPDB.	source.tdeExportedKeyFileSecret	Yes
Target keystore password	Password for the target keystore. This parameter can be updated if the password is changed.	sourceconfig.tdeKeystorePassword	Yes
Keystores root directory path	Path to a directory on the target host under which all Delphix related TDE artifacts will be created. This includes keystores used by the auxiliary CDB during provisioning and the artifact directories for TDE-enabled vPDBs.	host.oracleParameters.tdeKeystoresRootPath	Yes for cluster targets, optional for single instance targets
Target vCDB TDE Keystore Location	Path of the location at which Delphix Engine will create the keystore for vCDB provision jobs. This keystore must be located on the target system. For Oracle 12.2 the path must match what is specified by sqlnet.ora. For higher versions, Delphix Engine will set the wallet_root parameter to the provided location. This parameter must be updated if the keystore location is changed or else future Delphix Engine operations may fail.	source.targetVcdbTdeKeystorePath	Yes for vCDB targets. Not applicable to linked CDB targets
Target vCDB TDE Keystore Password	Password for the vCDB keystore. This parameter can be updated if the password is changed.	virtualCdb.sourceConfig.tdeKeystorePassword	Yes for vCDB targets. Not applicable to linked CDB targets

Linking of source TDE PDB

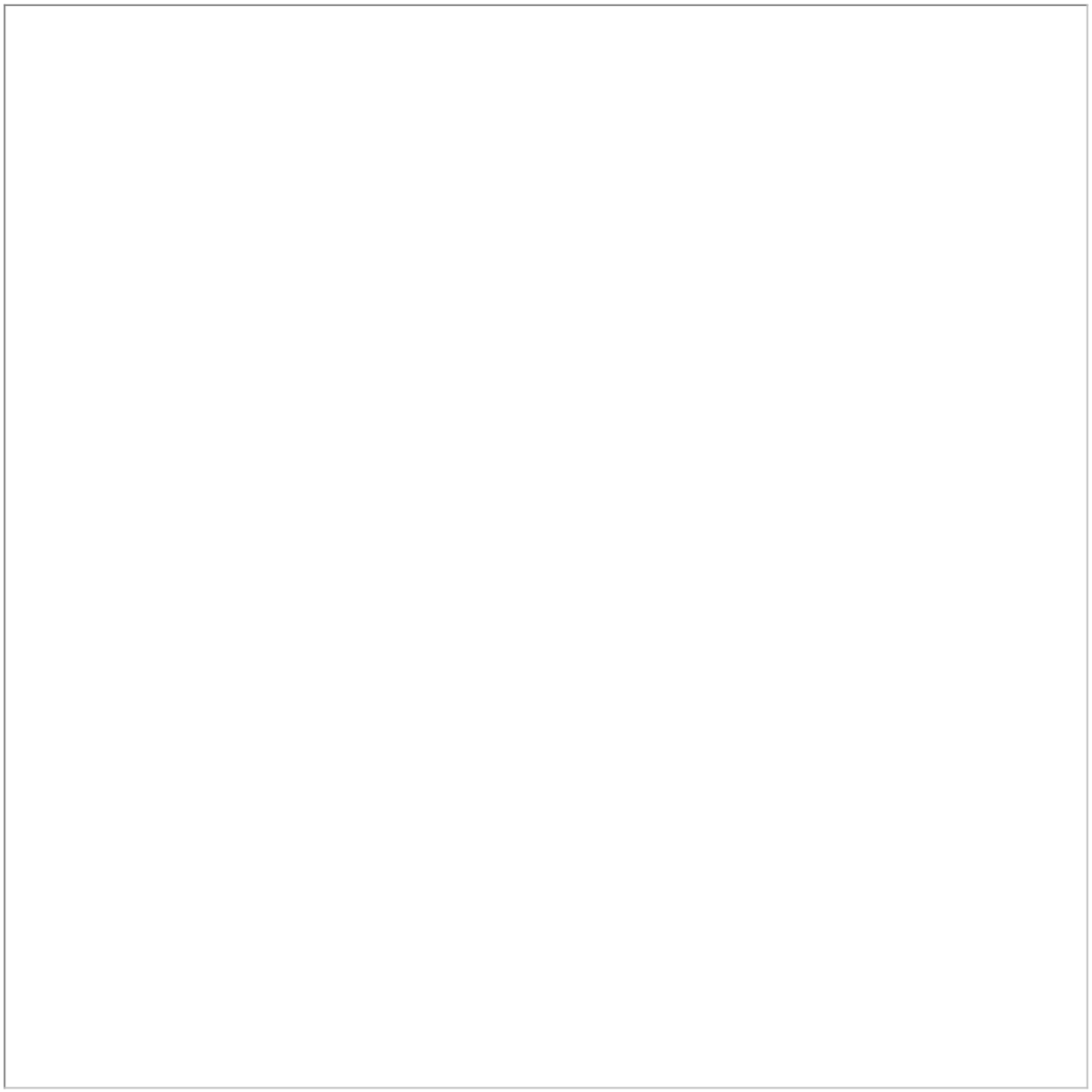
1. Login to the **Delphix Management** application.
2. Go to **Manage > Environments**.
3. Click the **Details** tab.
4. Click on the edit icon next to **ATTRIBUTES**.

5. In the **TDE Keystores Root field,** enter the path for the TDE Keystores root path**.**
6. Click the tick mark button to save the configuration.

Adding or editing the TDE keystores root directory path

The TDE Keystores root directory path is specified on the **Details** tab under **Environments**. To edit the path, do the following:

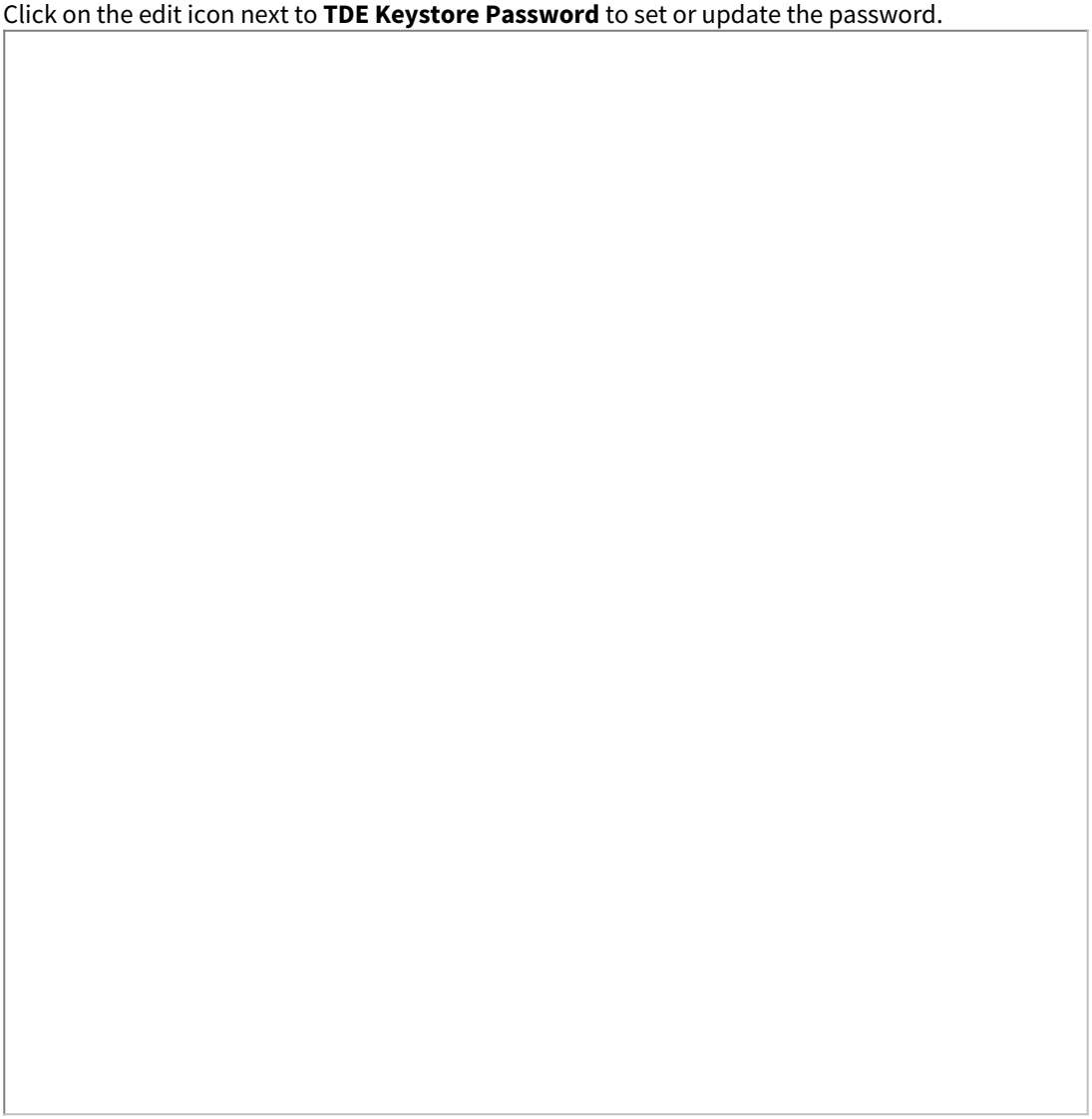
1. Login to the **Delphix Management** application.
2. Go to **Manage > Environments**.
3. Click the **Details** tab of the environment.
4. Click on the edit icon next to **ATTRIBUTES** to set or update attributes, including the Keystores root directory path.



Adding or editing the target keystore password

The target Keystore password is specified on the **Databases** tab under **Environments**. To add or edit the target Keystore password, do the following:

1. Login to the **Delphix Management** application.
2. Go to **Manage > Environments**.
3. Click the **Databases** tab of the Environment.
4. Click on the edit icon next to **TDE Keystore Password** to set or update the password.



5. Click on **Add source**. The **Add dSource** wizard displays.
6. Fill in the details and click **Submit**.

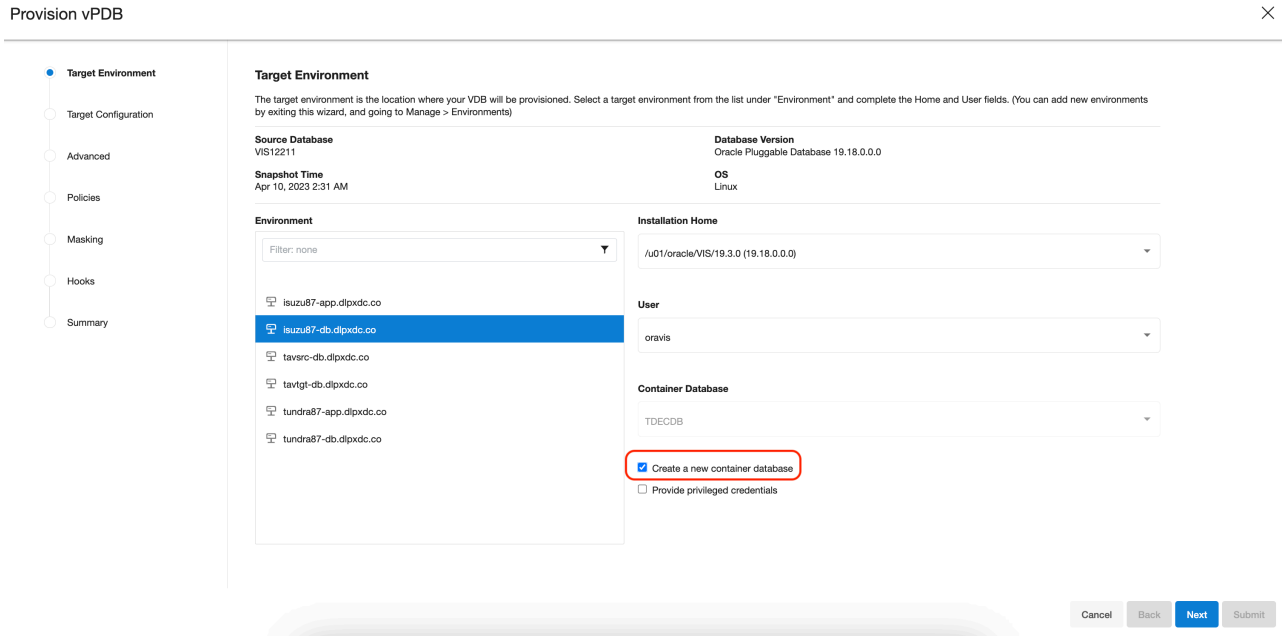
For more information, see Linking an Oracle pluggable database & Linking data sources with Oracle

After you have successfully linked the source PDB:

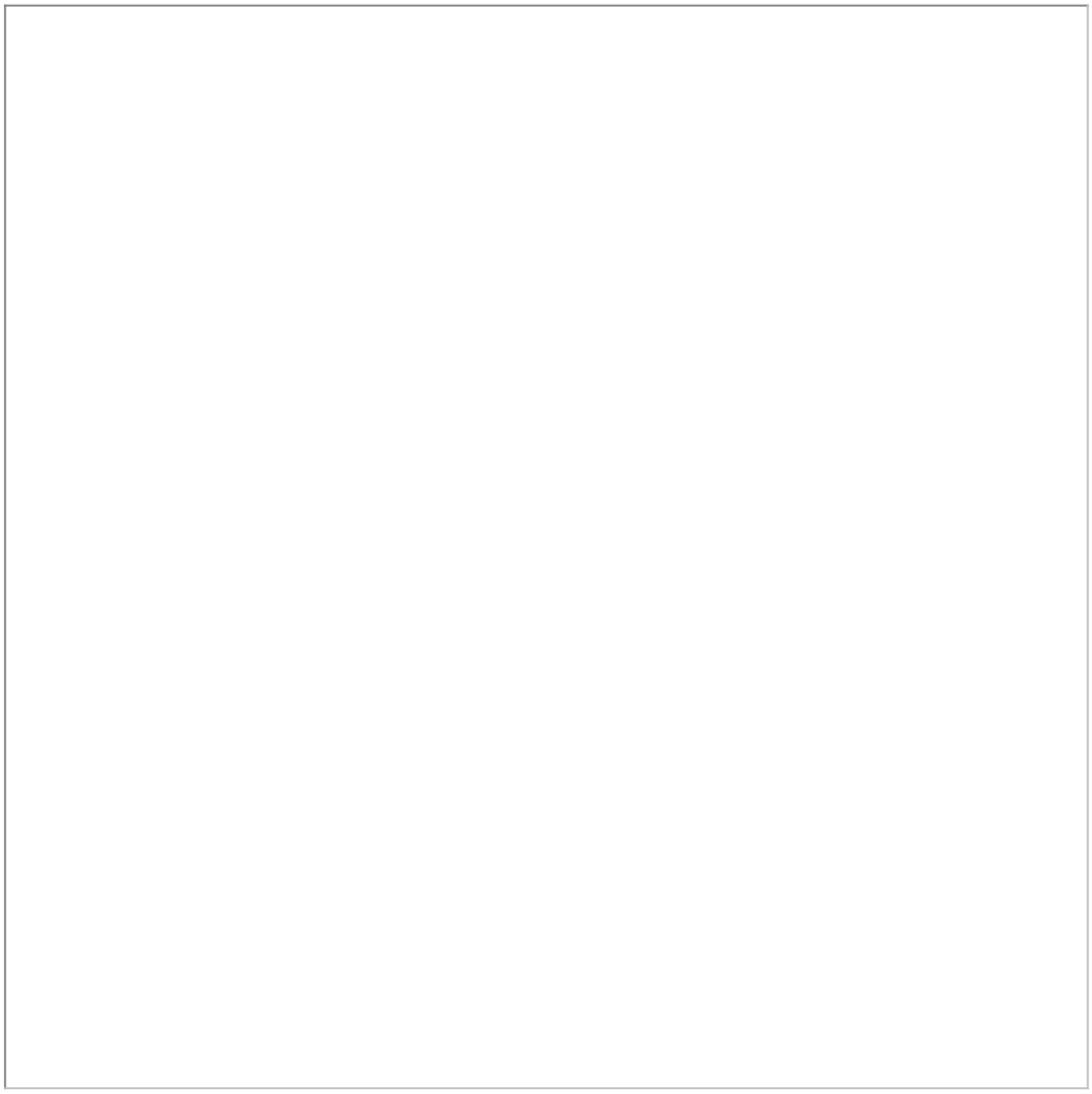
- Create a dbTechStack on the target server using the plugin.
- Refresh the target host environment in the Delphix Engine UI.
- Copy the source DB keystore (WALLET_ROOT) files `cwallet.sso` , `ewallet.p12` to target server and mention that location in **Parent Database TDE Keystore Location**.

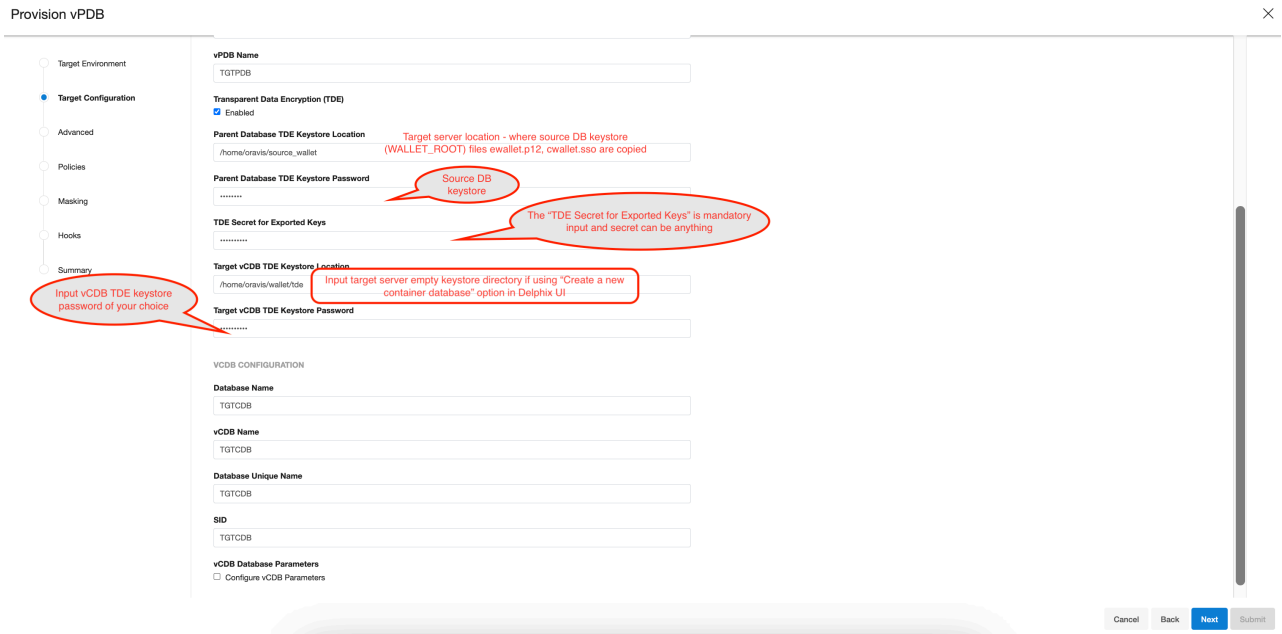
```
On Source CDB, run below query to get WALLET_ROOT(files cwallet.sso,
ewallet.p12) location -
select wrl_parameter from v$encryption_wallet;
WRL_PARAMETER
------------------------
/home/oravis/wallet/tde/
```

It is mandatory to copy the `ewallet.p12` file from the source to the target server for the target container database to use it. No exporting of keys needed, this is done by the Delphix engine itself using the path and secrets provided in the configuration. On target server, `cwallet.sso`, `ewallet.p12` files should be owned by ORACLE_HOME installation owner.



The **Parent Database TDE keystore Location** is the location of the keystore for the dSource, which can be the actual location if it's a provision back to the source, or else it needs to be copied to the target. the parent keystore password is the password for the dSource keystore. You can decide the **TDE Secret for Exported Keys,** which is a mandatory input and secret can be anything, it is used when exporting and importing the encryption keys.





Once a TDE-enabled vPDB is provisioned, it can be used the same as a non-TDE-enabled vPDB within Delphix, with the exception of migrate. There are few caveats:

- A refresh operation will use the parent keystore for the recovery. If the dSource is rekeyed then the user will need to update the parent keystore with the new keys. Similarly, if the location or password to the parent keystore has changed then they should be updated before the refresh.
- A rewind operation will use the target keystore for the recovery. If the vPDB is rekeyed after it is provisioned, then the rekey will update the target keystore, so it does not need to be updated in Delphix.
- For a single vPDB in a vCDB, if the vCDB keystore location is changed, the new path must be updated in Delphix before refresh or rewind.
- Each disable operation will result in the keys being exported to an exported keyfile in the artifact directory, to be used for a subsequent enable. Refresh and rewind operations will first disable the existing vPDB, so those will also result in a new exported keyfile in the artifact directory.
- Provisioning a second-generation vPDB (vvPDB) from a TDE-enabled vPDB is done in the same manner as a first-generation vPDB, by specifying the TDE parameters during provision. The current keystore for the vPDB can be specified as the parent keystore.

TDE keystores root and artifact directory

The artifact directory stores the exported keyfiles used during the workflows for TDE-enabled vPDBs. It is located under the keystores root, in the directory `oracle_tde_keystores`. Each TDE-enabled vPDB will have its own directory within the `oracle_tde_keystores` directory, identified by the vPDB name, group name, and a unique identifier, separated by an underscore. If the keystores root directory is not specified, then it defaults to the toolkit directory path.

For example, if the keystores root directory is `/work` (or keystores root is not specified, and the toolkit directory is `/work`), the artifact directory for the vPDB tde_vpdb in the group Encrypted could be

```
/work/oracle_tde_keystores/tde_vpdb_Encrypted_ce7a47e6-8860-4398-bab0-cf0233fc5e3c
```

Within the artifact directory, there is a subdirectory `exported_keys` which contains within it the exported keyfiles for each timeflow associated with that vPDB. Each time an export is performed, a new exported keyfile is

generated with a timestamp. The contents of the artifact directory may change for future releases, but the path to the artifact directory and the naming convention is not anticipated to change.

As the default keystores root directory root is at the same level as the toolkit directory, it will not be overwritten if a host is refreshed through the Delphix Engine and the toolkit updated. It is the customer's responsibility to maintain the artifact directory and ensure that the contents are not lost, as a disk failure could prevent a TDE-enabled vPDB from being accessed. Thus it is recommended that the keystores root directory be on a disk which is regularly backed up.

If a vPDB is moved to a different host (either through the migrate workflow or an enable after a failover, then the artifact directory will need to be copied to the new target host. See Migrating a TDE-enabled vPDB for details on the manual steps needed for migration.

The artifact directory is not removed when a TDE-enabled vPDB is deleted; the customer can remove it after confirming that the vPDB has been removed (including from any replicated Delphix Engines).

For TDE enabled vPDBs, use manual hooks in configure clone and Pre-snapshot. You should not use hooks utility feature with TDE enabled virtual database.

For versions 19.3 or later, follow the below guidelines if you see the following error:

- ORA-01017: invalid username/password; logon denied or ORA-28040 During cloning: No Matching Authentication Protocol

EBS application database user accounts created in the earlier release use a case-insensitive password version from an earlier release authentication protocol, such as the 10G password version. In case, if your database release user account passwords have not been reset and the following configuration persists:

On source: The database server has been configured with SEC_CASE_SENSITIVE_LOGON set to FALSE, so that it can only authenticate users who have a 10G case-insensitive password version, then, on Target container VDB, it becomes necessary to set the SEC_CASE_SENSITIVE_LOGON to FALSE, to make authentication of users successful.

To take advantage of the password protections introduced in Oracle Database 19c, users must change their passwords. Occasionally, it is necessary to restart the database to resolve connectivity to the database. This is atypical but may be necessary.

To set-up target:

On target container database, alter system set SEC_CASE_SENSITIVE_LOGON=FALSE scope=both;

Configure clone first hook for a 19c multi-tenant high privileged user:

- Configure clone hook will blindly create and start the services, No validation on services has been implemented, but this will get improve over time.

```
#!/usr/bin/env bash
#
# Copyright (c) 2023 by Delphix. All rights reserved.
#
# shellcheck source=/dev/null
# NOTE: Ensure the below environment variables will be set up correctly by the
# shell. If not, hardcode or generate the values below.
HOSTNAME=$(uname -n | cut -d '.' -f1)
CONTEXT_NAME=${DELPHIX_PDB_NAME}_${HOSTNAME}
SOURCE_PDB_NAME=$SOURCE_PDB_NAME_PASSWORD
```

```
APPS_PASSWD=$DLPX_SOURCE_APPS_PASSWORD_PASSWORD
. "${ORACLE_HOME}/${ORACLE_SID}_${HOSTNAME}.env";
# Check for local_listener parameter is set for PDB, otherwise set it
# appropriately
check_value=$(sqlplus -s "/ as sysdba" <<EOF
alter session set container="${DELPHIX_PDB_NAME}";
show parameter local_listener;
EOF
)
local_listener=$(echo "$check_value" | awk '{print $11}')
value=("${local_listener//:/ }")
host="${value[0]}"
port="${value[1]}"
curr_port=$(grep PORT < "${ORACLE_HOME}/network/admin/listener.ora" | awk '{print
$9}' | sed 's///g')
if [[ $port != "$curr_port" || $host != "${HOSTNAME}" ]]; then
sqlplus -s "/ as sysdba" <<EOF
alter session set container="${DELPHIX_PDB_NAME}";
alter system set local_listener='${HOSTNAME}:${curr_port}';
alter system register;
EOF
fi

#Create, Start EBS PDB services and Delete Source PDB services
. "${ORACLE_HOME}/${ORACLE_SID}_$(hostname -s).env";
sqlplus -s "/ as sysdba" <<EOF
alter session set container="${DELPHIX_PDB_NAME}";
BEGIN DBMS_SERVICE.STOP_SERVICE( service_name => '${SOURCE_PDB_NAME}_ebs_patch');
end ;
/
BEGIN DBMS_SERVICE.STOP_SERVICE( service_name => 'ebs_${SOURCE_PDB_NAME}'); end;
/
BEGIN DBMS_SERVICE.DELETE_SERVICE( service_name => '${SOURCE_PDB_NAME}_ebs_patch');
end ;
/
BEGIN DBMS_SERVICE.DELETE_SERVICE( service_name => 'ebs_${SOURCE_PDB_NAME}'); end;
/
BEGIN DBMS_SERVICE.CREATE_SERVICE( service_name => 'ebs_${DELPHIX_PDB_NAME}',
network_name => 'ebs_${DELPHIX_PDB_NAME}');
end;
/
BEGIN DBMS_SERVICE.CREATE_SERVICE( service_name => '${DELPHIX_PDB_NAME}_ebs_patch',
network_name => '${DELPHIX_PDB_NAME}_ebs_patch');
end;
/
BEGIN DBMS_SERVICE.START_SERVICE( service_name => 'ebs_${DELPHIX_PDB_NAME}');
end;
/
BEGIN DBMS_SERVICE.START_SERVICE( service_name => '${DELPHIX_PDB_NAME}_ebs_patch');
end;
/
alter system register;
EOF
```

```
#For compatibility with version 6.0.16.0 or later
. ${ORACLE_HOME}/${ORACLE_SID}_${HOSTNAME}.env; sqlplus -s "/ as sysdba" <<EOF
shutdown immediate;
startup;
ALTER PLUGGABLE DATABASE ALL OPEN read write services=all;
alter pluggable database ${DELPHIX_PDB_NAME} open read write services=all;
alter pluggable database all save state instances=all;
EOF
sqlplus "/ as sysdba" <<EOF
@${ORACLE_HOME}/appsutil/install/${CONTEXT_NAME}/adupdlib.sql so
EOF
. "${ORACLE_HOME}/${CONTEXT_NAME}.env";
perl "${ORACLE_HOME}/appsutil/clone/bin/adcfgclone.pl" dbconfig "${ORACLE_HOME}/
appsutil/${CONTEXT_NAME}.xml" <<EOF # noqa
${APPS_PASSWD}
EOF
```

Configure clone first hook for a 19c multi-tenant low privileged user:

```
#!/usr/bin/env bash
#
# Copyright (c) 2022 by Delphix. All rights reserved.
#
# shellcheck source=/dev/null
# NOTE: Ensure the below environment variables will be set up correctly by the
# shell. If not, hardcode or generate the values below.
# Input the SOURCE_PDB_NAME enclosed in double quotes.
# Input the DLPX_PRIV_USER according to your environment setup.
SOURCE_PDB_NAME="TDEPDB"
DLPX_DB_EXEC_SCRIPT="<Remote BIN location till dlpx_db_exec script>"
DLPX_PRIV_USER=oravis
SOURCE_PDB_NAME=$SOURCE_PDB_NAME_PASSWORD
APPS_PASSWD=$DLPX_SOURCE_APPS_PASSWORD_PASSWORD
CONTEXT_NAME=${DELPHIX_PDB_NAME}_$(hostname -s)

"${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/${ORACLE_SID}_$
(hostname -s).env;"

# Check for local_listener parameter is set for PDB, otherwise set it appropriately
check_value=$("${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/$
{CONTEXT_NAME}.env; sqlplus -s \"/ as sysdba\" <<-EOF
alter session set container=${DELPHIX_PDB_NAME};
show parameter local_listener;
EOF
")

local_listener=$(echo "$check_value" | awk '{print $11}')
value=("${local_listener//:/ }")
host="${value[0]}"
port="${value[1]}"
```

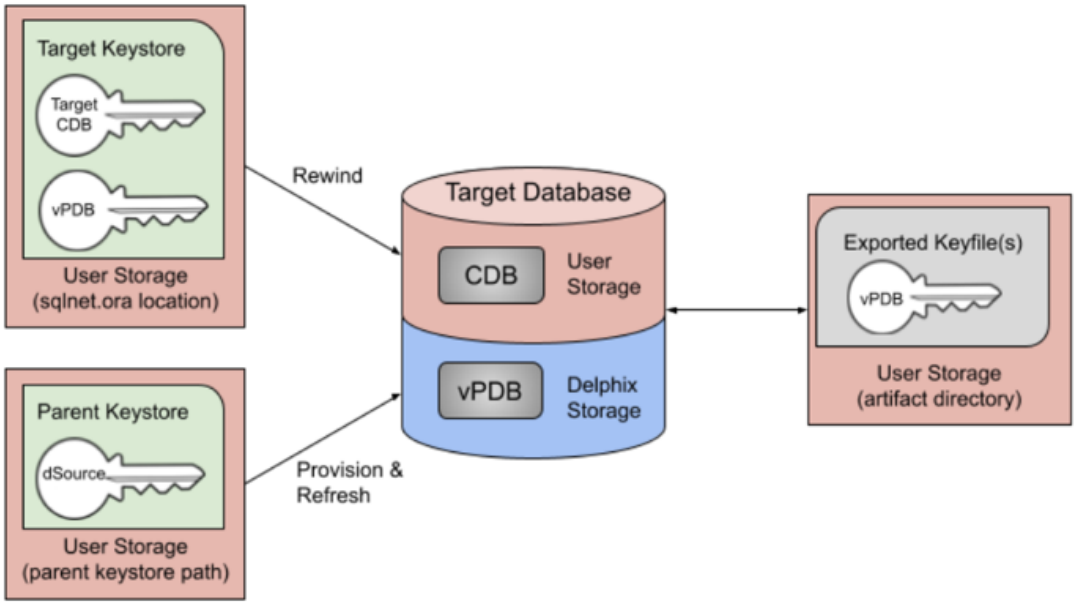
```
curr_port=$(grep PORT < "${ORACLE_HOME}/network/admin/listener.ora" | awk '{print
$9}' | sed 's/)//g')

if [[ $port != "$curr_port" || $host != "$(hostname -s)" ]]; then
    "${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/${ORACLE_SID}_$
(hostname -s).env; sqlplus -s \"/ as sysdba\" <<EOF
    alter session set container=${DELPHIX_PDB_NAME};
    alter system set local_listener='$(hostname -s):${curr_port}';
    alter system register;
EOF
"
fi

#Create, Start EBS PDB services and Delete Source PDB services
"${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/${ORACLE_SID}_$
(hostname -s).env; sqlplus -s "/ as sysdba" <<EOF
alter session set container="${DELPHIX_PDB_NAME}";
BEGIN DBMS_SERVICE.STOP_SERVICE( service_name => '${SOURCE_PDB_NAME}_ebs_patch');
end ;
/
BEGIN DBMS_SERVICE.STOP_SERVICE( service_name => 'ebs_${SOURCE_PDB_NAME}'); end;
/
BEGIN DBMS_SERVICE.DELETE_SERVICE( service_name => '${SOURCE_PDB_NAME}_ebs_patch');
end ;
/
BEGIN DBMS_SERVICE.DELETE_SERVICE( service_name => 'ebs_${SOURCE_PDB_NAME}'); end;
/
BEGIN DBMS_SERVICE.CREATE_SERVICE( service_name => 'ebs_${DELPHIX_PDB_NAME}',
network_name => 'ebs_${DELPHIX_PDB_NAME}');
end;
/
BEGIN DBMS_SERVICE.CREATE_SERVICE( service_name => '${DELPHIX_PDB_NAME}_ebs_patch',
network_name => '${DELPHIX_PDB_NAME}_ebs_patch');
end;
/
BEGIN DBMS_SERVICE.START_SERVICE( service_name => 'ebs_${DELPHIX_PDB_NAME}');
end;
/
BEGIN DBMS_SERVICE.START_SERVICE( service_name => '${DELPHIX_PDB_NAME}_ebs_patch');
end;
/
alter system register;
EOF
"

"${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/${ORACLE_SID}_$
(hostname -s).env; sqlplus \"/ as sysdba\" <<-EOF
@${ORACLE_HOME}/appsutil/install/${CONTEXT_NAME}/adupdlib.sql so
EOF
"
"${DLPX_DB_EXEC_SCRIPT}" -u"${DLPX_PRIV_USER}" ". ${ORACLE_HOME}/${CONTEXT_NAME}.env;
perl ${ORACLE_HOME}/appsutil/clone/bin/adcfgclone.pl dbconfig ${ORACLE_HOME}/
appsutil/${CONTEXT_NAME}.xml <<-EOF1
```

```
${APPS_PASSWD}
EOF1
"
```

The remaining configure clone & Pre-snapshot hooks remains unchanged. It is mandatory to use hooks mentioned in documentation while provisioning EBS virtual PDB or database.

Refreshing and rewinding a TDE-enabled vPDB

Just like a non-TDE-enabled vPDB, a TDE-enabled vPDB can be refreshed from the dSource or rewound to a previous snapshot or point in time. In each case, no additional manual steps or input from the user is required. The first step of a refresh or rewind operation is to disable the existing vPDB, which will result in a new keyfile exported to the artifact directory. The appropriate snapshot files are then mounted for the auxiliary database so that it can be recovered and brought to a consistent state. Since the vPDB is TDE-enabled, a keystore is needed for the recover operation. For a refresh, the Delphix Engine will use the parent keystore, and for a rewind, the Delphix Engine will use the target keystore, as shown below.



Key rotation

There are two potential places for keys to be rotated in a vPDB environment:

1. dSource: If the dSource keys are rotated and a new snapshot taken with the new key, the customer is responsible for updating the parent keystore before refreshing from the later snapshot encrypted with the new key. The parent keystore would then contain both the new key and the original keys.
2. Target: If the target CDB keys are rotated, the target keystore will be updated. This is why the Delphix Engine uses the target keystore for rewind operations.

In either scenario, the keystore used for recovery will contain the current and all prior keys used to encrypt the datafiles and archive logs, for both the vPDB and CDB used in the auxiliary container.

vPDB encryption key management

During the provisioning process of a TDE-enabled vPDB, Delphix generates a unique encryption key for the vPDB. This unique key is not associated with the parent keystore to ensure that no keys from the parent are imported by the target. During refresh and rewind operations, Delphix reuses that key after recovery has finished. It is possible to customize the key that is used by updating the `tdeKeyIdentifier` parameter of the source via the CLI. If a valid key_id is entered for a key that is already present in the keystore, that key will be used as the active encryption key for the vPDB at the end of refresh/rewind. If the field is unset, Delphix will generate a new encryption key for the vPDB to be used from that point onward. This procedure is the same when using a vCDB, in which case Delphix will also generate a new unique encryption key for the vCDB that is reused for refresh and rewind, and which can be customized by updating the `tdeKeyIdentifier` parameter of the CDB source. See the CLI steps for Locating and updating the value of tde encryption key

EBS database port flexibility/customization

This topic describes the process of provisioning VDBs in EBS dbTechStack & AppsTier plugin to run on a customized port.

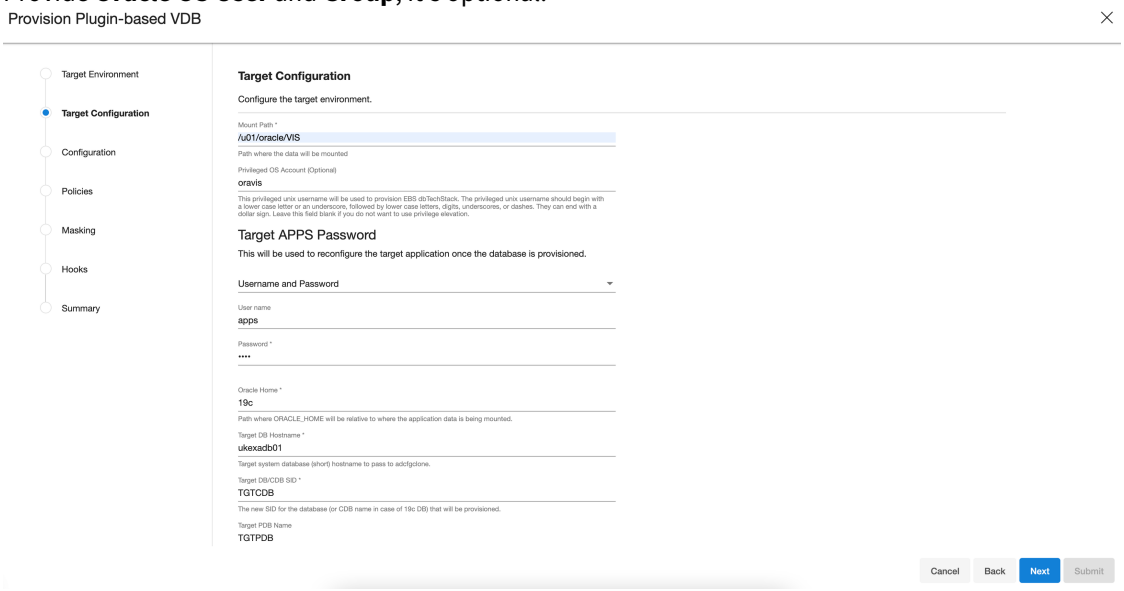
Provisioning the EBS dbTechStack

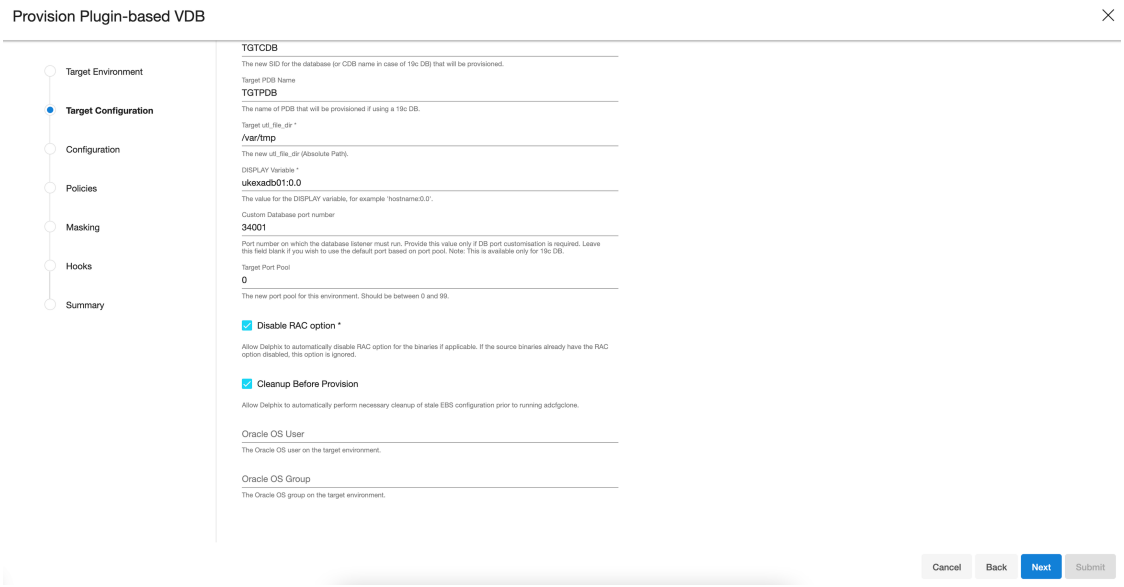
1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **dbTechStack dSource**.
5. Click the **TimeFlow** tab.
6. Select a dSource **snapshot**.
7. Click **Provision**. The **Provision VDB** wizard will open.
8. Select an **Environment.** This environment will host the virtual dbTechStack and be used to execute hook operations specified in step 15 in the **s** section.
9. Select **E-Business Suite R12.2 dbTechStack** from the **Installation** dropdown.
10. Select an **Environment User**.
11. This user should be the `oracle` user outlined in Preparing target EBS R12.2 environments for provisioning
12. Enter a **Mount Path** for the virtual dbTechStack files.
13. Enter the **EBS-specific parameters** for the virtual dbTechStack. A subset of these parameters is discussed in more detail below.
    a. The **Privileged OS Account (Optional)** field should contain the high privileged user when the low privileged user is being used for provisioning.
    b. The **Target APPS Password** is the new apps password that is required to configure virtual dbTechStack. This password is encrypted when stored within the Delphix Engine and is available as an environment variable to the `adcfgclone` process.
    c. Provide source oracle home directory name relative to oracle base installation directory that was provided during linking of dbTechStack in Oracle Home input field.
    d. Ensure that the **Target DB Hostname** value is the short hostname, not the fully-qualified hostname.
    e. The **Target DB/CDB SID** is the new database SID (CDB SID in case of 19c) that is required to configure virtual dbTechStack.
    f. The **Target PDB Name** field is added for the pluggable database to be configured for Oracle 19c database. Please leave this field empty when using Oracle 11g/12c database.
    g. Provide **Target utl_file_dir** , the default value for this field will be “/var/tmp".
    The ***utl_file_dir*** initialization parameter is deprecated in Oracle Database 12c Release 2 (12.2.0.1), and maybe unsupported in a future release.
    h. Provide **DISPLAY Variable**, the default value is hostname:0.0
    i. [Optional] **Custom Database Port Number -** Provide the port number on which database listener must run. **Note:** Enter a numeric port value in this field only if you want to use DB port customisation. Leaving this field blank automatically uses the default port based on port pool. The custom database port number is applicable only for Oracle 19c database.
    Note: If you want the dbTechStack and listener to run on custom DB port, it is recommended to provide **Target Port Pool** value either as default or 0; also the **Target Port Pool** value in dbTechStack is not related with **Run/Patch Edition Port Pool** value in Appstier.
    j. Provide **Target Port Pool**, please provide a port pool that is available. By default, the value is 1. The range for this value is 0 to 99.
    k. Enable the **Disable RAC** option if you want to permit the Delphix Engine to automatically disable the RAC option for the binaries when applicable. This option is necessary if provisioning from a dSource with RAC dbTier because the binaries are relinked with the `rac_on` option even after running

`adcfgclone` . If the source binaries already have the RAC option disabled (also the case for SI dbTier), the Delphix Engine ignores this option.

l. Enable the **Cleanup Before Provision**option if you want to permit the Delphix Engine to automatically clean up stale EBS configuration during a refresh. This option is recommended, but only available if your Oracle Home is patched with Oracle Universal Installer (OUI) version 10.2 or above.
    i. With this option enabled, the Delphix Engine will inspect the target environment's oraInventory prior to refreshing this virtual dbTechStack. If any Oracle Homes are already registered within the specified **Mount Path**, the Delphix Engine will detach them from the inventory prior to running adcfgclone. These homes must be detached prior to running post-clone configuration. If they are not detached, `adcfgclone` will fail, citing conflicting oraInventory entries as an issue.
    ii. Without this option enabled, Oracle Homes that conflict with the specified **Mount Path** will be reported in an error instead of automatically detached. For refresh to succeed, you must manually detach conflicting Oracle Homes prior to refresh.

m. Provide **Oracle OS User** and **Group**, it's optional.





14. Click **Next**.
15. Enter a **VDB Name**.
16. Select a **Target Group** for the VDB.
    If necessary, click the green **Plus** icon to add a new group.
17. Select a **Snapshot Policy** for the VDB. If necessary, click the **Plus** icon to create a new policy.
    Note: Snapshot conflicts: when Snapshot is running against the dbTechStack, database, or appsTier, the Delphix Engine also executes pre-clone logic to ensure the latest configuration is staged in the captured snapshots. Unfortunately, if multiple Snapshots are running against the same EBS instance concurrently, this pre-clone logic may fail and produce bad snapshots.
    To avoid SnapSync conflicts, spread out your SnapSync policies for an EBS instance by one hour or more.
18. Click **Next**.
19. Enter any **custom hook operations** that are needed to help correctly manage the virtual dbTechStack files. For more information about these hooks, when they are run, and how operations are written. The Configure Clone hook will be run after the `adcfgclone.pl` tool has both mounted and configured the dbTechStack.
20. Click **Next**.
21. Click **Submit**.

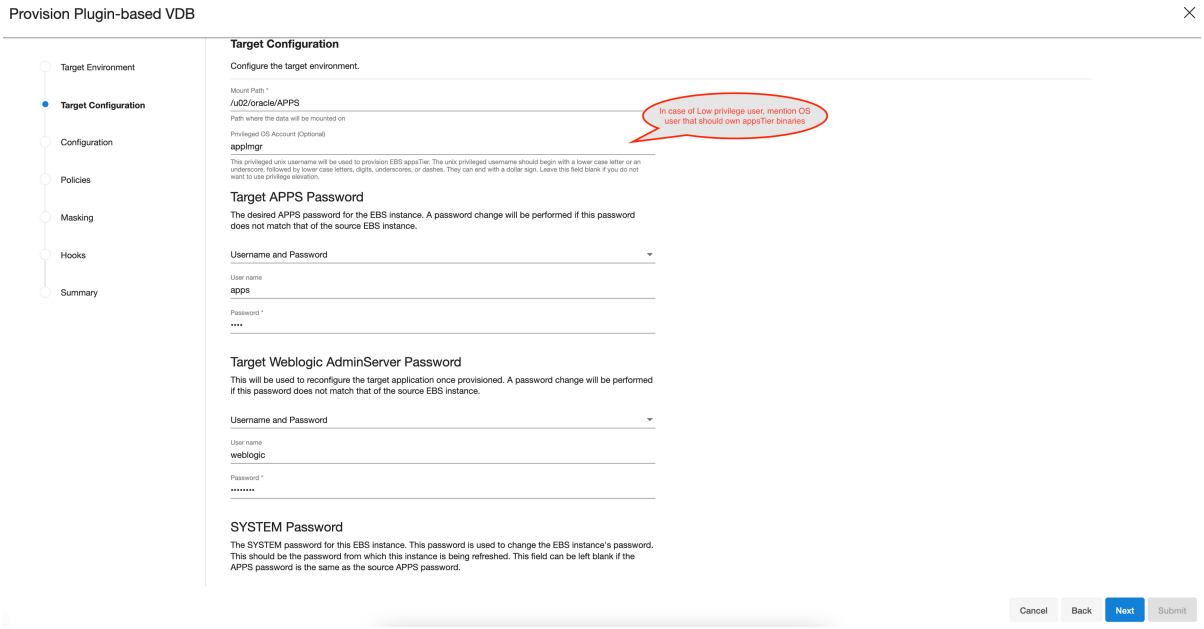
When provisioning starts, you can review the progress of the job in the **Datasets** panel, or in the **Job History** panel of the **Dashboard**. When provisioning is complete, the dbTechStack VDB will be included in the group you designated and listed in the **Datasets** panel. If you select the dbTechStack VDB in the **Datasets** panel and click the **Configuration** tab, you can view information about the virtual files and its Data Management settings.

For tips on monitoring the progress of dbTechStack provisioning, see Monitoring EBS R12.2 dbTechStack provisioning progress

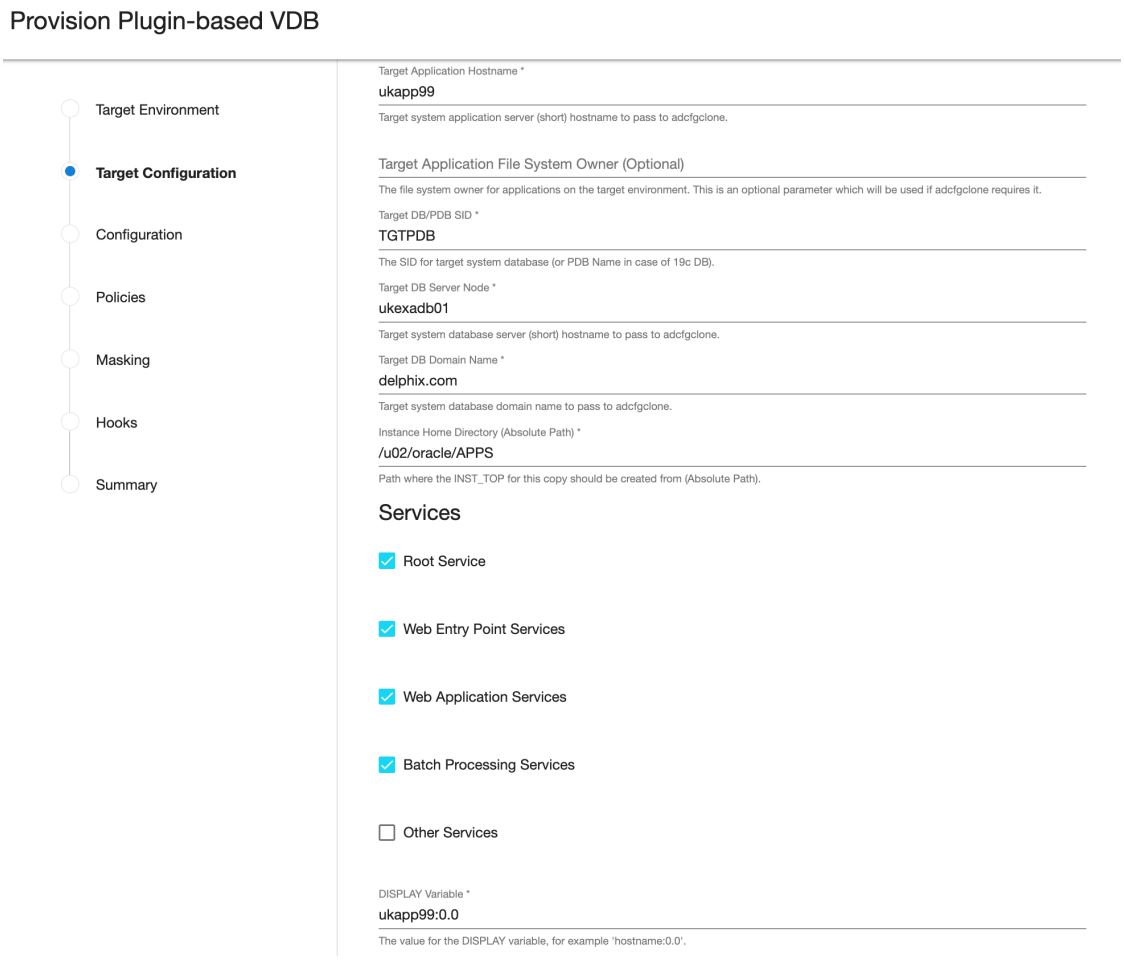
Provisioning the EBS appsTier

1. Login to the **Delphix Management** application using **Admin** credentials.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **appsTier dSource**.
5. Select a dSource **snapshot**. All snapshots will have staged configuration prepared by `adpreclone.pl` and any hook operations placed on the dSource.
6. Click **Provision**. The **Provision VDB** wizard will open.

7. Select an **Environment.** This environment will host the virtual appsTier and be used to execute hook operations specified in a few steps. This environment will also run the WebLogic Admin server (Web Administration service) for the virtual appsTier. If you are provisioning a multi-node appsTier, you will be able to specify additional environments to host the virtual appsTier in a few steps.
8. Select E-Business Suite R12.2 appsTier from the **Installation** dropdown.
9. Select an Environment User. This user should be the `applmgr` user-outlined in Preparing target EBS R12.2 environments for provisioning
10. Enter a **Mount Path** for the virtual appsTier VDB.If you are provisioning a multi-node appsTier, this mount path will be used across all target environments.
11. Enter the **EBS-specific parameters** for the virtual appsTier. A subset of these parameters are discussed in more detail below.



   a. **Privileged OS Account (Optional)** field should contain high privileged user when low privileged user is being used for provisioning.
   b. The **Target APPS Password** is the new apps password that is required to configure and manage the virtual appsTier. This password is encrypted when stored within the Delphix Engine and is available as an environment variable to the `adcfgclone` , `adstrtal` , and `adstpall` processes.
   c. The **Weblogic AdminServer Password** is the new WebLogic password required to configure the virtual appsTier. A password change will be performed if this password does not match that of the source EBS instance. This password is encrypted when stored within the Delphix Engine and is available as an environment variable to the adcfgclone, adstrtal, and adstpall processes.
   d. The **SYSTEM Password** is required to configure the virtual appsTier with the new apps password. This password is not required if the Target Apps Password is the same as the source.
   e. Provide the Target Application hostname.
   f. The **Target DB/PDB SID** is the new database SID (PDB SID in case of 19c) that is required to configure and manage the virtual appsTier.
   g. Provide the Target DB server node hostname.
   h. Provide **Target DB Domain Name**, this will be provided to the ***adcfgclone.***
   i. Delphix recommends specifying an **Instance Home Directory** under the **Mount Path** so that instance-specific EBS files live on Delphix-provided storage.



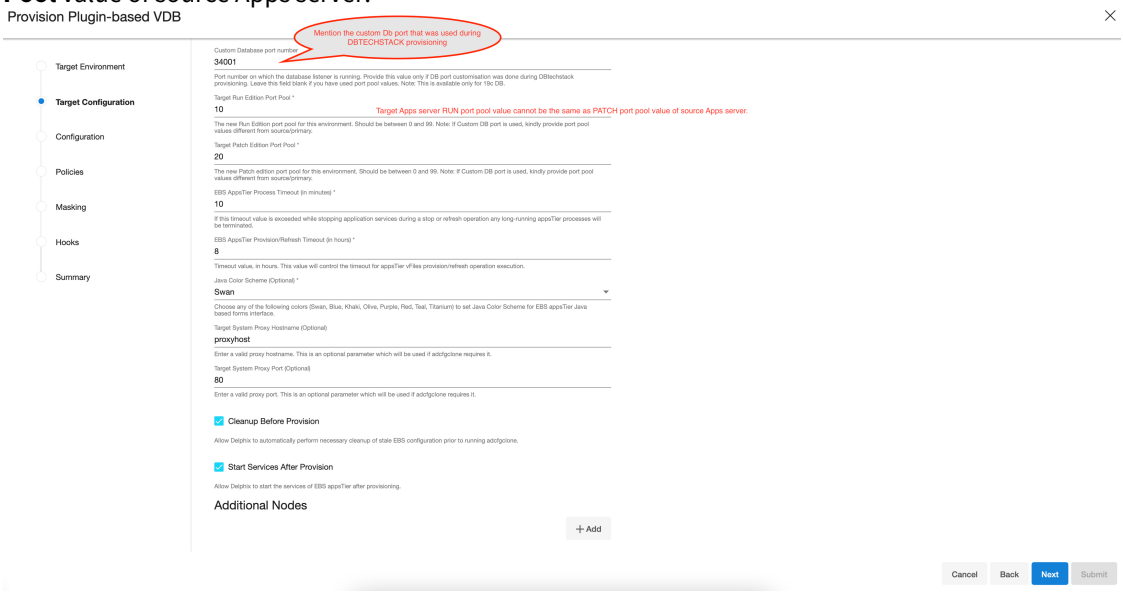
For example, if the provided **Mount Path** is `/u01/oracle/VIS`, then providing an **Instance Home Directory** of `/u01/oracle/VIS` would allow EBS to generate virtual application INST_TOP in `/u01/oracle/VIS/fs1/inst/apps/<CONTEXT_NAME>` > and `/u01/oracle/VIS/fs2/inst/apps/<CONTEXT_NAME>` >.

   i. If you are provisioning a single-node appsTier, this recommendation is OPTIONAL; putting instance-specific EBS files on Delphix-provided storage merely eases the administration of the virtual EBS instance.
   ii. If you are provisioning a multi-node appsTier, this recommendation is REQUIRED; the Delphix Engine's automation requires that all nodes in the appsTier have access to instance-specific files via Delphix-provided storage.

j. Ensure that the **Target Application Hostname** and **Target DB Server Node** values are the short hostnames, not the fully-qualified hostnames.
k. Select **Services**, by default all the ***Root***, ***Web Entry Point***, ***Web Application*** and ***Batch Processing*** would be checked. In case you are using multinode environment and want to start a few services on a secondary node then this field is going to be very useful for you.
l. Provide **DISPLAY Variable**, the default value is hostname:0.0
m. [Optional] **Custom Database Port Number -** Provide the port number on which database listener must run. **Note:** Enter a numeric port value in this field only if you want to use DB port customisation, this value should match with the custom database port number given during the DBTechStack

provisioning. Leaving this field blank automatically uses the default port based on port pool. The custom database port number is applicable only for Oracle 19c database.

n. Provide **Target Run Edition Port Pool**, the value of this field should be 0 to 99.
o. Provide **Target Patch Edition Port Pool**, the value of this field should be 0 to 99. This value should be Target Run Edition Port Pool plus 1. For example, if Run Edition Port Pool is 9 then the value should be 10.
Note: Target Apps server **RUN Edition Port Pool** value cannot be the same as **PATCH Edition Port Pool** value of source Apps server.



p. The **EBS AppsTier Timeout** is required to terminate all the long-running appsTier processes which have exceeded the timeout value when stopping the applications as part of a refresh. This timeout will be calculated in minutes. For example, if set to 30, then we run adstpall at the start of the refresh, and if after 30 minutes the application has not stopped, then the processes will be terminated to allow the refresh to continue.
q. The **EBS AppsTier Provision/Refresh Timeout** is required to terminate configure and cloning via `adcfgclone` process which have exceeded the timeout as part of a provision/refresh. This timeout will be calculated in hours and the default value will be 8 hours. For example, if set to 9, then plugin will detect a timeout after 9 hours instead of getting in a hung state and UI error for the same will be prompted.
r. The **Java Color Scheme** is an **optional** parameter and is used to change the JAVA color for provisioned Oracle EBS JAVA based form interface. Please choose any of the following colors (Swan, Blue, Khaki, Olive, Purple, Red, Teal, Titanium) to set Java Color Scheme for EBS appsTier java-based forms interface.
s. Provide **Target System Proxy Hostname (Optional)**, This is an optional parameter that will be used if ***adcfgclone*** requires it.
t. Provide **Target System Proxy Port (Optional)**, This is an optional parameter that will be used if ***adcfgclone*** requires it.
u. Enable the **Cleanup Before Provision**option if you want to permit the Delphix Engine to automatically cleanup stale EBS configuration during a refresh. This option is recommended, but only available if your Oracle Home is patched with Oracle Universal Installer (OUI) version 10.2 or above.
    i. With this option enabled, the Delphix Engine will inspect the target environment's oraInventory prior to refreshing this virtual appsTier. If any Oracle Homes are already registered within the specified **Mount Path**, the Delphix Engine will detach them from the

inventory prior to running `adcfgclone`. These homes must be detached prior to running post-clone configuration. If they are not detached, `adcfgclone` will fail, citing conflicting oraInventory entries as an issue. The Delphix Engine will also remove any conflicting INST_TOP directories left on the environment. Non-conflicting INST_TOP directories will not be modified.

ii. Without this option enabled, Oracle Homes or INST_TOP directories that conflict with the specified **Mount Path** or desired INST_TOP location will be reported in errors instead of automatically cleaned up. For refresh to succeed, you must manually detach conflicting Oracle Homes and manually remove conflicting INST_TOP directories prior to refresh.

v. Enable the **Start Services After Provision** option if you want to permit the Delphix Engine to automatically start the services for EBS appsTier after provisioning.
   i. With this option enabled, the Delphix Engine will start all the services of EBS appsTier after provisioning.
   ii. Without this option enabled, the Delphix Engine will **NOT** start the services of EBS appsTier after provisioning. This will allow customers to prevent EBS services from being started at the conclusion of provisioning or refreshing. That way, customers can perform post-clone processing automation using a configure-clone hook without having to stop services first.

w. If you are provisioning a multi-node appsTier, enter additional appsTier nodes as **Additional Nodes**.
   i. Select the Environment for the secondary node.
   ii. The **Environment User** for each node should be the `applmgr` user-outlined in Preparing target EBS R12.2 environments for provisioning.
   iii. Ensure that the **Hostname** value for each node is the short hostname, not the fully-qualified hostname.
   iv. Provide DISPLAY Variable, the default value is hostname:0.0.
   v. The **Mount Path** is not configurable for each node individually. The **Mount Path** provided for the primary environment will be used for each additional node.

12. Click **Next**.
13. Enter a **VDB Name**.
14. Select a **Target Group** for the VDB. If necessary, click the **Plus** icon to add a new group.
15. Select a **Snapshot Policy** for the VDB. If necessary, click the **Plus**icon to create a new policy.
    Warning: When Snapshot is running against the dbTechStack, database, or appsTier, the Delphix Engine also executes pre-clone logic to ensure the latest configuration is staged in the captured snapshots. Unfortunately, if multiple Snapshots are running against the same EBS instance concurrently, this pre-clone logic may fail and produce bad snapshots.
    To avoid SnapSync conflicts, spread out your SnapSync policies for an EBS instance by one hour or more.
    Click **Next**.
16. Enter any **custom hook operations** that are needed to help correctly manage the virtual appsTier. The Configure Clone hook will be run after the `adcfgclone.pl` tool has both mounted and configured the appsTier. All hook operations run against the environment specified for provision. For a multi-node appsTier, hook operations never run against additional nodes specified.
17. Click **Next.**
18. Click **Submit**.
    Note: dbTier Must Be Accessible During appsTier Provisioning
    Post-clone configuration will fail if the appsTier cannot connect to the database. Ensure the target dbTier is accessible to the appsTier during the provisioning process. Ensure both the virtual database and database listener are running.
    Note:
    If your source setup is EBS ISG enabled, then during virtual AppsTier creation on the target server, perform post cloning steps as described in Installing Oracle E-Business Suite Integrated SOA Gateway, Release 12.2 (Doc ID 1311068.1).

19. For tips on monitoring the progress of appsTier provisioning, see Monitoring EBS R12.2 appsTier provisioning progress. Once all three EBS virtual datasets have been provisioned successfully, your virtual EBS instance should be running and accessible.

Rewind Operation on Snapshot

Perform the rewind operation on the snapshot after Appstier provisioning in the following sequence.

1. Take a snapshot of the Database VDB after Appstier provision.
2. Stop the appsTier, DB, and dbTechStack one by one.
3. Rewind dbTechStack, DB, and appsTier one by one.

If the DB snapshot was taken before appsTier provisioning, rewinding the DB will cause loss of new data updated by appsTier during provisioning and the next operation of appsTier rewind will fail. So, to ensure that the Rewind operation of appsTier succeeds, take a snapshot of DB after Appstier provision.

Managing data operations for virtual EBS instances

This section contains the following topics:

Starting and stopping a virtual EBS instance

This topic describes the process of starting and stopping a virtual Oracle E-Business Suite (EBS) instance.

You can start and stop virtual EBS instances through the **Delphix Management** application or through the standard Oracle command-line interface (CLI) utilities, `adstrtal` and `adstpall`. The Delphix Engine will show the dbTechStack and appsTier as running as long as there are processes using the dbTechStack and appsTier filesystem mounts on the target environments.

> ⚠ Be careful! Services running on the appsTier depend on the availability of services on the dbTier. The steps below are explicitly ordered with these dependencies in mind. Executing steps out of order may lead to errors in accessing the virtual EBS instance.

Stopping

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **appsTier vFiles** for your EBS instance.
5. Click the **Stop** icon. Stopping the appsTier will run Oracle's `adstpall.sh` utility. For multi-node appsTier, secondary nodes will be stopped sequentially, followed by the primary node.
   Note: Stopping the appsTier may take a long time. The Delphix Engine will wait for all Oracle application processes to exit before declaring the appsTier as stopped.
6. Select the **VDB** utilized by your EBS instance.
7. Click the **Stop** icon. This action will shutdown the database instance.
8. Select the **dbTechStack vFiles** hosting your virtual EBS database.
9. Click the **Stop** icon. Stopping the dbTechStack will shutdown the database listener.

Starting

1. Login to the **Delphix Management** application using **Delphix Admin** credentials.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **dbTechStack vFiles** hosting your virtual EBS database.
5. Click the **Start** icon.
6. Starting the dbTechStack will start the database listener.
7. Select the **VDB** utilized by your EBS instance.
8. Click the **Start** icon. Starting the database will open the database.
9. Select the **appsTier vFiles** for your EBS instance.
10. Click the **Start** icon. Starting the appsTier will run Oracle's `adstrtal.sh` utility. For multi-node appsTier, the primary node will be started first, followed by secondary nodes sequentially.

Rewinding a virtual EBS instance

This topic describes the process of rewinding a virtual Oracle E-Business Suite (EBS) instance.

> Changes applied to EBS and picked up only in some dSource snapshots may make certain combinations of snapshots across the appsTier and dbTier incompatible. When provisioning, refreshing or rewinding a virtual EBS instance, be sure the points in time you select for each dataset are compatible with each other.

> Be careful! Services running on the appsTier depend on the availability of services on the dbTier. The steps below are explicitly ordered with these dependencies in mind. Executing steps out of order may lead to errors in accessing the virtual EBS instance.

Prerequisites

The appsTier **Instance Home Directory** of the virtual EBS instance must reside under the specified **Mount Path**.

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **appsTier vFiles** for your EBS instance.
5. Click the **Stop** icon to shutdown the appsTier services.
6. Select the **VDB** utilized by your EBS instance.
7. Click the **Stop** icon to shutdown the database.
8. Select the **dbTechStack vFiles** hosting your virtual EBS database.
9. Click the **Stop** icon to shutdown the database listener.
10. Rewind the dbTechStack vFiles.
    a. Select a **snapshot**.
    b. Click the **Rewind** button.
11. Rewind the EBS VDB.
    a. Select a **snapshot**.
    b. Click the **Rewind** button.
12. Rewind the appsTier vFiles.
    a. Select a **snapshot**.
    b. Click the **Rewind** button.

Once you have rewound all three EBS virtual datasets successfully, your virtual EBS instance should be running and accessible.

Refreshing a virtual EBS instance

This topic describes the process of refreshing a virtual Oracle E-Business Suite (EBS) instance.

> Changes applied to EBS and picked up only in some dSource snapshots may make certain combinations of snapshots across the appsTier and dbTier incompatible. When provisioning, refreshing, or rewinding a virtual EBS instance, be sure the points in time you choose for each dataset are compatible with each other.

> Be careful! Services running on the appsTier depend on the availability of services on the dbTier. The steps below are explicitly ordered with these dependencies in mind. Executing steps out of order may lead to errors in accessing the virtual EBS instance.

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**
4. Select the **appsTier vFiles** for your EBS instance.
5. Click the **Stop** icon to shut down the appsTier services.
6. Select the **VDB** utilized by your EBS instance.
7. Click the **Stop** icon to shut down the database.
8. Select the **dbTechStack vFiles** hosting your virtual EBS database.
9. Click the **Stop** icon to shut down the database listener.
   Warning: If you did NOT specify the **Cleanup Before Provision** option for either your virtual dbTechStack or appsTier, you must manually clean up your target environments prior to refreshing. If you have specified this option for both datasets, no manual work is required.
   To manually clean up a target environment prior to refresh, remove instance-specific directories and or inventory entries that will conflict with files and entries recreated during the refresh. Without this clean-up, post-clone configuration performed during a refresh will fail with an error claiming that a conflicting EBS instance is already installed.
10. Refresh the dbTechStack vFiles.
    a. On the vFiles, click the **Refresh**.
    b. Select a **snapshot** from which to refresh.
11. Refresh the EBS VDB.
    a. On the VDB, click the **Refresh**.
    b. Select a **snapshot** from which to refresh.
12. Refresh the appsTier vFiles.
    a. On the vFiles, click the **Refresh**.
    b. Select a **snapshot** from which to refresh.

Once you have refreshed all three EBS virtual datasets successfully, your virtual EBS instance should be running and accessible.

Enabling and disabling a virtual EBS instance

This topic describes the process of enabling and disabling a virtual Oracle E-Business Suite (EBS) instance.

An enabled virtual EBS instance will be running and fully available to end-users. A disabled virtual EBS instance will be neither running nor mounted to the target environments.

Be careful! Services running on the appsTier depend on the availability of services on the dbTier. The steps below are explicitly ordered with these dependencies in mind. Executing steps out of order may lead to errors in accessing the virtual EBS instance.

Disabling

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **appsTier vFiles** for your EBS instance.
5. From the Actions menu (...) select **Disable**. Disabling the appsTier vFiles will stop the appsTier services and unmount the appsTier files.
   Note: Stopping the appsTier may take a long time. The Delphix Engine will wait for all Oracle application processes to exit before declaring the appsTier as stopped.
6. Select the **VDB** utilized by your EBS instance.
7. From the Actions menu (...) select **Disable**. Disabling the VDB will stop the database instance and unmount the data files.
8. Select the **dbTechStack vFiles** hosting your virtual EBS database.
9. From the Actions menu (...) select **Disable**. Disabling the dbTechStack vFiles will stop the database listener and unmount the dbTechStack files.

Once you have disabled all three EBS virtual datasets successfully, your virtual EBS instance should be fully removed from the target environment.

Enabling

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **dbTechStack vFiles** hosting your virtual EBS database.
5. From the Actions menu (...) select **Enable**. Enabling the dbTechStack vFiles will mount the dbTechStack files and start the database listener.
6. Select the **VDB** utilized by your EBS instance.
7. From the Actions menu (...) select **Enable.** Enabling the VDB will mount the data files and start the database instance.
8. Select the **appsTier vFiles** hosting your virtual EBS database.
9. From the Actions menu (...) select **Enable.** Enabling the dbTechStack vFiles will mount the appsTier files and start the application services.

Once you have enabled all three EBS virtual datasets successfully, your virtual EBS instance should be running and accessible.

Deleting a virtual EBS instance

This topic describes the process of deleting a virtual Oracle E-Business Suite (EBS) instance.

> ⚠ Be careful! Services running on the appsTier depend on the availability of services on the dbTier. The steps below are explicitly ordered with these dependencies in mind. Executing steps out of order may lead to errors in accessing the virtual EBS instance.

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **appsTier vFiles** for your EBS instance.
5. Delete the appsTier vFiles by clicking the **Trash Can** icon in the lower left-hand corner.
6. Select the **VDB** utilized by your EBS instance.
7. Delete the VDB by clicking the **Trash Can** icon in the lower left-hand corner.
8. Click **Manage**.
9. Select **Environments**.
10. Select the **target dbTier environment**.
11. Click the **Databases** tab.
12. In the list of **Installation Homes** on the environment, click the **Trash Can** icon next to the dbTechStack Oracle Home you want to delete.
13. Click **Manage**.
14. Select **Datasets**.
15. Select the **dbTechStack vFiles** for your EBS instance.
16. Delete the dbTechStack vFiles by clicking the **Trash Can** icon in the lower left-hand corner.
17. Clean up any files that the virtual EBS instance might have created outside of the Delphix mount points on the target environments. These typically include the instance-specific directories, oraInventory files, and oraTab entries.

Once you have deleted all three EBS virtual datasets successfully, your virtual EBS instance should be fully removed from the target environments.

Modifying the appsTier topology

This topic describes the process of modifying the appsTier topology of a virtual EBS instance.

appsTier Topology Changes Invalidate Rewind
If you modify the appsTier topology of a virtual EBS instance, all snapshots in existence prior to the modification will no longer be valid for rewind. The appsTier topology in the snapshots will no longer match the Delphix Engine's configuration and rewind targeting these snapshots will fail if attempted.

Procedure

1. Disable the virtual EBS instance by following the procedure outlined in Enabling and disabling a virtual EBS instance
2. Select the **appsTier vFiles** for your EBS instance.
3. On the back of the card, modify the **EBS-specific parameters** for the virtual appsTier. This process will normally entail adding or deleting **Additional Nodes** and configuring its corresponding **Services**.
4. Apply the configuration changes by refreshing the entire virtual EBS instance. Follow the procedure outlined in Refreshing a virtual EBS instance

Replicating a virtual EBS instance

This topic describes the process of replicating a virtual Oracle E-Business Suite (EBS) instance.

Replication

1. Login to the *source* Delphix Engine's **Delphix Management** application.
2. Configure replication between the source Delphix Engine and a target Delphix Engine. For a detailed outline of the replication process, see Configuring replication.
3. Select the **dbTechStack vFiles**, **VDB**, and **appsTier vFiles** objects to be replicated. These objects have dependencies on all other Delphix Engine objects relevant to the virtual EBS instance: you do not need to specify any additional objects for EBS replication.
4. Schedule or perform the replication.

Failover

1. Login to the *target* Delphix Engine's **Delphix Management** application**.**
2. Failover the replica **dbTechStack vFiles**, **VDB**, and **appsTier vFiles**. Failing over these object will sever future replication, but will not enable the datasets. For a detailed outline of the failover process, see Controlled failover
3. Enable the dbTier and appsTier environments.
    a. Click **Manage**.
    b. Select **Environments**.
    c. For each environment, from the Actions menu (...) select **Disable**.
4. Enable the virtual EBS instance by following the procedure outlined in Enabling and disabling a virtual EBS instance

Provisioning from a replicated EBS instance

You can provision new virtual EBS instances from replicated datasets regardless of whether these datasets have been failed over. The provisioning process for each version of EBS is outlined in Virtualizing Oracle E-Business suite. For more information about provisioning from replicated datasets, see Replicating a virtual EBS instance

Upgrading a Delphix Engine hosting a virtual EBS instance

This topic describes the process of upgrading a Delphix Engine hosting a virtual Oracle E-Business Suite (EBS) instance.

Procedure

1. Disable the virtual EBS instance by following the procedure outlined in Enabling and Disabling a Virtual EBS Instance
2. Once you have safely disabled the virtual EBS instance, upgrade the Delphix Engine by following the procedure outlined in Upgrading the Delphix Engine
3. Enable the virtual EBS instance by following the procedure outlined in Enabling and Disabling a Virtual EBS Instance

Migrating an EBS VDB

This topic describes how to migrate an EBS Virtual Database (VDB) from one target environment to another.

There may be situations in which you want to migrate a virtual database to a new target environment, for example when upgrading the host on which the VDB resides, or as part of a general data center migration. This is easily accomplished by first disabling the database, then using the Migrate VDB feature to select a new target environment.

There are three components for any EBS environment - dbTechStack, Database, and appsTier. Whenever we create virtual copies for any EBS environment, we provision these components in the following sequence: (1) dbTechStack (2) Database (3) appsTier.

We can **only** migrate the EBS database (VDB) with the following prerequisite.

Prerequisite

You should have already set up a new target environment that is compatible with the VDB that you want to migrate.

Procedure

1. Login to your Delphix Management application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **VDB** you want to migrate.
5. From the Actions menu (...) select **Disable.**
6. Click **Disable** to confirm. When the VDB is disabled, its icon will turn grey.
7. From the Actions menu (...) select **Migrate**.
8. Select the new target environment for the VDB, the user for that environment, and the database installation where the VDB will reside.
9. Select **Migrate** to confirm your selections.
10. From the Actions menu (...) select the **Enable**
11. Click **Enable**to confirm.
    Within a few minutes, your VDB will restart in the new environment, and you can continue using it.

- You need to provision the EBS dbTechStack on the new target environment with a similar DB name VDB that you want to migrate.
- EBS appsTier vFile cannot be migrated, you need to re-configure the EBS appsTier by modifying the custom configuration for Target DB server node information to the target environment where VDB is migrated.

Virtual EBS instance recipes

This section contains the following topics:

Configuring a Delphix self-service data template for EBS

This topic describes the process of configuring a Delphix Self-Service (Jet Stream) data template for Oracle E-Business Suite (EBS).

By configuring a data template for EBS, you eliminate the need to order operations applied across the EBS datasets.

The ordering of operations is a result of the dataset dependencies discussed in topics under Managing Data Operations of Virtual EBS Instances, such as Refreshing a Virtual EBS Instance.

Procedure

1. Create a Delphix Self-Service data template by following the procedure outlined in Understanding Data Templates.
    a. For EBS, the data template will have three data sources: the dbTechStack, database, and appsTier.
2. Be sure to set the following **ordering of the data sources** when creating the data template. This ordering will ensure that Delphix Self-Service operations do not violate the EBS dataset dependencies.

Order	Dataset
1	dbTechStack
2	Database
3	appsTier

Once you have created a Delphix Self-Service data template, you can configure Delphix Self-Service data containers to manage virtual EBS instances. Delphix Self-Service data containers will follow the ordering of data sources configured in the template. All Delphix Self-Service operations should works as expected for virtual EBS instances.

Managing the APPS password

This topic outlines how to manage the APPS password on a virtual EBS instance.

The Delphix Engine requires a virtual EBS instance's APPS password in order to manage its dbTechStack and appsTier. This password is encrypted when stored within the Delphix Engine and is exposed as an environment variable to RapidClone tools.

When a virtual EBS instance is being provisioned, the Delphix Engine's copy of the APPS password should match the source EBS instance's APPS password during its time of the snapshot. After provisioning, you can change the APPS password of the virtual instance in both EBS and the Delphix Engine.

Changing the APPS password

1. Provision a virtual EBS instance.
2. Change the APPS password in the virtual EBS instance. You can perform the password change using Oracle's `FNDCPASS` utility.
3. Stop the virtual appsTier manually using `adstpall`. You cannot stop the virtual appsTier through the Delphix Engine, because the APPS password being stored is now out of sync.
4. Disable the entire virtual EBS instance. For an outline of this process, see Enabling and Disabling a Virtual EBS Instance.
5. Change the APPS password on the dbTechStack.
    a. Select the **dbTechStack vFiles** hosting your virtual EBS database.
    b. On the back of the card, click the **Custom** tab.
    c. Edit the **APPS Password** field.
6. Change the APPS password in any hook operations defined on the EBS virtual database.
    a. Select the **VDB** utilized by your EBS instance.
    b. On the back of the card, click the **Hooks** tab.
    c. Edit the relevant hook operations. Typically, you will need to edit the Pre-Snapshot hook operation running `adpreclone`.
7. Change the APPS password on the appsTier.
    a. Select the **appsTier vFiles** hosting your virtual EBS database.
    b. On the back of the card, click the **Custom** tab.
    c. Edit the **APPS Password** field.
8. Enable the entire virtual EBS instance. For an outline of this process, see Enabling and Disabling a Virtual EBS Instance.

Refreshing or rewinding

The APPS password stored across individual snapshots of a virtual EBS instance will not be consistent after a password change. Old snapshots of EBS data will refer to a different APPS password than new snapshots of EBS data. To perform a refresh or rewind, you must explicitly manipulate the Delphix Engine's copy of the APPS password to ensure that the virtual EBS instance is being accessed with the correct APPS password at every step.

> Recipe Not Needed If Password Not Changed
> These steps are only necessary if the virtual EBS instance has a different APPS password than the snapshots being targeted by the refresh or rewind.
> If the APPS password has not been changed, follow the instructions in Refreshing a Virtual EBS Instance or Rewinding a Virtual EBS Instance.

1. Before refreshing or rewinding the virtual EBS instance, disable the entire virtual EBS instance. For an outline of this process, see Enabling and Disabling a Virtual EBS Instance.

2. Identify the APPS password for the snapshots being targeted by the refresh or rewind. Modify the virtual EBS instance to refer to this password.
    a. Change the APPS password on the dbTechStack.
        i. Select the **dbTechStack vFiles** hosting your virtual EBS database.
        ii. On the back of the card, click the **Custom** tab.
        iii. Edit the **APPS Password** field.
    b. Change the APPS password in any hook operations defined on the EBS virtual database.
        i. Select the **VDB** utilized by your EBS instance.
        ii. Select the Configuration tab, then select the **Hooks** tab.
        iii. Edit the relevant hook operations. Typically, you will need to edit the Pre-Snapshot hook operation running `adpreclone` and the Configure Clone operation running `adcfgclone`.
    c. Change the APPS password on the appsTier.
        i. Select the **appsTier vFiles** hosting your virtual EBS database.
        ii. Click the **Custom** tab.
        iii. Edit the **APPS Password** field.
3. Perform the refresh or rewind while the EBS instance is disabled. See Refreshing a Virtual EBS Instance or Rewinding a Virtual EBS Instance for an outline of these processes.

Refreshing a target dbTier environment

This topic describes how to properly refresh a target dbTier environment hosting an EBS VDB.

When an environment is initially registered with the Delphix Engine, a discovery process is responsible for registering resources such as Oracle Homes, listeners, and databases. When an environment is modified – for example, when an Oracle Home is upgraded or removed – you must run the discovery process again in order to register new resources and unregister removed ones.

The virtual dbTechStack plays a unique role in the Delphix Engine because it acts both as a virtual dataset AND a discovered resource on the target dbTier environment. Because of this unique role, you must mount and start this dataset (hosting the EBS database listener) before you refresh the target dbTier environment.

If you unmount or stop the virtual dbTechStack, the discovery process will assume that the EBS Oracle Home and database listener have been removed from the environment. Further interactions with the EBS VDB will result in an error, because neither the database listener nor Oracle Home appears available to the database.

Procedure

1. Prior to refreshing a target dbTier environment, ensure that the virtual dbTechStack is both enabled and started. For more information, see Enabling and Disabling a Virtual EBS Instance and Starting and Stopping a Virtual EBS Instance.
2. Refresh the target dbTier environment.

Managing the Weblogic password

Changing the Weblogic password

For existing EBS 12.2 customers who intend to use this WebLogic password change feature need to follow the below steps for their EBS application:

1. Disable all the virtual datasets created on Delphix Engine in the following order:
    a. appsTier vFiles
    b. VDB utilized by your EBS instance
    c. dbTechStack vFiles
2. Disable all the EBS dSource created on Delphix Engine once you have disabled all three EBS virtual datasets in the above step.
3. Upgrade Delphix Engine to 5.3.2.0 release and upload EBS 1.4.0 plugin release.
   Go to Manage>Plugins
    - Upload a plugin file to upgrade the existing plugin by clicking plus (+) symbol.
4. Refresh all host environments added on the Delphix Engine.
   Go to Manage> Environments
    - From the symbol (...) located in the upper left-hand corner, select Refresh All.
5. Change the Weblogic admin password on EBS appsTier vFile.
    - Select the appsTier vFiles hosting your virtual EBS database.
    - Go to Configurations >Custom
    - Update the **Target Weblogic AdminServer Password** field with the password you would like to set for EBS appsTier.
6. Enable all the EBS dSource created on Delphix Engine.
7. Enable all the virtual datasets created on Delphix Engine in the following order:
    a. appsTier vFiles
    b. VDB utilized by your EBS instance
    c. dbTechStack vFiles
8. Verify the changed WebLogic password by logging into the WebLogic console.

EBS 12.2 customers configuring appsTier on a multi-node environment will need to manually enable passwordless SSH on all target nodes *only when* they want to change the Weblogic password and then proceed with EBS appsTier provision. The passwordless SSH **should** be enabled between all nodes and for both low privileged and high privileged user.

Enabling passwordless SSH between target application nodes

To enable passwordless SSH between target application nodes complete the following

The following is an example of a two-node EBS appsTier.

Node 1: test1-ap1

Node 2: test1-ap2

1. Create Authentication SSH-Kegen Keys on Node 1.

```
[applvis@test1-ap1 ~]$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/applvis/.ssh/id_rsa):
Created directory '/home/applvis/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/applvis/.ssh/id_rsa.
Your public key has been saved in /home/applvis/.ssh/id_rsa.pub.
The key fingerprint is:
b8:1d:83:ef:40:d3:54:d6:50:2d:74:28:66:8a:4d:15 applvis@test1-ap1.dcenter.delphix.com
The key's randomart image is:
+--[ RSA 2048]----+
|          .E*oo.  |
|         .o+ +..  |
|        +.+ . .   |
|       .=o        |
|       = S        |
|      . = o       |
|       o o        |
|        o         |
|         .        |
+-----------------+
[applvis@test1-ap1 ~]$
```

2. Create .ssh Directory on Node 2.

```
[applvis@test1-ap1 ~]$ ssh applvis@test1-ap2 mkdir -p .ssh
The authenticity of host 'test1-ap2 (10.43.17.170)' can't be established.
RSA key fingerprint is 22:83:2b:d2:53:03:3e:b6:8b:fe:6a:87:21:35:d5:fc.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'test1-ap2,10.43.17.170' (RSA) to the list of known hosts.
applvis@test1-ap2's password:
[applvis@test1-ap1 ~]$
```

3. Upload Generated Public Keys to Node 2.

```
[applvis@test1-ap1 ~]$ cat .ssh/id_rsa.pub | ssh applvis@test1-ap2 'cat >> .ssh/authorized_keys'
applvis@test1-ap2's password:
[applvis@test1-ap1 ~]$
```

4. Set Permissions on Node 2 and Login to verify without password.

```
[applvis@test1-ap1 ~]$ ssh applvis@test1-ap2 "chmod 700 .ssh; chmod 640 .ssh/authorized_keys"
[applvis@test1-ap1 ~]$
[applvis@test1-ap1 ~]$ ssh test1-ap2
Last login: Mon Jan 28 21:46:12 2019 from 10.43.15.129
[applvis@test1-ap2 ~]$
[applvis@test1-ap2 ~]$ hostname -s
test1-ap2
```

Working with EBS snapshots

Taking a snapshot creates a new snapshot entry in the dSource's Timeflow. The plugin-managed dSource allows you to select the parameters before taking the snapshot. These parameters are provided as an input to pre-snapshot and post snapshot functions. On the other hand, in the default snapshot, the parameters are pre-defined in the dSource environment.

Snapshot (Default)

This section lists the steps to take a snapshot and delete the same.

1. Login to the **Delphix Management** application.
2. Click **Manage** and select **Datasets** from the dropdown list.
3. Select the dSource you want to Snapshot.
4. Click the **Camera** icon. Alternatively, click the arrow next to the Camera icon and select **Snapshot (default)**.
5. From the Snapshot dialog, select **Perform Snapshot**.
6. Click **View:** under **Timeflow** to verify the Snapshot you just created.
7. To delete, select the snapshot you just created, and from the Actions menu (...) select **Delete Snapshot**.
8. In the Delete Snapshot dialog, select **Delete**.
9. Click **View:** under **Timeflow** to verify the Snapshot, you just deleted.

Oracle EBS hook operations

Shell operations

RunCommand operation

The RunCommand operation runs a shell command on a Unix environment using whatever binary is available at `/bin/sh`. The environment user runs this shell command from their home directory. The Delphix Engine captures and logs all output from this command. If the script fails, the output is displayed in the Delphix Management application and command-line interface (CLI) to aid in debugging.

If successful, the shell command must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Examples of RunCommand operations

You can input the full command contents into the RunCommand operation.

```
remove_dir="$DIRECTORY_TO_REMOVE_ENVIRONMENT_VARIABLE"

if test -d "$remove_dir"; then
    rm -rf "$remove_dir" || exit 1
fi

exit 0
```

If a script already exists on the remote environment and is executable by the environment user, the RunCommand operation can execute this script directly.

```
/opt/app/oracle/product/10.2.0.5/db_1/dbs/myscript.sh "$ARG_ENVIRONMENT_VARIABLE"
"second argument in double quotes" 'third argument in single quotes'
```

RunBash operation

The RunBash operation runs a Bash command on a Unix environment using a `bash` binary provided by the Delphix Engine. The environment user runs this Bash command from their home directory. The Delphix Engine captures and logs all output from this command. If the script fails, the output is displayed in the Delphix Management application and command-line interface (CLI) to aid in debugging.

If successful, the Bash command must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Example of RunBash operations

You can input the full command contents into the RunBash operation.

```
remove_dir="$DIRECTORY_TO_REMOVE_ENVIRONMENT_VARIABLE"
```

```
# Bashisms are safe here!
if [[ -d "$remove_dir" ]]; then
    rm -rf "$remove_dir" || exit 1
fi

exit 0
```

Shell operation tips

Using `nohup`

You can use the `nohup` command and process backgrounding from resource in order to "detach" a process from the Delphix Engine. However, if you use `nohup` and process backgrounding, you MUST redirect `stdout` and `stderr.`

Unless you explicitly tell the shell to redirect `stdout` and `stderr` in your command or script, the Delphix Engine will keep its connection to the remote environment open while the process is writing to either `stdout` or `stderr` . Redirection ensures that the Delphix Engine will see no more output and thus not block waiting for the process to finish.

For example, imagine having your `RunCommand` operation background a long-running Python process. Below are the bad and good ways to do this.

Bad examples

- `nohup python file.py & # no redirection`
- `nohup python file.py 2>&1 & # stdout is not redirected`
- `nohup python file.py 1>/dev/null & # stderr is not redirected`
- `nohup python file.py 2>/dev/null & # stdout is not redirected`

Good examples

- `nohup python file.py 1>/dev/null 2>&1 & # both stdout and stderr redirected, Delphix Engine will not block`

Other operations

RunExpect operation

The RunExpect operation executes an Expect script on a Unix environment. The Expect utility provides a scripting language that makes it easy to automate interactions with programs which normally can only be used interactively, such as `ssh` . The Delphix Engine includes a platform-independent implementation of a subset of the full Expect functionality.

The script is run on the remote environment as the environment user from their home directory. The Delphix Engine captures and logs all output of the script. If the operation fails, the output is displayed in the Delphix Management application and CLI to aid in debugging.

If successful, the script must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Example of a RunExpect operation

Start an `ssh` session while interactively providing the user's password.

```
spawn ssh user@delphix.com
expect {
    -re {Password: } {
        send "${env(PASSWORD_ENVIRONMENT_VARIABLE)}\n"
    }
    timeout {
        puts "Timed out waiting for password prompt."
        exit 1
    }
}
exit 0
```

EBS environment variables

Operations that run user-provided scripts have access to environment variables. For operations associated with specific dSources or virtual databases (VDBs), the Delphix Engine will always set certain environment variables so that the user-provided script can use them to access the dSource or VDB.

dSource environment variables

Environment Variable	Description
`DLPX_DATA_DIRECTORY`	The primary mount path provided by the user on the UI. The first one, in case multiple mount specifications are provided.
`USER`	The OS user used to create the dSource.

VDB environment variables

Environment Variable	Description
`DLPX_DATA_DIRECTORY`	The primary mount path provided by the user on the UI. The first one, in case multiple mount specifications are provided.
`USER`	The OS user used to provision the VDB.

PostgreSQL environments and data sources

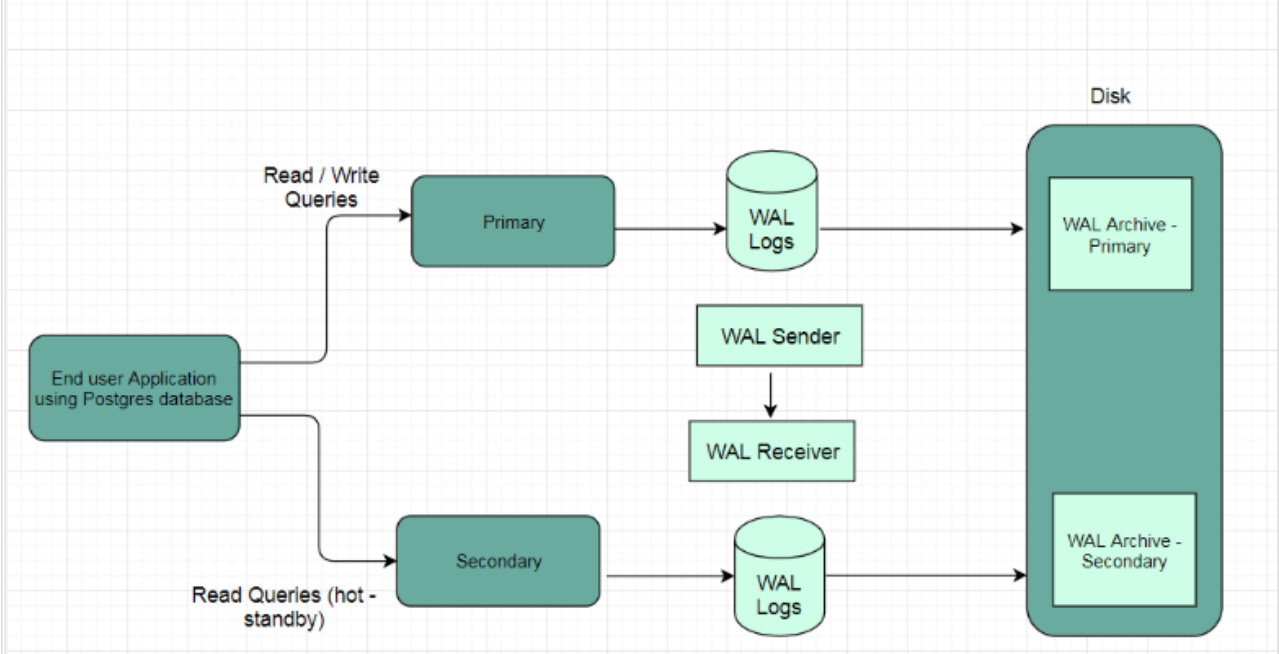
This section covers the following topics:

Delphix architecture with PostgreSQL

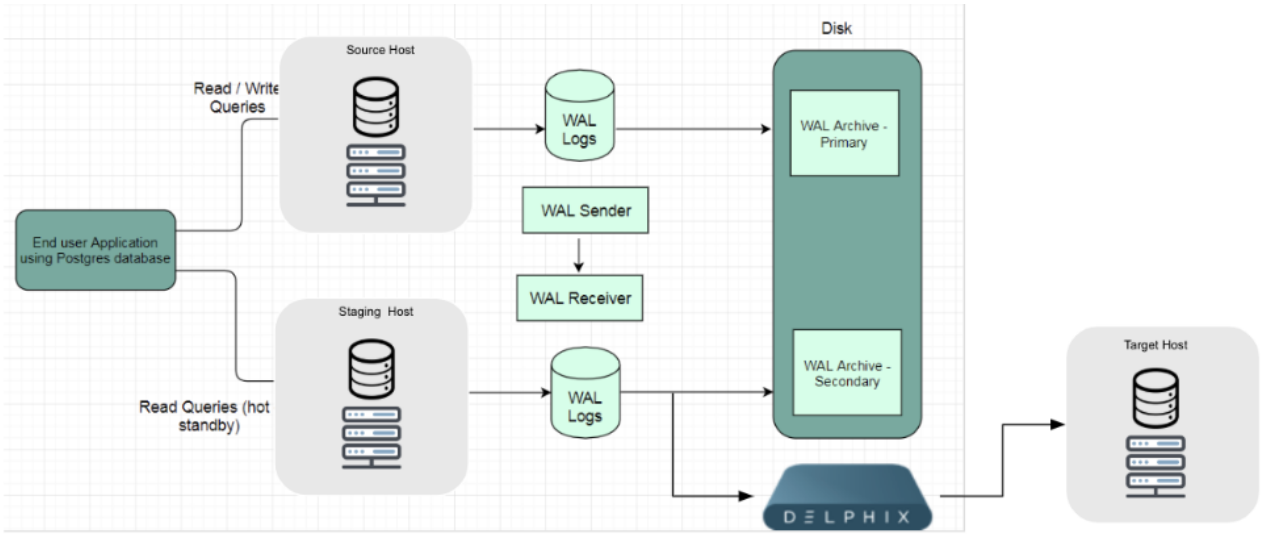
Primer: PostgreSQL replication methodology

Delphix achieves virtualization for the PostgreSQL database through leveraging PostgreSQL's streaming replication protocol between a Primary (parent) and Secondary (child) database configuration. Delphix uses the secondary, hot-standby database as a source database to ingest from.

In a typical PostgreSQL replication workflow, PostgreSQL leverages a parent and child configuration to maintain high-availability with Write Ahead Logs (WAL - a log file where all the modifications to the database are written before they're applied/written to data files) to maintain sync between both databases. This can be seen in the architectural diagram below.



Delphix leverages this feature by capturing the replication stream via file-based log shipping or Streaming WAL records depending on your PostgreSQL configuration. Below is the same PostgreSQL replication architecture diagram with Delphix use case overlayed:



Core concepts:

Write Ahead Logs - WAL:

- These are log files that record all modifications to the database before they're applied/written to data files.

File-Based Log Shipping:

- A method of syncing parent and child databases by applying WAL log files.
- One WAL log file can contain up to 16MB of data
- The WAL file is shipped only after it reaches that threshold
- Note: This will cause a delay in replication and also increase the chances of losing data if the parent crashes and logs are not archived.

Streaming WAL Records:

- WAL record chunks are streamed by database servers to keep data in sync.
- The standby server connects to the parent to receive the WAL chunks.
- The WAL records are streamed as they are generated.
- The streaming of WAL records need not wait for the WAL file to be filled.
- *This allows a standby server to stay more up-to-date than is possible with file-based log shipping.*
- By default, streaming replication is asynchronous even though it also supports synchronous replication.

Delphix supports both methods of ingestion (Log Shipping and Streaming WAL Records). It is important to note that both methods have pros/cons. For example, streaming replication doesn't have as much lag between parent and child, as records are sent as they are generated. However, streaming requires both parent and child to be online and able to communicate directly. It also requires the replica to keep up well enough that the parent still has on-disk copies of the WAL the replica needs, and generally requires you to spend extra pg_xlog space on retaining extra WAL for the replica.

PostgreSQL architecture for streaming replication

Transaction logs - WAL (Write Ahead Logs) - are replayed from source to the staging environment to maintain data sync between the two.

Steps involved in supporting this architectural setup:

1. The end-user application connecting the PostgreSQL source database may perform read/write queries on the database.

2. The database changes are recorded as WAL segments in the PostgreSQL database under the directory **pg_xlog**.
3. Set up the source PostgreSQL server as Standby Node.
4. Configure replication security by creating a replication user and specifying the authentication protocol.
5. Initiate a base backup on secondary.
6. Configure postgresql.conf file as per the source environment.
7. Start the standby server.

WAL receiver

The WAL receiver process at the secondary continuously listens for any incoming WAL segments from primary in its receive queue and applies the same on the source database.

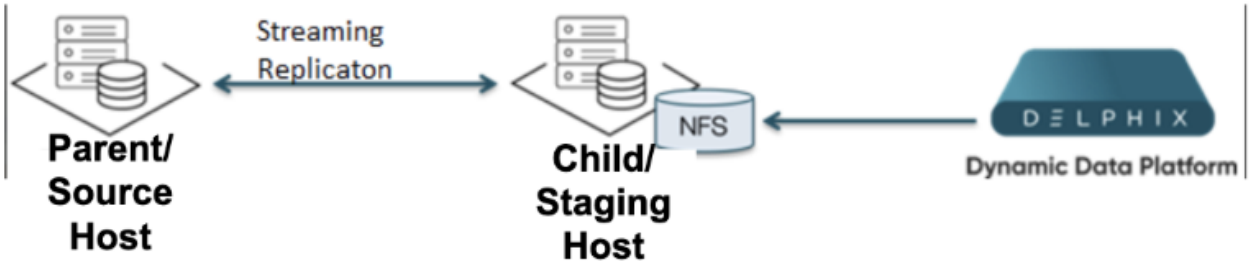
WAL segments

WAL segments are archived when a segment size reaches 16 MB (default).

Delphix architecture with postgreSQL

This topic describes the high-level process for adding PostgreSQL environments, linking PostgreSQL data sources to the Delphix Engine, and provisioning virtual databases from PostgreSQL data sources.

Given the preceding primer on Streaming Replication and WAL Logs, A high-level breakdown of the PostgreSQL/ Delphix Architecture breaks down as follows:



To match Delphix Terminology the PostgreSQL-denoted parent Host will be referred to as the Source Host and the PostgreSQL-denoted child Host will be referred to as the Staging Host.

For greater detail - Delphix has three ingestion options that leverage a combination of the PostgreSQL Replication Syncing Method as well as backup ingestion initiation:

1. Delphix Initiated Backup Ingestion with PostgreSQL Streaming Replication.
2. Customer Initiated Backup Ingestion with PostgreSQL Streaming Replication.
3. Customer Initiated Backup Ingestion with WAL log Shipping.

Below is a brief breakdown of the pros/cons of the three ingestions options:

Ingestion method	Backup mechanism	Syncing mechanism	Delphix considerations
Delphix Backup + Streaming Replication	The initial backup is taken by Plugin through the `pg_basebackup` method.	Streaming ensures that source and staging databases are closely synched	This method ensures that Delphix has the capability to ingest as frequently as WAL logs are generated by the Source. This leads to less snapshots than a customer-initiated backup, but those snapshots are aligned to policy.
Customer Backup + Streaming Replication	The initial backup is provided by the customer. Backup is taken through the `pg_basebackup` method.	Streaming ensures that source and staging databases are closely synched	This method ensures that the Delphix Engine dSource is always closely aligned to the Source Database
Customer Backup + Log Shipping	The initial backup is provided by the customer. Backup is taken through the `pg_basebackup` method.	The less bandwidth-intensive method which does not require source and staging databases to be directly connected at all times	This method does not require all resources to be concurrent, which leads to a less resource-intensive setup.

Environment setup

Linking architecture between **PostgreSQL and Delphix Engine**

At a high level, the Delphix Engine maintains a logical representation of the source database files, from which one can provision virtual databases (VDBs). In order to link a data source and provision a VDB, the following types of environments are required:

PostgreSQL provisioning architecture - source environment - to - Delphix engine

A source environment is where the un-virtualized source database runs. The Delphix Engine uses the backup, restore, and replication features of the PostgreSQL DBMS to maintain its internal representation of the source database, to be used for provisioning VDBs. The Delphix Engine must be able to connect to the source environment in order to discover each running source database and to orchestrate the backup, restore, and replication functionality necessary to keep its representation synchronized with the source database. The Delphix Engine is designed to have a minimal impact on the performance of the source database and the source environment.

Target environments for PostgreSQL

A target environment is where virtualized databases run. PostgreSQL target environments serve two purposes:

1. Since PostgreSQL does not provide a native incremental backup API, a warm standby server (in other words, one in log-shipping mode) must be created where all database files are stored remotely on a Delphix Engine. We refer to the process of applying those WAL files to the staging database as validated sync. During validated sync, we retrieve shipped WAL logs from the source, roll the staging database forward, and create a snapshot of that data state.
2. Once a source database has been set up, you can provision virtual databases from any of the discrete snapshots along the Timeflow mentioned above to any compatible target environment (for more information, see Requirements for PostgreSQL Target Hosts and Databases). Database files are exported over the network to the target environment, where the virtual database instance runs.

Workflow for PostgreSQL environments

Linking environments

Prior to linking a data source, both the source environment and a compatible target environment (to be used for the source database mentioned above) must be added to the Delphix Engine. Prior to provisioning a virtual database, a target environment (with similar settings/versioning to the source environment) must be added to the Delphix Engine.

TARGET hOSTS for PostgreSQL

Container for VDBs

This topic describes the basic concepts involved with provisioning VDBs from Postgres dSources or even other Postgres VDBs.

Once an environment is added to the Delphix Engine, environment discovery takes place. Environment discovery is the process of enumerating PostgreSQL installations and configurations when a source or target environment is added to the Delphix Engine. The discovery process is repeated during environment refresh in order to detect new PostgreSQL installations and clusters.

A dSource is a copy of a physical database that is created when the Delphix Engine links to and loads the database. The Delphix Engine keeps the dSource in sync with the source database in order to facilitate the provisioning of Virtual Databases (VDBs) from the dSource's TimeFlow. While creating a dsource, the Delphix Engine initiates a full database backup of the source database by running pg_basebackup on the source host. The initial snapshot of the dSource is derived from this backup.

After obtaining the initial snapshot and linking the dSource, the Delphix Engine keeps the source and staging database in sync by monitoring the source database for new transaction logs on the source host and then applying those transaction logs on the source database. PostgreSQL streaming replication protocol is used to achieve data replication between the source database and the staging database.

When provisioning a VDB, Delphix creates the database with the default Postgres database options. If the database options have been altered on the source database and you wish for the VDB to reflect these same options, they would need to be altered via a Post-Script or by hook operations.

Dependencies

If there are dependencies on the snapshot, you will not be able to delete the snapshot free space; the dependencies rely on the data associated with the snapshot.

Quick start guide for PostgreSQL

Linking a PostgreSQL data source

This quick start guide, which is excerpted from the larger User Guide, is intended to provide you with a quick overview of working with PostgreSQL database objects in the Delphix Engine. It does not cover any advanced configuration options or best practices, which can have a significant impact on performance. It assumes that you are working in a Lab/Dev setting and attempting to quickly test Delphix functionality. It assumes you will use the VMware Hypervisor.

Overview

In this guide, we will walk through deploying a Delphix Engine, starting with configuring Source and Target database environment. We will then create a dSource, and provision a VDB.

For purposes of the QuickStart, you can ignore any references to Replication or Masking, such as the engines shown in the diagram below.

Deploy OVA on VMware

Use the Delphix-supplied OVA file to install the Delphix Engine. The OVA file is configured with many of the minimum system requirements. The underlying storage for the install is assumed to be redundant SAN storage.

1. Download the OVA file from https://download.delphix.com. You will need a support login from your sales team or a welcome letter.
    a. Navigate to the Delphix Product Releases/ /appliance="" images="">
2. Login using the vSphere client to the vSphere server (or vCenter Server) where you want to install the Delphix Engine.
3. In the vSphere Client, click **File**.
4. Select **Deploy OVA Template**.
5. Browse to the OVA file.
6. Click **Next**.
7. Select a **hostname** for the Delphix Engine. This hostname will also be used in configuring the Delphix Engine network.
8. Select the **data center** where the Delphix Engine will be located.
9. Select the **cluster** and the **ESX host**.
10. Select one (1) **data store** for the **Delphix OS**. This datastore can be **thin-provisioned** and must have enough free space to accommodate the 127GB comprising the Delphix operating system.
11. Select four (4) or more **data stores** for Database Storage for the Delphix Engine. The Delphix Engine will stripe all of the Database Storage across these VMDKs, so for optimal I/O performance, each VMDK must be equal in size and be configured **Thick Provisioned - Eager Zeroed**. Additionally, these VMDKs should be distributed as evenly as possible across all four SCSI I/O controllers.
12. Select the **virtual network** you want to use. If using multiple physical NICs for link aggregation, you must use vSphere NIC teaming. Do not add multiple virtual NICs to the Delphix Engine itself. The Delphix Engine should use a single virtual network. For more information, see Optimal Network Architecture for the Delphix Engine
13. Click **Finish**. The installation will begin and the Delphix Engine will be created in the location you specified.
14. Once the installation has completed, power on the Delphix Engine and proceed with the initial system configuration as described in Setting Up Network Access to the Delphix Engine.

If your source database is 4 TB, you probably need 4 TB of storage for the Delphix Engine. Add at least 4 data disks of similar size for the Delphix VM. For example: for a source database of 4 TB, create 4 VMDKs of 1 TB each.

For a full list of requirements and best practice recommendations, see Virtual Machine Requirements for VMware Platform

Setup network access to the Delphix Engine

1. Power on the Delphix Engine and open the Console.
2. Wait for the Delphix Management Service and Delphix Boot Service to come online. This might take up to 10 minutes during the first boot. Wait for the large orange box to turn green.
3. Press any key to access the sysadmin console.
4. Enter `sysadmin@SYSTEM` for the username and `sysadmin` for the password.
5. You will be presented with a description of available network settings and instructions for editing.

```
Delphix Engine Network Setup To access the system setup through the browser,
the system must first be configured for networking in your environment. From
here, you can configure the primary interface, DNS, hostname, and default
 route. When DHCP is configured, all other properties are derived from DHCP
settings. To see the current settings, run "get." To change a property, run
"set =." To commit your changes, run "commit." To exit this setup and return to
the standard CLI, run "discard." defaultRoute    IP address of the gateway for
 the default route -- for example, "1.2.3.4." dhcp            Boolean value
indicating whether DHCP should be used for the primary interface. Setting this
 value to "true" will cause all other properties (address, hostname, and DNS)
to be derived from the DHCP response dnsDomain       DNS Domain -- for example,
"delphix.com" dnsServers      DNS server(s) as a list of IP addresses -- for
 example, "1.2.3.4,5.6.7.8." hostname        Canonical system hostname, used in
alert and other logs -- for example, "myserver" primaryAddress  Static address
for the primary interface in CIDR notation -- for example, "1.2.3.4/22" Current
settings: defaultRoute: 192.168.1.1 dhcp: false dnsDomain: example.com
dnsServers: 192.168.1.1 hostname: Delphix primaryAddress: 192.168.1.100/24
```

6. Configure the `hostname`. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set hostname=<hostname>
```

Note:
Use the same `hostname` you entered during the server installation.

7. Configure DNS. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set dnsDomain=<domain> delphix network setup
update *> set dnsServers=<server1-ip>[,<server2-ip>,...]
```

8. Configure either a static or DHCP address.
DHCP Configuration

```
delphix network setup update *> set dhcp=true
```

Static Configuration

```
delphix network setup update *> set dhcp=false delphix network setup update *>
set primaryAddress=<address>/<prefix-len>
```

Note:
The static IP address must be specified in CIDR notation (for example, `192.168.1.2/24`)

9. Configure a default gateway. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set defaultRoute=<gateway-ip>
```

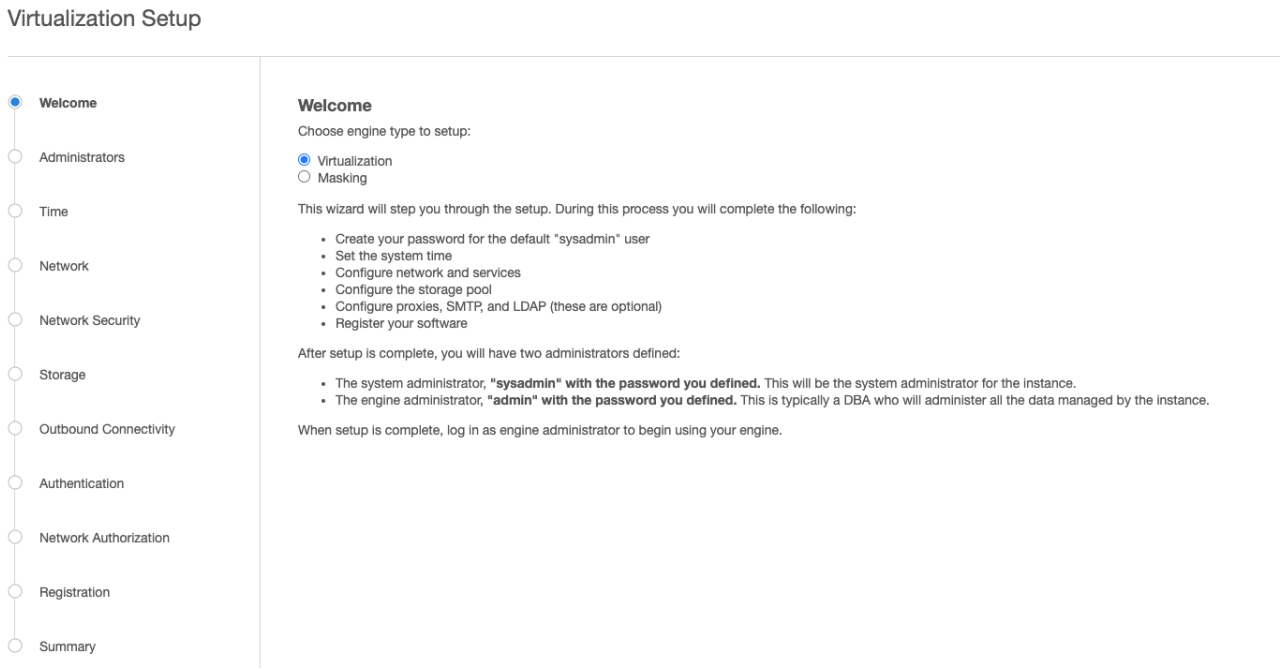
10. Commit your changes. Note that you can use the `get` command prior to committing to verify your desired configuration.

```
delphix network setup update *> commit Successfully committed network settings.
Further setup can be done through the browser at: http://<address> Type "exit"
to disconnect, or any other commands to continue using the CLI.
```

11. Check that you can now access the Delphix Engine through a Web browser by navigating to the displayed IP address, or hostname if using DNS.
12. Exit setup.

```
delphix> exit
```

Setting up the Delphix Engine



The setup procedure uses a wizard process to take you through a set of configuration screens:

- Administrators
- Time
- Network
- Network Security
- Storage
- Outbound Connectivity
- Authentication
- Network Authorization
- Registration
- Summary

1. Connect to the Delphix Engine at `http:///login/index.html#serverSetup`. The **Delphix Setup** application will launch when you connect to the server. Enter your **sysadmin** login credentials, which initially defaults to the username **sysadmin**, with the initial default password of **sysadmin**. On first login, you will be prompted to change the initial default password.
2. Click **Next**.

Administrators

The Delphix Engine supports two types of administrators:

- System Administrator (**sysadmin**) - this is the engine system administrator. The sysadmin password is defined here**.**
- Engine Administrator (**admin)** - this is typically a DBA who will administer all the data managed by the engine.

On the Administrators tab, you set up the sysadmin password by entering an email address and password. The details for the admin are displayed for reference.

⚠ The default domain user created on Delphix Engines from 5.3.1 is known as **admin** instead of delphix_admin. When engines created before 5.3.1 are upgraded to 5.3.1 or later they will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then Disabling delphix_admin.

Time

The engine time is used as the baseline for setting policies that coordinate between virtual databases and external applications

Choose your option to set up system time in this section. For a Quick Start, simply set the time and your timezone. You can change this later.

Network

The initial out-of-the-box network configuration in the OVA file is set to use a single VMXNET3 network adapter.

You have already configured this in the initial configuration. Delphix supports more advanced configurations, but you can enable those later.

Storage

You should see the data storage VMDKs or RDMs you created during the OVA installation. Click **Next** to configure these for data storage.

Serviceability

Choose your options to configure serviceability settings.

For a Quick Start, accept the defaults. You can change this later.

Authentication service

Choose your options to configure authentication services.

For a Quick Start, accept the defaults. You can change this later.

Registration

If the Delphix Engine has access to the external Internet (either directly or through a web proxy), then you can auto-register the Delphix Engine:

1. Enter your **Support Username** and **Support Password**.
2. Click **Register**.

If external connectivity is not immediately available, you must perform manual registration.

1. Copy the **Delphix Engine registration code** in one of two ways:
    a. Manually highlight the registration code and copy it to clipboard. Or,
    b. Click **Copy Registration Code to Clipboard**.
2. Transfer the Delphix Engine's registration code to a workstation with access to the external network Internet. For example, you could e-mail the registration code to an externally accessible e-mail account.
3. On a machine with access to the external Internet, please use your browser to navigate to the Delphix Registration Portal at http://register.delphix.com
4. Login with your Delphix support credentials (username and password).
5. Paste the **Registration Code**.
6. Click **Register**.

Although your Delphix Engine will work without registration, we strongly recommend that you register each engine as part of the setup. Failing to register the Delphix Engine will impact its supportability and security in future versions.

To regenerate the registration code for a Delphix Engine please refer to, Regenerating the Delphix Engine Registration Code. Delphix strongly recommends that you regenerate this code and re-register the engine regularly to maximize the Support Security of the Delphix Engine. Delphix recommends doing this every six months.

Summary

The final summary tab will enable you to review your configurations for System Time, Network, Storage, Serviceability, and Authentication.

1. Click the **Back** button to go back and to change the configuration for any of these server settings.
2. If you are ready to proceed, then click **Submit**.
3. Click **Yes** to confirm that you want to save the configuration.
4. Click **Setup** to acknowledge the successful configuration.
5. There will be a wait of several minutes as the Delphix Engine completes the configuration.

Requirements for PostgreSQL source databases

Source database requirements

- The Delphix source database can be in READ-WRITE or READ only mode. It can be a standby database(streaming site) as well.
- The Staging host must have access to a PostgreSQL role on the source side that has replication, and login privileges. This can be the built-in PostgreSQL role or a newly-created role (for example, delphix).

```
To create a new role for use with the Delphix Engine, use the following
command: SQL> CREATE USER delphix WITH REPLICATION ENCRYPTED [ PASSWORD
'password'];
```

- You must make the following changes to postgresql.conf (for more information, see the Server Configuration chapter in the PostgreSQL documentation):
- TCP/IP connectivity must be configured to allow the role mentioned above to connect to the source database from the Delphix Engine and from the standby DBMS instance set up by the Delphix Engine on the staging environment. This can be done by modifying the parameterlisten_addresses, which specifies the TCP/IP addresses on which the DBMS is to listen for connections from client applications.

```
listen_addresses = '*'             # defaults to 'localhost'; use '*' for all
```

- The value of max_wal_senders, which specifies the maximum number of concurrent connections from standby servers or streaming base backup clients, must be increased from its desired value by four. That is, in addition to the allowance of connections for consumers other than the Delphix Engine, there must be an allowance for four additional connections from consumers set up by the Delphix Engine.

```
The default value max_wal_senders is zero, meaning replication is disabled. In
this configuration, the value ofmax_wal_senders must be increased to two for
 the Delphix Engine: max_wal_senders = 4       # Default is 0
```

- The value of, wal_level which determines how much information is written to the write-ahead log (WAL), must be set to archive or hot_standby to allow connections from standby servers. The valuelogicalwal_level (introduced in PostgreSQL 9.4) is also supported.

```
The default value of wal_level is minimal, which writes only the information
needed to recover from a crash or immediate shutdown to the WAL archives. In
this configuration, you must add the logging required for WAL archiving as
follows: wal_level = archive       # Default is minimal
```

- In releases prior to PostgreSQL 9.6.x, parameter 'wal_level' allowed the values 'archive' and 'hot_standby'. These values are still accepted but mapped to 'replica'. For example, The plugin will work with ALTER SYSTEM SET wal_level TO 'replica' and ALTER SYSTEM SET wal_level TO 'hot_standby'.
- You must configure PostgreSQL to allow PostgreSQL client connections and replication client connections from the staging target environment. To configure appropriately, add the following entries to pg_hba.conf:

```
host all <role> <ip-address_of_delphix_engine>/32 <auth-method> host all <role>
<ip-address_of_staging_target>/32 <auth-method> host replication <role> <ip-
address_of_staging_target>/32 <auth-method>
```

- <auth-method> can be set to trust, scram-sha-256, or md5 on source. Delphix inherits the same authentication method from the source while dSource creation and VDB provisioning. For more information on how to configure, pg_hba.conf see the Client Authentication chapter in the PostgreSQL documentation.
- `wal_keep_segments` parameter in postgresql.conf file on the Source environment should be large enough to support the WAL sync process once the dSource is building up. Postgres 13 onwards, the configuration parameter `wal_keep_segments` is changed to `wal_keep_size`. It is specified in megabytes rather than the number of files as with the `wal_keep_segments` parameter. If you previously used `wal_keep_segments`, the following formula will give you an approximate equivalent setting: `wal_keep_size` = `wal_keep_segments` * `wal_segment_size` (typically 16MB).
- It is mandatory to have "port" (can be commented or not commented) in postgresql.conf file.

Requirements for PostgreSQL target hosts and databases

Target environment requirements

1. The operating system and architecture of the target environment must match those of the source environment. It is recommended that the source and the target environments should be identical and hardware configurations should match.
2. For supported OS version and DBMS version, see PostgreSQL Matrix. The underlying Operating System for both the Source and Staging environment should be amongst these two versions.
3. There must be an installation of PostgreSQL on the target environment that is compatible with the installation of PostgreSQL on the source environment. Two installations of PostgreSQL are compatible if and only if:
    a. They share the same vendor (for example, PostgreSQL is incompatible with EnterpriseDB Postgres Plus Advanced Server).
    b. They share the same major version number (for example, 9.5.4 is compatible with 9.5, 9.5.3; however, it is incompatible with 9.3, 9.3.24, or 9.2).
    c. They are compiled against the same architecture (in other words, 32-bit and 64-bit installations of Postgres are incompatible).

   d. They are compiled with the same WAL segment size. The default WAL segment size of 16 MB is rarely changed in practice, so almost all installations of PostgreSQL are compatible with each other in terms of WAL segment size.
4. There must be an operating system user (e.g. PostgreSQL) with the following privileges:
   a. The Delphix Engine must be able to make an SSH connection to the target environment using the operating system user.
   b. The operating system user must have read and execute privileges on the PostgreSQL binaries installed on the target environment.
   c. The operating system user must have permission to run mount and umount as the superuser via sudo with neither a password nor a TTY. See Sudo Privilege Requirements for PostgreSQL Environments for examples of the /etc/sudoers file on different operating systems.
5. There must be a directory on the target environment where the Delphix Engine Plugin can be installed (for example, /var/tmp) with the following properties:
   a. The Plugin directory must be writable by the operating system user mentioned above.
   b. The Plugin directory must have at least 1.5 GB of available storage.
6. There must be a mount point directory (for example, /mnt) that will be used as the base for mount points that are created when provisioning a VDB with the following properties:
   a. The mount point directory must be writable by the operating system user mentioned above.
   b. The mount point directory should be empty.
7. TCP/IP connectivity to and from the source environment must be configured as described in General Network and Connectivity Requirements.
8. Hostname and IP must be correctly set in /etc/hosts file, for example [postgres@source ~]$ hostname -i
   The output of “hostname -i” command should produce the correct result as the IP address of the server. For example:
   [postgres@source postgres]$ hostname -i 10.110.207.113
9. When using the External Backup method for ingestion the directories being used for keeping the Backup file and WAL file should be accessible by OS User.
10. Zip must be installed on the Staging/Target Host for External Backup Support.
11. The backup command and File should be in the below format: PostgreSQL_T<YYYYMMDD>.zip where YYYY denotes Year, MM denotes month, DD denotes date. For example PostgreSQL_T20190314.zip. The above zip file should contain a Database Full Backup from the Source created in tar format. The below flags can be used with pg_basebackup for taking Source Database backup: **For PostgreSQL versions 9.4.x, 9.5.x, and 9.6.x** <*PATH_TO_PG_BIN*>/pg_basebackup -p<*PORT*>-D<*EMPTY_BACKUP_DIR*>-F t -v -w -P -x**For PostgreSQL versions 10.x and 11.x** <*PATH_TO_PG_BIN*>/pg_basebackup -p <*PORT*> -D <*EMPTY_BACKUP_DIR*> -F t -v -w -P -Xs
12. If there's no PostgreSQL instance running on the Target host, however, we have a PostgreSQL installation, even then the plugin relies on the "DELPHIX_PG_PATH" variable to discover the environment. In the absence of "DELPHIX_PG_PATH" variable or if the value of "DELPHIX_PG_PATH" variable is NULL then Linux "find" command will be used for the Environment discovery which may impact the overall performance. Hence, it is preferred to create this Environment Variable with correct entries. In order to optimize the performance, it is preferred to create an Environment Variable "DELPHIX_PG_PATH" which should be accessible by OS users. Below should be the syntax for "DELPHIX_PG_PATH" variable:
    DELPHIX_PG_PATH="binary_path:data_path1;binary_path:data_path2;"
    For example: DELPHIX_PG_PATH="/usr/pgsql-9.6/bin:/var/lib/pgsql/9.6/data;/opt/edb/as9.6/bin:/opt/edb/as9.6/data;/usr/pgsql-9.6/bin:/tmp/TESING/data;"

The variable "**DELPHIX_PG_PATH**" must be available to the environment user in a non-interactive way. You can test this variable for non-interactive logins using `ssh your_username@v-cardtr.tmydb.kv.aval "env | grep DELPHIX_PG_PATH".`

Adding a PostgreSQL environment

Prerequisites

- Make sure that the staging environment in question meets the requirements described in Requirements for PostgreSQL Hosts and Databases and Prerequisites for Discovery operation described in the Prerequisites for Privilege Elevation using DLPX_DB_EXEC Script section.

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Next, to Environments, click the Actions (**...**) menu and select **Add Environment**.
5. In the Add Environment dialog, select **Unix/Linux.**
6. Select **Standalone Host**.
7. Click **Next**.
8. Enter Name for the Environment.
9. Enter the **Host IP address** or **hostname**.
10. Enter the **SSH port**. The default value is 22.
11. Enter an **OS Username** for the Environment. If a low-privileged OS user is used, make sure the Prerequisites for Privilege Elevation using DLPX_DB_EXEC Script are met.
12. Select **Login Type**. — Username and Password - enter the OS username and password — Username and Public Key - enter the OS username. — Password Vault - select from an existing Enterprise Password Vault
    Info:
    Using Public Key Authentication
    If you want to use public-key authentication for logging into your Unix-based environment, there are two options: use the engine's key pair or provide a key pair for this environment.
    To use the engine's key pair:
    a. Select **Public Key** for the **Login Type.**
    b. Click **View Public Key**.
    c. Copy the public key that is displayed, and append it to the end of your `~/.ssh/authorized_keys` file. If this file does not exist, you will need to create it.
        i. Run chmod 600 ~/.ssh/authorized_keys to allow only the file's owner to read and write to it (make sure the file is owned by the user).
        ii. Run chmod 755 ~ to restrict access to the user's home directory so no other user may write to it.
        iii. Run chmod 700 ~/.ssh so that others cannot write to it. The ~/.ssh directory cannot be writable by group or other users. Otherwise, authentication will fail.

    As an alternative, you can provide a key pair specific for this environment via the API or CLI. See Option 2 in this CLI Cookbook article for instructions.
13. For Password, enter the password associated with the user in step 11.
14. If you want to use Public Key Encryption for logging into your environment:
    a. Select Public Key for the Login Type.
    b. Click View Public Key.
    c. Copy the public key that is displayed, and append it to the end of your `~/.ssh/authorized_keys` file. If this file does not exist, you will need to create it.
        i. Run chmod 600 ~/.ssh/authorized_keys to allow only the file's owner to read and write to it (make sure the file is owned by the user).
        ii. Run chmod 755 ~ to restrict access to the user's home directory so no other user may write to it.

iii. Run chmod 700 ~/.ssh so that others cannot write to it. The ~/.ssh directory cannot be writable by group or other users. Otherwise, authentication will fail.
The public key needs to be added only once per user and per environment.
You can also add public key authentication to an environment user's profile by using the command-line interface, as explained in the topic CLI Cookbook: Setting Up SSH Key Authentication for UNIX Environment Users

15. For Password Login, click Verify Credentials to test the username and password.
16. Enter **Toolkit Path** (make sure Toolkit path does not have spaces).
17. To provide your own Oracle Java select the **Provide my own JDK** checkbox and click **Next**.
18. In the Java Development Kit tab enter the absolute path to your Oracle JDK and click **Next**.
19. Click **Submit**.

As the new environment is added, you will see two jobs running in the Delphix platform Job History, one to Create and Discover an environment, and another to create an environment. When the jobs are complete, you will see the new environment added to the list in the Environments tab. If you do not see it, click the Refresh icon in your browser.

Once the environment is discovered, further linking would require adding a source config to the above-discovered installation. Please refer to Linking a PostgreSQL dSource for more information.

Linking a PostgreSQL data source

Prerequisites

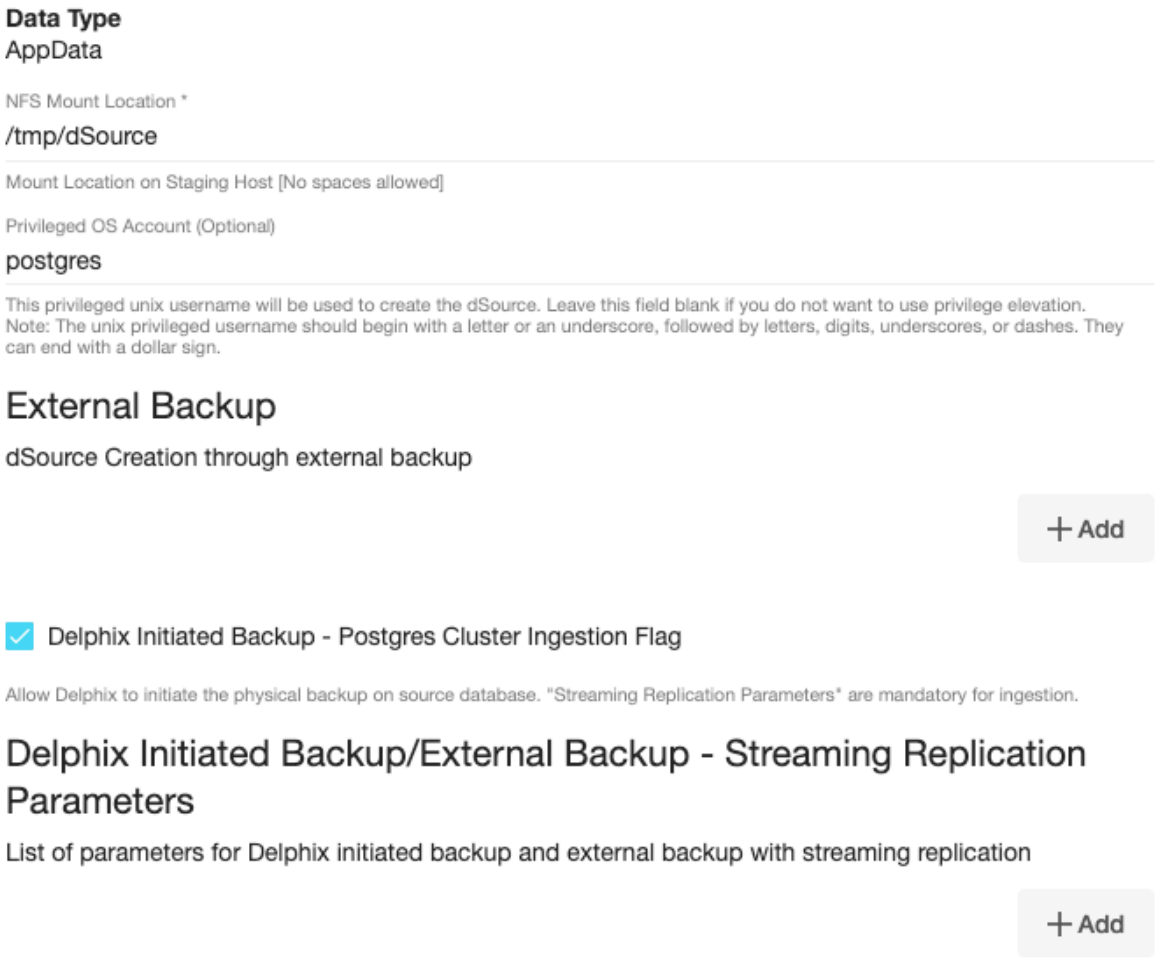
- The source and staging instances must meet the host requirements
- Databases must meet container requirements. For more information refer to Requirements for PostgreSQL Source Hosts and Databases

Limitation

It is not possible to access the staging server with PostgreSQL 9.4 and PostgreSQL 9.5 versions.

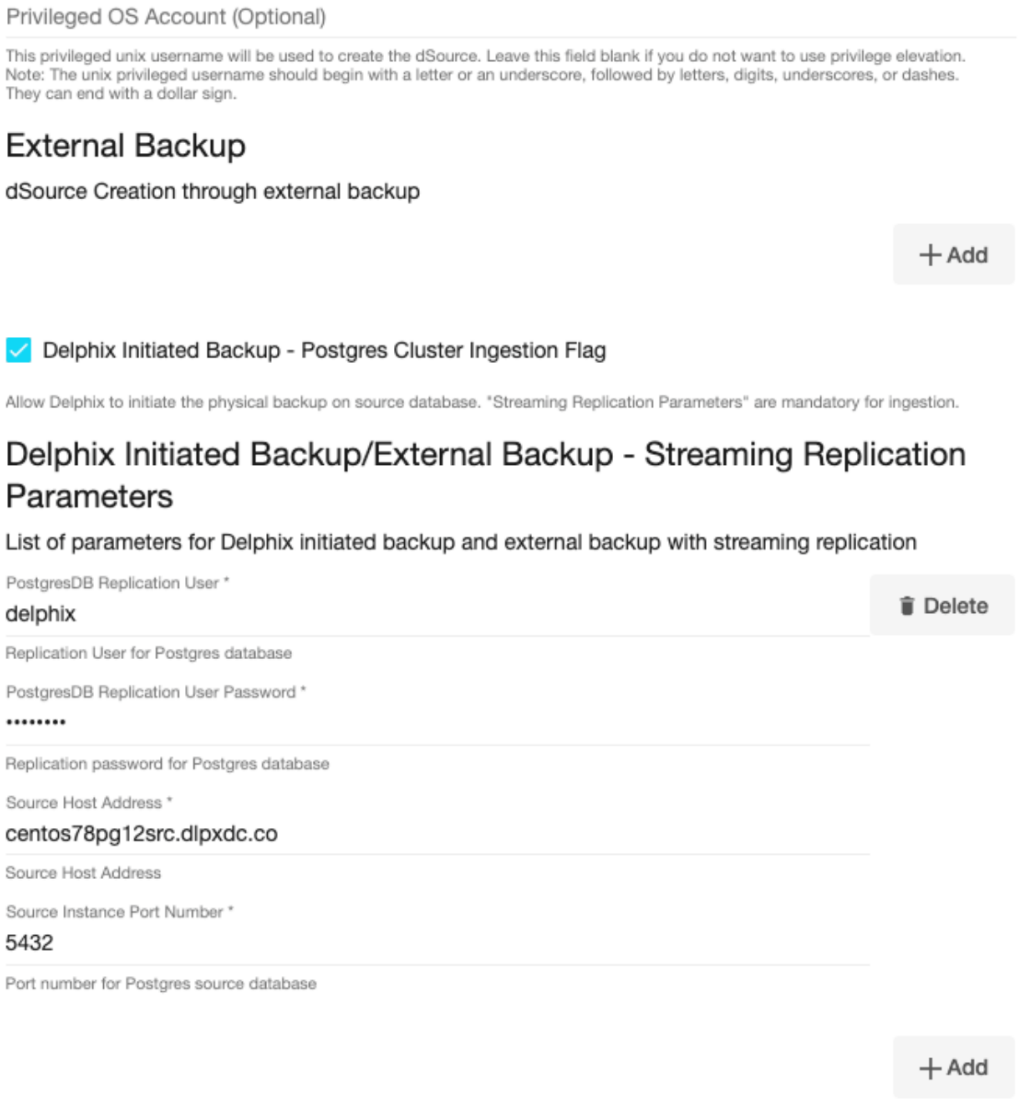
Procedure

1. Login to the **Delphix Management** application.
2. Select **Manage > Environments**.
3. From the **Databases** tab, select a repository from which you want to create your dSource and click the  icon.
4. In the **Add Database** dialog window enter the **Name** for your source config and click **Add**.
5. Select your source config and click the **Add dSource** link located to the right.
6. In the **Source** tab, do the following:

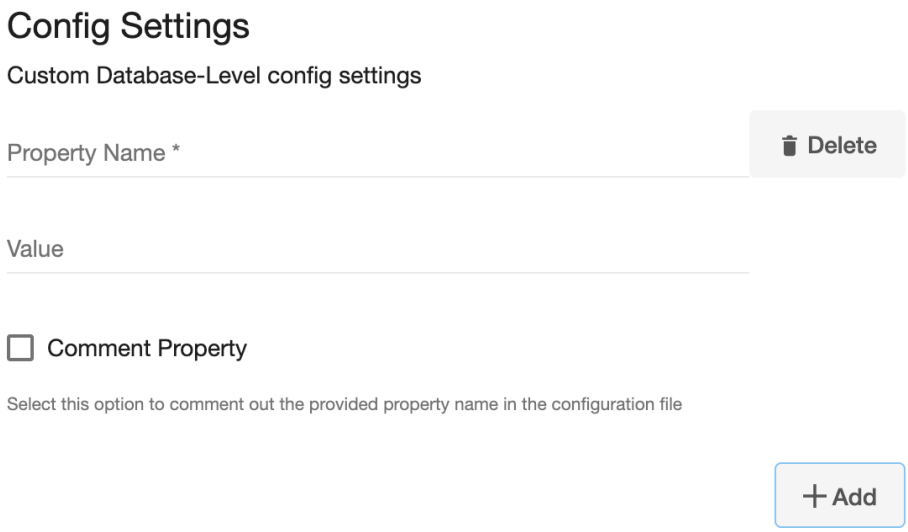


a. In the **NFS Mount Location** field, enter a mount location on the staging host.
b. If you want to use Privilege Elevation, provide the **Privileged OS Account** username. Make sure you meet all the requirements described in the Prerequisites for Privilege Elevation using DLPX_DB_EXEC Script section
   Info:
   You will currently not be able to update the Privileged OS account after dSource creation. The only way to achieve the change is to recreate the dSource.
   If the Environment OS user will be used to create the dSource, make sure you have the following privileges:
   - access to the the postgres commands
   - read+write+execute permissions on the `unix_socket_directories` configured in `postgresql.conf` [Default directories: `/var/run/postgresql` & `/tmp`]
c. Select the checkbox next to the **Delphix Initiated Backup - Postgres Cluster Ingestion Flag** option. Selecting this checkbox allows Delphix to initiate the physical backup on the source database.
d. Click on the **+Add** icon located next to the **Delphix Initiated Backup/External Backup - Streaming Replication Parameters** option.
   Note:
   The Add Button provided for Delphix Initiated Backup/External Backup - Streaming Replication is clickable multiple times but should be used **ONLY** once since the solution is built only to support a single backup path. Providing Multiple Paths will error out the linking process.
   The following fields are mandatory for this process :
   i. Delphix Initiated Backup - Postgres Cluster Ingestion Flag - Checkbox should be selected.

ii. **PostgresDB Replication User**
iii. **PostgresDB Replication User Password**
iv. **Source Host Address**
v. **Source Instance Port Number**
vi. **Staging Instance Port Number** - must be unique for a given instance.



e. Optionally, database configuration parameters can be defined during the linking operation in the **Config Settings** section.



Info:
After dSource creation, users can now configure the parameters and their values in the `postgresql.conf` file through the "Config Settings" section on the Delphix Engine UI. For example, to support **Replication Slots** through UI, users can now provide the "primary_slot_name" parameter and its value through the UI. This will update the already existing parameter value in the `postgresql.conf` file with the value provided by the user.
Similarly, if a user wants to disable a parameter then they can provide the parameter name and select the check box **Comment Property**. This will comment out the parameter in the `postgresql.conf` configuration file.

f. Click **Next.**

7. In the **dSource Configuration** tab, enter a dSource name and select a group for your dSource. Adding a dSource to a dataset group lets you set Delphix Domain user permissions for that database and its objects, such as snapshots. See the topics under Users and Groups for more information.
8. In the **Data Management** tab, specify your Staging Environment, User, and Snapshot Parameters. Select the **Resynchronize dSource** checkbox if you want to resynchronize the dSource. Resynchronizing the dSource will force a non-incremental load of data from the source. This operation is similar to creating a new dSource, but avoids duplicating storage requirements and maintains timeflow history. Click **Next**.
9. In the **Policies** tab, apply policy details to the dSource if required, and then click **Next**.
10. In the **Hooks** tab, select a Hook Point and then click **+** to add a script to run at that point. You can define scripts to run at multiple hook points in the process.
11. In the **Summary** tab, review the configuration profile for the dSource.
12. Click **Submit**. The Delphix Engine initiates two jobs to create the dSource: DB_Link and DB_Sync. You can monitor these jobs by clicking **Active Jobs** from the top menu bar, or by selecting **System > Event Viewer**. When the jobs have successfully completed, the database icon will change to a dSource icon on the **Environments > Host > Databases** screen, and the dSource will also appear in the list of Datasets under its assigned group.

The dSource Configuration Screen

After you have created a dSource, the dSource Configuration tab allows you to view information about it and make modifications to its policies and permissions. In the Datasets panel, select the dSource you wish to examine. You can now choose the configuration tab to see information such as the Source files, Data Management configuration, and Hook Operations.

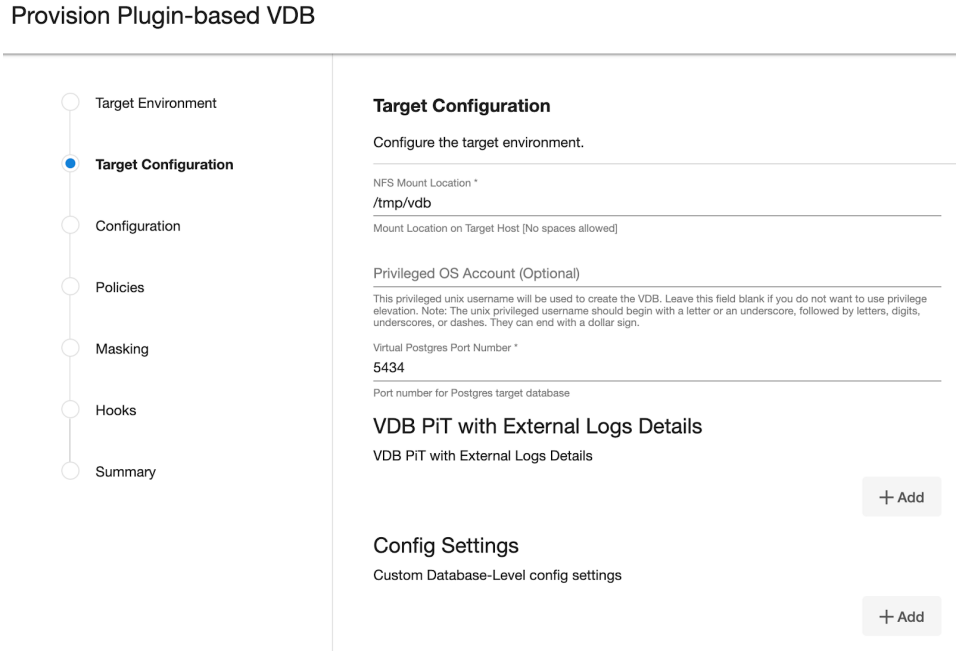
Provisioning a PostgreSQL VDB

Prerequisites

- You will need to have linked a dSource from a staging instance, as described in Linking a PostgreSQL dSource or have created a VDB from which you want to provision another VDB
- You should have set up the POSTGRES target environment with the necessary requirements as described in PostgreSQL Support and Requirements
- Make sure you have the required Instance Owner permissions on the target instance and environment
- The method for Database Permissions for Provisioned PostgreSQL VDBs is decided before the provisioning

Procedure

1. Navigate to **Manage**, and select **Datasets**.
2. Select a dSource and a snapshot from which you want to provision. Click the provision VDB icon to open the **provision VDB** wizard.
3. Select a target environment from the left pane, and an Installation to use from the dropdown list of available PostgreSQL instances on that environment.
4. Set the **Environment User**to be the Instance Owner.
   Note:
   The picking of instance owner is only possible if you have multiple environment users set on that host.
5. You will see the **Target Configuration** section where you need to specify **NFS Mount Location**.

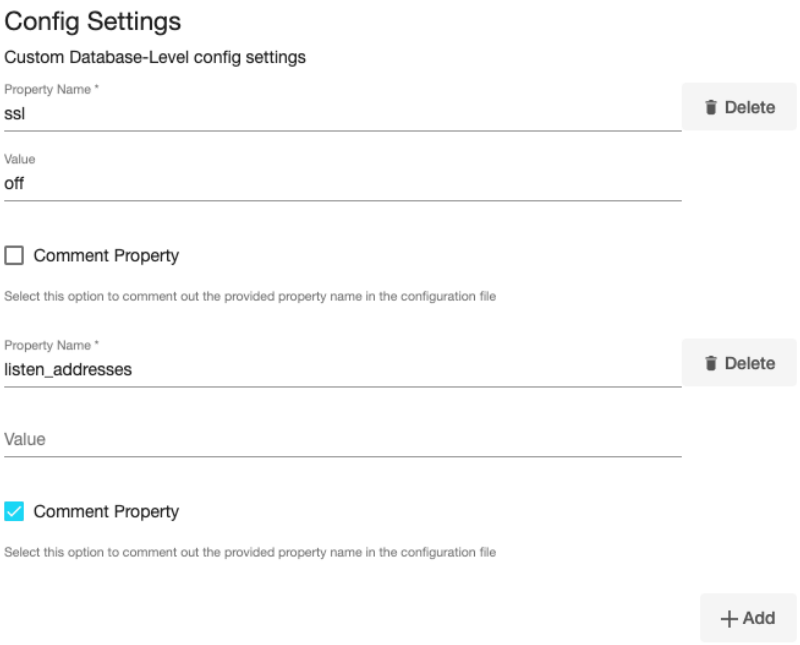


6. If you want to use Privilege Elevation, provide the **Privileged OS Account** username. Make sure you meet all the requirements described in the Prerequisite for Discovery Operation section.

Note:
To update the Privileged OS Account after VDB creation, first disable the instance, update the Privileged OS Account, and then enable it.
If the Environment OS user will be used to create the VDB, make sure you have the following privileges:

- access to the the postgres commands
- read+write+execute permissions on the `unix_socket_directories` configured in `postgresql.conf` [Default directories: `/var/run/postgresql` & `/tmp`]

7. Optionally, you can set the database configuration parameters for the VDB using **Config Settings**. Users can disable a configuration parameter by selecting the check box Comment Property. This will comment out the parameter in the `postgresql.conf` configuration file. Click **Next**.



8. On the **Configuration** page enter the PostgreSQL VDB name which will be displayed on the UI.
9. Select a **Target Group** for the VDB and click the green **Plus** icon to add a new group, if necessary.
10. Select a **Snapshot Policy** for the VDB then click **Next**.
11. Specify any desired hook operations.
12. Review the Provisioning Configuration and Data Management information.
13. Click **Submit**.

Once the VDB provisioning has successfully completed, if the source and target instance ids are not the same, you may want to grant secadm and dbadm on the database to the target instance id. Please refer to Database Permissions for Provisioned PostgreSQL VDBs for more information.

Next steps

Congratulations! You have provisioned your first virtual database!

Now, perform some simple functional tests with your application. You can connect your app to the VDB using standard TNS/JDBC techniques. Delphix has already registered the VDB for you on the target listener.

We suggest the following next steps:

1. Drop a table and use the VDB Rewind feature to test the recovery of your VDB.
2. Take a snapshot of your dSource and refresh your VDB to quickly get fresh production data.
3. Provision a new VDB from your VDB to test the sharing data quickly with other teams.

4. Mask your new VDB to protect sensitive data. Provision new VDBs from that masked VDB to quickly provide safe data to development and QA teams.

PostgreSQL plugin installation

The Plugin is provided as a zip file in the Delphix download site. Follow below steps to download plugin:

1. In the web browser, go to the Delphix download site.
2. Login to the download site using email and password credentials.
3. Navigate to the required version number of Delphix Engine.
4. Go to the Plugins > PostgreSQL > Plugin_<version_number>.zip folder and download the zip file.

Refer to the section Delphix Engine Plugin Management for further details.

PostgreSQL support and requirements

In order to begin using PostgreSQL environments with Delphix, you will need to configure the source and target hosts with the requirements described in this section.

This section covers the following topics:

- Requirements for PostgreSQL hosts and databases
- Network and connectivity requirements for PostgreSQL environments
- Sudo privilege requirements for PostgreSQL environments
- Sudo file configuration examples for PostgreSQL environments
- Prerequisites for privilege elevation using DLPX_DB_EXEC script

To view the PostgreSQL support matrix, see PostgreSQL matrix.

Requirements for PostgreSQL hosts and databases

In order to begin using PostgreSQL environments with Delphix, you will need to configure the source and target with the requirements described on this page.

PostgreSQL hosts and databases

On each host with Postgres, there must be an operating system user configured to the required specifications for Delphix, as explained in the table below. These requirements apply to both source and target environments. However, target environments have additional requirements which are detailed in the Target Host Requirements section below.

> The PostgreSQL plugin requires the use of the default user which gets created during the Postgres DB installation process. Usually, 'postgres'. In case you want to use a low-privileged OS user to add the environment and create datasets with the same, make sure that the additional requirements for Privilege elevation are met, see Prerequisites for Privilege Elevation using DLPX_DB_EXEC Script

Source database requirements

Database Requirement	Explanation
The source database can be in READ-WRITE or READ-only mode. It can be a standby database(streaming site) as well.	
The Staging host must have access to a PostgreSQL role on the source side that has replication, and login privileges. This can be the built-in PostgreSQL role or a newly-created role (for example, delphix).	**Creating a Role for Use with the Delphix Engine** To create a new role for use with the Delphix Engine, use the following command: `SQL> CREATE USER delphix WITH REPLICATION ENCRYPTED [ PASSWORD 'password'];`
You must make the following changes to postgresql.conf: modifying the parameter `listen_addresses`, which specifies the TCP/IP addresses on which the DBMS is to listen for connections from client applications.	**Note**: TCP/IP connectivity must be configured to allow the role mentioned above to connect to the source database from the Delphix Engine and from the standby DBMS instance set up by the Delphix Engine on the staging environment. **listen_addresses configuration** The simplest way to configure PostgresSQL is so that it listens on all available IP interfaces: `listen_addresses = '*'  # Default is 'localhost'` (for more information, see the Server Configuration chapter in the PostgreSQL documentation)

Database Requirement	Explanation
The value of `max_wal_senders`, which specifies the maximum number of concurrent connections from standby servers or streaming base backup clients, must be increased from its desired value by two. That is, in addition to the allowance of connections for consumers other than the Delphix Engine, there must be an allowance for two additional connections from consumers set up by the Delphix Engine.	**max_wal_senders Configuration** The default value max_wal_senders is zero, meaning replication is disabled. In this configuration, the value of max_wal_senders must be increased to two for the Delphix Engine: `max_wal_senders = 2 # Default is 0`
The value of, `wal_level` which determines how much information is written to the write-ahead log (WAL), must be set to archive or hot_standby to allow connections from standby servers. The value "logical" for `wal_level` (introduced in PostgreSQL 9.4) is also supported. In releases prior to PostgreSQL 9.6.x, parameter 'wal_level' allowed the values 'archive' and 'hot_standby'. These values are still accepted but mapped to 'replica'. For example, The plugin will work with ALTER SYSTEM SET wal_level TO 'replica' and ALTER SYSTEM SET wal_level TO 'hot_standby'.	**wal_level Configuration** The default value of wal_level is minimal, which writes only the information needed to recover from a crash or immediate shutdown to the WAL archives. In this configuration, you must add the logging required for WAL archiving as follows: `wal_level = archive # Default is minimal`

You must configure PostgreSQL to allow PostgreSQL client connections and replication client connections from the staging target environment.	To configure appropriately, add the following entries to pg_hba.conf:
	`pg_hba.conf Configuration`
	`host all <role> <ip-address_of_delphix_engine>/32 <auth-method>`
	`host all <role> <ip-address_of_staging_target>/32 <auth-method>`
	`host replication <role> <ip-address_of_staging_target>/32 <auth-method>`

Database Requirement	Explanation
<auth-method> can be set to trust, scram-sha-256, or md5 on source. Delphix inherits the same authentication method from the source while dSource creation and VDB provisioning. For more information on how to configure, pg_hba.conf see the Client Authentication section in the PostgreSQL documentation.	
`wal_keep_segments` parameter in postgresql.conf file on the Source environment should be large enough to support the WAL sync process once the dSource is building up. Postgres 13 onwards, the configuration parameter `wal_keep_segments` is changed to `wal_keep_size`. It is specified in megabytes rather than the number of files as with the `wal_keep_segments` parameter. If you previously used `wal_keep_segments`, the following formula will give you an approximate equivalent setting: `wal_keep_size` = `wal_keep_segments` * `wal_segment_size` (typically 16MB).	
It is mandatory to have "port" (can be commented or not commented) in postgresql.conf file.	

Target Host Requirements

Host Requirement	Explanation
The operating system and architecture of the target environment must match those of the source environment.	It is recommended that the source and the target environments should be identical and hardware configurations should match.
Compatible Operating System supported by the Plugin is CentOS 7.3/7.4/7.5/7.6/7.7/7.8/7.9 and RHEL 7.3/7.4/7.5/7.6/7.7/7.8/7.9/8.3/8.6 .	The underlying Operating System for both the Source and Staging environment should be among these versions.
There must be an installation of PostgreSQL on the target environment that is compatible with the installation of PostgreSQL on the source environment.	Two installations of PostgreSQL are compatible if and only if: 1. They share the same vendor (for example, PostgreSQL is incompatible with EnterpriseDB Postgres Plus Advanced Server). 2. They share the same major version number (for example, 9.5.4 is compatible with 9.5, 9.5.3; however, it is incompatible with 9.3, 9.3.24, or 9.2). 3. They are compiled against the same architecture (in other words, 32-bit and 64-bit installations of Postgres are incompatible). 4. They are compiled with the same WAL segment size. The default WAL segment size of 16 MB is rarely changed in practice, so almost all installations of PostgreSQL are compatible with each other in terms of WAL segment size.
There must be an operating system user (e.g Postgres) with the following privileges: 1. The Delphix Engine must be able to make an SSH connection to the target environment using the operating system user. 2. The operating system user must have read and execute privileges on the PostgreSQL binaries installed on the target environment. 3. The operating system user must have permission to run mount and unmount as the superuser via sudo with neither a password nor a TTY.	See Sudo Privilege Requirements for PostgreSQL Environments for further explanation of the commands, and Sudo File Configuration Examples for PostgreSQL Environments for examples of the /etc/sudoers file on different operating systems.

Host Requirement	Explanation
There must be a directory on the target environment where the Delphix Engine Plugin can be installed (for example, /var/tmp) with the following properties: 1. The Plugin directory must be writable by the operating system user mentioned above. 2. The Plugin directory must have at least 1.5 GB of available storage.	
There must be a mount point directory (for example, /mnt) that will be used as the base for mount points that are created when provisioning a VDB with the following properties: 1. The mount point directory must be writable by the operating system user mentioned above. 2. The mount point directory should be empty.	
TCP/IP connectivity to and from the source environment must be configured as described in General Network and Connectivity Requirements.	

Hostname and IP must be correctly set in /etc/hosts file, for example [postgres@source ~]$ hostname -i	The output of “hostname -i” command should produce the correct result as the IP address of the server.
	For example:
	`[postgres@source postgres]$ hostname -i`
	`10.110.207.113`

When using the External Backup method for ingestion the directories being used for keeping Backup file and WAL file should be accessible by OS User.	

Host Requirement	Explanation

Ensure that either the `netstat` or `ss` utility is installed on the Staging/Target Host.	Checking if **netstat** utility is installed:
	`which netstat`
	Checking if **ss** utility is installed:
	`which ss`

Zip must be installed on the Staging/Target Host for External Backup Support.	Checking if **zip** utility is installed:
	`which zip # or`
	`which unzip`

Host Requirement	Explanation

From PostgreSQL Plugin 3.2.0 onwards, the backups can be provided in three ways:
1. Database full Backup in tar format can be placed inside organized folders named **PostgreSQL_T<*YYYYMMDDhhmmss*>**, created within the backup path.
For example, **PostgreSQL_T20230224033939.**
2. Database full Backup in tar format can be directly placed within the backup path. 3. Providing the full backup encapsulated in a ZIP file in the format **PostgreSQL_T<*YYYYMMDD*>.zip** (supported by previous plugin versions). Make sure that the backup files are available directly at the root level of the ZIP.
For example, **PostgreSQL_T20230224.zip.**
 	
Precedence Rules:
• When all types of backups are provided within the same backup path, the plugin will choose the organized folders over other backups. The backup stored in organized folders has the highest precedence.
• If backups are placed directly within the backup path and the ZIP files are also present, the plugin will choose the direct backups over ZIP files.
• The ZIP files have the least precedence in the hierarchy of all backup types.
	Useful commands:
	`# For method 1`
	`DLPXPGBACKUPDIR=<backup_path>/PostgreSQL_T$(date "+ %Y%m%d%H%M%S")`
	`mkdir -p $DLPXPGBACKUPDIR`
	
	`# For method 2 & 3`
	`DLPXPGBACKUPDIR=<path_to_empty_directory>`
	`mkdir -p $DLPXPGBACKUPDIR`
	
	`# For PostgreSQL versions 9.4.x,9.5.x,9.6.x`
	`<path_to_bin>/pg_basebackup -D $DLPXPGBACKUPDIR -F t -v -w -P -x [-h <source_ip_domain_name>] [-p <source_port>] [-U <source_user>]`
	
	`# For PostgreSQL versions 10.x and above`
	`<path_to_bin>/pg_basebackup -D $DLPXPGBACKUPDIR -F t -v -w -P -Xs [-h <source_ip_domain_name>] [-p <source_port>] [-U <source_user>]`
	
	`# For method 3, zipping the backup`
	`cd $DLPXPGBACKUPDIR`
	`DLPXPGBACKUPZIP=PostgreSQL_T$(date +"%Y%m%d").zip`
	`zip $DLPXPGBACKUPZIP *`
	Checking if backups are available at the root of the ZIP file:
	`# As an example`
	`# The backup files are available at the root of the ZIP file`
	`$ zipinfo PostgreSQL_T20230224.zip`
	`Archive:  PostgreSQL_T20230224.zip`
	`Zip file size: 6883130 bytes, number of entries: 5`
	`-rw-------  3.0 unx  9093632 bx defN 23-Feb-24 02:12 24578.tar`
	`-rw-------  3.0 unx  9093632 bx defN 23-Feb-24 02:12 24579.tar`
	`-rw-------  3.0 unx  9093632 bx defN 23-Feb-24 02:12 24580.tar`
	`-rw-------  3.0 unx 26959360 bx defN 23-Feb-24 02:12 base.tar`
	`-rw-------  3.0 unx 16778752 bx defN 23-Feb-24 02:12 pg_wal.tar`
	`5 files, 71019008 bytes uncompressed, 6882378 bytes compressed:  90.3%`

Host Requirement	Explanation
Note: YYYY denotes year, MM denotes month, DD denotes date, hh denotes hours, mm denotes minutes and ss denotes seconds.	

If there's no PostgreSQL instance running on the Staging/Target host, however, we have a PostgreSQL installation, even then the plugin relies on the "DELPHIX_PG_PATH" variable to discover the environment.
 	
In the absence of "DELPHIX_PG_PATH" variable or if the value of "DELPHIX_PG_PATH" variable is NULL then Linux "find" command will be used for the Environment discovery which may impact the overall performance. Hence, it is preferred to create this Environment Variable with correct entries.
 	
In order to optimize the performance, it is preferred to create an Environment Variable "DELPHIX_PG_PATH" which should be accessible by OS users.
	Below should be the syntax for "DELPHIX_PG_PATH" variable:
	
	`DELPHIX_PG_PATH= "binary_path:data_path1;binary_path:data_path2;"`
	
	For example:
	
	`DELPHIX_PG_PATH="/usr/pgsql-9.6/bin:/var/lib/pgsql/9.6 /data;/opt/edb/as9.6/bin:/ opt/edb/as9.6/data;/usr/pgsql-9.6/bin:/tmp/TESING/ data;"`

The variable "**DELPHIX_PG_PATH**" must be available to the environment user in a non-interactive way. You can test this variable for non-interactive logins using `ssh <your_username>@<target_host> "env | grep DELPHIX_PG_PATH".`

Network and connectivity requirements for PostgreSQL environments

Overview

This topic covers the general network and connectivity requirements for the Delphix Engine, including connection requirements, port allocation, and firewall and Intrusion Detection System (IDS) considerations. For platform-specific network and connectivity requirements, see the relevant topics under the Requirements section for each platform.

General outbound from the Delphix engine port allocation

Protocol	Port Numbers	Use
TCP	25	Connection to a local SMTP server for sending email
TCP/UDP	53	Connections to local DNS servers
UDP	123	Connection to an NTP server
UDP	162	Sending SNMP TRAP messages to an SNMP Manager
TCP	443	HTTPS connections from the Delphix Engine to the Delphix Support upload server
TCP/UDP	636	Secure connections to an LDAP server
TCP	8415	Connections to a Delphix replication target. See Configuring Replication
TCP	50001	Connections to source and target environments for network performance tests.

General inbound to the Delphix Engine port allocation

Protocol	Port Numbers	Use
TCP	22	SSH connections to the Delphix Engine
TCP	80	HTTP connections to the Delphix GUI
UDP	161	Messages from an SNMP Manager to the Delphix Engine
TCP	443	HTTPS connections to the Delphix Management Application

Protocol	Port Numbers	Use
TCP	8415	Delphix Session Protocol connections from all DSP-based network services including Replication, SnapSync for Oracle, V2P, and the Delphix Connector.
TCP	50001	Connections from source and target environments for network performance tests via the Delphix CLI. See Network Performance Tool

Firewalls and Intrusion Detection Systems (IDS)

Production databases on source environments (for dSources) are often separated from the non-production environment by firewalls. Firewalls can add milliseconds to the latency between servers. Accordingly, for best performance, there should be no firewalls between the Delphix Engine and the virtual database (VDB) target environments. If the Delphix Engine is separated from a source environment by a firewall, the firewall must be configured to permit network connections between the Delphix Engine and the source environments for the application protocols (ports) listed above.

Intrusion detection systems (IDSs) should also be made permissive to the Delphix Engine deployment. IDSs should be made aware of the anticipated high volumes of data transfer between dSources and the Delphix Engine.

SSHD Configuration

Target Unix environments are required to have `sshd` running and configured such that the Delphix Engine can connect over `ssh`.

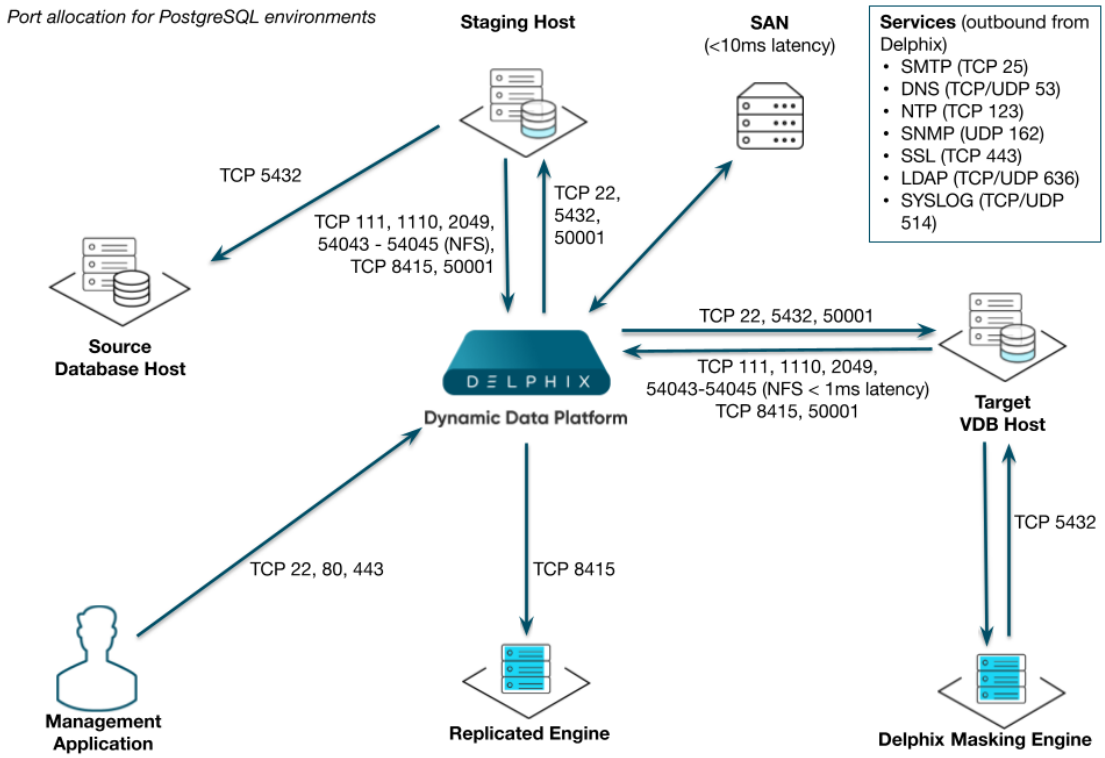
The Delphix platform expects to maintain long-running, highly performant `ssh` connections with remote Unix environments.

Network and connectivity requirements for PostgreSQL

- IP connections must exist between the Delphix Engine and the target environments.
- The Delphix Engine uses an **SSH** connection to each target environment, **NFS** connections from each target environment to the Delphix Engine, and **PostgreSQL client** connections to the virtual databases on the target environment.
- Once connected to a staging target environment through **SSH**, the Delphix Engine initiates a **PostgreSQL replication client** connection from the target environment to the source environment.

Port allocation for PostgreSQL environments

The following diagram describes the port allocations for PostgreSQL environments. It illustrates the ports that we recommend to be open from Delphix to remote services, to the Delphix Engine, and to the Target Environments.



Note: PostgreSQL listener typically runs on TCP 5432. In cases where other ports are used, substitute for 5432 above.

Outbound from the Delphix Engine port allocation

Protocol	Port Numbers	Use
TCP	22	SSH connections to the target database environment
TCP	xxx	PostgreSQL client connections to the PostgreSQL instances on the target environments (port 5432 by default)

Inbound to the Delphix Engine port allocation

Protocol	Port Number	Use
TCP/UDP	111	Remote Procedure Call (RPC) port mapper used for NFSv3 mounts
TCP	1110	Network Status Monitor (NSM) client from target hosts to the Delphix Engine
TCP	2049	NFS client from target hosts to the Delphix Engine (NFSv3 and NFSv4)

Protocol	Port Number	Use
TCP	54043	Client mount daemon (NFSv3 only)
TCP	54044	Lock state notification service (NFSv3 only)
TCP	54045	Network Lock Manager (NLM) client from target hosts to Delphix Engine (NFSv3 only)
UDP	33434 - 33464	Traceroute from the target database server to the Delphix Engine (optional)

Port allocation between source and staging target environments

Outgoing	Incoming	Protocol	Port Number	Use
Target Environment	Source Environment	PostgreSQL replication client	xxx	PostgreSQL replication client connection to the PostgreSQL instances on the source environment (port 5432 by default)

Sudo privilege requirements for PostgreSQL environments

This topic describes the rationale behind specific sudo privilege requirements for virtualizing PostgreSQL Databases.

The sudo configuration exists as /etc/sudoers file.

Below is the example of sudo configuration file contents as mentioned above for Postgres environment.

```
$ vi /etc/sudoers
```

```
Defaults:postgres !requiretty
```

```
postgres ALL=NOPASSWD:  /bin/mount,/bin/umount,/bin/mkdir,/bin/rmdir
```

Privilege	Sources	Targets	Rationale
mkdir/rmdir	Not Required	Required	Delphix dynamically creates and removes directories under the provisioning directory during VDB operations.
`mount/umount`	Not Required	**Required**	Delphix dynamically mounts and unmounts directories under the provisioning directory during VDB operations. This privilege is required because mount and unmount are typically reserved for superuser.

It is required to specify the NOPASSWD qualifier within the "sudo" configuration file, as shown here: Sudo File Configuration Examples for PostgreSQL Environments. This ensures that the "sudo" command does not demand the entry of a password, even for the "display permissions" (i.e. "sudo -l") command.

AppData mount options

Depending on the NFS version used options vers=3 or vers=4.x is added (x varies depending on what that platform supports. e.g. vers=4 or vers=4.1)	
Linux (NFSv3)	`-t nfs -o rw,fg,hard,rsize=1048576,wsize=1048576,nointr,timeo=600,tcp,noacl,vers=3` Copy

Linux (NFSv4)	`-t nfs4 -o rw,fg,hard,rsize=1048576,wsize=1048576,nointr,timeo=600,sec=sys,tcp,noacl` Copy
Note : (For some flavors of Linux and NFSv4.1, additional optional 'v4.1' is added) 1. "port=2049" option is added for all platforms.	
unmount options	"-f" is used for all platforms. For Linux, "-lf" is used.

Mount and unmount options subject to change

Please note that the mount and unmount options listed above are subject to change. For example, if Delphix finds that a certain option improves performance, Delphix may add, remove or change options at anytime. Therefore, it is highly recommended to create the sudo profiles using wildcards that allow any number of options.

Sudo file configuration examples for PostgreSQL environments

This topic describes the rationale behind specific sudo privilege requirements for virtualizing PostgreSQL Databases.

The sudo configuration exists as /etc/sudoers file.

> **Requiretty settings**
>
> Delphix requires that the `requiretty` setting be disabled for all Delphix users with `sudo` privileges.

Configuring `sudo` access on Linux for PostgreSQL target environments

Below is the example of sudo configuration file contents as mentioned above for the Postgres environment.

```
$ vi /etc/sudoers
Defaults:postgres !requiretty
postgres ALL=NOPASSWD:  /bin/mount,/bin/umount,/bin/mkdir,/bin/rmdir
```

> Note that the following examples are for illustrative purposes and the sudo file configuration options are subject to change.

Example 1

This example restricts the PostgreSQL user's use of `sudo` privileges to the directory `/postgres`.

Note that wildcards are allowed for the options on `mount` and `umount` because those commands expect a fixed number of arguments after the options. The option wildcard on the `mount` command also makes it possible to specify the file-system being mounted from the Delphix Engine.

However, wildcards are not acceptable on `mkdir` and `rmdir` because they can have any number of arguments after the options. For those commands, you must specify the exact options (`-p` , `-p -m 755`) used by the Delphix Engine.

Example: /etc/sudoers File Configuration on the Target Environment for sudo Privileges on the VDB Mount Directory Only (Linux OS)

```
Defaults:postgres !requiretty
delphix_os ALL=(root) NOPASSWD: \
/bin/mount  *        /postgres/*, \
/bin/umount *        /postgres/*, \
/bin/umount          /postgres/*, \
/bin/mkdir -p        /postgres/*, \
/bin/mkdir -p -m 755 /postgres/*, \
/bin/mkdir           /postgres/*, \
/bin/rmdir           /postgres/*
```

Example 2

This example restricts the PostgreSQL user's use of `sudo` privileges to the directory / `postgres`, restricts the mount commands to a specific Delphix Engine hostname and IP, and does not allow user-specified options for the `umount` command.

This configuration is more secure, but there is a tradeoff with deployment simplicity. This approach would require a different sudo configuration for targets configured for different Delphix Engines.

Example: Configuring the /etc/sudoers File on the Target Environment for Privileges on the VDB Mount Directory Only (Linux OS)

```
Defaults:delphix_os !requiretty
delphix_os ALL=(root) NOPASSWD: \
/bin/mount           <delphix-server-name>* /postgres/*, \
/bin/mount *         <delphix-server-name>* /postgres/*, \
/bin/mount           <delphix-server-ip>*   /postgres/*, \
/bin/mount *         <delphix-server-ip>*   /postgres/*, \
/bin/mount "", \
/bin/umount          /postgres/*, \
/bin/umount *        /postgres/*, \
/bin/mkdir [*]       /postgres/*, \
/bin/mkdir           /postgres/*, \
/bin/mkdir -p        /postgres/*, \
/bin/mkdir -p -m 755 /postgres/*, \
/bin/rmdir           /postgres/*
```

Prerequisites for privilege elevation using DLPX_DB_EXEC script

Updating DLPX_DB_EXEC script using Delphix Engine web APIs

In order to elevate privileges from a non-privileged OS account (like delphix_os) to a privileged OS account (like postgres), you must push a privilege elevation script (dlpx_db_exec) up into the Delphix Engine to become part of the Delphix common plugin.

DLPX_DB_EXEC script

The privilege elevation profile script dlpx_db_exec allows you to execute commands that require superuser privileges on the source and target machines. The privilege elevation script dlpx_db_exec can be created or pushed to Delphix Engine using Web API calls, CURL or dxtoolkit. For steps on creating a Privilege Elevation Profile, see CLI Cookbook: How to create or edit a privilege elevation profiles and profile scripts

Content of DLPX_DB_EXEC privilege elevation profile

```
#!/bin/sh
#
# Copyright (c) 2018 by Delphix. All rights reserved.
#

#
# This script allows customization of command execution with an alternate user
# account.
if [[ $1 != -u* ]]; then
    echo "Incorrect command line parameters, -u<optional user account> is required as
the first parameter"
    exit 1
fi
user_id=`echo $1 | sed -e "s/^-u//"`

shift 1
if [[ $user_id != "delphix_os" ]]; then
    command=$(printf "%s" "$@")
    cd /tmp
    sudo -E su $user_id -p -c "$command"
else
    $@
fi
```

Below is an example of how you can push privilege elevation script “dlpx_db_exec” onto Delphix Engine.

1. Create a session to Delphix Engine as Delphix OS User.

```
curl -i -c cookies.txt -X POST -H "Content-Type:application/json" http://
<Delphix-Engine>/resources/json/delphix/session -d '{
    "version":{
        "minor":11,
        "major":1,
        "micro": 5,
```

```
        "type":"APIVersion"
    },
    "type":"APISession"
}'
```

The API Version needs to be identified as per the Delphix Engine installed at the customer site.

2. Login to the Delphix Engine as Delphix OS User.

```
curl -i -c cookies.txt -b cookies.txt -X POST -H "Content-Type:application/
json" http://<DELPHIX_ENGINE>/resources/json/delphix/login -d '{
    "password":"<PASSWORD>",
    "type":"LoginRequest",
    "target":"DOMAIN",
    "username":"<USERNAME>"
}'
```

3. Push DLPX_DB_EXEC contents to Delphix Engine.

```
curl -i -b cookies.txt -X POST -H "Content-Type:application/json" http://
<DELPHIX_ENGINE>/resources/json/delphix/host/privilegeElevation/profileScript/
HOST_PRIVILEGE_ELEVATION_PROFILE_SCRIPT-7 -d '{
    "type": "HostPrivilegeElevationProfileScript",
    "contents": "#\n# Copyright (c) 2018 by Delphix. All rights reserved.
\n#\n#\n# This script allows customization of command execution with an
alternate user\n# account.\nif [[ $1 != -u* ]]; then\n    echo \"Incorrect
command line parameters, -u<optional user account> is required as the first
parameter\"\n    exit 1\nfi\nuser_id=`echo $1 | sed -e \"s\/^-u\/\/\"`\nshift
1\nif [[ $user_id != \"delphix_os\" ]]; then\n    command=$(printf \"%s\"
\"$@\")\n    cd /tmp\n    sudo -E su $user_id -p -c \"$command\"\nelse\n
$@\nfi\n"
}'
```

Make sure that you edit the <USERNAME>, <PASSWORD> & <DELPHIX_ENGINE> parameter values.

If the `dlpx_db_exec` script is updated after the Environment(s) is added/created, then you must perform the **Refresh** operation to propagate the changes.

Staging/Target host requirements

To accomplish necessary tasks and run all Plugin operations hosts, the Delphix OS user account (henceforth referred to as "delphix_os") requires privilege elevation specifications. Here is an example specification for the "sudo" privilege elevation utility, using the "visudo" to edit the "sudoers" configuration file. This specification makes the following assumptions:

- OS = Linux
- The Privileged OS account is named postgres.

The following sudoers entry is only for template purposes. Modify the path in the below sudoers entry with the appropriate binary paths of your environment.

Entry required for both Linking and Provisioning via low-privileged user (delphix_os):

For PLUGIN_VERSION >=4.1.0	Defaults:delphix_os !requiretty delphix_os ALL=NOPASSWD:SETENV: /bin/mount,\ /bin/umount,\ /bin/mkdir,\ /bin/rmdir,\ /bin/ps,\ /bin/su postgres -p -c '*'/createdb -p * '*',\ /bin/su postgres -p -c '*'/dropdb -p * '*',\ /bin/su postgres -p -c '*'/pg_dump -Fd '*' -p * -j * -h '*' -f '*/*_backup' -U * --verbose,\ /bin/su postgres -p -c '*'/pg_dumpall --globals-only --no-role-password --clean -p * -h '*' -f '*/*dumpall_file.sql' -U *,\ /bin/su postgres -p -c '*'/pg_restore -p * -d '*' -j * '*/*_backup' --verbose,\ /bin/su postgres -p -c */pg_basebackup -D */data -F t -v -w * -p * -U * --checkpoint\=fast,\ /bin/su postgres -p -c */pg_controldata */data,\ /bin/su postgres -p -c */pg_ctl --version,\ /bin/su postgres -p -c */pg_ctl initdb -D */data,\ /bin/su postgres -p -c */pg_ctl initdb -D */data -o '*',\ /bin/su postgres -p -c */pg_ctl initdb -D */postgres_init,\ /bin/su postgres -p -c */pg_ctl initdb -D */postgres_init -o '*',\ /bin/su postgres -p -c */pg_ctl start -D */data -t * &> */.delphix/postgres_db_log,\ /bin/su postgres -p -c */pg_ctl status -D */data,\ /bin/su postgres -p -c */pg_ctl stop -D */data &> */.delphix/postgres_db_log,\ /bin/su postgres -p -c */pg_isready -h * -p *,\ /bin/su postgres -p -c */psql -t -U * -p * -d * -h * -c 'select 1;',\ /bin/su postgres -p -c */psql -p * -d postgres -f '*/dumpall_file.sql',\ /bin/su postgres -p -c */psql -t -U * -p * -d * -h * -c 'select name\, setting from pg_settings;' -o */.delphix/source_postgresql_config_file.conf,\ /bin/su postgres -p -c */psql -t -U * -p * -d * -h * -c 'show server_version;',\ /bin/su postgres -p -c base64 -d $DLPX_PG_PLUGIN_ECHO_PATH/'database_oid' > */.delphix/database_oid,\ /bin/su postgres -p -c base64 -d $DLPX_PG_PLUGIN_ECHO_PATH/'ingestion_method' > */.delphix/ingestion_method,\ /bin/su postgres -p -c base64 -d $DLPX_PG_PLUGIN_ECHO_PATH/'postgresql_tmp.conf' > */data/postgresql_tmp.conf,\ /bin/su postgres -p -c base64 -d $DLPX_PG_PLUGIN_ECHO_PATH/'recovery.conf' >> */data/recovery.conf,\

```
/bin/su postgres -p -c base64 -d $DLPX_PG_PLUGIN_ECHO_PATH/'tablespace_file' > */.delphix/tablespace_file,\
/bin/su postgres -p -c base64 -d $DLPX_PG_PLUGIN_ECHO_PATH/'walControl.sh' > */data/scripts/walControl.sh,\
/bin/su postgres -p -c cat '*/.delphix/postgres_db_log',\
/bin/su postgres -p -c cat '*/data/PG_VERSION',\
/bin/su postgres -p -c cat '*/data/current_logfiles',\
/bin/su postgres -p -c cat '*/data/postgresql.conf',\
/bin/su postgres -p -c cat '*/dumpall_file.sql',\
/bin/su postgres -p -c cat '*/scratch_file.sql',\
/bin/su postgres -p -c cat '*/source_postgresql_config_file.conf',\
/bin/su postgres -p -c chmod 0700 '*/data',\
/bin/su postgres -p -c chmod 0700 '*/data/recovery.conf.delphix',\
/bin/su postgres -p -c chmod 0755 '*/data/scripts/walControl.sh',\
/bin/su postgres -p -c chmod 0666 '*/scratch_file.sql',\
/bin/su postgres -p -c cp '*' '*/data/postgresql.conf',\
/bin/su postgres -p -c cp '*/data/postgresql.conf' '*/data/postgresql.conf_backup',\
/bin/su postgres -p -c cp '*/dumpall_file.sql' '*/scratch_file.sql',\
/bin/su postgres -p -c echo '' > '*/data/postgresql.auto.conf',\
/bin/su postgres -p -c echo '' > '*/data/recovery.conf',\
/bin/su postgres -p -c echo '' > '*/data/standby.signal',\
/bin/su postgres -p -c echo '' > '*/data/recovery.signal',\
/bin/su postgres -p -c echo '' > '*/.delphix/staging_push',\
/bin/su postgres -p -c find * -type f -name PostgreSQL_T\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\].zip,\
/bin/su postgres -p -c find * -type d -name PostgreSQL_T\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\],\
/bin/su postgres -p -c find * -type f -name base.tar,\
/bin/su postgres -p -c find */data -type f -name \*.tar,\
/bin/su postgres -p -c find * -type f -name \[0-9\]\*tar,\
/bin/su postgres -p -c grep -w '^log_directory' '*/data/postgresql.conf',\
/bin/su postgres -p -c grep -w '^max_connections' '*/data/postgresql.auto.conf',\
/bin/su postgres -p -c grep -w '^max_connections' '*/data/postgresql_source.conf',\
/bin/su postgres -p -c grep -w '^max_wal_senders' '*/data/postgresql.auto.conf',\
/bin/su postgres -p -c grep -w '^max_wal_senders' '*/data/postgresql_source.conf',\
/bin/su postgres -p -c grep -w '^port' '*/data/postgresql.auto.conf',\
```

```
/bin/su postgres -p -c grep -w '^port' '*/data/postgresql.conf',\
/bin/su postgres -p -c grep -w '^wal_keep_segments' '*/data/postgresql.auto.conf',\
/bin/su postgres -p -c grep -w '^wal_keep_segments' '*/data/postgresql_source.conf',\
/bin/su postgres -p -c grep -w '^wal_keep_size' '*/data/postgresql.auto.conf',\
/bin/su postgres -p -c grep -w '^wal_keep_size' '*/data/postgresql_source.conf',\
/bin/su postgres -p -c grep -w '^timezone' '*/data/postgresql.conf',\
/bin/su postgres -p -c ln -s '../tablespace/*' '*/data/pg_tblspc',\
/bin/su postgres -p -c ls -lrt '*' | tail -n+2,\
/bin/su postgres -p -c mkdir -p '*',\
/bin/su postgres -p -c du -sb '*',\
/bin/su postgres -p -c mv '*/data/postgresql.auto.conf' '*/data/
postgresql.auto.conf.delphix',\
/bin/su postgres -p -c mv '*/data/postgresql.auto.conf' '*/data/postgresql.auto.source.conf',
\
/bin/su postgres -p -c mv '*/data/postgresql.conf' '*/data/postgresql_source.conf',\
/bin/su postgres -p -c mv '*/data/postgresql_tmp.conf' '*/data/postgresql.conf',\
/bin/su postgres -p -c mv '*/data/postmaster.opts' '*/data/postmaster.opts.delphix',\
/bin/su postgres -p -c mv '*/data/recovery.conf' '*/data/recovery.conf.delphix',\
/bin/su postgres -p -c mv '*/data/standby.signal' '*/data/standby.signal.delphix',\
/bin/su postgres -p -c mv '*/postgres_init/postgresql.auto.conf' '*/data',\
/bin/su postgres -p -c mv '*/postgres_init/postgresql.conf' '*/data',\
/bin/su postgres -p -c mv '*/postgres_init/pg_hba.conf' '*/data',\
/bin/su postgres -p -c mv '*/postgres_init/pg_ident.conf' '*/data',\
/bin/su postgres -p -c mv '*/scratch_file.sql' '*/dumpall_file.sql',\
/bin/su postgres -p -c printenv param_value >> */data/recovery.conf,\
/bin/su postgres -p -c rm -f '*/data/*.tar',\
/bin/su postgres -p -c rm -f '*/data/postmaster.pid',\
/bin/su postgres -p -c rm -f '*/data/tablespace_map',\
/bin/su postgres -p -c rm -r -f '*',\
/bin/su postgres -p -c tail -n 1 '*/data/scripts/WalBreakChainDetected',\
/bin/su postgres -p -c tail -n 1 '*/data/scripts/InvalidRecordFileFoundData',\
/bin/su postgres -p -c tail -n * '*/data/log/*',\
/bin/su postgres -p -c tail -n * '*/data/pg_log/*',\
/bin/su postgres -p -c tar -xf */base.tar -C */data,\
/bin/su postgres -p -c tar -xf */pg_wal.tar -C */data/pg_wal,\
/bin/su postgres -p -c tar -xf */*.tar -C */data/tablespace*,\
```

	/bin/su postgres -p -c test -d '*',\ /bin/su postgres -p -c test -f '*',\ /bin/su postgres -p -c test -r '*' && test -w '*' && test -x '*',\ /bin/su postgres -p -c test -r '*' && test -x '*',\ /bin/su postgres -p -c test -r '*',\ /bin/su postgres -p -c test -w '*' && test -x '*',\ /bin/su postgres -p -c unzip '*/PostgreSQL_T[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9].zip' -d '*/data',\ /bin/su postgres -p -c zipinfo* '*/PostgreSQL_T[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9].zip'

For PLUGIN_VERSION >=3.1.0	Defaults:delphix_os !requiretty delphix_os ALL=NOPASSWD:SETENV: /bin/mount,\ /bin/umount,\ /bin/mkdir,\ /bin/rmdir,\ /bin/ps,\ /bin/su postgres -p -c '*'/createdb -p * '*',\ /bin/su postgres -p -c '*'/dropdb -p * '*',\ /bin/su postgres -p -c '*'/pg_dump -Fd '*' -p * -j * -h '*' -f '*/*_backup' -U * --verbose,\ /bin/su postgres -p -c '*'/pg_dumpall --globals-only --no-role-password --clean -p * -h '*' -f '*/*dumpall_file.sql' -U *,\ /bin/su postgres -p -c '*'/pg_restore -p * -d '*' -j * '*/*_backup' --verbose,\ /bin/su postgres -p -c */pg_basebackup -D */data -F t -v -w * -p * -U * --checkpoint\=fast,\ /bin/su postgres -p -c */pg_controldata */data,\ /bin/su postgres -p -c */pg_ctl --version,\ /bin/su postgres -p -c */pg_ctl initdb -D */data,\ /bin/su postgres -p -c */pg_ctl initdb -D */postgres_init,\ /bin/su postgres -p -c */pg_ctl start -D */data -t * &> */.delphix/postgres_db_log,\ /bin/su postgres -p -c */pg_ctl status -D */data,\ /bin/su postgres -p -c */pg_ctl stop -D */data &> */.delphix/postgres_db_log,\ /bin/su postgres -p -c */pg_isready -h * -p *,\ /bin/su postgres -p -c */psql -t -U * -p * -d * -h * -c 'select 1;',\ /bin/su postgres -p -c */psql -p * -d postgres -f '*/dumpall_file.sql',\ /bin/su postgres -p -c */psql -t -U * -p * -d * -h * -c 'select name\, setting from pg_settings;' -o */.delphix/source_postgresql_config_file.conf,\ /bin/su postgres -p -c */psql -t -U * -p * -d * -h * -c 'show server_version;',\ /bin/su postgres -p -c */psql -t -U * -p * -c "select SUM(pg_database_size(pg_database.datname)) from pg_database;",\ /bin/su postgres -p -c */psql -t -U * -p * -d * -c "select pg_database_size(pg_database.datname) from pg_database where pg_database.datname\= '*';",\ /bin/su postgres -p -c base64 -d $DLPX_PG_PLUGIN_ECHO_PATH/'database_oid' > */.delphix/database_oid,\ /bin/su postgres -p -c base64 -d $DLPX_PG_PLUGIN_ECHO_PATH/'ingestion_method' > */.delphix/ingestion_method,\ /bin/su postgres -p -c base64 -d $DLPX_PG_PLUGIN_ECHO_PATH/'postgresql_tmp.conf' > */data/postgresql_tmp.conf,\

```
/bin/su postgres -p -c base64 -d $DLPX_PG_PLUGIN_ECHO_PATH/'recovery.conf' >> */data/recovery.conf,\
/bin/su postgres -p -c base64 -d $DLPX_PG_PLUGIN_ECHO_PATH/'tablespace_file' > */.delphix/tablespace_file,\
/bin/su postgres -p -c base64 -d $DLPX_PG_PLUGIN_ECHO_PATH/'walControl.sh' > */data/scripts/walControl.sh,\
/bin/su postgres -p -c cat '*/.delphix/postgres_db_log',\
/bin/su postgres -p -c cat '*/data/PG_VERSION',\
/bin/su postgres -p -c cat '*/data/current_logfiles',\
/bin/su postgres -p -c cat '*/data/postgresql.conf',\
/bin/su postgres -p -c cat '*/dumpall_file.sql',\
/bin/su postgres -p -c cat '*/scratch_file.sql',\
/bin/su postgres -p -c chmod 0700 '*/data',\
/bin/su postgres -p -c chmod 0700 '*/data/recovery.conf.delphix',\
/bin/su postgres -p -c chmod 0755 '*/data/scripts/walControl.sh',\
/bin/su postgres -p -c chmod 0666 '*/scratch_file.sql',\
/bin/su postgres -p -c cp '*' '*/data/postgresql.conf',\
/bin/su postgres -p -c cp '*/data/postgresql.conf' '*/data/postgresql.conf_backup',\
/bin/su postgres -p -c cp '*/dumpall_file.sql' '*/scratch_file.sql',\
/bin/su postgres -p -c echo '' > '*/data/postgresql.auto.conf',\
/bin/su postgres -p -c echo '' > '*/data/recovery.conf',\
/bin/su postgres -p -c echo '' > '*/data/standby.signal',\
/bin/su postgres -p -c echo '' > '*/data/recovery.signal',\
/bin/su postgres -p -c echo '' > '*/.delphix/staging_push',\
/bin/su postgres -p -c find * -type f -name PostgreSQL_T\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\].zip,\
/bin/su postgres -p -c find * -type d -name PostgreSQL_T\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\],\
/bin/su postgres -p -c find * -type f -name base.tar,\
/bin/su postgres -p -c find */data -type f -name \*.tar,\
/bin/su postgres -p -c find * -type f -name \[0-9\]\*tar,\
/bin/su postgres -p -c grep -w '^log_directory' '*/data/postgresql.conf',\
/bin/su postgres -p -c grep -w '^max_connections' '*/data/postgresql.auto.conf',\
/bin/su postgres -p -c grep -w '^max_connections' '*/data/postgresql_source.conf',\
/bin/su postgres -p -c grep -w '^max_wal_senders' '*/data/postgresql.auto.conf',\
/bin/su postgres -p -c grep -w '^max_wal_senders' '*/data/postgresql_source.conf',\
/bin/su postgres -p -c grep -w '^port' '*/data/postgresql.auto.conf',\
```

```
/bin/su postgres -p -c grep -w '^port' '*/data/postgresql.conf',\
/bin/su postgres -p -c grep -w '^wal_keep_segments' '*/data/postgresql.auto.conf',\
/bin/su postgres -p -c grep -w '^wal_keep_segments' '*/data/postgresql_source.conf',\
/bin/su postgres -p -c grep -w '^wal_keep_size' '*/data/postgresql.auto.conf',\
/bin/su postgres -p -c grep -w '^wal_keep_size' '*/data/postgresql_source.conf',\
/bin/su postgres -p -c grep -w '^timezone' '*/data/postgresql.conf',\
/bin/su postgres -p -c ln -s '../tablespace/*' '*/data/pg_tblspc',\
/bin/su postgres -p -c ls -lrt '*' | tail -n+2,\
/bin/su postgres -p -c mkdir -p '*',\
/bin/su postgres -p -c mv '*/data/postgresql.auto.conf' '*/data/
postgresql.auto.conf.delphix',\
/bin/su postgres -p -c mv '*/data/postgresql.auto.conf' '*/data/postgresql.auto.source.conf',
\
/bin/su postgres -p -c mv '*/data/postgresql.conf' '*/data/postgresql_source.conf',\
/bin/su postgres -p -c mv '*/data/postgresql_tmp.conf' '*/data/postgresql.conf',\
/bin/su postgres -p -c mv '*/data/postmaster.opts' '*/data/postmaster.opts.delphix',\
/bin/su postgres -p -c mv '*/data/recovery.conf' '*/data/recovery.conf.delphix',\
/bin/su postgres -p -c mv '*/data/standby.signal' '*/data/standby.signal.delphix',\
/bin/su postgres -p -c mv '*/postgres_init/postgresql.auto.conf' '*/data',\
/bin/su postgres -p -c mv '*/postgres_init/postgresql.conf' '*/data',\
/bin/su postgres -p -c mv '*/postgres_init/pg_hba.conf' '*/data',\
/bin/su postgres -p -c mv '*/postgres_init/pg_ident.conf' '*/data',\
/bin/su postgres -p -c mv '*/scratch_file.sql' '*/dumpall_file.sql',\
/bin/su postgres -p -c printenv param_value >> */data/recovery.conf,\
/bin/su postgres -p -c rm -f '*/data/*.tar',\
/bin/su postgres -p -c rm -f '*/data/postmaster.pid',\
/bin/su postgres -p -c rm -f '*/data/tablespace_map',\
/bin/su postgres -p -c rm -r -f '*',\
/bin/su postgres -p -c tail -n 1 '*/data/scripts/WalBreakChainDetected',\
/bin/su postgres -p -c tail -n 1 '*/data/scripts/InvalidRecordFileFoundData',\
/bin/su postgres -p -c tail -n * '*/data/log/*',\
/bin/su postgres -p -c tail -n * '*/data/pg_log/*',\
/bin/su postgres -p -c tar -xf */base.tar -C */data,\
/bin/su postgres -p -c tar -xf */pg_wal.tar -C */data/pg_wal,\
/bin/su postgres -p -c tar -xf */*.tar -C */data/tablespace*,\
/bin/su postgres -p -c test -d '*',\
```

	/bin/su postgres -p -c test -f '*',\ /bin/su postgres -p -c test -r '*' && test -w '*' && test -x '*',\ /bin/su postgres -p -c test -r '*' && test -x '*',\ /bin/su postgres -p -c test -r '*',\ /bin/su postgres -p -c test -w '*' && test -x '*',\ /bin/su postgres -p -c unzip '*/PostgreSQL_T[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9].zip' -d '*/data',\ /bin/su postgres -p -c zipinfo* '*/PostgreSQL_T[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9].zip'

For PLUGIN_VERSION <=3.0.0	Defaults:delphix_os !requiretty delphix_os ALL=NOPASSWD:SETENV: /bin/mount,\ /bin/umount,\ /bin/mkdir,\ /bin/rmdir,\ /bin/ps,\ /bin/su postgres -p -c '*'/createdb -p * '*',\ /bin/su postgres -p -c '*'/dropdb -p * '*',\ /bin/su postgres -p -c '*'/pg_dump -Fd '*' -p * -j * -h '*' -f '*/*_backup' -U * --verbose,\ /bin/su postgres -p -c '*'/pg_restore -p * -O -d '*' -j * '*/*_backup' --verbose,\ /bin/su postgres -p -c */pg_basebackup -D */data -F t -v -w -P -h * -p * -U * --checkpoint=fast,\ /bin/su postgres -p -c */pg_controldata */data,\ /bin/su postgres -p -c */pg_ctl --version,\ /bin/su postgres -p -c */pg_ctl initdb -D */data,\ /bin/su postgres -p -c */pg_ctl initdb -D */postgres_init,\ /bin/su postgres -p -c */pg_ctl start -D */data -t * &> */.delphix/postgres_db_log,\ /bin/su postgres -p -c */pg_ctl status -D */data,\ /bin/su postgres -p -c */pg_ctl stop -D */data &> */.delphix/postgres_db_log,\ /bin/su postgres -p -c */pg_isready -h * -p *,\ /bin/su postgres -p -c */psql -t -U * -p * -d * -h * -c 'select 1;',\ /bin/su postgres -p -c */psql -t -U * -p * -d * -h * -c 'select name\, setting from pg_settings;' -o */.delphix/source_postgresql_config_file.conf,\ /bin/su postgres -p -c */psql -t -U * -p * -d * -h * -c 'show server_version;',\ /bin/su postgres -p -c cat '*/.delphix/postgres_db_log',\ /bin/su postgres -p -c cat '*/data/PG_VERSION',\ /bin/su postgres -p -c cat '*/data/current_logfiles',\ /bin/su postgres -p -c cat '*/data/postgresql.conf',\ /bin/su postgres -p -c chmod 0700 '*/data',\ /bin/su postgres -p -c chmod 0700 '*/data/recovery.conf.delphix',\ /bin/su postgres -p -c chmod 0755 '*/data/scripts/walControl.sh',\ /bin/su postgres -p -c cp '*' '*/data/postgresql.conf',\ /bin/su postgres -p -c cp '*/data/postgresql.conf' '*/data/postgresql.conf_backup',\ /bin/su postgres -p -c echo '*' > '*/data/postgresql_tmp.conf',\ /bin/su postgres -p -c echo '*' > '*/data/scripts/walControl.sh',\ /bin/su postgres -p -c echo '*' >> */data/recovery.conf,\

```
/bin/su postgres -p -c echo [ ]>> '*/data/postgresql.auto.conf',\
/bin/su postgres -p -c echo [ ]>> '*/data/recovery.conf',\
/bin/su postgres -p -c echo [ ]>> '*/data/standby.signal',\
/bin/su postgres -p -c find * -type f -name PostgreSQL_T\[0-9\]\[0-9\]\[0-9\]\[0-9\]\[0-9\]
\[0-9\]\[0-9\]\[0-9\].zip,\
/bin/su postgres -p -c find */data -type f -name \*.tar,\
/bin/su postgres -p -c find */data -type f -name \[0-9\]\*tar,\
/bin/su postgres -p -c grep -w '^log_directory' '*/data/postgresql.conf',\
/bin/su postgres -p -c grep -w '^max_connections' '*/data/postgresql.auto.conf',\
/bin/su postgres -p -c grep -w '^max_connections' '*/data/postgresql_source.conf',\
/bin/su postgres -p -c grep -w '^max_wal_senders' '*/data/postgresql.auto.conf',\
/bin/su postgres -p -c grep -w '^max_wal_senders' '*/data/postgresql_source.conf',\
/bin/su postgres -p -c grep -w '^port' '*/data/postgresql.auto.conf',\
/bin/su postgres -p -c grep -w '^port' '*/data/postgresql.conf',\
/bin/su postgres -p -c grep -w '^wal_keep_segments' '*/data/postgresql.auto.conf',\
/bin/su postgres -p -c grep -w '^wal_keep_segments' '*/data/postgresql_source.conf',\
/bin/su postgres -p -c grep -w '^wal_keep_size' '*/data/postgresql.auto.conf',\
/bin/su postgres -p -c grep -w '^wal_keep_size' '*/data/postgresql_source.conf',\
/bin/su postgres -p -c ln -s '../tablespace/*' '*/data/pg_tblspc',\
/bin/su postgres -p -c ls -lrt '*/data' | tail -n+2,\
/bin/su postgres -p -c ls -lrt '*/data/pg_log' | tail -n+2,\
/bin/su postgres -p -c ls -lrt '*/data/pg_tblspc' | tail -n+2,\
/bin/su postgres -p -c mkdir -p '*/.delphix',\
/bin/su postgres -p -c mkdir -p '*/data',\
/bin/su postgres -p -c mkdir -p '*/data/scripts',\
/bin/su postgres -p -c mkdir -p '*/data/tablespace*',\
/bin/su postgres -p -c mkdir -p '*/postgres_init',\
/bin/su postgres -p -c mv '*/data/postgresql.auto.conf' '*/data/
postgresql.auto.conf.delphix',\
/bin/su postgres -p -c mv '*/data/postgresql.auto.conf' '*/data/postgresql.auto.source.conf',
\
/bin/su postgres -p -c mv '*/data/postgresql.conf' '*/data/postgresql_source.conf',\
/bin/su postgres -p -c mv '*/data/postgresql_tmp.conf' '*/data/postgresql.conf',\
/bin/su postgres -p -c mv '*/data/postmaster.opts' '*/data/postmaster.opts.delphix',\
/bin/su postgres -p -c mv '*/data/recovery.conf' '*/data/recovery.conf.delphix',\
/bin/su postgres -p -c mv '*/data/standby.signal' '*/data/standby.signal.delphix',\
```

	/bin/su postgres -p -c mv '*/postgres_init/postgresql.auto.conf' '*/data',\ /bin/su postgres -p -c mv '*/postgres_init/postgresql.conf' '*/data',\ /bin/su postgres -p -c printenv param_value >> */data/recovery.conf,\ /bin/su postgres -p -c rm -f '*/data/*.tar',\ /bin/su postgres -p -c rm -f '*/data/tablespace_map',\ /bin/su postgres -p -c rm -f '*/data/postmaster.pid',\ /bin/su postgres -p -c rm -r -f '*/*_backup',\ /bin/su postgres -p -c rm -r -f '*/.delphix',\ /bin/su postgres -p -c rm -r -f '*/data',\ /bin/su postgres -p -c rm -r -f '*/postgres_init',\ /bin/su postgres -p -c tail -n 1 '*/data/scripts/WalBreakChainDetected',\ /bin/su postgres -p -c tail -n 1 '*/data/scripts/InvalidRecordFileFoundData',\ /bin/su postgres -p -c tail -n * '*/data/log/*',\ /bin/su postgres -p -c tail -n * '*/data/pg_log/*',\ /bin/su postgres -p -c tar -xf */data/base.tar -C */data,\ /bin/su postgres -p -c tar -xf */data/pg_wal.tar -C */data/pg_wal,\ /bin/su postgres -p -c tar -xf */*.tar -C */data/tablespace*,\ /bin/su postgres -p -c test -d '*',\ /bin/su postgres -p -c test -f '*',\ /bin/su postgres -p -c test -r '*' && test -w '*' && test -x '*',\ /bin/su postgres -p -c test -r '*' && test -x '*',\ /bin/su postgres -p -c test -r '*',\ /bin/su postgres -p -c test -w '*' && test -x '*',\ /bin/su postgres -p -c unzip '*/PostgreSQL_T[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9].zip' -d '*/data',\ /bin/su postgres -p -c zipinfo '*/PostgreSQL_T[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9].zip'

Prerequisite for discovery operation

> The Discovery operation cannot be run by the high-privileged user, it will be executed by the low-privileged user.

1. The environment variable `DELPHIX_PG_PATH` should be accessible to the low privilege user.
   The variable `DELPHIX_PG_PATH` must be available to the environment (low-privileged) user in a non-interactive way. You can test this variable for non-interactive logins using `ssh <low_privileged_user>@<target_host> "env | grep DELPHIX_PG_PATH"`.
2. The paths mentioned inside the `DELPHIX_PG_PATH` should have read and execute permissions for a low-privileged user. Ex: DELPHIX_PG_PATH="/usr/pgsql-9.6/bin:/var/lib/pgsql/9.6/data;"

3. For the low-privileged user, If the `DELPHIX_PG_PATH` environment variable is not present, then the plugin traverses through the `/var` & `/opt` directories. And, if the installations are present in these parent folders, then you should have read and execute permissions for the whole set of directories that has the installation.
4. The `<postgres bin>` path [each directory in the path] should have read and execute permissions for the low privilege user.
5. The low-privilege user should be able to execute the `<postgres bin>/postgres --version` command. So, the plugin needs read and execute permissions for [just] `<postgres bin>/postgres` script. Other binaries/scripts will be run using the high-privileged user.

Managing PostgreSQL environments and hosts

This topic section the attributes of Postgres environments such as information for the Delphix Connector and covers the following topics:

- Setting up PostgreSQL environments: An overview
- Environment attributes for hosts with PostgreSQL
- Adding a PostgreSQL environment
- Adding an installation to a PostgreSQL environment
- Changing the hostname or IP address for PostgreSQL source and target environments

Setting up PostgreSQL environments: An overview

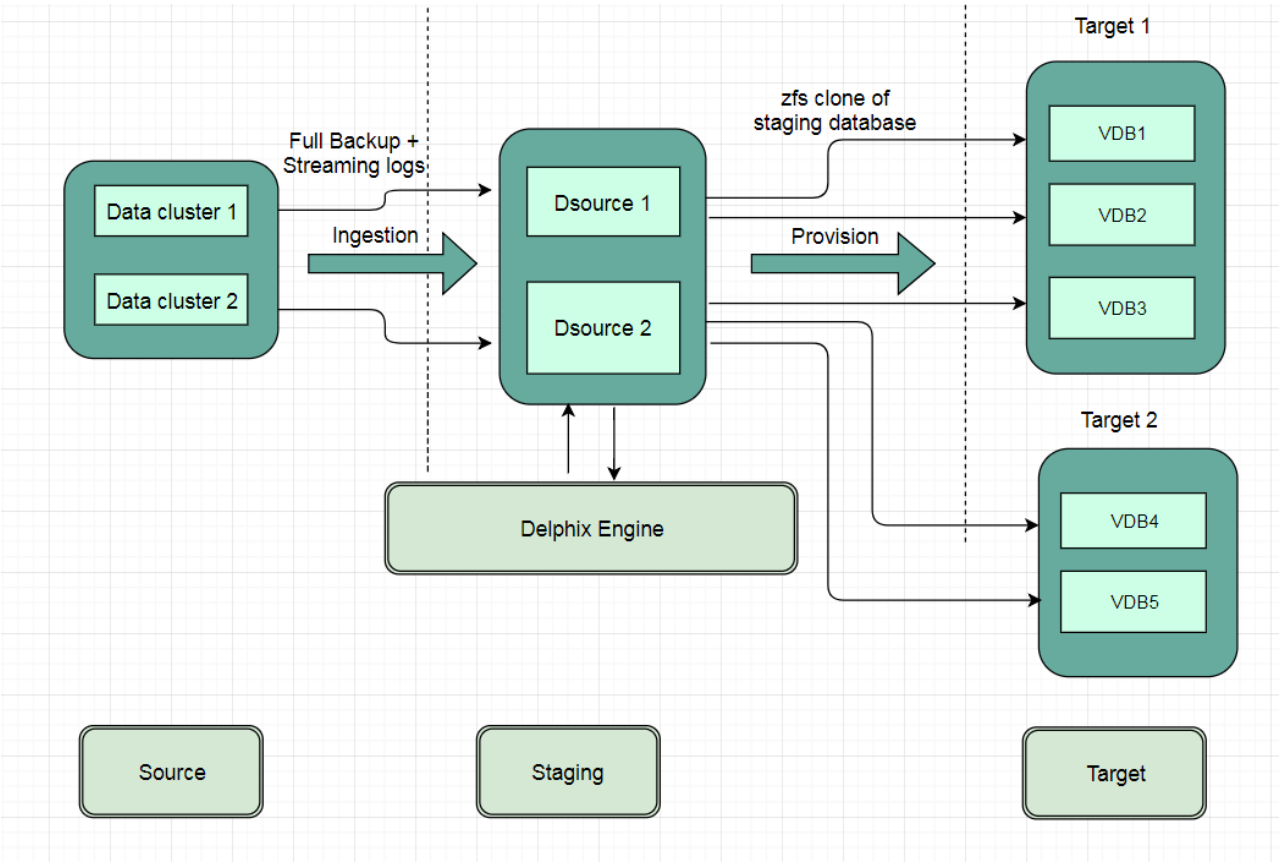
There are broadly three ways in which PostgreSQL source database can be made in sync with the staging Database:

1. Delphix Initiated Backup Ingestion + PostgreSQL Streaming Replication between Source and Staging Database.
2. Customer Initiated Backup Ingestion + PostgreSQL Streaming Replication between Source and Staging Database.
3. Customer Initiated Backup Ingestion + Source Database WAL logs application on Staging Server.

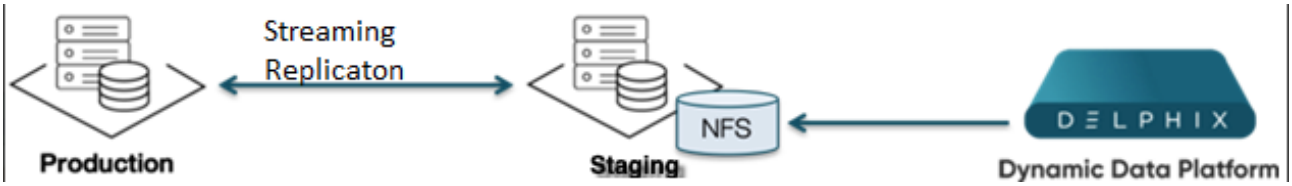
Each of the above approaches are described below:

Delphix for PostgreSQL architecture with Delphix initiated backup ingestion + PostgreSQL streaming replication between source and dSource

Below is the brief architecture of Delphix Engine support for PostgreSQL database with Delphix Initiated Backup + PostgreSQL Streaming Replication between Source and Staging Database.

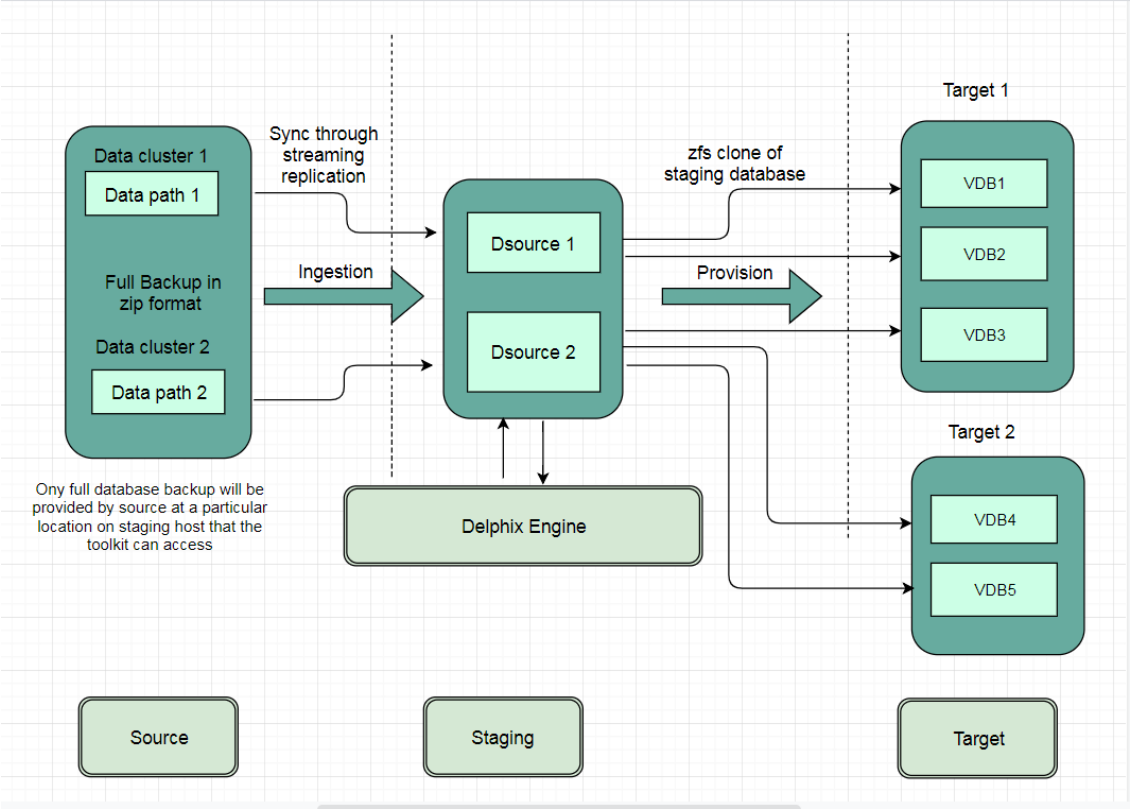


Block Diagram of Linking Architecture Between PostgreSQL Environments and the Delphix Initiated Backup Platform.



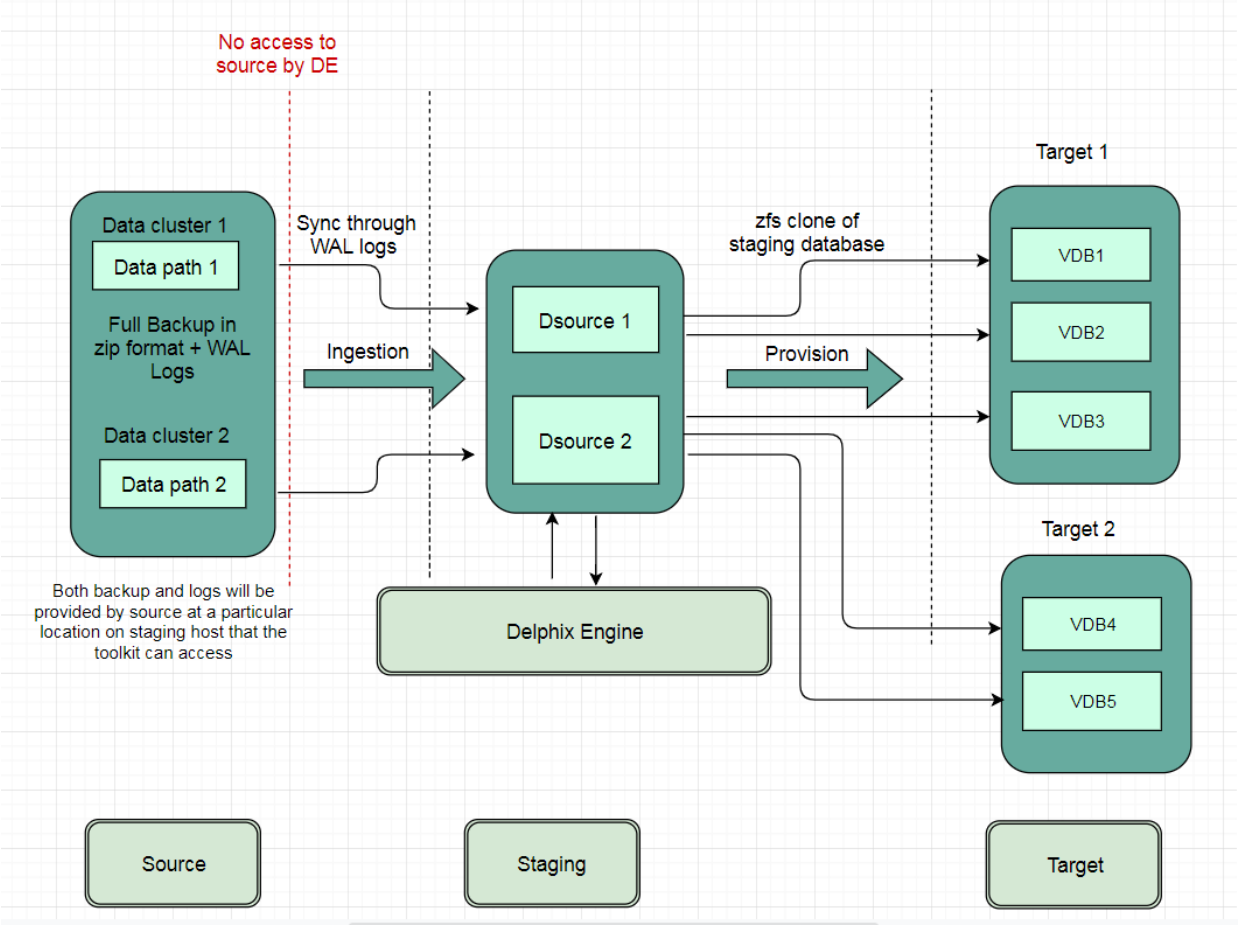
Delphix for PostgreSQL architecture with customer initiated backup ingestion + PostgreSQL streaming replication between source and dSource

Below is the brief architecture of Delphix Engine support for the PostgreSQL database with External Backup + PostgreSQL Streaming Replication between Source and Staging Database.



Delphix for PostgreSQL architecture with customer initiated backup ingestion + WAL apply for resync between source and dSource

Below is the brief architecture of Delphix Engine support for the PostgreSQL database with External Backup + WAL application between Source and Staging Database.



Types of PostgreSQL environments

At a high level, the Delphix Engine maintains a logical representation of the source database files, from which one can provision virtual databases (VDBs). In order to link a data source and provision a VDB, the following types of environments are required:

A source environment is where the un-virtualized source database runs. The Delphix Engine uses the backup, restore, and replication features of the PostgreSQL DBMS to maintain its internal representation of the source database, to be used for provisioning VDBs. The Staging host must be able to connect to the source environment in order to orchestrate the backup, restore, and replication functionality necessary to keep its representation synchronized with the source database. The Delphix Engine is designed to have a minimal impact on the performance of the source database and the source environment.

A target environment is where virtualized databases run. PostgreSQL target environments serve two purposes:

1. Since PostgreSQL does not provide a native incremental backup API, a warm standby server (in other words, one in log-shipping mode) must be created where all database files are stored remotely on a Delphix Engine. We refer to the creation and maintenance of this staging database as validated sync. During validated sync, we retrieve data from the source, roll the staged database forward, ensure that all the components necessary for provisioning a VDB have been validated, and create a snapshot of that data state. The result of validated sync is both a TimeFlow with consistent points from which you can provision a VDB and a faster provisioning process because there is no need for any database recovery when provisioning a VDB. In order to create a staging database, you must designate a target environment for this task when linking a dSource. During the linking process, database files are exported over the network to the target environment, where

the staging database instance runs as a warm standby server. A target environment that hosts one or more staging databases is referred to as a staging target for validated sync.

2. Once a staging database has been set up, you can provision virtual databases from any of the discreet snapshots along the TimeFlow mentioned above to any compatible target environment (for more information, see Requirements for PostgreSQL hosts and databases). Database files are exported over the network to the target environment, where the virtual database instance runs.

Workflow for PostgreSQL environments

Prior to provisioning a virtual database, a compatible target environment must be added to the Delphix Engine. This may be the same target environment as that used for the staging instance, or it may be a different target environment.

Once an environment is added to the Delphix Engine, environment discovery takes place. Environment discovery is the process of enumerating PostgreSQL installations and configurations when a source or target environment is added to the Delphix Engine. The discovery process is repeated during environment refresh in order to detect new PostgreSQL installations and clusters.

Environment attributes for hosts with PostgreSQL

This topic describes the attributes of PostgreSQL environments. Below you will see a section for common environment attributes shared by all types of environments as well as Postgres specific ones.

Procedure

1. Login to the Delphix Management application.
2. Click **Manage**.
3. Select **Environments**.
4. In the Environments panel, click the name of an environment to view its attributes.
5. Next to **Attributes**, click the Pencil icon to edit an attribute.

Common environment attributes

Attribute	Description
Environment Users	The users for that environment. These are the users who have permission to ssh into an environment or access the environment through the Delphix Connector. For more information on the environment user requirements, see the Requirements topics for specific data platforms.
Host Address	The IP address of the environment host.
DSP KeyStore Path	The path to the user-managed DSP Keystore.
DSP KeyStore Alias	The lowercase alias to use inside the user-managed DSP Keystore.
DSP KeyStore Password	The password for the user-managed DSP Keystore.
DSP TrustStore Path	The path to the user managed DSP truststore.
DSP TrustStore Password	The password for the user managed DSP truststore.
OS	The name of the host operating system.
Version	The version of the host operating system.
Release	The release of the host operating system.
Time Zone	The timezone of the host operating system.
Total RAM	The amount of RAM on the host machine.

Attribute	Description
Processor Type	The processor type of the host machine.
Traceroute	Traceroute info from target host to Delphix Engine.
Notes	Any other information you want to add about the environment.
Java Development Kit	The currently selected JDK kit will be shown.
Java Development Kit (JDK) Path	Location of the Java Development Kit (JDK) used for the host. Only specified if the feature to provide your own JDK is enabled, otherwise, the defaults are used per our Java support matrix.

PostgreSQL environment attributes

There are no postgres-specific environment attributes.

Adding a PostgreSQL environment

This topic describes how to add a PostgreSQL Staging environment.

Prerequisites

- Make sure that the staging environment in question meets the requirements described in Requirements for PostgreSQL hosts and databases and Prerequisites for Discovery operation described in the Prerequisites for privilege elevation using DLPX_DB_EXEC script section.

Procedure

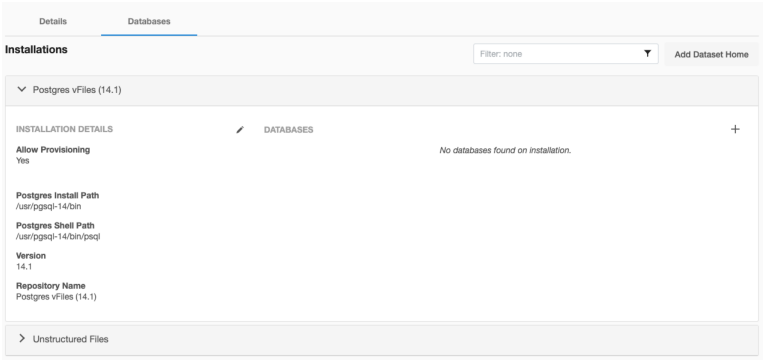
1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Next, to Environments, click the Actions (**...**) menu and select **Add Environment**.
5. In the Add Environment dialog, select **Unix/Linux.**
6. Select **Standalone Host**.
7. Click **Next**.
8. Enter Name for the Environment.
9. Enter the **Host IP address** or **hostname**.
10. Enter the **SSH port**. The default value is 22.
11. Enter an **OS Username** for the Environment. If a low-privileged OS user is used, make sure the Prerequisites for Privilege Elevation using DLPX_DB_EXEC Script are met.
12. Select **Login Type**. — Username and Password - enter the OS username and password — Username and Public Key - enter the OS username. — Password Vault - select from an existing Enterprise Password Vault
    Note:
    Using Public Key Authentication
    If you want to use public-key authentication for logging into your Unix-based environment, there are two options: use the engine's key pair or provide a key pair for this environment.
    To use the engine's key pair:
    a. Select **Public Key** for the **Login Type.**
    b. Click **View Public Key**.
    c. Copy the public key that is displayed, and append it to the end of your `~/.ssh/authorized_keys` file. If this file does not exist, you will need to create it.
        i. Run chmod 600 ~/.ssh/authorized_keys to allow only the file's owner to read and write to it (make sure the file is owned by the user).
        ii. Run chmod 755 ~ to restrict access to the user's home directory so no other user may write to it.
        iii. Run chmod 700 ~/.ssh so that others cannot write to it. The ~/.ssh directory cannot be writable by group or other users. Otherwise, authentication will fail.
    As an alternative, you can provide a key pair specific for this environment via the API or CLI. See Option 2 in this CLI Cookbook article for instructions.
13. For Password, enter the password associated with the user in step 11.
14. If you want to use Public Key Encryption for logging into your environment:
    a. Select Public Key for the Login Type.
    b. Click View Public Key.
    c. Copy the public key that is displayed, and append it to the end of your `~/.ssh/authorized_keys` file. If this file does not exist, you will need to create it.
        i. Run chmod 600 ~/.ssh/authorized_keys to allow only the file's owner to read and write to it (make sure the file is owned by the user).

ii. Run chmod 755 ~ to restrict access to the user's home directory so no other user may write to it.
iii. Run chmod 700 ~/.ssh so that others cannot write to it. The ~/.ssh directory cannot be writable by group or other users. Otherwise, authentication will fail.
The public key needs to be added only once per user and per environment.
You can also add public key authentication to an environment user's profile by using the command-line interface, as explained in the topic CLI Cookbook: Setting up SSH key authentication for UNIX environment users

15. For Password Login, click Verify Credentials to test the username and password.
16. Enter **Toolkit Path** (make sure Toolkit path does not have spaces).
17. To provide your own Oracle Java select the **Provide my own JDK** checkbox and click **Next**.
18. In the Java Development Kit tab enter the absolute path to your Oracle JDK and click **Next**.
19. Click **Submit**.

As the new environment is added, you will see two jobs running in the Delphix platform Job History, one to Create and Discover an environment, and another to create an environment. When the jobs are complete, you will see the new environment added to the list in the Environments tab. If you do not see it, click the Refresh icon in your browser.

Once the environment is discovered, further linking would require adding a source config to the above-discovered installation. Please refer to Linking a PostgreSQL dSource for more information.



Post requirements

To view information about an environment after you have created it:

1. Click **Manage**.
2. Select **Environments**.
3. Select the environment name.

Adding an installation to a PostgreSQL environment

If a new PostgreSQL Installation is added to the environment, then in order to reflect it into the Delphix Engine, it is required to refresh the environment. Please refer to the "Environment Refresh" section at Managing environments and hosts for more details.

This topic describes how to refresh a PostgreSQL environment to reflect the changes if any.

Changing the hostname or IP address for PostgreSQL source and target environments

Below is the procedure to change the Host Address of the PostgreSQL Source and Target Environments (If the Host address of the underlying Source and Target server has also changed):

Procedure

1. Disable the dSources.
2. Disable the VDBs.
3. Login to the **Delphix Management** application.
4. Click **Manage**.
5. Select **Environments**.
6. Click on the existing environment name you want to modify and open the environment information screen.
7. In the **Attributes** section, click the **Pencil** icon located next to it.
8. Update the **Host address** to the new one and save the results by clicking on the **Save (Tick)** icon. The new Host address will now be updated.
9. Re-enable the dSources.
10. Re-enable the VDBs.

Linking data sources and syncing data with PostgreSQL

Creating a dSource will ingest data from the source and create a dSource on the engine. The dSource is an object that the Continuous Data Engine uses to create and update virtual copies of your database. As a virtualized representation of your source data, it cannot be managed, manipulated, or examined by database tools.

This section covers the following topics:

Linking PostgreSQL data sources: Overview

This topic describes basic concepts behind the creation of a dSource from PostgreSQL instance.

For an overview of all dSource related actions, please visit Managing data sources and syncing data in the Getting Started section of the Datasets documentation.

When linking a dSource from a PostgreSQL source database, Delphix offers different methods of capturing backup information:

1. dSource created with External Backup and syncing data with WAL files.
2. dSource created with External Backup and syncing data with PostgreSQL Streaming Replication.
3. dSource created with Delphix Initiated Backup, leveraging PostgreSQL Streaming Replication to sync data from the source database.
4. dSource created with Delphix Initiated Backup, leveraging Single database ingestion to sync data from the source database.
5. dSource created with the Staging Push method.

If you are using an external backup and have checked both options above, then PostgreSQL Streaming Replication will take precedence over syncing with WAL files.

For Delphix initiated and external backups, the streaming replication is monitored by reading the Postgres database logs every minute. If the streaming replication stops due to any reason, a fault is generated and an error message is displayed under the **Status tab**.
The snapshot operation will also fail if the streaming replication stops with a fault.

After obtaining the initial snapshot and linking the dSource, the Delphix Engine keeps the staging and the source database in sync by monitoring the source database for new transaction logs on the staging target and then applying those transaction logs on the staging database. PostgreSQL streaming replication protocol is used to achieve data replication between the source database and the staging database.

We can also use the source host as a staging host because the PostgreSQL database runs on different ports.

Data management settings for PostgreSQL data sources

This topic describes advanced data management settings for dSources.

When linking a dSource, you can use custom data management settings to improve overall performance and match the needs of your specific server and data environment. If no specific settings are required, leverage default data management settings.

Accessing data management settings

There are three ways to set or modify data management settings for dSources:

1. During the dSource linking process, in the Policies tab of the Add dSource wizard, select the **Retention Policy**.
2. Or select **Manage**, then select **Policies**. To manage retention on target after the dSource snapshots have been deleted from the replication source, select the **Replica Retention** tab.
3. In the Policies screen, select the **Retention** tab.
4. To add a new retention policy click the **+Retention** button, and to add a new replica retention policy on the replication target, click the **+Replica Retention** button.
5. This will open the Retention Policy wizard. Enter a policy name, select your retention periods and click Submit. For more information, see Policies for scheduled jobs

Retention policies - retention/replica retention

Retention policies define the length of time Delphix Engine retains snapshots within its storage. To support longer retention times, you may need to allocate more storage to the Delphix Engine. The retention policy – in combination with the SnapSync policy – can significantly impact the performance and storage consumption of the Delphix Engine.

Replica Retention policies define how long the snapshots are retained on replicated namespaces for dSources and VDBs after they have been deleted on the replication source. Normally, the snapshots that have been deleted on the replication source engine are also deleted on the replication target engine. A new retention policy is introduced to provide an extended lifetime of such snapshots on the replication target.

Benefits of longer retention

With increased retention time for snapshots and logs, you allow a longer (older) rollback period for your data.

Common use cases for longer retention include:

- SOX compliance
- Frequent application changes and development
- Caution and controlled progression of data
- Reduction of project risk
- Restore to older snapshots

Linking a PostgreSQL dSource

Overview of the linking process

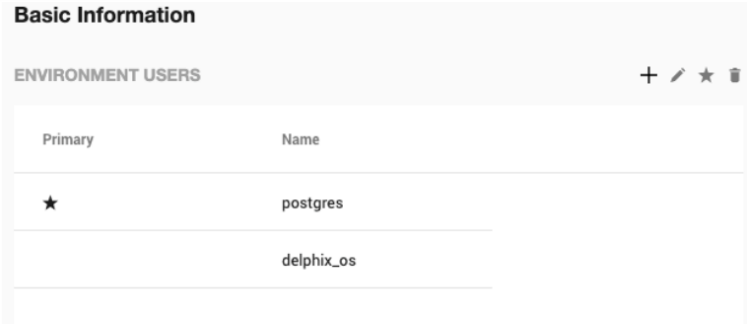
The **PostgreSQL plugin version 2.0.0** and above provides the following ways to create a dSource:

- dSource created with Delphix Initiated Backup - there are two ways to do this:
  - Cluster level backup via pg_basebackup and keep in sync through PostgreSQL Streaming Replication
  - Single database backup via pg_dump and pg_restore
- dSource created with External Backup - there are two ways to do this:
  - keep in sync with the source through WAL files
  - keep in sync with the source through PostgreSQL Streaming Replication

> If both options are used - **keep in sync with the source through WAL files** and **keep in sync with the source through PostgreSQL Streaming Replication,** then **PostgreSQL Streaming Replication** will take precedence, and syncing through WAL files will not be considered.

With PostgreSQL plugin version 2.1.0 and above, the end-user can use the Privilege Elevation feature:

- Initially, the discovery operation will be executed by the Environment OS user that was provided while adding the environment.
- If you are using the two environment users, make sure both the user can access [read+write+execute] the toolkit path & the scratch path: `<toolkit_path>/Delphix_COMMON_<id>/plugin/postgres-vsdk_<id>/` .



- Either create all the directories/files required by the dSource ingestion mechanisms using the high-privileged user or they should be accessible to the high-privileged user.
  - **External Backup Mechanism:**
    - PostgreSQL Backup File Path (Directory), with a minimum of read and execute permissions for the high-privileged user.
    - The provided backups (either the zip or full-backup tarballs) should be readable by the high-privileged user.
    - PostgreSQL Backup WAL Log Files Path (Directory), with read, write and execute permissions for the high-privileged user.
    - The WAL logs within the WAL Log Files Path should be readable by the high-privileged user.
  - **Single DB Backup:**
    - Postgres Backup Dump Directory, with a minimum of write and execute permissions for the high-privileged user.
    - The `Staging postgresql.conf` config file should be readable by the high-privileged user.

With PostgreSQL plugin version 3.2.0, an enhancement has been made to the External Backup mechanism. Previously the only way to provide source backup was via a compressed ZIP file(s) (in the format `PostgreSQL_T<YYYYMMDD>.zip`). Now we can provide backups in the following ways:

- Method 1: **Using structured folders** [highest precedence]
  - Place the *pg_basebackup* tar output inside the folder(s) named `PostgreSQL_T<YYYYMMDDhhmmss>`; created within the backup path.

```
# As an example
$ ls -l <backup_path>/PostgreSQL_T20230224033939
-rw-------. 1 postgres postgres  9093632 Feb 24 03:39 24578.tar
-rw-------. 1 postgres postgres  9093632 Feb 24 03:39 24579.tar
-rw-------. 1 postgres postgres  9093632 Feb 24 03:39 24580.tar
-rw-------. 1 postgres postgres   294300 Feb 24 03:39 backup_manifest
-rw-------. 1 postgres postgres 26959360 Feb 24 03:39 base.tar
-rw-------. 1 postgres postgres 16778752 Feb 24 03:39 pg_wal.tar
```

- Method 2: **Using the backup path directly** [preference given over ZIP files placed within the same backup path]
  - Place the *pg_basebackup* tar output inside the backup path directly

```
# As an example
$ ls -l <backup_path>
-rw-------. 1 postgres postgres  9093632 Feb 24 02:12 24578.tar
-rw-------. 1 postgres postgres  9093632 Feb 24 02:12 24579.tar
-rw-------. 1 postgres postgres  9093632 Feb 24 02:12 24580.tar
-rw-------. 1 postgres postgres 26959360 Feb 24 02:12 base.tar
-rw-------. 1 postgres postgres 16778752 Feb 24 02:12 pg_wal.tar
-rw-r--r--. 1 postgres postgres  6883130 Feb 24 02:28
 PostgreSQL_T20230224.zip
```

- Method 3: **Using the ZIP files** [existing method]
  - Place the ZIP file which contains the *pg_basebackup* output inside the backup path. Make sure that the backup files are available directly at the root level of the ZIP. The following sections provide step by steps instructions on how to do this.

```
# As an example
# The backup files are available at the root of the ZIP file
$ zipinfo PostgreSQL_T20230224.zip
Archive:  PostgreSQL_T20230224.zip
Zip file size: 6883130 bytes, number of entries: 5
-rw-------  3.0 unx  9093632 bx defN 23-Feb-24 02:12 24578.tar
-rw-------  3.0 unx  9093632 bx defN 23-Feb-24 02:12 24579.tar
-rw-------  3.0 unx  9093632 bx defN 23-Feb-24 02:12 24580.tar
-rw-------  3.0 unx 26959360 bx defN 23-Feb-24 02:12 base.tar
-rw-------  3.0 unx 16778752 bx defN 23-Feb-24 02:12 pg_wal.tar
5 files, 71019008 bytes uncompressed, 6882378 bytes compressed:  90.3%
```

With PostgreSQL plugin version 4.0.0, support for `pg_dumpall` has been added with Single Database Ingestion to get the global objects like roles and tablespaces.

Creating a dSource with external backups using WAL Logs

The following procedure provides details on creating a dSource with External Backup which uses WAL logs to keep in sync with the source.

Prerequisites

- The source and staging instances must meet the host requirements.
- Databases must meet container requirements.
- If Privilege Elevation is to be used:
  - The `PostgreSQL Backup File Path` should have read+execute permissions for the high-privileged user.
  - The backup file(s) should have read permission for the high-privileged user.
  - The `PostgreSQL Backup WAL Log Files Path` should have read+write+execute permissions for the high-privilege user.
  - The WAL logs within `WAL Log Files Path` should have read permission for the high-privilege user.
- Else, `PostgreSQL Backup File Path` & `PostgreSQL Backup WAL Log Files Path` must be available and accessible by the Environment OS user.
- **From plugin versions greater than 3.2.0,** the *pg_basebackup* database full backup can be provided in three ways, in the order of high-to-low precedence:
  a. The backup files are placed within the organized folder(s) inside the `PostgreSQL Backup File Path` named `PostgreSQL_T<YYYYMMDDhhmmss>`;. Backups from the folder with the latest timestamp in the name will be chosen when dSource creation/resync takes place.
     i. Creation of folder(s) on the staging server:

```
DLPXPGBACKUPDIR=PostgreSQL_T$(date "+%Y%m%d%H%M%S")
mkdir -p <backup_path>/$DLPXPGBACKUPDIR
```

     ii. Taking the source backup on the staging server using pg_basebackup:

```
# For PostgreSQL versions 9.4.x,9.5.x,9.6.x
<path_to_bin>/pg_basebackup -D <backup_path>/$DLPXPGBACKUPDIR -F t
-v -w -P -x -h <source_ip_domain_name> -p <source_port> -U
<source_user>

# For PostgreSQL versions 10.x and above
<path_to_bin>/pg_basebackup -D <backup_path>/$DLPXPGBACKUPDIR -F t
-v -w -P -Xs -h <source_ip_domain_name> -p <source_port> -U
<source_user>
```

  b. The backup files are placed directly within the `PostgreSQL Backup File Path`.
     i. Make sure to redirect the *pg_basebackup* output to an empty directory and then move it to `<backup_path>`.

```
# For PostgreSQL versions 9.4.x,9.5.x,9.6.x
<path_to_bin>/pg_basebackup -D <path_to_empty_directory> -F t -v -w
-P -x -h <source_ip_domain_name> -p <source_port> -U <source_user>

# For PostgreSQL versions 10.x and above
<path_to_bin>/pg_basebackup -D <path_to_empty_directory> -F t -v -w
-P -Xs -h <source_ip_domain_name> -p <source_port> -U <source_user>

# Move files
mv <path_to_empty_directory>/* <backup_path>
```

c. The backup files are encapsulated in a ZIP file named `PostgreSQL_T<YYYYMMDD>.zip` and placed within `PostgreSQL Backup File Path`.

   i. Taking the source backup on the source server using *pg_basebackup*:

```
# For PostgreSQL versions 9.4.x,9.5.x,9.6.x
<path_to_bin>/pg_basebackup -D <destination_path> -F t -v -w -P -x

# For PostgreSQL versions 10.x and above
<path_to_bin>/pg_basebackup -D <destination_path> -F t -v -w -P -Xs
```

   ii. Creating a ZIP file that can be moved to the staging server (please make sure the backup files are at the root of the ZIP file):

```
cd <destination_path>
# The back files should be at the root of the ZIP file
DLPXPGBACKUPZIP=PostgreSQL_T$(date +"%Y%m%d").zip
zip $DLPXPGBACKUPZIP *
```

For more information refer to Requirements for PostgreSQL source hosts and databases and Prerequisites for privilege elevation using DLPX_DB_EXEC Script section.

Limitation

It is not possible to access the staging server with PostgreSQL 9.4 and PostgreSQL 9.5 versions.

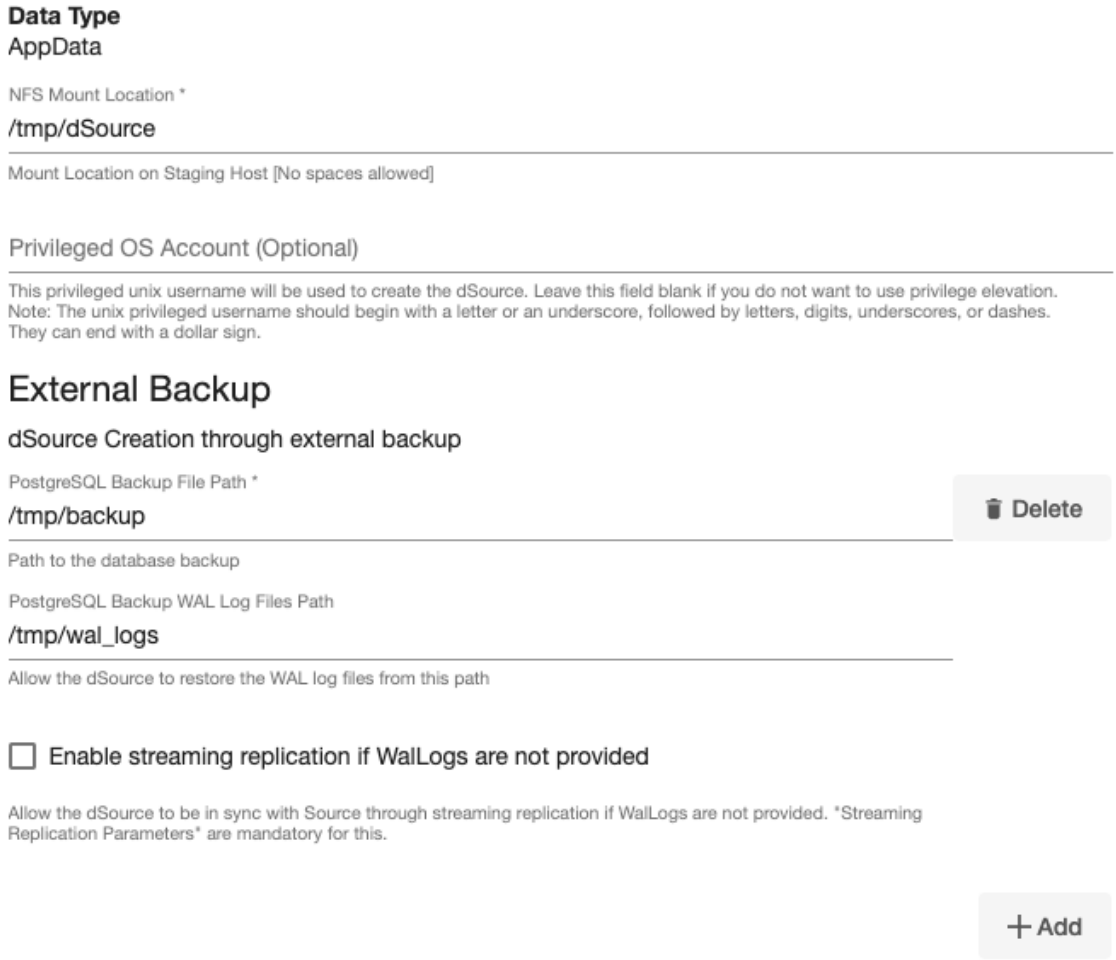
Procedure

1. Login to the **Delphix Management** application.
2. Select **Manage > Environments**.
3. From the **Databases** tab, select a repository from which you want to create your dSource and click the  icon.
4. In the **Add Database** dialog window enter the **Name** for your source config and click **Add**.
5. Select your source config and click the **Add dSource** link located to the right.
6. In the **Source** tab, do the following:
   a. In the **NFS Mount Location** field, enter a mount location on the staging host.
   b. If you want to use Privilege Elevation, provide the **Privileged OS Account** username. Make sure you meet all the requirements described in the Prerequisites for Privilege Elevation using DLPX_DB_EXEC Script section.

Note:
You will currently not be able to update the Privileged OS account after dSource creation. The only way to achieve the change is to recreate the dSource.
If the Environment OS user will be used to create the dSource, make sure you have the following privileges:

- access to the Postgres commands
- read+write+execute permissions on the `unix_socket_directories` configured in `postgresql.conf` [Default directories: `/var/run/postgresql` & `/tmp`]

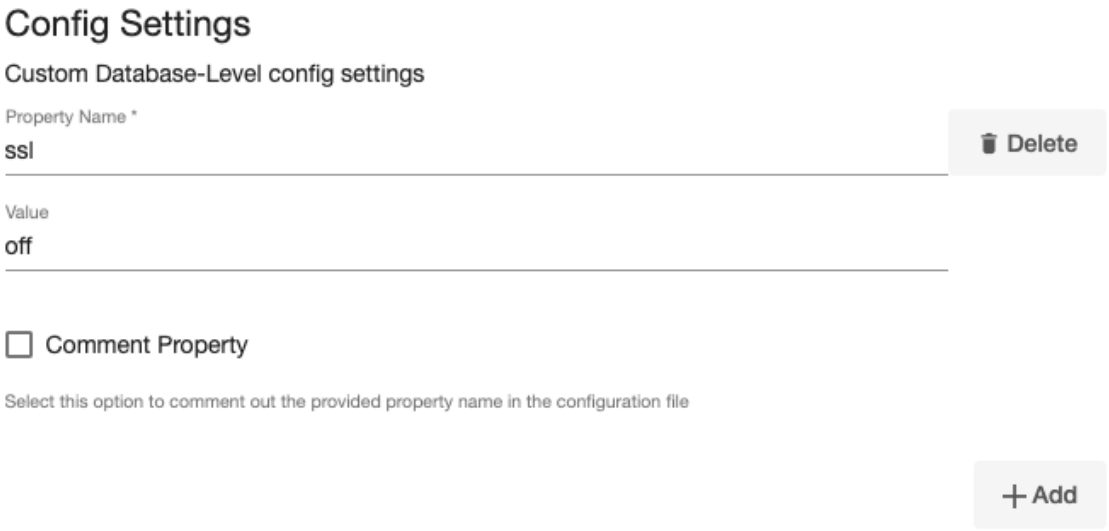
c. Click on the **+Add** icon located next to **External Backup** Parameter.
Note:
The Add Button provided for External Backup is clickable multiple times but should be used **ONLY** once since the solution is built only to support a single backup path. Providing Multiple Paths will error out the linking process.
d. **PostgreSQL Backup File Path** - the location of the backup file
e. **PostgreSQL Backup WAL Log Files Path** - the location of the WAL log files



Note:
Once the dSource is up using the Full External Backup provided on the staging environment, the dSource will continue to run in recovery mode and will wait for the next expected WAL file under the specified **PostgreSQL backup WAL log Files** path. When the expected WAL files are placed on this path, the dSource applies this WAL file and starts waiting for the next expected WAL files in series. If the WAL file provided is invalid or not in the sequence, then the dSource will throw an error on UI with the information of the expected WAL filename when the next dSource snapshot is taken. In the case

of a WAL chain break, users need to either resynchronize the dSource or provide the missing WAL file, then disable and enable the dSource database

f. **Staging Instance Port Number** - must be unique for a given instance.
g. Optionally, database configuration parameters can be defined during the linking operation in the Config Settings section.



h. Click **Next**.

7. In the **dSource Configuration** tab, enter a dSource name and select a group for your dSource. Adding a dSource to a dataset group lets you set Delphix Domain user permissions for that database and its objects, such as snapshots. See the topics under Users and Groups for more information.
8. In the **Data Management** tab, specify your Staging Environment, User, and Snapshot Parameters. Select the **Resynchronize dSource** checkbox if you want to resynchronize the dSource. Resynchronizing the dSource will force a non-incremental load of data from the source. This operation is similar to creating a new dSource, but avoids duplicating storage requirements and maintains timeflow history. Click **Next**.
9. In the **Policies** tab, apply policy details to the dSource if required, and then click **Next**.
10. In the **Hooks** tab, select a Hook Point and then click **+** to add a script to run at that point. You can define scripts to run at multiple hook points in the process.
11. In the **Summary** tab, review the configuration profile for the dSource.
12. Click **Submit**. The Delphix Engine initiates two jobs to create the dSource: DB_Link and DB_Sync. You can monitor these jobs by clicking **Active Jobs** from the top menu bar, or by selecting **System > Event Viewer**. When the jobs have successfully been completed, the database icon will change to a dSource icon on the **Environments > Host > Databases** screen, and the dSource will also appear in the list of Datasets under its assigned group.

The dSource Configuration Screen

After you have created a dSource, the dSource Configuration tab allows you to view information about it and make modifications to its policies and permissions. In the Datasets panel, select the dSource you wish to examine. You can now choose the configuration tab to see information such as the Source files, Data Management configuration, and Hook Operations

Creating a dSource with external backups using streaming replication

The following procedure provides details on creating a dSource with External Backup which uses streaming replication to keep in sync with the source.

Prerequisites

- The source and staging instances must meet the host requirements
- Databases must meet container requirements.
- If Privilege Elevation is being used:
  - The `PostgreSQL Backup File Path` should have read+execute permissions for the high-privileged user.
  - The backup file `>>` should have read permission for the high-privileged user.
- Else, PostgreSQL Backup Files Path must be available and accessible by the Environment OS user.
- **For plugin versions greater than 3.2.0,** the *pg_basebackup* database full backup can be provided in three ways, in the order of high-to-low precedence:
  a. The backup files are placed within the organized folder(s) inside the `PostgreSQL Backup File Path`, named `PostgreSQL_T<YYYYMMDDhhmmss>`. Backups from the folder with the latest timestamp in the name will be chosen when dSource creation/resync takes place.
     i. Creation of the folder(s) on the staging server:

```
DLPXPGBACKUPDIR=PostgreSQL_T$(date "+%Y%m%d%H%M%S")
mkdir -p <backup_path>/$DLPXPGBACKUPDIR
```

     ii. Taking the source backup on the staging server using *pg_basebackup*:

```
# For PostgreSQL versions 9.4.x,9.5.x,9.6.x
<path_to_bin>/pg_basebackup -D <backup_path>/$DLPXPGBACKUPDIR -F t
-v -w -P -x -h <source_ip_domain_name> -p <source_port> -U
<source_user>

# For PostgreSQL versions 10.x and above
<path_to_bin>/pg_basebackup -D <backup_path>/$DLPXPGBACKUPDIR -F t
-v -w -P -Xs -h <source_ip_domain_name> -p <source_port> -U
<source_user>
```

  b. The backup files are placed directly within the `PostgreSQL Backup File Path.`
     i. Make sure to redirect the *pg_basebackup* output to an empty directory and then move it to <backup_path>.

```
# For PostgreSQL versions 9.4.x,9.5.x,9.6.x
<path_to_bin>/pg_basebackup -D <path_to_empty_directory> -F t -v -w
-P -x -h <source_ip_domain_name> -p <source_port> -U <source_user>

# For PostgreSQL versions 10.x and above
<path_to_bin>/pg_basebackup -D <path_to_empty_directory> -F t -v -w
-P -Xs -h <source_ip_domain_name> -p <source_port> -U <source_user>

# Move files
mv <path_to_empty_directory>/* <backup_path>
```

  c. The backup files are encapsulated in a ZIP file named `PostgreSQL_T<YYYYMMDD>.zip` and placed within `PostgreSQL Backup File Path`.
     i. Taking the source backup on the source server using *pg_basebackup***:**

```
# For PostgreSQL versions 9.4.x,9.5.x,9.6.x
<path_to_bin>/pg_basebackup -D <destination_path>  -F t -v -w -P -x

# For PostgreSQL versions 10.x and above
<path_to_bin>/pg_basebackup -D <destination_path> -F t -v -w -P -Xs
```

ii. Creating a ZIP file that can be moved to the staging server (please make sure the backup files are at the root of the ZIP file):

```
cd <destination_path>

# The back files should be at the root of the ZIP file
DLPXPGBACKUPZIP=PostgreSQL_T$(date +"%Y%m%d").zip
zip $DLPXPGBACKUPZIP *
```

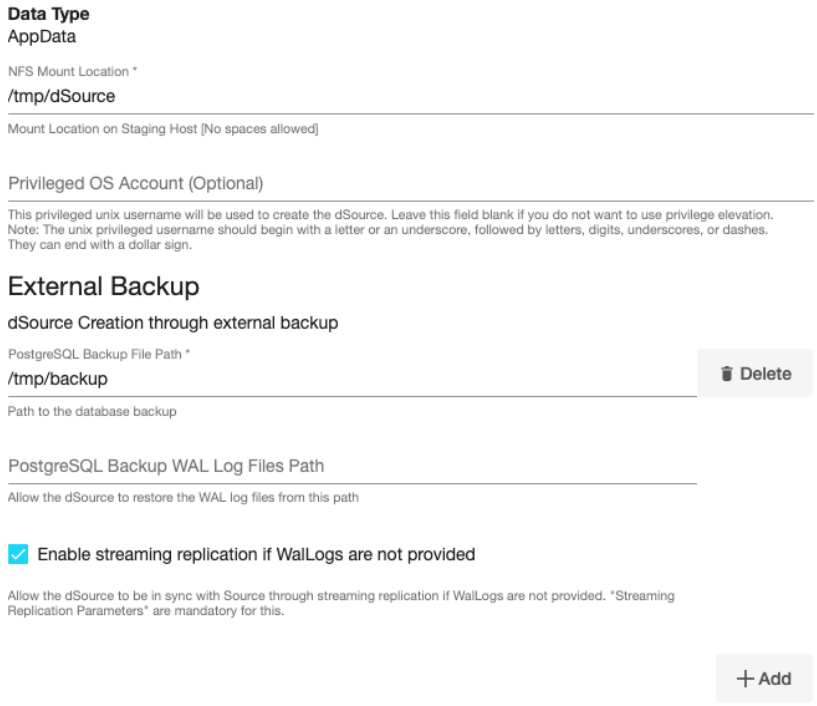
For more information refer to Requirements for PostgreSQL hosts and databases and Prerequisites for privilege elevation using DLPX_DB_EXEC script section.

Limitation

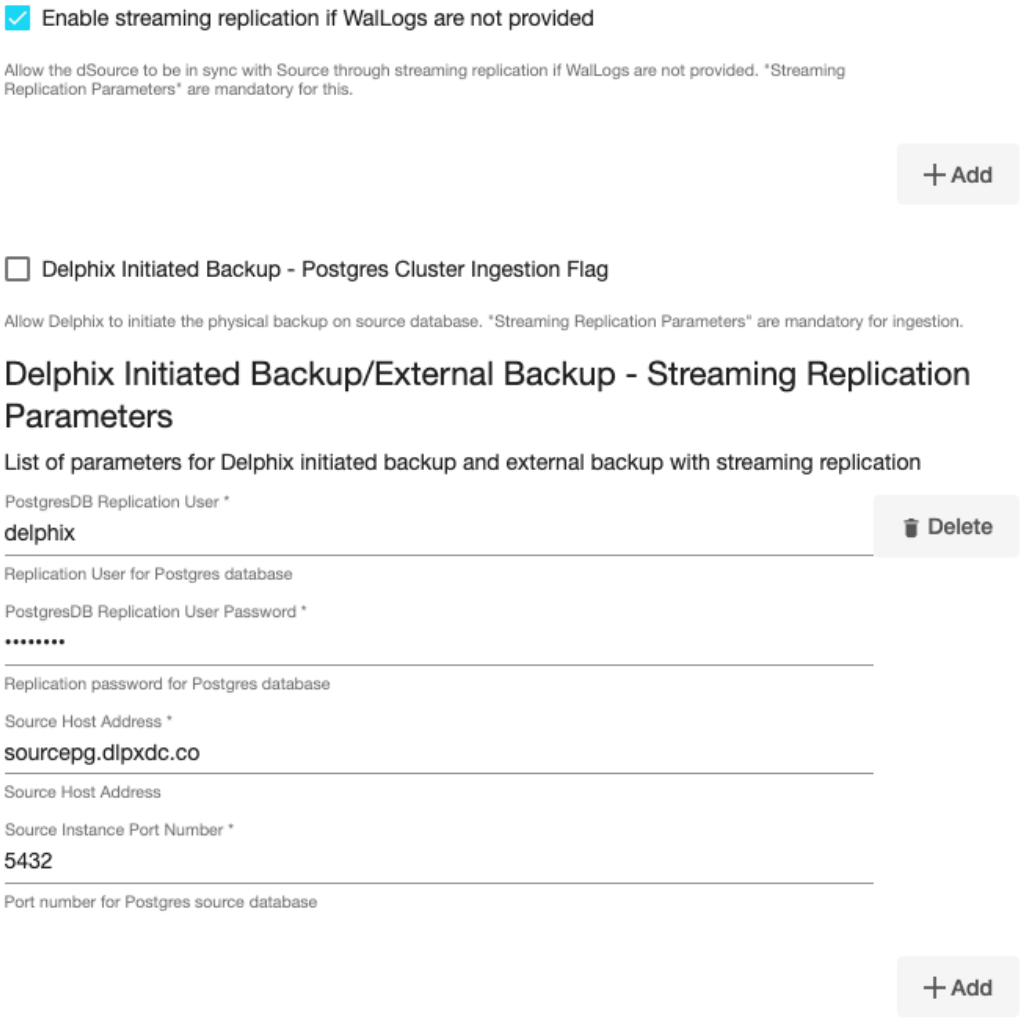
It is not possible to access the staging server with PostgreSQL 9.4 and PostgreSQL 9.5 versions.

Procedure

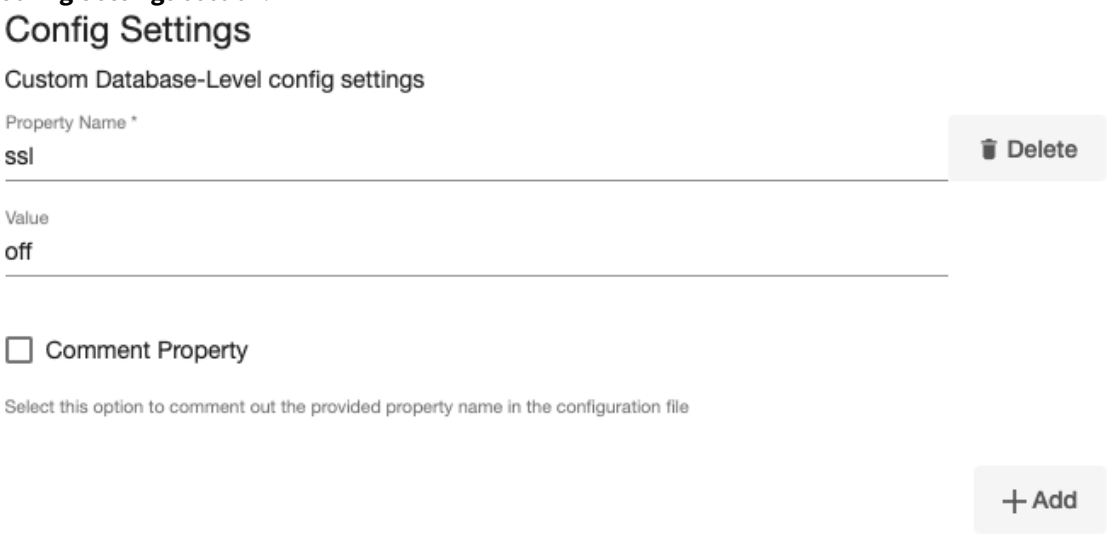
1. Login to the **Delphix Management** application.
2. Select **Manage > Environments**.
3. From the **Databases** tab, select a repository from which you want to create your dSource and click the  icon.
4. In the **Add Database** dialog window enter the **Name** for your source config and click **Add**.
5. Select your source config and click the **Add dSource** link located to the right.
6. In the **Source** tab, do the following:
    a. In the **NFS Mount Location** field, enter a mount location on the staging host.
    b. If you want to use Privilege Elevation, provide the **Privileged OS Account** username. Make sure you meet all the requirements described in the Prerequisites for Privilege Elevation using DLPX_DB_EXEC Script section.
    Note:
    You will currently not be able to update the Privileged OS account after dSource creation. The only way to achieve the change is to recreate the dSource.
    If the Environment OS user will be used to create the dSource, make sure you have the following privileges:
        - access to the Postgres commands
        - read+write+execute permissions on the `unix_socket_directories` configured in `postgresql.conf` [Default directories: `/var/run/postgresql` & `/tmp`]
    c. Click on the **+Add** icon located next to **External Backup** Parameter.
    d. **PostgreSQL Backup File Path -** the location of the backup file
    e. Select the checkbox '**Enable streaming replication if WalLogs are not provided**'



f. Click on the **+Add** icon located next to the **Delphix Initiated Backup/External Backup - Streaming Replication Parameters** option.
Note:
The Add Button provided for Delphix Initiated Backup/External Backup - Streaming Replication is clickable multiple times but should be used **ONLY** once since the solution is built only to support a single backup path. Providing Multiple Paths will error out the linking process.
The following fields are mandatory for this process :
   i. **PostgresDB Replication User**
   ii. **PostgresDB Replication User Password**
   iii. **Source Host Address**
   iv. **Source Instance Port Number**



g. **Staging Instance Port Number** - must be unique for a given instance.
h. Optionally, database configuration parameters can be defined during the linking operation in the **Config Settings** section.



Note:
After dSource creation, users can now configure the parameters and their values in the `postgresql.conf` file through the "Config Settings" section on the Delphix Engine UI. For example, to support **Replication Slots** through UI, users can now provide the "primary_slot_name" parameter and its value through the UI. This will update the already existing parameter value in the `postgresql.conf` file with the value provided by the user.
Similarly, if a user wants to disable a parameter then they can provide the parameter name and select the checkbox **Comment Property**. This will comment out the parameter in the `postgresql.conf` configuration file.

i. Click **Next.**

7. In the **dSource Configuration** tab, enter a dSource name and select a group for your dSource. Adding a dSource to a dataset group lets you set Delphix Domain user permissions for that database and its objects, such as snapshots. See the topics under Users and Groups for more information.
8. In the **Data Management** tab, specify your Staging Environment, User, and Snapshot Parameters. Select the **Resynchronize dSource** checkbox if you want to resynchronize the dSource. Resynchronizing the dSource will force a non-incremental load of data from the source. This operation is similar to creating a new dSource, but avoids duplicating storage requirements and maintains timeflow history. Click **Next**.
9. In the **Policies** tab, apply policy details to the dSource if required, and then click **Next**.
10. In the **Hooks** tab, select a Hook Point and then click **+** to add a script to run at that point. You can define scripts to run at multiple hook points in the process.
11. In the **Summary** tab, review the configuration profile for the dSource.
12. Click **Submit**. The Delphix Engine initiates two jobs to create the dSource: DB_Link and DB_Sync. You can monitor these jobs by clicking **Active Jobs** from the top menu bar, or by selecting **System > Event Viewer**. When the jobs have successfully been completed, the database icon will change to a dSource icon on the **Environments > Host > Databases** screen, and the dSource will also appear in the list of Datasets under its assigned group.

The dSource Configuration Screen

After you have created a dSource, the dSource Configuration tab allows you to view information about it and make modifications to its policies and permissions. In the Datasets panel, select the dSource you wish to examine. You can now choose the configuration tab to see information such as the Source files, Data Management configuration, and Hook Operations.

Creating a dSource with Delphix initiated backup using streaming replication

Prerequisites

- The source and staging instances must meet the host requirements
- Databases must meet container requirements. For more information refer to Requirements for PostgreSQL source hosts and databases

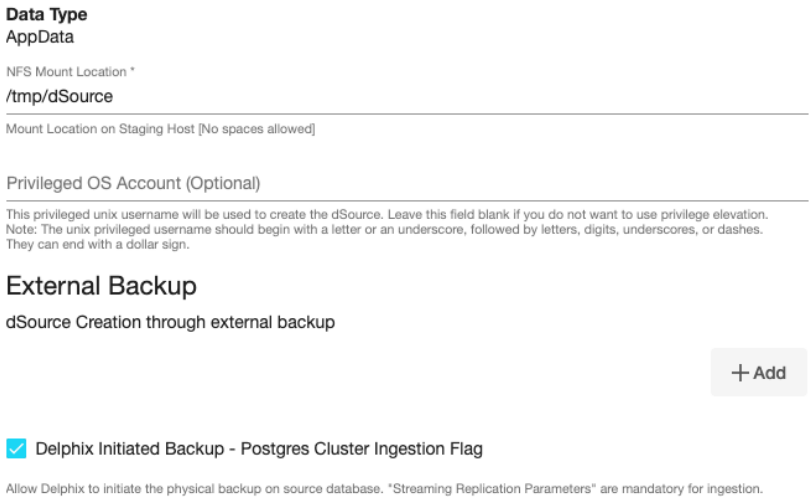
Limitation

It is not possible to access the staging server with PostgreSQL 9.4 and PostgreSQL 9.5 versions.

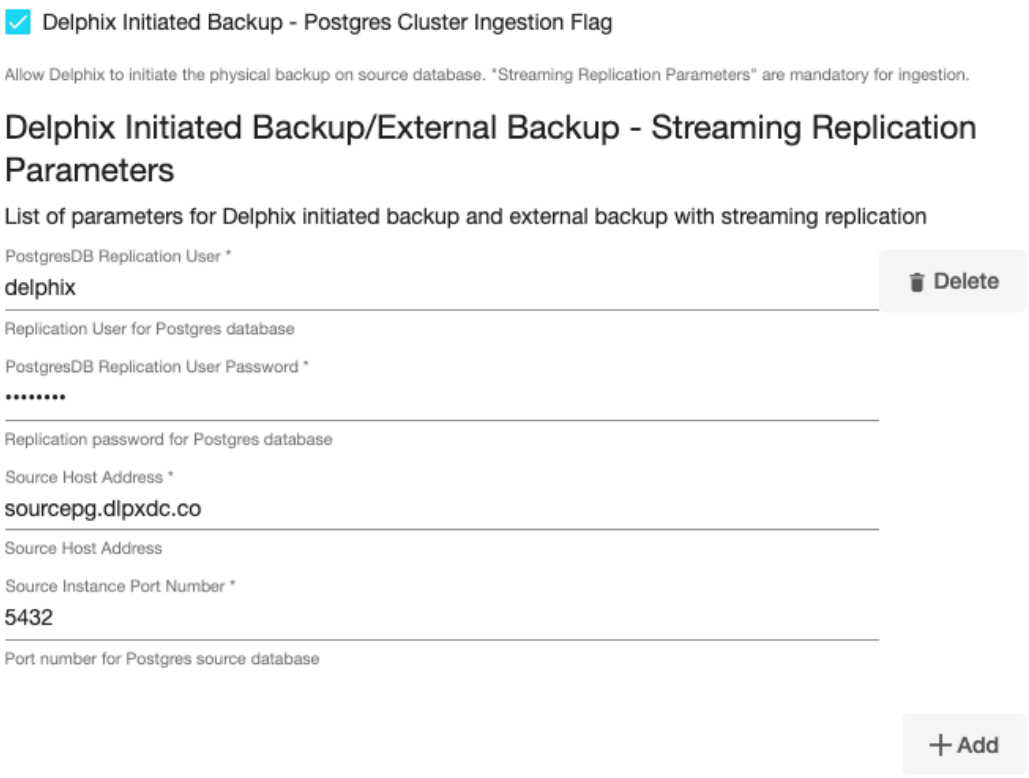
Procedure

1. Login to the **Delphix Management** application.
2. Select **Manage > Environments**.
3. From the **Databases** tab, select a repository from which you want to create your dSource and click the  icon.

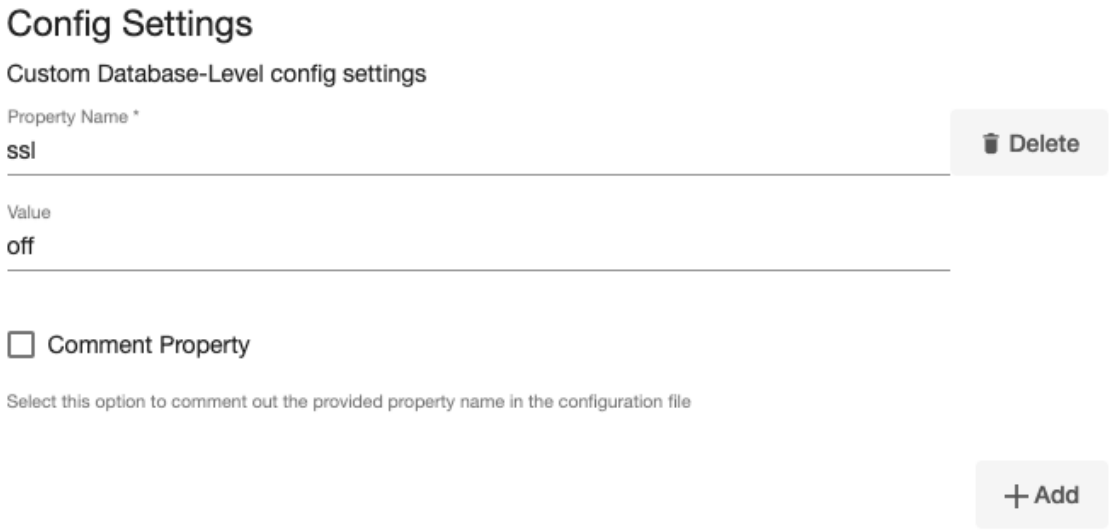
4. In the **Add Database** dialog window enter the **Name** for your source config and click **Add**.
5. Select your source config and click the **Add dSource** link located to the right.
6. In the **Source tab**, do the following:
    a. In the **NFS Mount Location** field, enter a mount location on the staging host.
    b. If you want to use Privilege Elevation, provide the **Privileged OS Account** username. Make sure you meet all the requirements described in the Prerequisites for privilege elevation using DLPX_DB_EXEC script section.
    Note:
    You will currently not be able to update the Privileged OS account after dSource creation. The only way to achieve the change is to recreate the dSource.
    If the Environment OS user will be used to create the dSource, make sure you have the following privileges:
        - access to the Postgres commands
        - read+write+execute permissions on the `unix_socket_directories` configured in `postgresql.conf` [Default directories: `/var/run/postgresql` & `/tmp`]
    c. Select the checkbox next to the **Delphix Initiated Backup - Postgres Cluster Ingestion Flag** option. Selecting this checkbox allows Delphix to initiate the physical backup on the source database.



    d. Click on the **+Add** icon located next to the **Delphix Initiated Backup/External Backup - Streaming Replication Parameters** option.
    Note : The Add Button provided for Delphix Initiated Backup - Single Database Ingestion is clickable multiple times but should be used ONLY once since the solution is built only to support a single backup path. Providing Multiple Paths will error out the linking process.
    The following fields are mandatory for this process :
        i. **Delphix Initiated Backup - Postgres Cluster Ingestion Flag** - Checkbox should be selected.
        ii. **PostgresDB Replication User**
        iii. **PostgresDB Replication User Password**
        iv. **Source Host Address**
        v. **Source Instance Port Number**



e. **Staging Instance Port Number** - must be unique for a given instance.
f. Optionally, database configuration parameters can be defined during the linking operation in the **Config Settings** section.



Note:
After dSource creation, users can now configure the parameters and their values in the `postgresql.conf` file through the "Config Settings" section on the Delphix Engine UI. For example, to support **Replication Slots** through UI, users can now provide the "primary_slot_name" parameter and its value through the UI. This will update the already existing parameter value in the `postgresql.conf` file with the value provided by the user.
Similarly, if a user wants to disable a parameter then they can provide the parameter name and

select the check box **Comment Property**. This will comment out the parameter in the `postgresql.conf` configuration file.

g. Click **Next.**

7. In the **dSource Configuration** tab, enter a dSource name and select a group for your dSource. Adding a dSource to a dataset group lets you set Delphix Domain user permissions for that database and its objects, such as snapshots. See the topics under Users and Groups for more information.
8. In the **Data Management** tab, specify your Staging Environment, User, and Snapshot Parameters. Select the **Resynchronize dSource** checkbox if you want to resynchronize the dSource. Resynchronizing the dSource will force a non-incremental load of data from the source. This operation is similar to creating a new dSource, but avoids duplicating storage requirements and maintains timeflow history. Click **Next**.
9. In the **Policies** tab, apply policy details to the dSource if required, and then click **Next**.
10. In the **Hooks** tab, select a Hook Point and then click **+** to add a script to run at that point. You can define scripts to run at multiple hook points in the process.
11. In the **Summary** tab, review the configuration profile for the dSource.
12. Click **Submit**. The Delphix Engine initiates two jobs to create the dSource: DB_Link and DB_Sync. You can monitor these jobs by clicking **Active Jobs** from the top menu bar, or by selecting **System > Event Viewer**. When the jobs have successfully been completed, the database icon will change to a dSource icon on the **Environments > Host > Databases** screen, and the dSource will also appear in the list of Datasets under its assigned group.

The dSource Configuration Screen

After you have created a dSource, the dSource Configuration tab allows you to view information about it and make modifications to its policies and permissions. In the Datasets panel, select the dSource you wish to examine. You can now choose the configuration tab to see information such as the Source files, Data Management configuration, and Hook Operations.

Creating a dSource with Delphix initiated backup using single database ingestion

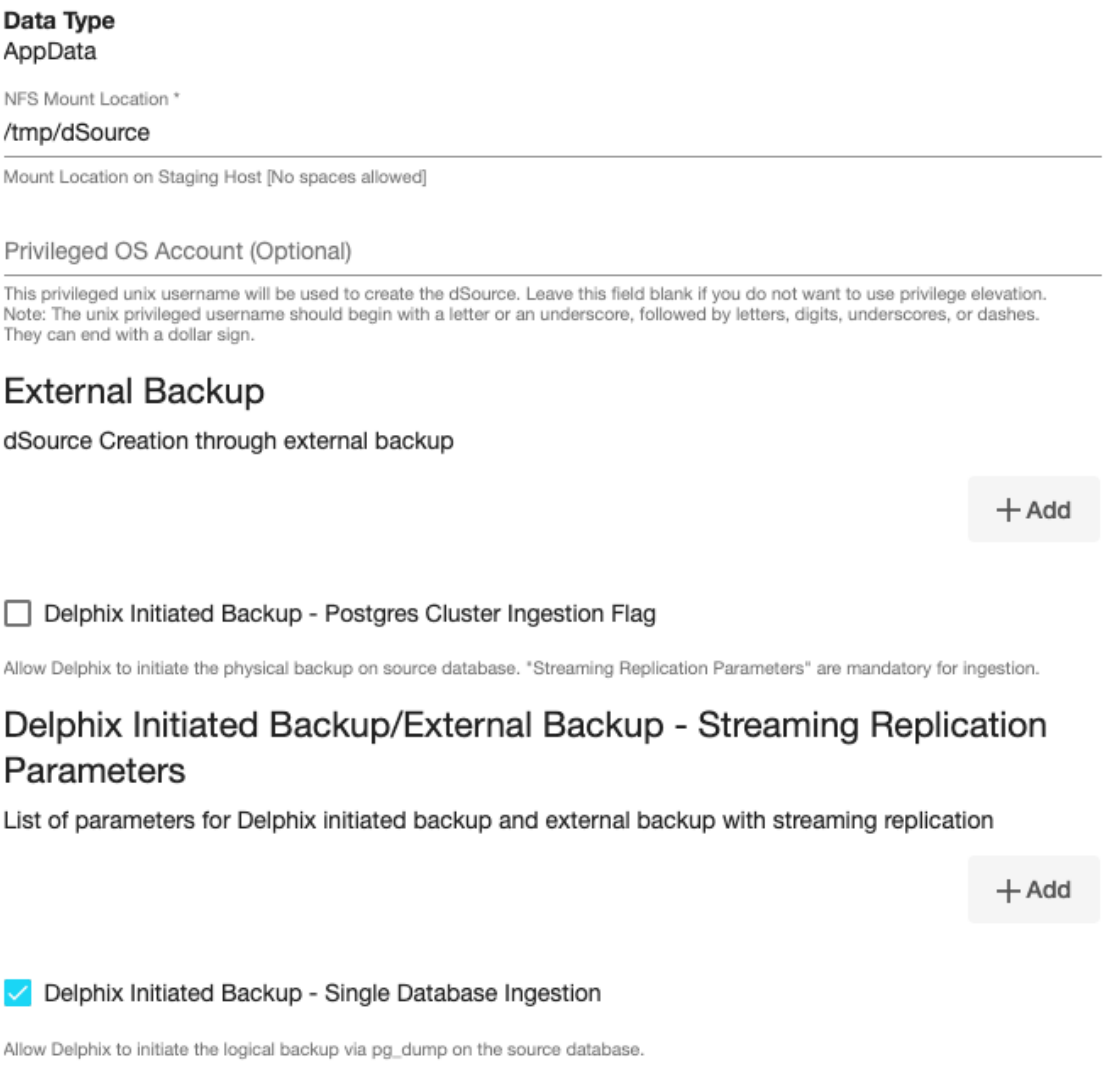
Prerequisites

- The source and staging instances must meet the host requirements
- Databases must meet container requirements
- If Privilege Elevation is being used:
  - The `Postgres Backup Dump Directory` should have write+execute permissions for the high-privileged user.
  - The `Staging postgresql.conf config file` should be accessible to the high-privileged user.
- Else, `Postgres Backup Dump Directory` & `Staging postgresql.conf` config file must be available and accessible by the Environment OS user.
  For more information refer to Requirements for PostgreSQL source hosts and databases and Prerequisites for privilege elevation using DLPX_DB_EXEC script section.

Procedure

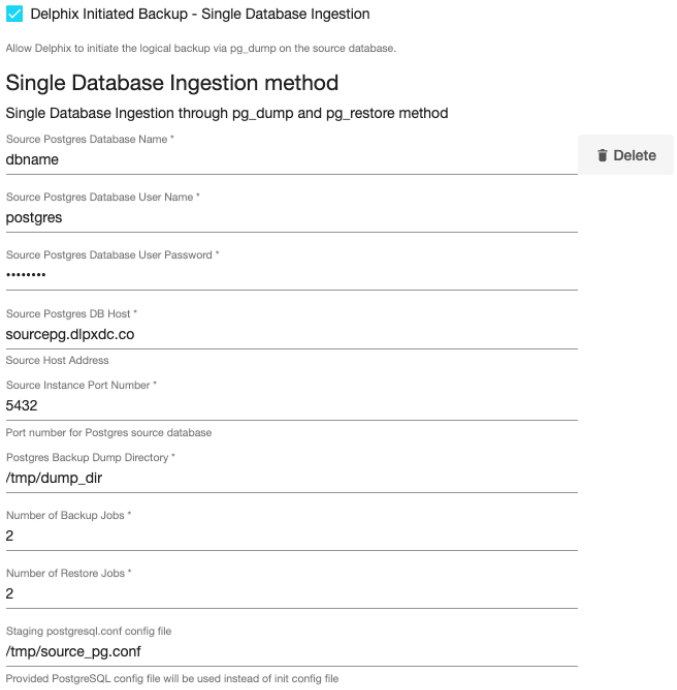
With PostgreSQL plugin version 4.0.0, support for `pg_dumpall` has been added with Single Database Ingestion to get the global objects like roles and tablespaces.

1. Login to the **Delphix Management** application.
2. Select **Manage > Environments**.

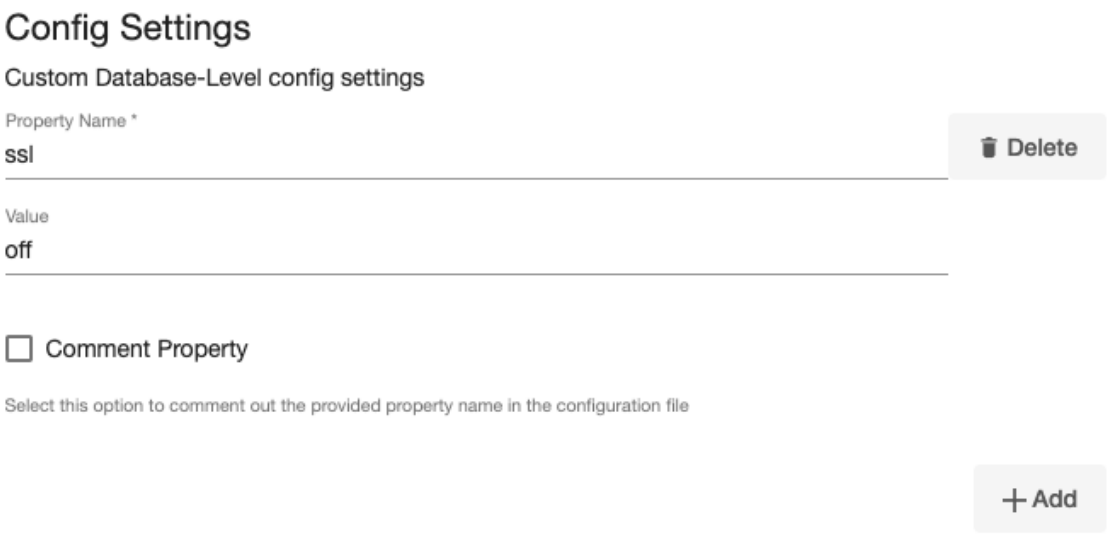
3. From the **Databases** tab, select a repository from which you want to create your dSource and click the icon.
4. In the **Add Database** dialog window, enter the **Name** for your source config and click **Add**.
5. Scroll down to the source config that you added and click the **Add dSource** option located to the right.
6. In the **Source** tab, do the following:
    a. In the **NFS Mount Location** field, enter a mount location on the staging host.
    b. If you want to use Privilege Elevation, provide the **Privileged OS Account** username. Make sure you meet all the requirements described in the Prerequisites for privilege elevation using DLPX_DB_EXEC script section.
    Note:
    You will currently not be able to update the Privileged OS account after dSource creation. The only way to achieve the change is to recreate the dSource.
    If the Environment OS user will be used to create the dSource, make sure you have the following privileges:
        - access to the the postgres commands
        - read+write+execute permissions on the `unix_socket_directories` configured in `postgresql.conf` [Default directories: `/var/run/postgresql` & `/tmp`]
    c. Select the checkbox next to the **Delphix Initiated Backup - Single Database Ingestion** option. Selecting this checkbox allows Delphix to initiate the logical backup on the source database.
    d. 

Click on the **+Add** icon located next to the **Single Database Ingestion method** option.
Note : The Add Button provided for Delphix Initiated Backup - Single Database Ingestion is clickable multiple times but should be used ONLY once since the solution is built only to support a single backup path. Providing Multiple Paths will error out the linking process.
The following fields are mandatory for this process.

i. **Source Postgres Database Name:** Provide the source Postgres database name that needs to be ingested.
ii. **Source Postgres Database User Name:** Provide the user name of the source Postgres database. The default value is postgres.
iii. **Source Postgres Database User Password:** Provide the password of the source Postgres database.
iv. **Source Postgres DB Host:** Provide the source database host address.
v. **Source Instance Port Number:** Enter the port number for Source Database. The default value is 5432.
vi. **Postgres Backup Dump Directory:** Provide the directory on the staging host on which backup will be created. This path should exist in the host environment with read and write access to the OS user.
vii. **Number of Backup Jobs:** Provide the number of parallel jobs to be executed to take backup. The default value is 2.
viii. **Number of Restore Jobs:** Provide the number of parallel jobs to be executed to restore backup. The default value is 2.
ix. **(Optional) Staging postgresql.conf config file:** Provide the path to a template Postgres config file that will be used when starting the dSource.



e. **Staging Instance Port Number** - must be unique for a given instance.
f. Optionally, database configuration parameters can be defined during the linking operation in the **Config Settings** section.



g. Click **Next**.

7. In the **dSource Configuration** tab, enter a dSource name and select a group for your dSource. Adding a dSource to a dataset group lets you set Delphix Domain user permissions for that database and its objects, such as snapshots. See the topics under Users and Groups for more information.
8. In the **Data Management** tab, specify your Staging Environment, User, and Snapshot Parameters. Select the **Resynchronize dSource** checkbox if you want to resynchronize the dSource. Resynchronizing the dSource will force a non-incremental load of data from the source. This operation is similar to creating a new dSource, but avoids duplicating storage requirements and maintains timeflow history. Click **Next**.
9. In the **Policies** tab, apply policy details to the dSource if required, and then click **Next**.
10. In the **Hooks** tab, select a Hook Point and then click **+** to add a script to run at that point. You can define scripts to run at multiple hook points in the process.
11. In the **Summary** tab, review the configuration profile for the dSource.
12. Click **Submit**.

The dSource Configuration Screen

After you have created a dSource, the dSource Configuration tab allows you to view information about it and make modifications to its policies and permissions. In the Datasets panel, select the dSource you wish to examine. You can now choose the configuration tab to see information such as the Source files, Data Management configuration, and Hook Operations.

Creating a dSource with Staging Push

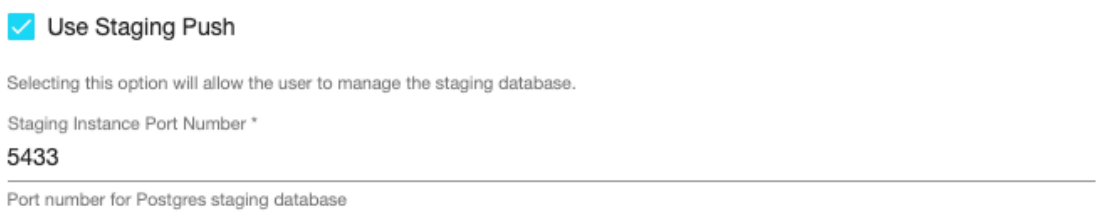
For staging push implementation details, see Staging Push Implementation.

Procedure

1. Login to the **Delphix Management** application.
2. Select **Manage > Environments**.
3. From the **Databases** tab, select a repository from which you want to create your dSource and click the  icon.
4. In the **Add Database** dialog window, enter the **Name** for your source config and click **Add**.
5. Select your source config and click the **Add dSource** option located to the right.
6. In the **Source** tab, do the following:
   a. In the **NFS Mount Location** field, enter a mount location on the staging host.

b. If you want to use Privilege Elevation, provide the **Privileged OS Account** username. Make sure you meet all the requirements described in the Prerequisites for Privilege Elevation using DLPX_DB_EXEC Scriptsection.
Note:
You will currently not be able to update the Privileged OS account after dSource creation. The only way to achieve the change is to recreate the dSource.
If the Environment OS user will be used to create the dSource, make sure you have the following privileges:
   - access to the postgres commands
   - read+write+execute permissions on the `unix_socket_directories` configured in `postgresql.conf` [Default directories: `/var/run/postgresql` & `/tmp`]

c. Select the checkbox next to the **Use Staging Push** option. Selecting this checkbox allows you to use the Staging Push Flow.
d. Provide **Staging Instance Port Number**. It must be unique for a given instance. Make sure the port number you have provided on the UI must match the port number in the postgresql.conf file.
Note:
The staging push flow database configuration parameters defined in the Config Settings section will be ignored by the Plugin. You will have to take care of these configurations on your own from the backend.



e. Click **Next**.

7. In the **dSource Configuration** tab, enter a dSource name and select a group for your dSource.
Adding a dSource to a dataset group lets you set Delphix Domain user permissions for that database and its objects, such as snapshots. See the topics under Users and Groups for more information.
8. In the **Data Management** tab, specify your Staging Environment, User, and Snapshot Parameters. Select the **Resynchronize dSource** checkbox if you want to resynchronize the dSource. Resynchronizing the dSource will force a non-incremental load of data from the source. This operation is similar to creating a new dSource, but avoids duplicating storage requirements and maintains timeflow history. Click **Next**.
9. In the **Policies** tab, apply policy details to the dSource if required, and then click **Next**.
10. In the **Hooks** tab, select a Hook Point and then click **+** to add a script to run at that point. You can define scripts to run at multiple hook points in the process.
11. In the **Summary** tab, review the configuration profile for the dSource.
12. Click **Submit**.

The dSource configuration screen

After you have created a dSource, the dSource Configuration tab allows you to view information about it and make modifications to its policies and permissions. In the Datasets panel, select the dSource you wish to examine. You can now choose the configuration tab to see information such as the Source files, Data Management configuration, and Hook Operations.

Staging push implementation

The Staging Push feature enables you to push data into the Delphix-provided mount point on your own. It is an alternative Delphix dSource data ingestion approach that enables you to push the data in Delphix, which is different from the conventional pull model where Delphix pulls the data.

There are two mechanisms to create dSource on the staging server.

- Pull architecture: The Postgres plugin pulls the data into the Delphix-provided mount location.
- Staging Push architecture: You push the data into the Delphix-provided mount point on your own.

Staging push implementation details

Staging Push gives you control over some staging database processes. It will give you control of the staging database to ingest data at either the cluster or individual database level. Staging database files will be stored on Delphix storage.

The following ingestion mechanisms are verified:

- Postgres Streaming Replication
- Backups + Logs w/ pg_basebackup
- Backups + Logs w/ pgbackrest
- Logical Replication and
- pg_dump and pg_restore method

> The first snapshot created as a part of dSource creation contains a data directory within the Delphix created mount point. This directory will be empty for the first snapshot.

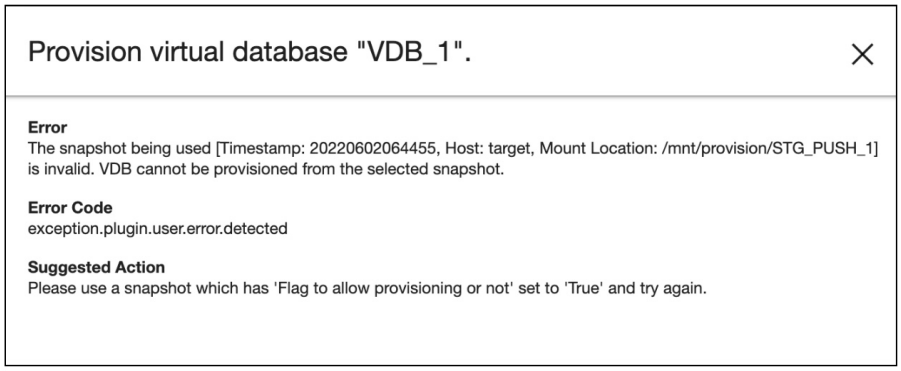
The first snapshot of the dSource will be empty. In this case, dSource snapshot metadata will contain the information whether the snapshot taken is provisionable or not. If you try to create a VDB from the first snapshot, then the Plugin will error out the operation with a proper error message.
The below steps show how to create a dSource using the Staging Push mechanism.

dSource Creation

The first snapshot created as a part of dSource creation contains a data directory within the Delphix created mount point. This directory will be empty for the first snapshot. The following snippet shows how the file system appears inside the Delphix-created mount point.

```
[postgres@target Staging_Push]$ tree
.
└── data

1 directory, 0 files
[postgres@target Staging_Push]$ pwd
/mnt/provision/Staging_Push
```

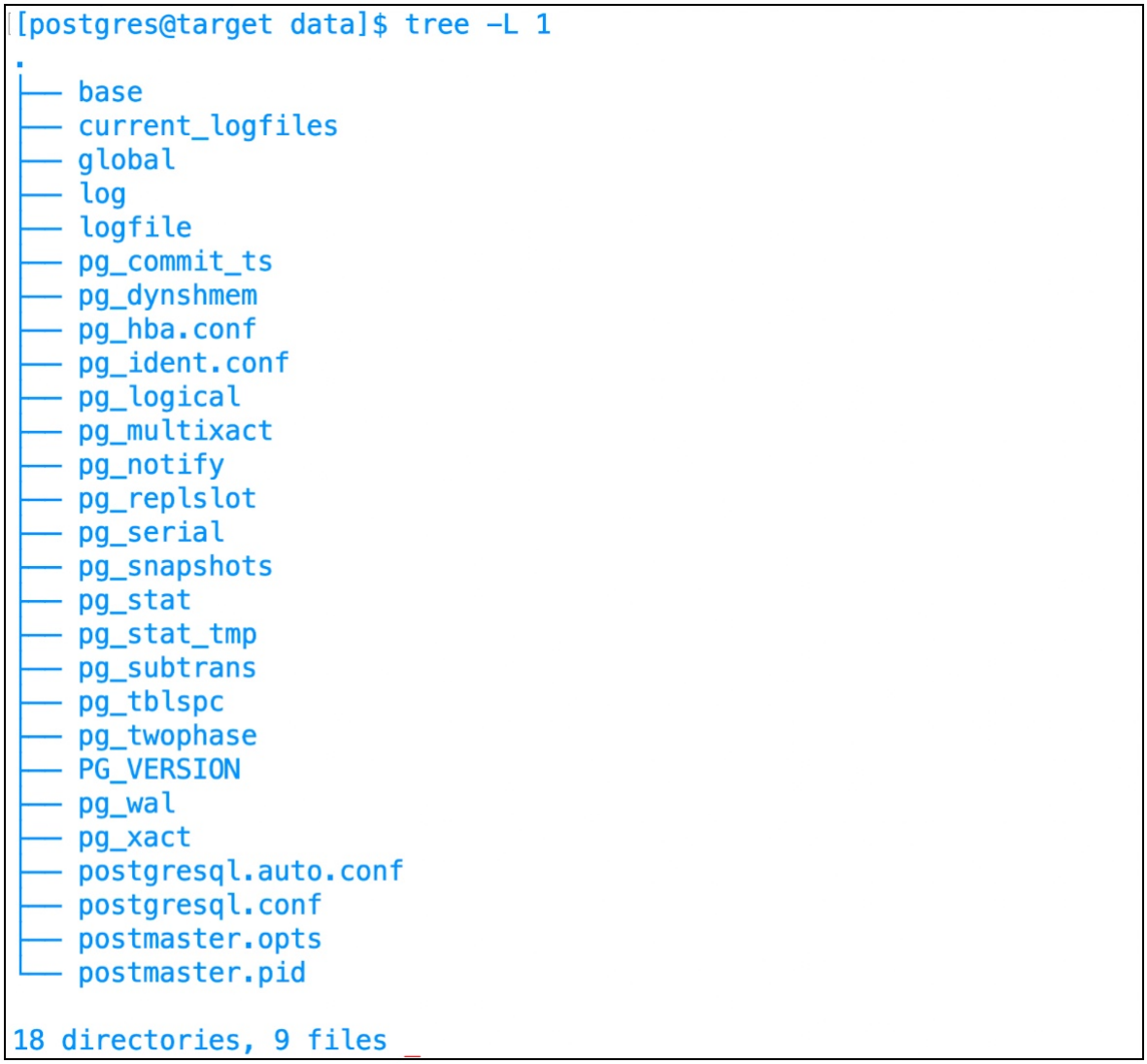
If you attempt to create a VDB from this snapshot, it will fail with the below error message.



Subsequent Snapshots

The system user needs to follow the steps and commands below before taking any subsequent snapshots.

1. You must recover the physical backup (pg_basebackup or pgBackRest) in the data directory or if you are recovering the backups created through pg_dump then you first need to initialize the Postgres Cluster on the data directory and then you can recover the backup through pg_restore.
2. While recovering the physical backup, you must make sure that all the tablespaces (if exists) should be present under the mount point only.
   The following screenshot represents what the file system will look like once the recovery is complete.



3. You can now proceed to take a snapshot and the VDB creation using the new snapshot.
   It's a pre-requisite for the end-user to stop the cluster before taking a snapshot and start the cluster after successful snapshot. If the cluster is not stopped, then the snapshot operation will fail with tan error message.

Disable/Delete Operation

For dSource disable and delete operation, it is a pre-requisite for the end-user to stop the cluster before kicking these operations from the UI. If it's not stopped, then the operations will fail with an error message.

Resync Operation

If the data directory under the mount path contains any file, then the resync operation will fail with an error message. So, for the resync operation, it is a pre-requisite that the data directory should be empty.

Working with PostgreSQL snapshots

Taking a snapshot creates a new snapshot entry in the dSource's Timeflow. The plugin-managed dSource allows you to select the parameters before taking the snapshot. These parameters are provided as an input to pre-snapshot and post snapshot functions. On the other hand, in the default snapshot, the parameters are pre-defined in the dSource environment.

Snapshot (Default)

This section lists the steps to take a snapshot and delete the same.

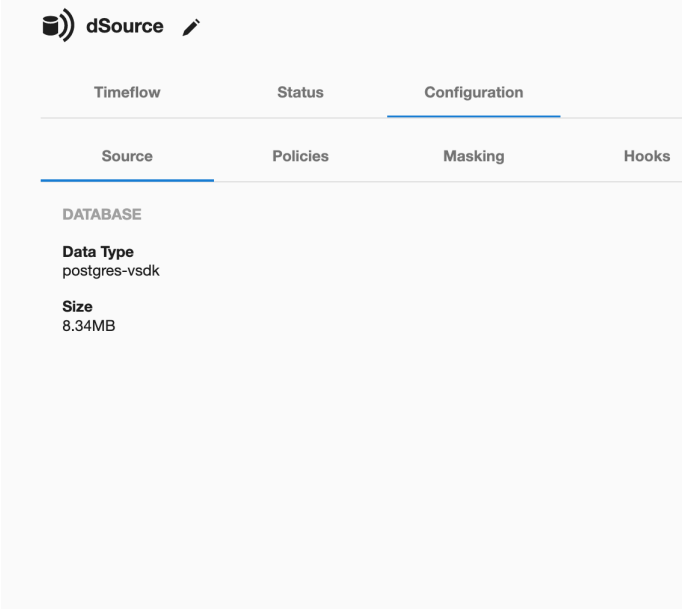
1. Login to the **Delphix Management** application.
2. Click **Manage** and select **Datasets** from the dropdown list.
3. Select the dSource you want to Snapshot.
4. Click the **Camera** icon. Alternatively, click the arrow next to the Camera icon and select **Snapshot (default)**.
5. From the Snapshot dialog, select **Perform Snapshot**.
6. Click **View:** under **Timeflow** to verify the Snapshot you just created.
7. To delete, select the snapshot you just created, and from the Actions menu (...) select **Delete Snapshot**.
8. In the Delete Snapshot dialog, select **Delete**.
9. Click **View:** under **Timeflow** to verify the Snapshot, you just deleted.

Source sizing implementation for dSource

The Source Sizing feature aligns the data source experience with Delphix database standards that will improve your ingestion reporting experience on the per TB pricing model. This feature enables you to calculate the database size based on whether they are ingested at the PostgreSQL cluster level or database level

Procedure

For all datasets (dSources), the dataset size can be calculated by executing `du -sb <dataset mount path>` command, which provides the correct volume and apparent size of the datasets.



Provisioning and managing VDBs from PostgreSQL dSources

Virtual databases are a key data management concept for Delphix. In order to create or provision a virtual database, you will need a linked dSource from a source host and a compatible target environment.

From a dSource, you can select a snapshot or point in time to create a VDB. PostgreSQL VDBs each have their own configuration settings as described in the following topics:

- Provisioning PostgreSQL VDBs: Overview
- Upgrading PostgreSQL VDBs
- Provisioning a PostgreSQL VDB
- Provisioning a PostgreSQL VDB PiT with external logs
- Rewinding a PostgreSQL VDB
- Database permissions for provisioned PostgreSQL VDBs

Provisioning PostgreSQL VDBs: Overview

This topic describes the basic concepts involved with provisioning VDBs from PostgreSQL.

Virtual databases are a key data management concept for Delphix. In order to create or provision a virtual database, you will need a linked dSource from a source host and a compatible target environment, as described in the overview for Managing environments and hosts and Overview of requirements for PostgreSQL

From a dSource, you can select a snapshot or point in time to create a VDB. PostgreSQL VDBs each have their own configuration settings as described in Configuration Settings for PostgreSQL Virtual Databases

For an overview of the high-level components involved in provisioning a PostgreSQL VDB refer to Setting up PostgreSQL environments: An overview

Once you have provisioned a VDB, you can also take snapshots of it. As with the dSource snapshots, you can find these when you select the VDB in the Datasets panel. You can then provision additional VDBs from these VDB snapshots.

Dependencies

If there are dependencies on the snapshot, you will not be able to delete the snapshot free space; the dependencies rely on the data associated with the snapshot.

Upgrading PostgreSQL VDBs

This topic describes how to upgrade a PostgreSQL VDB to a higher version of PostgreSQL.

> Currently, only minor upgrades are supported. For example, from PostgreSQL version 12.2 to 12.7.

Procedure for VDB In-Place Upgrade

1. Remove any VDB Refresh Policy assigned to the VDB.
2. Upgrade the target PostgreSQL instance.
3. Refresh the target environment.

Provisioning a PostgreSQL VDB

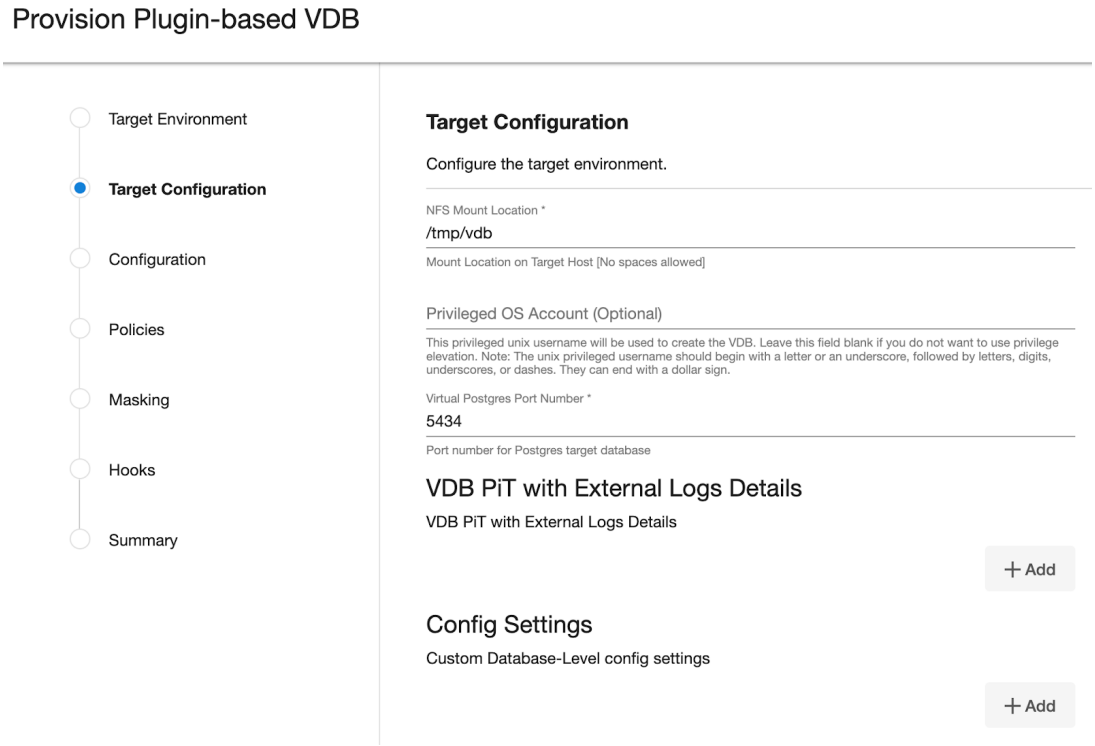
You can take a new snapshot of the dSource by clicking the Camera icon on the dSource card. Once the snapshot is complete you can provision a new VDB from it. This topic describes how to provision a virtual database (VDB) from a POSTGRES dSource.

Prerequisites

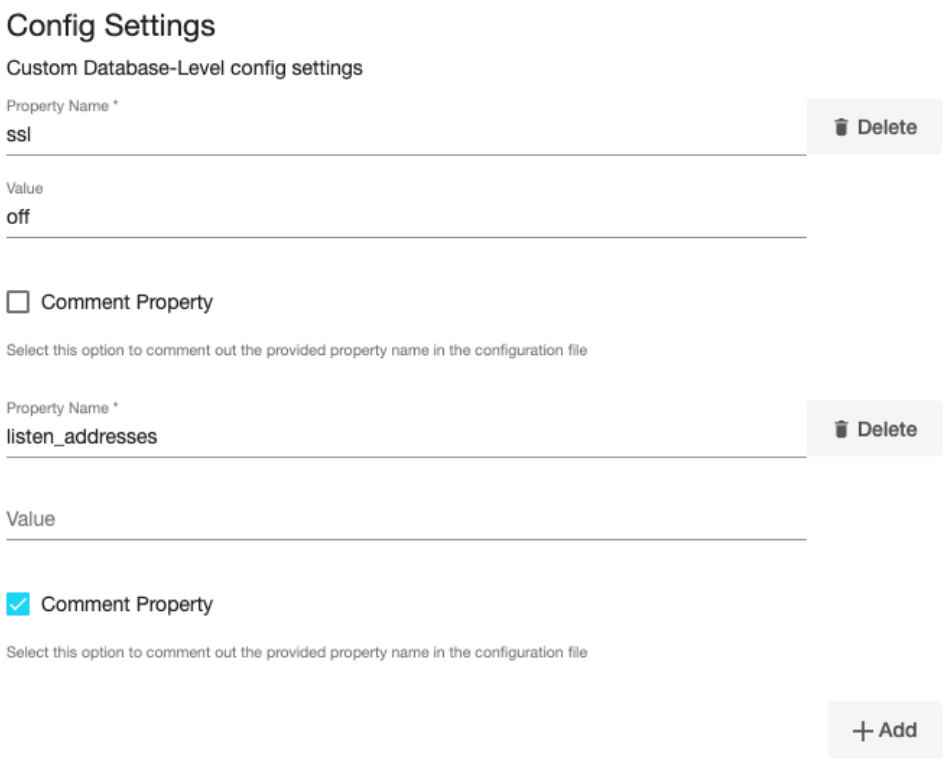
- You will need to have linked a dSource from a staging instance, as described in Linking a PostgreSQL dSource or have created a VDB from which you want to provision another VDB
- You should have set up the POSTGRES target environment with the necessary requirements as described in PostgreSQL support and requirements
- Make sure you have the required Instance Owner permissions on the target instance and environment
- The method for Database permissions for provisioned PostgreSQL VDBs is decided before the provisioning

Procedure

1. Navigate to **Manage**, and select **Datasets**.
2. Select a dSource and a snapshot from which you want to provision. Click the provision VDB icon to open the **provision VDB** wizard.
3. Select a target environment from the left pane, and an Installation to use from the dropdown list of available PostgreSQL instances on that environment.
4. Set the **Environment User** to be the Instance Owner.
   Note:
   The picking of instance owner is only possible if you have multiple environment users set on that host.
5. You will see the **Target Configuration** section where you need to specify **NFS Mount Location**.



6. If you want to use Privilege Elevation, provide the **Privileged OS Account** username. Make sure you meet all the requirements described in the Prerequisite for discovery operation section.
   Note:
   To update the Privileged OS Account after VDB creation, first disable the instance, update the Privileged OS Account, and then enable it.
   If the Environment OS user will be used to create the VDB, make sure you have the following privileges:
   - access to the the postgres commands
   - read+write+execute permissions on the `unix_socket_directories` configured in `postgresql.conf` [Default directories: `/var/run/postgresql` & `/tmp`]
7. Optionally, you can set the database configuration parameters for the VDB using **Config Settings**. Users can disable a configuration parameter by selecting the check box Comment Property. This will comment out the parameter in the `postgresql.conf` configuration file. Click **Next**.



8. On the **Configuration** page enter the PostgreSQL VDB name which will be displayed on the UI.
9. Select a **Target Group** for the VDB and click the green **Plus** icon to add a new group, if necessary.
10. Select a **Snapshot Policy** for the VDB then click **Next**.
11. Specify any desired hook operations.
12. Review the Provisioning Configuration and Data Management information.
13. Click **Submit**.

Once the VDB provisioning has successfully completed, if the source and target instance ids are not the same, you may want to grant secadm and dbadm on the database to the target instance id. Please refer to Database permissions for provisioned postgreSQL VDBs for more information.

Provisioning a PostgreSQL VDB PiT with external logs

With the latest PostgreSQL plugin 4.0.0 and onwards, you can provision a VDB to a point in time by providing the WAL logs on the target host.

Point in Time (PiT) recovery in PostgreSQL - Overview

PostgreSQL allows you to recover a cluster to a point in time (PiT) from a base backup by replaying the archived WAL logs. The main requirements to achieve PiT Recovery are:

- A base backup, which is essentially the file system backup. In the Delphix environment, the snapshots are nothing but file system backups.
- The archived WAL logs.
- The recovery target timestamp.

Understanding the PiT Recovery target timestamp

- A recovery point from PostgreSQL's perspective in the transaction commit timestamp. The recovery is performed till the commit timestamp less than or equal to the provided target time.
- From PostgreSQL version 13 and onwards, the point-in-time recovery fails if there are no transactions (no activity) post the requested target time.
- The `pg_waldump` (`pg_xlogdump` for PostgreSQL 9.x versions) can be used to analyze the WAL Logs and identify commit timestamps.
- The default timezone of the PiT timestamp (unless mentioned in the timestamp string) will the considered as the `timezone` configured in `postgresql.conf` file.

For more information about

WAL Log archiving and Point in Time (PiT) Recovery, see PostgreSQL Documentation

Setup

Scope of VDB PiT with External Logs feature

Provisioning of VDB PiT with External Logs is **supported** from the snapshots of:

- A dSource created with external backups using WAL logs
- A dSource created with external backups using streaming replication
- A dSource created with Delphix Initiated backup using streaming replication

Provisioning of VDB PiT with External Logs is **not supported** from the snapshots of:

- A dSource created with Delphix Initiated Backup using Single Database Ingestion
- A dSource created with Staging Push
- Any VDB
- Any dSource and VDB created using previous plugin versions (< 4.0.0). After upgrading from the plugin versions 3.2.0 and below, perform a **SnapSync** operation

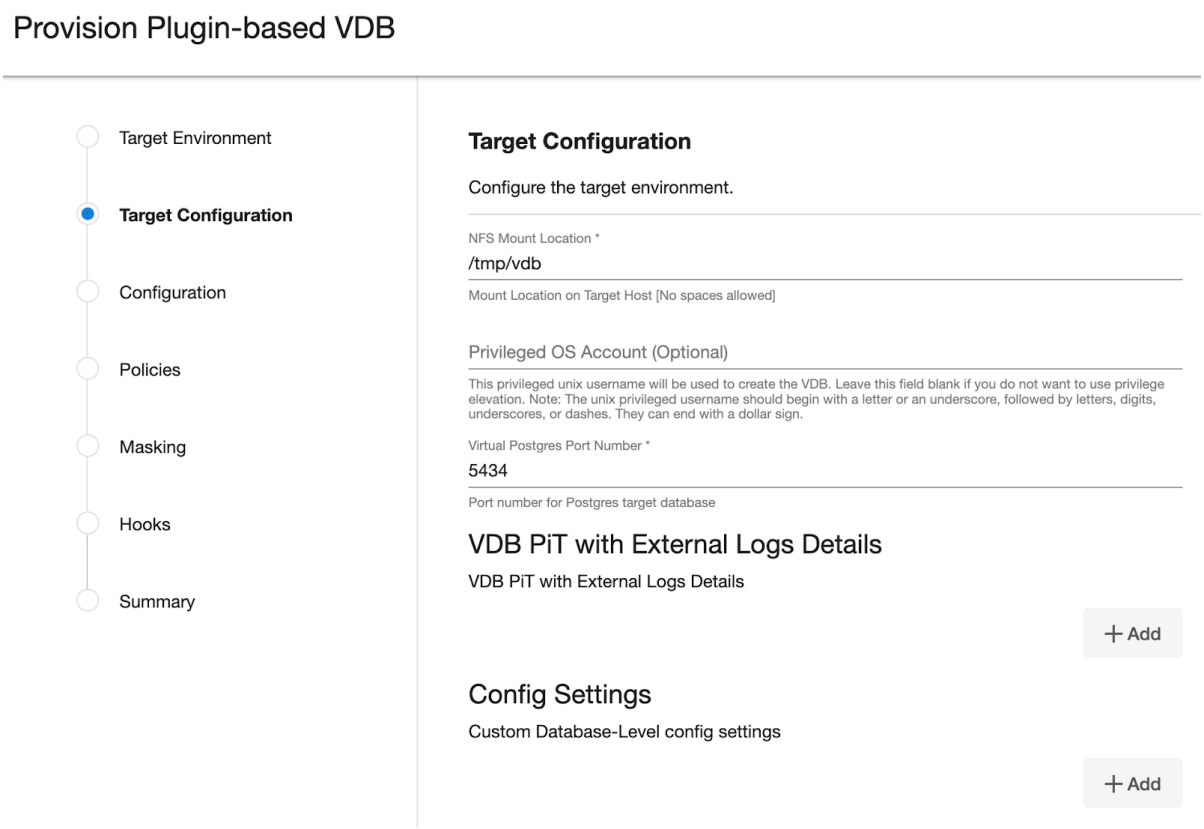
Prerequisites

- You will have to link a supported dSource (mentioned in the above section) from a staging instance, as described in Linking a PostgreSQL dSource
- You should set up the POSTGRES target environment with the necessary requirements as described in PostgreSQL support and requirements
- Make sure you have the required Instance Owner permissions on the target instance and environment.

- The target host must have the required WAL Logs (at least from the REDO WAL Log captured in the snapshot metadata) to perform point-in-time recovery, possible ways to achieve this are:
  - Archiving the WAL Logs on the source host and mounting (NFS) the archive directory on the target host. The below steps can be followed to enable WAL archiving:
    - Set the `wal_level` configuration parameter to `replica` or higher.
    - Set the `archive_mode` to `on`.
    - Use the `archive_command` parameter to copy the WAL Logs to an archive directory, ex: `cp %p <ARCHIVE_DIRECTORY>/%f`
  - The `pg_receivewal` utility can be used on the target host to start a background process that collects the WAL logs from the source database using streaming replication. Note, this will require increasing the `max_wal_senders` count in the source configuration (`postgresql.conf`) and will only collect the WAL Logs only from the time we start the process.
  - Copying the required WAL logs from the archive directory on the source host to the target host.
  - Using Delphix-provided solution Unstructured Files to mount the archive directory on the source host as a dSource and later provision the VFiles on the target host. With this approach, the archived WAL logs can be kept in sync using the dSource SnapSync and VFiles Refresh operations.
    Info:
    For more information, see Unstructured files and app data
- The method for Database Permissions for provisioned PostgreSQL VDBs is decided before provisioning.

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**, and select **Datasets**.
3. Select a **VDB PiT supported dSource** and a snapshot from which you want to provision. Click the provision VDB icon to open the **provision VDB** wizard.
4. Select a target environment from the left pane, and an Installation to use from the dropdown list of available PostgreSQL instances on that environment.
5. Set the **Environment User** to be the Instance Owner.
   Note: The picking of the instance owner is only possible if you have multiple environment users set on that host.
6. You will see the **Target Configuration** section where you need to specify **NFS Mount Location**.



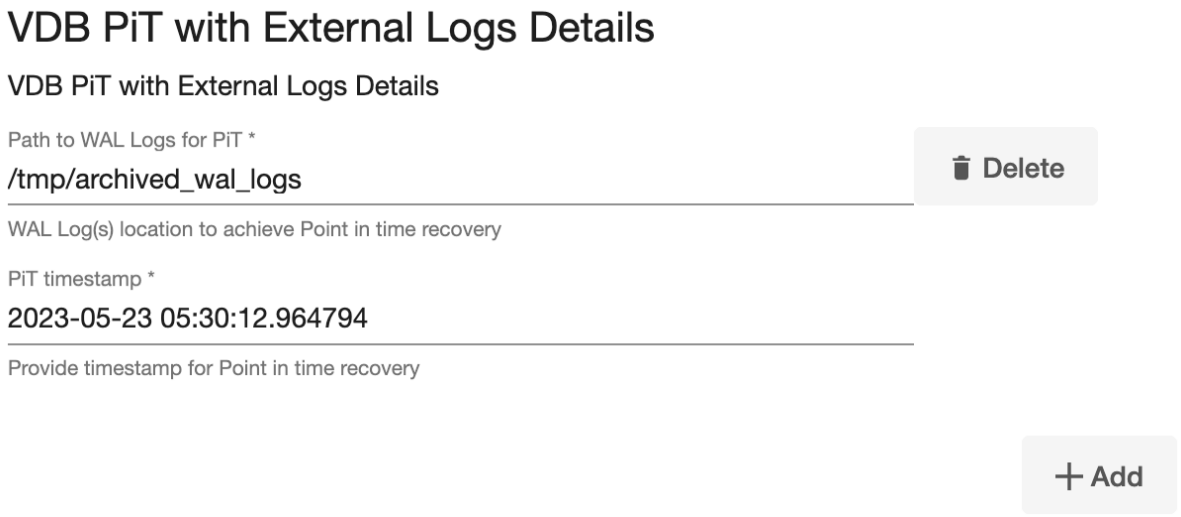
7. If you want to use Privilege Elevation, provide the **Privileged OS Account** username. Make sure you meet all the requirements described in the Prerequisites for Privilege Elevation using DLPX_DB_EXEC Script section.
   Note:
   - To update the Privileged OS Account after VDB creation, first disable the instance, update the Privileged OS Account, and then enable it.
   - If the Environment OS user will be used to create the VDB, make sure you have the following privileges:
     - access to the Postgres commands
     - read+write+execute permissions on the `unix_socket_directories` configured in `postgresql.conf` [Default directories: `/var/run/postgresql` & `/tmp`]
8. Provide a unique **Virtual Postgres Port Number** for a given instance.
9. Click on **Add (+)** to fill the **VDB PiT with External Logs Details**.
   Note: The Add Button provided for VDB PiT with External Logs Details is clickable multiple times but should be used ONLY once since the solution is built only to support a single entry. Providing multiple entries will error out the linking process.
   a. All fields within the **VDB PiT with External Logs Details** are mandatory.
   b. Provide the **absolute path** to the archived WAL logs available on the target host in the **Path to WAL Logs for PiT** field.
      Note:
      If privilege elevation is being used, please make sure the high-privileged user:
      - Is able to access the directory containing the WAL logs and has a minimum of read+execute permissions on the same.
      - Has read permission on individual WAL Logs.

      Warning:
      If the VDB PiT is utilizing the **same WAL Logs path** provided to the dSource created with External Backup using WAL logs (via the old plugin **versions 3.2.0 and below**), provisioning of the VDB PiT will

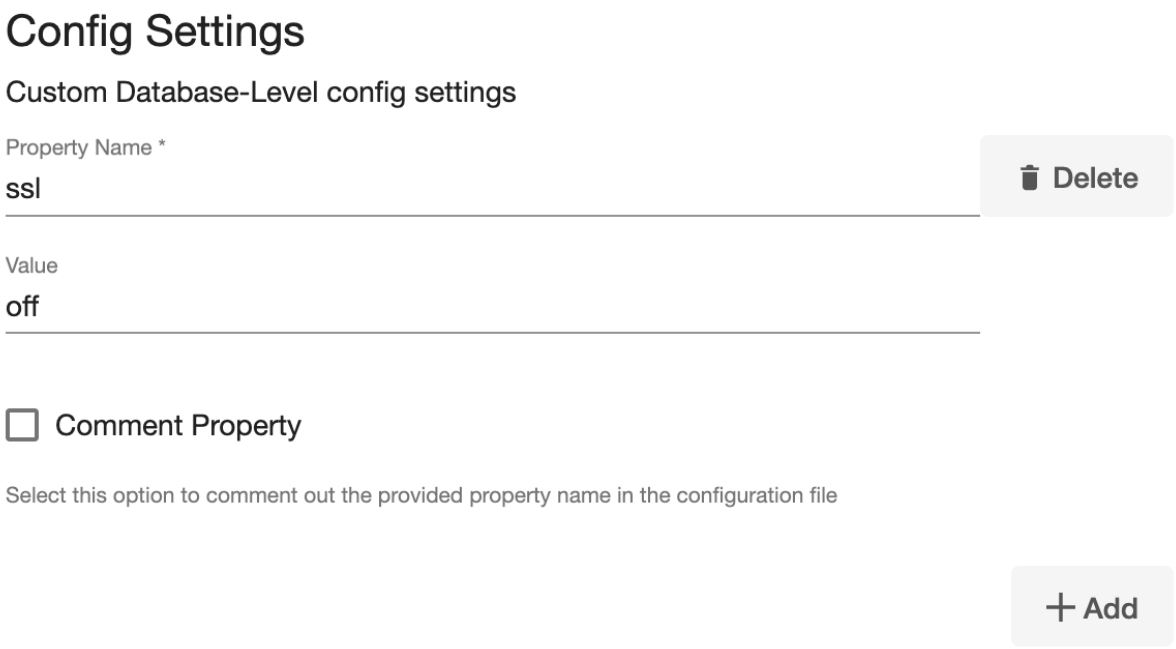
fail due to the error **cp: cannot stat**, which can be seen in the PostgreSQL logs. This is because the restore performed by the dSource renames the WAL Logs with a suffix **.done** (ex: 000000010000000000000020.done).
In order to proactively prevent the above error, please run the **Resynchronise dSource** operation on the dSource and then provision the VDB PiT.

c. Provide the target timestamp in the **PiT timestamp** field.
Note:
The plugin accepts timestamps with ISO 8601 basic format. The allowed timestamp formats can be expressed as follows:
    - Without the timezone offset from UTC:
        - YYYY-mm-dd HH:MM:SS
        - YYYY-mm-dd HH:MM:SS.ssssss
        - In this case, the timezone is assumed to be the same as the timezone parameter configured in `postgresql.conf` file.
    - With the timezone offset from UTC:
        - YYYY-mm-dd HH:MM:SS(+/-)HHMM
        - YYYY-mm-dd HH:MM:SS.ssssss(+/-)HHMM

    Here YYYY represents the year, mm represents the month, dd represents the day, HH represents the hour in 24-hour format, MM represents the minutes, SS represents the seconds and ssssss represents the microseconds.
    Also, make sure the PiT timestamp is greater than the checkpoint timestamp available in the snapshot metadata.



10. Optionally, you can set the database configuration parameters for the VDB using **Config Settings**. You can disable a configuration parameter by selecting the check box **Comment Property**. This will comment out the parameter in the `postgresql.conf` configuration file.



11. Click **Next**.
12. On the **Configuration** screen, enter the **PostgreSQL VDB** name which will be displayed on the UI.
13. Select a **Target Group** for the VDB and click add **(+),** to add a new group, if necessary.
14. Select a **Snapshot Policy** for the VDB then click **Next**.
15. Specify any desired hook operations.
16. Review the Provisioning Configuration and Data Management information.
17. Click **Submit**.

Once the VDB provisioning has been successfully completed, if the source and target instance ids are not the same, you may want to grant secadm and dbadm on the database to the target instance id. See Database permissions for provisioned PostgreSQL VDBs for more information.

There are two snapshot metadata that can be used to identify the results of the VDB PiT restore operation:

- **User-Provided Recovery Timestamp** records the timestamp provided by the end-user.
- **PiT recovery Restore Timestamp** records the last committed transaction timestamp the recovery has happened till.

Rewinding a PostgreSQL VDB

Rewinding a VDB rolls it back to a previous point in its Timeflow and re-provisions the VDB. The VDB will no longer contain changes after the rewind point.

Although the VDB no longer contains changes after the rewind point, the rolled-over Snapshots and Timeflow still remain in Delphix and are accessible through the Command Line Interface (CLI). Refer to CLI Cookbook: Rolling forward a VDB for instructions on how to use these snapshots to refresh a VDB to one of its later states after it has been rewound.

Delphix clones a new Timeflow from the closest Snapshot older than or equal to the rewind point. This creates a dependency between the new Timeflow and the parent Snapshot and Timeflow. The parent Snapshot and Timeflow cannot be deleted because of this dependency. The VDB must first be refreshed before the parent Snapshot and Timeflow can be removed.

Prerequisites

To rewind a VDB, you must have the following permissions:

A user with Delphix Admin credentials can perform a VDB rewind on any VDB in the system.

Procedure

1. Login to the **Delphix Management** application.
2. Under **Datasets**, select the VDB you want to rewind.
3. On the **Timeflow** tab, select the rewind point as a snapshot or a point in time.
4. Click **Rewind**.
5. If you want to use login credentials on the target environment other than those associated with the environment user, click Provide Privileged Credentials.
6. Click **Yes** to confirm.

Using TimeFlow Bookmarks

You can use a Timeflow bookmark as the rewind point when using the CLI. Bookmarks can be useful to:

- Mark where to rewind to – for example, before starting a batch job on a VDB
- Provide a semantic point to revert back to in case the chosen rewind point turns out to be incorrect.

For a CLI example using a Timeflow bookmark, refer to CLI Cookbook: Provisioning a VDB from a timeflow bookmark.

Database permissions for provisioned PostgreSQL VDBs

This topic describes the database permissions on provisioned POSTGRES virtual instances

PostgreSQL authentication

Authentication is the process by which the database server establishes the identity of the client, and by extension determines whether the client application (or the user who runs the client application) is permitted to connect with the database username that was requested.

PostgreSQL offers a number of different client authentication methods. The method used to authenticate a particular client connection can be selected on the basis of (client) host address, database, and user.

PostgreSQL database user names are logically separate from user names of the operating system in which the server runs. If all the users of a particular server also have accounts on the server's machine, it makes sense to assign database user names that match their operating system usernames. However, a server that accepts remote connections might have many database users who have no local operating system account, and in such cases, there need be no connection between database user names and OS user names.

Delphix postgreSQL authentication

Delphix for PostgreSQL requires that the staging and target hosts must already have the necessary users and authentication systems created/installed on them. Delphix will neither create users nor change database passwords as part of the provisioning process. It is required to have ‘trust’ as a PostgreSQL authentication mechanism on the source with a target server.

The following section describes some important authentication methods used by PostgreSQL architecture.

Trust authentication

When trust authentication is specified, PostgreSQL assumes that anyone who can connect to the server is authorized to access the database with whatever database user name they specify (even superuser names). Of course, restrictions made in the database and user columns still apply. This method should only be used when there is adequate operating-system-level protection on connections to the server.

Password authentication

The password-based authentication methods are md5 and password. These methods operate similarly except for the way that the password is sent across the connection, namely MD5-hashed and clear-text respectively.

PostgreSQL database passwords are separate from operating system user passwords. The password for each database user is stored in the pg_authid system catalog. Passwords can be managed with the SQL commands CREATE USER and ALTER ROLE, e.g., CREATE USER foo WITH PASSWORD 'secret'. If no password has been set up for a user, the stored password is null and password authentication will always fail for that user.

There are some more authentication mechanism which is less often used, refer to the Postgresql link embedded for more insights on them.

- GSSAPI Authentication
- SSPI Authentication
- Ident Authentication
- Peer Authentication
- LDAP Authentication
- RADIUS Authentication
- Certificate Authentication
- PAM Authentication

- BSD Authentication

PostgreSQL hook operations

Shell operations

RunCommand operation

The RunCommand operation runs a shell command on a Unix environment using whatever binary is available at `/bin/sh`. The environment user runs this shell command from their home directory. The Delphix Engine captures and logs all output from this command. If the script fails, the output is displayed in the Delphix Management application and command-line interface (CLI) to aid in debugging.

If successful, the shell command must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Examples of RunCommand operations

You can input the full command contents into the RunCommand operation.

```
remove_dir="$DIRECTORY_TO_REMOVE_ENVIRONMENT_VARIABLE"

if test -d "$remove_dir"; then
    rm -rf "$remove_dir" || exit 1
fi

exit 0
```

If a script already exists on the remote environment and is executable by the environment user, the RunCommand operation can execute this script directly.

```
/opt/app/oracle/product/10.2.0.5/db_1/dbs/myscript.sh "$ARG_ENVIRONMENT_VARIABLE"
"second argument in double quotes" 'third argument in single quotes'
```

RunBash operation

The RunBash operation runs a Bash command on a Unix environment using a `bash` binary provided by the Delphix Engine. The environment user runs this Bash command from their home directory. The Delphix Engine captures and logs all output from this command. If the script fails, the output is displayed in the Delphix Management application and command-line interface (CLI) to aid in debugging.

If successful, the Bash command must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Example of RunBash operations

You can input the full command contents into the RunBash operation.

```
remove_dir="$DIRECTORY_TO_REMOVE_ENVIRONMENT_VARIABLE"
```

```
# Bashisms are safe here!
if [[ -d "$remove_dir" ]]; then
    rm -rf "$remove_dir" || exit 1
fi

exit 0
```

Shell operation tips

Using `nohup`

You can use the `nohup` command and process backgrounding from resource in order to "detach" a process from the Delphix Engine. However, if you use `nohup` and process backgrounding, you MUST redirect `stdout` and `stderr.`

Unless you explicitly tell the shell to redirect `stdout` and `stderr` in your command or script, the Delphix Engine will keep its connection to the remote environment open while the process is writing to either `stdout` or `stderr` . Redirection ensures that the Delphix Engine will see no more output and thus not block waiting for the process to finish.

For example, imagine having your `RunCommand` operation background a long-running Python process. Below are the bad and good ways to do this.

Bad Examples

- `nohup python file.py & # no redirection`
- `nohup python file.py 2>&1 & # stdout is not redirected`
- `nohup python file.py 1>/dev/null & # stderr is not redirected`
- `nohup python file.py 2>/dev/null & # stdout is not redirected`

Good Examples

- `nohup python file.py 1>/dev/null 2>&1 & # both stdout and stderr redirected, Delphix Engine will not block`

Other operations

RunExpect operation

The RunExpect operation executes an Expect script on a Unix environment. The Expect utility provides a scripting language that makes it easy to automate interactions with programs which normally can only be used interactively, such as `ssh` . The Delphix Engine includes a platform-independent implementation of a subset of the full Expect functionality.

The script is run on the remote environment as the environment user from their home directory. The Delphix Engine captures and logs all output of the script. If the operation fails, the output is displayed in the Delphix Management application and CLI to aid in debugging.

If successful, the script must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Example of a RunExpect Operation

Start an `ssh` session while interactively providing the user's password.

```
spawn ssh user@delphix.com
expect {
    -re {Password: } {
        send "${env(PASSWORD_ENVIRONMENT_VARIABLE)}\n"
    }
    timeout {
        puts "Timed out waiting for password prompt."
        exit 1
    }
}
exit 0
```

PostgreSQL environment Variables

Operations that run user-provided scripts have access to environment variables. For operations associated with specific dSources or virtual databases (VDBs), the Delphix Engine will always set certain environment variables so that the user-provided script can use them to access the dSource or VDB.

dSource environment variables

Environment Variable	Description
`DLPX_DATA_DIRECTORY`	The mount path provided by the user on the UI.
`USER`	The OS user used to link the dSource.

VDB environment variables

Environment Variable	Description
`DLPX_DATA_DIRECTORY`	The primary mount path provided by the user on the UI. The first one, in case multiple mount specifications are provided.
`USER`	The OS user used to provision the VDB.

Managing plugin configurations

The plugin config file ***dlpx_postgres_config.ini*** is available on the remote host and located at ***<toolkit path>/Delphix_COMMON_<long id>/plugin/postgres-vsdk_<long plugin id>/dlpx_postgres_config.ini***.

Managing log rotation and log level

Delphix PostgreSQL plugin provides a feature that helps users to manage the log rotation for the plugin logs found within the scratch path, located at *<toolkit path>/Delphix_COMMON_<long id>/plugin/postgres-vsdk_<long plugin id>/<os user>_logs/delphix_postgres_debug.log.*

Delphix users can configure the logging mechanism by modifying the below parameters defined in the *dlpx_postgres_config.ini* file:

Setting maximum size

Delphix users can set the parameter MAX_FILE_SIZE_KB available in the *dlpx_postgres_config.ini* file to set the maximum size of the active log file in KB. Once this limit is reached, the plugin will rotate the log. The default/maximum value of this parameter is 10240 KB/10 MB. This parameter accepts only a positive integer value, such as 10240.

Setting number of log files to retention

Delphix users can set a retention number for the rotated log files using the parameter NUM_OF_FILES_TO_KEEP. The default value of the parameter is 50. That means, the maximum number of rotated log files that will be retained is 50 and each file will be of a size as defined in the MAX_FILE_SIZE_KB parameter. If the number of files exceeds this value, then the old files will be deleted in a FIFO order.

Setting the log level

Delphix users can set the amount of log being generated on the remote host by defining the LOG_LEVEL parameter. Accepted values are **DEBUG, INFO, WARNING, and ERROR**, with **INFO** being the default value.

Managing PostgreSQL timeouts

PGCTLTIMEOUT

The PGCTLTIMEOUT parameter defines the default limit on the number of seconds to wait for the Postgres startup to complete. This parameter is introduced to compensate for any network latency between the Delphix Engine & the remote host. The default value is set to 600 seconds.

Managing plugin timeouts

PIT_VDB_RECOVERY_MAX_WAIT_TIME

Provisioning of VDB PiT with External Logs has a time-bounded wait for the recovery to complete. The PIT_VDB_RECOVERY_MAX_WAIT_TIME defines the maximum amount of time the plugin will wait for the operation to succeed. It is defined in seconds and the default value is 7200 seconds (i.e. 2 hours).

> The wait cycles within the plugin are of 4 seconds each. So keeping the value of the parameter in multiples of 4 will result in a better experience.

dlpx_postgres_config.ini

```
[log_config]
MAX_FILE_SIZE_KB = 10240
NUM_OF_FILES_TO_KEEP = 50
LOG_LEVEL = INFO # Accepted values: DEBUG, INFO, WARNING and ERROR

[database_config]
PGCTLTIMEOUT = 600
[plugin_config]
PIT_VDB_RECOVERY_MAX_WAIT_TIME = 7200
```

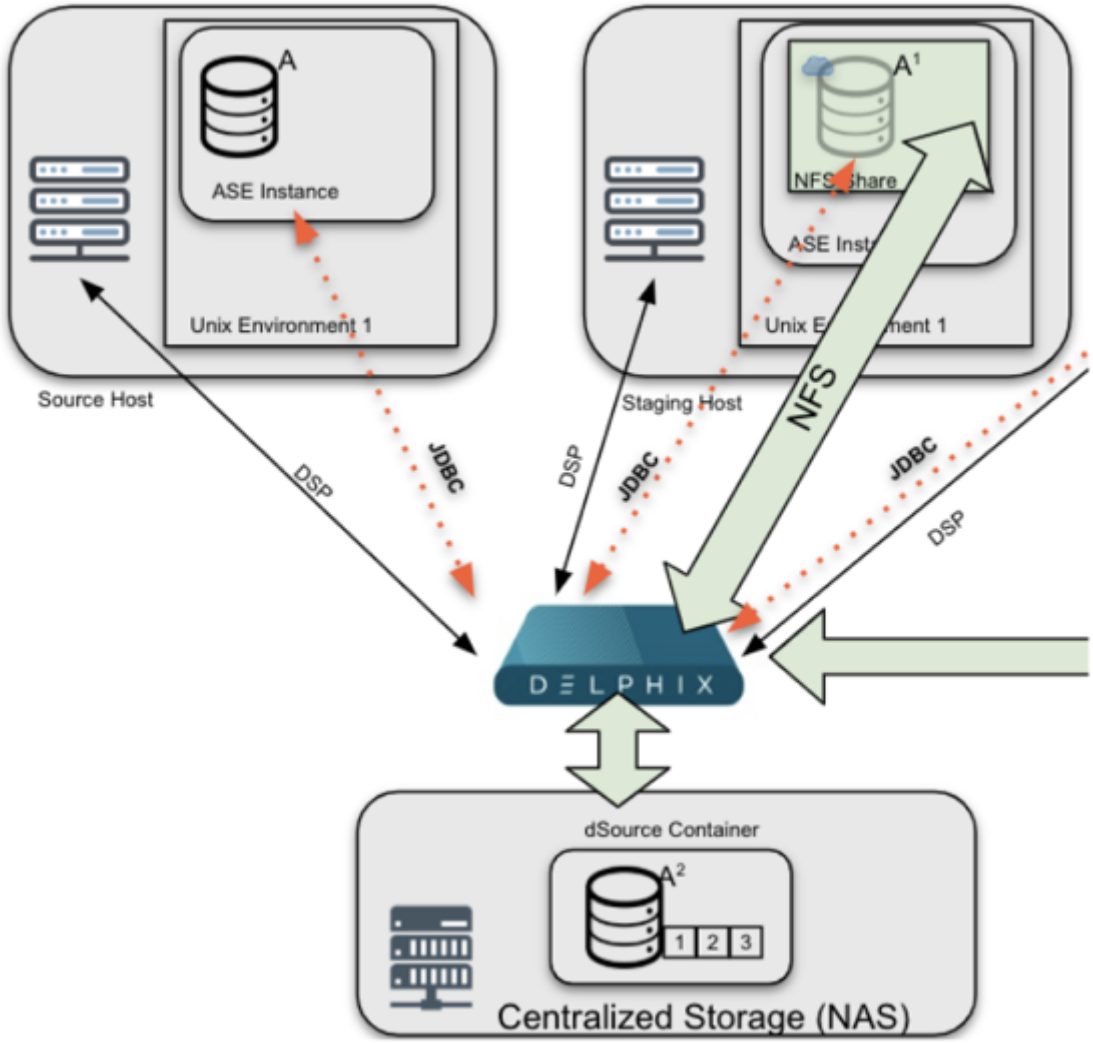
SAP ASE environments and data sources

This section covers the following topics:

Delphix architecture with SAP ASE

This topic describes the high-level process for adding SAP ASE-supported environments, linking SAP ASE databases to the Delphix Engine, and provisioning virtual databases.

Diagram of underlying Linking Architecture to support ingestion workflows between SAP ASE-supported Environments and the Delphix Engine.



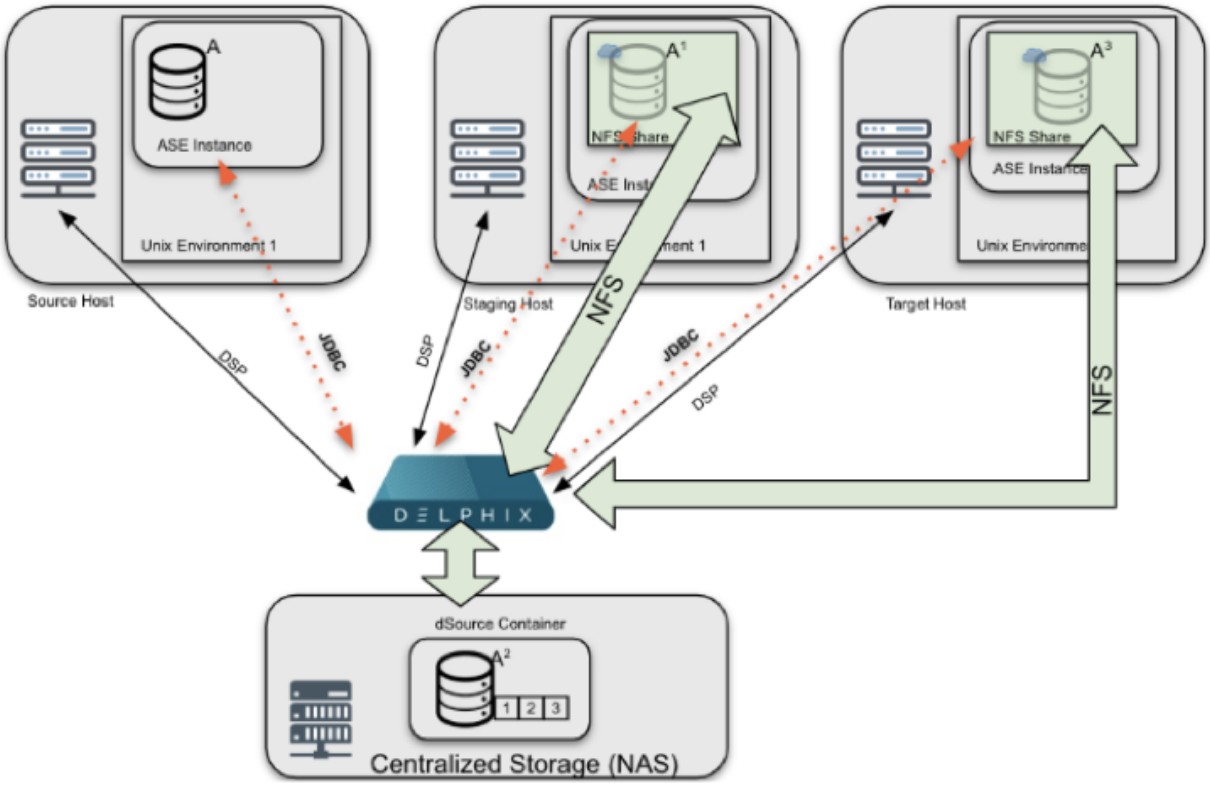
Linking architecture between SAP ASE and Delphix engine

The diagram featured above shows a common architecture for SAP ASE-based systems. The diagram shows the infrastructure used to ingest data. (From the above diagram) The workflow starts with a Source Host (Left, which in this example is a Production Database Server), that provides critical data in the form of dump files (used for SnapSync) and transaction logs (used for LogSync).

The Delphix Engine (Bottom), continuously monitors the source database to determine when new dumps are available. When a dump is available, the Delphix Engine will contact the SAP ASE Staging host (Right) via Delphix Session Protocol (DSP). The Staging host will read the dumps from the Source Host and recover them through a Staging Database that is automatically set up on Delphix NFS storage. Once recovery is complete, the backup data

is incorporated into the Delphix dSource as a new snapshot card and is available for use to provision a new virtual or physical database.

Diagram of underlying provisioning architecture to support VDB provisioning workflows between the Delphix engine and SAP ASE-supported target environments.



If no remote load location (aka Source Host) is configured, dump files must be made available on the staging host via NFS or some other copy mechanism. If no remote load location is configured, the Delphix Engine searches for the files on the staging server. If they are not found, the Delphix Engine cannot load them

SAP ASE provisioning architecture

This diagram is an extension of the previous diagram with the new content showing how Delphix provisions Virtual Databases (VDBs) to a Target Environment. The Delphix Engine (bottom) creates a set of virtual files from a snapshot that becomes the VDB. These files are presented to the Target host (right) via NFS. Delphix uses DSP to communicate with Target and initiates the creation of a new database. Once complete, the VDB is brought online and made available for use.

Environment setup

Target and staging environments for SAP ASE

For SAP ASE-based Delphix Architectures, the Staging Host plays a critical role:

1. Provide the “staging” point in which the Delphix Engine coordinates data ingestion to a dSource by restoring backups to a staging database and creating a dSource from that ingested data.

2. Tracks changes on disk by running validated sync, a process that identifies when new backups are available and initiates the ingestion process if new backups and/or transaction log backups are found.
3. Minimizes touch to production by providing an intermediate host between source databases and Delphix.

For SAP ASE-based Delphix architectures, the target host has two potential roles:

1. Host a target environment for provisioning virtual databases (VDBs).

> Staging and Target hosts can be run on a target server, although a dedicated staging server is recommended for optimal performance.

SAP ASE dSources are representations of a Source Database replica on the staging database that runs on a target or staging host. There is no requirement for additional local storage with either host option, as the storage is mounted over NFS from the Delphix Engine. For a deeper, technical discussion, please see the Technical Deep Dive section below.

At Delphix, we refer to the creation and maintenance of this staging database on the staging host as "validated sync," because it prepares the dSource data on the Delphix Engine for provisioning VDBs later on. After the Delphix Engine creates the staging database, it continuously monitors the source database for new transaction log/Full backups (if “truncate log on checkpoint” is active, Delphix will need Full Backups for Validated Sync). When it detects a new transaction log backup, it restores that backup to the staging database. The result is a TimeFlow with consistent points from which you can provision a VDB, and a faster provisioning process, because there is no need for any database recovery during provisioning. If Log Sync is enabled, you may provision a VDB to a point in time, in between snapshots.

When you later provision a VDB, you can specify any environment as a target, including the environment that contains the staging database. However, for best performance, we recommend that you choose a different target environment. The only requirement for the target is:

- It must have an operating system that is compatible with the one running on the validated host.

Target hosts for ASE

Container for VDBs

This topic describes the basic concepts involved with provisioning VDBs from SAP ASE dSources or even other SAP ASE VDBs.

A dSource is a virtualized representation of a physical or logical source database. As a virtual representation, it cannot be accessed or manipulated using database tools. Instead, you must create a virtual database (VDB) from a dSource snapshot. A VDB is an independent, writable copy of a dSource snapshot. You can also create VDBs from other VDBs. Once you have provisioned a VDB to a target environment, you can also implement snapshot and retention policies for the VDB, which will determine how frequently Delphix Engine will take a database snapshot and how long the snapshots will be retained for recovery and provisioning purposes.

When provisioning a VDB, Delphix creates the database with the default SAP ASE database options. If the database options have been altered on the source database and you wish for the VDB to reflect these same options, they would need to be altered via a Post-Script or by Hook Scripts for Automation and Customization

For an overview of the high-level components involved in provisioning an SAP ASE VDB, see Overview of Provisioning SAP ASE Virtual Databases

Validated sync and logSync

To run validated sync, a staging environment must be specified to host a staging database for the validated sync process. In this process, the Delphix Engine continuously monitors the source database for new full and transaction

log backups if the source database is using a simple recovery model, and only transaction log backups if using a full recovery model. When it detects a new backup, it restores that backup to the staging database with the storage residing in Delphix. The result is a Timeflow with consistent points from which you can provision a VDB, also known as snapshots.

If the “truncate log on checkpoint” is set, Delphix will only apply full backups.

Snapshots accumulate over time. To view a snapshot:

1. From the Datasets panel, click the group containing the dSource.
2. Select dSource.
3. Click the Timeflow tab.

Each snapshot is displayed and includes information about the source database, operating system, and time stamp. You can scroll through these cards to select the one you want, or you can enter a date and time to search for a specific snapshot.

Once you have provisioned a VDB, you can also take snapshots of it. As with the dSource snapshots, you can find these when you select the VDB in the Datasets panel. You can then provision additional VDBs from these VDB snapshots.

Dependencies

If there are dependencies on the snapshot, you will not be able to delete the snapshot free space; the dependencies rely on the data associated with the snapshot.

Overview of ASE database encryption

Beginning with 16.0, SAP ASE supports the Database Encryption feature. The SAP ASE Database Encryption feature encrypts the data at rest, without changing the applications. This encryption can be done on entire databases or only on columns. This ensures that the authorized users access the data and thus prevents the misuse of the data against theft and security breaches.

Data is encrypted with the help of encryption keys. These encryption keys are stored in the database in an encrypted form. You can encrypt an encryption key using a key encryption key (KEK).

In the SAP ASE Database Encryption, column and database encryption uses a symmetric encryption algorithm, which means that the same key is used for encryption and decryption. SAP ASE tracks the key that encrypts the data.

Starting 6.0.8.0, Delphix Engine will support the SAP ASE encrypted databases.

For more information on SAP ASE Database Encryption, see the SAP ASE Encryption Documentation

Delphix implementation of database encryption

This topic describes various configurations to support encrypted databases with Delphix. Follow the mandatory steps below on the ASE instance that hosts the staging databases and virtual databases.

1. If the source database is not encrypted already.
    a. Install the license option ASE_ENCRYPTION.
    b. Create a master key that will serve as the KEK.
    Command

    ```
    > create encryption key master with passwd "sybase"
    ```

    c. If the database is not encrypted already, create the database encryption key and use it.
    Commands

    ```
    > create encryption key <encryption-key-name> for database encryption
    > sp_configure "number of worker processes", 2
    > alter database <database-name> encrypt with <encryption-key-name>
    ```

    d. Export the master key and the encryption key to a location that is shared among source, staging, and target hosts. The command-line version of the ddlgen tool is located at *$SYBASE/$SYBASE_ASE/bin*. You need to find out this location for your instance if it is different.
        i. cd $SYBASE/$SYBASE_ASE/bin
        ii. ddlgen -Usa -Psybase -SASE160_SRC -TEK -N master.dbo.master -XOD -O<shared-path>/master_ddl.sql
        iii. ddlgen -Usa -Psybase -SASE160_SRC -TEK -N master.dbo.<key_name> -XOD -O<shared-path>/<key_name>_ddl.sql
    e. Enable encryption in SAP ASE by executing the below command on the staging/target instance.
    Command

    ```
    > sp_configure 'enable encrypted columns', 1
    ```

2. Import the keys on the staging and target instances by running the below commands from the directory where the SQL files are present (the shared location between instances) or mention the entire path of the files to be imported.
    a. isql -Usa -Psybase -SASE160_TGT -w 220 -i master_key.sql
    b. isql -Usa -Psybase -SASE160_TGT -w 220 -i <key_name>.sql
3. Set the encryption password by executing the below command on the staging/target instance.
Command

```
> set encryption passwd "sybase" for key master
```

4. Setup for automatic master key access. Refer create the master key start-up file. In order to avoid issues on the master key password after the reboot of the ASE instance, a master key startup file needs to be created by running the following steps on the staging and the target instance.
    a. Command

```
> sp_configure 'automatic master key access',1
```

    b. Command

```
> alter encryption key master with passwd 'sybase' add encryption for
 automatic_startup
```

    c. Command

```
> sp_encryption mkey_startup_file,default_location,sync_with_mem
```

    d. Verify if the master key startup file has been successfully created on the instance.
    Command

```
> sp_encryption mkey_startup_file
```

    e. Reboot the ASE instance to get the master key startup file in effect.

If you perform a reboot or plan to perform a reboot on the source host, then you would need to repeat step 4 on the source host as well. By doing so, you don't need to set the master key password again after reboot.

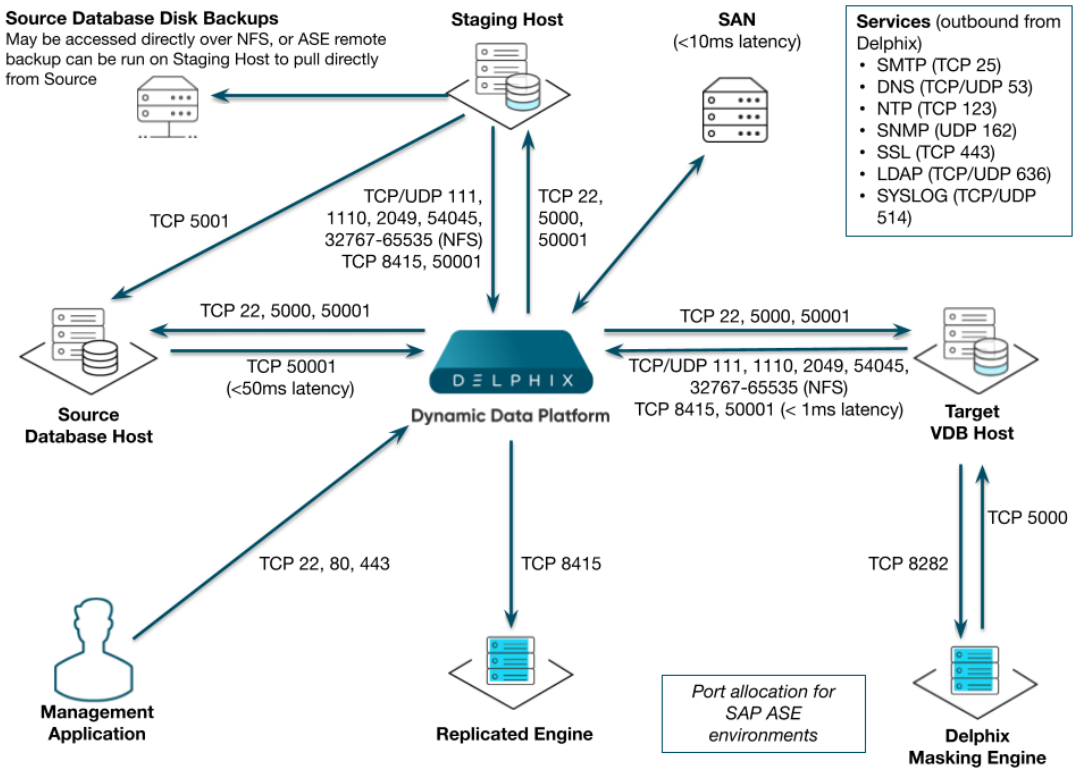
Quick start guide for SAP ASE

This quick start guide, which is excerpted from the larger User Guide, is intended to provide you with a focused overview of working with SAP ASE database objects in the Delphix Dynamic Data Platform. It does not cover advanced configuration options. It does not cover advanced configuration options or best practices for performance. It assumes that you are working in a Lab/Dev setting and attempting to quickly test Delphix Engine functionality. It assumes you will use the VMware Hypervisor.

Overview

In this guide, we will walk through deploying a Delphix Engine, starting with configuring SAP ASE Source and Target environments. We will then create a dSource, and provision a VDB. A detailed list can be referred to in the Network and Connectivity Requirements for SAP ASE Environments section of the manual.

For purposes of the QuickStart, you can ignore any references to Replication or Masking, such as the engines shown in the diagram below.

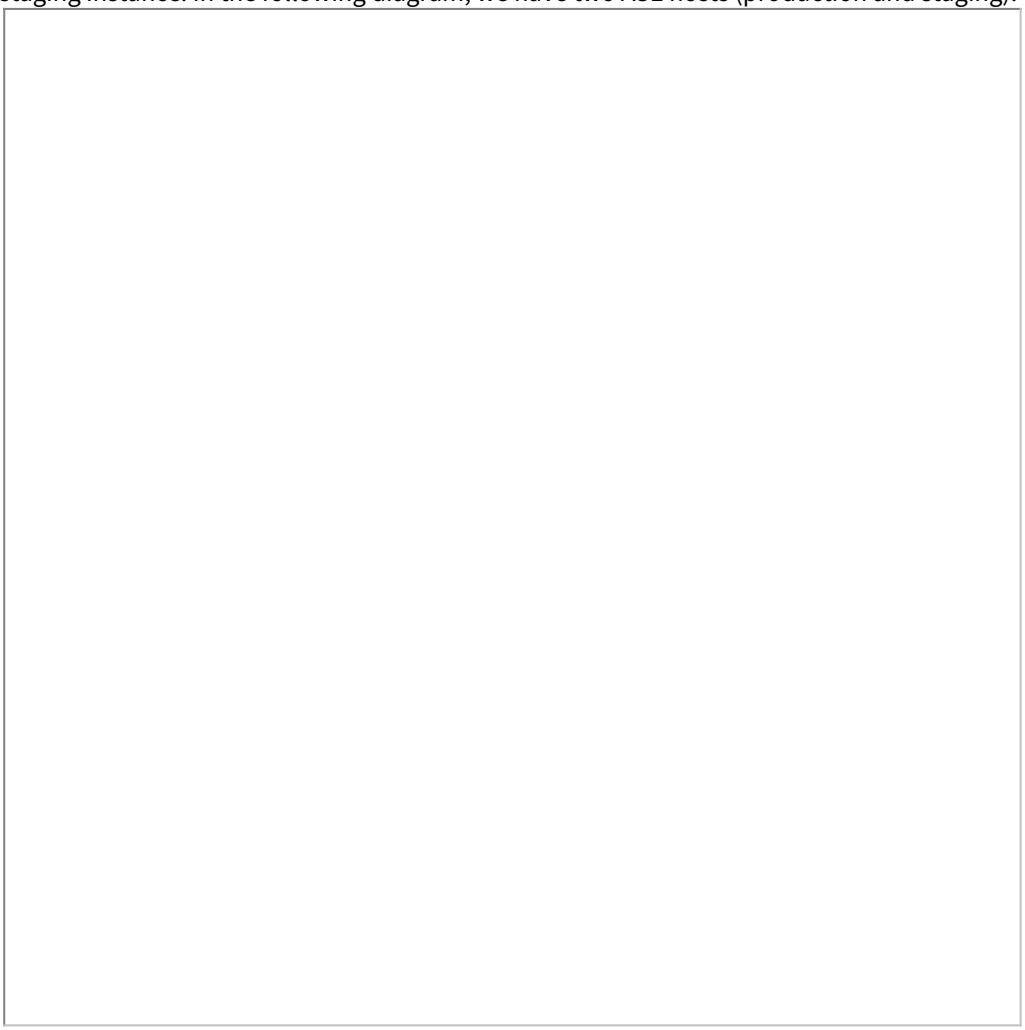


Delphix stays in sync with source databases by monitoring the ASE backup server log. When it sees a new database dump or transaction log has been created, it attempts to load it into the ASE staging instance.

Color	Supported?
Y	Yes
N	No

Validated sync Mode	SnapShot for FULL BACKUP	SnapShot for TLOGS	Point-in-Time Restore
truncate log on chkpt = true	Y	Y	Y
truncate log on chkpt = false + LogSynctruncate log on chkpt = false	N	Y	N
truncate log on chkpt = true + Log Sync	Y	N	N
truncate log on chkpt = false + LogSync	N	Y	Y

Before proceeding with setting up environments, decide where the databases will be located relative to the ASE staging instance. In the following diagram, we have two ASE hosts (production and staging).



If the database dumps (or transaction logs) are available locally to the staging Backup Server, Delphix considers them “local”. Whereas if they are available to a Backup Server on production, they are “remote”. If the files will be local to the staging server, there are many options to get them from the production host to the staging host including but not limited to:

- Sharing the directory on the production host over NFS to the staging host.
- Using Network Attached Storage (NAS) to replicate the files.
- Using scp to copy the files from one host to the other.

If the files are only available on the production host, Delphix will login to the ASE staging instance and issue the "LOAD DATABASE" command using the remote server syntax (for example LOAD DATABASE pubs2 FROM "/dumps/pubs2.full.9_5_16" AT Prod_BS).

Deploy OVA on VMware

Use the Delphix-supplied OVA file to install the Delphix Engine. The OVA file is configured with many of the minimum system requirements. The underlying storage for the install is assumed to be redundant SAN storage.

1. Download the OVA file from https://download.delphix.com. You will need a support login from your sales team or a welcome letter.
    a. Navigate to the Delphix Product Releases/<Current Version>/Appliance Images page.
    b.
2. Login using the vSphere client to the vSphere server (or vCenter Server) where you want to install the Delphix Engine.
3. In the vSphere Client, click **File**.
4. Select **Deploy OVA Template**.
5. Browse to the OVA file.
6. Click **Next**.
7. Select a **hostname** for the Delphix Engine. This hostname will also be used in configuring the Delphix Engine network.
8. Select the **data center** where the Delphix Engine will be located.
9. Select the **cluster** and the **ESX host**.
10. Select one (1) **data store** for the **Delphix OS**. This datastore can be **thin-provisioned** and must have enough free space to accommodate the 127GB comprising the Delphix operating system.
11. Select four (4) or more **data stores** for Database Storage for the Delphix Engine. The Delphix Engine will stripe all of the Database Storage across these VMDKs, so for optimal I/O performance, each VMDK must be equal in size and be configured **Thick Provisioned - Eager Zeroed**. Additionally, these VMDKs should be distributed as evenly as possible across all four SCSI I/O controllers.
12. Select the **virtual network** you want to use. If using multiple physical NICs for link aggregation, you must use vSphere NIC teaming. Do not add multiple virtual NICs to the Delphix Engine itself. The Delphix Engine should use a single virtual network. For more information, see Optimal Network Architecture for the Delphix Engine
13. Click **Finish**. The installation will begin and the Delphix Engine will be created in the location you specified.
14. Once the installation has completed, power on the Delphix Engine and proceed with the initial system configuration as described in Setting Up Network Access to the Delphix Engine

> ⓘ If your source database is 4 TB, you probably need 4 TB of storage for the Delphix Engine. Add at least 4 data disks of similar size for the Delphix VM. For example: for a source database of 4 TB, create 4 VMDKs of 1 TB each.

> ⓘ For a full list of requirements and best practice recommendations, see Virtual Machine Requirements for VMware Platform

Setup network access to Delphix engine

1. Power on the Delphix Engine and open the Console.

2. Wait for the Delphix Management Service and Delphix Boot Service to come online. This might take up to 10 minutes during the first boot. Wait for the large orange box to turn green.
3. Press any key to access the sysadmin console.
4. Enter `sysadmin@SYSTEM` for the username and `sysadmin` for the password.
5. You will be presented with a description of available network settings and instructions for editing.

```
Delphix Engine Network Setup To access the system setup through the browser,
the system must first be configured for networking in your environment. From
here, you can configure the primary interface, DNS, hostname, and default
 route. When DHCP is configured, all other properties are derived from DHCP
settings. To see the current settings, run "get." To change a property, run
"set =." To commit your changes, run "commit." To exit this setup and return to
the standard CLI, run "discard." defaultRoute    IP address of the gateway for
 the default route -- for example, "1.2.3.4." dhcp            Boolean value
indicating whether DHCP should be used for the primary interface. Setting this
 value to "true" will cause all other properties (address, hostname, and DNS)
to be derived from the DHCP response dnsDomain       DNS Domain -- for example,
"delphix.com" dnsServers      DNS server(s) as a list of IP addresses -- for
 example, "1.2.3.4,5.6.7.8." hostname        Canonical system hostname, used in
alert and other logs -- for example, "myserver" primaryAddress  Static address
for the primary interface in CIDR notation -- for example, "1.2.3.4/22" Current
settings: defaultRoute: 192.168.1.1 dhcp: false dnsDomain: example.com
dnsServers: 192.168.1.1 hostname: Delphix primaryAddress: 192.168.1.100/24
```

6. Configure the `hostname`. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set hostname=<hostname>
```

Note: Use the same `hostname` you entered during the server installation.

7. Configure DNS. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set dnsDomain=<domain> delphix network setup
update *> set dnsServers=<server1-ip>[,<server2-ip>,...]
```

8. Configure either a static or DHCP address.
   DHCP Configuration

```
delphix network setup update *> set dhcp=true
```

   Static Configuration

```
delphix network setup update *> set dhcp=false delphix network setup update *>
set primaryAddress=<address>/<prefix-len>
```

   Note: The static IP address must be specified in CIDR notation (for example, `192.168.1.2/24`)
9. Configure a default gateway. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set defaultRoute=<gateway-ip>
```

10. Commit your changes. Note that you can use the `get` command prior to committing to verify your desired configuration.

```
delphix network setup update *> commit Successfully committed network settings.
Further setup can be done through the browser at: http://<address> Type "exit"
to disconnect, or any other commands to continue using the CLI.
```

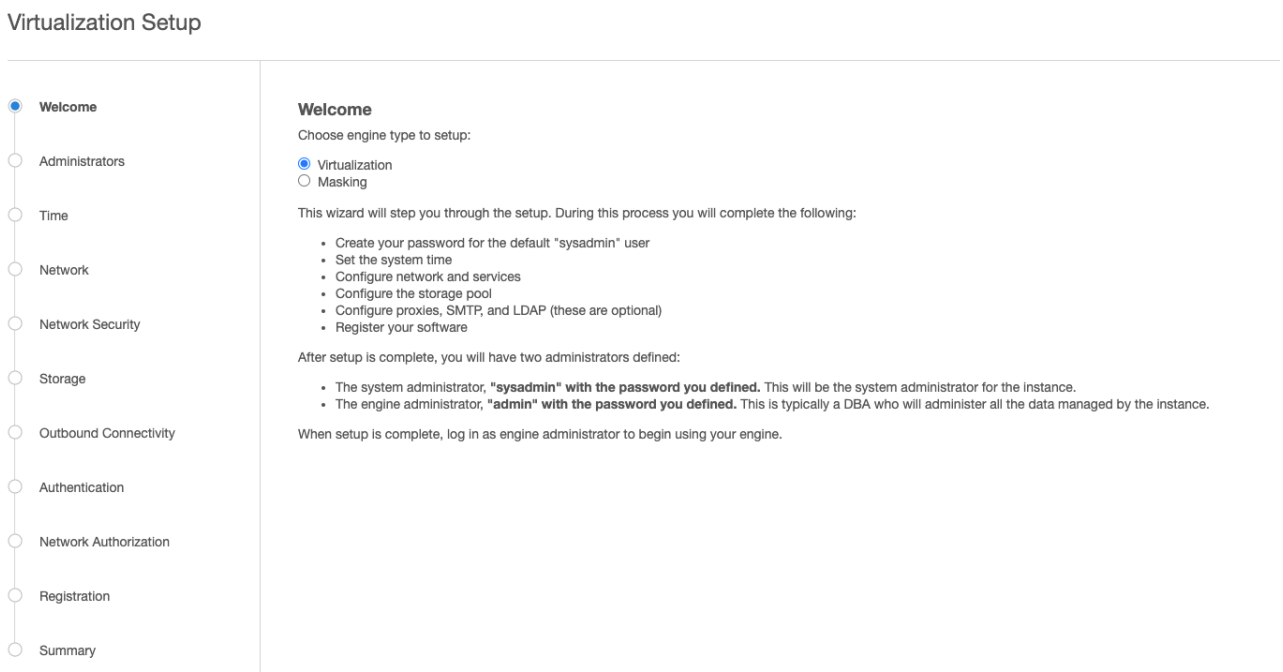
11. Check that you can now access the Delphix Engine through a Web browser by navigating to the displayed IP address, or hostname if using DNS.
12. Exit setup.

```
delphix> exit
```

Setting up the Delphix engine

Once you set up the network access for Delphix Engine, navigate to the Delphix Engine URL in your browser for server setup.

The welcome screen below will appear for you to begin your Delphix Engine setup.



The setup procedure uses a wizard process to take you through a set of configuration screens:

- Administrators
- Time
- Network
- Network Security
- Storage
- Outbound Connectivity

- Authentication
- Network Authorization
- Registration
- Summary

1. Connect to the Delphix Engine at `http://<Delphix Engine>/login/index.html#serverSetup` The **Delphix Setup** application will launch when you connect to the server. Enter your **sysadmin** login credentials, which initially defaults to the username **sysadmin**, with the initial default password of **sysadmin**. On first login, you will be prompted to change the initial default password.
2. Click **Next**.

Administrators

The Delphix Engine supports two types of administrators:

- System Administrator (**sysadmin**) - this is the engine system administrator. The sysadmin password is defined here**.**
- Engine Administrator (**admin)** - this is typically a DBA who will administer all the data managed by the engine.

On the Administrators tab, you set up the sysadmin password by entering an email address and password. The details for the admin are displayed for reference.

> The default domain user created on Delphix Engines from 5.3.1 is known as **admin** instead of delphix_admin. When engines created before 5.3.1 are upgraded to 5.3.1 or later they will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then Disabling delphix_admin.

System Time

The engine time is used as the baseline for setting policies that coordinate between virtual databases and external applications.

Choose your option to set up system time in this section. For a Quick Start, simply set the time and your timezone. You can change this later.

Network

The initial out-of-the-box network configuration in the OVA file is set to use a single VMXNET3 network adapter.

You have already configured this in the initial configuration. Delphix supports more advanced configurations, but you can enable those later.

Storage

You should see the data storage VMDKs or RDMs you created during the OVA installation. Click **Next** to configure these for data storage.

Serviceability

Choose your options to configure serviceability settings.

For a Quick Start, accept the defaults. You can change this later.

Authentication

Choose your options to configure authentication services.

For a Quick Start, accept the defaults. You can change this later.

Registration

If the Delphix Engine has access to the external Internet (either directly or through a web proxy), then you can auto-register the Delphix Engine:

1. Enter your **Support Username** and **Support Password**.
2. Click **Register**.

If external connectivity is not immediately available, you must perform manual registration.

1. Copy the **Delphix Engine registration code** in one of two ways:
    a. Manually highlight the registration code and copy it to clipboard. Or,
    b. Click **Copy Registration Code to Clipboard**.
2. Transfer the Delphix Engine's registration code to a workstation with access to the external network Internet. For example, you could e-mail the registration code to an externally accessible e-mail account.
3. On a machine with access to the external Internet, please use your browser to navigate to the Delphix Registration Portal at http://register.delphix.com.
4. Login with your Delphix support credentials (username and password).
5. Paste the **Registration Code**.
6. Click **Register**.

> Although your Delphix Engine will work without registration, we strongly recommend that you register each Delphix Engine as part of the setup. Failing to register the Delphix Engine will impact its supportability and security in future versions.

To regenerate the registration code for a Delphix Engine please refer to, Regenerating the Delphix Engine Registration Code. Delphix strongly recommends that you regenerate this code and re-register the engine regularly to maximize the Support Security of the Delphix Engine. Delphix recommends doing this every six months.

Summary

The final summary tab will enable you to review your configurations for System Time, Network, Storage, Serviceability, and Authentication.

1. Click the **Back** button to go back and to change the configuration for any of these server settings.
2. If you are ready to proceed, then click **Submit**.
3. Click **Yes** to confirm that you want to save the configuration.
4. Click **Setup** to acknowledge the successful configuration.
5. There will be a wait of several minutes as the Delphix Engine completes the configuration.

Requirements for SAP ASE hosts and databases

In order to begin using SAP ASE environments with Delphix, you will need to configure the source and target hosts with the requirements described on this page.

SAP ASE source host requirements

There must be an operating system user, such as delphix_os, that meets the following requirements:

- The $SYBASE environment variable is defined for non-interactive shells (such as via the .bashrc configuration file).
    - Set the PermitUserEnvironment configuration parameter to "yes" in the sshd_config file
    - Add the variable to the user's .ssh/environment file
    - Restart the SSH daemon

- To test this requirement:

```
ssh delphix_user@ase_hostname env | grep SYBASE
```

- Can login to the source host via SSH (TCP port 22)
- Delphix requires superuser permission to run **pargs** in order to discover Solaris ASE instances. For more information, see Sudo Privilege Requirements for SAP ASE Environments.
- Designating the Delphix operating system user's primary group to be the same as the ASE instance's means the file system permissions can be more restrictive and is a better security practice than granting world read access to the toolkit or the backup files. If the target host is used to host the staging databases, consider the following:
    - If you don't add the Delphix operating system user to the ASE instance owner's group, greater permissions will need to be **granted to the backup files to ensure read access to the dumps and/or transaction logs**. Delphix looks for the backup files on the staging host (unless a "remote" backup server is used in which case, the remote host is used which is often the source environment).
    - If you sync Delphix with a dSource by asking ASE to create a new backup, the ASE instance owner will need the **write permission to the toolkit** (or the mount point if you use the CLI to specify a directory other than the toolkit). Delphix will issue the "DUMP DATABASE" command to write to the staging database's "temp" directory which is mounted on the staging host.
    - Has **write permission for the mount-point directory** (by default the toolkit directory but can be a separate mount point specified in the command line interface).
- There must be a directory on the source host where you can install the Delphix platform toolkit, for example: /var/opt/delphix/Toolkit
    - The delphix_os user must own the directory
    - The directory must have permissions 0770, for example, -rwxrwx---. However, you can also use more permissive settings.
    - The directory should have 256MB of available storage.

Source database requirements

When adding a source ASE environment to Delphix, you may use a single login to discover the ASE instances and link the source databases OR you may use a single login to discover all of the ASE instances and separate logins to link each dSource.

- Delphix uses a single database user for the discovery of all ASE instances and their databases for each environment added to Delphix.
    - The discovery database user (delphix_disc for example) must have SELECT privileges on the following tables for each ASE instance on the source host:
        - **sysdatabases**
        - **sysservers**
        - **syslisteners**
        - **sysconfigures**
        - **syscurconfigs**
- Another user must be specified when linking each dSource (delphix_link for example) that has SELECT privileges on the above tables.
    - If you will select New Full Backup when linking, this user must also have privileges to take a new full database dump of the source database. For more information about linking options, see Linking an SAP ASE Data Source
    - The link database user can be different for each instance and database on the source host.

If the source database is resized and trunc log on chkpt is disabled, take a transaction log dump immediately after the resize operation completes. If trunc log on chkpt is enabled, take a full database dump immediately after the resize operation completes. If multiple resizing operations are performed without taking transaction log dumps between each operation it may be necessary to manually sync the dSource with a new full database dump for Delphix to be able to continue ingesting source database dumps.

Target host requirements

- The operating system on the target environment must be the same as, or binary compatible with, the operating system on the source environment.
- As the Delphix Engine supports both NFSv3 and NFSv4 for mounting target host filesystems, the prerequisite packages that support NFSv3 or NFSv4 client communication are required for normal operation, and the required services to support NFS client communications (including file locking) must be running.This includes:
    a. portmapper / rpcbind
    b. status daemon (rpc.statd)
    c. NFS lock manager (rpc.lockd/lockmgr)
- The SAP ASE major version on the target environment must be the same as the version on the source environment. However, EBF/SP version on target environment can be different. If the target is used as a staging server, the ASE major version must be the same. The only caveat is for ASE 15.7 - where the source and staging patch levels must both be either any version below SP64 or both be any version above SP100.
- There must be an operating system user, such as delphix_os, that meets the following requirements:
    a. Set the PermitUserEnvironment configuration parameter to "yes" in the sshd_config file
    b. Add the variable to the user's .ssh/environment file
    c. Restart the SSH daemon
    d. The $SYBASE environment variable is set for non-interactive shells (such as via the .bashrc configuration file). Set the variable as follows:
    e. To test this requirement:

```
ssh delphix_user@ase_hostname env | grep SYBASE
```

- Can login to the target host via Secure Shell (SSH)
- Can login to ASE instances using isql with LANG=C set
- Designating the Delphix operating system user's primary group to be the same as the ASE instance's means the file system permissions can be more restrictive and is a better security practice than granting world read access to the toolkit or the backup files. If the target host is used to host the staging databases, consider the following:
    - If you don't add the Delphix operating system user to the ASE instance owner's group, greater permissions will need to be **granted to the backup files to ensure read access to the dumps and/or transaction logs**. Delphix looks for the backup files on the staging host (unless a "remote" backup server is used in which case, the remote host is used which is often the source environment).
    - If you sync Delphix with a dSource by asking ASE to create a new backup, the ASE instance owner will need **write permission to the toolkit** (or the mount point if you use the CLI to specify a directory other than the toolkit). Delphix will issue the "DUMP DATABASE" command to write to staging database's "temp" directory which is mounted on the staging host.
    - Has **write permission for the mount-point directory** (by default the toolkit directory but can be a separate mount point specified in the command line interface).
- The following permissions are usually granted via sudo authorization of the commands. See Sudo Privilege Requirements for SAP ASE Environments for further explanation of this requirement, and Sudo File Configuration Examples for SAP ASE Environments for example file configurations.

a. Permission to run **mount** and **umount**as super-user.
b. On Solaris, permission to run **pargs**on Solaris
c. On AIX, permission to run the **nfso**command as super-user.
d. (Optional) On AIX and Linux, permission to run **ps**as super-user.
e. Disable **tty** for the delphix_os user for mount and umount.

- There must be a directory on the source host where you can install the Delphix platform toolkit, for example: /var/opt/delphix/Toolkit
  - The delphix_os user must own the directory
  - The directory must have permissions 0770, for example, -rwxrwx--. However, you can also use more permissive settings.
  - The directory should have 1GB of available storage
  - Avoid using the home directory of the delphix_os user
  - If you intend to use the LogSync feature, it is recommended to make the toolkit directory as short as possible to keep the full path to the transaction log file names under ASE's 127 character limit. For example, create the toolkit directory as /tk. Alternatively, link the dSource using the command line interface and specify the "mountBase" parameter to mount the staging database's devices under a directory following your own naming convention.

Target database requirements

- There must be a database user, such as delphix_db, with the **sa_role** on each instance on the target environment
- The database user such as delphix_db for any staging instances must also have the **sybase_ts_role**
- If the target host will be used as a staging target environment, at least one of the following two options must be configured:
  - You must use **sp_addserver** to add the staging ASE instance's Backup Server to **sysservers** on the source ASE instance (so that remote database dump/load works)
    OR
  - Full and transaction dump files from the source database must be available locally to the staging database (over NFS, replication, scp, etc.)

Specific ASE tuning recommendations

- In ASE 15.7 SP60 and higher, there is a configuration parameter named "**enable large pool for load**". ASE automatically tunes the caches for recovery on reboot, but does not do this for load database and load transaction recovery, since it could impact other databases on a production server. Delphix recommends that the staging ASE instance be separate from the ASE production instance and ASE instances hosting VDBs so enabling this parameter should have ***a beneficial impact on performance for the ASE staging instance.***
- Specify the "**relaxed**" strategy for replacing the cache in the default data cache (**sp_cacheconfig 'default data cache', relaxed**) in the ASE staging instance. Since the mount/unmount process invalidates the pages anyway, the page chain is really just unneeded overhead on the staging server.
- Staging and target ASE instances should have disk mirroring disabled.

sp_configure "disable disk mirroring" – run value should be 1, which is the default. If it is 0, change it using

sp_configure "disable disk mirroring", 1 – this parameter is static so the ASE instance must be restarted for this change to take effect.

- Delphix will mirror the number of devices used on the source database for the staging database (dSource) and each VDB created from that source database. The number of devices parameter should be scaled appropriately based on the max number of virtual databases that will be provisioned to the ASE instance. This parameter can be changed using: **sp_configure "number of devices",** .

ASE 15.7.0 SP100 and later releases support the shrink command. In some cases, Delphix must increase the number of devices used for databases if this command is used. Delphix creates a minimum of the same number of devices as the source database for the staging database (dSource) and each VDB and will add more devices for every 4TB of fragment holes. See SAP ASE issue CR#799273 for additional details.

To support multiple VDBs and the staging databases, you may need to increase the parameter number **of alarms**.

Delphix uses ASE operations that use alarm structures such as **MOUNT** and **UNMOUNT**. The number of alarms limits the number of these operations which can be run concurrently. Various ASE instance failures can occur if the available alarm structures are exhausted. The amount of memory consumed by increasing the number of alarm structures is small. Delphix recommends that the **number of alarms** value is increased to at least 4096.

Adding SAP ASE source and target environments

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. From the **Actions (...)** menu select **Add Environment**.
5. In the **Host and Server** tab window, select **Unix/Linux**.
6. Select **Standalone Host.**
7. Click **Next.**
8. Enter a **Name** for the environment.
9. Enter the **Host IP** address.
10. For NFS Addresses (Optional): Enter one or more comma-separated **IP Address/Hostname**
    Note: If specified, Delphix Engine only allows NFS requests (mount, etc) originated from IP Addresses specified for the host.
11. Enter the **SSH** port. The default value is **22**.
12. Select a **Login Type**. — Username and Password - enter the OS username and password — Username and Public Key - enter the OS username. — Password Vault - select from an existing Enterprise Password Vault

    Using public key authentication
    If you want to use public-key authentication for logging into your Unix-based environment, there are two options: use the engine's key pair or provide a key pair for this environment.
    To use the engine's key pair:
    a. Select **Public Key** for the **Login Type.**
    b. Click **View Public Key**.
    c. Copy the public key that is displayed, and append it to the end of your `~/.ssh/authorized_keys` file. If this file does not exist, you will need to create it.
        i. Run chmod 600 ~/.ssh/authorized_keys to allow only the file's owner to read and write to it (make sure the file is owned by the user).
        ii. Run chmod 755 ~ to restrict access to the user's home directory so no other user may write to it.
        iii. Run chmod 700 ~/.ssh so that others cannot write to it. The ~/.ssh directory cannot be writable by group or other users. Otherwise, authentication will fail.

    As an alternative, you can provide a key pair specific for this environment via the API or CLI. See Option 2 in this CLI Cookbook article for instructions
13. For **Password Login**, click **Verify Credentials** to test the username and password.

14. Enter a **Toolkit Path**. The toolkit directory stores scripts used for Delphix Engine operations. It must have a persistent working directory rather than a temporary one. The toolkit directory will have a separate subdirectory for each database instance. The toolkit path must have 0770 permissions.
15. To provide your own Oracle Java select the **Provide my own JDK** checkbox and click **Next**.
16. In the Java Development Kit tab enter the absolute path to your Oracle JDK and click **Next**.
17. Click the **Discover SAP ASE** checkbox.
18. Click **Next**.
19. In the Summary, tab confirm your selections.
20. Click **Submit**.

ASE manual discovery

When an environment is added, Delphix discovers your ASE instances. Manual discovery allows users to add instances that were not automatically discovered. This feature is currently only supported via the CLI.

This topic describes how to use CLI commands to manually add ASE repositories to an SAP ASE environment. Discovery is the process by which the Delphix platform identifies data sources and data dependencies on a remote environment. ASE repository discovery is done automatically when an environment is added to the Delphix platform or when an already added environment is refreshed. In some cases, automatic discovery does not discover all of the repositories in an SAP ASE environment. These repositories may be added using manual discovery.

Unlike automatically discovered instances, manually discovered instances are not automatically deleted if the environment is refreshed when the instance isn't running. Manually discovered instances are not updated during an environment refresh either. So for example, if you upgrade ASE to a new version or change the listener port, you must manually update the repository.

To manually discover an ASE repository you will need to:

- Add the environment to Delphix
- Use CLI to manually discover a repository

Creating an ASE environment

Please refer to Adding an SAP ASE Environment for detailed steps.

Manually discovering a repository

ⓘ In the following example, we are using **sc-dev3.dc2** as our example environment.

1. Log into CLI and cd to repository menu:

```
$ ssh delphix_admin@sc-dev3.dc2
Password:
sc-dev3.dc2> cd repository
sc-dev3.dc2 repository>
```

2. Add (manually discover) an ASE repository instance:

Note: The values used in the following code block are specific to the example instance we are adding.

```
sc-dev3.dc2 repository> create
sc-dev3.dc2 repository create *> ls
```

```
Properties
  type: ASEInstance
  credentials: (unset)
  dbUser: (unset)
  environment: (required)
  installationPath: (required)
  instanceName: (required)
  instanceOwner: (required)
  ports: (required)
  version: (unset)
sc-dev3.dc2 repository create *> set credentials.password=sybase
sc-dev3.dc2 repository create *> set dbUser=sa
sc-dev3.dc2 repository create *> set environment=sc-rhel64-sybase-ase-0
sc-dev3.dc2 repository create *> set installationPath=/opt/sybase/15-7
sc-dev3.dc2 repository create *> set instanceName=ASE1570_S1
sc-dev3.dc2 repository create *> set instanceOwner=sybase
sc-dev3.dc2 repository create *> set ports=5100
sc-dev3.dc2 repository create *> ls
Properties
  type: ASEInstance
  credentials:
      type: PasswordCredential (*)
      password: ******** (*)
  dbUser: sa (*)
  environment: sc-rhel64-sybase-ase-0 (*)
  installationPath: /opt/sybase/15-7 (*)
  instanceName: ASE1570_S1 (*)
  instanceOwner: sybase (*)
  ports: 5100 (*)
  version: (unset)
sc-dev3.dc2 repository create *> commit
  `ASE_INSTANCE-22
sc-dev3.dc2 repository>
```

Updating a repository

Adding onto the above, the following example illustrates updating an ASE instance's version after upgrading ASE:

```
sc-dev3.dc2> repository
sc-dev3.dc2 repository> select ASE1570_S1
sc-dev3.dc2 repository 'ASE1570_S1'> update
sc-dev3.dc2 repository 'ASE1570_S1' update *> set version="15.7 SP138"
sc-dev3.dc2 repository 'ASE1570_S1' update *> ls
Properties
   type: ASEInstance
   credentials:
       type: PasswordCredential
       password: ********
   dbUser: sa
   installationPath: /opt/sybase/15-7
   instanceOwner: sybase
```

```
    linkingEnabled: true
    ports: 5100
    provisioningEnabled: true
    servicePrincipalName: (unset)
    staging: false
    version: 15.7 SP138 (*)
sc-dev3.dc2 repository 'ASE1570_S1' update *> commit
```

Be careful when setting the version string. Make sure that it matches output as displayed by the "select @@version" query all the way out to the patch level (PL). For example "15.7 SP138" or "16.0 SP02 PL01".

Enable linking and provisioning for SAP ASE environments

This topic describes how to enable and disable provisioning and linking for SAP ASE databases.

Before a database can be used as a dSource, you must first make sure that you have enabled linking to its SAP ASE instance. Similarly, before you can provision a VDB to a target database, you must make sure that you have enabled provisioning to its SAP ASE instance.

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Select the **Databases** tab.
5. Click the **pencil** icon located next to the database **Installation Details**.
6. Select the **Allow Provisioning** checkbox to enable provisioning, deselect the checkbox to disable provisioning.
7. Click **show details** for the database.
8. Slide the button next to **Allow Linking** to **On** or **Off** to enable or disable linking.

Linking an SAP ASE data source

The dSource is an object that the Delphix Virtualization Engine uses to create and update virtual copies of your database. As a virtualized representation of your source data, it cannot be managed, manipulated, or examined by database tools. For an overview of all dSource related actions, please visit Managing Data Sources. Delphix Virtualization for SAP ASE databases leverages backup-based ingestion, which means that Delphix will look for, or sometimes initiate the creation of, a backup through your SAP ASE backup server. From there, the backup is restored on a staging server and the staging copy is then ingested into Delphix. See Delphix Architecture with SAP ASE for more information.

When linking a dSource from an SAP ASE source database, Delphix offers several different methods of capturing backup information:

- ASE Managed Backups, where the SAP ASE source database schedules and initiates backups. This method supports various backup types which include:
    - Full backups
    - Transaction log backups (with LogSync disabled)
    - Transaction log backups (with LogSync enabled)
- Delphix Managed Backups, where the Delphix Engine schedules and initiates the backups from the source database, and captures them.

ASE managed backups

Further contextual information on the various backup types (listed above):

- Full Backups - A snapshot will be created on the Delphix Timeflow for each Full backup.
- Transaction log backups (with LogSync disabled) - A snapshot will be created on the Delphix Timeflow for each transaction log backup.

> ⚠ Transaction Logs
> Transaction logs are not collected if:
> - a) there are gaps in the sequence of log backups (a break in the “log chain”).
> - b) the available log backups do not include any changes since the last successful Delphix snapshot.

Transaction log backups (with LogSync enabled) - A snapshot will be created on the Delphix Timeflow for each transaction log backup. In addition, point-in-time provisioning will be an available option if you would like to provision from any point in between snapshots.

> ⚠ Log Files
> Log files consume additional space on the Delphix Engine and are managed according to the defined retention policy for logs.

Delphix managed backups

When the checkbox for Delphix Managed Backups is selected, the Delphix Engine will initiate a full backup of the source database for the initial load of the dSource. Thereafter, the Delphix Engine will initiate full backups of the source database using the schedule specified by the selected SnapSync Policy. If you select the None policy, the Delphix Engine will not automatically initiate a full backup, but you can initiate them manually using the snapshot (camera) icon.

Delphix looks for the backup files on the staging host (unless a "remote" backup server is used in which case, the remote host is used which is often the source environment)

Procedure

1. Login to the **Delphix Management** application.
2. Navigate to **Manage > Datasets.**
3. Click the plus icon and select **Add dSource**.
   Note: Delphix looks for the backup files on the staging host (unless a "remote" backup server is used in which case, the remote host is used which is often the source environment.
4. In the **Add dSource** wizard, select the source environment with the correct environment-based user.
5. Enter your login credentials for the source database and click **Next**.
6. Enter a name and select a group for your dSource. Adding a dSource to a dataset group lets you set Delphix Domain user permissions for that database and its objects, such as snapshots. See the topics under Users and Groups for more information.
7. Select the **Data Management** settings needed, as described in Data Management Settings for ASE Data Sources.
8. From the **Data Management** tab under the **Initial Load** option, select and enter any **additional settings** needed. There are three options for the initial load of the dSource:
   a. If the source ASE instance resides on the same server as the staging ASE instance, the staging database's NFS mounted "temp" directory will be present for the source database to write to in response to the "DUMP DATABASE" command.
   b. If the source and staging ASE instances are configured to allow remote access to the backup servers and the dSource is linked using the Remote Server option as described below. Delphix will then issue the "DUMP DATABASE" command and append the "AT <staging_backup_server_name>" clause so that the dump is written to the staging backup server.

   c. (Recommended) **New Full Backup** - Lets Delphix create a new full backup file and load it. Note - that when Delphix creates the backup, it is moved to Delphix's NFS-mounted storage located on the stage host rather than. The backup will be located in the "temp" directory and will be deleted once the Delphix Engine has restored the backup and created a dSource from the restored staging database. This means that this option will work under two scenarios:
   d. Most Recent **Existing Full Backu**p – Find the most recent existing full backup file in the Backup Location and load it.
   Note: If Dump History is not active on the Source Database: Choosing this option can delay completion of the dSource link as Delphix attempts to find and catalog every single backup listed in the source database's backup server log file.
   e. Specific **Existing Full Backu**p – Specify which backup files in the Backup Location you want to load. Choosing this option is much faster because Delphix will skip directly to loading the desired backup and only start to search for and catalog backups in the background after the linking of the source database has completed.
   Note: When using a dump taken with the deprecated compression syntax, select the Specific Existing Full Backup option for Initial Load and, for each stripe, type compress::<file name> into the text box.
9. Select the Staging environment and ASE instance that will be used to manage the staging database used for validated sync of the dSource.
10. Select any policies for the new dSource.
11. Click **Next**, then specify any pre-hook and post-hook scripts.
12. Review the dSource Configuration and Data Management information, and click **Submit** to begin provisioning the VDB.

Provisioning an SAP ASE VDB

Procedure

1. In the Datasets panel on the left-hand side, click the group containing the dSource or VDB from which you want to provision.
2. Click the **Timeflow** tab.
3. Select a snapshot or open LogSync timeline to provision by a specific log or point in time.
   - Find more detail about initial provisioning options in the section 'Provisioning by Snapshot or LogSync' below.
4. Click to open the Provision VDB wizard, and select a compatible Target Environment for the new ASE VDB
5. Review the information presented for Target Configuration and edit as necessary.
6. Select a Snapshot Policy for the VDB.
7. (Optional) - Selective Data Distribution - After policies, there is a masking option.
8. Enter any operations that should be run in the Hooks page. These scripts can be managed after provision in the VDB's configuration page.

When provisioning starts, you can review the progress of the job by selecting the VDB and clicking on the Status tab, or by selecting Manage/Dashboards and viewing the Job History panel. Alternatively, you could see this in the Actions Sidebar. When provisioning is complete, the VDB will be included in the group you designated and listed in the Datasets panel. If you select the VDB in the Datasets panel and click the Configuration tab, you can view information about the database and its Data Management settings.

Provisioning by snapshot or LogSync

When provisioning by Snapshot, you can provision to the start of any particular snapshot by time.

Provisioning by snapshot/time	description
Provisioning By Snapshot	You can provision by using a Snapshot. In that case, a new VDB will be provisioned to the database state as of the Snapshot.
Provision by Time	If you have enabled Log Sync, you can provision a new database to a point in time. You can select a snapshot and then using time entry fields, specify a Point in Time. Delphix will use the selected snapshot to restore the VDB and use the log files to roll forward the VDB to the selected time.

Configuration settings for ASE virtual databases

Each VDB has its own data management settings, found during the provisioning workflow as well as in the configuration page for that VDB. When you create a SAP ASE VDB, Delphix copies most configuration settings from the dSource and uses them to create the VDB. However, you can customize these with the following settings:

Setting	Explanation
Recovery Model	The current recovery model of the source database. This field will auto-populate with information from the dSource.
Auto VDB Restart	Enabling this option will automatically restart this VDB whenever its target host is rebooted.

Automatic VDB restart on target server after reboot

The Delphix platform now automatically detects whether a target server has been rebooted, and proactively restarts any VBD on that server that was previously up and running. This is independent of the data platform. It is done as if you realized a target server was restarted and issued a start command from the Delphix platform. This feature is compatible with Self-Service ordering dependencies and is limited to non-clustered VDBs.

To enable automatic restart, complete the following steps:

- When provisioning a new VDB in the VDB Provisioning wizard, check the **Auto VDB Restart** box.

Once the VDB has been provisioned, you will be able to turn **Automatic VDB Restart** on.

1. In the **Datasets** panel, select the VDB.
2. Select the **Configuration** tab.
3. Select **Source** sub-tab.
4. Select **Database edit**.

Next steps

Congratulations! You have provisioned your first virtual database!

Now, perform some simple functional tests with your application. You can connect your app to the VDB using standard TNS/JDBC techniques. Delphix has already registered the VDB for you on the target listener.

We suggest the following next steps:

1. Drop a table and use the VDB Rewind feature to test the recovery of your VDB.
2. Take a snapshot of your dSource and refresh your VDB to quickly get fresh production data.
3. Provision a new VDB from your VDB to test sharing data quickly with other teams.
4. Mask your new VDB to protect sensitive data. Provision new VDBs from that masked VDB to quickly provide safe data to development and QA teams.

SAP ASE support and requirements

In order to begin using SAP ASE environments with Delphix, you will need to configure the source and target hosts with the requirements described in this section.

This section covers the following topics:

- Requirements for SAP ASE environments and databases
- Network and connectivity requirements for SAP ASE environments
- Sudo privilege requirements for SAP ASE environments
- Sudo file configuration examples for SAP ASE environments

To view SAP ASE support matrix, see SAP ASE matrix.

Requirements for SAP ASE environments and databases

In order to begin using SAP ASE environments with Delphix, you will need to configure the source and target hosts with the requirements described on this page.

SAP ASE host requirements

> When working with ASE dSources, it is necessary that the Staging and Target ASE instances have an identical **Page Size** configuration to the dSource (e.g. 2KB / 4KB / 8KB). This is set during ASE instance installation.
> The Delphix Engine will prevent the use of incompatible instances.

These requirements are applicable to source, staging, and target environments. Target environments have additional requirements, which are detailed in the **Target Host Requirements** section below.

On each host with SAP ASE, there must be an operating system user (e.g. delphix_os) configured to the required specifications for Delphix.

Host requirement	Explanation
Profile and privileges should be the same as the ASE user (e.g. Sybase) on the host.	For example, delphix_os should have the same environment variable (e.g. $PATH), umask, and ulimit settings as the user sybase.
Delphix operating system user's primary group should be the same as SAP ASE instance's group.	To perform Sync/Validated Sync operations on the source database, Delphix reads a number of files like SAP ASE Backup server log file, ASE Dump History file, and backup files. If Delphix OS user does not have the same group as of the SAP ASE instance, these files will need "everyone" read permissions, which is less restrictive.
Delphix Engine Toolkit Directory on Source Host (e.g. /var/opt/delphix/toolkit) • The delphix_os user must own the directory • The directory must have permissions 0770, for example, -rwxrwx---. However, you can also use more permissive settings. • The directory should have 256MB of available storage	• If you sync Delphix with a dSource by asking SAP ASE to create a new backup, the SAP ASE instance owner will need write permissions to the toolkit (or the mount point if you use the CLI to specify a directory other than the toolkit). Delphix will issue the "DUMP DATABASE" command to write to the staging database's "temp" directory which is mounted on the staging host. • Has write permission for the mount-point directory (by default the toolkit directory but can be a separate mount point specified in the command line interface)
Backup Server Log File Permissions	• In the case that the Delphix OS user is not in the same group as the SAP ASE instance user, you need to change permissions to SAP ASE Backup server log file from "rw-r---" to "rw-r-r--". • The "rw-r---" to "rw-r-r--" permissions change is also required if Dump History files are in use.

Host requirement	Explanation
The Delphix Engine must be able to make an SSH connection to the source host (typically port 22).	

SAP ASE database requirements

Delphix for SAP ASE requires a specific login to run discovery and linking activities for instances and databases. Delphix requires specific user permissions for database discovery and linking (detailed below).

⚠ Discovery and Linking can be separated into separate users if desired.

⚠ If you are using an ASE backup server instance shared with multiple ASE data server instances, the use of the dump history file feature is mandatory. This should be enabled on all instances and the dump history file in a location accessible by the Delphix OS user account from the Staging host.

SAP ASE source database requirements

The discovery database user (delphix_disc for example) must have SELECT privileges on the following tables for each SAP ASE instance on the source host:

- Sysdatabases
- Sysservers
- Syslisteners
- Sysconfigures
- Syscurconfigs

Database linking

Permissions for linking environments can be included under the “discovery” user or can be a separate, dedicated role - e.g. delphix_link. If separate, the “linking” user must be specified when linking each dSource (delphix_link for example) that has SELECT privileges on the above tables.

If you will select **New Full Backup** when linking, this user must also have privileges to take a new full database dump of the source database. For more information about linking options, see Linking an SAP ASE Data Source

The link database user can be different for each instance and database on the source host.

Sample Script to create delphix_link on Linux

```
note: run as sa

sp_addlogin delphix_link, "StrongPassword"

go

sp_adduser delphix_link

go
```

```
grant select on sysdatabases to delphix_link

go

grant select on sysservers to delphix_link

go

grant select on syslisteners to delphix_link

go
```

Requirements when resizing source databases

For SAP ASE on Delphix, transaction logs are helpful for maintaining a Timeflow via logsync. However, not all customers will want to maintain these logs for bandwidth purposes. The following section will help guide users on requirements to maintain timeflows when it comes to resizing databases with and without (`trunc log on chkpt`).

Active SAP ASE Feature on Source Host	Requirement
(`trunc log on chkpt`) is disabled	take a transaction log dump immediately after the resize operation completes.
(`trunc log on chkpt`) is enabled	take a full database dump immediately after the resize operation completes.

If multiple resizing operations are performed without taking transaction log dumps between each operation it may be necessary to manually sync the source database dumps.

Additional target host requirements

This section describes the user privileges, and environment discovery requirements, that are required for SAP ASE target hosts and databases, collectively referred to as target environments.

Target host requirement	Explanation
The operating system on the target environment must be the same as, or binary compatible with, the operating system on the source environment	

Target host requirement	Explanation
As the Delphix Engine uses NFSv3 for mounting target host filesystems, the prerequisite packages to support NFSv3 client communication are required for normal operation, and the required services to support NFS client communications (including file locking) must be running This includes: • portmapper / rpcbind • status daemon (rpc.statd) • NFS lock manager (rpc.lockd/lockmgr)	The Delphix Engine uses NFSv3 as a defacto mounting standard for target host file systems.
SAP ASE Source/Target Environment Version Compatibility	• The major/minor/macro version of the target ASE instance should be the same or higher than the version of the source ASE instance. Delphix supports the ASE databases to be restored from lower to higher version or within different patch/ sp levels as long as SAP ASE supports it. • If the target environment is used as staging then the SAP ASE version on the target environment must be the same as the version on the source environment. However, EBF/SP version on the target environment can be different.
Create OS User (e.g. delphix_os)	There must be an operating system user, such as delphix_os, that meets the following requirements:
OS User must have - The $SYBASE environment variable set for non-interactive shells (such as via the .bashrc configuration file)	Additional Settings for the $SYBASE environment variable: • Set the PermitUserEnvironment configuration parameter to "yes" in the sshd_config file • Add the variable to the user's .ssh/environment file • Restart the SSH daemon
OS User must have appropriate access to Target Hosts/Instances	• Can login to the target host via Secure Shell (SSH) • Can login to ASE instances using isql with LANG=C set
OS User’s Primary Group must be the same as ASE Instance’s Primary Group	• If not, the file system permissions can be more restrictive. • This also is a better security practice than granting world read access to the toolkit or the backup files.

Target host requirement	Explanation
Permissions granted via sudo authorization of the commands: • Disable tty for the delphix_os user for mount and umount • Permission to run mount and umount as super-user. • On Solaris, permission to run pargs on Solaris • On AIX, permission to run the nfso command as super-user. • (Optional) On AIX and Linux, permission to run ps as super-user.	See Sudo Privilege Requirements for SAP ASE Environments for further explanation of this requirement, and Sudo File Configuration Examples for SAP ASE Environments for example file configurations.
There must be a database user, such as delphix_db, with the sa_role on each instance on the target environment	
The database user such as delphix_db for any staging instances must also have the sybase_ts_role	
There must be a directory on the target environment where you can install the Delphix Engine toolkit, for example,/var/opt/delphix/Toolkit.	• The delphix_os user must own the directory • The directory must have permissions 0770, for example, -rwxrwx--. However, you can also use more permissive settings. • The directory should have 1GB of available storage • Avoid using the home directory of the delphix_os user • If you intend to use the LogSync feature, it is recommended to make the toolkit directory as short as possible to keep the full path to the transaction log file names under ASE's 127 character limit. For example, create the toolkit directory as /tk. Alternatively, link the dSource using the command line interface and specify the "mountBase" parameter to mount the staging database's devices under a directory following your own naming convention.

Target host requirement	Explanation
(Optional - If the target host will be used as a staging target environment) Must have at least one of the following two options configured: • use sp_addserver to add the staging ASE instance's Backup Server to sysservers on the source ASE instance (so that remote database dump/load works) OR • Full and transaction dump files from the source database must be available locally to the staging database (over NFS, replication, scp, etc.)	(see Managing SAP ASE Environments)

Special considerations - OS user settings (If target host is hosting staging DBs)

Target host requirement	Explanation
If you don't add the Delphix operating system user to the SAP ASE instance owner's group, greater permissions will need to be granted to the backup files to ensure read access to the dumps and/or transaction logs.	Delphix looks for the backup files on the staging host (unless a "remote" backup server is used in which case, the remote host is used which is often the source environment).
ASE Instance Owner Write Permissions to the toolkit (or mount point if you use the CLI to specify a directory other than the toolkit)	If you sync Delphix with a dSource by asking ASE to create a new backup, the SAP ASE instance owner will need write permission to the toolkit (or the mount point if you use the CLI to specify a directory other than the toolkit). Delphix will issue the "DUMP DATABASE" command to write to the staging database's "temp" directory which is mounted on the staging host.
Has write permission for the mount-point directory	The mount-point directory defaults to the toolkit directory, which has write permissions. However, the mount-point directory can be customized, it only needs to have write permissions.

Additional ASE tuning recommendations

- In SAP ASE 15.7 SP60 and higher, there is a configuration parameter named "enable large pool for load". SAP ASE automatically tunes the caches for recovery on reboot but does not do this for load database and load transaction recovery, since it could impact other databases on a production server. Delphix recommends that the staging SAP ASE instance be separate from the SAP ASE production instance and SAP ASE instances hosting VDBs so enabling this parameter should have *a beneficial impact on performance for the SAP ASE staging instance.*

- Specify the "relaxed" strategy for replacing the cache in the default data cache (sp_cacheconfig ‘default data cache’, relaxed) in the SAP ASE staging instance. Since the mount/unmount process invalidates the pages anyway, the page chain is really just unneeded overhead on the staging server.
- Staging and target SAP ASE instances should have disk mirroring disabled.
  sp_configure "disable disk mirroring" – run value should be 1, which is the default. If it is 0, change it using sp_configure "disable disk mirroring", 1 – this parameter is static so the ASE instance must be restarted for this change to take effect.
- Delphix will mirror the number of devices used on the source database for the staging database (dSource) and each VDB created from that source database. The number of devices parameter should be scaled appropriately based on the max number of virtual databases that will be provisioned to the SAP ASE instance. This parameter can be changed using: sp_configure "number of devices", .
  SAP ASE 15.7.0 SP100 and later releases support the shrink command. In some cases, Delphix must increase the number of devices used for databases if this command is used. Delphix creates a minimum of the same number of devices as the source database for the staging database (dSource) and each VDB and will add more devices for every 4TB of fragment holes. See SAP ASE issue CR#799273 for additional details.

To support multiple VDBs and the staging databases, you may need to increase the parameter number of alarms.

> Delphix uses SAP ASE operations that use alarm structures such as MOUNT and UNMOUNT. The number of alarms limits the number of these operations which can be run concurrently. Various SAP ASE instance failures can occur if the available alarm structures are exhausted. The amount of memory consumed by increasing the number of alarm structures is small. Delphix recommends that the number of alarms value is increased to at least 4096.

ASE manual discovery

When an SAP ASE environment is added Delphix automatically discovers your SAP ASE instances. Manual discovery allows users to add instances that were not automatically discovered. This feature is currently only supported via the CLI. For more information please refer to Configuring ASE Manual Discovery

Network and connectivity requirements for SAP ASE environments

General outbound from the Delphix engine port allocation

Protocol	Port numbers	Use
TCP	25	Connection to a local SMTP server for sending email
TCP/UDP	53	Connections to local DNS servers
UDP	123	Connection to an NTP server
UDP	162	Sending SNMP TRAP messages to an SNMP Manager
TCP	443	HTTPS connections from the Delphix Engine to the Delphix Support upload server
TCP/UDP	636	Secure connections to an LDAP server
TCP	8415	Connections to a Delphix replication target. See Configuring Replication
TCP	50001	Connections to source and target environments for network performance tests.

General inbound to the Delphix engine port allocation

Protocol	Port number	Use
TCP	22	SSH connections to the Delphix Engine
TCP	80	HTTP connections to the Delphix GUI
UDP	161	Messages from an SNMP Manager to the Delphix Engine
TCP	443	HTTPS connections to the Delphix Management Application
TCP	8415	Delphix Session Protocol connections from all DSP-based network services including Replication, SnapSync for Oracle, V2P, and the Delphix Connector.

Protocol	Port number	Use
TCP	50001	Connections from source and target environments for network performance tests via the Delphix CLI.

Firewalls and intrusion detection systems (IDS)

Production databases on source environments (for dSources) are often separated from the non-production environment by firewalls. Firewalls can add milliseconds to the latency between servers. Accordingly, for best performance, there should be no firewalls between the Delphix Engine and the virtual database (VDB) target environments. If the Delphix Engine is separated from a source environment by a firewall, the firewall must be configured to permit network connections between the Delphix Engine and the source environments for the application protocols (ports) listed above.

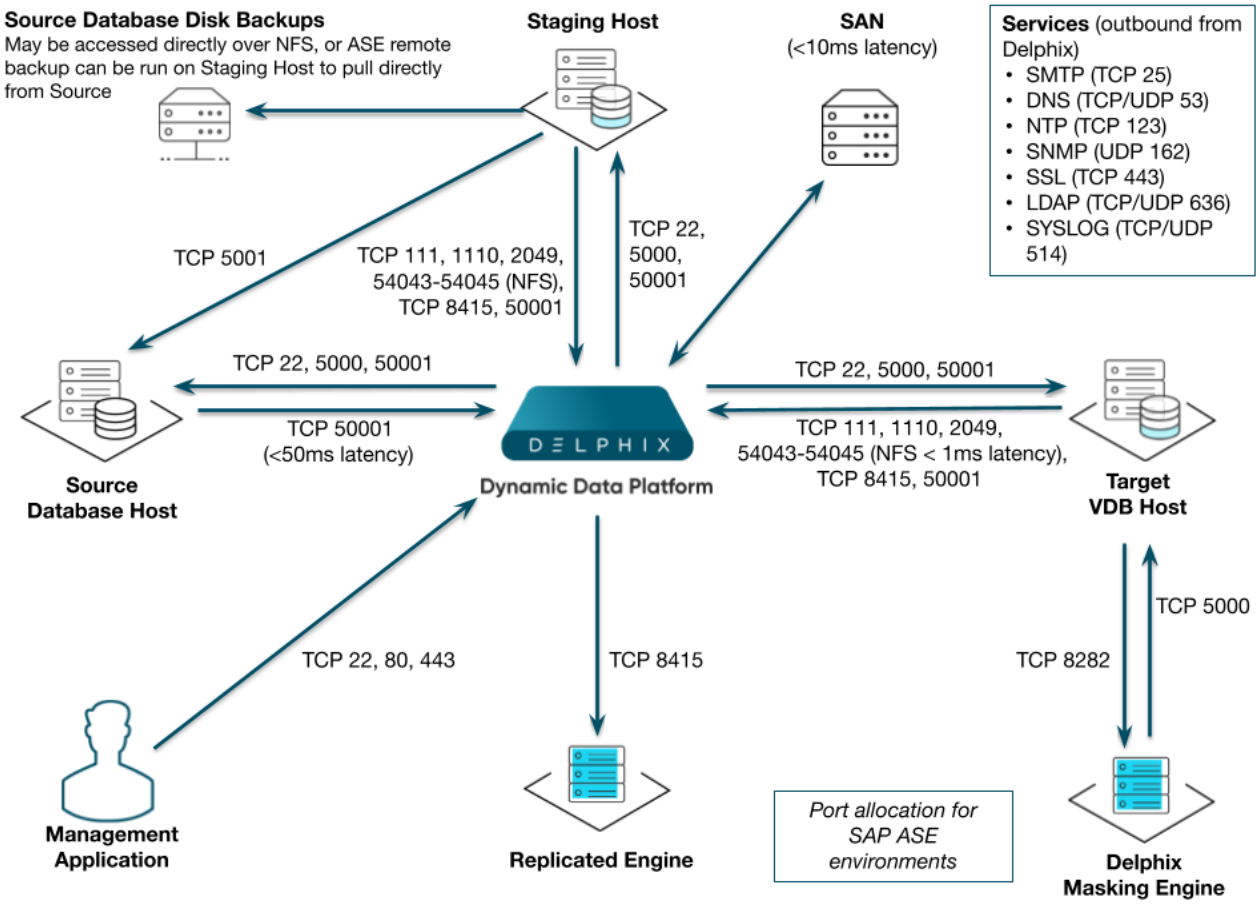
Intrusion detection systems (IDSs) should also be made permissive to the Delphix Engine deployment. IDSs should be made aware of the anticipated high volumes of data transfer between dSources and the Delphix Engine.

Connection requirements for SAP ASE environments

- The Delphix Engine uses an **SSH** connection to each source environment and **SAP ASE client** connections to the SAP ASE instances on the source environment.
- The Delphix Engine uses an **SSH** connection to each target environment, **NFS** connections from each target environment to the Delphix Engine, and **SAP ASE JDBC** connections to the virtual databases on the target environment.

Port allocation for SAP ASE environments

The following diagram describes the port allocations for SAP ASE environments. It illustrates the ports that we recommend to be open from Delphix to remote services, to the Delphix Engine, and to the Target Environments.



Refer to Managing SAP ASE Environments for information on SAP ASE environments. The Delphix Engine makes use of the following network ports for SAP ASE dSources and VDBs:

Outbound from the Delphix engine port allocation

Protocol	Port numbers	Use
TCP	Configuration dependent	JDBC Connections to the SAP ASE instances on the source environments

Inbound to the Delphix engine port allocation

Protocol	Port number	Use
UDP	33434-33464	Traceroute from source and target database servers to the Delphix Engine (optional)
TCP/UDP	111	Remote Procedure Call (RPC) port mapper used for NFSv3 mounts

Protocol	Port number	Use
TCP	2049	NFS client from target hosts to the Delphix Engine (NFSv3 and NFSv4)
TCP	1110	Network Status Monitor (NSM) client from target hosts to Delphix Engine
TCP	54043	Client mount daemon (NFSv3 only)
TCP	54044	Lock state notification service (NFSv3 only)
TCP	54045	Network Lock Manager (NLM) client from target hosts to Delphix Engine (NFSv3 only)

Port allocation between source and staging target environments

Protocol	Port numbers	Use
TCP	Configuration dependent	SAP ASE Remote Backup Server protocol. Applies if linking using the **New Full Backup** option, or if linking with the **Remote Backup Server** option.

Port allocation between staging target environments and shared backup fileserver

Protocol	Port numbers	Use
TCP/UDP	NFS and related port numbers: • Portmap (111) • NFS (2049) • Network Lock Manager (NLM) • Network Status Monitor (NSM)	NFS mount point exported by an NFS shared backup fileserver. Applies if linking using the **Local Backup Server** option.

AppData port requirements

The use of AppData requires the following ports/protocols. Two important notes about these specifications:

1. The next release of the Delphix Engine will significantly augment the port/protocol utilization of AppData. The upcoming-only requirements have been marked with a *.
2. AppData V2P uses RSYNC to export to the target. RSYNC between the target and Delphix Engine is not required for general virtualization usage. The V2P-only requirements have been marked with a ^.

From Source to Delphix Engine	From Delphix Engine to Source	From Target to Delphix Engine	From Delphix Engine to Target
RSYNC (TCP Port 873)	RSYNC (TCP Port 873)	DSP (Default TCP Port 8415)	DSP (Default TCP Port 8415)
DSP (Default TCP Port 8415)	SSH (TCP Port 22)	NFS	SSH (TCP Port 22)
*NFS	DSP (Default TCP Port 8415)	^RSYNC (TCP Port 873)	^RSYNC (TCP Port 873)

Sudo privilege requirements for SAP ASE environments

This topic describes the rationale behind specific `sudo` privilege requirements for virtualizing SAP ASE Databases.

Privilege	Sources	Targets	Rationale
`pargs`	**Required on Solaris**	**Required on Solaris**	Delphix attempts to call `pargs` to discover the arguments of the ASE processes. It needs the name of each running dataserver or backupserver process so that it can try to connect to the instances to gather further information during the discovery process.
`ps`	Optional on Linux, AIX	Optional on Linux, AIX	Delphix attempts to call `ps` to discover the arguments of the ASE processes. It needs the name of each running dataserver or backupserver process so that it can try to connect to the instances to gather further information during the discovery process. Unlike Solaris, Delphix can usually determine the arguments without sudo privileges on Linux/AIX. But Delphix will attempt " `sudo ps` " before attempting a regular `ps` command, and this could cause locking of the **delphix_os** account. To avoid locking issues, you can grant `sudo ps` to **delphix_os.**
`mount/umount`	Not Required	**Required**	Delphix dynamically mounts and unmounts directories under the provisioning directory during VDB operations. This privilege is required because `mount` and `umount` are typically reserved for superuser.
`nfso`	Not Required	**Required on AIX**	Delphix monitors NFS read and write sizes on an AIX target host. It uses the `nfso` command to query the sizes in order to optimize NFS performance for VDBs running on the target host. Only a superuser can issue the `nfso` command.

Default Mount Directory

By default, Delphix mounts the NFS directories for VDBs and staging databases under the toolkit directory. Sudo permissions should be granted to allow the mount/umount commands to execute under these directories unless the dSource is linked using the command-line interface (CLI) and a different NFS mount base is specified. Please refer to the Reference manual for more information on linking the dSource using the CLI and specifying the "mountBase" parameter.

Specify the NOPASSWD qualifier

It is required to specify the NOPASSWD qualifier within the "sudo" configuration file, as shown here: Sudo File Configuration Examples for SAP ASE Environments. This ensures that the "sudo" command does not demand the entry of a password, even for the "display permissions" (i.e. "sudo -l") command.

Delphix issues "sudo -l" in some scripts to detect if the operating system user has the correct sudo privileges. If it is unable to execute this command, some actions may fail and Delphix will raise an alert suggesting it does not have the correct sudo permissions. Restricting the execution of "sudo -l" by setting “listpw=always” in the “/etc/sudoers” file when the Delphix operating system user is configured to use public key authentication will cause the Delphix operating system user to be prompted for a password which will fail certain Delphix actions. Use a less restrictive setting for listpw than "always" when the Delphix operating system user is using public-key authentication.

SAP ASE and appData mount options	
AIX	`-o cio,rw,fg,hard,rsize=$nfs_rsize,wsize=$nfs_wsize,nointr,timeo=600,proto=tcp,noacl`
HPUX	`-o rw,hard,rsize=1048576,wsize=1048576,nointr,timeo=600,proto=tcp,suid`
Solaris	`-F nfs -o rw,fg,hard,rsize=1048576,wsize=1048576,nointr,timeo=600,proto=tcp,suid,sec=sys`
For these platforms, depending on the NFS version used, additional options vers=3 or vers=4.x is added (x varies depending on what that platform supports. e.g. vers=4 or vers=4.1)	
Linux (NFSv3)	-t nfs -o rw,fg,hard,rsize=1048576,wsize=1048576,nointr,timeo=600,tcp,noacl,vers=3
Linux (NFSv4)	-t nfs4 -o rw,fg,hard,rsize=1048576,wsize=1048576,nointr,timeo=600,sec=sys,tcp,noacl
(For some flavors of Linux and NFSv4.1, additional optional 'v4.1' is added)	

1. AppData plugins and toolkits have some additional mount options depending on the type of toolkit/plugin.
2. "port=2049" is added for all the platforms.

SAP ASE and zppData unmount options	"-f" in most cases. Certain cases, SAP ASE uses "-lf". (Lazy unmount option)

Mount and unmount options subject to change

Please note that the mount and unmount options listed above are subject to change. For example, if Delphix finds that a certain option improves performance, Delphix may add, remove or change options at anytime. Therefore, it is highly recommended to create the sudo profiles using wildcards that allow any number of options.

Sudo file configuration examples for SAP ASE environments

This topic provides sample `sudo` file privilege configurations for using the Delphix Engine with various operating systems and SAP ASE.

Configuring `sudo` access on Solaris for SAP ASE source and target environments

Sudo access to `pargs` on the Solaris operating system is required to discover the arguments of the ASE processes both source and target environments.

Example: Solaris /etc/sudoers entries for a Delphix Source for SAP ASE

```
Defaults:delphix_os !requiretty
delphix_os ALL=NOPASSWD:/usr/bin/pargs
```

On a Solaris target, `sudo` access to `mount` and `umount` is also required.

Example: Solaris /etc/sudoers entries for a Delphix Target for SAP ASE

```
# Delphix issues sudo -l so we need to allow it via listpw. Never set it to always
when using public key authentication
Defaults      listpw=all
User_Alias DELPHIX_USER=delphix_os
Cmnd_Alias DELPHIX_CMDS= \
/usr/sbin/mount, \
/usr/sbin/umount, \
/usr/bin/pargs
DELPHIX_USER ALL=(ALL) NOPASSWD: DELPHIX_CMDS
```

Configuring `sudo` access on Linux for SAP ASE source and target environments

On a Linux target, sudo access to `mount` and `umount` is required.

Example: Linux /etc/sudoers file for a Delphix Target for SAP ASE

```
# Delphix issues sudo -l so we need to allow it via listpw. Never set it to always
when using public key authentication
Defaults      listpw=all

Defaults:delphix_os !requiretty
delphix_os ALL=NOPASSWD: \
/bin/mount, /bin/umount
```

Configuring `sudo` access on AIX for SAP ASE source and target environments

In addition to sudo access to the `mount` and umount commands on AIX target hosts, Delphix also requires `sudo` access to `nfso`. This is required on target hosts for the Delphix Engine to monitor the NFS read write sizes configured on the AIX system. Super-user access level is needed to run the `nfso` command.

Example: AIX /etc/sudoers File for a Delphix Target

```
# Delphix issues sudo -l so we need to allow it via listpw. Never set it to always
when using public key authentication
Defaults      listpw=all
Defaults:delphix_os !requiretty
delphix_os ALL=NOPASSWD: \
/bin/mount, \
/bin/umount, \
/usr/sbin/nfso
```

Examples of limiting `sudo` access for the Delphix OS user

In situations where security requirements prohibit giving the Delphix user root privileges to mount, unmount, make a directory, and remove directory on the global level, it is possible to configure the `sudoers` file to provide these privileges only on specific mount points or from specific Delphix Engines, as shown in these two examples.

ⓘ The Delphix Engine tests its ability to run the `mount` command using `sudo` on the target environment by issuing the `sudo mount` command with no arguments. Many of the examples shown in this topic do not allow that. This causes a warning during environment discovery and monitoring but otherwise does not cause a problem. If your VDB operations succeed, it is safe to ignore this warning.
Similarly, the `ps` or `pargs` the command is used for target environment operations such as initial discovery and refresh.
Some organizations configure the security on the target environments to monitor `sudo` failures and lockout the offending account after some threshold. In those situations, the failure of the sudo commands might cause the **delphix_os** account to become locked. One workaround for this situation is to increase the threshold for locking out the user account. Another option is to modify `/etc/sudoers` to permit the **delphix_os** user to run `ps (pargs), mkdir, rmdir,` and `mount` command without parameters.

⚠ Note that the following examples are for illustrative purposes and the sudo file configuration options are subject to change.

Example 1

This example restricts the **delphix_os** user's use of `sudo` privileges to the directory `/sybase`.

Note that wildcards are allowed for the options on `mount` and `umount` because those commands expect a fixed number of arguments after the options. The option wildcard on the `mount` command also makes it possible to specify the file-system being mounted from the Delphix Engine.

However, wildcards are not acceptable on `mkdir` and `rmdir` because they can have any number of arguments after the options. For those commands, you must specify the exact options (`-p` , `-p -m 755`) used by the Delphix Engine.

Example /etc/sudoers File Configuration on the Target Environment for sudo Privileges on the VDB Mount Directory Only (Linux OS)

```
# Delphix issues sudo -l so we need to allow it via listpw. Never set it to always
when using public key authentication
Defaults      listpw=all

Defaults:delphix_os !requiretty
delphix_os ALL=(root) NOPASSWD: \
/bin/mount  *        /sybase/*, \
/bin/mount "", \
/bin/umount *        /sybase/*, \
/bin/umount          /sybase/*, \
/bin/mkdir -p        /sybase/*, \
/bin/mkdir -p -m 755 /sybase/*, \
/bin/mkdir           /sybase/*, \
/bin/rmdir           /sybase/*, \
/bin/ps
```

Example /etc/sudoers File Configuration on the Source Environment to grant Super-User privileges when running PS

```
# Delphix issues sudo -l so we need to allow it via listpw. Never set it to always
when using public key authentication
Defaults      listpw=all

Defaults:delphix_os !requiretty
delphix_os ALL=(root) NOPASSWD: /bin/ps
```

Example 2

This example restricts the **delphix_os** user's use of `sudo` privileges to the directory /sybase, restricts the mount commands to a specific Delphix Engine hostname and IP, and restricts user-specified options for the `umount` command.

This configuration is more secure, but there is a tradeoff with deployment simplicity. This approach would require a different sudo configuration for targets configured for different Delphix Engine.

Configuring the /etc/sudoers File on the Target Environment for Privileges on the VDB Mount Directory Only, and Allows Mounting only from a Single Server (Linux OS)

```
# Delphix issues sudo -l so we need to allow it via listpw. Never set it to always
when using public key authentication
Defaults      listpw=all

Defaults:delphix_os !requiretty
```

```
delphix_os ALL=(root) NOPASSWD: \
/bin/mount           <delphix-server-name>* /sybase/*, \
/bin/mount *         <delphix-server-name>* /sybase/*, \
/bin/mount           <delphix-server-ip>*   /sybase/*, \
/bin/mount *         <delphix-server-ip>*   /sybase/*, \
/bin/mount "", \
/bin/umount          /sybase/*, \
/bin/umount *        /sybase/*, \
/bin/mkdir [*]       /sybase/*, \
/bin/mkdir           /sybase/*, \
/bin/mkdir -p        /sybase/*, \
/bin/mkdir -p -m 755 /sybase/*, \
/bin/rmdir           /sybase/*, \
/bin/ps
```

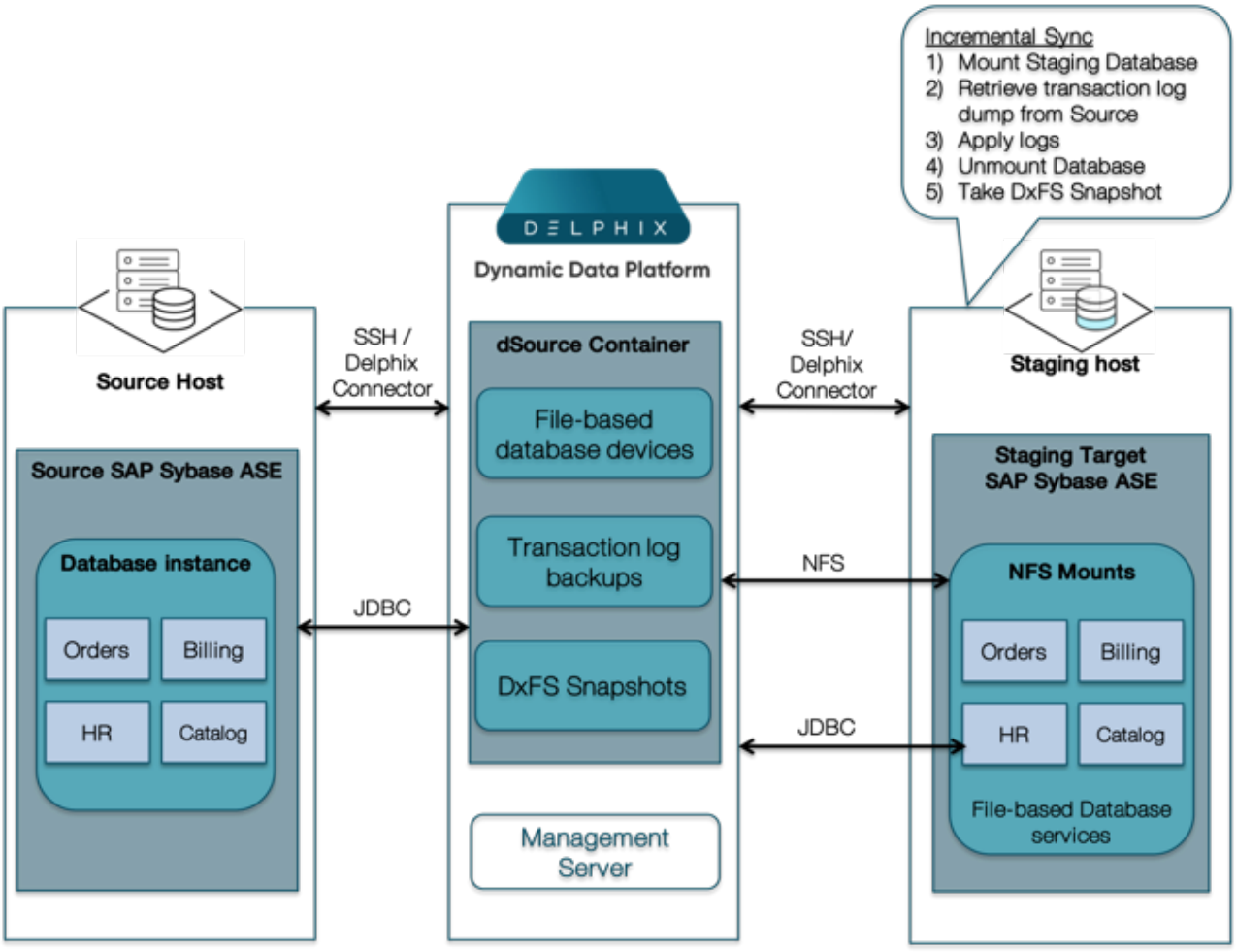
Managing SAP ASE environments and hosts

This section describes the attributes of SAP ASE environments such as information for the Delphix Connector and covers the following topics:

Managing SAP ASE environments overview

This topic describes the high-level process for adding SAP ASE environments, linking SAP ASE databases to the Delphix Engine, and provisioning virtual databases.

Block diagram of linking architecture between SAP ASE environments and the Delphix engine



Environment setup

SAP ASE dSources are backed by a staging database that runs on a target host, as shown in the diagram. There is no requirement for additional local storage on this host, as the storage is mounted over NFS from the Delphix Engine. At Delphix, we refer to the creation and maintenance of this staging database on the staging host as "validated sync," because it prepares the dSource data on the Delphix Engine for provisioning VDBs later on. After the Delphix Engine creates the staging database, it continuously monitors the source database for new transaction log dumps. When it detects a new transaction log dump, it loads that dump to the staging database. The result is a TimeFlow with consistent points from which you can provision a virtual database (VDB), and a faster provisioning process because there is no need for any database recovery during provisioning.

When you later provision a VDB, you can specify any environment as a target, including the environment that contains the staging database. However, for best performance, Delphix recommends that you choose a different

target environment. The target must have an operating system that is compatible with the one running on the validated host, as described in Requirements for SAP ASE Environments and Databases

Workflow and tasks for SAP ASE environments

1. Add the desired source environments as described in Managing SAP ASE Environments
2. Add the desired target environments as described in Managing SAP ASE Environments
3. Link the source database as described in Linking Data Sources with SAP ASE
4. Provision VDBs as described in Provisioning an SAP ASE VDB

> Best Practice
> If possible, it is recommended to use separate SAP ASE instances for staging databases (dSources) and VDBs on target environments. This allows better control on target environments based on the resources required for SAP ASE full/transaction log dump ingestion vs VDB performance.

Adding an SAP ASE environment

Prerequisites

See Requirements for SAP ASE Environments and Databases

Procedure

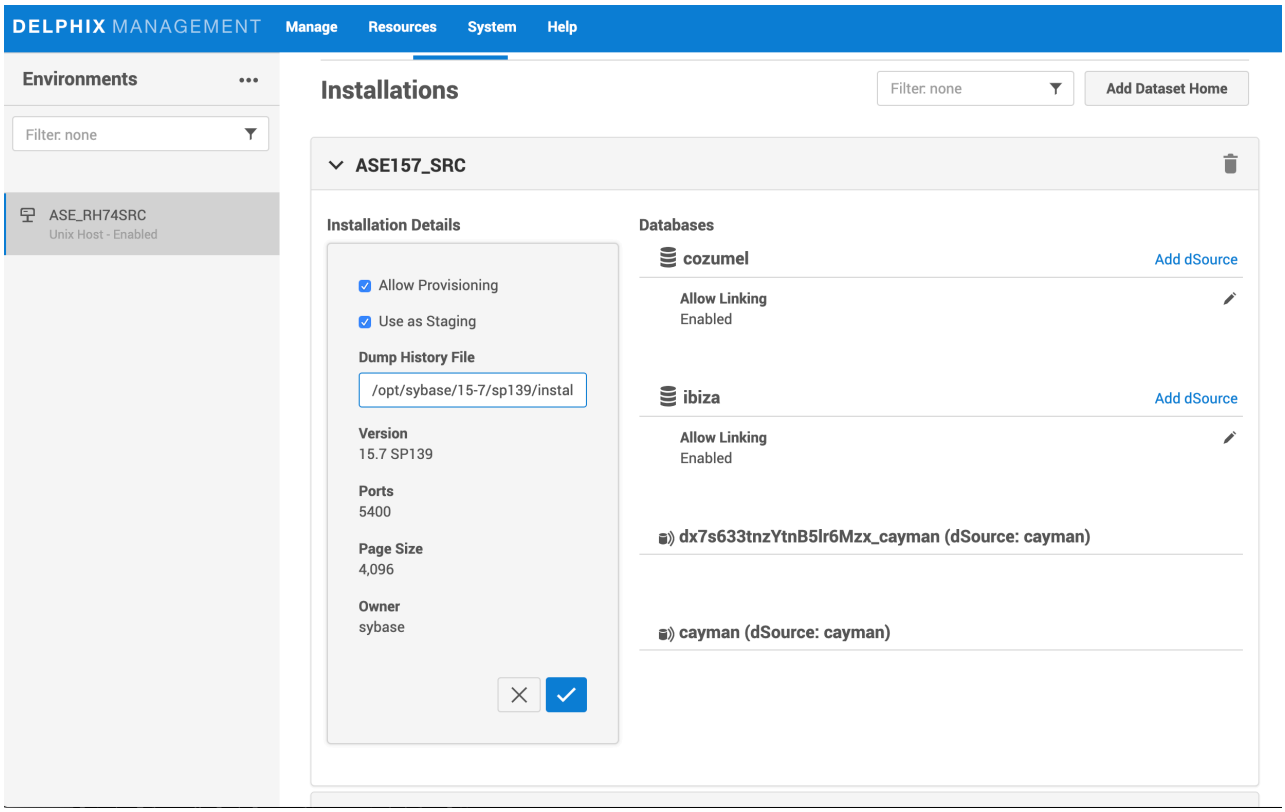
1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. From the **Actions (...)** menu select **Add Environment**.
5. In the **Host and Server** tab window, select **Unix/Linux**.
6. Select **Standalone Host.**
7. Click **Next.**
8. Enter a **Name** for the environment.
9. Enter the **Host IP** address.
10. For NFS Addresses (Optional): Enter one or more comma-separated **IP Address/Hostname**
    Note: If specified, Delphix Engine only allows NFS requests (mount, etc) originated from IP Addresses specified for the host.
11. Enter the **SSH** port. The default value is **22**.
12. Select a **Login Type**. — Username and Password - enter the OS username and password — Username and Public Key - enter the OS username. — Password Vault - select from an existing Enterprise Password Vault
    Note:
    Using Public Key Authentication
    If you want to use public-key authentication for logging into your Unix-based environment, there are two options: use the engine's key pair or provide a key pair for this environment.
    To use the engine's key pair:
    a. Select **Public Key** for the **Login Type.**
    b. Click **View Public Key**.
    c. Copy the public key that is displayed, and append it to the end of your `~/.ssh/authorized_keys` file. If this file does not exist, you will need to create it.
        i. Run chmod 600 ~/.ssh/authorized_keys to allow only the file's owner to read and write to it (make sure the file is owned by the user).
        ii. Run chmod 755 ~ to restrict access to the user's home directory so no other user may write to it.
        iii. Run chmod 700 ~/.ssh so that others cannot write to it. The ~/.ssh directory cannot be writable by group or other users. Otherwise, authentication will fail.
    As an alternative, you can provide a key pair specific for this environment via the API or CLI. See Option 2 in this CLI Cookbook article for instructions.
13. For **Password Login**, click **Verify Credentials** to test the username and password.
14. Enter a **Toolkit Path**. The toolkit directory stores scripts used for Delphix Engine operations. It must have a persistent working directory rather than a temporary one. The toolkit directory will have a separate subdirectory for each database instance. The toolkit path must have 0770 permissions.
15. To provide your own Oracle Java select the **Provide my own JDK** checkbox and click **Next**.
16. In the Java Development Kit tab enter the absolute path to your Oracle JDK and click **Next**.
17. Click the **Discover SAP ASE** checkbox.
18. Click **Next**.
19. In the Summary, tab confirm your selections.
20. Click **Submit**.

Post-requisites

After you create the environment, you can view information about it by selecting **Manage > Environments** and then selecting the **environment name**.

Configuring a non-default dump history file

Delphix, by default, uses the Dump History file that is configured for the source ASE instance. Delphix will query the source ASE instance to find this file name and use this file to find dump history. However, users can specify any other file on the source host to be used for querying the backup history. In that case, Delphix will not use the file configured for the source ASE instance but will use the file specified to get backup information. To specify a custom Dump History file, go to the **source environment > Databases**. Click the pencil button in front of **Installation Details** for the desired instance. Enter the fully qualified name of the Dump History file to be used in the text box next to **Dump History File**.



Delphix does not support rolled over Dump History Files.

Environment attributes for hosts with SAP ASE

This topic describes the attributes of SAP ASE environments. Below you will see a section for common environment attributes shared by all types of environments as well as SAP ASE specific ones.

Procedure

1. From your Delphix Engine, click **Manage**.
2. Select **Environments**.
3. In the **Environments** panel, click the name of an environment to view its attributes.
4. Next to **Attributes**, click the Pencil icon to edit an attribute.

Common environment attributes

Attribute	Description
Environment Users	The users for that environment. These are the users who have permission to ssh into an environment or access the environment through the Delphix Connector. For more information on the environment user requirements, see the Requirements topics for specific data platforms.
Host Address	The IP address of the environment host.
DSP KeyStore Path	The path to the user-managed DSP Keystore.
DSP KeyStore Alias	The lowercase alias to use inside the user-managed DSP Keystore.
DSP KeyStore Password	The password for the user-managed DSP Keystore.
DSP TrustStore Path	The path to the user managed DSP truststore.
DSP TrustStore Password	The password for the user managed DSP truststore.
OS	The name of the host operating system.
Version	The version of the host operating system.
Release	The release of the host operating system.
Time Zone	The timezone of the host operating system.
Total RAM	The amount of RAM on the host machine.

Attribute	Description
Processor Type	The processor type of the host machine.
Traceroute	Traceroute info from target host to Delphix Engine.
Notes	Any other information you want to add about the environment.
Java Development Kit	The currently selected JDK kit will be shown.
Java Development Kit (JDK) Path	Location of the Java Development Kit (JDK) used for the host. Only specified if the feature to provide your own JDK is enabled, otherwise, the defaults are used per our Java Support Matrix

SAP ASE environment attributes

Attribute	Description
Database User	User for Delphix to use for SAP ASE database operations.
DB Password	Credentials to use for the database user in this environment.

How to discover SAP ASE instances which use multiple network handlers

When adding a SAP ASE or Sybase ASE environment, the Delphix Engine may not be able to discover ASE instances that are not listening on the same network address that the engine used for discovery. In these cases, you may receive the following warning:

```
WARNING: Error during discovery for instance "INSTANCENAME" : "Failed to
connect to instance "INSTANCENAME over JDBC.". Skipping discovery for
instance "INSTANCENAME".
```

Troubleshooting

An ASE instance may have been configured to listen on other network interfaces for one of the following reasons:

- To support the use of clustering/failover technologies such as Veritas Cluster Server
- To support the concept of "virtual hostnames" or "virtual IPs" (VIPs) to provide abstraction for client applications
- To facilitate a previous database migration to new infrastructure
- Database administrators configured it manually for other reasons

For guidance and best practices for configuring the Delphix Engine to work with clustering or failover technologies, contact your Professional Services or Customer Success representative.

To verify whether an ASE instance is using multiple network interfaces, use the `sp_listener` stored procedure when connected to the instance using `isql`:

```
1> sp_listener status 2> go  proto host port status
------------------------------------------------------------------------  tcp
secondary-hostname-or-ip-address 4559 active
```

As described in the ASE document Configuring the Server for Multiple Network Handlers, this configuration is read from the `$SYBASE/interfaces` file on your ASE server during instance startup.

Resolution

To allow discovery of instances using multiple network interfaces, use the **Add Environment** dialog to add the environment ONCE per network interface.

Each environment added in this way should have both of the following:

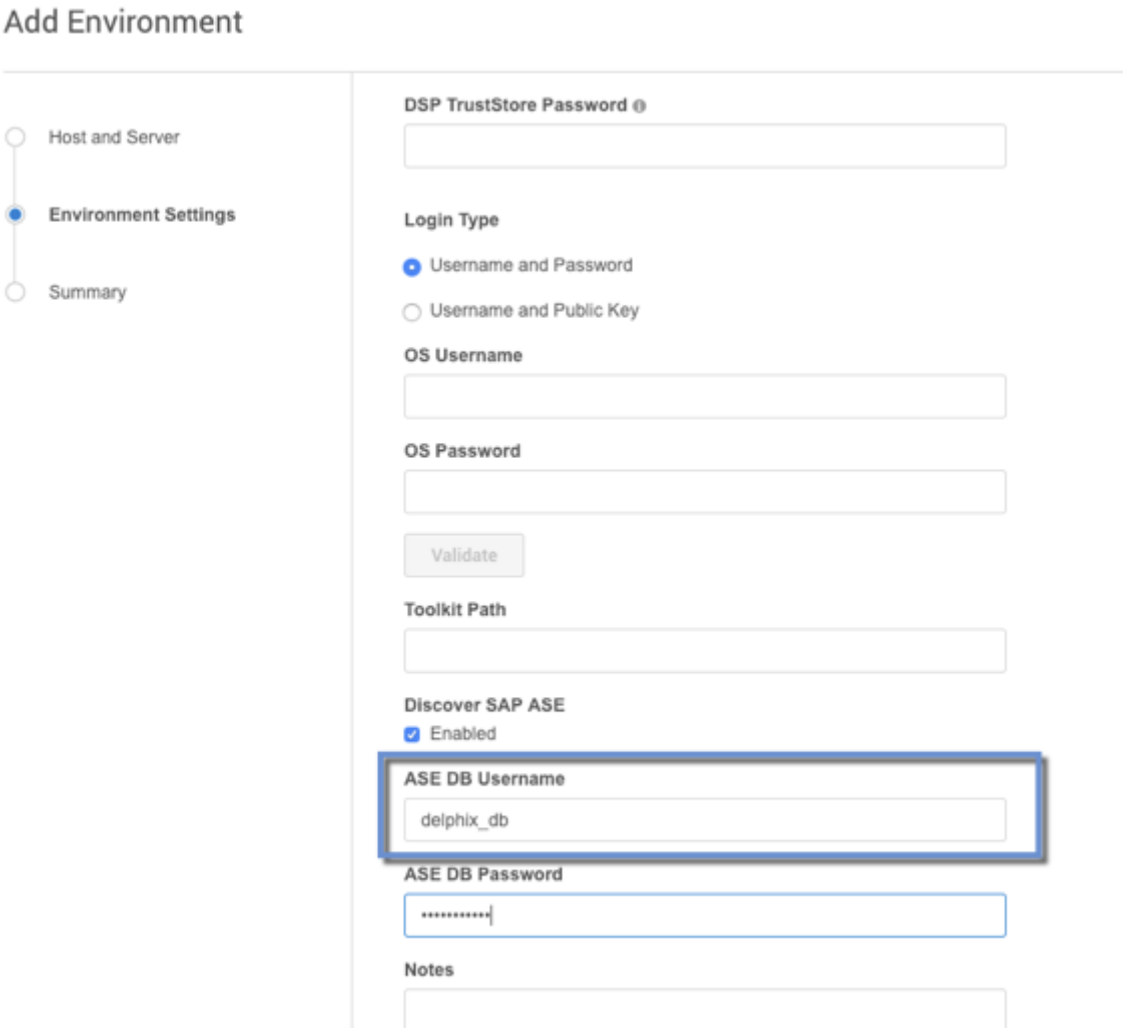
- A unique Host Address; and
- A unique Toolkit Path

The Delphix Engine expects that the files deployed to the Delphix Toolkit directory are persistent. Any failover of the ASE service to a different host must ensure that the Delphix Toolkit directory remains consistent between both hosts, with all files and permissions intact.

Deploying the Delphix Toolkit multiple times will result in increased disk space consumption. It may circumvent concurrency limitations that the Delphix Engine introduces to minimize its impact on your source environments. If your environment requires you to add the same host several times using this process, contact your Delphix

Professional Services representative or Customer Success Manager to discuss best practices or alternative solutions.

Using the example interfaces file provided above, you could use the following inputs to add the `INSTANCENAME` instance:



Changing the host name or IP address of an SAP ASE environment

This topic describes how to change the hostname or IP address for source and target environments, and for the Delphix Engine.

Procedure

For source environments

1. Disable the dSource as described in Enabling and Disabling SAP ASE dSources
2. If the **Host Address** field contains an IP address, edit the IP address.
3. If the **Host Address** field contains a hostname, update your Domain Name Server to associate the new IP address to the hostname. The Delphix Engine will automatically detect the change within a few minutes.
4. In the **Environments** screen of the Delphix Engine, refresh the host.
5. Enable the dSource.

For VDB target environments

1. Disable the VDB as described in Enabling and Disabling SAP ASE dSources
2. If the **Host Address** field contains an IP address, edit the IP address.
3. If the **Host Address** field contains a hostname, update your Domain Name Server to associate the new IP address to the hostname. The Delphix Engine will automatically detect the change within a few minutes.
4. In the **Environments** screen of the Delphix Engine, refresh the host.
5. Enable the VDB.

For the Delphix engine

1. To stop running your VDB select the red **Stop** button located on the VDB **Configuration** tab.
2. Disable all dSources as described in Enabling and Disabling SAP ASE dSources
3. You can use either the command-line interface or the Delphix Setup application to change the IP address of the Delphix Engine.
    a. To use the command-line interface, press **F2** and follow the instructions described in Setting Up Network Access to the Delphix Engine
    b. To use the Delphix Setup application, go to **Delphix Management > Engine Setup** in the Delphix Management application, or click **Server Setup**in the Delphix Engine login screen.
        i. In the **Network** panel, click **Modify**.
        ii. Under **DNS Services**, enter the new IP address.
        iii. Click **OK**.
4. Refresh all Environments by clicking the **Refresh** symbol on the **Environments** screen.
5. Enable all dSources as described in Enabling and Disabling SAP ASE dSources
6. To start all VDBs, click the **Start** button located on the VDB **Configuration** tab.

Using HostChecker to validate SAP ASE source and target environments

What is HostChecker?

The HostChecker is a standalone program which validates that host machines are configured correctly before the Delphix Engine uses them as data sources and provision targets.

Please note that HostChecker does not communicate changes made to hosts back to the Delphix Engine. If you reconfigure a host, you must refresh the host in the Delphix Engine in order for it to detect your changes.

You can run the tests contained in the HostChecker individually, or all at once. You must run these tests on both the source and target hosts to verify their configurations. As the tests run, you will either see validation messages that the test has completed successfully, or error messages directing you to make changes to the host configuration.

Prerequisites

- Make sure that your source and target environments meet the requirements specified in SAP ASE Support and Requirements

Procedure

1. Download the HostChecker tarball matching the host's operating system from https://download.delphix.com/(for example: hostchecker_linux_x86.tar).
2. Create a working directory and extract the HostChecker files from the HostChecker tarball.

```
mkdir dlpx-host-checker
cd dlpx-host-checker/
tar -xf hostchecker_linux_x86.tar
```

3. Change to the working directory and enter this command. Note that for the target environments, you would change `source` to `target`.

```
$ cd hostchecker
$ ./hostchecker.sh
Extracting the JDK from the tarball jdk.tar.gz.
Please enter whether this machine is a source or a target:target
1: Check ASE environment
2: Check all ASE instances
3: Check all the Oracle installations
4: Check homedir permissions
5: Check Linux Performance Settings
6: Check mkdir and rmdir
7: Check the MySQL installation
8: Check network port access
9: Check the Oracle CRS home
10: Check for ssh connectivity
11: Check sshd_config for timeout configuration
12: Check user sudo privileges
13: Check toolkit path
all: Execute all checks
quit: Exits
```

```
Please select an option:
```

Note:
Don't Run as Root
Don't run the HostChecker as root; this will cause misleading or incorrect results from many of the checks.

4. Select which checks you want to run. We recommend you run all checks (excluding Oracle and MySQL) if you are running Hostchecker for the first time.
5. Pass in the arguments the checks ask for.
6. Read the output of the check.
7. The error or warning messages will explain any possible problems and how to address them. Resolve the issues that the HostChecker describes. Don't be surprised or undo your work if more errors appear the next time you run HostChecker, because the error you just fixed may have been masking other problems.
8. Repeat steps 3–7 until all the checks return no errors or warnings.

Tests run

Test	SAP ASE Source	SAP ASE Target	Description
Check SAP ASE environment	X	X	• Checks that the **$SYBASE** environment variable exists for interactive logs. **NOTE**: It is still necessary to check that it is defined for non-interactive logins (ssh user@host env \| grep SYBASE). • Checks that the "**isql_r64**" binary can be found underneath the **$SYBASE** environment variable.
Check All SAP ASE instances		X	Attempts to connect to each running SAP ASE instance via "**isql_r64**" (utilizing the $SYBASE/interfaces file) and execute the following queries: select @@servername select count(*) from master..syslisteners select count(*) from master..sysservers select count(*) from master..sysdatabases select count(*) from master..sysusages select @@version select @@maxpagesize SELECT srvnetname FROM master..sysservers WHERE srvname='SYB_BACKUP'
Check all the Oracle installations	N/A	N/A	

Test	SAP ASE Source	SAP ASE Target	Description
Check homedir permissions	X	X	Check that the home directory, the ~/.ssh directory, and the ~/.ssh/authorized_keys file exist, that they are owned by the user invoking this check, and that they are not 'group' or 'other' writable.
Check performance="">>		X	Check target's kernel settings necessary to optimize performance.
Check mkdir and rmdir		X	Tests that the user can mkdir and rmdir under both the toolkit directory and the specified mount path.
Check the MySQL installation	N/A	N/A	
Check network port access	X	X	Can be used to test access to specified ports on the Delphix Engine. See Network and Connectivity Requirements for SAP ASE Environments for a list of ports.
Check the Oracle CRS home	N/A	N/A	
Check Oracle DB Instance	N/A	N/A	
Check for ssh connectivity	X	X	Verifies that the environment is accessible via SSH
Check sshd_config for timeout configuration		X	Check that sshd_config does not contain ClientAliveInterval or ClientAliveCountMax entries.
Check toolkit Path	X	X	Verifies that the toolkit installation location has the proper ownership, proper permissions, and enough free space.

Test	SAP ASE Source	SAP ASE Target	Description
Check user sudo privileges	X	X	Verifies that the operating system user can execute certain commands with necessary privileges via `sudo`. This only needs to be run on target environments. See the topic Requirements for SAP ASE Environments and Databases for more information.

Enabling linking and provisioning for SAP ASE environments

This topic describes how to enable and disable provisioning and linking for SAP ASE databases.

Before a database can be used as a dSource, you must first make sure that you have enabled linking to it. Similarly, before you can provision a VDB to a target database, you must make sure that you have enabled provisioning to it.

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Select the **Databases** tab.
5. Click the **pencil** icon located next to the database **Installation Details**.
6. Select the **Allow Provisioning** checkbox to enable provisioning, deselect the checkbox to disable provisioning.
7. Click **show details** for the database.
8. Slide the button next to **Allow Linking** to **On** or **Off** to enable or disable linking.

Linking data sources and Syncing Data with SAP ASE

Creating a dSource will ingest data from the source and create a dSource on the engine. The dSource is an object that the Continuous Data Engine uses to create and update virtual copies of your database. As a virtualized representation of your source data, it cannot be managed, manipulated, or examined by database tools.

This section covers the following topics:

Linking SAP ASE data sources: an overview

This topic describes basic concepts behind the creation of dSources from SAP ASE databases.

Initial linking and staging databases

A dSource is a copy of a physical database that is created when the Delphix Engine links to and loads the database from a backup. The database backup can be a new full database backup that the Delphix Engine initiates, the most recent existing database backup, or an existing database backup specified by the user. When loading from an existing backup, the backup should be in a location that the source environment user can access.

After obtaining the initial snapshot and linking the dSource, the Delphix Engine keeps the dSource and the source database in sync by monitoring the source database for new transaction log dumps and then applying those backups on a standby database. This database is called the "staging database." A target environment that hosts one or more staging databases is referred to as a "staging target."

After you have linked a database into the Delphix Engine, you can re-initialize it by performing a sync on the dSource.

> ⚠ There is an SAP ASE feature that affects the size of database dumps which correspondingly affect the amount of space consumed on the storage attached to the Delphix Engine. If the SAP ASE "sp_dumpoptimize" feature is set to maximum as follows, more disk space will be consumed on Delphix storage because SAP ASE writes both allocated and unallocated pages into the dump files:

```
sp_dumpoptimize 'archivespace = maximum'
```

SAP ASE source and staging database compatibility

You may use different SAP ASE patch level versions between source and staging hosts. The major version of SAP ASE must still be the same. The only caveat is for SAP ASE 15.7 - where the source and staging patch levels must both be either any version below SP64 or both be any version above SP100.

Linking data sources with SAP ASE

The dSource is an object that the Continuous Data Engine uses to create and update virtual copies of your database. As a virtualized representation of your source data, it cannot be managed, manipulated, or examined by database tools. For an overview of all dSource related actions, please visit Managing Data Sources.

Continuous Data for SAP ASE databases leverages backup-based ingestion, which means that Delphix will look for, or sometimes initiate the creation of, a backup through your SAP ASE backup server. From there, the backup is restored on a staging server and the staging copy is then ingested into Delphix. See Delphix Architecture with SAP ASE for more information.

When linking a dSource from an SAP ASE source database, Delphix offers several different methods of capturing backup information:

- ASE Managed Backups, where the SAP ASE source database schedules and initiates backups. This method supports various backup types which include:
  - Full backups
  - Transaction log backups (with LogSync disabled)
  - Transaction log backups (with LogSync enabled)
- Delphix Managed Backups, where the Delphix Engine schedules and initiates the backups from the source database, and captures them.

ASE managed backups

Further contextual information on the various backup types (listed above):

- Full Backups - A snapshot will be created on the Delphix Timeflow for each Full backup.
- Transaction log backups (with LogSync disabled) - A snapshot will be created on the Delphix Timeflow for each transaction log backup.

> **Transaction logs**
> Transaction logs are not collected if:
> - a) there are gaps in the sequence of log backups (a break in the “log chain”).
> - b) the available log backups do not include any changes since the last successful Delphix snapshot.

- Transaction log backups (with LogSync enabled) - A snapshot will be created on the Delphix Timeflow for each transaction log backup. In addition, point-in-time provisioning will be an available option if you would like to provision from any point in between snapshots.

> **Log files**
> Log files consume additional space on the Delphix Engine and are managed according to the defined retention policy for logs.

Delphix managed backups

When the checkbox for Delphix Managed Backups is selected, the Delphix Engine will initiate a full backup of the source database for the initial load of the dSource. Thereafter, the Delphix Engine will initiate full backups of the source database using the schedule specified by the selected SnapSync Policy. If you select the None policy, the Delphix Engine will not automatically initiate a full backup, but you can initiate them manually using the snapshot (camera) icon.

Delphix looks for the backup files on the staging host (unless a "remote" backup server is used in which case, the remote host is used which is often the source environment)

Procedure

1. Login to the **Delphix Management** application.
2. Navigate to **Manage > Datasets.**
3. Click the plus icon and select **Add dSource**.
   Note:
   Delphix looks for the backup files on the staging host (unless a "remote" backup server is used in which case, the remote host is used which is often the source environment.
4. In the **Add dSource** wizard, select the source environment with the correct environment-based user.
5. Enter your login credentials for the source database and click **Next**.
6. Enter a name and select a group for your dSource. Adding a dSource to a dataset group lets you set Delphix Domain user permissions for that database and its objects, such as snapshots. See the topics under Users and Groups for more information.
7. Select the **Data Management** settings needed, as described in Data Management Settings for ASE Data Sources.
8. From the **Data Management** tab under the **Initial Load** option, select and enter any **additional settings** needed. There are three options for the initial load of the dSource:
   a. If the source ASE instance resides on the same server as the staging ASE instance, the staging database's NFS mounted "temp" directory will be present for the source database to write to in response to the "DUMP DATABASE" command.
   b. If the source and staging ASE instances are configured to allow remote access to the backup servers and the dSource is linked using the Remote Server option as described below. Delphix will then issue the "DUMP DATABASE" command and append the "AT <staging_backup_server_name>" clause so that the dump is written to the staging backup server.
   c. (Recommended) **New Full Backup** - Lets Delphix create a new full backup file and load it. Note - that when Delphix creates the backup, it is moved to Delphix's NFS-mounted storage located on the stage host rather than. The backup will be located in the "temp" directory and will be deleted once the Delphix Engine has restored the backup and created a dSource from the restored staging database. This means that this option will work under two scenarios:
   d. Most Recent **Existing Full Backu**p – Find the most recent existing full backup file in the Backup Location and load it.
      Note:
      If Dump History is not active on the Source Database: Choosing this option can delay completion of the dSource link as Delphix attempts to find and catalog every single backup listed in the source database's backup server log file.
   e. Specific **Existing Full Backu**p – Specify which backup files in the Backup Location you want to load. Choosing this option is much faster because Delphix will skip directly to loading the desired backup and only start to search for and catalog backups in the background after the linking of the source database has completed.
      Note:
      When using a dump taken with the deprecated compression syntax, select the Specific Existing Full Backup option for Initial Load and, for each stripe, type compress::<file name> into the text box.
9. Select the Staging environment and ASE instance that will be used to manage the staging database used for validated sync of the dSource.
10. Select any policies for the new dSource.
11. Click **Next**, then specify any pre-hook and post-hook scripts.
12. Review the dSource Configuration and Data Management information, and click **Submit** to begin provisioning the VDB.

Linking data sources with encrypted SAP ASE database

This topic describes the behavior of the Delphix Engine when linking to a dSource based on an encrypted SAP ASE database.

Beginning with version 16.0, SAP ASE supports the Database Encryption feature. You can link the encrypted source databases to a dSource by following the basic procedure described in Linking Data Sources with SAP ASE but the user needs to ensure that the same key with the same name is present on the staging instance as it was on the source instance. The keys should be imported from the source instance. Refer to the configuration steps 4 and 5 of Delphix Implementation of Database Encryption to export and import the master and encryption key.

- If a user links a database when it is unencrypted and then encrypts it, the subsequent syncs will fail.
- If a user links an encrypted database and then decrypts the source database, the subsequent syncs will fail.
- If a user changes the encryption key, the subsequent syncs will fail.

In the case of the above failures, the user will be prompted to apply a full backup with the “Drop and Recreate Devices” option enabled (for once), and subsequent snap and validate sync will run without any failures. See Working with Snapshots

Exceptions with suggested actions are raised in the following scenarios:

- If there has been a change in the encryption status of a source database since it was last synced or the key has changed.
- If the required encryption key is not present on the staging instance during the linking.

Data management settings for SAP ASE dSources

Each dSource has its own data management settings, which can be configured during the linking workflow as well as on the configuration page for that dSource. You can configure data management settings to improve overall performance and meet your requirements.

The following settings are available for SAP ASE data sources:

Setting	Explanation
Backup Path	The directory where Delphix will search for backups for data ingestion. Delphix recursively searches this location, so the database backups or transaction logs can reside in any subdirectories below the path entered.
Validated Sync Mode	ASE Validated Sync is either enabled or disabled. Delphix stays in sync with source databases by monitoring the ASE backup server log or dump history file (depending on the dSource configuration). When it sees a new database dump or transaction log has been created, it attempts to load it into the ASE staging instance. • Enabled - Delphix automatically ingests either transaction logs or full database dumps depending on the source database's configuration. • Disabled - Delphix does not actively monitor the source database's backup server log or dump history file and does not automatically ingest backups.
Dump History	Delphix Validated Sync will leverage Dump History files to locate and ingest backups. This setting is recommended to improve the data ingestion and syncing processes.
LogSync	When enabled, Delphix copies the transaction logs to Delphix storage to allow precise point-in-time provisioning. **Note**: If Dump History is not enabled, due to SAP ASE CR 800569, Delphix can only support transaction logs generated in intervals greater than one minute apart for SAP ASE versions 16.0 SP02 through SAP ASE 16.0 SP02 PL04. This ASE bug inadvertently removed the second and millisecond precision from the dump header sequence dates preventing Delphix from knowing what order to apply the transaction logs when there are multiple transaction logs dumped within the same minute.
Source of Production Backup	You may select a staging or remote server as the location for the Delphix Engine to ingest backups. **Note** - Selecting a staging server is the most common application for ingesting source-based backups. One major reason for this is the ability to use Validated Sync to coordinate backup ingestion.
Dump Password	SAP ASE has an optional field for database dumps (backups) to prevent unauthorized loads. If enabled, Delphix will need to store the dump password in order to restore the SAP ASE backup for ingestion. Delphix also provides the option to propagate password protection to established dSources in the case of an active password on the originating database dump or transaction log dump files. Setting this option causes Delphix to add the "WITH passwd=" clause to the "LOAD" commands.

Procedure

1. Enter the **Backup location**. This is the directory where the database backups are stored. Delphix recursively searches this location, so the database backups or transaction logs can reside in any subdirectories below the path entered.
2. Select the **Staging environment** and ASE instance name.
3. Enable or disable Validated Sync Mode. Validated Sync Mode (also known as ValidatedSync) is a background process that looks for new full or transaction log backups either by monitoring ASE Backup Server's log file or Dump History file for a new database or transaction log dumps. When Delphix detects a new dump is available it attempts to load it into the staging ASE database.
4. Enable or disable **Use dump history**. If Dump History is enabled, Delphix uses the ASE Dump History file to find information about backups being performed on the source database and to find the next backup to be used for Validated Sync.
5. Enable or disable **LogSync**. LogSync copies the transaction logs to Delphix storage which enables provisioning VDBs from a specific point-in-time in rather than just a particular backup
   Note:
   LogSync support limitations
   [If Dump History is not active on the Source Database] Due to ASE CR 800569, Delphix can only support transaction logs generated in intervals greater than one minute apart in ASE versions 16.0 SP02 through ASE 16.0 SP02 PL04. This ASE bug inadvertently removed the second and millisecond precision from the dump header sequence dates preventing Delphix from knowing what order to apply the transaction logs in when there are multiple transaction logs dumped within the same minute.
6. Select **Backup location type**.
7. Click **Advanced**to edit Source of Production Dump, External Data Directory, Retention policies, or Dump Password.
   Note:
   External Data Directory
   The External Data Directory feature is not currently used with ASE dSources and is targeted for removal in a future release of Delphix.
   Remote Server should be selected when database dumps cannot be found on the Staging Environment. This option can be used with any of the initial load selections (New Full Backup, Most Recent Existing Full Backup or Specific Existing Full Backup). If selected, fill out additional settings as needed:
   a. Enter the Remote Server Name. This is the name of the backup server used when the dump was created.
   b. Select the Remote Host and Remote User that the backup server is located on.
   c. As noted, the interfaces file on both the staging and remote environments must be modified to point at each other's backup servers.
   d. The Create Dump Password sets a dump password for the dSource. Select this only if the dump password option was used to create a password on the database dump or transaction log dump files. Setting this option causes Delphix to add the "WITH passwd=" clause to the "LOAD" commands.

Upgrading a SAP ASE dSource

This topic describes how to upgrade SAP ASE dSources.

Prerequisites

- The staging SAP ASE instance must be the same version as the SAP ASE instance hosting the upgraded source database.

Procedure

1. Login to the **Delphix Management** application.
2. In the top menu bar, click **Manage** and select **Datasets**.
3. Select the **dSource** to be upgraded.
4. Disable the dSource.
    a. From the **Actions (...) menu** select **Disable**.
    b. In the Disable dialog select **Disable**.
5. Upgrade the source and staging ASE instances (Delphix requires the source and staging ASE instances to be at the same major release level).
6. Click the **Refresh** icon (under the **Manage > Environments** menu) to refresh the Delphix environment hosting the source and staging ASE instances so that Delphix registers the new version of ASE.
7. Enable the dSource.
    a. From the **Actions (...) menu** select **Enable**.
    b. In the Enable dialog select **Enable**.

Additional steps

If Delphix was unable to discover the SAP ASE instances automatically using Manual Discovery, you will need to manually update the version by completing the following procedure, CLI Cookbook: Configuring SAP ASE Manual Discovery

Detaching and Re-attaching SAP ASE dSources

This topic describes how to detach dSources and re-attach them to a different source database.

Each dSource contains metadata that associates it with the source database, as well as the data it has ingested from the source database in the form of snapshots up to that point. It is possible to detach, or unlink, a dSource from its source database. This breaks the association with the source database without affecting the data within the Delphix Engine. Detached dSources and their source databases have these properties:

- A detached dSources can still be used to provision a virtual database (VDB).
- You can re-link the source database as a different dSource.
- Any child VDBs that were provisioned from this dSource will only be able to be refreshed from the most recent snapshot available on the dSource.
- If you need a VDB from a newer snapshot, you would need to provision a new VDB. Once you have provisioned the new VDB you can delete the old VDBs provisioned from this dSource. You can delete the old dSource when it is no longer needed.

Detaching a dSource

1. Login to the **Delphix Management** application as a user with **OWNER** privileges on the dSource, group, or domain.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **Dataset** you want to unlink.
5. From the **Actions** menu (...) select **Unlink dSource.** A warning message will appear.
6. Click **Unlink** to confirm. The status of the dSource will show as Detached.

Rebuilding source databases and using VDBs

In situations where you want to rebuild a source database but retain the existing dSource, you will need to detach the original dSource and create a new one from the rebuilt data source.

1. Detach the dSource as described in the procedure on this page.
2. You cannot attach a dSource with the same name as a dSource that is already attached. If you intend to give the new dSource the same name as the original one, rename the detached dSource.
    a. At the top of the **Configuration** tab, next to the dSource's name, click the **Edit** (pencil) icon.
    b. After renaming the dSource, click the **checkmark**.
3. Create the new dSource from the rebuilt database.

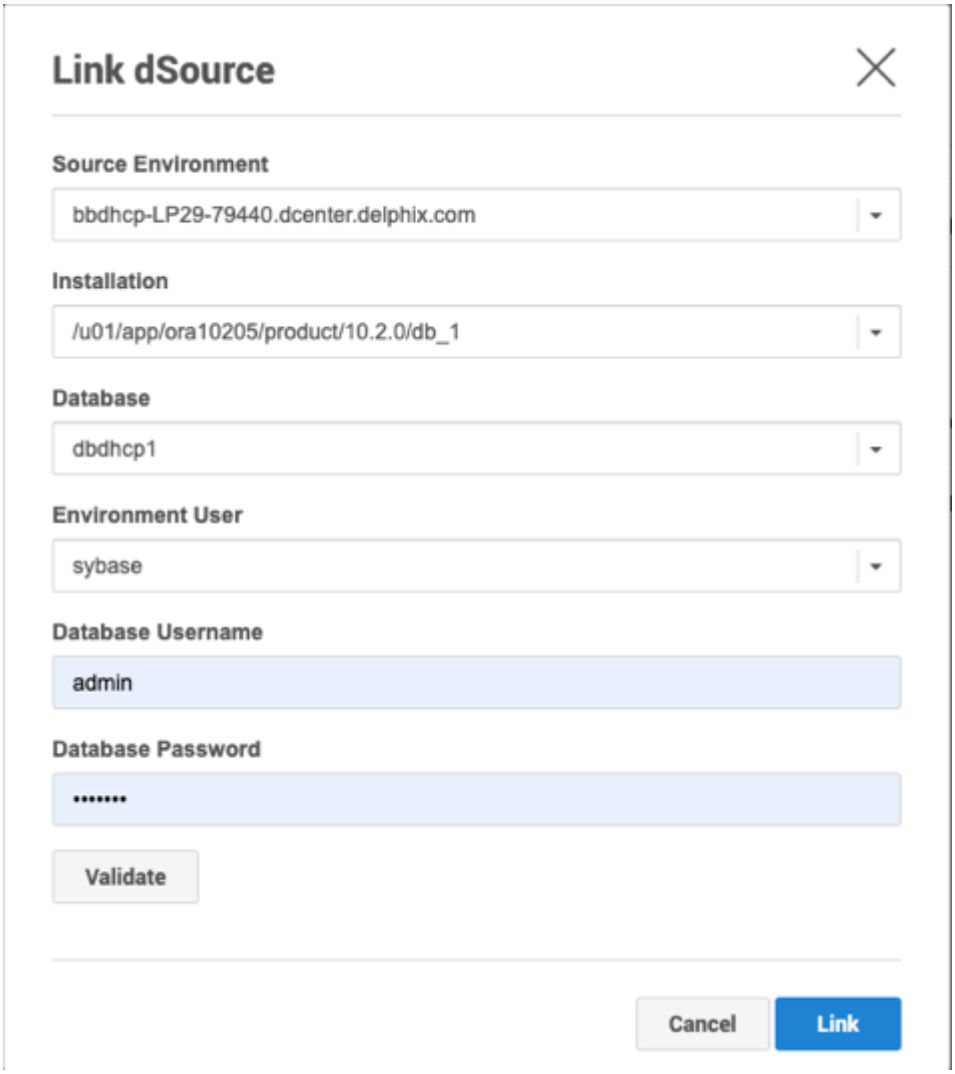
You will now be able to provision VDBs from both the detached dSource and the newly created one, but the detached dSource will only represent the state of the source database prior to being detached.

Attaching a previously detached dSource

The attach operation is supported via the UI. You can only re-attach databases that represent the same physical database.

Attaching a dsource via the UI:

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **Dataset** you want to link.
5. From the **Actions** menu (...) select **Link dSource.**
6. In the Link dSource window click **Link** to confirm.


Link dSource
Source Environment
bbdhcp-LP29-79440.dcenter.delphix.com
Installation
/u01/app/ora10205/product/10.2.0/db_1
Database
dbdhcp1
Environment User
sybase
Database Username
admin
Database Password
•••••••
Validate
Cancel
Link

Deleting an SAP ASE dSource

This topic describes how to delete an SAP ASE dSource.

Deleting a dSource will delete the dSource metadata for a particular source database, along with all snapshots, logs, and policies stored in Delphix. This is a permanent operation, and re-attaching the dSource will require a new linking operation. If a dSource is deleted, it does not affect your source database.

If you wish to temporarily disable your dSource without deleting it, you can follow the steps to Enabling and Disabling SAP ASE dSources instead.

Prerequisites

You cannot delete a dSource that has dependent virtual databases (VDBs). Before deleting a dSource, make sure that you have deleted all dependent VDBs as described in Deleting a VDB.

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Click **Datasets**.
4. In the **Datasets** list, select the **dSource** you want to delete.
5. From the **Actions** menu (...) select **Delete.**
6. Click **Delete** to confirm.

Deleting a dSource will also delete all snapshots, logs, and descendant VDB refresh policies for that dSource. You cannot undo the deletion.

Enabling and disabling SAP ASE dSources

This topic describes how to enable and disable dSources when certain operations against the source database must occur outside of Delphix.

Some operations, such as restoring the source database from a backup, will require that the dSource be temporarily disabled. Disabling a dSource turns off communication between it and the source database, but it does not tear down the configuration that enables communication and data updating to take place. When a disabled dSource is later enabled, it will resume communication and incremental data updates from the source database according to the original policies and data management configurations that you set.

Disabling a dSource is also a prerequisite for several other operations, such as database migration and upgrading the dSource metadata after upgrade of the associated data source.

Procedure

Disabling a dSource will stop further operations on the Delphix Engine related to the dSource.

1. Login to the Delphix Management application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **dSource** you want to disable.
5. In the upper right-hand corner, from the **Actions** menu (...) select **Disable.**
6. In the Disable dialog select **Disable**.

When you are ready to enable the dSource again, from the Actions menu (...) select **Enable**, and the dSource will continue to function as it did previously.

Working with SAP ASE snapshots

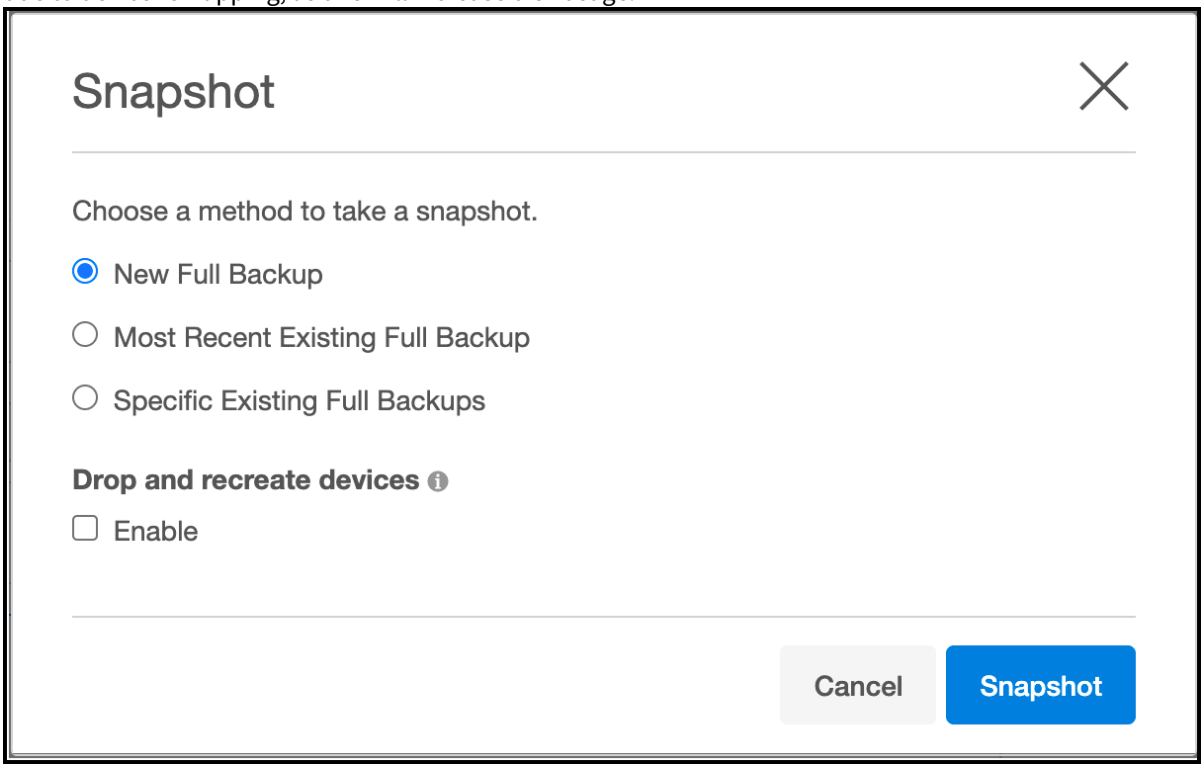
Taking a snapshot creates a new snapshot entry in the dSource's Timeflow.

Snapshot

This section lists the steps to take a snapshot and delete the same.

Login to the **Delphix Management** application.

1. Click **Manage** and select **Datasets** from the dropdown list.
2. Select the dSource you want to Snapshot.
3. Click the **Camera** icon.
4. Select the method to take the snapshot. See Quick Start Guide for SAP ASE for information on Delphix Managed Backups.
   You can optionally select the **Enable** checkbox under **Drop and recreate devices**. This option can be used if the previous sync failed because of device mapping issues. When the sync operation is performed with this option set to true, Delphix Engine will not try to re-map the existing devices; instead, it will delete them and create new devices on the staging database. This option should only be used if the previous sync had failed due to device remapping, as this will increase disk usage.



5. From the Snapshot dialog, click **Snapshot**.
6. Under the **Timeflow** tab, choose an option to view the snapshot and verify the snapshot you just created.
7. To delete, select a snapshot you want to delete, and from the Actions menu (...) select **Delete Snapshot**.
8. Under the **Timeflow** tab, choose an option to view the snapshot and verify the snapshot you just deleted.

Provisioning and managing VDBs from SAP ASE

Virtual databases are a key data management concept for Delphix. In order to create or provision a virtual database, you will need a linked dSource from a source host and a compatible target environment.

From a dSource, you can select a snapshot or point in time to create a VDB. SQL Server VDBs each have their own configuration settings as described in Configuration Settings for ASE Virtual Databases below.

This section covers the following topics:

Overview of provisioning SAP ASE virtual databases

Virtual databases are a key data management concept for Delphix. In order to create or provision a virtual database, you will need a linked dSource from a source host and a compatible target environment, as described in the overview for Managing Environments and Hosts and Requirements for SAP ASE Environments and Databases.

From a dSource, you can select a snapshot or point in time to create a VDB. SQL Server VDBs each have their own configuration settings as described in Configuration Settings for ASE Virtual Databases below.

Procedure

1. In the Datasets panel on the left-hand side, click the group containing the dSource or VDB from which you want to provision.
2. Click the **Timeflow** tab.
3. Select a snapshot or open LogSync timeline to provision by a specific log or point in time.
    - Find more detail about initial provisioning options in the section 'Provisioning by Snapshot or LogSync' below.
4. Click to open the Provision VDB wizard, and select a compatible Target Environment for the new ASE VDB
5. Review the information presented for Target Configuration and edit as necessary.
6. Select a Snapshot Policy for the VDB.
7. (Optional) - Selective Data Distribution - After policies, there is a masking option.
8. Enter any operations that should be run in the Hooks page. These scripts can be managed after provision in the VDB's configuration page.

When provisioning starts, you can review the progress of the job by selecting the VDB and clicking on the Status tab, or by selecting Manage/Dashboards and viewing the Job History panel. Alternatively, you could see this in the Actions Sidebar. When provisioning is complete, the VDB will be included in the group you designated and listed in the Datasets panel. If you select the VDB in the Datasets panel and click the Configuration tab, you can view information about the database and its Data Management settings.

Provisioning by snapshot or LogSync

When provisioning by Snapshot, you can provision to the start of any particular snapshot by time.

Provisioning by snapshot/time	Description
Provisioning By Snapshot	You can provision by using a Snapshot. In that case, a new VDB will be provisioned to the database state as of the Snapshot.
Provision by Time	If you have enabled Log Sync, you can provision a new database to a point in time. You can select a snapshot and then using time entry fields, specify a Point in Time. Delphix will use the selected snapshot to restore the VDB and use the log files to roll forward the VDB to the selected time.

Configuration settings for ASE virtual databases

Each VDB has its own data management settings, found during the provisioning workflow as well as in the configuration page for that VDB. When you create a SAP ASE VDB, Delphix copies most configuration settings from the dSource and uses them to create the VDB. However, you can customize these with the following settings:

Setting	Explanation
Recovery model	The current recovery model of the source database. This field will auto-populate with information from the dSource.
Auto VDB restart	Enabling this option will automatically restart this VDB whenever its target host is rebooted.

Automatic VDB restart on target server sfter reboot

The Delphix platform now automatically detects whether a target server has been rebooted, and proactively restarts any VBD on that server that was previously up and running. This is independent of the data platform. It is done as if you realized a target server was restarted and issued a start command from the Delphix platform. This feature is compatible with Self-Service ordering dependencies and is limited to non-clustered VDBs.

To enable automatic restart, complete the following steps:

- When provisioning a new VDB in the VDB Provisioning wizard, check the **Auto VDB restart** box.

Once the VDB has been provisioned, you will be able to turn **Automatic VDB restart** on.

1. In the **Datasets** panel, select the VDB.
2. Select the **Configuration** tab.
3. Select **Source** sub-tab.
4. Select **Database edit**.

Provisioning an SAP ASE VDB

This topic describes how to provision a virtual database (VDB) from an SAP ASE dSource.

Prerequisites

- You must have already linked a dSource from a source database, as described in Linking Data Sources with SAP ASE, or have already created a VDB from which you want to provision another VDB.
- You must have already set up target environments as described in Adding an SAP ASE Environment.
- Ensure that you have the required privileges on the target environment, as described in Requirements for ASE Environments and Databases.
- If you are provisioning to a target environment that is different from the one in which you set up the staging database, you must make sure that the two environments have compatible operating systems, as described in Requirements for SAP ASE Target Hosts and Databases. For more information on the staging database and the validated sync process, see Managing SAP ASE Environments Overview.

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**
3. Select **Datasets**.
4. Select a **dSource**.
5. Select a **means of provisioning**.
6. Click **Provision**. The **Provision VDB** panel will open, and the **Instance** and **Database Name** fields will auto-populate with information from the dSource.
7. Select whether to enable **Truncate Log on Checkpoint** database option for the VDB.
8. Click **Next**.
9. Select a **Target Group** for the VDB. Click the green **Plus** icon to add a new group, if necessary.
10. Select a **Snapshot Policy** for the VDB. Click the green **Plus** icon to create a new policy, if necessary.
11. Click Auto VDB Restart to enable VDBs to be automatically restarted when staging/target host gets rebooted, if necessary.
12. Specify any **Hooks** to be used during the provisioning process. For more information, see SAP ASE Hook Operations.
13. If your Delphix Engine system administrator has configured the Delphix Engine to communicate with an SMTP server, you will be able to specify one or more people to notify when the provisioning is done. You can choose other Delphix Engine users or enter email addresses.
14. Click **Submit**. When provisioning starts, the VDB will appear in the **Datasets** panel. Select the VDB and navigate to the **Status** tab to see the progress of the job. When provisioning is complete, more information can be seen on the **Configuration** tab.

Provisioning a VDB from an encrypted SAP ASE database

This topic describes how to provision a VDB from an encrypted database.

The Delphix Engine supports provisioning from a dSource linked to a physical database that has been encrypted with SAP ASE Database Encryption, which can be used to encrypt columns or complete databases.

You can provision a VDB from a dSource linked to an encrypted database using the normal process to Provision an SAP ASE VDB but the user needs to ensure that the same key with the same name is present on the target instance as it was on the staging instance because we mount the database onto the target that was created on the staging instance. The keys should be imported from the source instance. Refer to configuration steps 4 and 5 of Delphix Implementation of Database Encryption to export and import the master and encryption key.

Exception with suggested actions is raised in the following scenario:

- If the required encryption key is not present on the target instance during the provisioning.

Provisioning an SAP ASE VDB from a replicated VDB or dSource

This topic describes how to provision from a replicated dSource or virtual database (VDB). The process for provisioning from replicated objects is the same as the typical VDB provisioning process, except that first you need to select the **replica** containing the replicated object.

Prerequisites

- You must have replicated a dSource or a VDB to the target host, as described in Replication Overview
- You must have added a compatible target environment on the target host

Procedure

1. Login to the **Delphix Management** application for the target host.
2. Click **Manage**.
3. Select **Datasets**.
4. In the list of replicas, select the **replica** that contains the dSource or VDB you want to provision.
5. The provisioning process is now identical to the process for provisioning standard objects.

Post-requisites

Once the provisioning job has started, the user interface will automatically display the new VDB in the live system.

Resizing an SAP ASE VDB

When resizing ASE VDBs, we recommend:

- Ensure that the VDB's database devices reside on the Delphix storage, in the same directory as the database's other devices. Use the ASE "sp_helpdb" and "sp_helpdevice" commands to get details on the VDB's devices.
- If you need to add a new device, use the "skip_alloc=true" clause because the storage on Delphix is already zeroed (initialized). This should allow the "DISK INIT" command to complete almost instantaneously.
- If you add a new device to the database by issuing the ASE "DISK INIT" followed by the "ALTER DATABASE" commands, click the camera icon to take a new snapshot of the VDB.
  Note: It is important to take a snapshot after altering the VDB's device layout so that Delphix rewrites the database's "manifest" file to include the new database device.
- Delphix 6.x uses the new "ALLOW_DBID_MISMATCH,FIXDBID" clause to fix in dbid mismatches on versions of ASE that support these clauses. If you are on an older version of Delphix or ASE that does not support this clause, you may be required to run the DBCC CHECKALLOC() command on the VDB.

If you use the ASE **ALTER DATABASE** command to increase the amount of space allocated to a VDB, you may need to run the **DBCC CHECKALLOC()** command on the VDB. If the VDB is unmounted and remounted without this command being run on it, you may run into the following ASE CR which leaves the VDB in an unusable state:

SAP ASE change request number	Description
798271	The errors 14545, 14547 and 14519 will be raised by MOUNT DATABASE if the unmounted database had previously been mounted on a server where the database ID was used and a new database ID was assigned, and this database was extended without previously running DBCC CHECKALLOC(dbname, fix). Additionally, two new options have been added to the MOUNT command: WITH FIXDBID, to instruct the MOUNT command to fix any possible database ID mismatch, and WITH ALLOW_DBID_MISMATCH, to prevent that MOUNT DATABASE fails if the database has different database ID values in the allocation pages. The fix for this issue has been ported to **15.7 SP138** and **16.0 SP02 PL05**.

Determining the need to run DBCC CHECKALLOC

DBID mismatches are most likely to occur under the following circumstances:

- When provisioning VDBs to the same ASE instance hosting the staging database of the dSource the VDB was provisioned from.
- When provisioning multiple copies of the same VDB to the same ASE instance.

Delphix has found the following technique as a reliable way to determine whether or not you need to run DBCC CHECKALLOC() after increasing the amount of space allocated to the VDB.

In the following example, a VDB has been provisioned back to the same ASE instance where the staging database was being hosted. The staging database had a **dbid=11865**, so ASE automatically assigns the VDB the next lowest available dbid (**dbid=22** in this case).

1. Resize the database:

```
1> SELECT name, dbid FROM sysdatabases WHERE name = 'Vdb5'
2> go
 name                           dbid
 ------------------------------ ------
 Vdb5                             22

1> DISK RESIZE name="AAA4$ff367xUvd67P7II_db1_dev26", size="4M"
2> go

1> ALTER DATABASE Vdb5 on AAA4$ff367xUvd67P7II_db1_dev26="4M"
2> go
Extending database by 1024 pages (4.0 megabytes) on disk
AAA4$ff367xUvd67P7II_db1_dev26
Database Vdb5 which is currently offline has been altered from size 2816
 logical pages (2816 physical pages) to 3840 logical pages
```

2. Run the following queries to determine of there is a dbid mismatch on the pages of the VDB. **DBCC PAGE**() requires the **sybase_ts_role** role.

```
1> -- Run a query to generate DBCC commands:
1> SELECT 'dbcc page (Vdb5,' + convert(varchar(5), lstart) + ', 0, 0)' FROM
master..sysusages WHERE dbid=db_id('Vdb5') AND segmap != 0
2> go

 ----------------------------
 dbcc page (Vdb5,0, 0, 0)
 dbcc page (Vdb5,1536, 0, 0)
 dbcc page (Vdb5,1792, 0, 0)
 dbcc page (Vdb5,2816, 0, 0)

1> -- Turn on DBCC traceflag 3604 to direct output to stdout
2> dbcc traceon(3604)
3> GO

1> dbcc page (Vdb5,0, 0, 0)
2> dbcc page (Vdb5,1536, 0, 0)
3> dbcc page (Vdb5,1792, 0, 0)
4> dbcc page (Vdb5,2816, 0, 0)
5> go | grep dbid
    bmass_next=0x0 bmass_prev=0x0 bdbid=22
pageno=0 dealloc_count=102 allocnode=0 ptnid=99 allocation_page dbid=11865
 timestamp=0000 00002730, segmap=0x00000003 (0x00000002
    bmass_next=0x0 bmass_prev=0x0 bdbid=22
pageno=1536 dealloc_count=0 allocnode=0 ptnid=99 allocation_page dbid=11865
 timestamp=0000 00000001, segmap=0x00000004 (0x00000004
    bmass_next=0x0 bmass_prev=0x0 bdbid=22
pageno=1792 dealloc_count=0 allocnode=0 ptnid=99 allocation_page dbid=11865
 timestamp=0000 00000001, segmap=0x00000003 (0x00000002
    bmass_next=0x0 bmass_prev=0x0 bdbid=22
pageno=2816 dealloc_count=0 allocnode=0 ptnid=99 allocation_page dbid=22
 timestamp=0000 00000001, segmap=0x00000003 (0x00000002
```

Notice that three pages (0, 1536 and 1792) have the staging database's dbid (11865) and one page (2816) has the correct dbid (22) that matches the dbid of 22 stored in sysdatabases.

If all the dbid's match the dbid in sysdatabases, there is no need to run DBCC CHECKALLOC().

This means that DBCC CHECKALLOC() must be run on the VDB to correct the dbid mismatch:

```
1> dbcc checkalloc(Vdb5)
2> go
...etc...
11 allocation pages have been corrected to match database ID 22.
```

Note: In the code block above 11 pages were fixed.

Now if you run the **DBCC PAGE()** query again you can see that all of the dbid's match:

```
1> dbcc page (Vdb5,0, 0, 0)
2> dbcc page (Vdb5,1536, 0, 0)
3> dbcc page (Vdb5,1792, 0, 0)
4> dbcc page (Vdb5,2816, 0, 0)
5> go | grep dbid
    bmass_next=0x0 bmass_prev=0x0 bdbid=22
pageno=0 dealloc_count=102 allocnode=0 ptnid=99 allocation_page dbid=22 timestamp=000
0 00002730, segmap=0x00000003 (0x00000002
    bmass_next=0x0 bmass_prev=0x0 bdbid=22
pageno=1536 dealloc_count=0 allocnode=0 ptnid=99 allocation_page dbid=22 timestamp=00
00 00000001, segmap=0x00000004 (0x00000004
    bmass_next=0x0 bmass_prev=0x0 bdbid=22
pageno=1792 dealloc_count=0 allocnode=0 ptnid=99 allocation_page dbid=22 timestamp=00
00 00000001, segmap=0x00000003 (0x00000002
    bmass_next=0x0 bmass_prev=0x0 bdbid=22
pageno=2816 dealloc_count=0 allocnode=0 ptnid=99 allocation_page dbid=22 timestamp=00
00 00000001, segmap=0x00000003 (0x00000002
```

Recovering from DBID mismatch errors

If a VDB containing mismatched dbids gets unmounted, it will likely get ASE errors 14545, 14547 and 14519 during the next attempt to **MOUNT** it and it will now be in an unusable state. Options for recovery include:

- For ASE versions that do not have the fix for ASE CR 798271, you may rewind the VDB to a snapshot taken prior to resizing the VDB. Any data changes made since this point in time will be lost.

On newer versions of ASE where ASE CR 798271 has been fixed, you can manually mount the VDB using the new **MOUNT** command options:

```
1> -- Get the device listing for the VDB from the manifest
2>  MOUNT DATABASE Vdb5 FROM '/export/home/sybase/toolkit/564dfe60-40b6-
aec9-26fb-3baeb4e7b23b-vdb-19/datafile/manifest' WITH LISTONLY
3> go
[database]
Vdb5
[device]
```

```
'/export/home/sybase/toolkit/564dfe60-40b6-aec9-26fb-3baeb4e7b23b-vdb-19/datafile/
dxnff367xUvd67P7II_db1_dev26' =
'AAA4$ff367xUvd67P7II_db1_dev26'
'/export/home/sybase/toolkit/564dfe60-40b6-aec9-26fb-3baeb4e7b23b-vdb-19/datafile/
dx9279VtfPAvfOm5xY_db1_dev27' =
'AAA5$279VtfPAvfOm5xY_db1_dev27'

1> -- Mount the VDB using the WITH ALLOW_DBID_MISMATCH clause
2> MOUNT DATABASE [Vdb5] AS [Vdb5] FROM '/export/home/sybase/toolkit/564dfe60-40b6-
aec9-26fb-3baeb4e7b23b-vdb-19/datafile/manifest'
3> WITH ALLOW_DBID_MISMATCH USING
4> '/export/home/sybase/toolkit/564dfe60-40b6-aec9-26fb-3baeb4e7b23b-vdb-19/datafile/
dxnff367xUvd67P7II_db1_dev26' = 'AAA4$ff367xUvd67P7II_db1_dev26',
5> '/export/home/sybase/toolkit/564dfe60-40b6-aec9-26fb-3baeb4e7b23b-vdb-19/datafile/
dx9279VtfPAvfOm5xY_db1_dev27' = 'AAA5$279VtfPAvfOm5xY_db1_dev27'
6> go
The physical device '/export/home/sybase/toolkit/564dfe60-40b6-
aec9-26fb-3baeb4e7b23b-vdb-19/datafile/dxnff367xUvd67P7II_db1_dev26'
has been automatically assigned the logical device name
'AAA6$ff367xUvd67P7II_db1_dev26'.
The physical device '/export/home/sybase/toolkit/564dfe60-40b6-
aec9-26fb-3baeb4e7b23b-vdb-19/datafile/dx9279VtfPAvfOm5xY_db1_dev27'
has been automatically assigned the logical device name
'AAA7$279VtfPAvfOm5xY_db1_dev27'.
Started estimating recovery log boundaries for database 'Vdb5'.
Database 'Vdb5', checkpoint=(1543, 13), first=(1543, 13), last=(1543, 13).
Completed estimating recovery log boundaries for database 'Vdb5'.
Started ANALYSIS pass for database 'Vdb5'.
Completed ANALYSIS pass for database 'Vdb5'.
Started REDO pass for database 'Vdb5'. The total number of log records to process is
1.
Completed REDO pass for database 'Vdb5'.
MOUNT DATABASE: Completed recovery of mounted database 'Vdb5'.
MOUNT DATABASE: A new database id was required for database 'Vdb5' in order to mount
it. Execute DBCC CHECKALLOC(Vdb5, fixdbid) to
correct it.


1> DBCC CHECKALLOC(Vdb5, fixdbid)
2> go
Checking Vdb5: Logical pagesize is 4096 bytes
Total (# alloc pages = 15, # of alloc pages modified = 15).
DBCC execution completed. If DBCC printed error messages, contact a user with System
Administrator (SA) role.

1> -- Confirm pages all have the correct dbid
2> SELECT 'dbcc page (Vdb5,' + convert(varchar(5), lstart) + ', 0, 0)' FROM
master..sysusages WHERE dbid=db_id('Vdb5') AND segmap != 0
3> go

 ----------------------------
 dbcc page (Vdb5,0, 0, 0)
```

```
 dbcc page (Vdb5,1536, 0, 0)
 dbcc page (Vdb5,1792, 0, 0)
 dbcc page (Vdb5,2816, 0, 0)

1> dbcc traceon(3604)
2> dbcc page (Vdb5,0, 0, 0)
3> dbcc page (Vdb5,1536, 0, 0)
4> dbcc page (Vdb5,1792, 0, 0)
5> dbcc page (Vdb5,2816, 0, 0)
6> go | grep dbid
    bmass_next=0x0 bmass_prev=0x0 bdbid=23
pageno=0 dealloc_count=102 allocnode=0 ptnid=99 allocation_page dbid=23 timestamp=000
0 00002730, segmap=0x00000003 (0x00000002
    bmass_next=0x0 bmass_prev=0x0 bdbid=23
pageno=1536 dealloc_count=0 allocnode=0 ptnid=99 allocation_page dbid=23 timestamp=00
00 00000001, segmap=0x00000004 (0x00000004
    bmass_next=0x0 bmass_prev=0x0 bdbid=23
pageno=1792 dealloc_count=0 allocnode=0 ptnid=99 allocation_page dbid=23 timestamp=00
00 00000001, segmap=0x00000003 (0x00000002
    bmass_next=0x0 bmass_prev=0x0 bdbid=23
pageno=2816 dealloc_count=0 allocnode=0 ptnid=99 allocation_page dbid=23 timestamp=00
00 00000001, segmap=0x00000003 (0x00000002
```

More information

VDBs are unmounted and remounted during operations such as:

- Disable/Enable of the VDB.
- Stop/Start of the VDB.
- Rebooting of the server hosting the VDB's ASE instance.

DBAs are unlikely to have found the need to run **DBCC CHECKALLOC()** on physical databases in the normal workflow of resizing a database unless the ASE **MOUNT/UNMOUNT** command is used. When you **MOUNT** a database on a server that already has the mounted database's dbid in use, ASE assigns the mounted database a new dbid and this gets reflected in the **sysdatabases** and **sysusages** tables in the master database and in the **dbtable** memory structure for the mounted database. However, the dbid in the dbinfo structure and allocation pages in the mounted database are not updated by mount. Doing so can take a long time on a big database. **DBCC CHECKALLOC()** will correct the values in all the allocation pages.

In Delphix version 5.1.3.0 and higher, Delphix will create databases using a high `dbid` (using the `"CREATE DATABASE with dbid="` clause). By using a high `dbid` (ASE supports a maximum of 32,767 dbids), it is unlikely that the `dbid` will be reused and thus Delphix will be able to avoid the errors (14545, 14547 and 14519). It attempts to assign unique **dbid** values for staging databases in the range 10000-30000.

The second part of this fix involves keeping the staging database mounted so that the **dbid** is pinned and no other database gets created and uses the staging database's **dbid**. The staging database will remain mounted except for a brief window where we unmount it to create a manifest file.

The final part of this fix utilizes the ASE **DBCC PAGE()** command to confirm the **dbid** on select pages from the sysusages table all match the database's actual **dbid**. If there is a mismatch found, Delphix will call **DBCC CHECKALLOC()** to fix the database. Delphix calls **DBCC PAGE()** in order to avoid calling **DBCC CHECKALLOC()** as the **CHECKALLOC()** routine can add significant time to database recovery (performance considerations for CHECKALLOC).

V2P with an SAP ASE dSource or VDB

Requirements

This topic describes the procedure for exporting a virtual database (VDB) to a physical one, also known as V2P. Before performing the V2P operation, you must have created a database on the target instance into which you will load the exported data. It must be sufficiently large, and it must have been created with the **for load** SAP ASE option.

The Delphix Engine will initiate a load command using the database specified. The V2P operation will overwrite any existing data in this database. After a V2P operation takes place, the target physical database is no longer in FOR LOAD state.

Procedure

1. Login to the **Delphix Management** application.
2. Select the **dSource** or **VDB** you want to export.
3. Select the **snapshot** of the dSource or VDB state you want to export.
4. Click **V2P**.
5. Select the **target environment**. The target environment should have been added to Delphix previously and should meet all target host requirements, see Requirements for SAP ASE Environments and Databases.
6. Under **Installation**, select which **instance** that you want to export to.
7. Enter the **Name** of the database on the target instance into which you want to load the exported data.
8. Select whether or not to **Run Recovery After V2P**. When this option is set, the Delphix Engine will online the database when the export is done.
9. Click **Next**.
10. Select whether you want to have an email sent to you when the export process completes.
11. Click **Submit**.

Additional SAP ASE VDB topics

Deleting an SAP ASE VDB

Procedure Deleting a VDB is an unrecoverable operation. Proceed only if you want to permanently destroy the unique data that was created in the VDB. Login to the Delphix Management application. Click Manage . Select Datasets . Click th...

Updated on : 25 May 2023

Enabling and disabling SAP ASE VDBs

This topic describes how to enable and disable a virtual database (VDB). Disabling a VDB is a pre-requisite for procedures such as VDB migration or upgrade. Disabling a VDB removes all traces of it, including any configuration files, from the targe...

Updated on : 25 May 2023

Migrating an SAP ASE VDB

This topic describes how to migrate a virtual database (VDB) from one target environment to another. In certain situations, you may want to migrate a VDB to a new target environment. For example: When upgrading the host on which the VDB resides ...

Updated on : 25 May 2023

Refreshing an SAP ASE VDB

This topic describes how to manually refresh a virtual database (VDB). Refreshing a VDB will re-provision it from the dSource. As with the normal provisioning process, you can choose to refresh the VDB from a snapshot or a specific point in time. H...

Updated on : 25 May 2023

Rewinding an SAP ASE VDB

This topic describes the procedure for rewinding an SAP ASE virtual database (VDB). Rewinding a VDB rolls it back to a previous point in its TimeFlow and re-provisions the VDB. The VDB will no longer contain changes that were made after the rewind ...

Updated on : 25 May 2023

Upgrading SAP ASE VDBs

This topic describes how to upgrade an SAP ASE VDB to a higher version of SAP ASE instance. Procedure Login to the Delphix Management application. Click Manage . Select Datasets . Select the VDB to be upgraded. From the ...

Updated on : 25 May 2023

Deleting an SAP ASE VDB

Procedure

Deleting a VDB is an unrecoverable operation. Proceed only if you want to permanently destroy the unique data that was created in the VDB.

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Click the **VDB** that you want to delete.
5. From the Actions menu (...) select **Delete.**
6. If stopping or starting the VDB requires particular credentials for the target environment other than those of the default environment user:
    a. Check **Provide Privileged Credentials**.
    b. Enter the **username** and **password**.
    c. Click **Validate Credentials**.
7. Click **Delete** to confirm that you want to delete the VDB.

If the VDB was currently active, the Delphix Engine will shut it down, unmount all filesystems from the target environment, and finally delete the VDB itself.

Enabling and disabling SAP ASE VDBs

This topic describes how to enable and disable a virtual database (VDB).

Disabling a VDB is a pre-requisite for procedures such as VDB migration or upgrade. Disabling a VDB removes all traces of it, including any configuration files, from the target environment to which it was provisioned. When you later enable the VDB again, these configuration files are restored on the target environment.

Procedure

1. Click **Manage**.
2. Select **Datasets**.
3. Click the **VDB** you want to disable.
4. From the Actions menu (...) select **Disable**.
5. Click **Disable** to acknowledge the warning.

When you are ready to enable the VDB again, from the Actions menu (...) select **Enable**, and the VDB will continue to function as it did previously.

Migrating an SAP ASE VDB

This topic describes how to migrate a virtual database (VDB) from one target environment to another.

In certain situations, you may want to migrate a VDB to a new target environment. For example:

- When upgrading the host on which the VDB resides
- During a general data center migration
- To distribute the VDB load across target environments

This is easily accomplished by first disabling the database, then using the Migrate VDB feature to select a new target environment.

Prerequisites

- You must first disable the VDB before migrating it. Follow the steps outlined in Enabling and Disabling SAP ASE VDBs
- You must have already set up a new target environment that is compatible with the VDB that you want to migrate.

Procedure

1. Login to your Delphix Management application.
2. Click **Manage**.
3. Select **Datasets**.
4. Click the **VDB** you want to migrate.
5. From the **Actions** menu (...) select **Disable**.
6. Click **Disable** to confirm.
7. From the **Actions** menu (...) select **Migrate.**
8. In the Migrate window select:
    a. Environment: the new **target environment** for the VDB
    b. User: select a user
    c. Installation: the **installation** where you want to migrate
9. Click Migrate to confirm your selections.
10. From the **Actions** menu (...) select **Enable**.
11. Click **Enable** to confirm.

Within a few minutes, your VDB will re-start in the new environment, and you can continue to work with it as you would with any other VDB.

Refreshing an SAP ASE VDB

This topic describes how to manually refresh a virtual database (VDB).

Refreshing a VDB will re-provision it from the dSource. As with the normal provisioning process, you can choose to refresh the VDB from a snapshot or a specific point in time. However, you should be aware that refreshing a VDB would delete any changes that have been made to it over time. When you refresh a VDB, you are essentially re-setting it to the state you select during the refresh process. You can refresh a VDB manually, as described in this topic, or you can set a VDB refresh policy, as described in Managing Policies

> Although the VDB no longer contains the previous contents, the previous Snapshots and Timeflow still remain in Delphix and are accessible through the Command Line Interface (CLI).

Prerequisites

To refresh a VDB, you must have the following permissions:

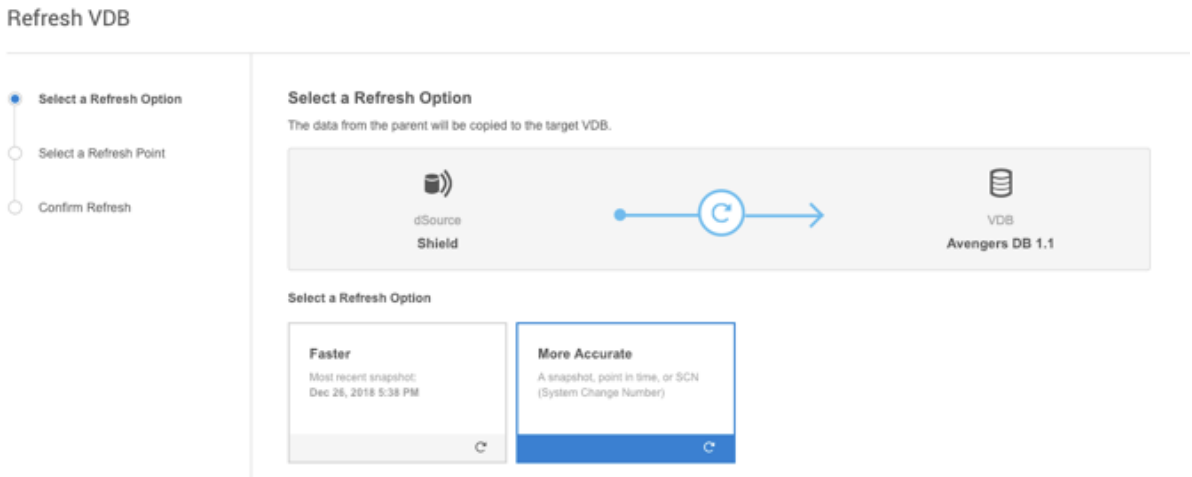
- **PROVISIONER** permissions on the dSource associated with the VDB
- **PROVISIONER** permissions on the group that contains the VDB
- **Owner** permissions on the VDB itself
- **Data** is a role that allows DB_ROLLBACK, DB_REFRESH, READ_ACTION, DB_SYNC, JOB_CANCEL**.**
- **Read** is a role that allows the user to inspect objects via the READ_ACTION permission.

A user with admin credentials can perform a VDB Refresh on any VDB in the system.

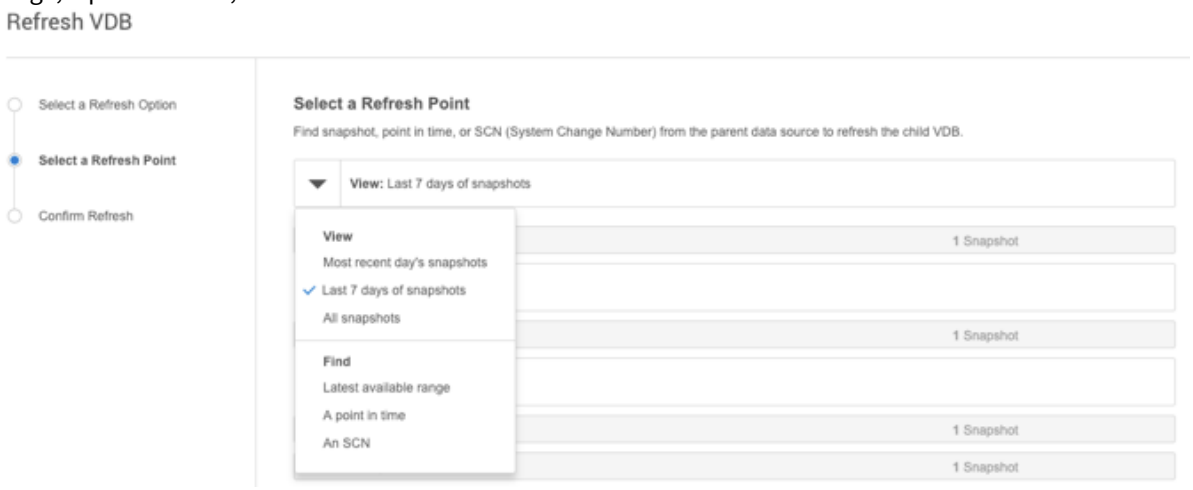
Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **VDB** you want to refresh.
5. Click the **Refresh VDB** button.

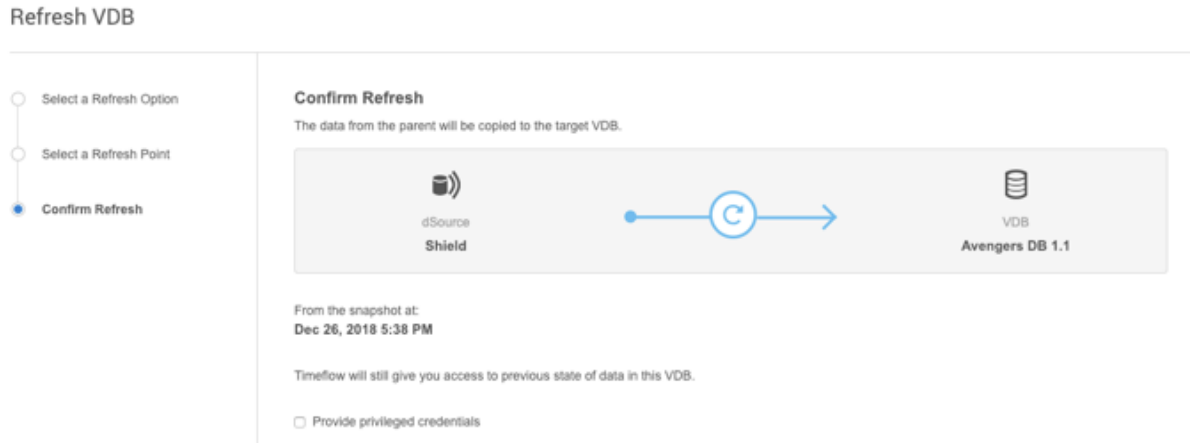
6. Select **More Accurate** and **Next**.
7.



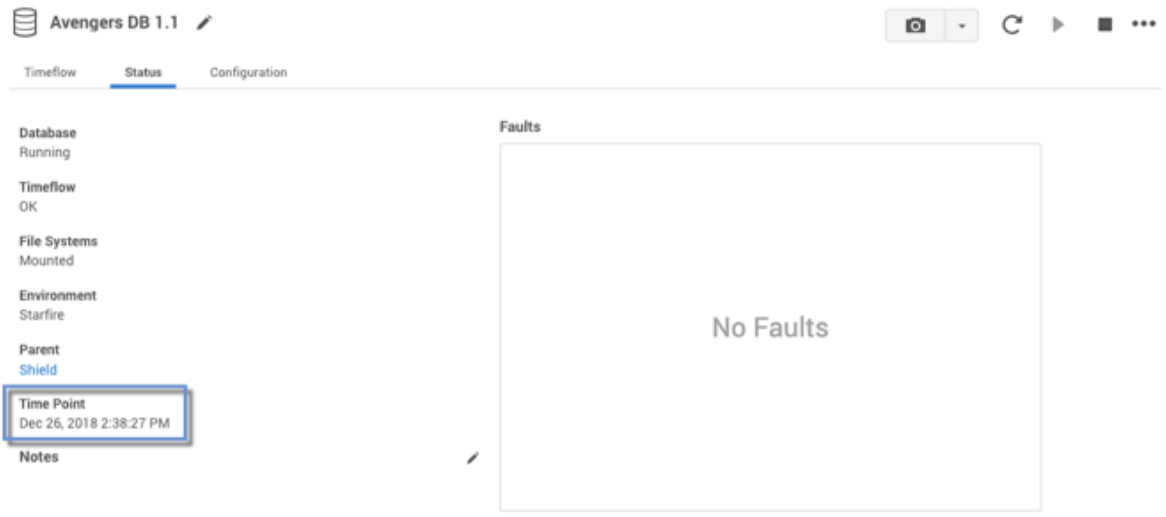
Select the desired **refresh point** snapshot or click the **down arrow** icon to choose the Latest available range, A point in time, or An SCN to refresh from.



8. Click **Next**.
9. Click **Submit** to confirm.



10. Click the **Actions**link to watch the progress of the refresh job.
11. To see when the VDB was last refreshed/provisioned, check the **Time Point** on the Status page


Avengers DB 1.1
Timeflow
Status
Configuration
Database
Running
Timeflow
OK
File Systems
Mounted
Environment
Starfire
Parent
Shield
Time Point
Dec 26, 2018 2:38:27 PM
Notes
Faults
No Faults

Rewinding an SAP ASE VDB

This topic describes the procedure for rewinding an SAP ASE virtual database (VDB).

Rewinding a VDB rolls it back to a previous point in its TimeFlow and re-provisions the VDB. The VDB will no longer contain changes that were made after the rewind point.

> Although the VDB no longer contains changes made after the rewind point, the rolled-over snapshots and TimeFlow still remains in Delphix and are accessible through the Command Line Interface (CLI). See the topic CLI Cookbook: Rolling Forward a VDB for instructions on how to use these snapshots to refresh a VDB to one of its later states after it has been rewound.

Prerequisites

To rewind a VDB, you must have the following permission:

- **Owner** permissions on the VDB itself

You do NOT need owner permissions for the group that contains the VDB. A user with Delphix Admin credentials can perform a VDB rewind on any VDB in the system.

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **VDB** you want to rewind.
5. Click the **TimeFlow** tab.
6. Select the rewind point as a snapshot or a point in time.
7. Click **Rewind**.
8. If you want to use login credentials on the target environment other than those associated with the environment user, click **Provide Privileged Credentials**.
9. Click **Yes** to confirm.

> You can use TimeFlow bookmarks as the rewind point when using the CLI. Bookmarks can be useful to:
> - Mark where to rewind to – before starting a batch job on a VDB, for example.
> - Provide a semantic point to revert back to, in case the chosen rewind point turns out to be incorrect.
>
> For a CLI example using a TimeFlow bookmark, see CLI Cookbook: Provisioning a VDB from a Timeflow Bookmark.

Upgrading SAP ASE VDBs

This topic describes how to upgrade an SAP ASE VDB to a higher version of SAP ASE instance.

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **VDB** to be upgraded.
5. From the **Actions** menu (...) select **Disable**.
6. Click **Disable** to confirm.
7. Upgrade the ASE instance(s) hosting the VDBs.
8. Click the **Refresh** icon (under the **Manage > Environments** menu) to refresh the Delphix environment hosting the VDBs so that Delphix registers the new version of ASE.
9. Enable the VDB. The VDBs will be upgraded the first time they’re enabled.
10. Repeat steps 4 to 9 for each VDB you want to upgrade.

Backup server best practices

- The backupserver log should have read permissions for the Delphix environment user
- Dump file names should be unique per source backup server.
- Dumps should be done to a local backupsever. Dumps done using the “at <backupserver>” syntax are not discovered and ingested/applied.
- Dumps of production databases that use the same backup server and same load backup path should be available on the same staging host that was pointed at during the lin
- If using a remote backup server as load location, make sure “interfaces” file changes are done to ensure staging host backupserver can communicate with the remote backup server.
- A user running backupserver or process owner of the backupserver on the staging host should have read access for all the dump files that belong to dSources that use this backupserver.

Recommendations/best practices

1. Name the dump files as unique as possible.
2. Delphix recursively searches for backup files in a given load backup path. So assign or choose the load backup path for a given dSource to be as specific and unique as possible. For e.g, if the load backup path is /dumps for dSource db1, but the database dumps are copied to /dumps/db1, choose /dumps/db1 instead to avoid duplicate files that can happen with other dSources or databases.
3. Avoid linking sources where it’s backup location is a child directory of another dSource’s backup location. Ideally, each dSource using the same staging host should have its backup path set to sibling directories for ease of search.
4. Avoid copying files (of any type) with the same name as backup files that are being ingested.
5. Avoid copying other dSource or database backups to the same load backup path of the dSource of interest. You can copy, as long as dump file names are unique.
6. Only use a remote backup server as the load location if dumps cannot be available on the staging host.
7. Avoid, if possible, using production backup server as load location and the backup files are on production host when configuring a dSource with remote backupserver.
8. DE by default assumes backup files are available on the staging host when searching for them. If the backup files are available on the staging host, avoid using the remote backupserver option on the link/data management wizard.
9. If validated sync reports any fault, look at other faults that were raised, resolve all of them as well along with the current fault.
10. When copying striped backups to staging, keep all stripes in the same location.
11. If possible, create separate database dumps and/or transaction logs by more than 1 minute apart so that the dump header name is unique.
12. The Delphix Engine, by default, looks for backups that happened within the last six months from the current date/time when a source database is linked for the first time
    a. If there is ever a need to ingest an older backup, use “Specific Backup” sync parameters and specify the location of the backup files to ingest.
    b. Please reach out to Delphix Support for help if there is a reason for concern regarding the 6 months window.
13. The Delphix Engine periodically purges discovered backup files from its internal catalog that are older than 6 months (default). If there is a need to ingest an old backup, use the “Specific Backup” option as there is no mechanism to rediscover old backups.

SAP ASE hook operations

Shell operations

RunCommand operation

The RunCommand operation runs a shell command on a Unix environment using whatever binary is available at `/bin/sh`. The environment user runs this shell command from their home directory. The Delphix Engine captures and logs all output from this command. If the script fails, the output is displayed in the Delphix Management application and command-line interface (CLI) to aid in debugging.

If successful, the shell command must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Examples of RunCommand operations

You can input the full command contents into the RunCommand operation.

```
remove_dir="$DIRECTORY_TO_REMOVE_ENVIRONMENT_VARIABLE"

if test -d "$remove_dir"; then
    rm -rf "$remove_dir" || exit 1
fi

exit 0
```

If a script already exists on the remote environment and is executable by the environment user, the RunCommand operation can execute this script directly.

```
/opt/app/oracle/product/10.2.0.5/db_1/dbs/myscript.sh "$ARG_ENVIRONMENT_VARIABLE"
"second argument in double quotes" 'third argument in single quotes'
```

RunBash operation

The RunBash operation runs a Bash command on a Unix environment using a `bash` binary provided by the Delphix Engine. The environment user runs this Bash command from their home directory. The Delphix Engine captures and logs all output from this command. If the script fails, the output is displayed in the Delphix Management application and command-line interface (CLI) to aid in debugging.

If successful, the Bash command must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Example of RunBash Operations

You can input the full command contents into the RunBash operation.

```
remove_dir="$DIRECTORY_TO_REMOVE_ENVIRONMENT_VARIABLE"
```

```
# Bashisms are safe here!
if [[ -d "$remove_dir" ]]; then
    rm -rf "$remove_dir" || exit 1
fi

exit 0
```

Shell operation tips

Using `nohup`

You can use the `nohup` command and process backgrounding from resource in order to "detach" a process from the Delphix Engine. However, if you use `nohup` and process backgrounding, you MUST redirect `stdout` and `stderr.`

Unless you explicitly tell the shell to redirect `stdout` and `stderr` in your command or script, the Delphix Engine will keep its connection to the remote environment open while the process is writing to either `stdout` or `stderr` . Redirection ensures that the Delphix Engine will see no more output and thus not block waiting for the process to finish.

For example, imagine having your `RunCommand` operation background a long-running Python process. Below are the bad and good ways to do this.

Bad Examples

- `nohup python file.py & # no redirection`
- `nohup python file.py 2>&1 & # stdout is not redirected`
- `nohup python file.py 1>/dev/null & # stderr is not redirected`
- `nohup python file.py 2>/dev/null & # stdout is not redirected`

Good Examples

- `nohup python file.py 1>/dev/null 2>&1 & # both stdout and stderr redirected, Delphix Engine will not block`

Other operations

RunExpect operation

The RunExpect operation executes an Expect script on a Unix environment. The Expect utility provides a scripting language that makes it easy to automate interactions with programs which normally can only be used interactively, such as `ssh` . The Delphix Engine includes a platform-independent implementation of a subset of the full Expect functionality.

The script is run on the remote environment as the environment user from their home directory. The Delphix Engine captures and logs all output of the script. If the operation fails, the output is displayed in the Delphix Management application and CLI to aid in debugging.

If successful, the script must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Example of a RunExpect operation

Start an `ssh` session while interactively providing the user's password.

```
spawn ssh user@delphix.com
expect {
    -re {Password: } {
        send "${env(PASSWORD_ENVIRONMENT_VARIABLE)}\n"
    }
    timeout {
        puts "Timed out waiting for password prompt."
        exit 1
    }
}
exit 0
```

SAP ASE environment variables

Operations that run user-provided scripts have access to environment variables. For operations associated with specific dSources or virtual databases (VDBs), the Delphix Engine will always set environment variables so that the user-provided operations can use them to access the dSource or VDB.

dSource environment variables

Environment variables	Description
`ASE_ENVUSER`	Environment username for the dSource
`ASE_DBUSER`	Database username for the dSource
`ASE_DATABASE`	Database name for the dSource
`ASE_INSTANCE`	SAP ASE Instance name for the dSource
`ASE_PORT`	SAP ASE Instance port for the dSource

VDB environment variables

Environment variables	Description
`ASE_ENVUSER`	Environment username for the VDB
`ASE_DBUSER`	Database username for the VDB
`ASE_DATABASE`	Database name for the VDB
`ASE_INSTANCE`	SAP ASE Instance name for the VDB
`ASE_PORT`	SAP ASE Instance port for the VDB

Staging server environment variables

Environment variables	Description
`ASE_ENVUSER`	

Support for dump history file

Overview

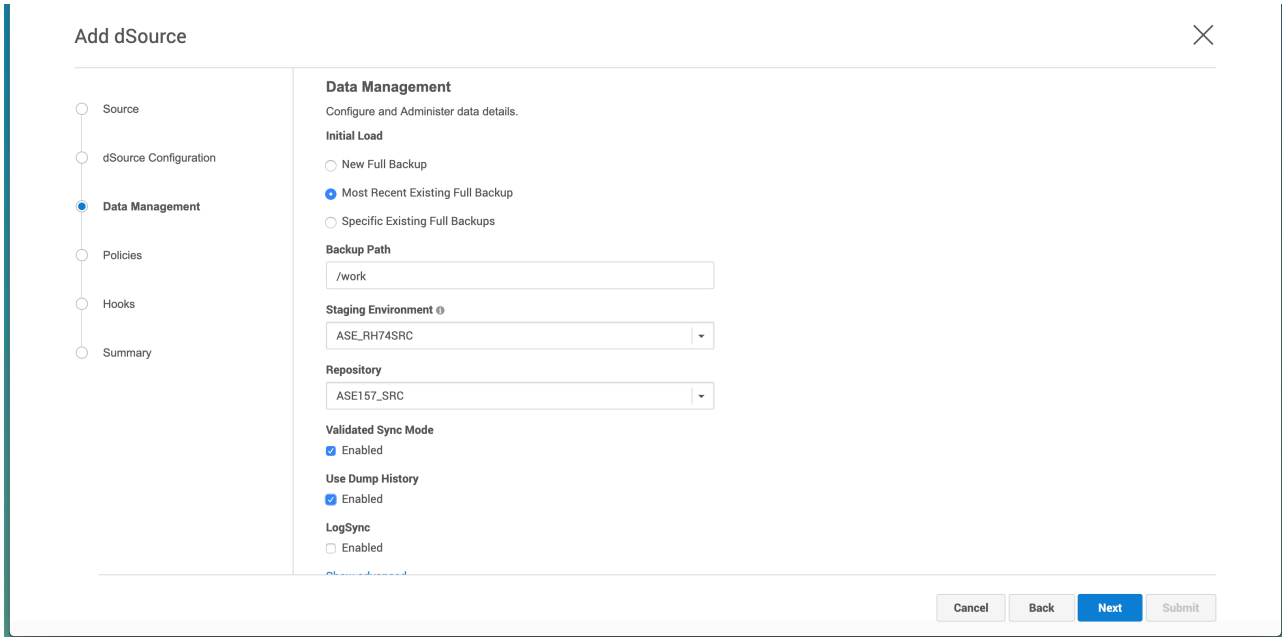
Prior to ASE version 15.7 ESD#2, ASE did not provide any structured information about backup/restore operations performed on databases. Delphix relied on the backup server log file to find information about any backups performed using a particular backup server. The Delphix Engine parsed this log file, read each backup file to find which database the backup belonged to and restored the ones required for linked databases. As backup information is not very structured in this log file, getting backup history from this file resulted in the collection and parsing of unnecessary data. Starting with ASE 15.7 ESD#2, users can enable an option at the ASE instance level to record all backup and restore operations in a particular file, known as the Dump History file. Backup and restore operation information is stored in this file in a structured way and can be reliably used to get information about backups performed on a given database. When Dump History is enabled for a linked source, Delphix will use this feature to get information about backups for all sync operations including validated sync.

Prerequisites

To use this feature, Dump history needs to be enabled at the ASE Source instance level.

Linking a database with dump history enabled

By default, Delphix uses the backup server log file to discover backup information. Users can switch to Dump History by explicitly enabling it for the dSource. Before linking a database with Dump History enabled, make sure that Dump History is enabled on the source ASE instance. To enable Dump History for the database being linked, go through the linking wizard and on the **Data Management** page, check the box marked "Use Dump History".



SAP HANA environments and data sources

Starting 6.0.8.0 and HANA plugin version 6.0.0, Delphix provides you with a vSDK-based solution. **Delphix highly recommends their users to use this HANA Staged (vSDK-based) plugin over the existing HANA Direct (Lua-based) plugin**.
Currently, the Lua-based HANA plugin 4.3.1 and earlier versions are also supported. Please refer to SAP HANA Release Notes for supported versions. For Lua-based HANA plugin documentation, refer to Delphix Engine 6.0.7.0 Documentation and the earlier versions.

This section covers the following topics:

SAP HANA overview

SAP HANA is an in-memory database, columnar-oriented database management system developed by SAP. The key differentiators of HANA are its tight integration into the SAP application stack coupled with its columnar store, the latter of which is intended to enable customers to fold both OLTP and OLAP operations into a single database entity. In practical terms, this means an SAP HANA database stores both analytical and transactional data. Analysis of the data combination is possible on the fly, helping businesses make real-time analytical decisions.

SAP HANA as a database product continues to evolve dramatically. Key among recent changes has been the movement from a Single Database Container (SDC) to Multi-Tenant Database Containers (MDC), an architectural change that has become the exclusive architecture of HANA version 2.0.

Implications of the HANA column store

In a row-based relational database, data is stored by row, usually in large files. Data is then accessed from the database by reading and returning full rows of data.

Row

```
Country Product Sales
1   US Alpha   3000
2   US Beta    1250 <-----storing this line across------>
3   JP Alpha   400
4   UK Alpha   700
```

Column

```
Country Product Sales    ^
US      Alpha 3000       |
US      Beta 1250    storing each column up / down
JP      Alpha 400        |
UK      Alpha 700        v
```

Select

In a columnar database, the entire row does not need to be read, only the columns associated with the query. This columnar structure improves (read) efficiency and, therefore, query performance, particularly for extensive data sets.

Columnar data storage allows highly efficient compression because most of the columns contain only a few distinct values (compared to the number of rows). The data, therefore, has a smaller memory footprint and takes fewer processing cycles to interrogate.

Better parallel Processing: in a column store, data is already vertically partitioned. This means that operations on different columns can easily be processed in parallel. If multiple columns need to be searched or aggregated, each of these operations can be assigned to a different processor core.

Insert / Update

Storing data in columns instead of rows is challenging for workloads with many data modifying operations. Therefore, the concept of a differential buffer was introduced, where new entries are written to a differential buffer

first. In SAP HANA, this is referred to as the delta store. In contrast to the main store, the delta store is optimized for inserts. At a later point in time and depending on thresholds, e.g. the frequency of changes and new entries, the data in the delta store is merged into the main store. This process is referred to as delta merge.

The delta merge process has important implications for filesystem-based table modifications. Delta merge on any table will reorganize all data in that table and as a corollary will re-write the entire table to different filesystem blocks. *The space savings Delphix can offer by tracking block changes between snapshots becomes far less effective or predictable when entire tables are reorganized as a function of delta merge.*

Delphix architecture

The Delphix HANA plugin is based on a staging architecture, i.e. the linking process and SnapSync requires a staging environment. The staging environment is a HANA DB server host that leverages storage allocated directly on the Delphix Engine using NFS, and it must contain an instance of SAP HANA that is higher or matches the same version of the Source. To create a dSource on a staging server, a full backup would be required. The Recover command deposits its backup data onto the mounted storage area. The Delphix Engine snapshots the Staging server and the snapshots can be provisioned out to one or more target servers.

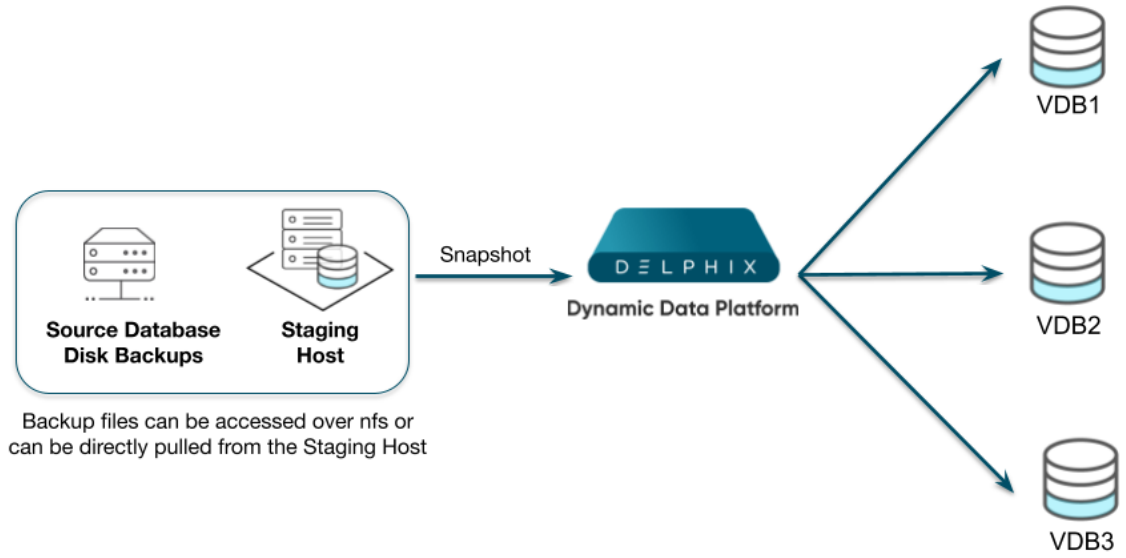
To create a dSource on a staging server, a full backup of one particular tenant database is required. With this solution, virtualization will be done at the tenant level only. This topic describes the high-level architecture of the Delphix HANA plugin solution.

This solution invokes HANA recovery of database backup files when a user selects to create a dSource on the staging server. The necessary backups/logs are created outside of Delphix and must exist before dSource creation.

Virtual databases can only be provisioned from a dSource (inheriting the state of the dSource).

The following is the high-level Delphix architecture with SAP HANA.

Delphix for HANA Architecture



There are two mechanisms to create dSource on the staging server.

- Pull architecture, where the HANA plugin pulls the data into the Delphix provided mount location.
- Staging Push architecture, where the user pushes the data into the Delphix engine.

HANA plugin with pull architecture

In Delphix HANA with Pull architecture, dSource is created using External backups which must be a HANA Native backup (ie: no support for third-party Backint tools). With External backups, the first dSource snapshot must include a Full backup, but can optionally include one or more incremental or differential backups. Subsequent dSource snapshots can be based upon additional incremental or differential backups (This will require a full backup along with incremental/differential backups, as recovery in Hana always requires a full backup).
HANA plugin supports an External backup, also known as Native Backups. Native backup implies that the backups are written in native storage format to local storage without a third-party backup tool.

The dSource creation operation invokes the HANA database recovery of the relevant HANA backup files. HANA database recovery can be a lengthy process. The duration of recovery time is subject to the size of the database being recovered. After completion of HANA recovery, the HANA instance will contain the newly recovered dSource tenant.

VDB provision operations from a dSource snapshot do not require HANA database recovery and can occur very quickly.

Failure to follow this recommendation diminishes the key value proposition for HANA, which is agility. Below are the recommended tiers:

- **dSource:** A virtual HANA database constructed by recovery.
- **Virtual Database (VDB):** A copy of a specific dSource snapshot, cloned using Delphix technology. These copies are fast to create with a small initial storage footprint. VDBs are the recommended workbench for developers. Multiple VDBs can exist from a single dSource. Each VDB corresponds to a dSource snapshot.

SAP HANA plugin with staging push architecture

Starting Delphix Engine version 6.0.12.0 and HANA plugin version 6.2.0, a new data ingestion mechanism has been introduced that will help users to push data into the Delphix provided mount point on their own.

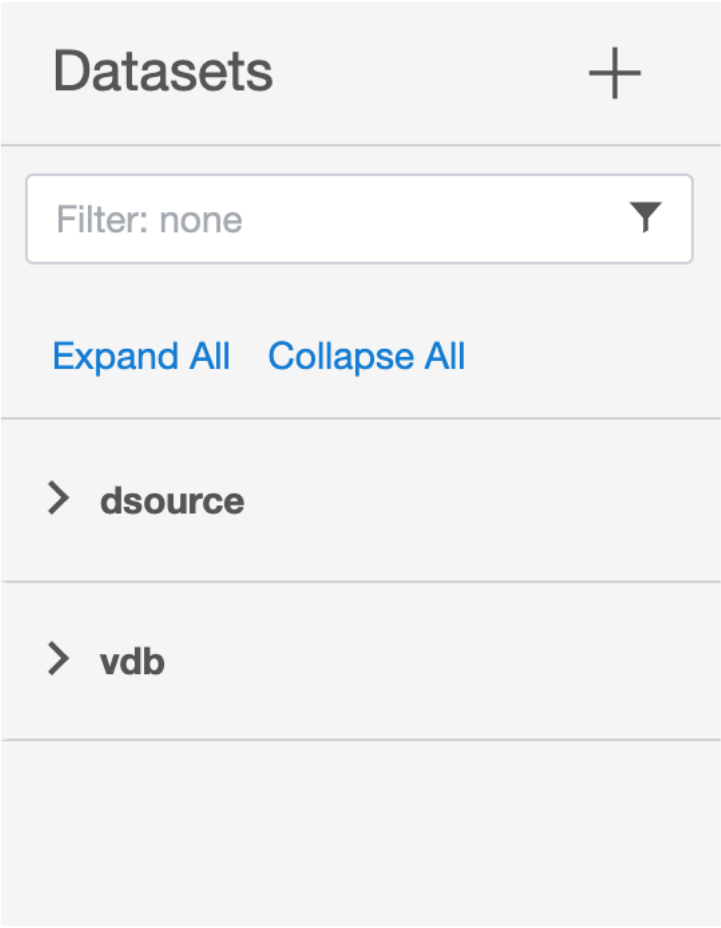
The previous versions of the HANA plugin were based only on the Pull approach and it was difficult to support the different ingestion mechanisms as it required a lot of effort to analyze, code, and develop the solution. With the Staging Push mechanism, data ingestion can be achieved with minimum effort.

Few points to be noted about the Staging Push mechanism.

- dSource creation creates an empty mount point on the staging host
- User managed backup and restore

Operational best practice

- Construct a dSource using the backup files. Construct multiple VDBs descended from the dSource.
- Create a meaningful object name.
- Create Dataset Groups that help delineate to which tier an object belongs.



- Segregate user targets (where VDBs reside) from the staging where HANA recovery must take place to create the dSource.
- Provision end-user VDBs, so that they can be created, refreshed, and rewound quickly.
- Layer VDBs to perform any common post-recovery operations to further streamline provisioning steps.
- Closely monitor and manage the quantity and lifespan of dSource and VDB snapshots, since each can consume additional Delphix storage.

Benefits of HANA plugin

- Virtualization of individual tenant databases instead of the entire system.
- Ability to mix and match virtualized tenants across any HANA system.
- Reduced Storage footprint. VDBs can use significantly less storage when initially provisioned from dSource.
- Relaxes strict requirement of exact HANA version match between source and target, leveraging HANAs ability to recover an earlier version against a more recent version.
- Source backup files can be ingested from a surrogate host without impact to the true production source host.

Limitations of HANA plugin

- The tenant's name should contain at least one alphanumeric character or one underscore. Also, it should not start with a digit.
- A single quote, double quote, and space characters are not allowed in the password.
- Datasets created using the Lua version cannot be upgraded to the vSDK version.
- Manual cleanup/unmounting would be required after **force delete** operation on any dataset.

SAP HANA 2.0 plugin installation

The HANA 2.0 Plugin is provided as a zip file on the Delphix download site. Follow the steps below to download the plugin:

1. In the web browser, go to the Delphix download site.
2. Login to the download site using email and password credentials.
3. Navigate to the required version number of Delphix Engine.
4. Go to the Plugins > HANA > HANA 2.0 > Plugin_<version_number>.zip folder and download the zip file.

For initial and upgrade installations refer to the section Delphix Engine Plugin Management for further details.

SAP HANA 2.0 plugin tools

Delphix provides the **create_backup_metadata.sh** script, which is bundled with the plugin. This script is required to be executed by customer for customer-created HANA backups, which are to be used to create or refresh a dSource. This script can be found at:

```
<path>/scripts/create_backup_metadata.sh
```

This script must be run prior to creating or refreshing a dSource from a customer-created backup. The script is not required for Delphix-initiated backups. The script must be run on the source instance where the source tenant database exists. The script retrieves and persists important metadata about the specific backup set. This information will be used by the plugin during the dSource creation and will then be stored in the metadata section of the dSource.

Prerequisites

- The script assumes the user has the credentials of the tenant on the source HANA instance.
- The script must be copied to the source host as the operating system user should have access to hdbsql.
- The script must be executable.
- Correct backup files of the source must exist on the source host or the backup files should be accessible from the source host.
- The script should have the write permissions in the directory where the backups have been kept.

You can run the `create_backup_metadata.sh` script in the following two ways.

1. Interactive mode
2. Providing the complete list of parameters (Non-Interactive mode)

Interactive mode

The mode allows you to input the values to the required/optional parameters one by one on the CLI. See the sample script run example below the Parameters table.

Parameters

The following table lists the optional and required parameters used by the script.

Parameter	Optional/Required	Description
Please enter the location of the Backup files:	Required	Location of the Backup files For example: */usr/sap/HDB/HDB00/source_backup*

Parameter	Optional/Required	Description
If the above provided location of the Backup files is different than the original location where backups were taken then please enter original location, otherwise press enter(OPTIONAL):	Optional	This is the original location where the backup files were created during the dSource creation which can be used later for recovery purposes Provide this parameter when the location provided at the "*Please enter the location of the Backup files:*" parameter is different from the location where the backup files are kept now For example: */usr/sap/HDB/HDB00/backup/data/DB_TEST1*
Please enter the location of the Log files(OPTIONAL):	Optional	Location of the Log files You may use logs for the recovery For example: */usr/sap/HDB/HDB00/source_log*
If the above provided location of the log files is different than the original location where log files were created then please enter original location, otherwise press enter(OPTIONAL):	Optional	This is the original location where the log files were created during the dSource creation which can be used later for recovery purposes Provide this parameter when the location provided at the "*Please enter the location of the Backup files:*" parameter is different from the location where the log files are kept now For example: */usr/sap/HDB/HDB00/backup/log/DB_TEST1* (Note that this is not a valid directory)
Please enter the Tenant Database Name:	Required	Name of the tenant database for which backups/logs were created For example: TEST1
Please enter the SID of the HANA database:	Required	SID of the tenant database for which backups/logs are present For example: HDB
Enter the Tenant User Name:	Required	Username to connect to the tenant database For example: SYSTEM
Tenant User Password:	Required	Password to connect to the tenant database
Please enter the Instance Number:	Required	Instance number of the tenant database For example: 00

Example

The following example shows an execution of the script with input parameters:

```
hdbadm@ip-10-110-244-207.delphix:/usr/sap/HDB/HDB00> ./create_backup_metadata.sh
Starting create_backup_metadata.sh
Please enter the location of the Backup files:/usr/sap/HDB/HDB00/source_backup

If the above provided location of the Backup files is different than the original location where backups were taken then please enter original locatio
n, otherwise press enter(OPTIONAL):/usr/sap/HDB/HDB00/backup/data/DB_TEST1

Please enter the location of the Log files(OPTIONAL):/usr/sap/HDB/HDB00/backup/log/DB_TEST1

If the above provided location of the log files is different than the original location where log files were created then please enter original locati
on, otherwise press enter(OPTIONAL):

Please enter the Tenant Database Name:TEST1

Please enter the SID of the HANA database:HDB

Please enter the Tenant User Name:SYSTEM

Tenant User Password:

Please enter the Instance Number:00
```

Usage

```
./create_backup_metadata.sh
```

Providing the complete list of parameters

This mode allows you to input the parameter values in a specified order at the run time. See the sample script run example below the Parameters table.

Parameters

The following table lists the optional and required parameters in the order that in which you need to pass the parameter values along with the script.

Parameter	Optional/Required	Description
TENANT_DATABASE_NAME	Required	Tenant database name for which the script needs to be run
SID	Required	SID of the HANA database
TENANT_USER_NAME	Required	Username to connect to the tenant database
TENANT_USER_PASSWORD	Required	Password to connect to the tenant database
INSTANCE_NUMBER	Required	Instance number of the HANA database
DATA_BACKUP_PATH	Required	The location where the backups are present
LOG_BACKUP_PATH	Optional	The location where the logs are present

Parameter	Optional/Required	Description
ORIG_DATA_BACKUP	Optional	Original location where these backups were created. Do not provide any value if backups are present at the same location where they were created
ORIG_LOG_BACKUP_PATH	Optional	Original location where logs were created. Do not provide any value if the logs are present at the same location where they were created

Example 1

The following example shows an execution of the script with input parameter values in the above-specified order.

```
hdbadm@target:/usr/sap/HDB/HDB00> ./create_backup_metadata.sh TEST HDB SYSTEM Delphix_123 00 /usr/sap/HDB/HDB00/source_backup
Starting create_backup_metadata.sh
Checking : /usr/sap/HDB/HDB00/source_backup


Running the script for TEST tenant
BACKUP ID of latest complete backup: 1633404668463
Metadata information of backup is stored in file: /usr/sap/HDB/HDB00/source_backup/backupTimeInfo
```

If all the mandatory parameters are not provided or parameters are not provided in the specified order, then the plugin will throw an error.

Example 2

The following example shows an error when all the mandatory parameter values are not provided.

```
hdbadm@target:/usr/sap/HDB/HDB00> ./create_backup_metadata.sh TEST HDB SYSTEM Delphix_123
Starting create_backup_metadata.sh


Incorrect number of parameters provided to the script. Please provide the below parameters to the script or do not pass any parameters which will invo
ke the script in interactive mode.

TENANT_DATABASE_NAME – Tenant Database name for which script needs to be run.

SID – SID of the HANA Database

TENANT_USER_NAME – User name to connect to the Tenant Database

TENANT_USER_PASSWORD – Password to connect to the Tenant Database

INSTANCE_NUMBER – Instance number of the HANA Database

DATA_BACKUP_PATH – Location where backups are present

LOG_BACKUP_PATH – Location where logs are present. This is an optional field.

ORIG_DATA_BACKUP – Original location where these backups were created. This is an optional field. Do not provide any value if Backups are present at t
he same location where they were created.

ORIG_LOG_BACKUP_PATH – Original location where logs were created. This is an optional field. Do not provide any value if Logs are present at same loca
```

Quick start guide for SAP HANA plugin

This quick start guide, which is excerpted from the larger User Guide, is intended to provide you with a quick overview of working with HANA database objects in the Delphix Engine. It does not cover any advanced configuration options or best practices, which can have a significant impact on performance. It assumes that you are working in a Lab/Dev setting and attempting to quickly test Delphix functionality. It assumes you will use the VMware Hypervisor.

Overview

In this guide, we will walk through deploying a Delphix Engine, starting with configuring Source and Target database environments on Windows servers. We will then create a dSource, and provision a VDB.

For purposes of the QuickStart, you can ignore any references to Replication or Masking.

Deploy OVA on VMware

Use the Delphix-supplied OVA file to install the Delphix Engine. The OVA file is configured with many of the minimum system requirements. The underlying storage for the install is assumed to be redundant SAN storage.

1. Download the OVA file from https://download.delphix.com. You will need a support login from your sales team or a welcome letter.
    a. Navigate to the Delphix Product Releases/<Current Version>/Appliance Images page.
2. Login using the vSphere client to the vSphere server (or vCenter Server) where you want to install the Delphix Engine.
3. In the vSphere Client, click **File**.
4. Select **Deploy OVA Template**.
5. Browse to the OVA file.
6. Click **Next**.
7. Select a **hostname** for the Delphix Engine. This hostname will also be used in configuring the Delphix Engine network.
8. Select the **data center** where the Delphix Engine will be located.
9. Select the **cluster** and the **ESX host**.
10. Select one (1) **data store** for the **Delphix OS**. This datastore can be **thin-provisioned** and must have enough free space to accommodate the 127GB comprising the Delphix operating system.
11. Select four (4) or more **data stores** for Database Storage for the Delphix Engine. The Delphix Engine will stripe all of the Database Storage across these VMDKs, so for optimal I/O performance, each VMDK must be equal in size and be configured **Thick Provisioned - Eager Zeroed**. Additionally, these VMDKs should be distributed as evenly as possible across all four SCSI I/O controllers.
12. Select the **virtual network** you want to use. If using multiple physical NICs for link aggregation, you must use vSphere NIC teaming. Do not add multiple virtual NICs to the Delphix Engine itself. The Delphix Engine should use a single virtual network. For more information, see Optimal Network Architecture for the Delphix Engine
13. Click **Finish**. The installation will begin and the Delphix Engine will be created in the location you specified.
14. Once the installation has completed, power on the Delphix Engine and proceed with the initial system configuration as described in Setting Up Network Access to the Delphix Engine

> ⓘ If your source database is 4 TB, you probably need 4 TB of storage for the Delphix Engine. Add at least 4 data disks of similar size for the Delphix VM. For example: for a source database of 4 TB, create 4 VMDKs of 1 TB each.

For a full list of requirements and best practice recommendations, see Virtual Machine Requirements for VMware Platform

Setup network access to the Delphix engine

1. Power on the Delphix Engine and open the Console.
2. Wait for the Delphix Management Service and Delphix Boot Service to come online. This might take up to 10 minutes during the first boot. Wait for the large orange box to turn green.
3. Press any key to access the sysadmin console.
4. Enter `sysadmin@SYSTEM` for the username and `sysadmin` for the password.
5. You will be presented with a description of available network settings and instructions for editing.

```
Delphix Engine Network Setup

To access the system setup through the browser, the system must first be
configured for networking in your
environment. From here, you can configure the primary interface, DNS, hostname,
and default route. When DHCP is
configured, all other properties are derived from DHCP settings.

To see the current settings, run "get." To change a property, run "set =." To
commit your changes,
run "commit." To exit this setup and return to the standard CLI, run "discard."

    defaultRoute    IP address of the gateway for the default route -- for
 example, "1.2.3.4."

    dhcp            Boolean value indicating whether DHCP should be used for
 the primary interface. Setting this value
                    to "true" will cause all other properties (address,
hostname, and DNS) to be derived from the DHCP
                    response

    dnsDomain       DNS Domain -- for example, "delphix.com"

    dnsServers      DNS server(s) as a list of IP addresses -- for example,
"1.2.3.4,5.6.7.8."

    hostname        Canonical system hostname, used in alert and other logs --
for example, "myserver"

    primaryAddress  Static address for the primary interface in CIDR notation
-- for example, "1.2.3.4/22"

Current settings:

    defaultRoute: 192.168.1.1
    dhcp: false
    dnsDomain: example.com
```

```
    dnsServers: 192.168.1.1
    hostname: Delphix
    primaryAddress: 192.168.1.100/24
```

6. Configure the `hostname`. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set hostname=<hostname>
```

Note:
Use the same `hostname` you entered during the server installation.

7. Configure DNS. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set dnsDomain=<domain>
delphix network setup update *> set dnsServers=<server1-ip>[,<server2-ip>,...]
```

8. Configure either a static or DHCP address.
DHCP Configuration

```
delphix network setup update *> set dhcp=true
```

Static Configuration

```
delphix network setup update *> set dhcp=false
delphix network setup update *> set primaryAddress=<address>/<prefix-len>
```

Note:
The static IP address must be specified in CIDR notation (for example, `192.168.1.2/24`)

9. Configure a default gateway. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set defaultRoute=<gateway-ip>
```

10. Commit your changes. Note that you can use the `get` command prior to committing to verify your desired configuration.

```
delphix network setup update *> commit
Successfully committed network settings. Further setup can be done through the
browser at:

        http://<address>

Type "exit" to disconnect, or any other commands to continue using the CLI.
```

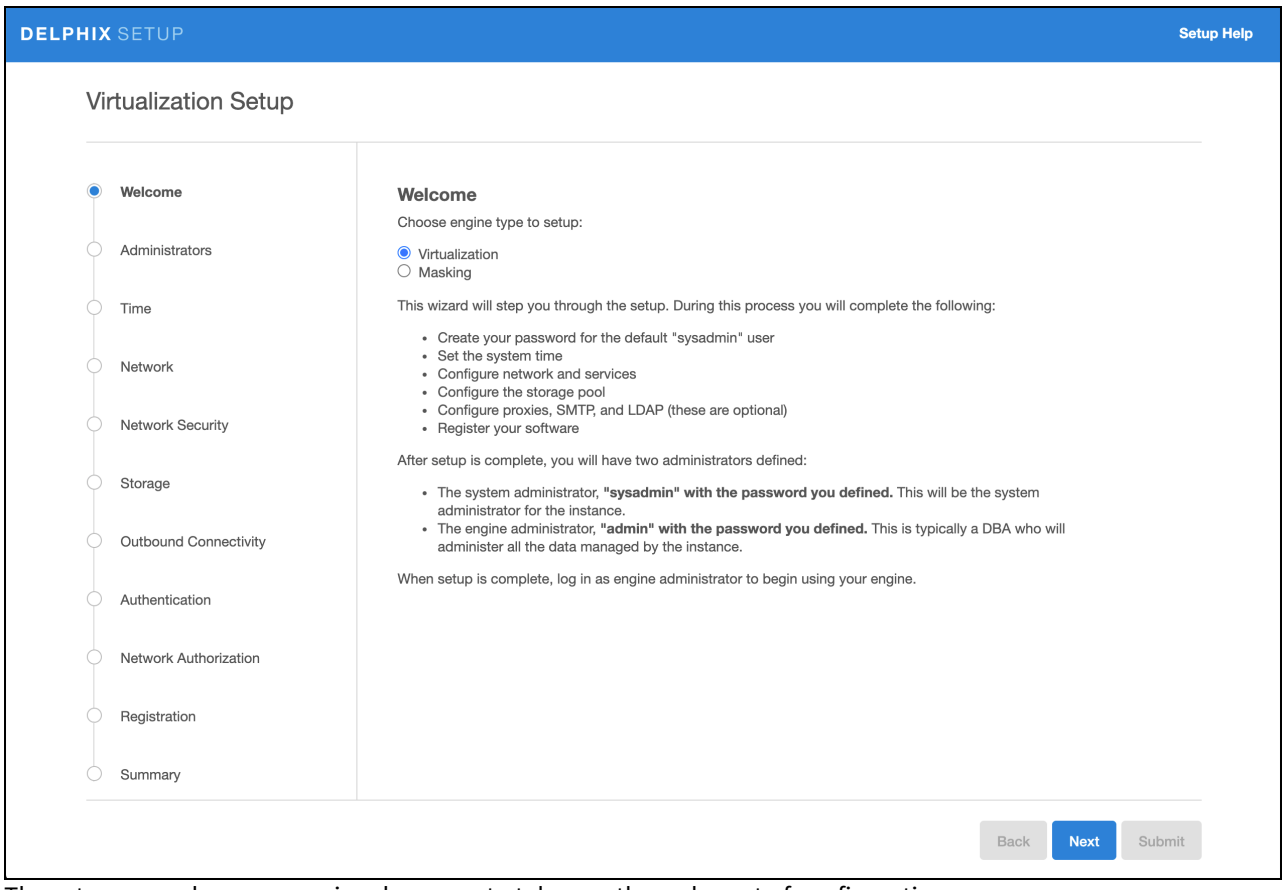
11. Check that you can now access the Delphix Engine through a Web browser by navigating to the displayed IP address, or hostname if using DNS.
12. Exit setup.

```
delphix> exit
```

Setting up the Delphix engine

Once you set up the network access for Delphix Engine, navigate to the Delphix Engine URL in your browser for server setup.

The welcome screen below will appear for you to begin your Delphix Engine setup.



The setup procedure uses a wizard process to take you through a set of configuration screens:

- Administrators
- Time
- Network
- Network Security
- Storage
- Outbound Connectivity
- Authentication
- Network Authorization
- Registration
- Summary

1. Connect to the Delphix Engine at `http:///<Delphix Engine>login/index.html#serverSetup`. The **Delphix Setup** application will launch when you connect to the server.

Enter your **sysadmin** login credentials, which initially defaults to the username **sysadmin**, with the initial default password of **sysadmin**. On first login, you will be prompted to change the initial default password.

2. Click **Next**.

Administrators

The Delphix Engine supports two types of administrators:

- System Administrator (**sysadmin**) - this is the engine system administrator. The sysadmin password is defined here**.**
- Engine Administrator (**admin)** - this is typically a DBA who will administer all the data managed by the engine.

On the Administrators tab, you set up the sysadmin password by entering an email address and password. The details for the admin are displayed for reference.

The default domain user created on Delphix Engines from 5.3.1 is known as **admin** instead of delphix_admin. When engines created before 5.3.1 are upgraded to 5.3.1 or later they will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then disabling the delphix_admin.

Time

The engine time is used as the baseline for setting policies that coordinate between virtual databases and external applications

Choose your option to setup system time in this section. For a Quick Start, simply set the time and your timezone. You can change this later.

Network

The initial out-of-the-box network configuration in the OVA file is set to use a single VMXNET3 network adapter.

You have already configured this in the initial configuration. Delphix supports more advanced configurations, but you can enable those later.

Storage

You should see the data storage VMDKs or RDMs you created during the OVA installation. Click **Next** to configure these for data storage.

Serviceability

Choose your options to configure serviceability settings.

For a Quick Start, accept the defaults. You can change this later.

Authentication service

Choose your options to configure authentication services.

For a Quick Start, accept the defaults. You can change this later.

Registration

If the Delphix Engine has access to the external Internet (either directly or through a web proxy), then you can auto-register the Delphix Engine:

1. Enter your **Support Username** and **Support Password**.

2. Click **Register**.

If external connectivity is not immediately available, you must perform manual registration.

1. Copy the **Delphix Engine registration code** in one of two ways:
    a. Manually highlight the registration code and copy it to clipboard. Or,
    b. Click **Copy Registration Code to Clipboard**.
2. Transfer the Delphix Engine's registration code to a workstation with access to the external network Internet. For example, you could e-mail the registration code to an externally accessible e-mail account.
3. On a machine with access to the external Internet, please use your browser to navigate to the Delphix Registration Portal at http://register.delphix.com.
4. Login with your Delphix support credentials (username and password).
5. Paste the **Registration Code**.
6. Click **Register**.

Although your Delphix Engine will work without registration, we strongly recommend that you register each engine as part of the setup. Failing to register the Delphix Engine will impact its supportability and security in future versions.

To regenerate the registration code for a Delphix Engine please refer to, Regenerating the Delphix Engine Registration Code. Delphix strongly recommends that you regenerate this code and re-register the engine regularly to maximize the Support Security of the Delphix Engine. Delphix recommends doing this every six months.

Summary

The final summary tab will enable you to review your configurations for System Time, Network, Storage, Serviceability, and Authentication.

1. Click the **Back** button to go back and to change the configuration for any of these server settings.
2. If you are ready to proceed, then click **Submit**.
3. Click **Yes** to confirm that you want to save the configuration.
4. Click **Setup** to acknowledge the successful configuration.
5. There will be a wait of several minutes as the Delphix Engine completes the configuration.

Requirements for HANA staging/target host and databases

Backup of the tenant must be present which needs to be recovered. Along with backup, metadata information generated by script create_backup_metadata.sh is also expected in the backup location. As we will be recovering a tenant through backup files the important thing while creating the dSource which should be taken care of is the user should have access to the backup directory where the backup file resides.

General staging Host Requirements

- HANA database must be installed and running.
- Read-write permission on the backup directory.
- There must be an operating system user which has HANA administrative user privileges <sid>adm:
    - The Delphix Engine must be able to make an SSH connection to the staging environment using the operating system user.
    - The operating system user must have read and execute privileges on the HANA binaries installed on the staging environment.
    - The operating system user must have read, write, and execute access to the HANA data directories on the staging environment.
- A plugin directory must exist on the staging host. The directory must have the following properties:
    - Should be writable by the operating system user described above.
    - Should have at least 256 MB of available storage.

- TCP/IP connectivity to and from the staging environment must be configured as described in General Network and Connectivity Requirements
- Hostname and IP address must be correctly configured.

Requirements for HANA target host and databases

A **target environment** is where virtualized databases run. A target environment must have a HANA instance installed and configured. The VDB will become a tenant on the target database instance.

General target environment requirements

1. The operating system of the target environment must match the staging environment.
2. The SAP HANA installation on the target environment must have the same or higher version than the staging environment.
3. There must be an operating system user with the following privileges:
    a. The Delphix Engine must be able to make an SSH connection to the target environment using the operating system user.
    b. The operating system user must have read and execute privileges on the HANA binaries installed on the target environment.
    c. The operating system user must have permission to run mount and unmount as the superuser via sudo with neither a password nor a TTY. See Sudo Requirements for the HANA 2.0 Plugin
4. Prior to the discovery, a plugin directory on the target environment must exist with the following properties:
    a. The plugin directory must be writable by the operating system user mentioned above.
    b. The plugin directory must have at least 256 MB of available storage.
5. There must be a mount point directory (for example, /mnt) that will be used as the base for mount points that are created when provisioning a VDB with the following properties:
    a. The mount point directory must be writable by the operating system user mentioned above.
    b. The mount point directory should be empty.
6. TCP/IP connectivity to and from the source environment must be configured as described in General Network and Connectivity Requirements.
7. The hostname and IP address must be correctly configured.
8. HANA instance on which the VDB creation is being done should have the empty ports to host the services for the tenant database. The default port number range for tenant databases is 3<instance>40—3<instance>99. This means that the maximum number of tenant databases that can be created per instance is 20. However, you can increase this by reserving the port numbers of further instances. You do this by configuring the property [multidb] reserved_instance_numbers in the global.ini file. The default value of this property is 0. If you change the value to 1, the port numbers of one further instance are available (for example, 30040—30199 if the first instance is 00). If you change it to 2, the port numbers of two further instances are available (for example, 30040—30299 if the first instance is 00). And so on.

Adding a HANA environment

Prerequisites

- Make sure that the HANA environment in question meets the requirements described in Requirements for HANA Staging Hosts and Databases and Requirements for HANA Target Hosts and Databases

Procedure

- Login to the **Delphix Management** application.
- Click **Manage**.
- Select **Environments**.

- Next to **Environments**, click the **Actions (...) menu and** select **Add Environment**.
- In the Add Environment dialog, In the **Host OS** select **Unix/Linux** on the menu.
- In **Server Type**, select **Standalone Host**.
- Click **Next**.
- Enter **Name** for the environment.
- Enter the **Host IP address** or **fully qualified hostname**.
- Enter the **SSH port**. The default value is 22.
- Enter an **OS Username** for the environment which should be ‘’.>’.>
- Select **Login Type**.
  Note:
  Currently, the Enterprise Password Vault option is not supported.
- Username and Password - enter the OS username and password
- Username and Public Key - enter the OS username.
- Password Vault - select from an existing Enterprise Password Vault

Using public key authentication
If you want to use public-key authentication for logging into your Unix-based environment, there are two options: use the engine's key pair or provide a key pair for this environment.
To use the engine's key pair:

1. Select **Public Key** for the **Login Type.**
2. Click **View Public Key**.
3. Copy the public key that is displayed, and append it to the end of your `~/.ssh/authorized_keys` file. If this file does not exist, you will need to create it.
   a. Run chmod 600 ~/.ssh/authorized_keys to allow only the file's owner to read and write to it (make sure the file is owned by the user).
   b. Run chmod 755 ~ to restrict access to the user's home directory so no other user may write to it.
   c. Run chmod 700 ~/.ssh so that others cannot write to it. The ~/.ssh directory cannot be writable by group or other users. Otherwise, authentication will fail.

As an alternative, you can provide a key pair specific for this environment via the API or CLI. See Option 2 in this CLI Cookbook article for instructions.

1. For Password, enter the password associated with the user in step 11.
2. Select **Public Key** for the **Login Type.**
3. Click **View Public Key**.
4. Copy the public key that is displayed, and append it to the end of your `~/.ssh/authorized_keys` file.
   If this file does not exist, you will need to create it.
   a. Run chmod 600 ~/.ssh/authorized_keys to allow only the file's owner to read and write to it (make sure the file is owned by the user).
   b. Run chmod 755 ~ to restrict access to the user's home directory so no other user may write to it.
   c. Run chmod 700 ~/.ssh so that others cannot write to it. The ~/.ssh directory cannot be writable by group or other users. Otherwise, authentication will fail.
5. For Password Login, click ‘Validate’ to test the username and password.
6. Enter a **Plugin Path.** The location specified must be a string without spaces. Ensure this directory path exists on the host in question. Environment discovery will fail if the directory does not exist on the host.
7. To provide your own Oracle Java select the **Provide my own JDK** checkbox and click **Next**.
8. In the Java Development Kit tab enter the absolute path to your Oracle JDK and click **Next**. For more information, see Providing Your Own Oracle Java
9. Click **Submit**. On success, the new environment will be added to the list in the Environments tab.
10. Select the “Databases” tab to view discovered HANA databases:

Details | Databases

Installations

Filter: none | Add Dataset Home

HANA 2.00.052.00.1599235305 /usr/sap/HDB/HDB00

INSTALLATION DETAILS

Allow Provisioning
Yes

System Administrator User
hdbadm

SAP Mount Path
/hana/shared

HANA Version
2.00.052.00.1599235305

Hostname
source

HANA Instance Number
00

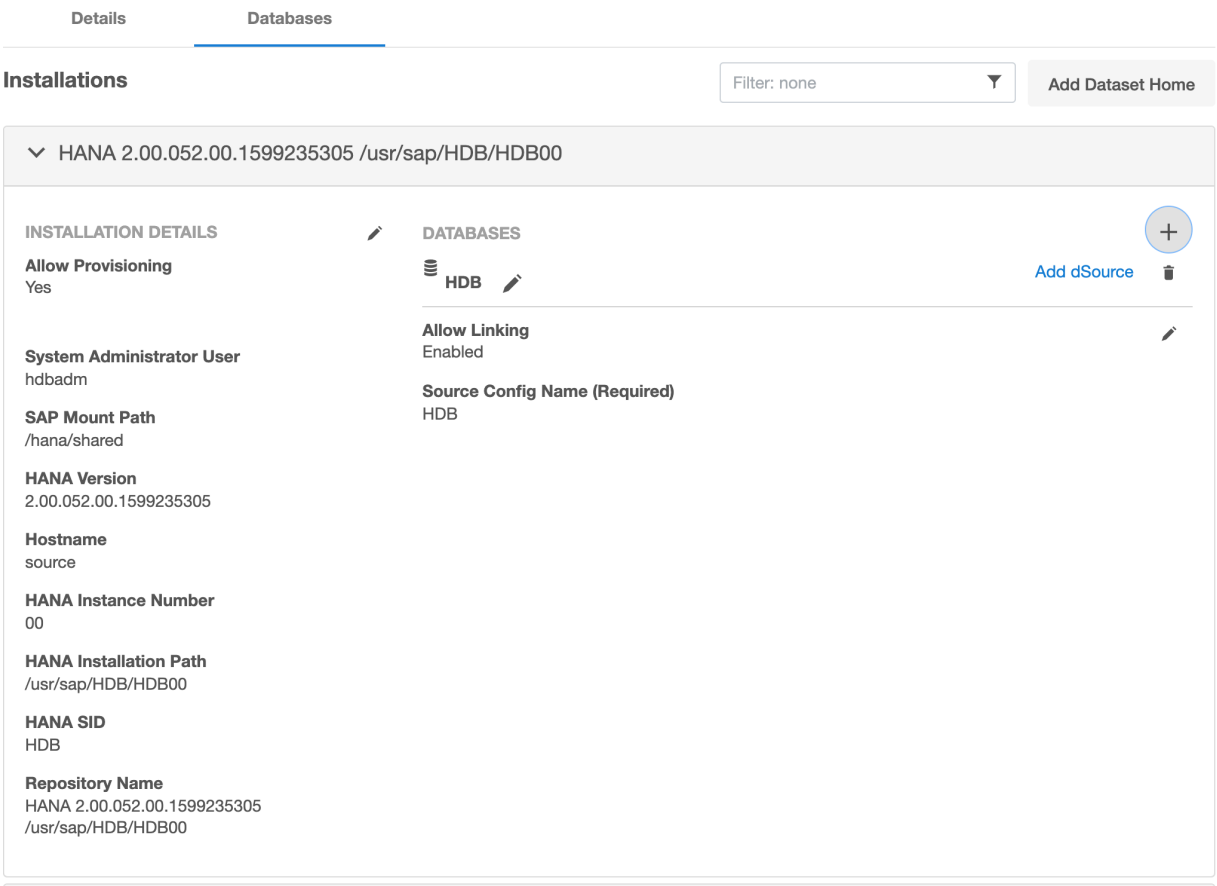
HANA Installation Path
/usr/sap/HDB/HDB00

HANA SID
HDB

Repository Name
HANA 2.00.052.00.1599235305
/usr/sap/HDB/HDB00

DATABASES

No databases found on installation.



Linking a HANA data source

This topic describes basic concepts behind the creation of a dSource for a HANA database using conventional pull and Staging push mechanisms.

> For Staging Push, only native/third-party backups are used to populate data in the staging database.

For Third-party Backups, users need to provide a “Backup Path” containing FULL and Incremental backups. These files will be copied to the Delphix Engine during the dSource linking process. In this scenario, the Delphix Engine will not require any credentials of the source database. Although credentials for the tenant database (for which backup has been placed) and systemdb of staging instances would be required.

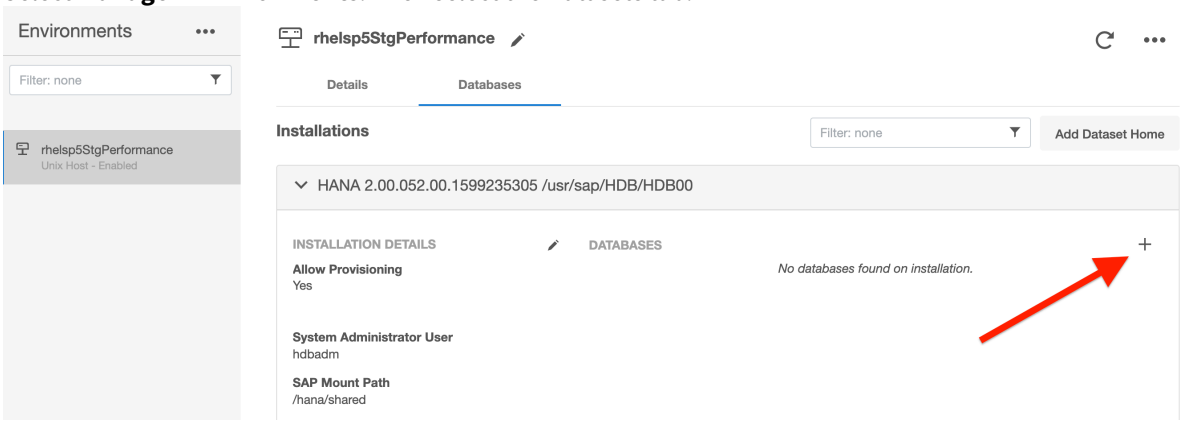
Prerequisites for customer provided backups ingestion mechanism

- Users need to run the create_backup_metadata.sh. This script will create the backupTimeInfo file at the same location where backups are kept. This script will gather all the required metadata information and will store that information in the backupTimeInfo file. This information will then get used by the plugin at the time of creating dSource.
- More details regarding the script have been included in the HANA 2.0 Plugin Tools

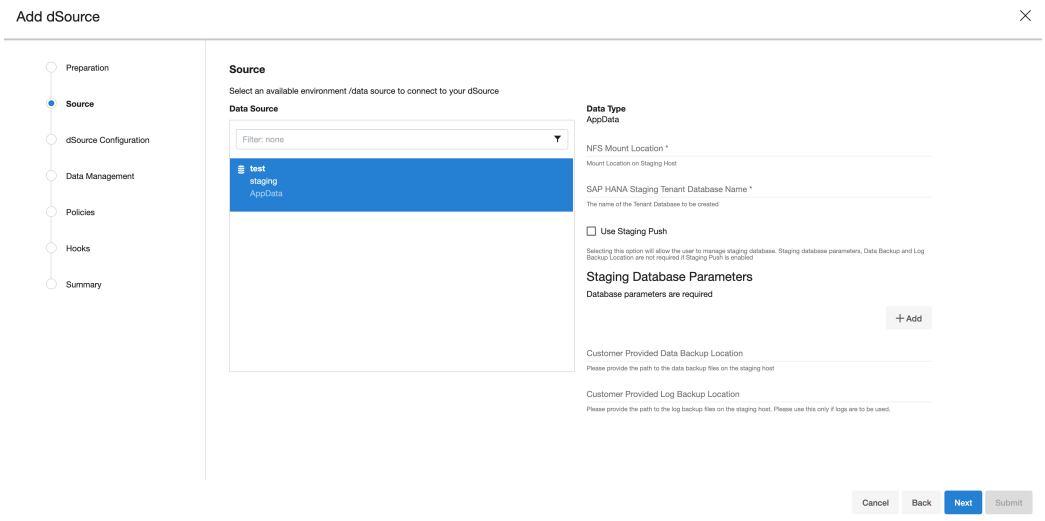
Procedure

1. Login to the Delphix Management application.

2. Select **Manage > Environments**. Then select the Datasets tab.

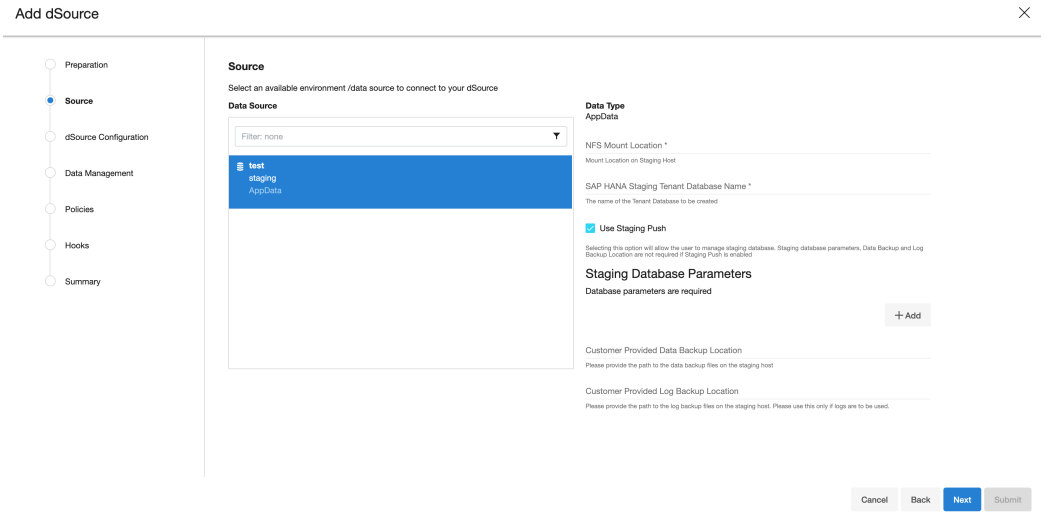


3. Create a source configuration on which to build the dSource. You can do this by clicking on the **+** symbol. The Source configuration can have any name.
4. In the **Add dSource** wizard, select the required source configuration.
5. After selecting the source, below are the parameters required to create the dSource.
    a. Parameters required for the conventional **pull** mechanism:
        i. Mount Location on Staging Host
        ii. SAP HANA Source Tenant Database Name
        iii. Source SystemDB User Name
        iv. Source SystemDB Password
        v. Source Tenant Database User Name
        vi. Source Tenant Database Password
        vii. Customer Provided Data Backup Location: This represents the path to the location of the customer-provided backup files.
        viii. Customer Provided Log Backup Location: This represents the path to log the location of the customer-provided log backup files. This parameter is optional.



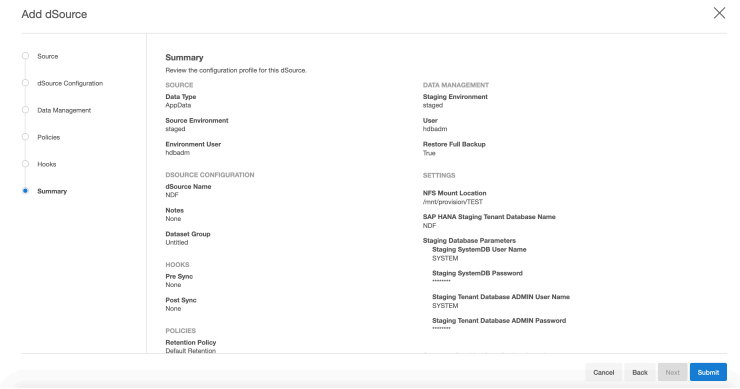
    b. Parameters required for the **Staging push** mechanism
        i. Mount Location on Staging Host
        ii. SAP HANA Source Tenant Database Name
        iii. Enable the **Staging Push** feature by selecting the checkbox "**Use Staging Push**". With this feature, the user handles the Source Tenant database and its state on their own and you don't

need to specify any other staging database parameters, data backup, and log backup locations. If this check box is selected, the HANA plugin will not invoke the tenant creation and recovery operation and if this checkbox is not selected then the default pull approach will be used.



c. Click **Next.**

6. Enter a name and select a group for your dSource. Adding a dSource to a dataset group lets you set Delphix Domain user permissions for that database and its objects, such as snapshots. See the topics under Users and Groups for more information.
7. Configure the SnapSync and Retention Policy. The SnapSync policy can be set to 'None' for customers who wish not to run a scheduled SnapSync job.
8. Optionally, users can provide the bash script as a plugin-defined hook. Both Pre-Snapshot and Post-SnapShot hooks can be used.
9. Select **Next.**
10. At the end of the workflow, review the dSource Configuration in the Summary pane as shown below:



Once the dSource linking job is submitted it is possible to monitor progress by selecting **Active Jobs** in the top menu bar. On successful completion, the new dSource will be visible in the list of Datasets under its assigned group.

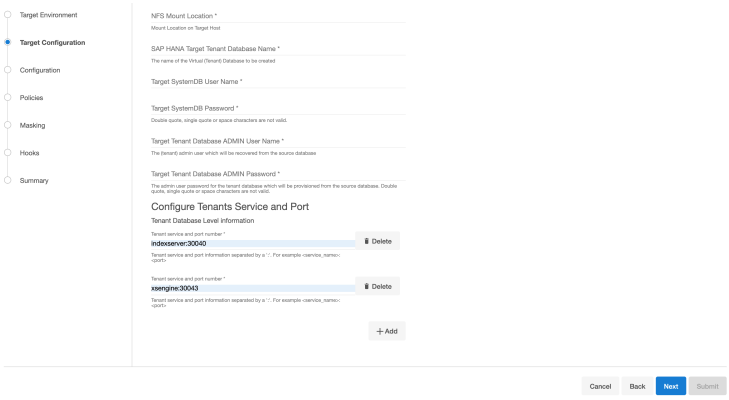
After dSource creation using the Staging Push mechanism, an empty snapshot will be created but users can not create VDB's from the same. Also, before creating the subsequent snapshots, users will have to create the database and point the database to use the Delphix provided mount point.

Provisioning a HANA VDB

Prerequisites

- The provision requires a linked dSource built off a physical source tenant, as described in Linking a HANA dSource, or a VDB.
- The HANA target environment should adhere to the Requirements for HANA Hosts and Databases
- Operating system users and database users should be configured.

VDB creation UI



Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select a **dSource**.
5. Select a **snapshot** from which you want to provision.
   Note:
   If the first or empty dSource Snapshot that was created using the Staging Push mechanism is detected by the plugin and an error message is displayed. This check is done for every VDB creation operation.
6. Click the **Provision VDB** icon to open the Provision VDB wizard.
7. Select a target environment from the left pane.
8. The HANA 2.0 Plugin supports one installation on the same host, hence the value in the **Installation** dropdown will be already populated. Multiple drop-down options should not be available.
9. Set the **Environment User** to be the Instance Owner. Note: The picking of instance owner is only possible if you have multiple environment users set on that host.
10. Click **Next**.
11. Target Configuration pane you will need to specify the following:
    Note:
    '*****' indicates the mandatory fields.
    a. NFS Mount Location*: Delphix recommends setting the trailing directory to match the VDB name to delineate mount points and support administration and troubleshooting. In the example shown, the **NFS Mount Location** is /mnt/provision/dev where dev matches the **SAP HANA Target Tenant Database Name**.
    b. SAP HANA Target Tenant Database Name*: This refers to the target tenant database name which will be created upon VDB creation.
    c. Target SystemDB User Name*: This refers to the SystemDB username in the target database.
    d. Target SystemDB Password*: This refers to the SystemDB password in the target database.
    e. Target Tenant Database ADMIN User Name*: This refers to the username of the tenant system user.

f. Target Tenant Database ADMIN Password*: This refers to the password of the tenant system user.
Note:
The SYSTEM username and password derived from the physical source database should be entered during the provision of a VDB because those credentials will be recovered during provision.
g. Configure Tenants Service and Port: This refers to the list of tenant services and port numbers in the following format.

```
<service>:<port>
For example, indexserver:30043
```

You can click "**+Add**" to add more services.
Note:
- This is not a mandatory field. If a user doesn't provide a value to this, then the plugin will proceed with the above fields and it will allocate the service and port on its own.
- If an invalid service is provided, the plugin will proceed with input, but will eventually fail while creating the service and will display an error on the Delphix Engine UI.

12. In the configuration section
    a. Set the vFile Name or accept the default, which is a random string. Delphix strongly recommends applying a consistent naming convention. We suggest this name should match the Target Tenant Name from the previous pane. Maintaining a consistent naming convention will support administration and troubleshooting activities. In the example shown, the vFile Name “dev” matches the actual tenant name and mount point from the previous pane.
    b. Select a **Target Group** for the VDB.Click the **Add Dataset Group** icon to add a new group if required.
    c. Set **Auto VDB Restart** option to automatically restart the VDB, where the target host is rebooted. Delphix recommends enabling this setting.
13. Click **Next**.
14. VDB **Snapshot Policy**. Select the Snapshot policy if an automated VDB snapshot schedule is required. Optionally, the Snapshot policy may be disabled by selecting “None”.
15. Click **Next**.
16. **Masking**. This page offers the opportunity to link the provision job to a masking job.
17. **Hooks**. Run a customized shell script via a hook operation at a particular moment in the provision operation.
18. Click **Next**.
19. **Summary.** Presents an opportunity to review all the information as a part of the initial provision workflow.
20. Click **Submit**.Once a provision job is submitted it is possible to monitor its progress by selecting “Active Jobs” in the top menu bar. On successful completion, VDB will be visible in the list of Datasets under its assigned group.

Next steps

Congratulations! You have provisioned your first virtual database!

Now, perform some simple functional tests with your application. You can connect your app to the VDB using standard TNS/JDBC techniques. Delphix has already registered the VDB for you on the target listener.

We suggest the following next steps:

1. Drop a table and use the VDB Rewind feature to test the recovery of your VDB.
2. Take a snapshot of your dSource and refresh your VDB to quickly get fresh production data.
3. Provision a new VDB from your VDB to test the sharing data quickly with other teams.
4. Mask your new VDB to protect sensitive data. Provision of new VDBs from that masked VDB to quickly provide safe data to development and QA teams.

SAP HANA support and requirements

This section covers the following topics:

- Requirements for SAP HANA staging hosts and databases
- Requirements for SAP HANA target hosts and databases
- Network and connectivity requirements for SAP HANA environments
- Sudo requirements for the SAP HANA 2.0 plugin

To view SAP HANA support matrix, see SAP HANA matrix.

Requirements for SAP HANA staging hosts and databases

Backup of the tenant must be present which needs to be recovered. Along with backup, metadata information generated by script create_backup_metadata.sh is also expected in the backup location. As we will be recovering a tenant through backup files the important thing while creating the dSource which should be taken care of is the user should have access to the backup directory where the backup file resides.

General staging Host Requirements

- HANA database must be installed and running.
- Read-write permission on the backup directory.
- There must be an operating system user which has HANA administrative user privileges :
    - The Delphix Engine must be able to make an SSH connection to the staging environment using the operating system user.
    - The operating system user must have read and execute privileges on the HANA binaries installed on the staging environment.
    - The operating system user must have read, write, and execute access to the HANA data directories on the staging environment.
- A plugin directory must exist on the staging host. The directory must have the following properties:
    - Should be writable by the operating system user described above.
    - Should have at least 256 MB of available storage.
- TCP/IP connectivity to and from the staging environment must be configured as described in General Network and Connectivity Requirements
- Hostname and IP address must be correctly configured.

⚠ Non-database users/Low Privileged users/Users for which privileges need to be elevated, are not supported for performing Delphix operations.

Requirements for SAP HANA target hosts and databases

This section describes user privileges and other requirements for HANA target hosts and databases collectively referred to as the target environment.

A **target environment** is where virtualized databases run. A target environment must have a HANA instance installed and configured. The VDB will become a tenant on the target database instance.

General target environment requirements

1. The operating system of the target environment must match the staging environment.
2. The SAP HANA installation on the target environment must have the same or higher version than the staging environment.
3. There must be an operating system user with the following privileges:
    a. The Delphix Engine must be able to make an SSH connection to the target environment using the operating system user.
    b. The operating system user must have read and execute privileges on the HANA binaries installed on the target environment.
    c. The operating system user must have permission to run mount and unmount as the superuser via sudo with neither a password nor a TTY. See Sudo Requirements for the HANA 2.0 Plugin
4. Prior to the discovery, a plugin directory on the target environment must exist with the following properties:
    a. The plugin directory must be writable by the operating system user mentioned above.
    b. The plugin directory must have at least 256 MB of available storage.
5. There must be a mount point directory (for example, /mnt) that will be used as the base for mount points that are created when provisioning a VDB with the following properties:
    a. The mount point directory must be writable by the operating system user mentioned above.
    b. The mount point directory should be empty.
6. TCP/IP connectivity to and from the source environment must be configured as described in General Network and Connectivity Requirements.
7. The hostname and IP address must be correctly configured.
8. HANA instance on which the VDB creation is being done should have the empty ports to host the services for the tenant database. The default port number range for tenant databases is 3<instance>40—3<instance>99. This means that the maximum number of tenant databases that can be created per instance is 20. However, you can increase this by reserving the port numbers of further instances. You do this by configuring the property [multidb] reserved_instance_numbers in the global.ini file. The default value of this property is 0. If you change the value to 1, the port numbers of one further instance are available (for example, 30040—30199 if the first instance is 00). If you change it to 2, the port numbers of two further instances are available (for example, 30040—30299 if the first instance is 00). And so on.

⚠ Non-database users/Low Privileged users/Users for which privileges need to be elevated, are not supported for performing Delphix operations.

Network and connectivity requirements for SAP HANA environments

Please see General Network and Connectivity Requirements and Architecture Best Practices for Network Configuration for detailed network and connectivity requirements for the Delphix Engine, including connection requirements, port allocation, and firewall and Intrusion Detection System (IDS) considerations.

Sudo requirements for the SAP HANA 2.0 plugin

The following sudo privileges are requirements for the HANA 2.0 Plugin.

Privilege	Source	Target	Rationale
mkdir/rmdir	Not Required	Required	Delphix dynamically makes and removes directories under the provisioning directory during VDB operations. This privilege is optional, provided the provisioning directory permissions allow the delphix os user to make and remove directories.
mount/umount	Not Required	Required	Delphix dynamically mounts and unmounts directories under the provisioning directory during VDB operations. This privilege is required because `mount` and `umount` are typically reserved for superuser.

It is required to specify the NOPASSWD qualifier within the "sudo" configuration file. This ensures that the "sudo" command does not demand the entry of a password.

Example with user hdbadm

```
Defaults:hdbadm !requiretty
hdbadm ALL=NOPASSWD:/sbin/mount, /sbin/umount, /bin/mkdir, /bin/rmdir
```

Example of limiting `sudo` access for the Delphix OS user

In situations where security requirements prohibit giving the Delphix user root privileges to mount, unmount, make directory, and remove directory on the global level, it is possible to configure the `sudoers` file to provide these privileges only on specific mount points or from specific Delphix Engines, as shown in the below example.

```
Defaults:hdbadm !requiretty
hdbadm ALL=(root) NOPASSWD: \
/bin/mount    *<delphix-server-name>*   /mnt/provision/*, \
/bin/umount    /mnt/provision/*, \
/bin/mkdir * /mnt/*
```

that the following example is for illustrative purposes and the sudo file configuration options are subject to change.

Example 1

This example restricts the **delphix_os** user's use of `sudo` privileges to the directory `/hana`.

Note that wildcards are allowed for the options on `mount` and `umount` because those commands expect a fixed number of arguments after the options. The option wildcard on the `mount` command also makes it possible to specify the file-system being mounted from the Delphix Engine.

However, wildcards are not acceptable on `mkdir` and `rmdir` because they can have any number of arguments after the options. For those commands, you must specify the exact options (`-p` , `-p -m 755`) used by the Delphix Engine.

Example /etc/sudoers File Configuration on the Target Environment for sudo Privileges on the VDB Mount Directory Only (Linux OS)

```
Defaults:delphix_os !requiretty
delphix_os ALL=(root) NOPASSWD: \
/bin/mount  *        /hana/*, \
/bin/umount *        /hana/*, \
/bin/umount          /hana/*, \
/bin/mkdir -p        /hana/*, \
/bin/mkdir -p -m 755 /hana/*, \
/bin/mkdir           /hana/*, \
/bin/rmdir           /hana/*
```

Example 2

This example restricts the **delphix_os** user's use of `sudo` privileges to the directory / `hana` , restricts the mount commands to a specific Delphix Engine hostname and IP, and does not allow user-specified options for the `umount` command.

This configuration is more secure, but there is a tradeoff with deployment simplicity. This approach would require a different sudo configuration for targets configured for different Delphix Engines.

A Second Example of Configuring the /etc/sudoers File on the Target Environment for Privileges on the VDB Mount Directory Only, and Allows Mounting Only from a Single Server (Linux OS)

```
Defaults:delphix_os !requiretty
delphix_os ALL=(root) NOPASSWD: \
/bin/mount           <delphix-server-name>* /hana/*, \
/bin/mount *         <delphix-server-name>* /hana/*, \
/bin/mount           <delphix-server-ip>*   /hana/*, \
/bin/mount *         <delphix-server-ip>*   /hana/*, \
/bin/mount "", \
/bin/umount          /hana/*, \
/bin/umount *        /hana/*, \
/bin/mkdir [*]       /hana/*, \
/bin/mkdir           /hana/*, \
/bin/mkdir -p        /hana/*, \
/bin/mkdir -p -m 755 /hana/*, \
/bin/rmdir           /hana/*
```

Managing SAP HANA environments and hosts

This section covers the following topics:

Workflow for SAP HANA environments

A staging environment must be added to the Delphix Engine before linking. Similarly, a compatible target environment must be added to the Delphix Engine in order to provide a location to provision to.

Installation of the HANA Plugin is required before the discovery of a HANA environment can occur.

Once an environment is added to the Delphix Engine, environment discovery takes place. Environment discovery is a process that enables the HANA Plugin to determine HANA installation details on a host. The same Environment Discovery process can be repeated during an Environment Refresh in order to detect new HANA installations.

Adding a SAP HANA environment

This topic describes how to add a HANA environment.

Prerequisites

- Make sure that the HANA environment in question meets the requirements described in Requirements for HANA Staging Hosts and Databases and Requirements for HANA Target Hosts and Databases

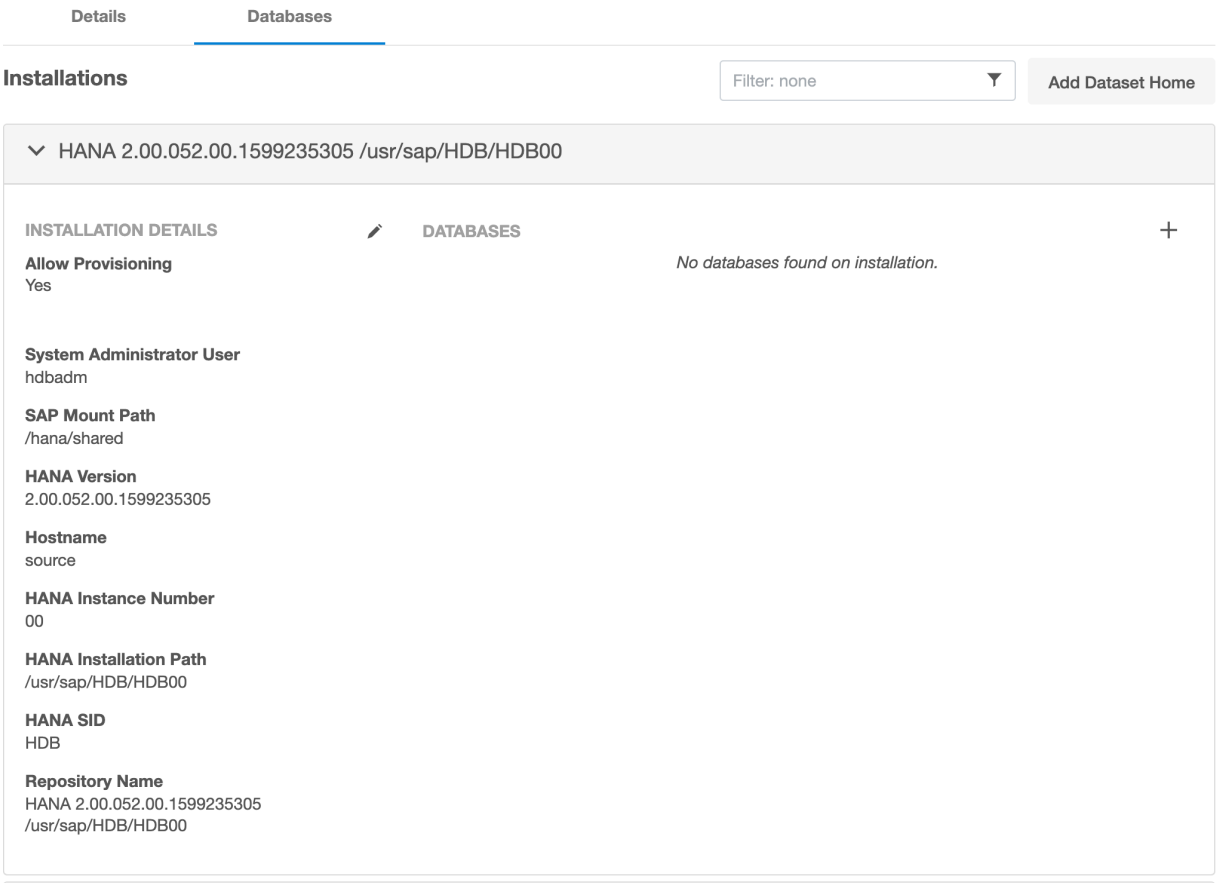
Procedure

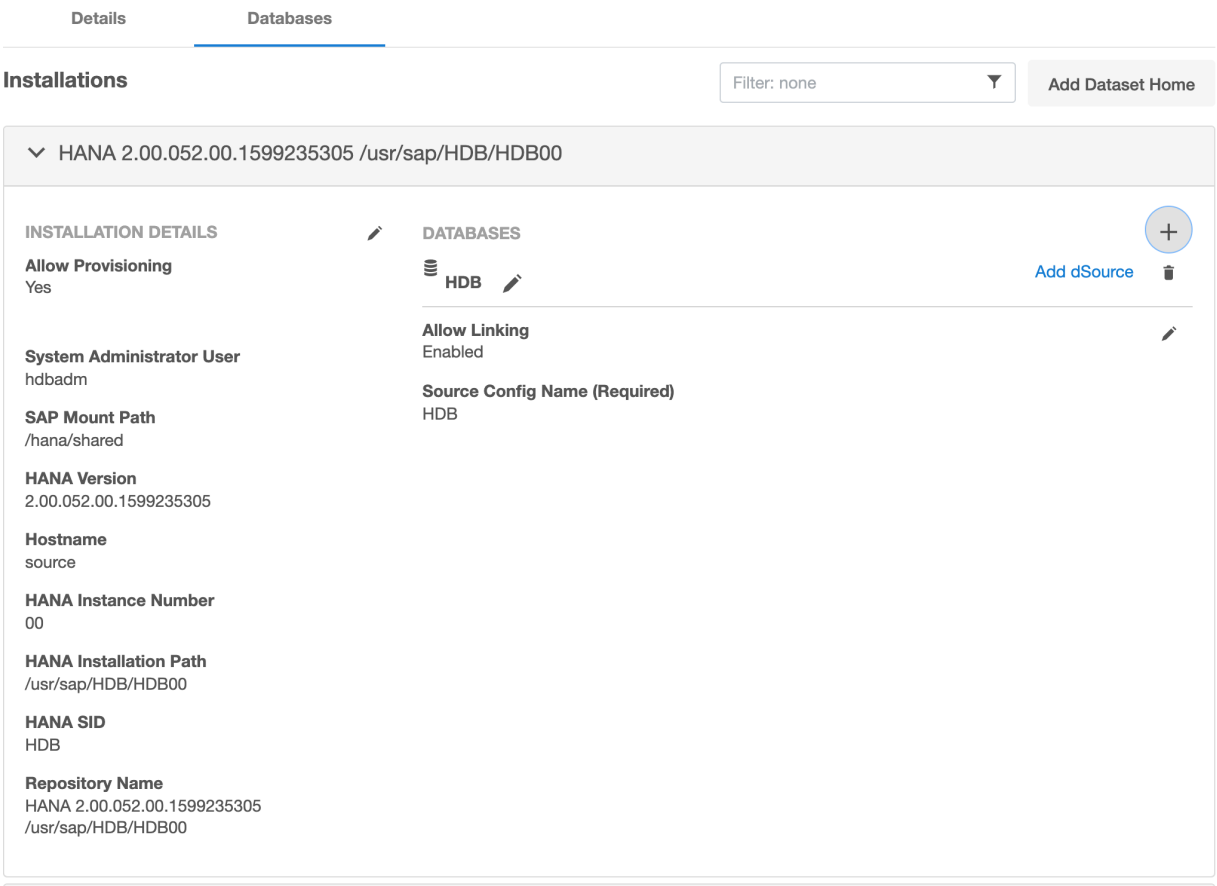
- Login to the **Delphix Management** application.
- Click **Manage**.
- Select **Environments**.
- Next to **Environments**, click the **Actions (...) menu and** select **Add Environment**.
- In the Add Environment dialog, In the **Host OS** select **Unix/Linux** on the menu.
- In **Server Type**, select **Standalone Host**.
- Click **Next**.
- Enter **Name** for the environment.
- Enter the **Host IP address** or **fully qualified hostname**.
- Enter the **SSH port**. The default value is 22.
- Enter an **OS Username** for the environment which should be ‘’.>’.>
- Select **Login Type**.
  Note:
  Currently, the Enterprise Password Vault option is not supported.
- Username and Password - enter the OS username and password
- Username and Public Key - enter the OS username.
- Password Vault - select from an existing Enterprise Password Vault

> **Using public key authentication**
> If you want to use public-key authentication for logging into your Unix-based environment, there are two options: use the engine's key pair or provide a key pair for this environment.
> To use the engine's key pair:
> 1. Select **Public Key** for the **Login Type.**
> 2. Click **View Public Key**.
> 3. Copy the public key that is displayed, and append it to the end of your `~/.ssh/authorized_keys` file. If this file does not exist, you will need to create it.
>   a. Run chmod 600 ~/.ssh/authorized_keys to allow only the file's owner to read and write to it (make sure the file is owned by the user).
>   b. Run chmod 755 ~ to restrict access to the user's home directory so no other user may write to it.
>   c. Run chmod 700 ~/.ssh so that others cannot write to it. The ~/.ssh directory cannot be writable by group or other users. Otherwise, authentication will fail.
>
> As an alternative, you can provide a key pair specific for this environment via the API or CLI. See CLI Cookbook: VDBs for instructions.

1. For Password, enter the password associated with the user in step 11.
2. Select **Public Key** for the **Login Type.**
3. Click **View Public Key**.
4. Copy the public key that is displayed, and append it to the end of your `~/.ssh/authorized_keys` file. If this file does not exist, you will need to create it.

   a. Run chmod 600 ~/.ssh/authorized_keys to allow only the file's owner to read and write to it (make sure the file is owned by the user).
   b. Run chmod 755 ~ to restrict access to the user's home directory so no other user may write to it.
   c. Run chmod 700 ~/.ssh so that others cannot write to it. The ~/.ssh directory cannot be writable by group or other users. Otherwise, authentication will fail.
5. For Password Login, click 'Validate' to test the username and password.
6. Enter a **Plugin Path.** The location specified must be a string without spaces. Ensure this directory path exists on the host in question. Environment discovery will fail if the directory does not exist on the host.
7. To provide your own Oracle Java select the **Provide my own JDK** checkbox and click **Next**.
8. In the Java Development Kit tab enter the absolute path to your Oracle JDK and click **Next**. For more information, see Providing Your Own Oracle Java
9. Click **Submit**. On success, the new environment will be added to the list in the Environments tab.
10. Select the "Databases" tab to view discovered HANA databases:





Post-requisites

To view information about an environment after you have created it:

1. Click **Manage**.
2. Select **Environments**.
3. Select the **environment name**.

Editing SAP HANA environment details

This topic describes how to edit the details of an environment. Once an environment has been added it is possible to change any of the following values:

- Environment OS user
- Host IP Address/Name
- SSH port
- Path to plugin location

Procedure

1. Login to the **Delphix Management** application with admin credentials or as the owner of an environment.
2. Click **Manage**.
3. Select **Environments**.
4. In the **Environments** panel, click the name of an environment to view its details.
5. Next to **Environment Detail**, click the **Pencil** icon to edit.
6. Click the **Check** icon to save your edits.

Changing the hostname or IP address for SAP HANA staging and target environments

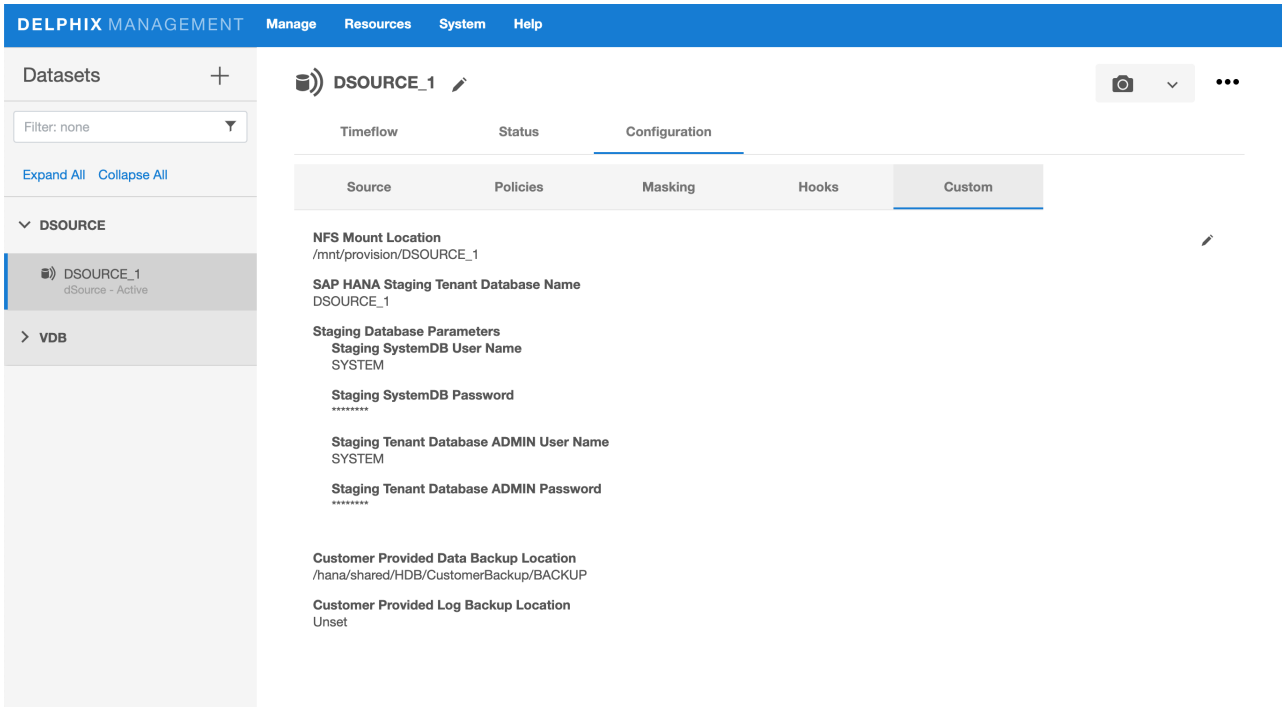
Below is the procedure to change the Host Address of the HANA Staging and Target Environments (If the Host address of the underlying staging and target server has also changed):

Procedure

1. Disable the dSources.
2. Disable the VDBs.
3. Login to the Delphix Management application.
4. Click **Manage**.
5. Select **Environments**.
6. Click on the existing environment name you want to modify and open the environment information screen.
7. In the Attributes section, click the **Pencil** icon located Next to it.
8. Update the Host address to the new one and save the results by clicking on .
9. The New Host address will now be updated.
10. Re-enable the dSources.
11. Re-enable the VDBs.

Steps to modify SAP HANA dSource properties

Select the dSource and then the dSource **Configuration** tab to modify dSource properties. For example, to change the **Source SystemDB Username** or **Password**, select the **Dataset**, **Configuration,** and then the **Custom** tab as shown below:



Linking data sources and syncing data with SAP HANA

This section covers the following topics:

- SAP HANA dSource ingestion
- Linking SAP HANA data sources
- Staging push implementation details
- Advanced data management settings for SAP HANA dSources
- Working with SAP HANA dSource snapshots
- Source sizing implementation for SAP HANA data sources

SAP HANA dSource ingestion

This release of the HANA 2.0 plugin uses Native Backups/Third-party backups to stage a dSource.

An External Backup is a form of HANA Native Backup that was created by users outside of Delphix. The necessary backup files must exist prior to the dSource operation and must be prepared by the user prior to ingestion by Delphix. In particular, a Delphix-supplied script must be run to persist vital backup metadata to Delphix during the dSource creation process. The backup files must be created on the source HANA instance must be made visible to the Staging host where the dSource is being created (ex: via shared storage, scp, etc).

A new data ingestion mechanism, Staging Push has been introduced that will help users to push data into the Delphix provided mount point on their own.

There are two mechanisms to create dSource on the staging server.

- Pull architecture, where the HANA plugin pulls the data into the Delphix provided mount location.
- Staging Push architecture, where the user pushes the data into the Delphix provided mount point on their own.

Linking SAP HANA data sources

This topic describes basic concepts behind the creation of a dSource for a HANA database using conventional pull and Staging push mechanisms.

> For Staging Push, only native/third-party backups are used to populate data in the staging database.

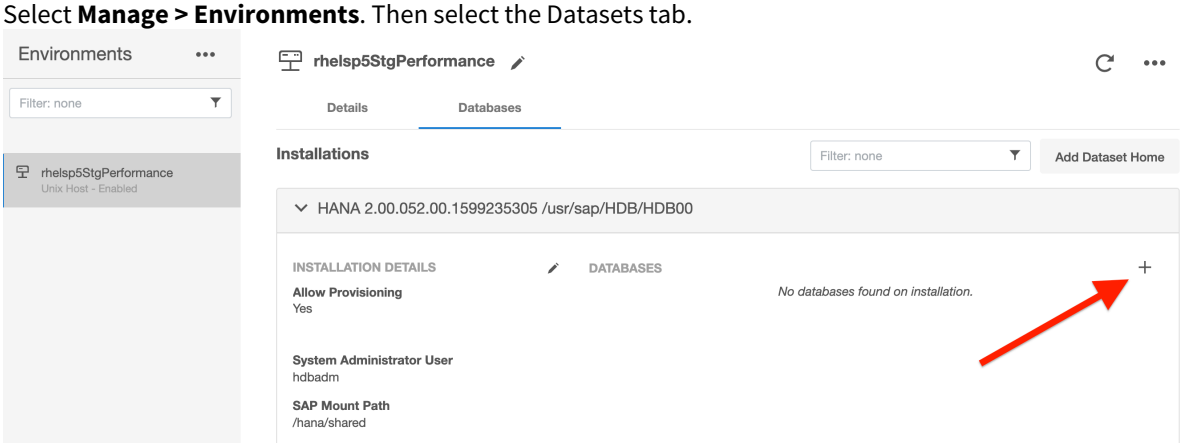
For Third-party Backups, users need to provide a “Backup Path” containing FULL and Incremental backups. These files will be copied to the Delphix Engine during the dSource linking process. In this scenario, the Delphix Engine will not require any credentials of the source database. Although credentials for the tenant database (for which backup has been placed) and system database of staging instances would be required.

Prerequisites for customer provided backups ingestion mechanism

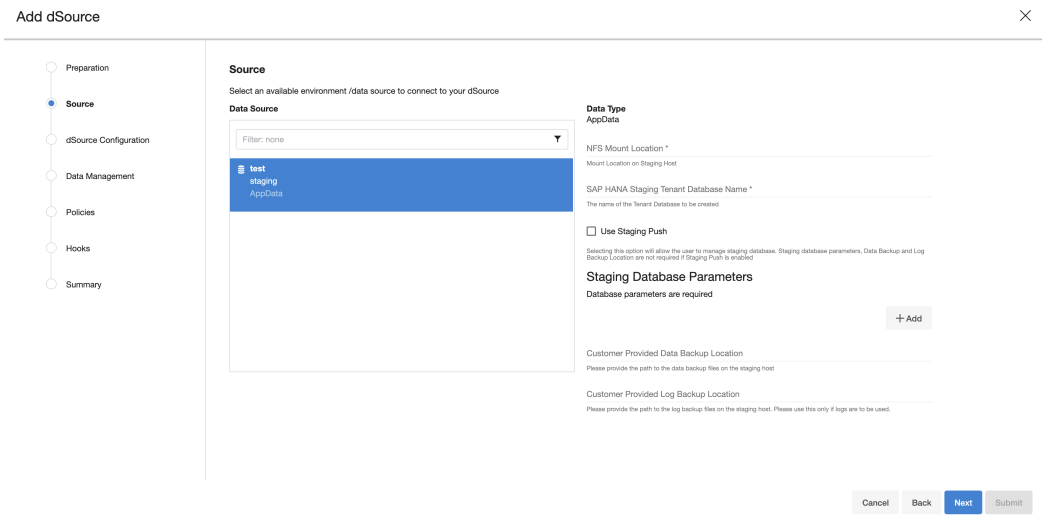
- Users need to run the create_backup_metadata.sh. This script will create the backupTimeInfo file at the same location where backups are kept. This script will gather all the required metadata information and will store that information in the backupTimeInfo file. This information will then get used by the plugin at the time of creating dSource.
- More details regarding the script have been included in the HANA 2.0 Plugin Tools

Procedure

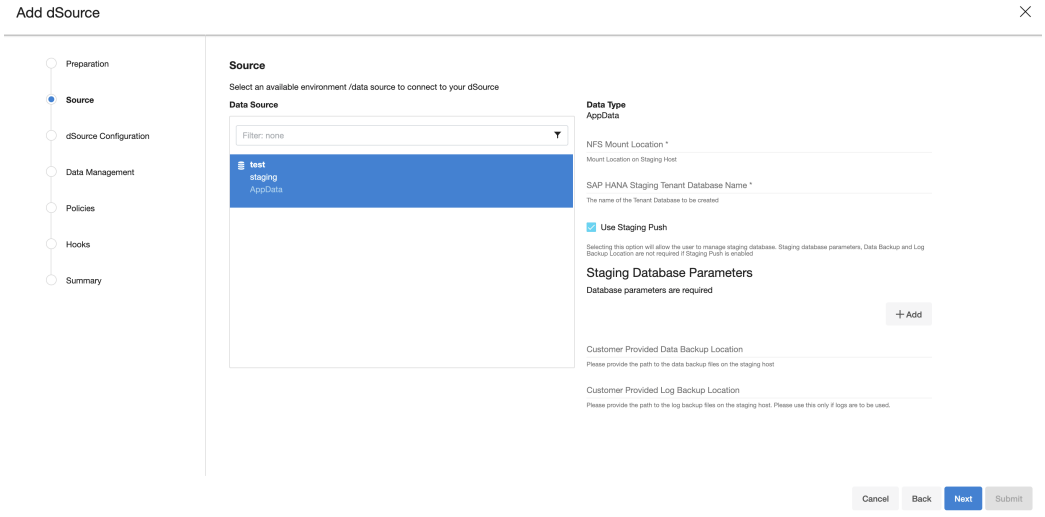
1. Login to the Delphix Management application.
2. Select **Manage > Environments**. Then select the Datasets tab.



3. Create a source configuration on which to build the dSource. You can do this by clicking on the **+** symbol. The Source configuration can have any name.
4. In the **Add dSource** wizard, select the required source configuration.
5. After selecting the source, below are the parameters required to create the dSource.
    a. Parameters required for the conventional **pull** mechanism:
        i. Mount Location on Staging Host
        ii. SAP HANA Source Tenant Database Name
        iii. Source SystemDB User Name
        iv. Source SystemDB Password
        v. Customer Provided Data Backup Location: This represents the path to the location of the customer-provided backup files.
        vi. Customer Provided Log Backup Location: This represents the path to log the location of the customer-provided log backup files. This parameter is optional.

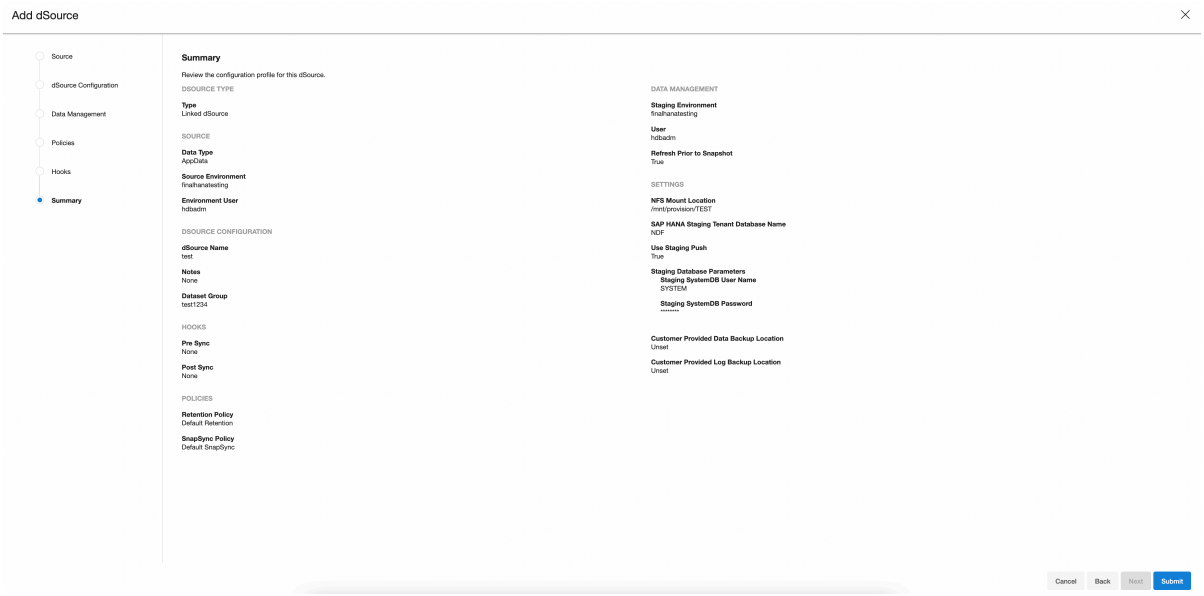


b. Parameters required for the **Staging push** mechanism
   i. Mount Location on Staging Host
   ii. SAP HANA Source Tenant Database Name
   iii. Enable the **Staging Push** feature by selecting the checkbox "**Use Staging Push**". With this feature, the user handles the Source Tenant database and its state on their own and you don't need to specify any other staging database parameters, data backup, and log backup locations. If this check box is selected, the HANA plugin will not invoke the tenant creation and recovery operation and if this checkbox is not selected then the default pull approach will be used.



c. Click **Next.**

6. Enter a name and select a group for your dSource. Adding a dSource to a dataset group lets you set Delphix Domain user permissions for that database and its objects, such as snapshots. See the topics under Users and Groups for more information.
7. Configure the **SnapSync** and **Retention Policy**. The SnapSync policy can be set to 'None' for customers who wish not to run a scheduled SnapSync job.
8. Optionally, users can provide the bash script as a plugin-defined hook. Both Pre-Snapshot and Post-SnapShot hooks can be used.
9. Select **Next.**
10. At the end of the workflow, review the **dSource Configuration** in the **Summary** pane as shown below:



Once the dSource linking job is submitted it is possible to monitor progress by selecting **Active Jobs** in the top menu bar. On successful completion, the new dSource will be visible in the list of Datasets under its assigned group.

After dSource creation using the Staging Push mechanism, an empty snapshot will be created but users can not create VDB's from the same. Also, before creating the subsequent snapshots, users will have to create the database and point the database to use the Delphix provided mount point.

Staging push implementation details

This topic provides implementation details of the Staging Push for the HANA plugin. Staging Push will help users to push data into the Delphix provided mount point on their own.

Staging Push gives users control over some Staging DB processes. It will give control of the staging database to the user to pull from any backup provider. Staging database files will be stored on Delphix Storage. Delphix will still be the one snapshotting the underlying data files, and gathering the metadata required to provision from the snapshot.

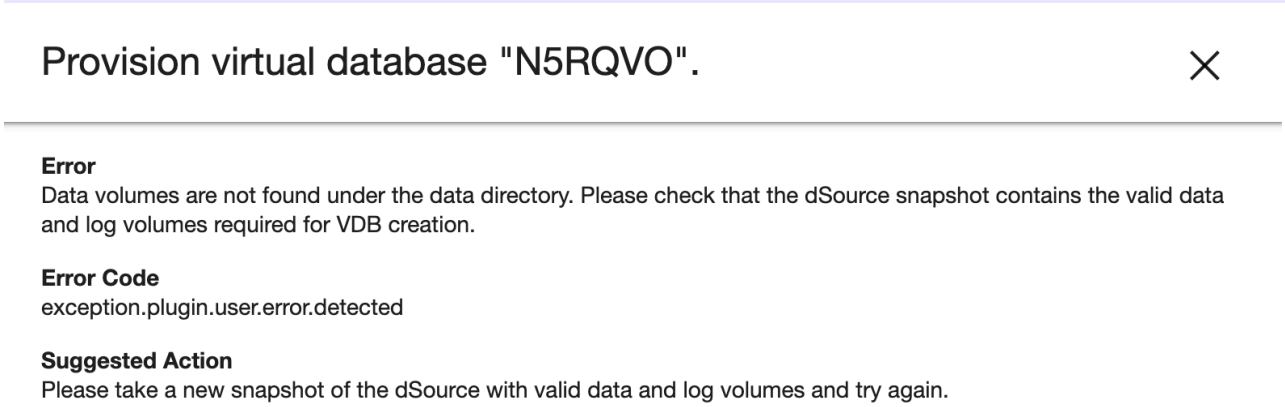
The below steps show how to create a dSource using the Staging Push mechanism.

dSource creation

The first snapshot created as a part of dSource creation contains data and log directories within the Delphix created mount point. These directories are empty for the first snapshot.
The following snippet shows how the file system appears inside the Delphix created mount point.

```
hdbadm@target:/delphix/zfs> tree
.
├── data
└── log

2 directories, 0 files
hdbadm@target:/delphix/zfs> pwd
/delphix/zfs
```

If a system user attempts to create a VDB from this snapshot, it will fail with the below error message.



Subsequent Snapshots

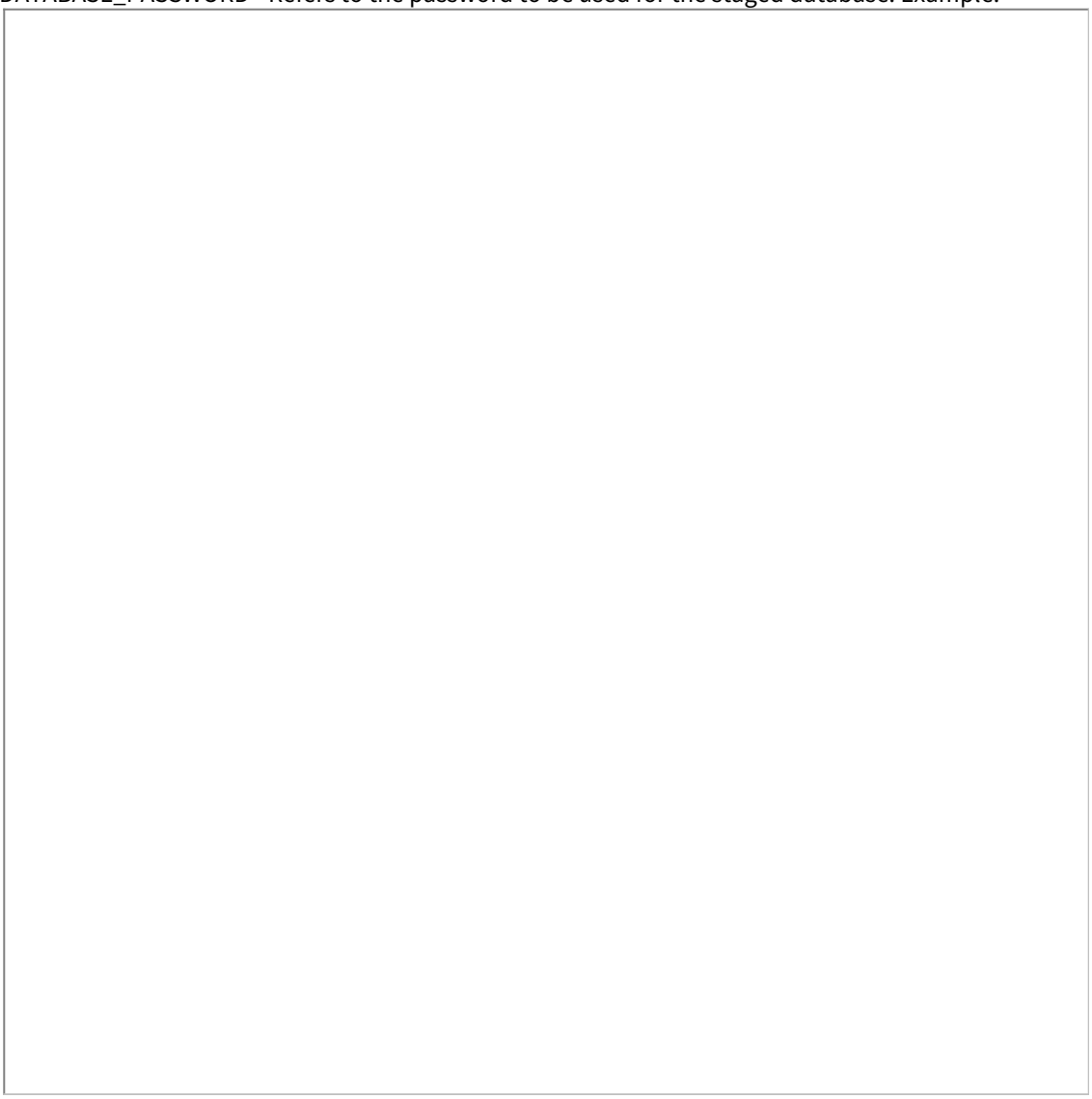
The system user needs to follow the steps and commands below before taking any subsequent snapshots.

Note: The system user needs to provide the values for all the parameters provided within { }.

1. Create a database in "no start" mode. Here, "no start" implies that the database is not started.
   create database {DATABASE_NAME} SYSTEM USER {DATABASE_PASSWORD} no start.

The parameters used in the below example are:

a. DATABASE_NAME - Refers to the database name to be used for staging Push. This is the same name that was provided by the system user on the UI while creating the dSource.
b. DATABASE_PASSWORD - Refers to the password to be used for the staged database. Example:



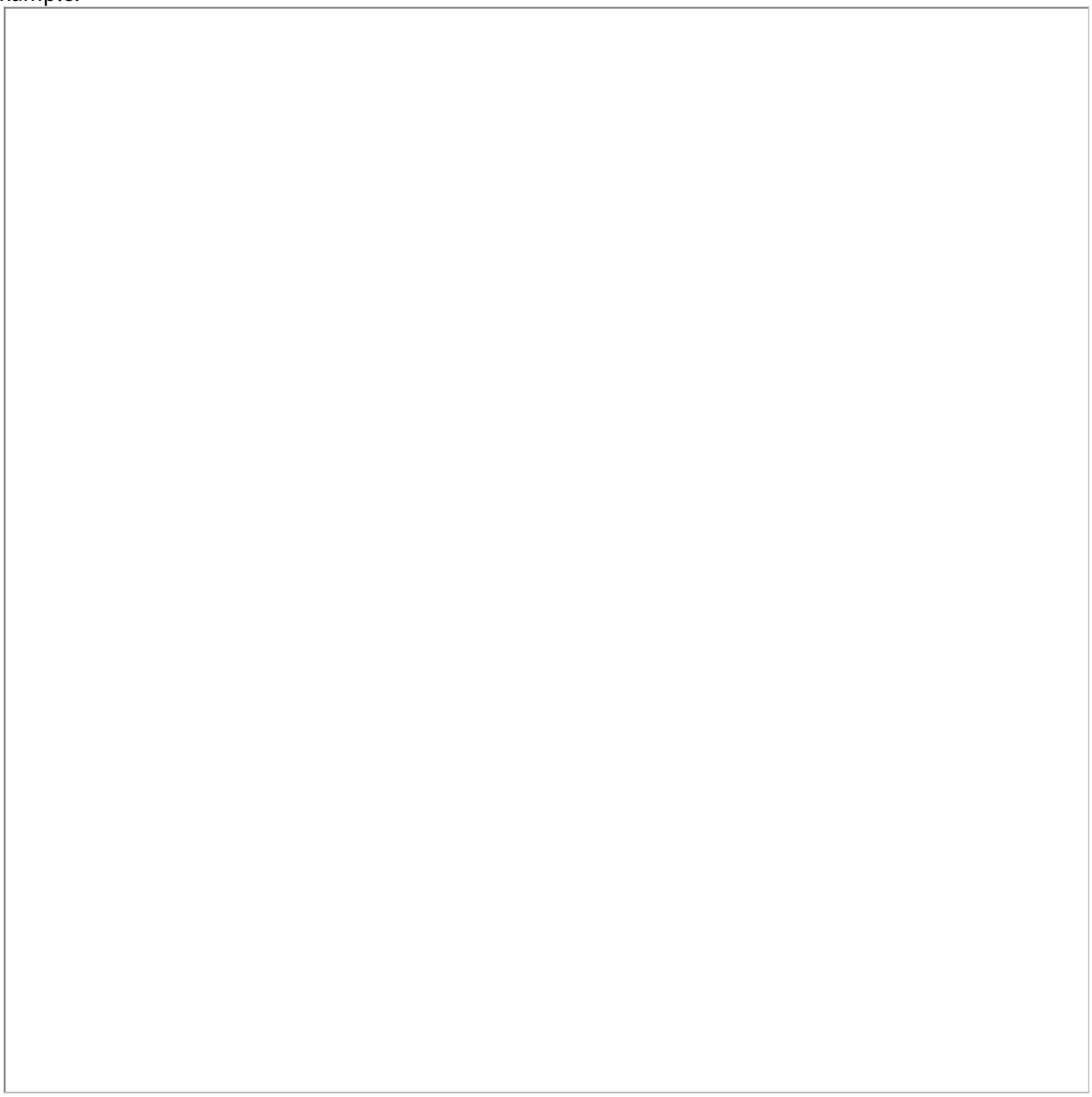
2. Set the HANA database to use the Delphix created mount point to store the data and log volume files. This is done by setting the `basepath_datavolumes` and `basepath_logvolumes` to *data* and *log*directory under the mount point, respectively. All the tenant data volumes and log volumes reside in the single mount point provided by Delphix.

```
//This query sets the value of basepath_datavolumes to within the delphix
storage.
alter system alter configuration('global.ini', 'DATABASE','{DATABASE_NAME}')
SET ('persistence','basepath_datavolumes') = '{DELPHIX_MOUNT_LOCATION}/data'
```

```
//This query sets the value of basepath_logvolumes to within the delphix
storage.
alter system alter configuration('global.ini', 'DATABASE','{DATABASE_NAME}')
SET (''persistence','basepath_logvolumes') = '{DELPHIX_MOUNT_LOCATION}/log'
```

Example:

Once the `basepath_datavolumes` and `basepath_logvolumes` parameters are set for the tenant, then their values are reflected in the `global.ini` file in the tenant database. The location of the global.ini file is `<sapmnt>/<SID>/SYS/global/hdb/custom/config/DB_<tenantdbname>.` E xample:



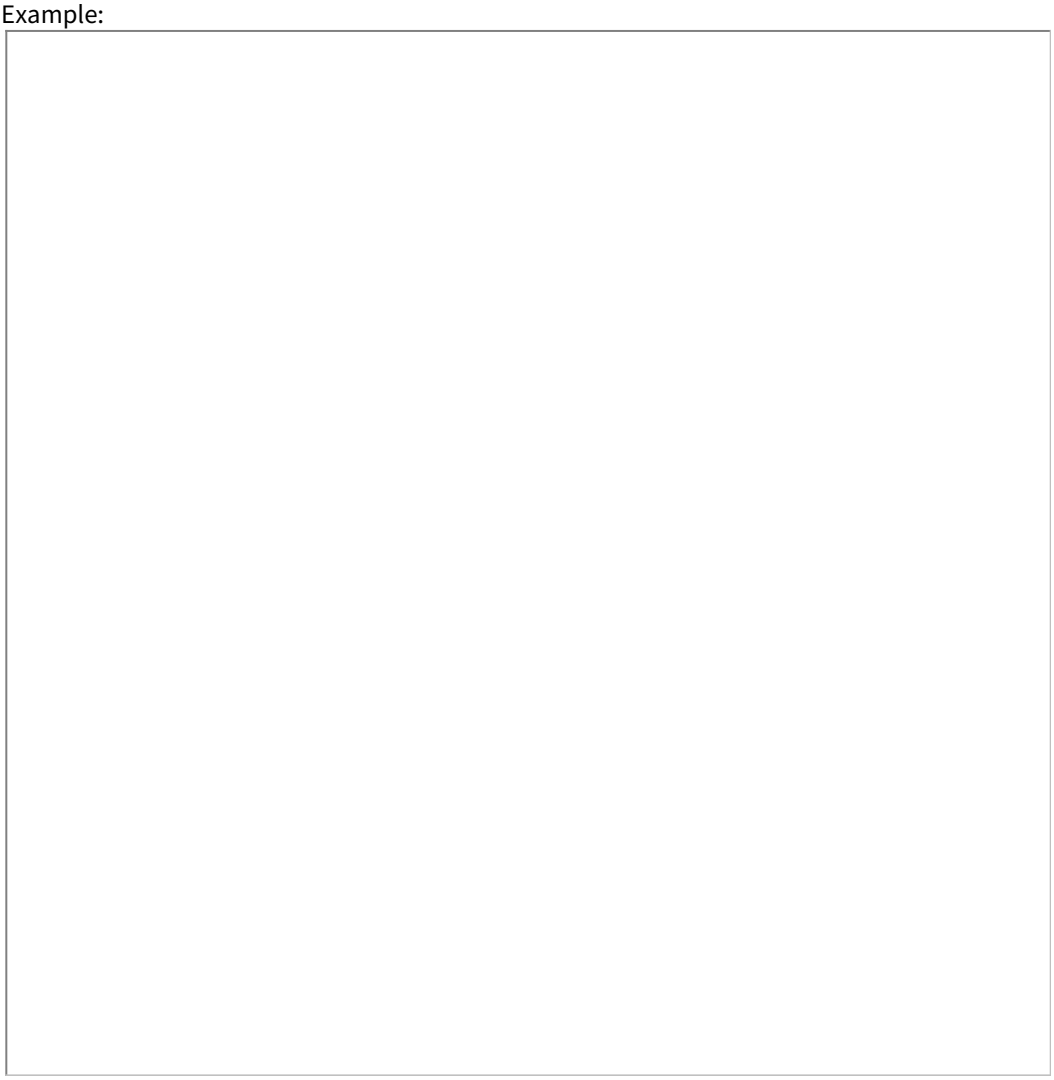
3. Create a backup catalog. This is used by HANA to determine whether recovery is possible and which backups are to be used to recover the database.

Note: Creating a backup catalog is not applicable for third-party backups. For example, Commvault-based backups.

The parameters used in the below example are:

a. logDirs - Refers to the location where logs are present for the database. If the system user does not intend to use logs then an empty directory can be created and the location of the empty directory can be provided here.
b. dataDir - Refers to the location where backups are present.
c. -d - Refers to the location where the catalog backup will be created.
Example:

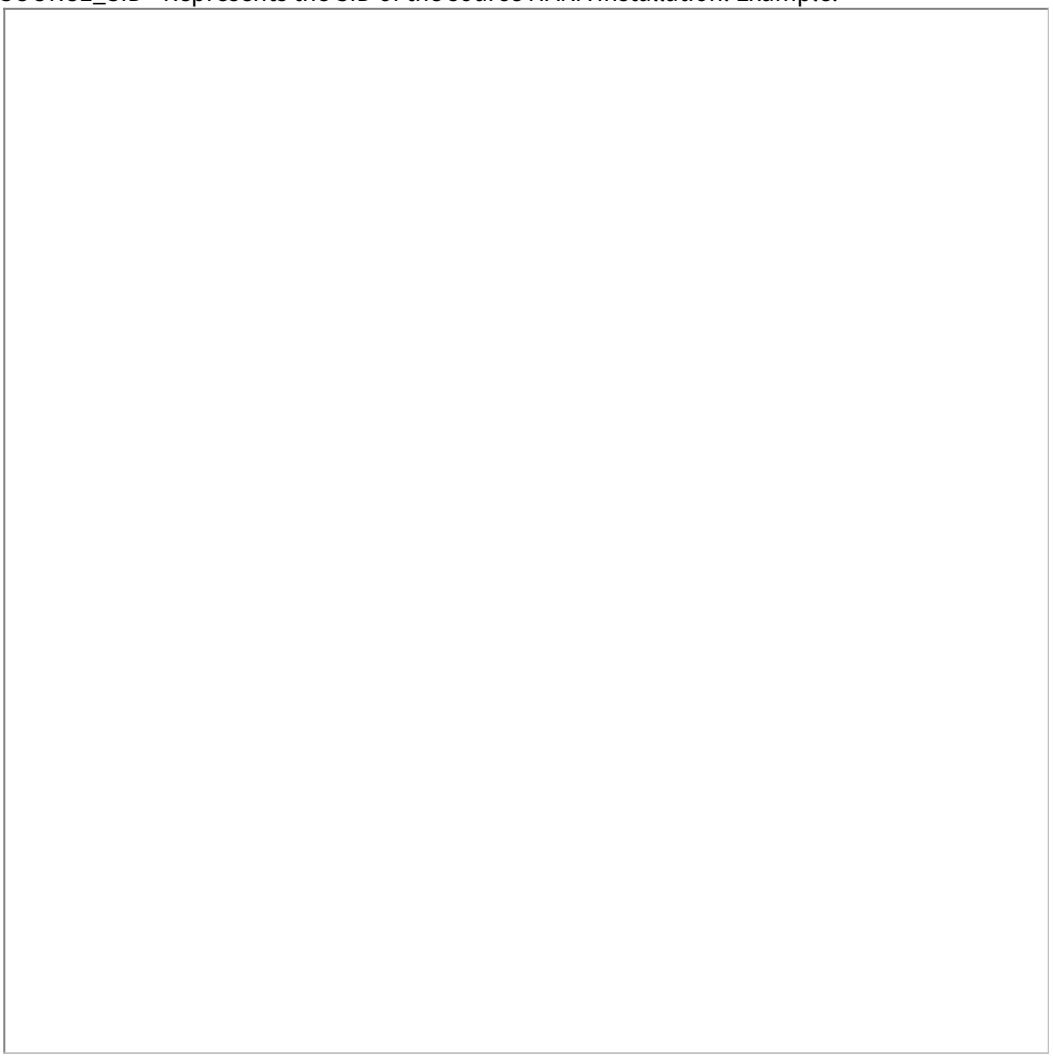
4. Perform recovery on the tenant database. This can be done in the following two ways:
    a. Recovery using data backups only The following parameters for the recovery command are listed below:
        i. Recover database for {DATABASE_NAME} - Represents the name of the database for which recovery is to be performed.
        ii. Until timestamp '{TIMESTAMP}' - Represents the timestamp value till which recovery is to be performed.
        iii. Clear log using catalog path ('{CATALOG_PATH}') - Refers to the path where the catalog backups are kept.
        iv. Using data path ('{DATA PATH}') - Refers to the path where the data backups are kept.
        Example:



Recovery using both data and log backup Along with the above parameters, you need to specify the below parameter:

v. Using log path ('{LOCATION_OF_THE_LOG PATH}') - Refers to the location of log backups.

b. Recovery using third-party data backups. For example, Commvault-based backups. The following parameters for the recovery command are listed below:
   i. STAGING_DATABASE_NAME - Represents the name of the tenant database to be created on the staging area.
   ii. RECOVERY_TIMESTAMP - Represents the time limit until which recovery has to be performed. The time has to be in the YYYY-MM-DD HH:MM:SS format.
   iii. SOURCE_TENANT_DATABASE - Represents the name of the source tenant database.
   iv. SOURCE_SID - Represents the SID of the source HANA installation. Example:



Once the recovery is done, data will be available inside the Delphix mount point and the database will be in an active state. The below snippet shows an example file system after the recovery.

You can now proceed to take a snapshot and the VDB creation using the new snapshot.

Advanced data management settings for SAP HANA dSources

This topic describes advanced data management settings for dSources.

When linking a dSource, you can use custom data management settings to improve overall performance and match the needs of your specific server and data environment. If no specific settings are required, leverage default data management settings.

Accessing data management settings

There are three ways to set or modify data management settings for dSources:

1. During the dSource linking process, in the Policies tab of the Add dSource wizard select the **Retention Policy**.
2. Or select **Manage**, then select **Policies**.
3. In the Policies screen select the **Retention** tab. To manage retention on target after the dSource snapshots have been deleted from the replication source, select the **Replica Retention** tab.
4. To add a new retention policy click the **+Retention** button, and to add a new replica retention policy on the replication target, click the **+Replica Retention** button.
5. This will open the Retention or the Replica Retention Policy wizard, depending upon the above selection. Enter a policy name along with required retention details and click **Submit**. For more information, see Policies for Scheduled Jobs

Retention policies - retention/replica retention

Retention policies define the length of time Delphix Engine retains snapshots within its storage. To support longer retention times, you may need to allocate more storage to the Delphix Engine. The retention policy – in combination with the SnapSync policy – can have a significant impact on the performance and storage consumption of the Delphix Engine.

Replica Retention policies define how long the snapshots are retained on replicated namespaces for dSources and VDBs after they have been deleted on the replication source. Normally, the snapshots that have been deleted on the replication source engine are also deleted on the replication target engine. A new retention policy is introduced to provide an extended lifetime of such snapshots on the replication target.

Benefits of longer retention

Increased retention times for snapshots extends the quantity of historical data the Delphix Engine will retain.

Common use cases for longer retention include:

- SOX compliance
- Frequent application changes and development
- Caution and controlled progression of data
- Reduction of project risk
- Restore to older snapshots

Working with SAP HANA dSource snapshots

Taking a snapshot creates a new snapshot entry in the dSource's Timeflow.

Snapshot (default)

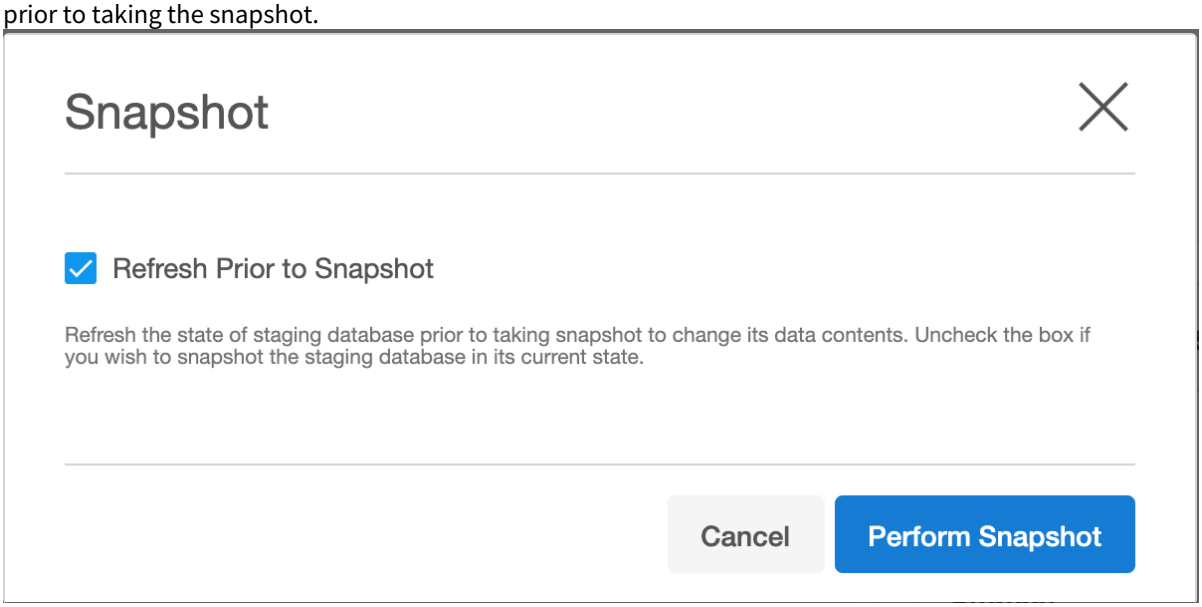
This section lists the steps to take a snapshot and delete the same.

1. Login to the **Delphix Management** application.
2. Click **Manage** and select **Datasets** from the dropdown list.
3. Select the dSource you want to Snapshot.
4. Click the **Camera** icon. Alternatively, click the arrow next to the Camera icon and select **Snapshot (default)**.
5. From the Snapshot dialog, select **Perform Snapshot**.
6. Click **View:** under **Timeflow** to verify the Snapshot you just created.
7. To delete, select the snapshot you just created, and from the Actions menu (...) select **Delete Snapshot**.
8. In the Delete Snapshot dialog, select **Delete**.
9. Click **View:** under **Timeflow** to verify the Snapshot, you just deleted.

Snapshot with parameters

This section lists the steps to take a snapshot with parameters and delete the same.

1. Login to the **Delphix Management** application.
2. Click **Manage** and select **Datasets** from the dropdown list.
3. Select the dSource you want to Snapshot.
4. Click the arrow next to the Camera icon and select **Snapshot with Params...**.
5. Select **Refresh Prior to Snapshot** to refresh the staging database. This will refresh the staging database prior to taking the snapshot.



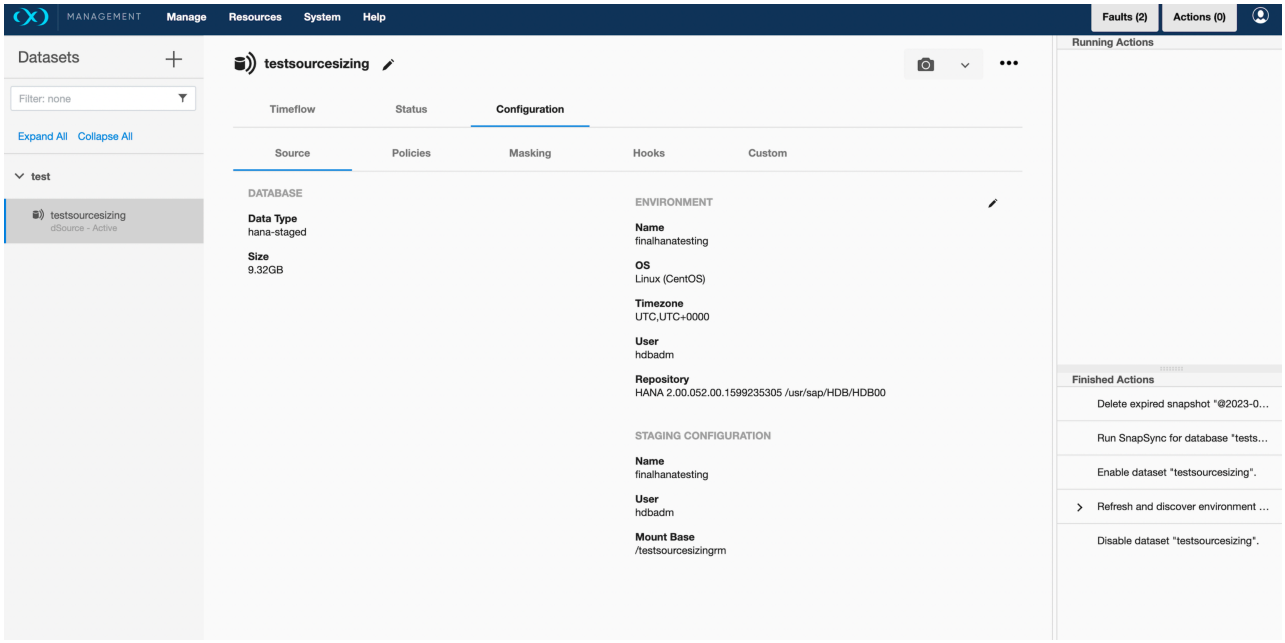
6. In the Snapshot dialog, select **Perform Snapshot**.
7. Click **View:** under **Timeflow** to verify the Snapshot you just created.
8. To delete, select the snapshot you just created, and from the Actions menu (...) select **Delete Snapshot**.
9. Click **View:** under **Timeflow** to verify the Snapshot, you just deleted.

Source sizing implementation for SAP HANA data sources

The source sizing feature aligns the data source experience with Delphix database standards that will improve your ingestion reporting experience on the per TB pricing model. This feature enables you to calculate the dataset size based on the total allocated space on the mount point provided by the Delphix Engine for the dataset.

Procedure

- For all datasets, the size of the dataset can be calculated using `du -sb <dataset>` command that provides the correct volume and apparent size of the datasets.
- Source sizing is implemented for below staging pull methods:
  - For the dSource created using external backup ingestion method
  - For the dSource created using staging push method



Provisioning and managing VDBs from SAP HANA dSources

This section covers the following topics:

Provisioning SAP HANA VDBs: an overview

This topic describes the basic concepts involved with provisioning of VDBs in HANA.

Provisioning of a VDB results in creation of a new tenant database in the specified HANA target environment, with storage for the new tenant backed by Delphix storage. The data inside the tenant reflects the same data in that exists in the dSource snapshot from which the VDB is being provisioned. During the provision process, Delphix will create a filesystem in Delphix specifically for the new VDB, populate the filesystem with the appropriate data files and datafile contents, then share and mount that filesystem onto the specified HANA target host. The new HANA tenant will have its storage paths re-configured to use a single subdirectory specific to the tenant, then traverse an NFS mount which points to the filesystem on Delphix for that tenant. The VDB provision operation is very fast because Delphix does not move nor copy data to create and populate the VDB filesystem.

During the VDB creation, the location of the basepath_datavolumes and basepath_logvolumes will be changed to use the storage inside the Delphix Engine. This will mean that any new services (even if a user creates a service post VDB creation by the plugin) which will now be created for the virtualized tenant will also be created on the tenant-specific filesystem on the Delphix Engine.

Setting log mode overwrite

As a part of the VDB creation process, if the correct tenant username and password are provided, then the log mode for the virtualized tenant will be set to 'overwrite'. It means that "No log backups" are created. When savepoints are written, log segments are immediately freed to be overwritten by new log entries.

HANA port control

Starting Delphix Engine 6.0.10.0 and HANA vSDK plugin, Delphix introduces a solution that keeps the port numbers consistent throughout the VDB life cycle so that the connections made to the VDBs are not disrupted during their life cycle.

Need for user specified ports?

Until now, Delphix HANA was dependent on the HANA-provided ports for any VDB. Every VDB has its own ports and connections for internal and external communication. When operations such as rewind, refresh, or enable are performed on VDBs, this can cause port instability or inconsistency problems because the port numbers could change even when the VDB name remains the same. The below points further elaborates on this limitation.

- The plugin uses HANA-provided port numbers which are selected from the available pool of port numbers in the bottom-up approach that means an idle port number at the bottom will be picked first for use.
- The plugin does not have any mechanism to remember these port numbers and use them from the pool at the time of creating any tenant database again at the time of enabling, refresh, or rewind.
  So if we had a tenant TEST having a service indexserver at 30043 port number, then after the refresh, rewind or enable, there will be a chance that
  a. The same port number(30043) is not available
  b. Any port number lesser than the one being used earlier(30043) is now available.

So, there are chances that the tenant TEST is provided with another port number for the same service by HANA and if there was any connection made to the tenant TEST at 30043 will now be disrupted. Hence there is a need for a solution that allows the database to keep the port number consistent throughout the VDB life cycle.

How does the HANA port allocation works?

The port control feature now provides an option to specify port numbers at the time of VDB creation. This can be done in the following two ways.

User-specified HANA ports

See the procedure for Provisioning HANA VDBs where the user can specify the HANA service and port number at the time of VDB creation.

By Plugin itself

If the service and port numbers are not provided by the user on the Plugin user interface, then the plugin itself will decide the port number and will keep the ports consistent throughout the life of a VDB.

This is how the plugin chooses the ports:

- It is based on the value of the `reserved_instance_number` parameter in the `global.ini` file. This file is located at `/hana/shared/{DATABASE_SID}/global/hdb/custom/config.` If this parameter is not defined in global.ini, then the value of the instance number will be used.
- Ports will be decided by the HANA plugin in a top-down manner which means that the topmost port which is free will be used first and so on.

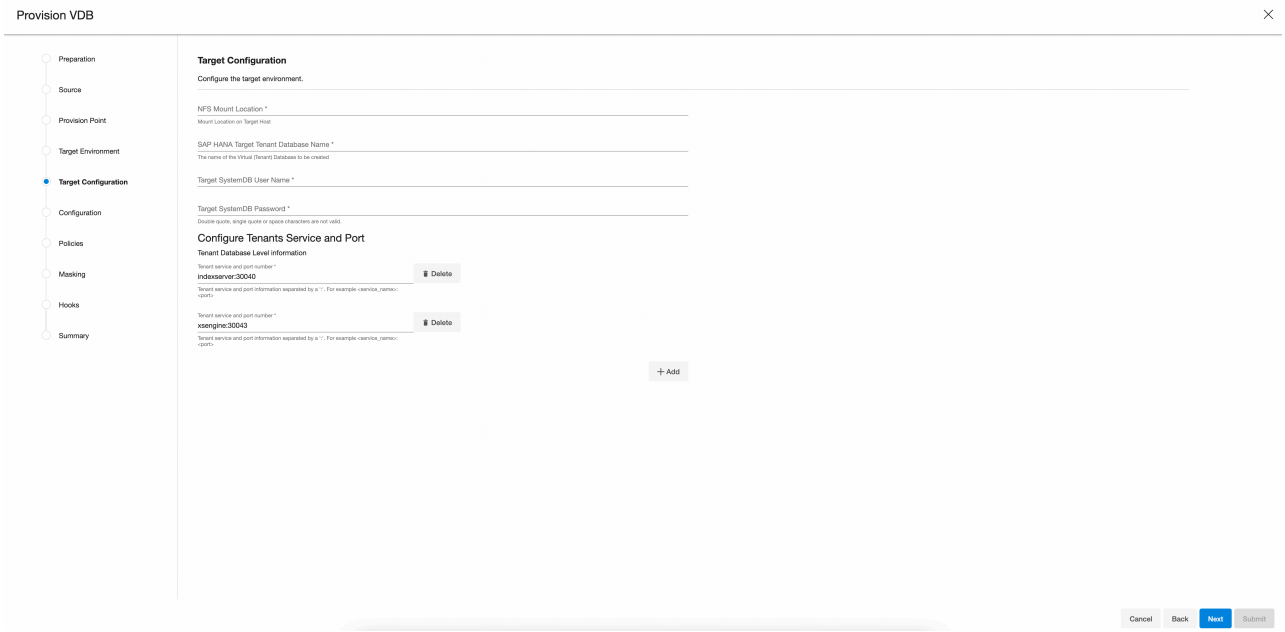
Provisioning SAP HANA VDBs

This topic describes how to provision a virtual database (VDB) from a HANA dSource.

Prerequisites

- The provision requires a linked dSource built off a physical source tenant, as described in Linking a HANA dSource, or a VDB.
- The HANA target environment should adhere to the Requirements for HANA Hosts and Databases
- Operating system users and database users should be configured.

VDB creation UI



Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select a **dSource**.
5. Select a **snapshot** from which you want to provision.
   Info:
   If the first or empty dSource Snapshot that was created using the Staging Push mechanism is detected by the plugin and an error message is displayed. This check is done for every VDB creation operation.
6. Click the **Provision VDB** icon to open the Provision VDB wizard.
7. Select a target environment from the left pane.
8. The HANA 2.0 Plugin supports one installation on the same host, hence the value in the **Installation** dropdown will be already populated. Multiple drop-down options should not be available.
9. Set the **Environment User** to be the Instance Owner. Note: The picking of instance owner is only possible if you have multiple environment users set on that host.
10. Click **Next**.
11. In the **Target Configuration**pane you will need to specify the following:
    Info: **'*'** indicates the mandatory fields.

a. NFS Mount Location *: Delphix recommends setting the trailing directory to match the VDB name to delineate mount points and support administration and troubleshooting. In the example shown, the **NFS Mount Location** is /mnt/provision/dev where dev matches the **SAP HANA Target Tenant Database Name**.
b. SAP HANA Target Tenant Database Name *: This refers to the target tenant database name which will be created upon VDB creation.
c. **Target SystemDB User Name ***: This refers to the SystemDB username in the target database.
d. **Target SystemDB Password ***: This refers to the SystemDB password in the target database.
Note:
The SYSTEM username and password derived from the physical source database should be entered during the provision of a VDB because those credentials will be recovered during provisioning.
e. Configure Tenants Service and Port: This refers to the list of tenant services and port numbers in the following format.

```
<service>:<port>For example, indexserver:30043
```

You can click "**+Add**" to add more services.
Info :
- This is not a mandatory field. If a user doesn't provide a value to this, then the plugin will proceed with the above fields and it will allocate the service and port on its own.
- If an invalid service is provided, the plugin will proceed with input, but will eventually fail while creating the service and will display an error on the Delphix Engine UI.

12. In the configuration section
    a. Set the vFile Name or accept the default, which is a random string. Delphix strongly recommends applying a consistent naming convention. We suggest this name should match the Target Tenant Name from the previous pane. Maintaining a consistent naming convention will support administration and troubleshooting activities. In the example shown, the vFile Name “dev” matches the actual tenant name and mount point from the previous pane.
    b. Select a **Target Group** for the VDB.Click the **Add Dataset Group** icon to add a new group if required.
    c. Set **Auto VDB Restart** option to automatically restart the VDB, where the target host is rebooted. Delphix recommends enabling this setting.
13. Click **Next**.
14. VDB **Snapshot Policy**. Select the Snapshot policy if an automated VDB snapshot schedule is required. Optionally, the Snapshot policy may be disabled by selecting “None”.
15. Click **Next**.
16. **Masking**. This page offers the opportunity to link the provision job to a masking job. At this time, Delphix Masking does not support HANA. Please do not use this option. “Mask this VDB” should remain deselected.
17. **Hooks**. Run a customized shell script via a hook operation at a particular moment in the provision operation.
18. Click **Next**.
19. **Summary.** Presents an opportunity to review all the information as a part of the initial provision workflow.
20. Click **Submit**.Once a provision job is submitted it is possible to monitor its progress by selecting “Active Jobs” in the top menu bar. On successful completion, VDB will be visible in the list of Datasets under its assigned group.

Working with SAP HANA VDB snapshots

Taking a snapshot creates a new snapshot entry in the VDB's Timeflow.

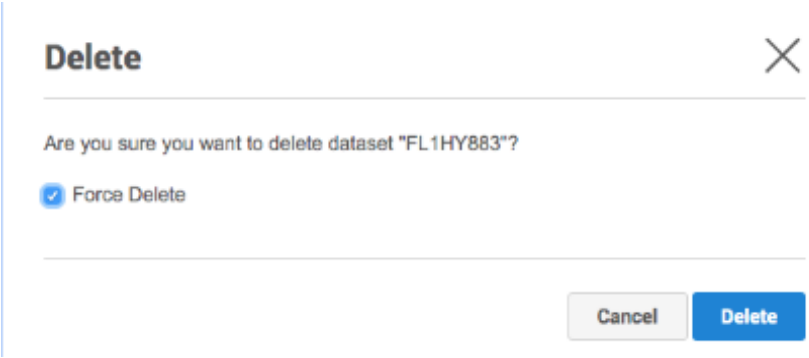
This section lists the steps to take a snapshot and delete the same.

1. Login to the **Delphix Management** application.
2. Click **Manage** and select **Datasets** from the dropdown list.
3. Select the VDB you want to perform Snapshot.
4. Click the **Camera** icon.
5. From the Snapshot dialog, select **Yes**.
6. Click **View:** under **Timeflow** to verify the Snapshot you just created.
7. To delete, select the snapshot you just created, and from the Actions menu (...) select **Delete Snapshot**.
8. In the Delete Snapshot dialog, select **Delete**.
9. Click **View:** under **Timeflow** to verify the Snapshot, you just deleted.

Failure during provision

Depending on where in the provision process failure occurs, and the type of failure, extra steps may be required:

- **Force Delete** of the provision object using the User Interface - The **Force Delete** option will remove underlying filesystems shared from the Delphix Engine, irrespective of any perceived dependencies on these filesystems. If provisioning fails very early in the process, it is common for the **Force Delete** option to be the only way to remove an object. This may happen when for example, the wrong password has been entered for the target-side system database user:



- In rare cases, removal of a failed virtual tenant may be required. For example a tenant with the name “TST”.

```
SELECT * FROM M_DATABASES WHERE DATABASE_NAME='TST';
ALTER SYSTEM STOP DATABASE TST IMMEDIATE;
DROP DATABASE TST DROP BACKUPS;
```

- If the **Force Delete** option could not unmount the virtual tenant mount points, then the unmount command should be run to remove the stale mount points.

```
umount -lf /mnt/provision/example
```

SAP HANA hook operations

Shell operations

RunCommand operation

The RunCommand operation runs a shell command on a Unix environment using whatever binary is available at /bin/sh. The environment user runs this shell command from their home directory. The Delphix Engine captures and logs all output from this command. If the script fails, the output is displayed in the Delphix Admin application and command-line interface (CLI) to aid in debugging.

If successful, the shell command must exit with an exit code of 0. All other exit codes will be treated as an operation failure.

Examples of RunCommand operations

You can input the full command contents into the RunCommand operation.

```
remove_dir="$DIRECTORY_TO_REMOVE_ENVIRONMENT_VARIABLE"
if test -d "$remove_dir"; then
    rm -rf "$remove_dir" || exit 1
fi
exit 0
```

If a script already exists on the remote environment and is executable by the environment user, the RunCommand operation can execute this script directly.

```
/opt/app/oracle/product/10.2.0.5/db_1/dbs/myscript.sh "$ARG_ENVIRONMENT_VARIABLE"
"second argument in double quotes" 'third argument in single quotes'
```

RunBash operation

The RunBash operation runs a Bash command on a Unix environment using a bash binary provided by the Delphix Engine. The environment user runs this Bash command from their home directory. The Delphix Engine captures and logs all output from this command. If the script fails, the output is displayed in the Delphix Admin application and command-line interface (CLI) to aid in debugging.

If successful, the Bash command must exit with an exit code of 0. All other exit codes will be treated as an operation failure.

Example of RunBash operations

You can input the full command contents into the RunBash operation.

```
remove_dir="$DIRECTORY_TO_REMOVE_ENVIRONMENT_VARIABLE"
# Bashisms are safe here!
if [[ -d "$remove_dir" ]]; then
    rm -rf "$remove_dir" || exit 1
fi
exit 0
```

Shell operation tips

Using nohup

You can use the nohup command and process backgrounding from the resource in order to "detach" a process from the Delphix Engine. However, if you use nohup and process backgrounding, you MUST redirect stdout and stderr.

Unless you explicitly tell the shell to redirect stdout and stderr in your command or script, the Delphix Engine will keep its connection to the remote environment open while the process is writing to either stdout or stderr. Redirection ensures that the Delphix Engine will see no more output and thus not block waiting for the process to finish.

For example, imagine having your RunCommand operation background a long-running Python process. Below are the bad and good ways to do this.

Bad examples

- nohup python file.py & # no redirection
- nohup python file.py 2>&1 & # stdout is not redirected
- nohup python file.py 1>/dev/null & # stderr is not redirected
- nohup python file.py 2>/dev/null & # stdout is not redirected

Good examples

- nohup python file.py 1>/dev/null 2>&1 & # both stdout and stderr redirected, Delphix Engine will not block

Other operations

RunExpect operation

The RunExpect operation executes an Expect script on a Unix environment. The Expect utility provides a scripting language that makes it easy to automate interactions with programs that normally can only be used interactively, such as ssh. The Delphix Engine includes a platform-independent implementation of a subset of the full Expect functionality.

The script is run on the remote environment as the environment user from their home directory. The Delphix Engine captures and logs all output of the script. If the operation fails, the output is displayed in the Delphix Admin application and CLI to aid in debugging.

If successful, the script must exit with an exit code of 0. All other exit codes will be treated as an operation failure.

Example of a RunExpect operation

Start an ssh session while interactively providing the user's password.

```
spawn ssh user@delphix.com
expect {
    -re {Password: } {

        send "${env(PASSWORD_ENVIRONMENT_VARIABLE)}\n"
    }
    timeout {
```

```
        puts "Timed out waiting for password prompt."

        exit 1
    }
}
exit 0
```

HANA environment variables

Operations that run user-provided scripts have access to environment variables. For operations associated with specific dSources or virtual databases (VDBs), the Delphix Engine will always set certain environment variables so that the user-provided script can use them to access the dSource or VDB.

dSource environment variables

Environment Variable	Description
`DLPX_DATA_DIRECTORY`	The primary mount path provided by the user on the UI. The first one, in case multiple mount specifications are provided.
`USER`	The OS user used to link the dSource

VDB environment variables

Environment Variable	Description
`DLPX_DATA_DIRECTORY`	The primary mount path provided by the user on the UI. The first one, in case multiple mount specifications are provided.
`USER`	The OS user used to provision the VDB

Shell operations

RunCommand operation

The RunCommand operation runs a shell command on a Unix environment using whatever binary is available at /bin/sh. The environment user runs this shell command from their home directory. The Delphix Engine captures and logs all output from this command. If the script fails, the output is displayed in the Delphix Admin application and command-line interface (CLI) to aid in debugging.

If successful, the shell command must exit with an exit code of 0. All other exit codes will be treated as an operation failure.

Examples of RunCommand operations

You can input the full command contents into the RunCommand operation.

```
remove_dir="$DIRECTORY_TO_REMOVE_ENVIRONMENT_VARIABLE"
if test -d "$remove_dir"; then
    rm -rf "$remove_dir" || exit 1
fi
exit 0
```

If a script already exists on the remote environment and is executable by the environment user, the RunCommand operation can execute this script directly.

/opt/app/oracle/product/10.2.0.5/db_1/dbs/myscript.sh "$ARG_ENVIRONMENT_VARIABLE" "second argument in double quotes" 'third argument in single quotes'

RunBash operation

The RunBash operation runs a Bash command on a Unix environment using a bash binary provided by the Delphix Engine. The environment user runs this Bash command from their home directory. The Delphix Engine captures and logs all output from this command. If the script fails, the output is displayed in the Delphix Admin application and command-line interface (CLI) to aid in debugging.

If successful, the Bash command must exit with an exit code of 0. All other exit codes will be treated as an operation failure.

Example of RunBash operations

You can input the full command contents into the RunBash operation.

```
remove_dir="$DIRECTORY_TO_REMOVE_ENVIRONMENT_VARIABLE"
# Bashisms are safe here!
if [[ -d "$remove_dir" ]]; then
    rm -rf "$remove_dir" || exit 1
fi
exit 0
```

Shell operation tips

Using nohup

You can use the nohup command and process backgrounding from the resource in order to "detach" a process from the Delphix Engine. However, if you use nohup and process backgrounding, you MUST redirect stdout and stderr.

Unless you explicitly tell the shell to redirect stdout and stderr in your command or script, the Delphix Engine will keep its connection to the remote environment open while the process is writing to either stdout or stderr. Redirection ensures that the Delphix Engine will see no more output and thus not block waiting for the process to finish.

For example, imagine having your RunCommand operation background a long-running Python process. Below are the bad and good ways to do this.

Bad examples

- nohup python file.py & # no redirection
- nohup python file.py 2>&1 & # stdout is not redirected
- nohup python file.py 1>/dev/null & # stderr is not redirected
- nohup python file.py 2>/dev/null & # stdout is not redirected

Good examples

- nohup python file.py 1>/dev/null 2>&1 & # both stdout and stderr redirected, Delphix Engine will not block

Other operations

RunExpect operation

The RunExpect operation executes an Expect script on a Unix environment. The Expect utility provides a scripting language that makes it easy to automate interactions with programs that normally can only be used interactively, such as ssh. The Delphix Engine includes a platform-independent implementation of a subset of the full Expect functionality.

The script is run on the remote environment as the environment user from their home directory. The Delphix Engine captures and logs all output of the script. If the operation fails, the output is displayed in the Delphix Admin application and CLI to aid in debugging.

If successful, the script must exit with an exit code of 0. All other exit codes will be treated as an operation failure.

Example of a RunExpect operation

Start an ssh session while interactively providing the user's password.

```
spawn ssh user@delphix.com
expect {
    -re {Password: } {

        send "${env(PASSWORD_ENVIRONMENT_VARIABLE)}\n"
    }
    timeout {

        puts "Timed out waiting for password prompt."

        exit 1
    }
}
exit 0
```

HANA environment variables

Operations that run user-provided scripts have access to environment variables. For operations associated with specific dSources or virtual databases (VDBs), the Delphix Engine will always set certain environment variables so that the user-provided script can use them to access the dSource or VDB.

dSource environment variables

Environment Variable	Description
`DLPX_DATA_DIRECTORY`	The primary mount path provided by the user on the UI. The first one, in case multiple mount specifications are provided.
`USER`	The OS user used to link the dSource

VDB environment variables

Environment Variable	Description
`DLPX_DATA_DIRECTORY`	The primary mount path provided by the user on the UI. The first one, in case multiple mount specifications are provided.
`USER`	The OS user used to provision the VDB

SQL Server environments and data sources

This section covers the following topics:

Delphix architecture with SQL Server

SQL Server is Microsoft's relational database which is typically run on Windows Server hosts. With Delphix, you can use various SQL Server configurations, ranging from Failover Clusters to Availability Groups. In this section, you'll find an overview of how Delphix works with SQL Server.

There are three key concepts when using Delphix with any data platform:

1. Environments: The server and software required to run a data set.
    a. Source Environment: Source data to be ingested into Delphix. These will be used to create dSources.
    b. Target Environment: Target hosts to provision VDBs.
2. dSources: A database that the Delphix Virtualization Engine uses to create and update or maintain virtual copies of your database
3. VDBs: A database provisioned from either a dSource or another VDB which is a copy of the source data. A VDB is created and managed by the Delphix Virtualization Engine.

With these concepts in mind, let's explore how Delphix connects to SQL Server environments and creates SQL Server dSources and VDBs.

Staging push mechanism with SQL server

Starting Delphix Engine version 6.0.13.0 a new data ingestion mechanism has been introduced that will help users to push data into the staging database on their own.

The previous data ingestion mechanism of the SQL Server has a few limitations like dependency on the source access and limited backup vendor's support. Currently, Delphix supports Native, Commvault, NetBackup, Lightspeed, and Redgate backup vendors.

With Staging Push implementation, we have removed the dependency of accessing the customer's production database and also enabled the customers who are using the backup vendors that Delphix does not support.

To summarize the Staging Push mechanism.

- Customers now have ownership of the staging databases.
- Customers are now responsible to keep the staging database in sync with the source database. Note that the database files of the staging database are stored on Delphix Storage.
- Delphix is still responsible to snapshot the underlying data files and gathering any metadata required for provisioning from the snapshots.

Delphix in multi-domain windows environments

General Overview

When considering the Delphix logical architecture, there are four primary components:

1. Source host(s)
2. Continuous Data Engine
3. VDB Target host(s)
4. Staging Target Host(s)

In SQL Server environments, the staging target host is used for staging data from the source database on the source host into Delphix. Although you can use any VDB target host on which the Delphix Connector service has been installed for this purpose, Delphix recommends a dedicated Staging Target Host for load isolation and separation of roles.

This page focuses on the process of getting source SQL Server data into the storage of the Continuous Data Engine (DVE) via the Staging Target Host.

When considering SQL Server deployments in different enterprise environments, we often see cases where the production, development, test, or reporting environments exist in different Windows domains which may or may not have trust relationships. Such varying domain approaches can come into play due to security, organizational, geographical, or other technical reasons, and can make communication between Windows hosts more complicated to manage. Delphix is flexible enough to work in many configurations, but we want to help you choose the solution that best suits your unique environment.

We listed the four primary components of the Delphix logical architecture for SQL Server above. In addition, a fifth component in the Delphix logical architecture might be considered for use-cases #3, #4, and #5 in the Technical Overview below: a Connector host. The function of the Delphix Connector on that host is the discovery of the source environment via remote registry and ODBC calls. There are no Delphix software installation requirements for Windows source hosts, but it might be helpful to note this role can co-reside directly on the Windows source host for consolidation purposes if desired.

Technical overview

Keep in mind that the Delphix Engine is always syncing with backups of the source database. It is never the live data that is ingested; it is always backups of different flavors.

If SQL Server simple recovery mode is used, these can be full or differential backups initiated by the source database. If full recovery mode is enabled, the Delphix Engine will typically leverage only transaction log backups after the initial data load. Again, the source database would initiate backups, and the Delphix Engine would collect the backup files that have been created by SQL Server. This approach of using transaction logs minimizes spikes in system load by ingesting smaller backups more often. Another option is copy-only backups, which the Delphix Engine initiates in a configuration known as Delphix Managed Backups. For more information refer to Delphix as a Backup Solution to SQL Server.

Delphix can ingest database and log data from native backups, as well as a number of third-party backup products. SQL Server restores the backups onto the shared Delphix storage on the staging target host running the databases in recovery mode. We call this process “validated sync,” which is why you may hear the staging target also referred to as a validated sync server.

It is important to note that the Continuous Data Engine (based on DxOS, itself derived from a UNIX-based OS) is not a domain member itself. The credentials we discuss in this document are between Windows servers, and the key domain-specific authentication is between the staging host and the UNC path to the SMB share where the backup data is stored.

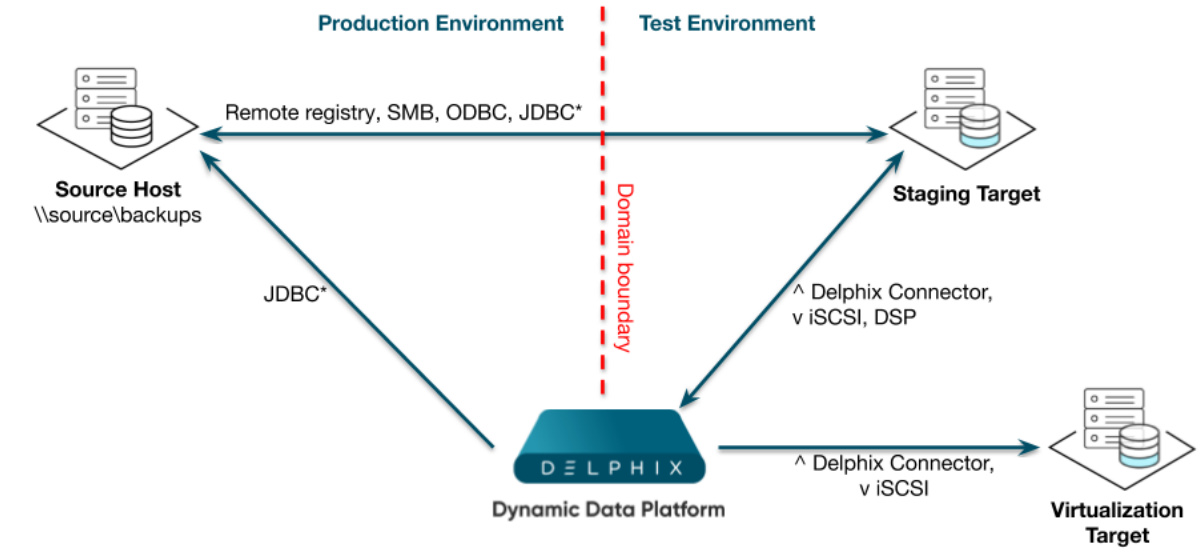
In the rest of this section, we will describe multiple scenarios. Review them to determine which will fit best in your environment.

Case 1: Staging target in test environment

In this case, we will review an environment with two domains: PRODUCTION and TEST, which have a domain trust relationship. This is one of the simplest and most straightforward approaches, as illustrated in the *Staging Target in Test Environment diagram below*.

In this example, the staging target host exists in the non-production TEST domain, but because of domain trust, accounts located in that domain can access resources in the PRODUCTION domain. This would allow the staging target host to connect to the PRODUCTION source host both for environment discovery and to the shared backup location "\source\backups" over Server Messaging Block (SMB) to access database and transaction log backups.

Case 1: Staging Target in Test Environment



Protocol & Port Connection information
SMB/445 – Server Messaging Block (file transfer)
iSCSI 3260/tcp – TCP network storage traffic initiated from the target
JDBC 1433/tcp – Get version, backup history, and LSN data
Delphix Connector v1 9100/tcp – Custom control and data specific to the Delphix Engine
DSP 8415/tcp - Delphix Session Protocol connections (Optionally needed for features like Windows Authentication, Netbackup/Commvault Support)
Remote Registry 445/tcp – Discovery of SQL instances and ports
* JDBC Connectivity is needed to the Source Database:
- directly from Delphix Engine if using Sql Authentication
- from Staging/Connector Host if using Windows Authentication

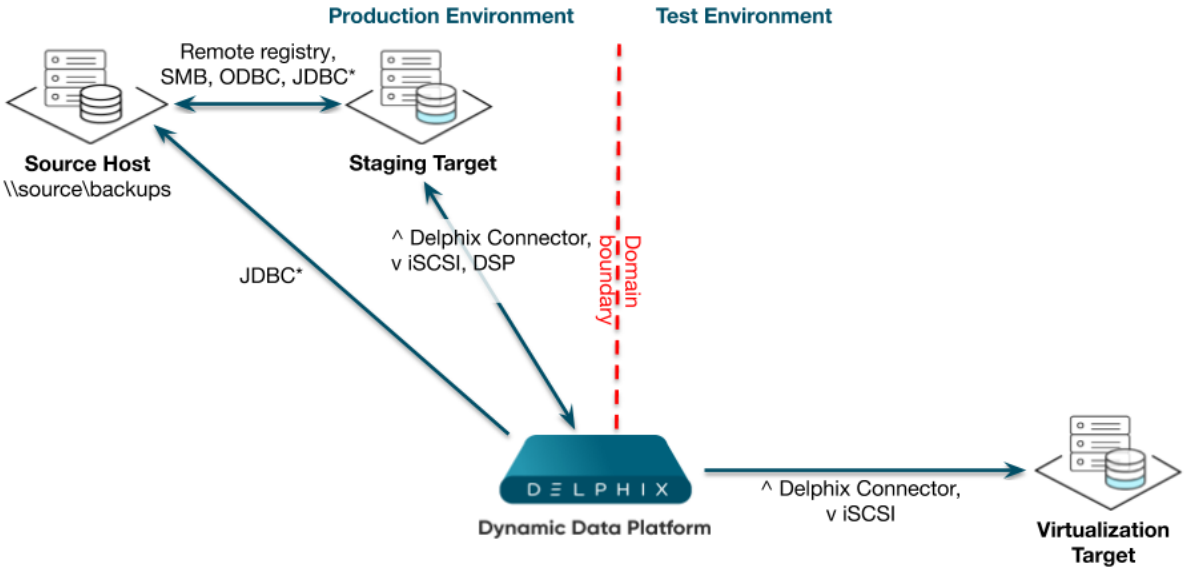
Staging Target in Test Environment

Case 2: Staging target in production environment

The scenario illustrated in the *Staging Target in Production Environment* diagram below shows a TEST domain that does not have access to resources in the PRODUCTION domain. However, the customer has determined that Delphix VDBs must be provisioned to the TEST domain. In this scenario, you can use the staging target host in the PRODUCTION domain to link to the PRODUCTION database and perform the normal restore of the DB and/or log files to the Delphix storage. You can then provision VDBs in the TEST domain.

In this case, VDBs can be completely isolated from the PRODUCTION domain, and there is no requirement for hosts in the TEST domain to have any direct access to resources in the PRODUCTION domain.

Case 2: Staging Target in Production Environment



Protocol & Port Connection information
SMB/445 – Server Messaging Block (file transfer)
iSCSI 3260/tcp – TCP network storage traffic initiated from the target
JDBC 1433/tcp – Get version, backup history, and LSN data
Delphix Connector v1 9100/tcp – Custom control and data specific to the Delphix Engine
DSP 8415/tcp - Delphix Session Protocol connections (Optionally needed for features like Windows Authentication, Netbackup/Commvault Support)
Remote Registry 445/tcp – Discovery of SQL instances and ports
* JDBC Connectivity is needed to the Source Database:
- directly from Delphix Engine if using Sql Authentication
- from Staging/Connector Host if using Windows Authentication

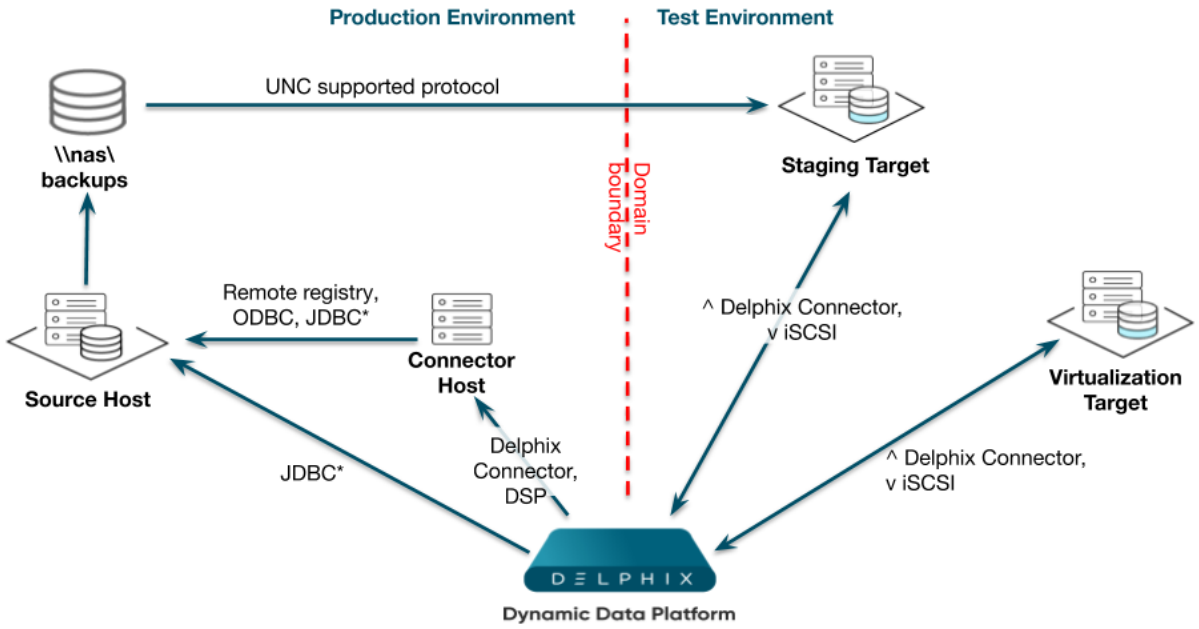
Staging Target in Production Environment

Case 3: Domain-agnostic storage

This example shows a shared backup location that is not dependent on trust relationships between the PRODUCTION and TEST domains. Because Delphix uses UNC paths, it can support any protocol which provides UNC access for that backup data access – for example, SMB or iSCSI.

This is shown by the *Domain-agnostic Storage* diagram below by the arrow – stretching from bottom-left toward the upper-right and crossing the domain boundary – representing any UNC-compatible protocol connecting the staging target host to the data on the NAS host. Provided that the Delphix environment users on both the source host and staging target host have read/write access to the shared backup location on network-attached storage (NAS), the SQL Server instance running on the staging target host will be able to access the backup files needed.

Although this option is not specific to this case, you may notice we separated a connector role to its own connector host. As you can infer from the diagram, the Delphix Connector's primary function on that host is the discovery of the source environment via remote registry and ODBC calls. Despite the fact that there are no software installation requirements for the source hosts in PRODUCTION, it may be helpful to note that you can even install this role directly on the source server for consolidation if you want to.



Protocol & Port Connection information
SMB/445 – Server Messaging Block (file transfer)
iSCSI 3260/tcp – TCP network storage traffic initiated from the target
JDBC 1433/tcp – Get version, backup history, and LSN data
Delphix Connector v1 9100/tcp – Custom control and data specific to the Delphix Engine
DSP 8415/tcp - Delphix Session Protocol connections (Optionally needed for features like Windows Authentication, Netbackup/Commvault Support)
Remote Registry 445/tcp – Discovery of SQL instances and ports
* JDBC Connectivity is needed to the Source Database:
- directly from Delphix Engine if using Sql Authentication
- from Staging/Connector Host if using Windows Authentication

Case 3: Domain-agnostic Storage

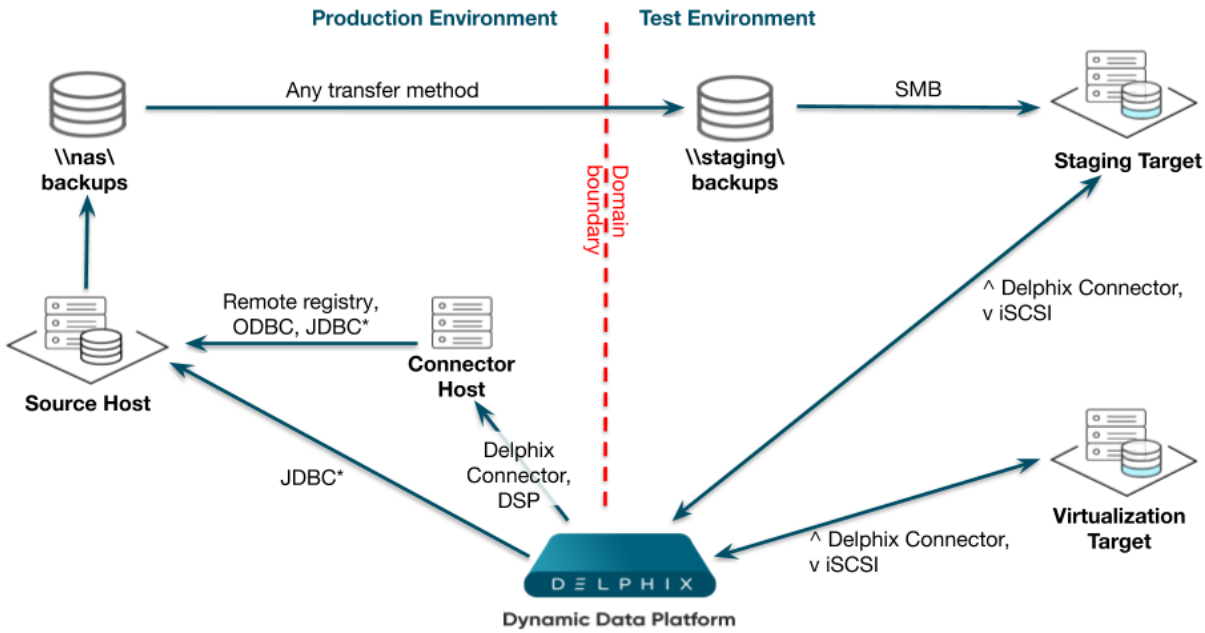
Domain-agnostic Storage

Case 4: Migrating backup files

In this somewhat more complex configuration, backup files are sent to storage in the PRODUCTION domain, while the host used to link to the source and perform the validated sync is in an isolated TEST domain. We have used a separate connector host in the PRODUCTION domain again, to perform the environment discovery of the source host there. Backup files for SOURCE are being stored on NAS.

We will link using the Staging Target Host and create VDBs in the TEST domain. When the Delphix Engine discovers that a new backup of PRODUCTION has been taken, it will attempt to find the relevant files in the shared backup location provided during linking. It does this by periodically performing a recursive search for the file names on the shared backup location. If it does not find the specific files, it will try again later. Knowing this, we can specify a shared backup location in the TEST domain and set up an automated process to copy the backup files from \nas\backups in the PRODUCTION domain to \staging\backups in the TEST domain. We can use any copy mechanism to transfer the files, such as FTP or ROBOCOPY. The files must be available long enough for the Delphix Engine to detect and apply them to the recovery database on the Staging Target Host before removal.

We have customers who also use this model in cases with multiple data centers (on-premise deployments) or virtual private clouds (cloud deployments) rather than multiple domains. These customers want database and transaction log backups to be available in secondary data centers or private clouds, but they want to make sure that the data is only copied over the WAN once.



Protocol & Port Connection information
SMB/445 – Server Messaging Block (file transfer)
iSCSI 3260/tcp – TCP network storage traffic initiated from the target
JDBC 1433/tcp – Get version, backup history, and LSN data
Delphix Connector v1 9100/tcp – Custom control and data specific to the Delphix Engine
DSP 8415/tcp - Delphix Session Protocol connections (Optionally needed for features like Windows Authentication, Netbackup/Commvault Support)
Remote Registry 445/tcp – Discovery of SQL instances and ports
* JDBC Connectivity is needed to the Source Database:
- directly from Delphix Engine if using Sql Authentication
- from Staging/Connector Host if using Windows Authentication

Case 4: Migrating Backup Files

Migrating Backup Files

Case 5: SMB anonymous access

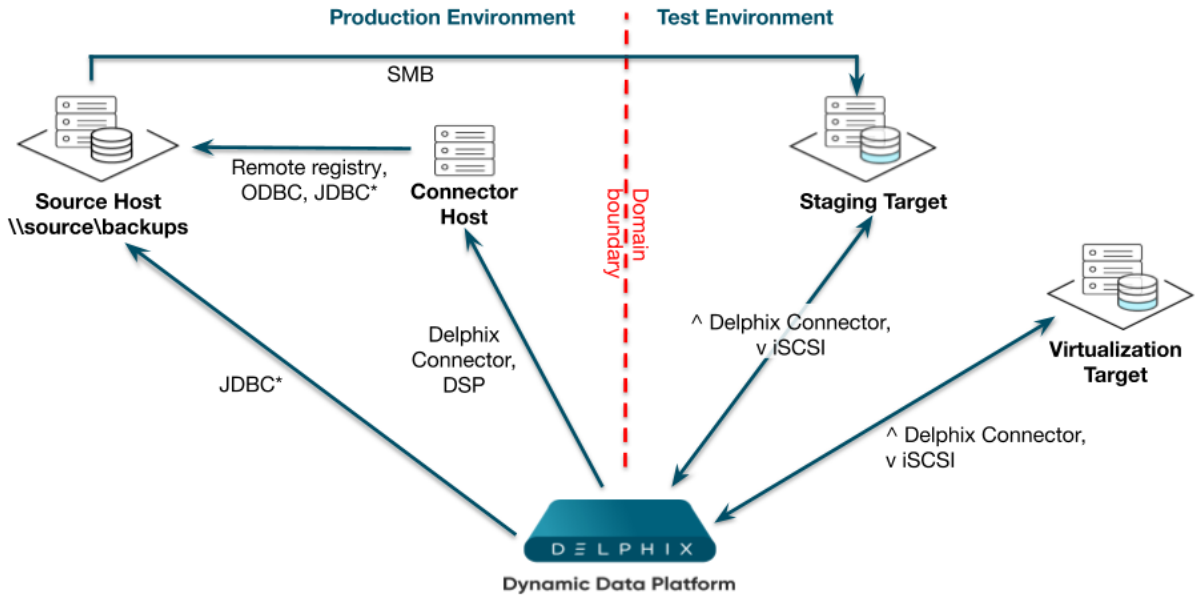
In this example (shown by the *SMB Anonymous Access* diagram below), a Windows SMB connection is traversing domains that do not have a trust relationship. This approach is problematic because there is no simple configuration for SMB file sharing that does not rely on domain trusts. As a result, there is no way to specifically grant accounts in the TEST domain access to SMB shares in the PRODUCTION domain.

Because such users cannot be authenticated, they are treated as “anonymous” users and do not have permission to any resources by default.

Windows provides an “Everyone” group. However, this group still only applies to accounts that can be authenticated in the domain, so you cannot use that group in this case. There is still a way to configure access to the shared backup location on \source\backups by accounts in other domains, such as TEST. However, because it relies on anonymous access, you will need to consider the security implications of enabling this method, as well as measures that could mitigate any additional risk in your environment – for example, a private VLAN or IPSEC between hosts.

1. Enable the “Guest” account on the server source Server – for example, **\SQLPROD**.
2. Create a share where full and transaction log backups will be stored – for example, **\SQLPROD\backups**.
3. Configure read-only security access for both the folder security permissions on the shared directory and the share permissions for the “guest” account.

Case 5: SMB Anonymous Access



Protocol & Port Connection information
SMB/445 – Server Messaging Block (file transfer)
iSCSI 3260/tcp – TCP network storage traffic initiated from the target
JDBC 1433/tcp – Get version, backup history, and LSN data
Delphix Connector v1 9100/tcp – Custom control and data specific to the Delphix Engine
DSP 8415/tcp - Delphix Session Protocol connections (Optionally needed for features like Windows Authentication, Netbackup/Commvault Support)
Remote Registry 445/tcp – Discovery of SQL instances and ports
* JDBC Connectivity is needed to the Source Database:
- directly from Delphix Engine if using Sql Authentication
- from Staging/Connector Host if using Windows Authentication

SMB Anonymous Access

Here are some additional links from Microsoft that relate to anonymous sharing:

- Network access: Let Everyone permissions apply to anonymous users
- Network access: Shares that can be accessed

Quick start guide for SQL Server (Microsoft SQL Server on Windows)

This quick start guide, which is excerpted from the larger User Guide, is intended to provide you with a quick overview of working with SQL Server database objects in the Delphix Engine. It does not cover any advanced configuration options or best practices, which can have a significant impact on performance. It assumes that you are working in a Lab/Dev setting and attempting to quickly test Delphix functionality. It assumes you will use the VMware Hypervisor. It assumes you are running supported combinations of software as explained here: Supported OS, SQL Server, and Backup Software Versions for SQL Server.

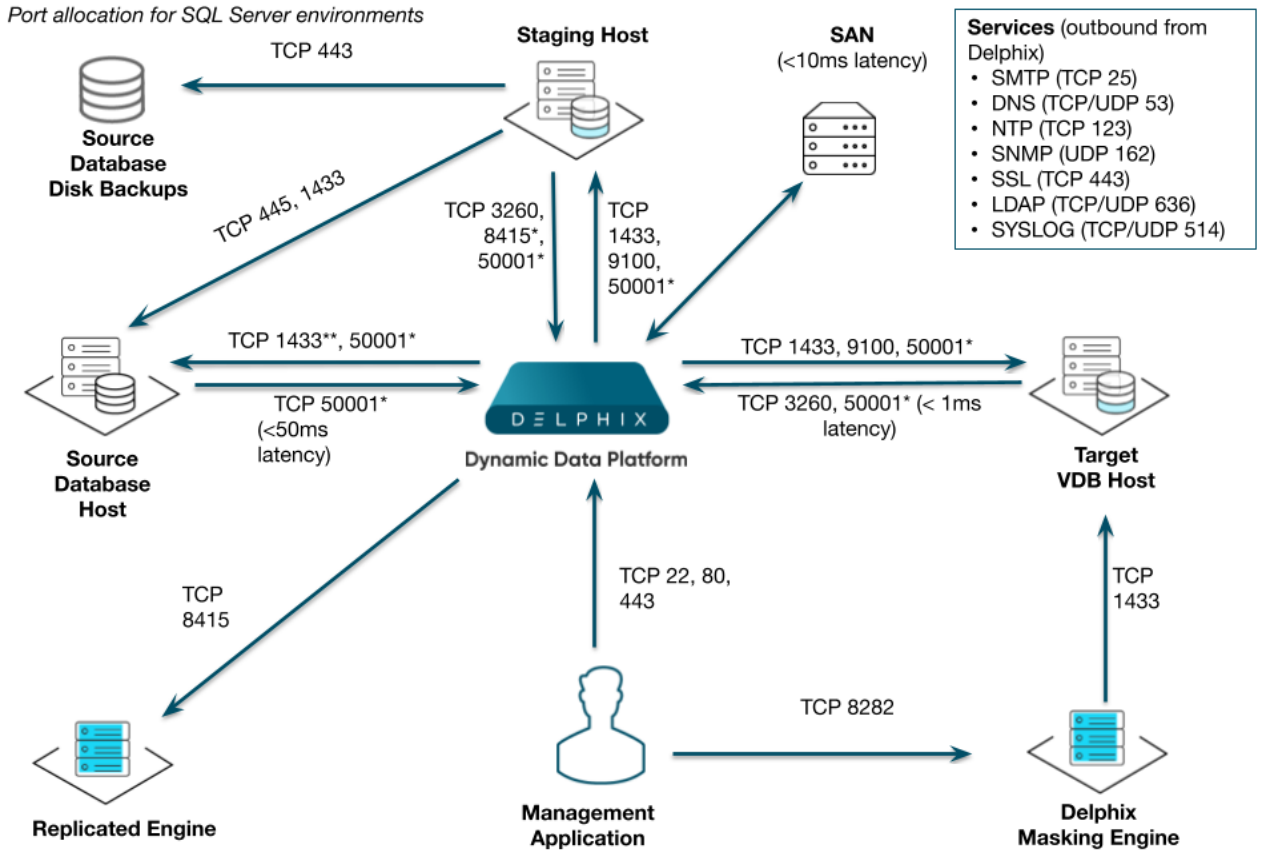
Overview

In this guide, we will walk through deploying a Delphix Engine, starting with configuring Source, Staging (aka Validated Sync), and Target database environments on Windows servers. We will then create a dSource, and provision a VDB.

The following diagram describes the engine topology for SQL Server environments. It illustrates the recommended ports to be open from the engine to remote services, to the Delphix Engine, and to the Source, Target and Validated Sync Environments.

Port 1433 is required between the Delphix Engine and AG (Availability Group) cluster sources.

For purposes of the QuickStart, you can ignore any references to Replication or Masking, such as the engines shown in the diagram below.



Note: SQL Server listener typically runs on TCP 1433. In cases where other ports are used, substitute for 1433 above.
**Port 1433 access to Source Database Host from Delphix Engine is needed only for SQL Authentication. It is not needed in case of Windows Authentication.

Deploy OVA on VMware

Use the Delphix-supplied OVA file to install the Delphix Engine. The OVA file is configured with many of the minimum system requirements. The underlying storage for the install is assumed to be redundant SAN storage.

1. Download the OVA file from https://download.delphix.com. You will need a support login from your sales team or a welcome letter.
    a. Navigate to the Delphix Product Releases
2. Login using the vSphere client to the vSphere server (or vCenter Server) where you want to install the Delphix Engine.
3. In the vSphere Client, click **File**.
4. Select **Deploy OVA Template**.
5. Browse to the OVA file.
6. Click **Next**.
7. Select a **hostname** for the Delphix Engine. This hostname will also be used in configuring the Delphix Engine network.
8. Select the **data center** where the Delphix Engine will be located.
9. Select the **cluster** and the **ESX host**.
10. Select one (1) **data store** for the **Delphix OS**. This datastore can be **thin-provisioned** and must have enough free space to accommodate the 127GB comprising the Delphix operating system.
11. Select four (4) or more **data stores** for Database Storage for the Delphix Engine. The Delphix Engine will stripe all of the Database Storage across these VMDKs, so for optimal I/O performance, each VMDK must be

equal in size and be configured **Thick Provisioned - Eager Zeroed**. Additionally, these VMDKs should be distributed as evenly as possible across all four SCSI I/O controllers.

12. Select the **virtual network** you want to use.
    If using multiple physical NICs for link aggregation, you must use vSphere NIC teaming. Do not add multiple virtual NICs to the Delphix Engine itself. The Delphix Engine should use a single virtual network. For more information, see Optimal Network Architecture for the Delphix Engine.
13. Click **Finish**. The installation will begin and the Delphix Engine will be created in the location you specified.
14. Once the installation has completed, power on the Delphix Engine and proceed with the initial system configuration as described in Setting Up Network Access to the Delphix Engine.

ⓘ If your source database is 4 TB, you probably need 4 TB of storage for the Delphix Engine. Add at least 4 data disks of similar size for the Delphix VM. For example: for a source database of 4 TB, create 4 VMDKs of 1 TB each.

ⓘ For a full list of requirements and best practice recommendations, see Virtual Machine Requirements for VMware Platform.

Setup network access to Delphix engine

1. Power on the Delphix Engine and open the Console.
2. Wait for the Delphix Management Service and Delphix Boot Service to come online. This might take up to 10 minutes during the first boot. Wait for the large orange box to turn green.
3. Press any key to access the sysadmin console.
4. Enter `sysadmin@SYSTEM` for the username and `sysadmin` for the password.
5. You will be presented with a description of available network settings and instructions for editing.

```
Delphix Engine Network Setup

To access the system setup through the browser, the system must first be
configured for networking in your
environment. From here, you can configure the primary interface, DNS, hostname,
and default route. When DHCP is
configured, all other properties are derived from DHCP settings.

To see the current settings, run "get." To change a property, run "set
<property>=<value>." To commit your changes,
run "commit." To exit this setup and return to the standard CLI, run "discard."

    defaultRoute    IP address of the gateway for the default route -- for
 example, "1.2.3.4."

    dhcp            Boolean value indicating whether DHCP should be used for
 the primary interface. Setting this value
                    to "true" will cause all other properties (address,
hostname, and DNS) to be derived from the DHCP
                    response

    dnsDomain       DNS Domain -- for example, "delphix.com"
```

```
    dnsServers      DNS server(s) as a list of IP addresses -- for example,
"1.2.3.4,5.6.7.8."

    hostname        Canonical system hostname, used in alert and other logs --
for example, "myserver"

    primaryAddress  Static address for the primary interface in CIDR notation
-- for example, "1.2.3.4/22"

Current settings:

    defaultRoute: 192.168.1.1
    dhcp: false
    dnsDomain: example.com
    dnsServers: 192.168.1.1
    hostname: Delphix
    primaryAddress: 192.168.1.100/24
```

6. Configure the `hostname`. If you are using DHCP, you can skip this step. **Note** : Use the same `hostname` you entered during the server installation.

```
delphix network setup update *> set hostname=<hostname>
```

7. Configure DNS. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set dnsDomain=<domain>
delphix network setup update *> set dnsServers=<server1-ip>[,<server2-ip>,...]
```

8. Configure either a static or DHCP address. **Note** The static IP address must be specified in CIDR notation (for example, `192.168.1.2/24`)
    - DHCP Configuration

    ```
    delphix network setup update *> set dhcp=true
    ```

    - Static Configuration

    ```
    delphix network setup update *> set dhcp=false
    delphix network setup update *> set primaryAddress=<address>/<prefix-len>
    ```

9. Configure a default gateway. If you are using DHCP, you can skip this step.

```
delphix network setup update *> set defaultRoute=<gateway-ip>
```

10. Commit your changes. Note that you can use the `get` command prior to committing to verify your desired configuration.

```
delphix network setup update *> commit
```

```
Successfully committed network settings. Further setup can be done through the
browser at:
        http://<address>
Type "exit" to disconnect, or any other commands to continue using the CLI.
```

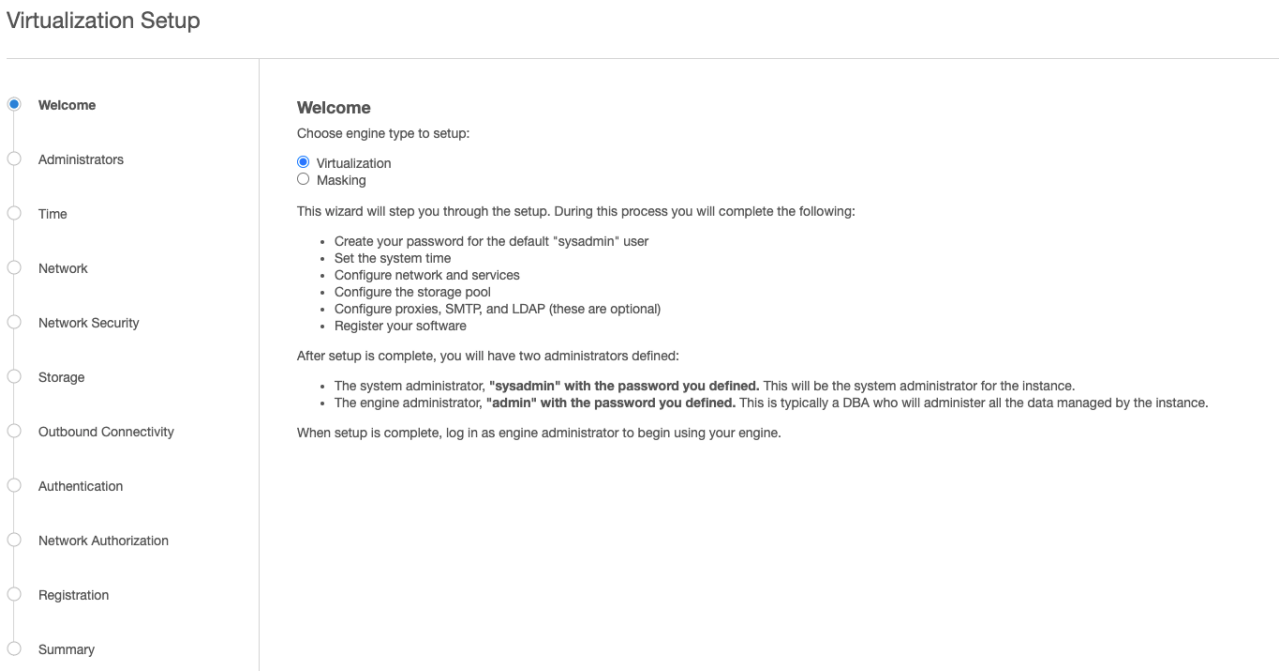
11. Check that you can now access the Delphix Engine through a Web browser by navigating to the displayed IP address, or hostname if using DNS.
12. Exit setup.

```
delphix> exit
```

Setting up the Delphix engine

Once you set up the network access for Delphix Engine, navigate to the Delphix Engine URL in your browser for server setup.

The welcome screen below will appear for you to begin your Delphix Engine setup.



The setup procedure uses a wizard process to take you through a set of configuration screens:

- Administrators
- Time
- Network
- Network Security
- Storage
- Outbound Connectivity
- Authentication
- Network Authorization
- Registration
- Summary

1. Connect to the Delphix Engine at `http:///login/index.html#serverSetup`.
   The **Delphix Setup** application will launch when you connect to the server.
   Enter your **sysadmin** login credentials, which initially defaults to the username **sysadmin**, with the initial default password of **sysadmin**. On first login, you will be prompted to change the initial default password.
2. Click **Next**.

Administrators

The Delphix Engine supports two types of administrators:

- System Administrator (**sysadmin**) - this is the engine system administrator. The sysadmin password is defined here**.**
- Engine Administrator (**admin)** - this is typically a DBA who will administer all the data managed by the engine.

On the Administrators tab, you set up the sysadmin password by entering an email address and password. The details for the admin are displayed for reference.

> The default domain user created on Delphix Engines from 5.3.1 is known as **admin** instead of delphix_admin. When engines created before 5.3.1 are upgraded to 5.3.1 or later they will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then Disabling delphix_admin.

System time

The engine time is used as the baseline for setting policies that coordinate between virtual databases and external applications.

Choose your option to set up system time in this section. For a Quick Start, simply set the time and your timezone. You can change this later.

Network

The initial out-of-the-box network configuration in the OVA file is set to use a single VMXNET3 network adapter.

You have already configured this in the initial configuration. Delphix supports more advanced configurations, but you can enable those later.

Storage

You should see the data storage VMDKs or RDMs you created during the OVA installation. Click **Next** to configure these for data storage.

Serviceability

Choose your options to configure serviceability settings.

For a Quick Start, accept the defaults. You can change this later.

Authentication

Choose your options to configure authentication services.

For a Quick Start, accept the defaults. You can change this later.

Registration

If the Delphix Engine has access to the external Internet (either directly or through a web proxy), then you can auto-register the Delphix Engine:

1. Enter your **Support Username** and **Support Password**.
2. Click **Register**.

If external connectivity is not immediately available, you must perform manual registration.

1. Copy the **Delphix Engine registration code** in one of two ways:
    a. Manually highlight the registration code and copy it to clipboard. Or,
    b. Click **Copy Registration Code to Clipboard**.
2. Transfer the Delphix Engine's registration code to a workstation with access to the external network Internet. For example, you could e-mail the registration code to an externally accessible e-mail account.
3. On a machine with access to the external Internet, please use your browser to navigate to the Delphix Registration Portal at http://register.delphix.com.
4. Login with your Delphix support credentials (username and password).
5. Paste the **Registration Code**.
6. Click **Register**.

Although your Delphix Engine will work without registration, we strongly recommend that you register each Delphix Engine as part of the setup. Failing to register the Delphix Engine will impact its supportability and security in future versions

To regenerate the registration code for a Delphix Engine please refer to, Regenerating the Delphix Engine Registration Code. Delphix strongly recommends that you regenerate this code and re-register the engine regularly to maximize the Support Security of the Delphix Engine. Delphix recommends doing this every six months.

Summary

The final summary tab will enable you to review your configurations for System Time, Network, Storage, Serviceability, and Authentication.

1. Click the **Back** button to go back and to change the configuration for any of these server settings.
2. If you are ready to proceed, then click **Submit**.
3. Click **Yes** to confirm that you want to save the configuration.
4. Click **Setup** to acknowledge the successful configuration.
5. There will be a wait of several minutes as the Delphix Engine completes the configuration.

SQL server source hosts and databases

Source host requirements

Windows servers that will be added as Source Environments must meet the following requirements:

Source Host Requirement	Explanation
To allow Delphix-initiated backups, the service account running each SQL Server instance (the Instance Owner) should be one of: • A domain user (e.g. `MYDOMAIN\accountname`) (RECOMMENDED) • A Managed Service Account or Group Managed Service Account (`MYDOMAIN\accountname$`) (requires Windows 2012 and later, and SQL Server 2008R2 or later) • The LOCAL SYSTEM account (`NT AUTHORITY\SYSTEM`) • The NETWORK SERVICE account (`NT AUTHORITY\NETWORK SERVICE`)	Backups initiated by the Delphix Engine (Delphix Managed Backups or manual snapshots which request a backup) will fail if the Delphix Engine cannot map the Instance Owner to an Active Directory user or computer object.
The source host, proxy, and staging environments must have appropriate cross-domain trust relationships	For more information on these requirements, see the document Delphix in Multi-domain Windows Environments.

Recommended source Windows user requirement

Aligning to our zero-trust approach, “Delphix OS” user permissions on the Source can now be configured with the least privilege necessary from previous super-user “Backup Operator” requirements.

Recommended source Windows user requirement	Explanation
Have the "Log on as batch" privilege on the source host	This permission is required for remote PowerShell execution. This privilege can be assigned through the Local Security Policy (Local Policies → User Rights Assignment → Log on as batch job).
Have read permission for “`Computer\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurePipeServers\winreg`” on the source host	This permission is required to have access to the remote registry. Delphix uses this privilege to discover SQL Server instances and gather system details, using Windows remote registry access.
Be a member of the Users group on the Staging Host	In order to discover and query SQL Server instances as the Source Windows User, scripts are run on that user at the Staging Host.

Recommended source Windows user requirement	Explanation
Have the "Log on as batch" privilege on the Staging host	This permission is required for remote PowerShell execution. This privilege can be assigned through the Local Security Policy (Local Policies → User Rights Assignment → Log on as batch job).
Be able to login to each SQL Server instance that the Delphix Engine will communicate with	These requirements are described in the *Source Database Login Requirements* section below.

This recommended permission requires editing the registry on each Source host(s) so that membership of the Backup Operators group is not required. The Backup Operators group confers additional permissions that are not used by Delphix, such as being able to shut down the host.

Deprecated source Windows user requirements

Source Windows User Requirement	Explanation
Be a member of the Backup Operators group on the Source Host	Delphix uses this privilege to discover SQL Server instances and gather system details, using Windows remote registry access.

Source database login requirements

The Delphix Engine requires SQL Server logins to be created on each SQL Server instance that the Delphix Engine will communicate with:

- A database login for Environment discovery and monitoring, specified when Adding an Environment. This must be a Windows Authentication login for the Source Windows User configured in the previous section.
- A database login for dSource (Source Database) monitoring and interaction, specified when Linking a dSource. This user can be:
- The same as the Source Windows User;
- A different Windows Authentication login (this user must also have Log on as a Batch Job privileges on the Staging host); or
- A SQL Authentication login

These users must have the following permissions on each instance:

Object	Privileges Required	dSource User	Environment User	Purpose
Server	`CONNECT SQL`	X	X	Access to the SQL Server instance
Database: `master`	`db_datareader`	X	X	Access to information about attached databases
Database: `msdb`	`db_datareader`	X		Access to the backup history

Object	Privileges Required	dSource User	Environment User	Purpose
Each user database to be linked	`PUBLIC`	X		Delphix will periodically run queries to check the current size of the database
Each user database to be linked	`db_backupoperator`	X		**Optional**: Required for backups to be initiated by Delphix (using Delphix Managed Backups, or when Delphix initiates a backup during a manual Snapshot)
Server	`VIEW ANY DEFINITION`	X	X	**Optional:** Required for the discovery of databases in Availability Groups
Server	`VIEW SERVER STATE`	X	X	**Optional:** Required for the SnapSync and discovery of Availability Groups
Object: `master.dbo.sqbutility`	`EXECUTE`	X		**Optional:** Required when using backups created by Red Gate SQL Backup
Object: `master.dbo.xp_sqllightspeed_version`	`EXECUTE`	X		**Optional:** Required when using backups created by Quest LiteSpeed for SQL Server

List of source tables accessed by the Delphix engine

Using the db_datareader permission, the Delphix Engine accesses the following system tables in the master and msdb databases on the source host:

System table	Justification
master.sys.databases	Used to determine the name and recovery model of databases within discover SQL Server instances
master.sys.availability_groups	Used for discovering all the availability groups within an Availability Group source environment.

System table	Justification
master.sys.availability_group_listeners	Used for discovering all the availability group listeners within an Availability Group source environment. • A requirement for dSource linking of SQL Server clustered databases (replicas) is to provide an AG (Availability Group) listener for AG cluster source discovery. This is implemented on AG cluster source discovery as a failsafe if AG cluster source database authentication configuration change down the line, ensuring the Delphix engine has a way to reach the cluster and continue certain operations.
master.sys.availability_databases_cluster	Used for discovering all the availability group clusters within an Availability Group source environment.
master.sys.availability_replicas	Used for discovering all the availability group replicas within an Availability Group source environment.
master.sys.database_files	Used to determine the size of databases and whether filestream is enabled for a database
master.sys.dm_exec_requests	Used to enable Delphix to report backup operation progress
master.sys.master_files	Used to determine the primary file of a database
master.sys.filegroups	Used to determine the filegroups of a database so that Delphix can configure VDBs with the same filegroups
msdb.dbo.backupset	Used to determine new backups that have been taken. This table is regularly queried to find out if a new backup image has been taken and needs to be synchronized with Delphix.
msdb.dbo.backupmediafamily	Used to determine the physical device names of the backup files comprising a backup.

SQL server staging hosts and databases

Staging host requirements

Windows servers which will be added as Staging Environments must meet the following requirements:

Staging Host Requirement	Explanation
The service account running each SQL Server instance (the Instance Owner) must be one of: • A domain user (e.g. `MYDOMAIN\accountname`) (RECOMMENDED) • A Managed Service Account or Group Managed Service Account (`MYDOMAIN\accountname$`)(requires Windows 2012 and later, and SQL Server 2008R2 or later) • The LOCAL SYSTEM account (`NT AUTHORITY\SYSTEM`) • The NETWORK SERVICE account (`NT AUTHORITY\NETWORK SERVICE`)	Access to existing database backups for Snapshots and Validated Sync operations will fail if the service account cannot access backups created by the Source database instance.
The source host, proxy and staging environments must have appropriate cross-domain trust relationships.	For more information on these requirements, see the document Delphix in Multi-domain Windows Environments.
The edition of the installed SQL Server instance(s) must support all database features used by linked Source databases.	SQL Server may raise an error during some dSource operations if the Staging SQL Server Instance does not support features used by the Source Database. This is most easily addressed by using the same edition of SQL Server as the Source database. Features in use on the Source database can be checked using the `sys.dm_db_persisted_sku_features` dynamic view.
Must have both the Source and Staging Windows Users configured as local Windows users.	See the sections *Source Windows User Requirements* and *Staging Windows User Requirements* for more detail.

Staging Host Requirement	Explanation
Delphix Connector software is installed and running.	See Installing the Delphix Connector Service on the Target Database Servers for instructions on installing the Delphix Connector.
Recommended iSCSI Registry settings must be in place.	See Requirements for Windows iSCSI Configuration and Knowledge Base Article KBA1251 for instructions on applying these settings*. *Please note that in 6.0.x, these performance settings are subject to change based on further tuning efforts by Delphix Engineering.
sqlcmd command line utility must be installed on the servers hosting VDBs and staging databases. The utility should be in the Delphix operating system user's PATH environment variable.	Delphix does not run sqlcmd (or any other process) directly on the source SQL Server instance. It runs sqlcmd on the staging/connector host.

Staging Windows user requirements

The Delphix Engine needs a Windows domain user — for example, `MYDOMAIN\delphix_os` — to be specified when adding Staging environments to the Delphix Engine. This user must have the following permissions:

Staging User Requirement	Explanation
Be a member of the Windows "Local Administrators" group on the Staging Host	We require this permission for mounting iSCSI LUNs presented by the Delphix Engine to the staging and target hosts. Microsoft utilities used by the Delphix Engine, such as diskpart, require membership of this group.
Have the "Log on as batch" privilege on the Staging host	We require this permission for remote PowerShell execution. This privilege can be assigned through the Local Security Policy (Local Policies → User Rights Assignment → Log on as batch job).
Be able to access existing backups from the Source Database	If accessing existing native, Red Gate, or LiteSpeed backups, the following permissions are required: • The Staging Windows User and the service account running SQL Server must both have permission to access the database backups via SMB (Windows file sharing). • The Staging Windows User and the service account running SQL Server must both have NTFS Permissions to access the database backups. Separate documents describe requirements if Linking a dSource from a NetBackup SQL Server Backup or Linking a dSource from a Commvault SQL Server Backup.

Staging User Requirement	Explanation
Be able to login to each SQL Server instance that the Delphix Engine will communicate with	These requirements are described in the *Staging Database Login Requirements* section below.

Staging database login requirements

The Delphix Engine requires a Windows Authentication login to be created for the Staging Windows User on each SQL Server instance that the Delphix Engine will communicate with, with the following permissions:

Object	Privileges Required	Delphix OS User (Windows Login)	Purpose
Server	`CONNECT SQL`	Y	Access to the SQL Server instance.
Server	`sysadmin`	Y	The staging and target databases are managed and administered completely by Delphix. Our functionality requires many administrative operations on those databases and requires full access to them. Since database ownership can be changed by customers as part of configuring virtual databases, we require the sysadmin role to continue to administer the databases.

SQL server target hosts and databases

Target host requirements

Windows servers which will be added as Target Environments must meet the following requirements:

Target Host Requirement	Explanation
Delphix Connector software is installed and running	See Installing the Delphix Connector Service on the Target Database Servers for instructions on installing the Delphix Connector.
Recommended iSCSI Registry settings must be in place	See Requirements for Windows iSCSI Configuration and Knowledge Base Article KBA1251 for instructions on applying these settings.

Target Host Requirement	Explanation
The edition of the installed SQL Server instance(s) must support all database features used by linked Source databases	SQL Server may raise an error during some dSource operations if the Staging SQL Server Instance does not support features used by the Source Database. This is most easily addressed by using the same edition of SQL Server as the Source database. Features in use on the Source database can be checked using the `sys.dm_db_persisted_sku_features` dynamic view.
sqlcmd command line utility must be installed on the servers hosting VDBs and staging databases. The utility should be in the Delphix operating system user's PATH environment variable.	Delphix does not run sqlcmd (or any other process) directly on the source SQL Server instance. It runs sqlcmd on the staging/connector host.

Target Windows user requirements

The Delphix Engine needs a Windows domain user — for example, `MYDOMAIN\delphix_os` — to be specified when adding Target environments to the Delphix Engine. This user must have the following permissions:

Target User Requirement	Explanation
Be a member of the Windows "Local Administrators" group on the Target Host	We require this permission for mounting iSCSI LUNs presented by the Delphix Engine to the staging and target hosts. Microsoft utilities used by the Delphix Engine, such as diskpart, require membership of this group.
Have the "Log on as batch" privilege on the Target Host	We require this permission for remote PowerShell execution. This privilege can be assigned through the Local Security Policy (Local Policies → User Rights Assignment → Log on as batch job).
Be able to login to each SQL Server instance that the Delphix Engine will communicate with	These requirements are described in the *Target Database Login Requirements* section below.
Delphix Connector software is installed and running	See Installing the Delphix Connector Service on the Target Database Servers for instructions on installing the Delphix Connector.

Target database login requirements

The Delphix Engine requires a Windows Authentication login to be created for the Target Windows User on each SQL Server instance that the Delphix Engine will communicate with, with the following permissions:

Object	Privileges Required	Delphix OS User (Windows Login)	Purpose
Server	`CONNECT SQL`	Y	Access to the SQL Server instance
Server	`sysadmin`	Y	The staging and target databases are managed and administered completely by Delphix. Our functionality requires many administrative operations on those databases and requires full access to them. Since database ownership can be changed by customers as part of configuring virtual databases, we require the sysadmin role to continue to administer the databases.

Supported roles for failover cluster instances and always on availability groups

Failover Cluster Instances and Always On Availability groups cannot be used as Staging Environments, and VDBs cannot be provisioned into Availability Groups.

When adding a Failover Cluster Instance or Always On Availability Group, all nodes of the cluster must meet the requirements described in this document.

The following table shows how different SQL Server instance types should be added from the Add Environment screen:

		Environment Role		
Instance Type	**Added As**	**Source Environment**	**Staging Environment**	**Target Environment**
Standalone Instance	Standalone	Y*	Y	Y
Failover Cluster Instance (FCI)	Standalone	Y*	N	N
Failover Cluster Instance (FCI)	Cluster	N	N	Y
Always On Availability Group (AG)	Cluster	Y	N	N**

* Databases that are participating in Availability Groups will not be discovered during the discovery of a Standalone environment.

** VDBs cannot be provisioned into availability groups. However, SQL Server instances that participate in an Availability Group can also be added as Standalone Instances.

Using a Failover Cluster Instance as both Source and Target

A Failover Cluster Instance added as an environment once (as either a Source or Target environment) cannot be used as both a Source and Target.

If this is required, the environment can be added twice:

- Once as a Standalone Source environment
- Once as a Cluster Target environment

The Standalone Source environment can be used for linking dSources, and the Cluster Target environment is used for provisioning VDBs.

As suggested by the Best Practice note earlier in this article, this is not a recommended configuration. Where possible, SQL Server failover cluster instances that the Delphix Engine will use as a target should not be used to host databases other than Delphix VDBs.

Requirements for failover cluster target environments

The following additional requirements exist for Windows Failover Cluster Instances added as Target Environments, to be used for VDB Provisioning.

- You must first add each node in the Window Failover Cluster individually as a standalone target environment, using a non-clustered address. See Adding a SQL Server Standalone Target Environment.
    - The Delphix Engine may show a warning that it cannot discover the Failover Cluster Instances on each standalone host. This is expected.
- Each clustered SQL Server instance must have at least one clustered disk added to the clustered instance resource group, which can be used for creating mount points to Delphix storage.
    - The clustered drive must have a drive letter assigned to it.
    - The clustered drive must be formatted using the "GUID Partition Table (GPT)" partition style, in order for the Delphix Engine to automatically discover the drive letter as a valid option for the cluster instance. An MBR-formatted disk requires manual verification outside of Delphix that the disk has been correctly added to the MSSQL clustered resource group prior to creating the VDB. When provisioning the VDB, you must manually specify the desired MBR disk, because it will not appear in the Delphix GUI.
    - The clustered drive must be added to the clustered instance resource group as a dependency in the Failover Cluster Manager.
- Each node in the cluster must have the Failover Cluster Module for Windows PowerShell feature installed.
    - While running Windows PowerShell as an administrator, enter this command to test that the module is available: `Import-Module FailoverClusters`
- An additional target environment that can be used as a Connector Environment must exist.
    - This environment must **not** be a node in the cluster.
    - This environment should be part of the same Active Directory domain as the cluster.

Add the SQL server source environment

Delphix does not require running the Connector on your source. Instead, you'll use the Validated Sync environment as a connector environment to discover the source by proxy.

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.

4. Next to **Environments**, click the **Actions** menu, and select **Add Environment**.
5. In the **Add Environment**wizard, Host and Server tab select:
    a. Host OS: **Windows**
    b. Host Type: **Source**
    c. Server Type:
    d. If you are adding a Windows Server Failover Cluster (WSFC), add the environment based on which WSFC feature the source databases use:
        - Failover Cluster Instances Add the environment as a **standalone** source using the **cluster name** or **address**.
        - AlwaysOn Availability Groups Add the environment as a **cluster** source using the **cluster name** or **address**.
    e. Otherwise, add the environment as a **standalone** source.
6. Click **Next**.
7. In the Environment Settings tab select a **Connector Environment**. Connector environments are used as a proxy for running discovery on the source. If no connector environments are available for selection, you will need to set them up as described in Adding a SQL Server Standalone Target Environment. Connector environments must:
    - have the Delphix Connector installed
    - be registered with the Delphix Engine from the host machine where they are located.
8. Enter the **Environment Name**, **Node Address**, **OSUsername**, and **OSPassword** for the source environment.
9. Click **Submit.**

As the new environment is added, you will see multiple jobs running in the Delphix Admin Job History to Create and Discover an environment. In addition, if you are adding a cluster environment, you will see jobs to Create and Discover each node in the cluster and their corresponding hosts. When the jobs are complete, you will see the new environment added to the list in the **Environments** panel. If you don't see it, click the **Actions** menu and select **Refresh All**.

Linking a SQL server data source (dSource)

Linking a dSource will ingest data from the source and create a dSource object on the engine. The dSource is an object that the Delphix Virtualization Engine uses to create and update virtual copies of your database. As a virtualized representation of your source data, it cannot be managed, manipulated, or examined by database tools.

For an overview of all dSource related actions, please Managing Data Sources and Syncing Data.

When linking a dSource from a SQL Server source database, Delphix offers several different methods of capturing backup information:

- SQL Server Managed Backups, where the SQL Server source database schedules and initiates backups and the Delphix Engine captures them
    - Full backups
    - Full or differential backups
    - Transaction log backups (with LogSync disabled)
    - Transaction log backups (with LogSync enabled)
- Delphix Managed Backups, where the Delphix Engine schedules and initiates the backups from the source database, and captures them

Procedure

1. Login to the **Delphix Management** application.
2. Navigate to **Manage > Datasets.**
3. Click the plus icon and select **Add dSource**.
4. In the **Add dSource** wizard, select the source database with the correct environment user-specified.

5. Select user type for source database authentication and enter the login credentials. Enter username and password for Database user or Domain (Windows) user. For Environment User, select a source environment user from the dropdown list and click **Next**.
6. Enter a name and select a group for your dSource. Adding a dSource to a dataset group lets you set Delphix Domain user permissions for that database and its objects, such as snapshots. See the topics under Users and Groups for more information.
7. Select the **Data Management** settings needed. For more information, Data Management Settings for SQL Server Data Sources.
8. Select the Staging environment and SQL Instance that will be used to manage the staging database used for validated sync of the dSource.
9. Select any policies for the new dSource.
10. Enter any scripts that should be run on the **Hooks** page.
11. Review the dSource Configuration and Data Management information, and then click **Submit**.

Provisioning a SQL server virtual database (VDB)

1. Login to the **Delphix Management** application.
2. Select **Manage** > **Datasets**.
3. Select a **dSource**.
4. Click **Timeflow** tab.
5. Next to a snapshot select the Provision VDB icon. The **Provision VDB** panel will open, and the **Database Name** and **Recovery Model** will auto-populate with information from the dSource.
6. Select a **target environment**.
7. Select an **Instance** to use.
8. If the selected target environment is a Windows Failover Cluster environment, select a drive letter from **Available Drives**. This drive will contain volume mount points to Delphix storage.

> Windows Cluster Volume Management Software Requirements
>
> Only cluster volumes managed by the native Windows Volume Manager are supported. For example, cluster volumes managed by Veritas VxVM are not supported.
>
> If you use third-party volume management software, create a new LU (recommended to be 10GB in size) and add this LU as a clustered resource to the SQL Server instance using native Windows volume management tools. Assign a drive letter for this LU. You can then use this LU as the volume mount point location for Delphix VDB provisioning

9. Enter a **VDB Name** and select a **Target Group** for the VDB.
10. Enable **Auto VDB Restart** to allow the Delphix Engine to automatically restart the VDB when it detects target host reboot.
11. Click **Next**.
12. Select a **Snapshot Policy** for the VDB. Click **Next**.
13. Specify any **Pre-** or **Post-Scripts** that should be used during the provisioning process.
14. Click **Next**.
15. The final summary tab will enable you to review your configurations.
16. Click **Submit**.

When provisioning starts, the VDB will appear in the **Datasets** panel. Select the VDB and navigate to the **Status** tab to see the progress of the job. When provisioning is complete, you can see more information on the **Configuration** tab.

> You can select a SQL Server instance that has a higher version than the source database and the VDB will be automatically upgraded. For more information about compatibility between different versions of SQL Server, see SQL Server Support Matrix.

Provisioning by snapshot or logSync

When provisioning by snapshot, you can provision to the start of any particular snapshot, either by time or LSN.

> You can take a new snapshot of the dSource and provision from it by clicking the **Camera** icon.

Provisioning By Snapshot	**Description**
Provision by Time	You can provision to the start of any snapshot by selecting that snapshot card from the **TimeFlow** tab, or by selecting the time icon and entering a value in the time entry fields. The values you enter will snap to the beginning of the nearest snapshot.
Provision by LSN	You can use **Provision by LSN** control to open the LSN entry field. Here, you can type or paste in the LSN to which you want to provision. After entering a value, it will "snap" to the start of the closest appropriate snapshot. Provisioning a SQL Server VDB Procedure

Next steps

Congratulations! You have provisioned your first virtual database!

Now, perform some simple functional tests with your application. You can connect your app to the VDB using standard TNS/JDBC techniques. Delphix has already registered the VDB for you on the target listener.

We suggest the following next steps:

1. Drop a table and use the VDB Rewind feature to test the recovery of your VDB.
2. Take a snapshot of your dSource and refresh your VDB to quickly get fresh production data.
3. Provision a new VDB from your VDB to test sharing data quickly with other teams.
4. Mask your new VDB to protect sensitive data. Provision new VDBs from that masked VDB to quickly provide safe data to the development and QA teams.

SQL Server support and requirements

This section covers the following topics:

- Overview of requirements for SQL Server environments
- Requirements for Windows iSCSI configuration
- Network access requirements for SQL Server

To view SQL Server support matrix, see SQL Server matrix.

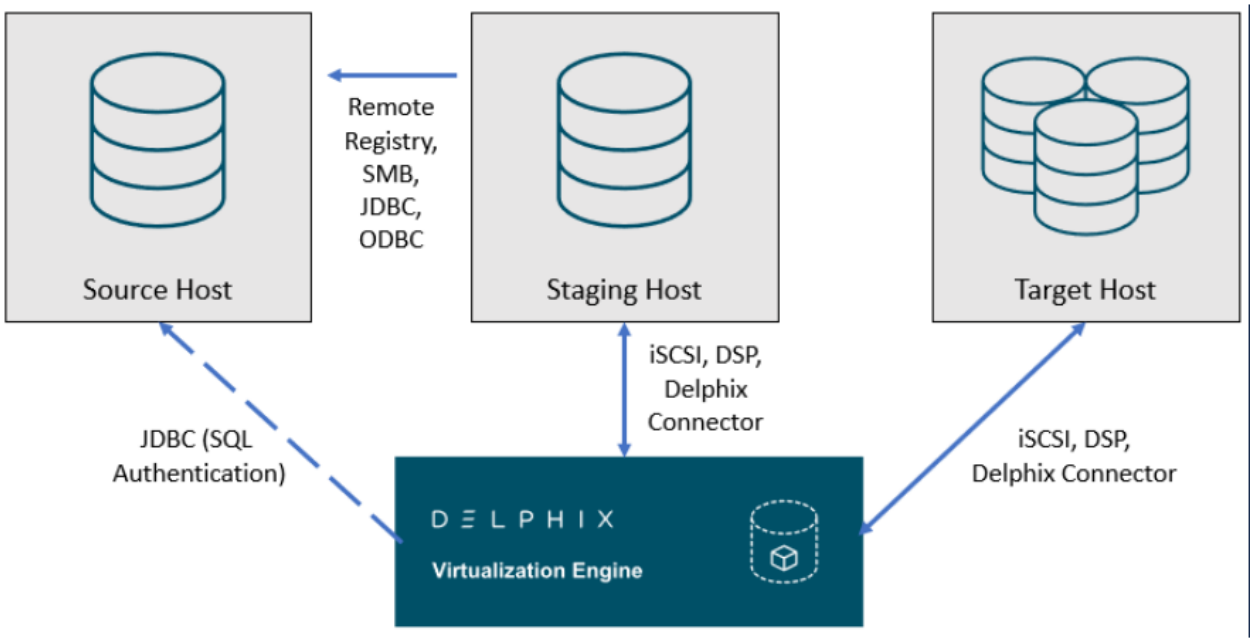
Overview of requirements for SQL Server environments

SQL server staging hosts and databases

This document identifies the requirements for interactions between the Delphix Engine and SQL Server environments and outlines the set of system tables to which we currently require access.

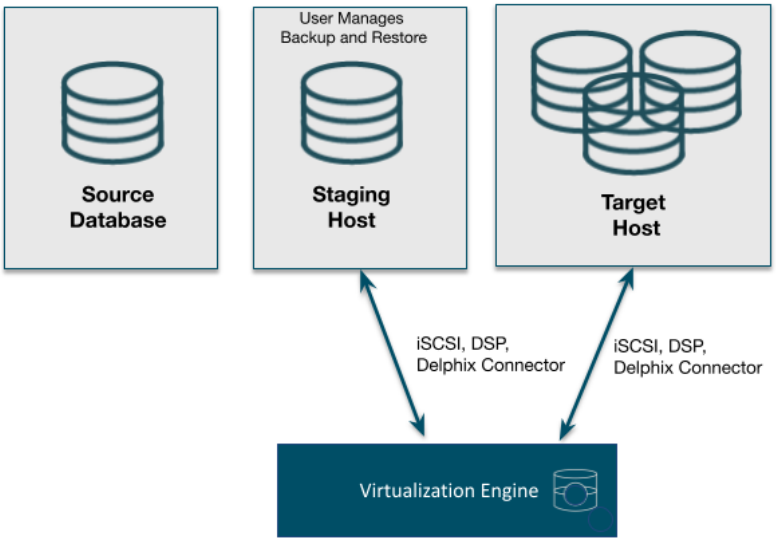
Delphix SQL server architectural diagram - Traditional pull architecture

This diagram depicts the environments and hosts with which the Delphix Engine interacts. Each type of environment has different requirements, which are described below.



Delphix SQL server architectural diagram - staging push architecture

This diagram depicts the Staging and target hosts with which the Delphix Engine interacts. Note that the Delphix Engine does not interact with the Source Database. For more details on the Staging Push mechanism, see Staging Push Mechanism with SQL Server and Staging Push Implementation for SQL Server.



SQL server source hosts and databases

Source host requirements

Windows servers that will be added as Source Environments must meet the following requirements:

Source Host Requirement	Explanation
To allow Delphix-initiated backups, the service account running each SQL Server instance (the Instance Owner) should be one of: • A domain user (e.g. `MYDOMAIN\accountname`) (RECOMMENDED) • A Managed Service Account or Group Managed Service Account (`MYDOMAIN\accountnameundefined`) (requires Windows 2012 and later, and SQL Server 2008R2 or later) • The LOCAL SYSTEM account (`NT AUTHORITY\SYSTEM`) • The NETWORK SERVICE account (`NT AUTHORITY\NETWORK SERVICE`)	Backups initiated by the Delphix Engine (Delphix Managed Backups or manual snapshots which request a backup) will fail if the service account cannot access backups created by the Source database instance
The source host, proxy and staging environments must have appropriate cross-domain trust relationships	For more information on these requirements, see the document Delphix in Multi-domain Windows Environments.

Recommended source Windows user requirements

Aligning to our zero-trust approach, “Delphix OS” user permissions on the Source can now be configured with the least privilege necessary from previous super-user “Backup Operator” requirements.

Recommended Source Windows User Requirements	Explanation
Have the "Log on as batch" privilege on the source host	This permission is required for remote PowerShell execution. This privilege can be assigned through the Local Security Policy (Local Policies → User Rights Assignment → Log on
Have read permission for `"Computer\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurePipeServers\winreg"` on the source host	This permission is required to have access to the remote registry. Delphix uses this privilege to discover SQL Server instances and gather system details, using Windows remote registry access.

Recommended Source Windows User Requirements	Explanation
Be a member of the Users group on the Staging Host	In order to discover and query SQL Server instances as the Source Windows User, scripts are run on that user at the Staging Host.
Have the "Log on as batch" privilege on the Staging host	This permission is required for remote PowerShell execution. This privilege can be assigned through the Local Security Policy (Local Policies → User Rights Assignment → Log on as batch job).
Be able to login to each SQL Server instance that the Delphix Engine will communicate with	These requirements are described in the Source Database Login Requirements section below.

This recommended permission requires editing the registry on each Source host(s) so that membership of the Backup Operators group is not required. The Backup Operators group confers additional permissions that are not used by Delphix, such as being able to shut down the host.

Deprecated source Windows user requirements

Source Windows User Requirement	Explanation
Be a member of the Backup Operators group on the Source Host	Delphix uses this privilege to discover SQL Server instances and gather system details, using Windows remote registry access.
Be a member of the Users group on the Staging Host	In order to discover and query SQL Server instances as the Source Windows User, scripts are run on that user at the Staging Host.
Have the "Log on as batch" privilege on the Staging host	We require this permission for remote PowerShell execution. This privilege can be assigned through the Local Security Policy (Local Policies → User Rights Assignment → Log on as batch job).
Be able to login to each SQL Server instance that the Delphix Engine will communicate with	These requirements are described in the Source Database Login Requirements section below.

Source database login requirements

The Delphix Engine requires SQL Server logins to be created on each SQL Server instance that the Delphix Engine will communicate with:

- A database login for Environment discovery and monitoring, specified when Adding an Environment. This must be a Windows Authentication login for the Source Windows User configured in the previous section.
- A database login for dSource (Source Database) monitoring and interaction, specified when Linking a dSource. This user can be:

- The same as the Source Windows User;
- A different Windows Authentication login (this user must also have Log on as a Batch Job privileges on the Staging host); or
- A SQL Authentication login

These users must have the following permissions on each instance:

Object	Privileges Required	dSource User	Environment User	Purpose
Server	`CONNECT SQL`	Y	Y	Access to the SQL Server instance
Database: `master`	`db_datareader`	Y	Y	Access to information about attached databases
Database: `msdb`	`db_datareader`	Y		Access to the backup history
Each user database to be linked	`PUBLIC`	Y		Delphix will periodically run queries to check the current size of the database
Each user database to be linked	`db_backupoperator`	Y		Optional: Required for backups to be initiated by Delphix (using Delphix Managed Backups, or when Delphix initiates a backup during a manual Snapshot)

Object	Privileges Required	dSource User	Environment User	Purpose
Server	`VIEW ANY DEFINITION`	Y	Y	Optional: Required for the discovery of databases in Availability Groups
Server	`VIEW SERVER STATE`	Y	Y	Optional: Required for the SnapSync and discovery of Availability Groups
Object: `master.dbo.sqbutility`	`EXECUTE`	Y		Optional: Required when using backups created by Red Gate SQL Backup
Object: `master.dbo.xp_sqllightspeed_version`	`EXECUTE`	Y		Optional: Required when using backups created by Quest LiteSpeed for SQL Server

List of source tables accessed by the Delphix engine

Using the db_datareader permission, the Delphix Engine accesses the following system tables in the master and msdb databases on the source host:

System table	Justification
master.sys.databases	Used to determine the name and recovery model of databases within discover SQL Server instances

System table	Justification
master.sys.availability_groups	Used for discovering all the availability groups within an Availability Group source environment.
master.sys.availability_group_listeners	Used for discovering all the availability group listeners within an Availability Group source environment. • A requirement for dSource linking of SQL Server clustered databases (replicas) is to provide an AG (Availability Group) listener for AG cluster source discovery. This is implemented on AG cluster source discovery as a failsafe if AG cluster source database authentication configuration change down the line, ensuring the Delphix engine has a way to reach the cluster and continue certain operations.
master.sys.availability_databases_cluster	Used for discovering all the availability group clusters within an Availability Group source environment.
master.sys.availability_replicas	Used for discovering all the availability group replicas within an Availability Group source environment.
master.sys.database_files	Used to determine the size of databases and whether filestream is enabled for a database
master.sys.dm_exec_requests	Used to enable Delphix to report backup operation progress
master.sys.master_files	Used to determine the primary file of a database

System table	Justification
master.sys.filegroups	Used to determine the filegroups of a database so that Delphix can configure VDBs with the same filegroups
msdb.dbo.backupset	Used to determine new backups that have been taken. This table is regularly queried to find out if a new backup image has been taken and needs to be synchronized with Delphix.
msdb.dbo.backupmediafamily	Used to determine the physical device names of the backup files comprising a backup.

SQL server staging hosts and databases

Staging host requirements

Windows servers which will be added as Staging Environments must meet the following requirements:

Staging Host Requirement	Explanation
The service account running each SQL Server instance (the Instance Owner) must be one of: • A domain user (e.g. `MYDOMAIN\accountname`) (RECOMMENDED) • A Managed Service Account or Group Managed Service Account (`MYDOMAIN\accountname$`)(requires Windows 2012 and later, and SQL Server 2008R2 or later) • The LOCAL SYSTEM account (`NT AUTHORITY\SYSTEM`) • The NETWORK SERVICE account (`NT AUTHORITY\NETWORK SERVICE`)	Access to existing database backups for Snapshots and Validated Sync operations will fail if the service account cannot access backups created by the Source database instance.
The source host, proxy and staging environments must have appropriate cross-domain trust relationships	For more information on these requirements, see the document Delphix in Multi-domain Windows Environments.

Staging Host Requirement	Explanation
The edition of the installed SQL Server instance(s) must support all database features used by linked Source databases	SQL Server may raise an error during some dSource operations if the Staging SQL Server Instance does not support features used by the Source Database. This is most easily addressed by using the same edition of SQL Server as the Source database. Features in use on the Source database can be checked using the `sys.dm_db_persisted_sku_features` dynamic view.
Must have both the Source and Staging Windows Users configured as local Windows users	See the sections *Source Windows User Requirements* and *Staging Windows User Requirements* for more detail.
Delphix Connector software is installed and running	See Installing the Delphix Connector Service on the Target Database Servers for instructions on installing the Delphix Connector.
Recommended iSCSI Registry settings must be in place	See Requirements for Windows iSCSI Configuration and Knowledge Base Article KBA1251 for instructions on applying these settings*. *Please note that in 6.0.x, these performance settings are subject to change based on further tuning efforts by Delphix Engineering.
sqlcmd command line utility must be installed on the servers hosting VDBs and staging databases. The utility should be in the Delphix operating system user's PATH environment variable.	Delphix does not run sqlcmd (or any other process) directly on the source SQL Server instance. It runs sqlcmd on the staging/connector host.

Staging Windows user requirements

The Delphix Engine needs a Windows domain user — for example, `MYDOMAIN\delphix_os` — to be specified when adding Staging environments to the Delphix Engine. This user must have the following permissions:

Staging User Requirement	Explanation
Be a member of the Windows "Local Administrators" group on the Staging Host	We require this permission for mounting iSCSI LUNs presented by the Delphix Engine to the staging and target hosts. Microsoft utilities used by the Delphix Engine, such as diskpart, require membership of this group.
Have the "Log on as batch" privilege on the Staging host	We require this permission for remote PowerShell execution. This privilege can be assigned through the Local Security Policy (Local Policies → User Rights Assignment → Log on as batch job).
Be able to access existing backups from the Source Database	If accessing existing native, Red Gate, or LiteSpeed backups, the following permissions are required: • The Staging Windows User and the service account running SQL Server must both have permission to access the database backups via SMB (Windows file sharing) • The Staging Windows User and the service account running SQL Server must both have NTFS Permissions to access the database backups Separate documents describe requirements if Linking a dSource from a NetBackup SQL Server Backup or Linking a dSource from a Commvault SQL Server Backup.
Be able to login to each SQL Server instance that the Delphix Engine will communicate with	These requirements are described in the Staging Database Login Requirements section below.

Staging database login requirements

The Delphix Engine requires a Windows Authentication login to be created for the Staging Windows User on each SQL Server instance that the Delphix Engine will communicate with, with the following permissions:

Object	Privileges Required	Delphix OS User (Windows Login)	Purpose
Server	`CONNECT SQL`	Y	Access to the SQL Server instance
Server	`sysadmin`	Y	The staging and target databases are managed and administered completely by Delphix. Our functionality requires many administrative operations on those databases and requires full access to them. Since database ownership can be changed by customers as part of configuring virtual databases, we require the sysadmin role to continue to administer the databases.

SQL server target hosts and databases

Target host requirements

Windows servers which will be added as Target Environments must meet the following requirements:

Target Host Requirement	Explanation
Delphix Connector software is installed and running	See Installing the Delphix Connector Service on the Target Database Servers for instructions on installing the Delphix Connector.
Recommended iSCSI Registry settings must be in place	See Requirements for Windows iSCSI Configuration and Knowledge Base Article KBA1251 for instructions on applying these settings.
The edition of the installed SQL Server instance(s) must support all database features used by linked Source databases	SQL Server may raise an error during some dSource operations if the Staging SQL Server Instance does not support features used by the Source Database. This is most easily addressed by using the same edition of SQL Server as the Source database. Features in use on the Source database can be checked using the `sys.dm_db_persisted_sku_features` dynamic view.

Target Host Requirement	Explanation
sqlcmd command line utility must be installed on the servers hosting VDBs and staging databases. The utility should be in the Delphix operating system user's PATH environment variable.	Delphix does not run sqlcmd (or any other process) directly on the source SQL Server instance. It runs sqlcmd on the staging/connector host.

Target Windows user requirements

The Delphix Engine needs a Windows domain user — for example, `MYDOMAIN\delphix_os` — to be specified when adding Target environments to the Delphix Engine. This user must have the following permissions:

Target User Requirement	Explanation
Be a member of the Windows "Local Administrators" group on the Target Host	We require this permission for mounting iSCSI LUNs presented by the Delphix Engine to the staging and target hosts. Microsoft utilities used by the Delphix Engine, such as diskpart, require membership of this group.
Have the "Log on as batch" privilege on the Target Host	We require this permission for remote PowerShell execution. This privilege can be assigned through the Local Security Policy (Local Policies → User Rights Assignment → Log on as batch job).
Be able to login to each SQL Server instance that the Delphix Engine will communicate with	These requirements are described in the *Target Database Login Requirements* section below.
Delphix Connector software is installed and running	See Installing the Delphix Connector Service on the Target Database Servers for instructions on installing the Delphix Connector.

Target database login requirements

The Delphix Engine requires a Windows Authentication login to be created for the Target Windows User on each SQL Server instance that the Delphix Engine will communicate with, with the following permissions:

Object	Privileges Required	Delphix OS User (Windows Login)	Purpose
Server	`CONNECT SQL`	Y	Access to the SQL Server instance
Server	`sysadmin`	Y	The staging and target databases are managed and administered completely by Delphix. Our functionality requires many administrative operations on those databases and requires full access to them. Since database ownership can be changed by customers as part of configuring virtual databases, we require the sysadmin role to continue to administer the databases.

Supported Roles for Failover Cluster Instances and Always On Availability Groups

Failover Cluster Instances and Always On Availability groups cannot be used as Staging Environments, and VDBs cannot be provisioned into Availability Groups.

When adding a Failover Cluster Instance or Always On Availability Group, all nodes of the cluster must meet the requirements described in this document.

The following table shows how different SQL Server instance types should be added from the Add Environment screen:

Color	Supported?
Y	Yes
N	No

		Environment Role		
Instance Type	**Added As**	**Source Environment**	**Staging Environment**	**Target Environment**
Standalone Instance	Standalone	Y *	Y	Y
Failover Cluster Instance (FCI)	Standalone	Y *	N	N

		Environment Role		
Failover Cluster Instance (FCI)	Cluster	N	N	Y
Always On Availability Group (AG)	Cluster	Supported	N	N **

* Databases that are participating in Availability Groups will not be discovered during the discovery of a Standalone environment.

** VDBs cannot be provisioned into availability groups. However, SQL Server instances that participate in an Availability Group can also be added as Standalone Instances.

Using a Failover Cluster Instance as both Source and Target

A Failover Cluster Instance added as an environment once (as either a Source or Target environment) cannot be used as both a Source and Target.

If this is required, the environment can be added twice:

- Once as a Standalone Source environment
- Once as a Cluster Target environment

The Standalone Source environment can be used for linking dSources, and the Cluster Target environment is used for provisioning VDBs.

As suggested by the Best Practice note earlier in this article, this is not a recommended configuration. Where possible, SQL Server failover cluster instances that the Delphix Engine will use as a target should not be used to host databases other than Delphix VDBs.

Requirements for failover cluster target environments

The following additional requirements exist for Windows Failover Cluster Instances added as Target Environments, to be used for VDB Provisioning.

- You must first add each node in the Window Failover Cluster individually as a standalone target environment, using a non-clustered address. See Adding a SQL Server Standalone Target Environment.
    - The Delphix Engine may show a warning that it cannot discover the Failover Cluster Instances on each standalone host. This is expected.
- Each clustered SQL Server instance must have at least one clustered disk added to the clustered instance resource group, which can be used for creating mount points to Delphix storage.
    - The clustered drive must have a drive letter assigned to it.
    - The clustered drive must be formatted using the "GUID Partition Table (GPT)" partition style, in order for the Delphix Engine to automatically discover the drive letter as a valid option for the cluster instance. An MBR-formatted disk requires manual verification outside of Delphix that the disk has been correctly added to the MSSQL clustered resource group prior to creating the VDB. When provisioning the VDB, you must manually specify the desired MBR disk, because it will not appear in the Delphix GUI.
    - The clustered drive must be added to the clustered instance resource group as a dependency in the Failover Cluster Manager.
- Each node in the cluster must have the Failover Cluster Module for Windows PowerShell feature installed.

    - While running Windows PowerShell as an administrator, enter this command to test that the module is available: `Import-Module FailoverClusters`
- An additional target environment that can be used as a Connector Environment must exist.
    - This environment must **not** be a node in the cluster.
    - This environment should be part of the same Active Directory domain as the cluster.

Additional requirements for azure SQL server availability groups

Microsoft's tutorial and deployment template for building a *SQL Server AlwaysOn Cluster in Azure* does not include the full range of network connectivity that is available in on-premise deployments.

In addition to the connectivity provided by this template:

- The Staging Server must be able to connect to the **Cluster IP Address** using the RPC / Remote Registry port **TCP 445**; and
- The Availability Group Listener must be configured with a TCP Port, such as the SQL Server default 1433

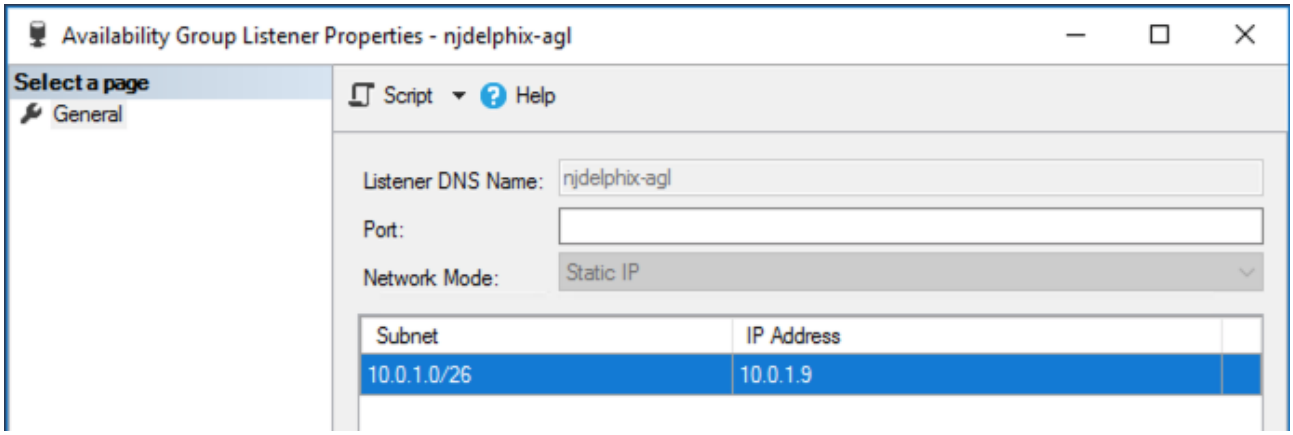
Connectivity to the Cluster IP Address can be tested using the following PowerShell command. This command should be run from the Staging Server, and specify the Cluster Hostname:

> *New-Object System.Net.Sockets.TcpClient("aodns-fc.mydomain.local", "445")*

For Azure clusters, Delphix does not support DNN (Distributed Network Name) at the WSFC level. However, DNN (Distributed Network Name) is supported for creating a SQL AG listener in SQL clustering.

To allow this connectivity on the marketplace template, the following additional steps may be required:

1. On a node of the SQL Server cluster, use PowerShell to define a "probe port". The active cluster node will open this port so that the Azure Load Balancer can detect it.
   Get-ClusterResource "Cluster IP Address" | Set-ClusterParameter -Name "ProbePort" -Value 59998
2. From the Failover Cluster Manager, ensure that the Core Cluster Resource has a valid IP address for its subnet (this may be configured with an invalid IP address such as 169.254.1.1).
3. On all nodes of the SQL Server cluster, ensure that the Windows Firewall allows inbound TCP traffic on TCP port 59998.
4. Extend the Azure Load Balancer (sqlLoadBalancer) configuration to route connections to the Cluster IP Address.
    a. Add a Frontend IP Address for the Cluster IP Address, with an IP address matching that configured for the cluster.
    b. Add a Health Probe for TCP port 59998.
    c. Add a Load Balancing Rule for Port 445, using the newly created Frontend IP Address and Health Probe.More complicated Azure AlwaysOn deployments, such as those including multiple networks, may require additional steps to allow this connectivity.
5. The default Availability Group Listener is not configured with a TCP Port. Current versions of Delphix require this to be configured with a valid port number, such as the default port 1433. This can be changed via the AlwaysOn object tree in SQL Server Management Studio:


Availability Group Listener Properties - njdelphix-agl
Select a page
General
Script
Help
Listener DNS Name:
njdelphix-agl
Port:
Network Mode:
Static IP
Subnet
IP Address
10.0.1.0/26
10.0.1.9

Requirements for Windows iSCSI configuration

Windows iSCSI configuration requirements are split into two types. These requirements are needed on both staging and target servers.

1. iSCSI configuration required for operational stability.
2. Optional iSCSI parameters for performance improvement.

> When target environments are discovered, Delphix will configure the Microsoft iSCSI Initiator Service for Automatic startup.

iSCSI configuration required for operational stability

The following Microsoft iSCSI Initiator configuration parameters are required for the target and staging Hosts. For details about configuring registry settings, see How to modify the Windows registry

> You must reboot the Windows server reboot after changing the iSCSI configuration parameters.

Registry key	Registry value	Type	Data
`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\iSCSI\Discovery`	MaxRequestHoldTime	REG_DWORD	0x384 (900)
`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Disk`	TimeOutValue	REG_DWORD	0x384 (900)
`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Class\{4D36E97B-E325-11CE-BFC1-08002BE10318}\<Instance Number>\Parameters`	MaxRequestHoldTime	REG_DWORD	0x12C (300)

These settings will improve operational stability for VDBs and staging databases. If these settings are not adjusted, SQL Server may raise errors if VDBs are accessed during a temporary infrastructure outage. Affected VDBs may need to be manually restarted using the Continuous Data Engine.

Delphix Knowledge Base article KB1251 includes scripts to validate or set registry parameters so that they meet current Delphix recommendations.

Optional iSCSI parameters for performance improvement

The following iSCSI Registry setting may improve the performance of SQL Server dSource and VDB on the staging and target hosts.

Registry Key	Registry Value	Type	Data
`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\<Interface GUID>`	TcpAckFrequency	REG_DWORD	0x1 (1)

This setting is recommended for storage networks in Microsoft's TechNet article iSCSI and the Nagle algorithm, described in Microsoft's document TcpAckFrequency to control the TCP ACK behaviour

In some environments, adjusting this setting may not improve performance compared to Windows defaults. Modifications to this registry parameter should be tested in each environment, to confirm that this provides a performance improvement.

Delphix engine validation for Windows iSCSI configuration

Delphix Engine validates the Windows iSCSI Configurations that are set on any supported windows staging and target host with the Delphix recommended configurations while performing the following operations:

1. Add environment operation
2. Refresh environment operation
3. Enable environment operation

Prerequisites

1. Supported if you are using Powershell 3.0 or above - If you are on Powershell version below 3.0, then the job will be updated with a warning that the Powershell version on your host is not supported for validating iSCSI parameters.
2. The below alerts are applicable only for staging or target Windows hosts.

Additional information

1. Delphix Engine will only validate and will not alter any configuration in the user environment.
2. On update of registry values on the target host to match Delphix recommendations, the faults from the Delphix engine will only be resolved if any of the operations (environment add, refresh or enable) is performed. Delphix engine will not monitor the state of the target host in the background and hence any change will not be picked up unless an operation is triggered. So, the user needs to take action for the change to reflect in faults.
3. On a successful Delphix Engine upgrade, the latest default iSCSI recommendations will be used for validations.

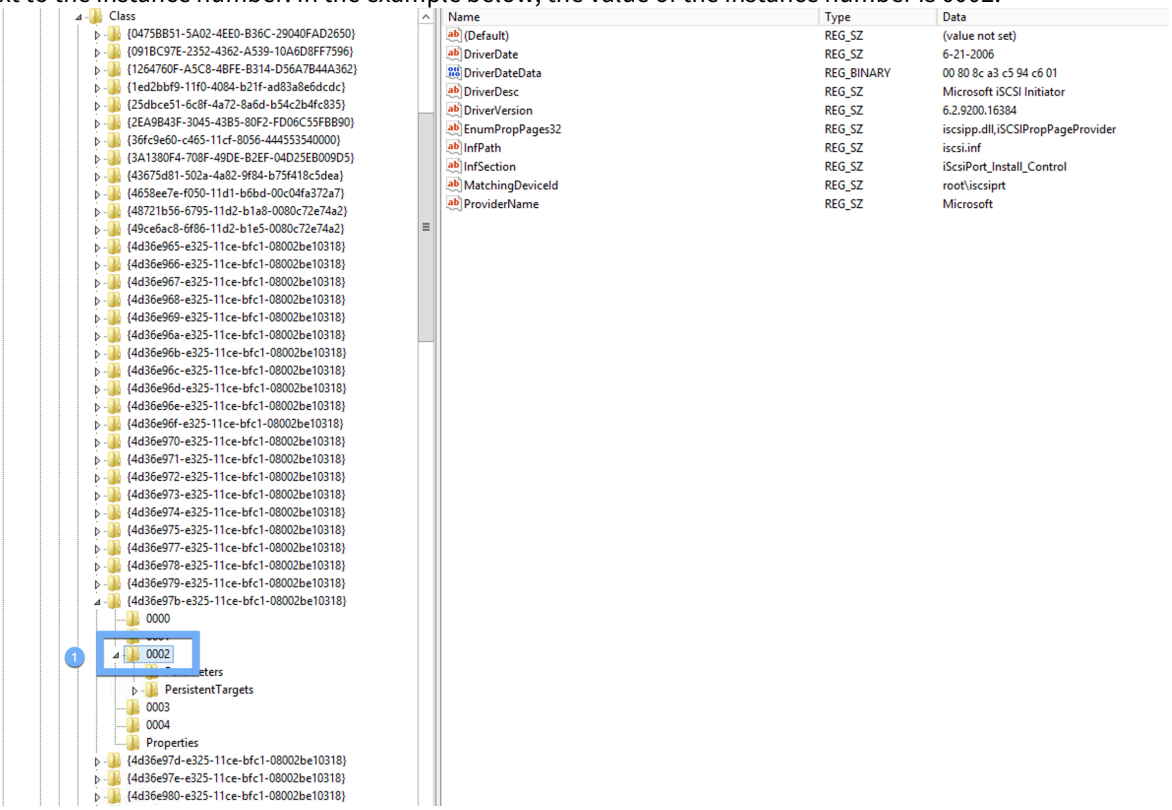
Troubleshooting

Severity	Type	Description
Warning	iSCSI configuration parameter values on environment <env-name> do not match Delphix Engine required values.	This single fault and/or job warning is thrown for all mismatched parameters.
Warning	Failed to fetch some iSCSI configuration parameters for the environment <env-name>.	This single fault and/or job warning is thrown for all parameters where Delphix Engine fails to fetch the value at the target host.

Severity	Type	Description
Warning	Failed to fetch some iSCSI configuration parameters for host <env-name> due to time out.	This job warning is raised, if Delphix Engine is unable to get the iSCSI parameters on the host within 5 minutes. No fault thrown at this point.
Warning	PowerShell version 3 (or above) is required for fetching iSCSI configuration parameters.	This job warning is raised if the PowerShell version is below 3 on the host during the validation of the iSCSI parameters. No fault thrown at this point. The validation is skipped.

Identifying the instance number for iSCSI control class initiator drivers

1. From the Windows toolbar, click **Start** and select **Run** from the menu.
2. Type `regedit` in the **Open** field and click **OK.**
3. Go to the following registry key: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Class\{4D36E97B-E325-11CE-BFC1-08002BE10318}\<Instance Number> where the value of <Instance Number> is the one that shows a **DriverDesc** value of **Microsoft iSCSI Initiator**. Under the registry key, locate and expand the plus (+) sign next to the instance number. In the example below, the value of the instance number is 0002.



1 Value of Instance Number

Windows iSCSI configuration and limits for target and staging hosts

> **Windows Limitation**
>
> Windows supports up to 255 iSCSI LUNs maximum. This creates a hard limit on the number of VDBs that can be created because each VDB requires one or more iSCSI connections.

iSCSI connections - staging

1. dSource linked with Logsync disabled = 1 LUN (DATA)
2. dSource linked with Logsync enabled = 2 LUNs (DATA and ARCHIVE)
3. dSource linked with Logsync disabled and SnapShot started (new COPY ONLY FULL BACKUP) = 3 LUNs (DATA and TEMP)
    a. Once the SnapShot is completed the TEMP LUN will be destroyed and 1 LUN used
4. dSource linked with Logsync enabled and SnapShot started (new COPY ONLY FULL BACKUP) = 3 LUNs (DATA, ARCHIVE and TEMP)
    a. Once the SnapShot is completed the TEMP LUN will be destroyed and 2 LUNs used
5. A maximum of ~120 dSources per Staging Target is recommended, assuming an average of 2 LUNs per source, which would mean 240 LUNS would be consumed for normal operation.
    a. The proposed scenario would leave 13 additional iSCSI connections available for COPY ONLY FULL BACKUPS
    b. For dedicated staging hosts, we do NOT use a Powershell process for monitoring.

iSCSI connections - targets

1. VDB normal operations = 1 LUN (DATA)
    a. No extra mounts required for SnapShot restore or refresh from source Snapsync
2. VDB point-in-time log actions (such as restore, refresh or provision from logs) = 2 LUNS (DATA and SOURCE_ARCHIVE)
    a. An extra LUN is not required for Snapsync operations, only Logsync
    b. Most users do not require enablement of the Logsync feature for MSSQL Sources, because sources in FULL RECOVERY mode create a Snapsync for each log file, providing a significant number of restore points even without retaining the logs.
    c. Once recovery is completed the SOURCE_ARCHIVE LUN will be destroyed and 1 LUN used
3. A maximum of ~120 VDB's per Target is recommended
    a. In 4.x, target host iSCSI connections are less likely to be a limitation while processing costs for Powershell threads may become prohibitive because each target VDB requires a Powershell process for monitoring
    b. In 5.x, this has been alleviated with a hard limit on Powershell processes

iSCSI connections - V2P

1. V2P normal operation = 1 LUN (DATA)
    a. Once the V2P operation is completed the DATA LUN will be destroyed leaving no LUNs used
2. V2P point in time log actions (provision from logs) = 2 LUNS (DATA and SOURCE_ARCHIVE)
    a. Once the V2P operation is completed both the DATA and SOURCE_ARCHIVE LUNs will be destroyed leaving no LUNs used

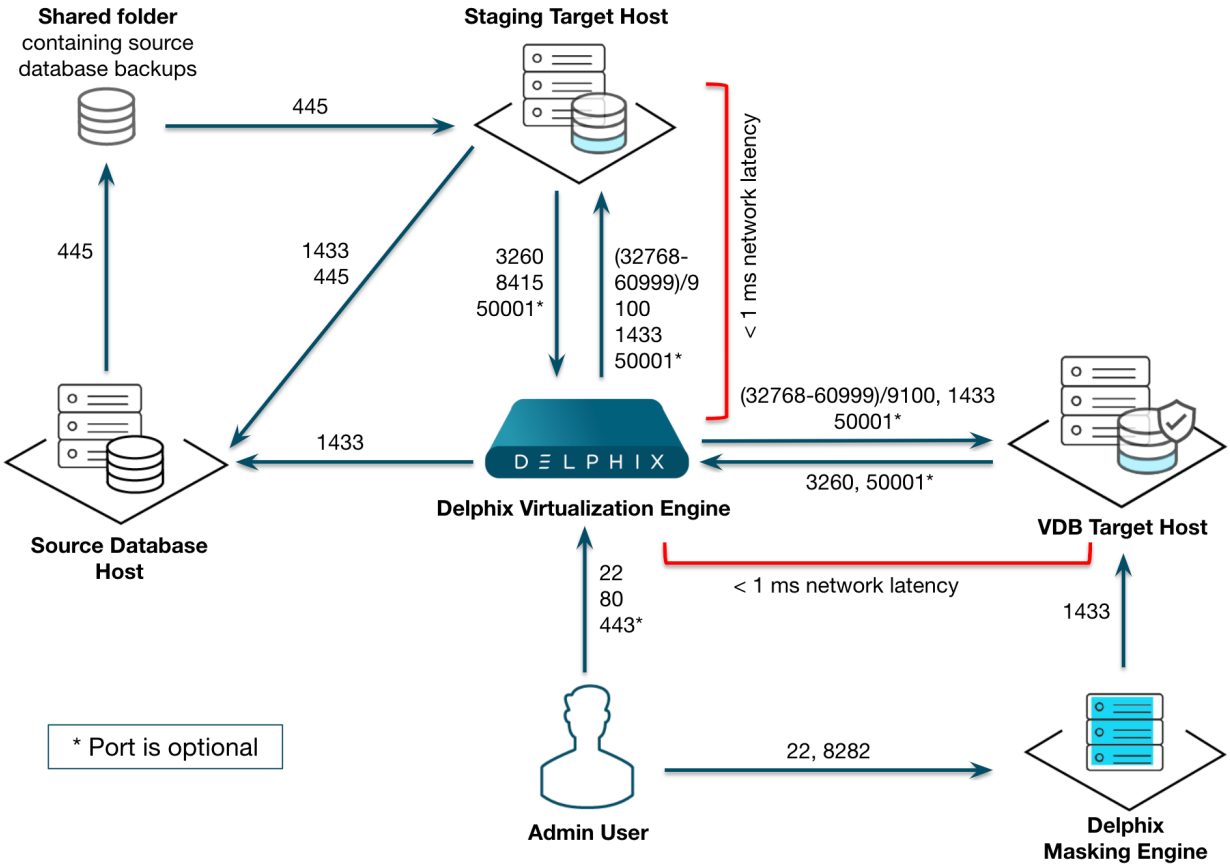
Network access requirements for SQL Server

Network architecture

The diagram *Delphix Virtualizing and Masking SQL Server Network Architecture* below depicts the overall network architecture for Delphix virtualizing and masking SQL Server. In the diagram, each of the arrows represents the direction of a network connection between two nodes. Next to each arrow is a label indicating the network protocol (TCP) and the port number indicating the network service. Also, indicated in red are the recommended network latencies between the major components of the architecture.

> You can optionally configure a separate Connector Environment, specifically used to discover databases on the source during Environment creation. You can also use your Staging Target Host to be used as the Connector Environment, as seen in the image below.

Delphix virtualizing and masking SQL Server network architecture



Delphix Virtualizing and Masking SQL Server Network Architecture

Ports

Based on the table below, the Windows Network Administrator needs to complete a series of tasks. For each port listed, determine whether it must be opened in your firewall between your Delphix Engine and source or target systems. Work with your Delphix Administrator to understand what requirements are there, and ensure that they have been met before proceeding.

Port	Network Service	Required for virtualization?	Required for masking?	Description and usage
22	SSH	Yes	Yes	Used for accessing command-line interface (CLI) and internal Delphix OS accounts
80	HTTP	Yes	No	Used for GUI console access on Delphix Engine by default, disabled when HTTPS in use
443	HTTPS	Yes	No	Used for GUI console access on Delphix Engine, disabled when HTTP in use
445	SMB	Yes	No	Used for attaching shared folders on Windows. To take a copy-only backup or use Delphix Managed backups, this port is required to allow the source environment access to the staging environment.
1433	JDBC	Yes	Yes	Used for accessing SQL Server databases for queries on data-dictionary. This port is the default, but you can use other ports instead. **Note:** Port 1433 is required between the Delphix Engine and AG (Availability Group) cluster sources.
3260	iSCSI	Yes	No	Used for network-attached storage (NAS) on Windows database servers
8415	DSP	Yes	No	Used for SQL Server hooks, when enabled. This port needs to be open between the Delphix Engine and the VDB target.
(32768-60999)/ 9100	Delphix Windows Connector	Yes	No	Used for connecting to the Delphix Connector service installed on Windows target database servers.

Port	Network Service	Required for virtualization?	Required for masking?	Description and usage
50001	iPERF	No	No	Used for network throughput testing with the open-source iPerf package through the Delphix CLI, this is purely optional (but useful) functionality

AppData port requirements

For unstructured files and AppData port requirements, please visit the Network and Connectivity Requirements for Windows Environment article.

Applying network access requirements to Windows cluster configurations

Follow the below points to apply Network Access Requirements to Windows Cluster configurations

- For Target Windows Cluster Environments (running Target Failover Cluster Instances), you need to access TCP 445 of the Windows Cluster Virtual IP from the Staging Server.
- For Source Windows Cluster Environments (running Source Always-On Availability Group databases), you need to access the following:
    - From the Staging Server:
        - TCP 445 of the Windows Cluster Virtual IP
        - The SQL Server port (default TCP 1433) of each SQL Server instance running on the cluster
        - The SQL Server port (default TCP 1433) of all Availability Group Listener Virtual IP addresses that contain Source Databases
    - From the Delphix Engine:
        - The SQL Server port (default TCP 1433) of each SQL Server instance running on the cluster
        - The SQL Server port (default TCP 1433) of all Availability Group Listener Virtual IP addresses that contain Source Databases
- For Source Windows Cluster Environments on Azure, some Azure configuration changes may be required, see Additional requirements for Azure SQL Server Availability Groups for more details.
- For both Source and Target Windows Cluster Environments, the required connectivity for a standalone host should be configured for every node of the cluster. For example, Delphix Engine should be able to connect to the Delphix Connector service installed on each target windows cluster node listening on the Delphix Connector port configured at installation.

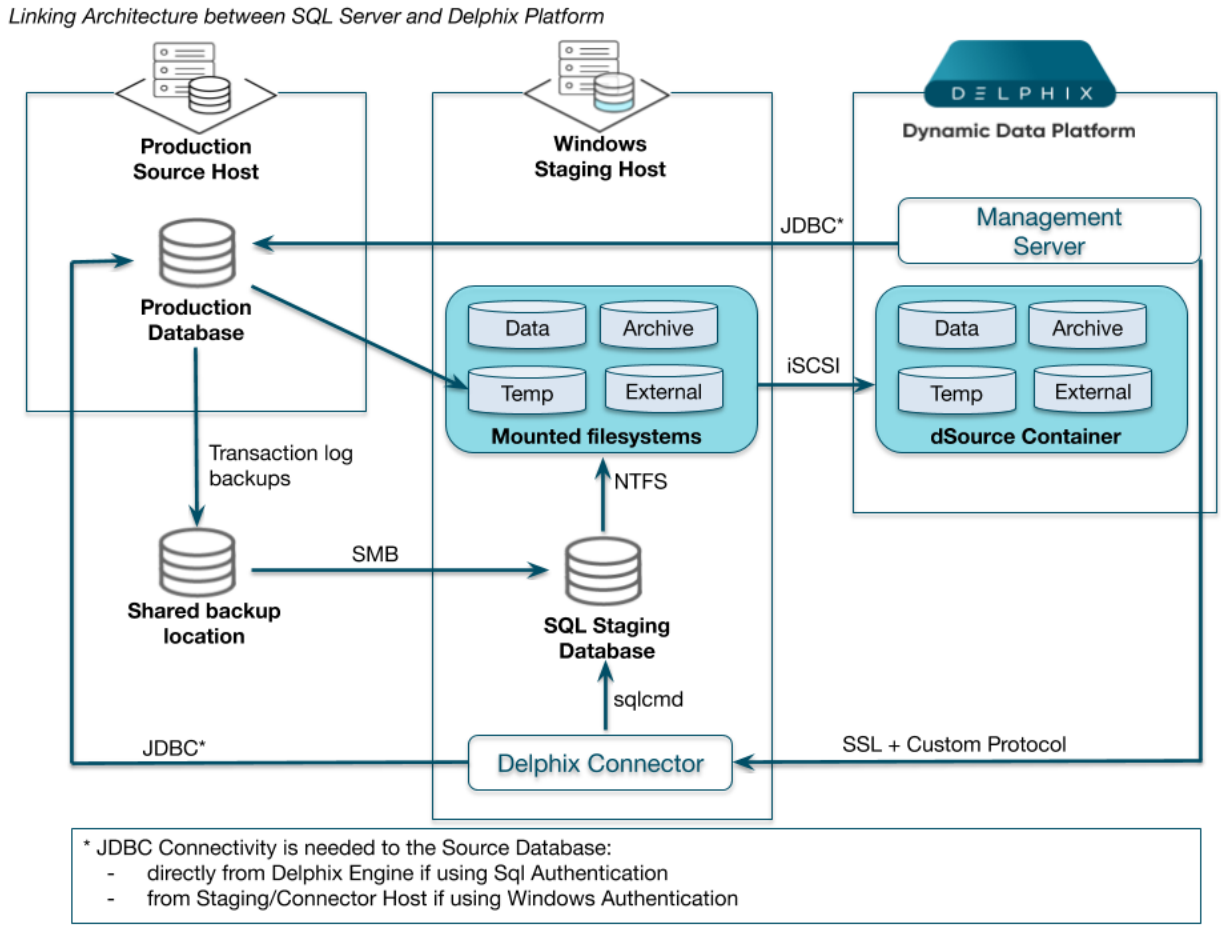
Managing SQL Server environments and hosts

This section describes the attributes of SQL Server environments such as information for the Delphix Connector and covers the following topics:

Overview of setting up SQL Server environments

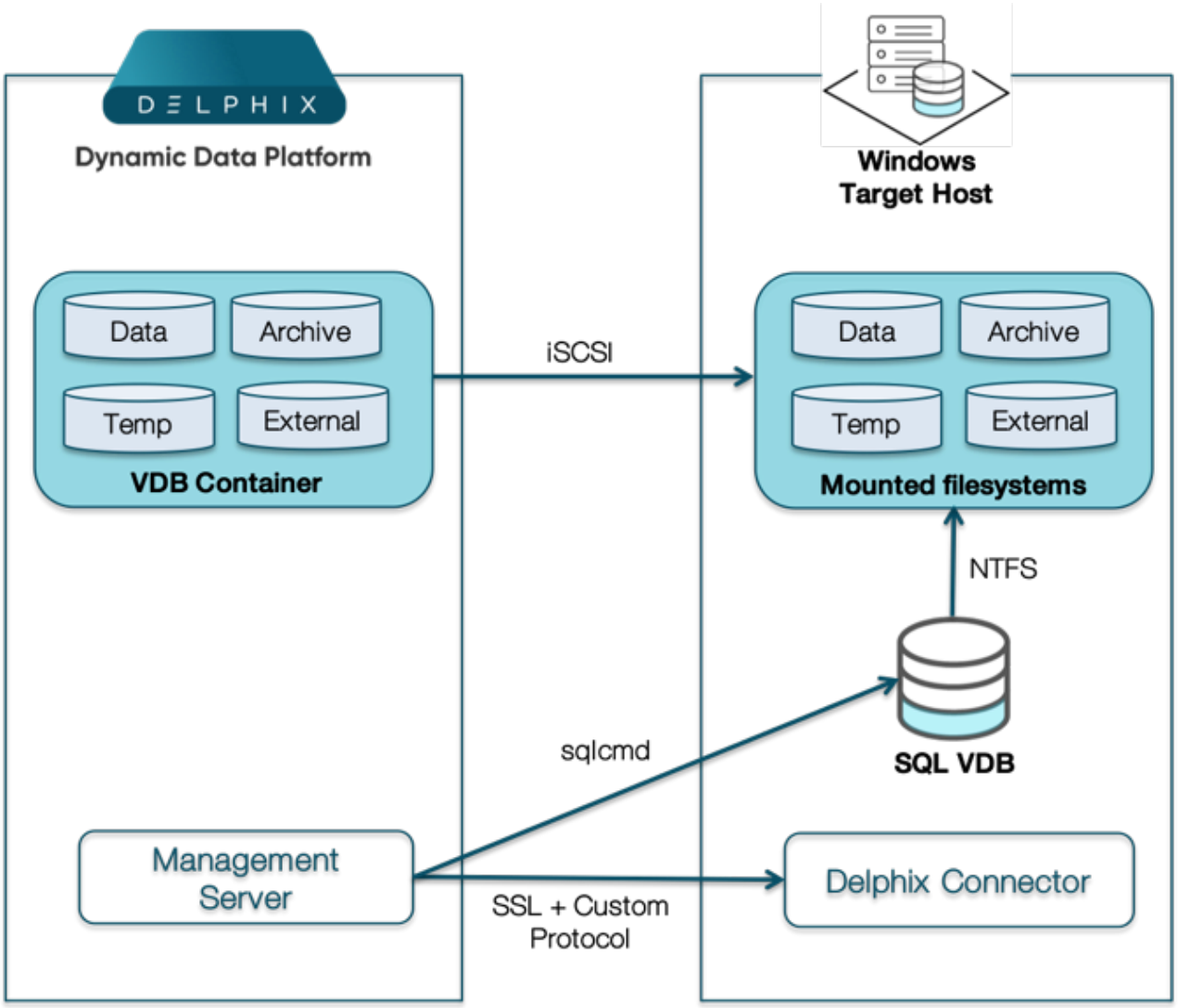
This topic describes the high-level process for adding SQL Server environments, linking SQL Server databases to the Delphix Engine, and provisioning virtual databases.

Block diagram of linking architecture between SQL server environments and the Delphix engine



This block diagram shows the host architecture when creating SQL Server Environments in the Delphix Engine. On the left, the Source Host (in this example a Production Database Server) creates backups to a shared backup location. The Delphix Engine (far right), continuously monitors the source database to determine when new backups are available. When a backup is available, the Delphix Engine will contact the Windows Staging host (center) via the Delphix Connector service. The Staging host will read the backups from the shared backup location and recover them through a Staging Database that is automatically set up on Delphix iSCSI storage. Once recovery is complete, the backup data is incorporated into the Delphix dSource as a new snapshot card, and is available for use.

Block diagram of SQL server provisioning architecture



This block diagram shows how Delphix provisions VDBs to a Target Environment. The Delphix Engine (left) creates a set of virtual files from a snapshot that becomes the VDB container. These files are presented to the Target host (right) via iSCSI. Delphix calls the Delphix Connector on the Target and initiates the creation of a new database using the Virtual files. Once complete, the Virtual Database is brought online and made available for use.

The Delphix connector and environment setup

The Delphix Connector is a Windows service that enables the communication between the Delphix Engine and the Windows target environment where it is installed.

- The directory on which you install the Delphix Connector should have at least 1GB of available space.

This target machine can serve three purposes in a Delphix Engine deployment. It can:

1. Serve as a proxy for database discovery on source hosts.
2. Host a staging database for a linked dSource and run the validated sync process.
3. Host a target environment for provisioning virtual databases (VDBs).

Database discovery is initiated during the environment set up process. When you specify a production source environment that contains the databases you want to manage with the Delphix Engine, you must also specify a

target environment where you have installed the Delphix Connector to act as a proxy for communication with the source environment. This is necessary because Delphix does not require that you install the Delphix Connector software on the production source environment. When you register the source environment with the Delphix Engine, the Delphix Engine uses the Delphix Connector on the proxy environment to discover SQL Server instances and databases on the source. You can then create dSources from the discovered databases. If you later refresh the source environment, the Delphix Engine will execute instance and database re-discovery through the proxy host.

SQL Server dSources are backed by a staging database that runs on a target host. There is no requirement for additional local storage on this target host, as the storage is mounted over iSCSI from the Delphix Engine. At Delphix, we refer to the creation and maintenance of this staging database on the staging host as "validated sync," because it prepares the dSource data on the Delphix Engine for provisioning VDBs later on. After the Delphix Engine creates the staging database, it continuously monitors the source database for new transaction log backups. When it detects a new transaction log backup, it restores that backup to the staging database. The result is a TimeFlow with consistent points from which you can provision a VDB, and a faster provisioning process, because there is no need for any database recovery during provisioning.

When you later provision a VDB, you can specify any environment as a target, including the environment that contains the staging database. However, for best performance, we recommend that you choose a different target environment. The only requirements for the target are:

- It must have the Delphix Connector installed.
- It must have an operating system that is compatible with the one running on the validated host, as described in SQL Server Support and Requirements.

Toolkit size and predicted growth

Each of the clients that run from the client-side toolkit generates its own logs. Each client generates 4 different log files, one for each level of logging: info, trace, debug, and error. Each level of logging is restricted to a maximum of 10 logfiles, and these logfiles are capped at 10MB each. Therefore, the Delphix Engine will consume a maximum of 400MB per client-side application. On the source server, there are currently two commonly run client-side applications, SnapSyncClient and the Delphix Connector.

> ⓘ V2P also generates its own logs, so if you intend to V2P to the source, you should account for an additional 400MB in your upper bound.

Thus, the max amount of growth for the toolkit from logging is 800MB without V2P (or 1.2GB with V2P).

Linking additional dSources does not impact the size of the toolkit on production, aside from the log messages generated during linking, which is included in the calculation above.

On the target server, unlike the source server, there would be only one client – the Delphix Connector, which would occupy around 400 MB maximum storage space. In addition, the Delphix Engine pushes new scripts each time you provision a VDB. This requires < 1MB space.

Therefore, the maximum space occupied by the toolkit directory on the source server is its initial size (~ 400MB) + 800MB = 1.2 GB. While on the target server, the maximum toolkit size is initial size (~ 400MB) + 400 MB + Number of VDBs * 1MB.

Environment attributes for hosts with SQL Server

This topic describes the attributes of SQL Server environments such as information for the Delphix Connector. Below you will see a section for common environment attributes shared by all types of environments as well as SQL Server-specific ones.

Procedure

1. Login to the Delphix Management application.
2. Click Manage.
3. Select Environments.
4. In the Environments panel, click the name of an environment to view its attributes.
5. Next to Attributes, click the Pencil icon to edit an attribute.

Common environment attributes

Attribute	Description
Environment Users	The users for that environment. These are the users who have permission to ssh into an environment or access the environment through the Delphix Connector. For more information on the environment user requirements, see the Requirements topics for specific data platforms.
Host Address	The IP address of the environment host.
DSP KeyStore Path	The path to the user-managed DSP Keystore.
DSP KeyStore Alias	The lowercase alias to use inside the user-managed DSP Keystore.
DSP KeyStore Password	The password for the user-managed DSP Keystore.
DSP TrustStore Path	The path to the user managed DSP truststore.
DSP TrustStore Password	The password for the user managed DSP truststore.
OS	The name of the host operating system.
Version	The version of the host operating system.
Release	The release of the host operating system.
Time Zone	The timezone of the host operating system.
Total RAM	The amount of RAM on the host machine.

Attribute	Description
Processor Type	The processor type of the host machine.
Traceroute	Traceroute info from target host to Delphix Engine.
Notes	Any other information you want to add about the environment.
Java Development Kit	The currently selected JDK kit will be shown.
Java Development Kit (JDK) Path	Location of the Java Development Kit (JDK) used for the host. Only specified if the feature to provide your own JDK is enabled, otherwise, the defaults are used per our Java Support Matrix.

SQL server environment attributes

Attribute	Description
Installed Powershell Version	The PowerShell version installed on the windows target host. This is not applicable for Windows source environments.
Delphix Connector Path	The path for the toolkit that resides on the host.
Delphix Connector Port	The port that the connector connects on.
Delphix Connector Version	The Windows Connector version that is installed on the provided host.
.NET Framework Version	The .NET Framework version used for Windows Connector Service.
Delphix Connector Host	The connector host for a windows environment. This is not applicable for Windows target environments.

Installing the Delphix connector service on the target database servers

This section lists the steps involved in installing a Delphix Connector on your target database server. Installing the Delphix Connector is vital for communication between the Delphix Engine and the targets. A minimum available space of 1GB is a prerequisite to installing the Delphix Connector.

1. From the machine that you want to use as a target, start a browser session and connect to the **Delphix Management** application**.**
2. Click **Manage**.
3. Select **Environments**.
4. Next to **Environments**, click the **Plus** icon.
5. In the **Add Environment** dialog, select **Windows** from the **operating system** menu.
6. Select **Target**.
7. Select **Standalone**.
8. Click **Next**.
9. Click the download link for the **Delphix Connector Installer**.
   The Delphix Connector will download to your local machine.

   Note: You can also download the Delphix Connector directly by navigating to this URL: `http://<name of your Delphix Engine>/connector/DelphixConnectorInstaller.exe`
10. On the Windows machine that you want to want to use as a target, run the Delphix Connector installer. Click **Next** to advance through each of the installation wizard screens.
    Note:
    The installer will only run on 64-bit Windows systems. 32-bit systems are not supported.
    a. For **Connector Configuration**, make sure there is no firewall in your environment blocking traffic to the port on the target environment that the Delphix Connector service will listen to.
    b. For **Java Configuration**, to provide your own Oracle Java enter the absolute path to your Oracle JDK and click **Next**. Otherwise, leave the field blank.
       Note:
       The NETWORK SERVICE user requires read and execute permissions on the Oracle JDK, its subfolders, and files.
    c. For **Select Installation Folder**, either accept the default folder or click **Browse** to select another.
    d. For **Choose .NET Framework (Delphix Engine 6.0.5.0 onwards),** select which version of .NET should be used by the Delphix Connector Windows service. The installer will default to the latest version detected on the system. This selection can be changed afterwards but make sure that chosen .NET framework version should exist in the system.
    e. Click **Next** on the installer final **Confirm Installation** dialog to complete the installation process.
    f. For successful installation a popup will come up stating "**DelphixConnector Installed Successfully**".
    g. Click **Close** to exit the Delphix Connector Install Program.

Relocating the Delphix connector

There are times when the Delphix Connector installation requires a move to a different directory or drive. It's not a trivial relocation. This requires disabling dSources and/or VDBs, uninstalling the current install and reinstalling to the new location. In addition to this, an upgrade to the Delphix Connector can also be achieved via the uninstall/reinstall methodology, including a change in location. These instructions cover 4.0 through 5.1. The steps are the same up to 5.1.2.0. At that version and forward it is no longer required to use the CLI to change the Delphix Connector location. All one needs to do is refresh the Delphix Connector environment and the new directory location is discovered and updated on the Delphix Engine. In the steps listed below, the example is moving the connector from "C:\Program Files" to "C:\", so the full connector path is "C:\Delphix\DelphixConnector".

1. **Disable** the **dSources** and/or **VDBs** associated to the **DelphixConnector** host. This will unmount the storage from the Windows host, removing the directories, the **dSources** represented by the staging databases, and the **VDBs**.
    a. **dSource** staging directories are in the form of "guid-staging-xx", where xx is the staging database number
    b. **VDB** directories are appended with "guid-vdb-xx".
2. When all **VDBs** and **dSources** are disabled and you are ready to move the **DelphixConnector** location, stop the **DelphixConnector service**.
3. Backup the remaining directories as a precaution, in particular, the **logs** directory.
4. Uninstall the **DelphixConnector**, using the instructions from Uninstall the Delphix Connector.
5. Reinstall the **DelphixConnector** to the new location, such as "C:\Delphix\DelphixConnector". Check that the **DelphixConnector** service has started.
6. Modify the new **DelphixConnector**location.
    a. On Delphix engine versions **prior to 5.1.2.0**, use the CLI to modify the directory

        i.
```
de4350.dcenter host> select winhost.delphix.com
de4350.dcenter host 'winhost.delphix.com'> update
de4350.dcenter host 'winhost.delphix.com' update *> set toolkitPath="
C:\Delphix\DelphixConnector"

de4350.dcenter host 'winhost.delphix.com' update *> commit
    Dispatched job JOB-3203
    HOST_UPDATE job started for "winhost.delphix.com".
    HOST_UPDATE job for "winhost.delphix.com" completed
successfully.
```
        ii. Refresh the windows environment for the change to take effect.
    b. On engine versions **5.1.2.0 and higher**, you only need to refresh the windows environment.
7. Enable the **dSources** and/or **VDBs**.

Replacing self-signed certificates on the Delphix connector

The Delphix Connector relies on a Java Keystore with a self-signed X.509 certificate in order to instantiate SSL. If this certificate does not conform to the customer's business standards, it is possible to run a PowerShell script (ReplaceConnectorKeystore.ps1) to replace the self-signed certificate with a certificate that is signed by a Certificate Authority of their choice (i.e. Verisign).

This script should only be used to replace the self-signed certificate in the Delphix Connector's Java Keystore with a signed certificate. Upon execution, the script will do the following:

1. Validate that a PrivateKeyEntry exists within the input keystore
2. Stop the DelphixConnector service
3. Rename the existing DelphixConnector keystore
4. Import the new keystore
5. Start the DelphixConnector service

Prerequisites:

1. The Delphix Connector is installed
2. The DelphixConnector.jks file exists at <Drive>:\<path to DelphixConnector>\connector\DelphixConnector.jks
3. The DelphixConnector.properties file exists at <Drive>:\<path to DelphixConnector>\connector\DelphixConnector.properties and has not been tampered with (STOREPASS, KEYPASS, UUID are present)
4. The Java Keytool utility exists at <Drive>:\ <path to DelphixConnector>\jre\bin\keytool.exe
5. The script, ReplaceConnectorKeystore.ps1 exists at <Drive>:\<path to DelphixConnector>\connector\ReplaceConnectorKeystore.ps1

User inputs:

1. A JKS/PKCS#12 formatted keystore containing a PrivateKeyEntry with a signed certificate
2. The alias of the PrivateKeyEntry in the new keystore
3. The password for the new JKS/PKCS#12 keystore
4. The password for the private key in the new JKS/PKCS#12 keystore

Running the script:

Open up a PowerShell console, and do the following:

1. Navigate to where ReplaceConnectorKeystore.ps1 lives
2. Run .\ReplaceConnectorKeystore.ps1
3. Enter the full path to the new JKS/PKCS#12 keystore
4. Enter the alias of the PrivateKeyEntry in the input keystore
5. Enter the password for the input keystore
6. Enter the password for the private key in the input keystore

How to check if your Java keystore contains a privateKeyEntry:

```
PS C:\Program Files\Delphix\DelphixConnector\jre\bin> .\keytool.exe -list
-keystore ..\..\connector\DelphixConnector.jks
-storepass <STOREPASS from DelphixConnector.properties file>
Keystore type: JKS
Keystore provider: SUN
Your keystore contains 1 entry
delphixconnector-4ef488a8-85df-4418-b56d-1e61b25c0aa2, Jul 28, 2017, PrivateKeyEntry,
```

```
Certificate fingerprint (SHA1): 67:79:DA:E2:64:7A:74:42:62:CA:13:66:29:16:81:0A:B9:7E
:4A:60
```

Example of a successful keystore replacement:

```
PS C:\Users\dtully\Documents> .\ReplaceConnectorKeyStore.ps1
Enter the full path to a JKS/PKCS#12 keystore: C:\Program
Files\Delphix\DelphixConnector\jre\bin\test.jks
Enter alias: leaf
Enter keystore password: ********
Enter private key password: ********
Verifying that a PrivateKeyEntry exists in C:\Program
Files\Delphix\DelphixConnector\jre\bin\test.jks
Stopping the Delphix Connector service
Renaming C:\Program Files\Delphix\DelphixConnector\connector\DelphixConnector.jks to
C:\Program Files\Delphix\DelphixConnector\connector\DelphixConnector.jks.old
Importing the keystore into DelphixConnector.jks
[Storing C:\Program Files\Delphix\DelphixConnector\connector\DelphixConnector.jks]
Starting the Delphix Connector service
```

Uninstalling the Delphix connector service from the target database servers

On occasion, the installed Delphix Connector may become out of date as newer versions of Delphix are released to improve stability or performance.

The following steps can be used to uninstall and reinstall the Delphix Connector as required:

- Uninstall the Delphix Connector
- Download the latest installer from your Delphix Engine
- Reinstall the Delphix Connector

While the Delphix Connector is uninstalled on target/staging environments, dSource and virtual database (VDB) operations (such as validated sync, manual snapshots, and provisioning operations) initiated by the Delphix Engine against those environments will fail.

When working with target/staging environments, or Delphix Engines with a large number of users, to reduce the risk of confusion or failure for other users consider disabling each dSource in the environment. In the case of VDBs, there is no need to disable or shutdown. If SnapSync, refresh or other VDB operations occur during the uninstall/install they can be restarted once the procedure is complete. This prevents the need to incur more downtime.

Uninstalling the Delphix connector

Before uninstalling the Delphix Connector, the following steps should be performed using the Delphix Management application:

1. Login to the Delphix Management application as an admin.

As a precaution, you can backup the binary directories, such as bin, connector, service, scripts, etc, to a temporary location.

2. Copy the logs directory, since uninstall will remove this directory. VDB or Staging folders can not be moved since the data is an iSCSI data mount. These will not be removed or manipulated during uninstall/reinstall.
3. Verify that no VDB provisioning or refresh operations are currently being performed on the target/staging environments.
4. If the environment being upgraded is a staging environment for any dSources, disable all dSources using this environment.
5. Just prior to disabling dSources, use the following query on each SQL Server instance to verify that no operations are currently running.

Show Restore Operations

```
SELECT r.session_id, r.command, CONVERT(NUMERIC(6,2), r.percent_complete) AS
[PercentComplete], CONVERT(VARCHAR(20), DATEADD(ms,r.estimated_completion_time,
GetDate()),20) AS [ETACompletionTime], CONVERT(NUMERIC(10,2), r.total_elapsed_time/
1000.0/60.0) AS [ElapsedMin], CONVERT(NUMERIC(10,2), r.estimated_completion_time/
1000.0/60.0) AS [ETAMin], CONVERT(NUMERIC(10,2), r.estimated_completion_time/1000.0/
60.0/60.0) AS [ETAHours], CONVERT(VARCHAR(1000), (SELECT SUBSTRING(text,
r.statement_start_offset/2, CASE WHEN r.statement_end_offset = -1 THEN 1000 ELSE
(r.statement_end_offset-r.statement_start_offset)/2 END) FROM
sys.dm_exec_sql_text(sql_handle)))
FROM sys.dm_exec_requests r
WHERE command IN ('RESTOREDATABASE','BACKUPDATABASE','RESTORELOG','BACKUPLOG')
```

If your Windows Server is used as a Staging / Validated Sync Target for one or more linked dSources, uninstalling the connector while a Validated Sync operation (i.e. database restore) is in progress may leave the database in an inconsistent state. Validated Sync operations are not visible in the Delphix GUI, so it is necessary to query the SQL Server instance(s) on this server directly to check whether a restore is in progress.

6. If any operations are in progress, wait for them to complete before disabling dSources. Once complete, disable dSources and proceed with the uninstall.

On the Target/Staging Environment (where the Delphix Connector is being upgraded):

1. From the Windows Services manager, Stop the **Delphix Connector Service**.
2. Via the **Task Manager** verify that the java process associated with the **Delphix Connector** has terminated.
3. From **Control Panel** select **Programs and Features**, then navigate to **Delphix Connector**, right-click it and select **uninstall**.
4. Install the new connector from the downloaded **DelphixConnectorInstaller.msi** or **DelphixConnectorInstaller.exe** file. For more information refer to Installing the Delphix Connector Service on the Target Database Servers.
5. Check the **Delphix Connector** version from the **Programs and Features** list. For more information on Connector versions please refer to How To Determine Connector Version.
6. Check the **Delphix Connector** service is running and set to automatic.
7. Verify your system is now working properly. You may need to manually start any disabled dSources or VDBs from the Delphix UI. For more information refer to Managing Data Sources and Syncing Data.

Troubleshooting Delphix connector

Troubleshooting removing the Delphix connector

If there are issues removing the original Delphix Connector please refer to these steps:

- You can remove the DelphixConnector from Control Panel > Program and Features.

Troubleshooting Delphix connector install

If you experience the following issues when re-installing the Delphix Connector refer to, Reinstall/Upgrade the Delphix Connector.

- Delphix Connector service might disappear from the service console.
- Installation failing with unidentified service error.
- The Delphix connector upgrade failed.
- Manually deleted the Delphix Connector folder to try reinstalling.

Troubleshooting if the Delphix connector service fails to start

- Check if the Delphix Connector service is available and running on the Staging Host. If it is not running, then start the service.
- If the Delphix Connector service automatically shuts down or has restarting problems, kill the running java process started by Delphix connector and then retry starting the Delphix Connector Service.
- Open a command prompt (cmd) as administrator.
- From the prompt run: netstat -noba | findstr "9100 java". If the connector is still running, the output should be similar to this:

```
C:\Users\username>netstat -noba | findstr "9100 java"
 [java.exe]
  TCP    0.0.0.0:9100           0.0.0.0:0              LISTENING       1928
 [java.exe]
  TCP    [::]:9100              [::]:0                 LISTENING       1928
 [java.exe]
```

  The final number on each line is the Process ID (PID) of the Delphix Connector process.
- If there is a java process bound to port 9100 as shown above, but the Delphix Connector still fails to start, you can kill the Java process using the taskkill command:

```
C:\Users\username>taskkill /PID 1928 /F
SUCCESS: The process with PID 1928 has been terminated.
```

  Alternatively, you can use Task Manager or Process Explorer to identify and end the correct process/PID and end it from there.
- Start the Delphix Connector service.

If the service still fails to start after performing the above steps, please attempt to reinstall the Delphix Connector. If issues persist, contact Delphix Support for assistance.

Troubleshooting if installation of third party software results in Delphix connect service to fail to start

- If third party software is installed on the Windows host running Delphix Connector and upon the reboot, Delphix Connector doesn't startup, please uninstall the Software and revert the host back to the state it was in before the software deployment and then check if the Delphix Connector service starts

- If it doesn't start check the steps in the section "The Delphix Connector service fails to start".
- If the service still doesn't start please open a Support case with Delphix for further investigation.

Upgrading the Delphix connector

The installed Delphix Connector may become outdated as newer versions of Delphix are released to improve stability or performance.

Upgrading the Delphix connector is a two-step process, which includes

- Uninstalling the Delphix connector
- Installing the Delphix connector

While the Delphix Connector is uninstalled on target/staging environments, dSource and virtual database (VDB) operations (such as validated sync, manual snapshots, and provisioning operations) initiated by the Delphix Engine against those environments will fail.

When working with target or staging environments or Delphix Engine with a large number of users to reduce the risk of confusion or failure for other users consider disabling each dSource in the environment. In the case of VDBs, there is no need to disable or shut down. If SnapSync, refresh or other VDB operations occur during the uninstall/ install they can be restarted once the procedure is complete. This prevents the need to incur more downtime.

If you are using the certificate that is signed by a certificate authority on the Delphix connector, you can use either of the following ways:

1. After upgrading the Delphix connector, re-execute the steps of Replacing Self-signed Certificates on the Delphix Connector.
2. If you want to use the existing certificates, make sure to follow the below steps:

- Before uninstalling, you must back up `DelphixConnector.jks` file and properties such as `Keystore, Keypass, Storepass` and `UUID` from `DelphixConnector.properties` file to a temporary location.
- Uninstall the Delphix connector.
- Install the Delphix connector.
- Stop the Delphix connector service.
- You can now copy the backed-up .jks file and properties.
- Paste the backed-up .jks file inside the installation directory and the properties to the `DelphixConnector.properties` file.
- Ensure that the `Keystore, Keypass, Storepass and UUID` properties are kept unchanged, and `Keystore` property must point to the correct installation directory.

  Note : For example, if the Keystore is `C\:\\Program Files\\Delphix\\Delphix Connector\\connector\\DelphixConnector.jks` , make sure to place this .jks file is in the same path as mentioned in Keystore.

If you are using windows connector host authentication, you can use either of the following ways:

1. After upgrading the Delphix connector, re-execute the steps for windows connector host authentication.
2. If you want to use the existing configuration of windows connector host authentication, make sure to follow the below steps:

- Before uninstalling, you must back up `DelphixConnector.jks` file and properties such as `Keystore, Keypass, Storepass` and `UUID` from `DelphixConnector.properties` file to a temporary location.
- Uninstall the Delphix connector and then
- Install the Delphix connector.
- Stop the Delphix connector service.

- You can now copy the backed-up .jks file and properties.
- Paste the backed-up .jks file inside the installation directory and the properties to the `DelphixConnector.properties` file.
- Ensure that the `Keystore, Keypass, Storepass and UUID` properties are kept unchanged, and `Keystore` property must point to the correct installation directory.

For example, if the Keystore is `C\:\\Program Files\\Delphix\\Delphix Connector\\connector\\DelphixConnector.jks`, make sure to place this .jks file is in the same path as mentioned in Keystore.

Manual discovery for SQL Server instances

Occasionally, SQL Server instances cannot be automatically discovered due to a problem in the Windows registry or if the instance is not running at that time. These SQL Server instances can be discovered using manual discovery.

Manually adding a SQL server instance

To manually add a SQL Server instance

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments** and select the **Databases** tab.
4. Click the **Add Instance** button and enter the following details:
    a. To allow linking and/or provisioning, select **Enabled**.
    b. If this environment will be used as a staging environment, select **Enabled**.
    c. Selecting **Enabled** for Fulltext Installed allows users to do full-text searches.
    d. Add as a **Failover Cluster Source**: When this is Enabled users are asked to enter the Server name. **Note:** This option is only applicable when adding a SQL Server failover cluster instance as a source, it is not available for clusters as a target.
    e. **Name**: Not currently used.
    f. **Port**: Used for IPC connection to the instance.
    g. **Installation path and instance owner**: Installation path of SQL Server. Not currently used.
    h. **Internal version**: The internal version of the SQL Server. This is important during linking and provisioning as the compatible instance is identified based on the internal version.
    i. **Version**: the SQL Server Version required for adding a source instance. **Note:** If the wrong version is entered, Delphix may fail to discover the databases and hence manual discovery may fail.
5. Click **Add**. At this time, if all values are correct, an instance will be created and all its databases will be auto-discovered internally.

> If Delphix Engine is not able to establish a connection using given details, appropriate errors will be displayed and the user will be asked to make the required changes.

Editing a SQL server instance

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. In the **Environments** panel, select the **Databases** tab.
5. Click the **Pencil** icon to edit an instance.
   If you are editing an instance which you added, you will be able to edit: - You will be able to edit all fields except the name.
6. Click **Save**.

Refresh a manually added SQL server instance

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. In the **Environments** panel, click the **refresh** icon on top.
5. Along with auto-discovered instances, it will now also refresh manually added instances and add/remove its databases, based on the current state of the instance.

If Delphix Engine is not able to establish a connection to the instance, then the appropriate warning will be displayed to the user during database discovery.

Deleting a manually added SQL server instance

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. In the **Environments** panel, select the **Databases** tab.
5. You will see a delete icon at the far right of all manually added instances.
6. Click **Delete**. This will delete the instance and all its databases.

This operation will fail if the chosen instance has databases that have dependencies such as database being used to provision VDBs etc.

Adding a SQL Server source environment

This topic describes how to add a SQL Server source environment.

Prerequisites

- You must have already set up SQL Server target environments, as described in Adding a SQL Server Standalone Target Environment
    - You will need to specify a target environment that will act as a proxy for running SQL Server instance and database discovery on the source, as explained in Overview of Setting Up SQL Server Environments
- Make sure your source environment meets the requirements

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Next to **Environments**, click the **Actions** menu and select **Add Environment**.
5. In the **Add Environment**wizard, Host and Server tab select:
    a. Host OS: **Windows**
    b. Host Type: **Source**
    c. Server Type:
        - If you are adding a Windows Server Failover Cluster (WSFC), add the environment based on which WSFC feature the source databases use:
            - Failover Cluster Instances Add the environment as a **standalone** source using the **cluster name** or **address**.
            - AlwaysOn Availability Groups Add the environment as a **cluster** source using the **cluster name** or **address**.
        - Otherwise, add the environment as a **standalone** source.
6. Click **Next**.
7. In the Environment Settings tab select a **Connector Environment**.
   Connector environments are used as proxy for running discovery on the source. If no connector environments are available for selection, you will need to set them up as described in Adding a SQL Server Standalone Target Environment. Connector environments must:
    - have the Delphix Connector installed
    - be registered with the Delphix Engine from the host machine where they are located.
8. Enter the **Environment Name**, **Node Address**, **OS Username**, and **OS Password** for the source environment.
9. Click **Submit.**

As the new environment is added, you will see multiple jobs running in the Delphix Admin Job History to Create and Discover an environment. In addition, if you are adding a cluster environment, you will see jobs to Create and Discover each node in the cluster and their corresponding hosts. When the jobs are complete, you will see the new environment added to the list in the **Environments** panel. If you don't see it, click the **Actions** menu and select **Refresh All**.

Adding a SQL Server standalone target environment

This topic describes how to add a SQL Server standalone target environment to the Delphix Engine.

You can use SQL Server targets for three purposes in a Delphix Engine deployment. They can:

- Host a target environment for the provisioning of Virtual Databases (VDBs).
- Host a staging database for a linked dSource and run the validated sync process.
- Serve as a proxy host for database discovery on source hosts.

Regardless of the specific purpose, all Windows targets must have the Delphix Connector installed to enable communication between the host and the Delphix Engine. The instructions in this topic cover initiating the Add Target process in the Delphix Engine interface, running the Delphix Connector installer on the target machine, and then verifying that the target has been added in the Delphix Engine interface.

When target environments are discovered, Delphix will configure the Microsoft iSCSI Initiator Service for Automatic startup.

Prerequisites

- Make sure that your target environment meets the requirements described in the following sections:
  - SQL Server Support and Requirements
  - Requirements for Windows iSCSI Configuration
  - Receive Side Scaling (RSS) for Windows Staging Target and Targets
- The Directory on which you install the Delphix Connector should have at least 1GB of available space.

If you have taken a snapshot on Windows 2012 or an earlier version, the provisioning, linking, or exporting to Windows 2022 will result in disk errors. Windows event logs list these errors. You can ignore these errors or run the `CHKDSK \F` command.

Procedure

1. From the machine that you want to use as a target, start a browser session and connect to the **Delphix Management** application**.**
2. Click **Manage**.
3. Select **Environments**.
4. Next to **Environments**, click the **Actions** menu and select **Add Environment**.
5. In the **Add Environment**wizard Host and Server tab, select:
   a. Host OS: **Windows**
   b. Host Type: **Target**.
   c. Server Type: **Standalone**.
6. Click **Next**.
7. In the Environment Settings tab click the download link for the **Delphix Connector Installer**. The Delphix Connector will download it to your local machine.
8. On the Windows machine that you want to want to use as a target, run the Delphix Connector installer. Click **Next**to advance through each of the installation wizard screens.
   The installer will only run on 64-bit Windows systems. 32-bit systems are not supported.
   a. For **Connector Configuration**, make sure there is no firewall in your environment blocking traffic to the port on the target environment that the Delphix Connector service will listen to.
   b. For **Select Installation Folder**, either accept the default folder or click **Browse** to select another.
   c. Click **Next** on the installer final 'Confirm Installation' dialog to complete the installation process and then **Close** to exit the Delphix Connector Install Program.

9. Return to the Delphix Management application.
10. Enter the **Environment Name**, **Host Address**, **Delphix Connector Port**, **OS Username**, and **OS Password** for the target environment.
11. To provide your own Oracle Java select the **Provide my own JDK** checkbox and click **Next**.
12. In the Java Development Kit tab enter the absolute path to your Oracle JDK and click **Next**.
13. Click **Submit.**
    As the new environment is added, you will see two jobs running in the **Delphix Admin Job History**, one to **Create and Discover** an environment, and another to **Create** an environment. When the jobs are complete, you will see the new environment added to the list in the **Environments** panel.

Post-requisites

1. On the target machine, in the **Windows Start Menu**, click **Services**.
2. Select **Extended Services**.
3. Ensure that the **Delphix Connector** service has a **Status** of **Started**.
4. Ensure that the **Startup Type** is **Automatic**.

Adding a SQL Server failover cluster target environment

This topic describes how to add a SQL Server failover cluster target environment to the Delphix Engine.

Adding a failover cluster target environment will discover SQL Server failover cluster instances that are running. You can then provision virtual databases (VDBs) to these failover cluster instances.

Prerequisites

- You must add each node in the Window Failover Cluster individually as a standalone target environment using a non-cluster address. See Adding a SQL Server Standalone Target Environment.
  - For a cluster node added as a standalone environment, the Delphix Engine will only discover **non-clustered** SQL Server instances.
  - For a cluster target environment, the Delphix Engine will only discover SQL Server **failover cluster** instances.
- **Each clustered SQL Server instance must have at least one clustered disk added to the clustered instance resource group, which can be used for creating mount points to Delphix storag**e.
  - The clustered drive must have a drive letter assigned to it.
  - The clustered drive must be formatted using the "GUID Partition Table (GPT)" partition style in order for the Delphix Engine to automatically discover the drive letter as a valid option for the cluster instance. An MBR-formatted disk requires manual verification outside of Delphix that the disk has been correctly added to the MSSQL clustered resource group prior to creating the VDB. When provisioning the VDB, you must manually specify the desired MBR disk, because it will not appear in the Delphix GUI.
  - The clustered drive must be added to the clustered instance resource group as a dependency in the Failover Cluster Manager.
- Each node in the cluster must have the **Failover Cluster Module for Windows Powershell**feature installed.
  - While running Windows PowerShell as an administrator, enter this command to load the module:
    ```
    import-module failoverclusters
    ```
- An additional target environment that can be used as a **Connector Environment** must exist. This environment must NOT be a node in the cluster. See Adding a SQL Server Standalone Target Environment.

Hotfix required for Windows 2008 R2 hosts
The following hotfix is required for Windows 2008 R2 Cluster nodes:
"0x80070490 Element Not found" error when you enumerate a cluster disk resource by using the WMI MSCluster_Disk class query in a Windows Server 2008 R2-based failover cluster
http://support.microsoft.com/kb/2720218

Windows Cluster Volume Management Software Requirements Only cluster volumes managed by the native Windows Volume Manager are supported. For example, cluster volumes managed by Veritas VxVM are not supported.
If you use third-party volume management software, create a new LU (recommended to be 10GB in size) and add this LU as a clustered resource to the SQL Server instance using native Windows volume management tools. Assign a drive letter for this LU. You can then use this LU as the volume mount point location for Delphix VDB provisioning.

Best Practices
SQL Server failover cluster instances that the Delphix Engine will use should not be used to host databases other than Delphix VDBs.

Cluster environment restrictions

You cannot use failover cluster target environments as staging environments.

However, for VDB **target hosts**, it is important to use the **same edition** of SQL Server software as the **source database**, so that all features available in the source are also available in the VDB.

Supported roles for each instance type

Failover Cluster Instances, and instances supporting Always On Availability Groups, support a subset of the operations available to Standalone SQL Server instances.

Color	Supported?
Y	Yes
N	No

		Environment Role		
Instance Type	**Added As**	**Source Environment**	**Staging Environment**	**Target Environment**
Standalone Instance	Standalone	Y	Y	Y
Failover Cluster Instance (FCI)	Standalone	Y	N	N
Failover Cluster Instance (FCI)	Cluster	N	N	Y
Always On Availability Group (AG)	Cluster	Y	N	N

* Databases that are participating in Availability Groups will not be discovered during the discovery of a Standalone environment.

VDBs cannot be provisioned into availability groups. However, SQL Server instances that participate in an Availability Group can also be added as Standalone Instances.

Using a Failover Cluster Instance as both Source and Target
A Failover Cluster Instance added as an environment once (as either a Source or Target environment) cannot be used as both a Source and Target.
If this is required, the environment can be added twice:

- Once as a Standalone Source environment

- Once as a Cluster Target environment

The Standalone Source environment can be used for linking dSources, and the Cluster Target environment is used for provisioning VDBs.
As suggested by the Best Practice note earlier in this article, this is not a recommended configuration. Where possible, SQL Server failover cluster instances that the Delphix Engine will use as a target should not be used to host databases other than Delphix VDBs.

⚠ If you have taken a snapshot on Windows 2012 or an earlier version, the provisioning, linking, or exporting to Windows 2022 Failover Cluster Target might result in disk errors. In such a case, contact Delphix Customer Support

⚠ Error
The following error may be encountered during the Windows 2022 AG and FCI clusters creation process on VMware ESXi.

```
An error occurred while executing the test.Unable to connect to <hostname.domainname.com> via WMI. This may be due to networking issues or firewall configuration href="http://hostname.domainname.com">hostname.domainname.com>.Invalid namespace
```

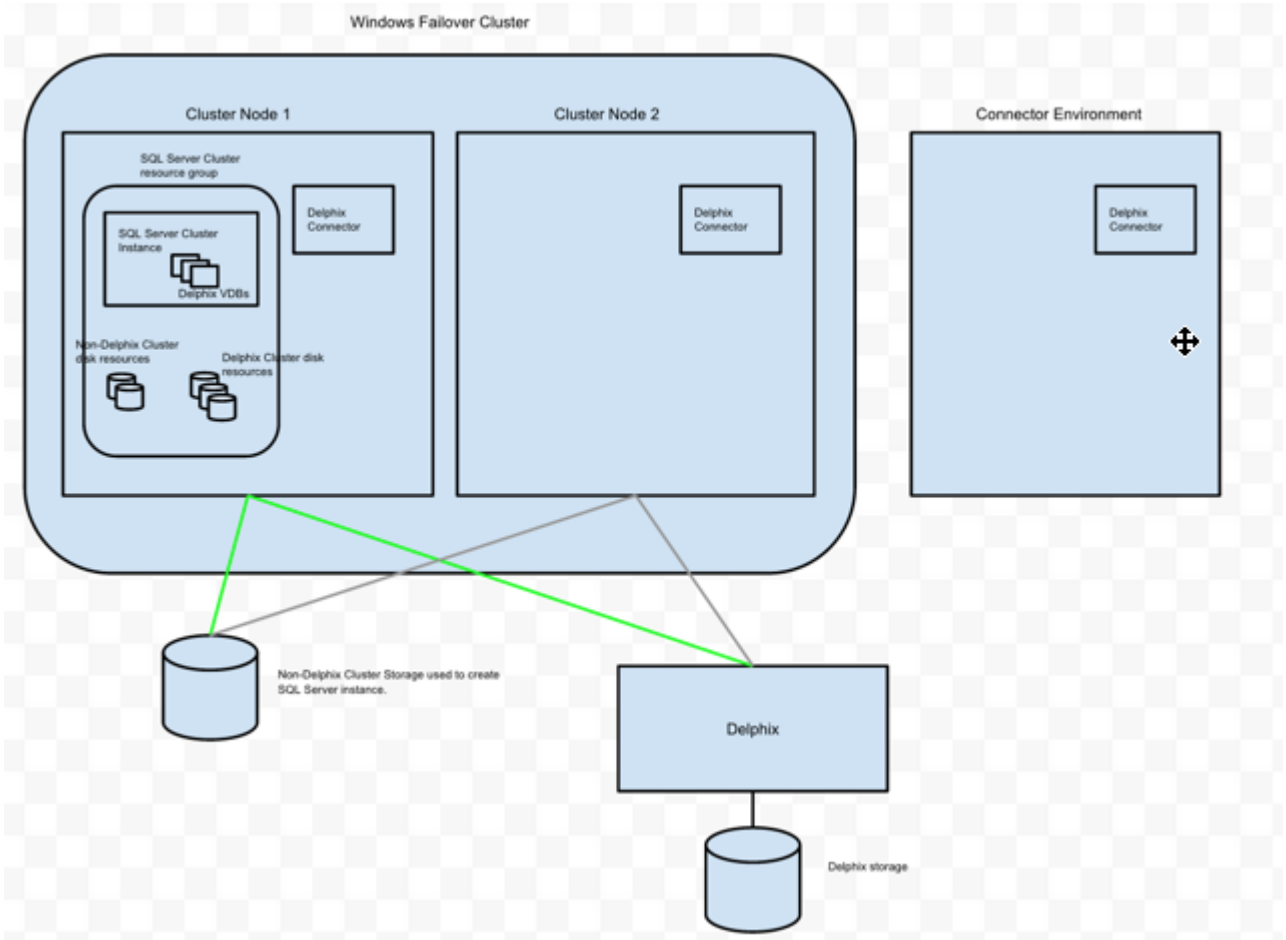
This failure is due to `Validate switch enabled Teaming configuration` validation. The workaround is to skip this validation.

Procedure

1. Click **Manage**.
2. Select **Environments**.
3. Next to **Environments**, click the **Plus** icon and select **Add Environment**.
4. In the **Add Environment**wizard, Host and Server tab select:
   a. Host OS: **Windows**
   b. Host Type: **Target**.
   c. Server Type: **Cluster**.
5. Click **Next**.
6. In the Environment Settings tab specify the environment Name and cluster address of the Windows Failover Cluster.
7. Select a host that is NOT a node in the cluster as the **Connector Environment**.
8. Enter the **OS Username** and **OS Password** for the target environment.
9. Click **Validate Credentials**.
10. Click **Submit** to confirm the target environment addition request.

In the Delphix Engine interface, you will see a new icon for the target environment, and two jobs running in the **Delphix Admin Job History**: one to **Create and Discover** an environment, and another to **Create** an environment. When the jobs are complete, click the icon for the new environment, and you will see the details for the environment.

Example environment



In this example environment, the Delphix Connector was installed on **Connector Environment**, **Cluster Node 1**, and **Cluster Node 2**. Each host was added to Delphix as standalone target environments.

Next, the **Windows Failover Cluster** was added as a Windows Target Cluster environment using the cluster address. **Cluster Node 1** is currently the active node for the SQL Server Failover Cluster resource group. Delphix has exported iSCSI LUs and has created the corresponding Cluster Disk resources for each VDB.

Additional SQL Server environment topics

This section covers the following topics:

- Microsoft SQL Server bypass powerShell execution policy
- Delphix as a backup solution to SQL Server
- Delphix managed backups on secondary nodes of SQL Server alwaysOn availability group cluster
- Receive side scaling for windows staging target and targets
- Validated sync environment
- Changing the hostname (IP address) or IQN of a SQL server target or staging host

Microsoft SQL Server bypass powerShell execution policy

Hooks are executed with “PowerShell -ExecutionPolicy RemoteSigned” whereas user-configured pre/post sync and pre/post provision PowerShell scripts are executed with “PowerShell -ExecutionPolicy Bypass”. If this PowerShell configuration exceeds the security privileges of business standards, a Group Policy Object can be configured in Windows to manage PowerShell execution policy. If configured, we require that the script execution policy is set to “RemoteSigned”. The primary benefit is that the Group Policy configuration takes precedence and overrides the execution policy set in PowerShell, even if you run PowerShell as an Administrator. Alternatively, using MSSQL hooks instead of pre/post scripts will naturally execute with a “RemoteSigned” execution policy.

Refer to the Microsoft Documentation on Execution Policies for more information: https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_execution_policies?view=powershell-6

The Group Policy is per machine and will apply to PowerShell 2.0 and later.

Delphix as a backup solution to SQL Server

Using Delphix as a backup solution to SQL Server

Delphix provides you the option to automatically manage backups from SQL Server source databases into the Delphix Engine. Prior to Delphix 4.2, users could not link source databases that were backed up by unsupported backup software. In Delphix 4.2, a feature known as Delphix Managed Backups was introduced that allows you to have Delphix take and manage backups from your source database directly into Delphix storage. This is the first step in Delphix becoming a full-featured backup solution for SQL Server databases. When the Delphix Engine manages the backups for a dSource, it takes regular, copy-only full backups of the source database, so activating the feature will not interfere with existing backup management solutions. You can configure the schedule of when the Delphix Engine takes these copy-only full backups by specifying a SnapSync policy for the dSource. You can change the SnapSync policy for a dSource at any time by visiting the policy screen; there, you can either select a new SnapSync policy or modify the current one.

If you use a backup solution that is not supported by Delphix, you cannot use your existing backups to keep your dSources in sync. However, enabling Delphix Managed Backups will overcome this issue by using automatic copy-only full backups to keep dSources in sync. Currently, dSources linked when this feature is enabled will not support LogSync functionality, which means that you can only provision VDBs from snapshots and not from any time between snapshots. Additionally, in the current release, the Delphix Engine cannot take differential or transaction log backups of the source database.

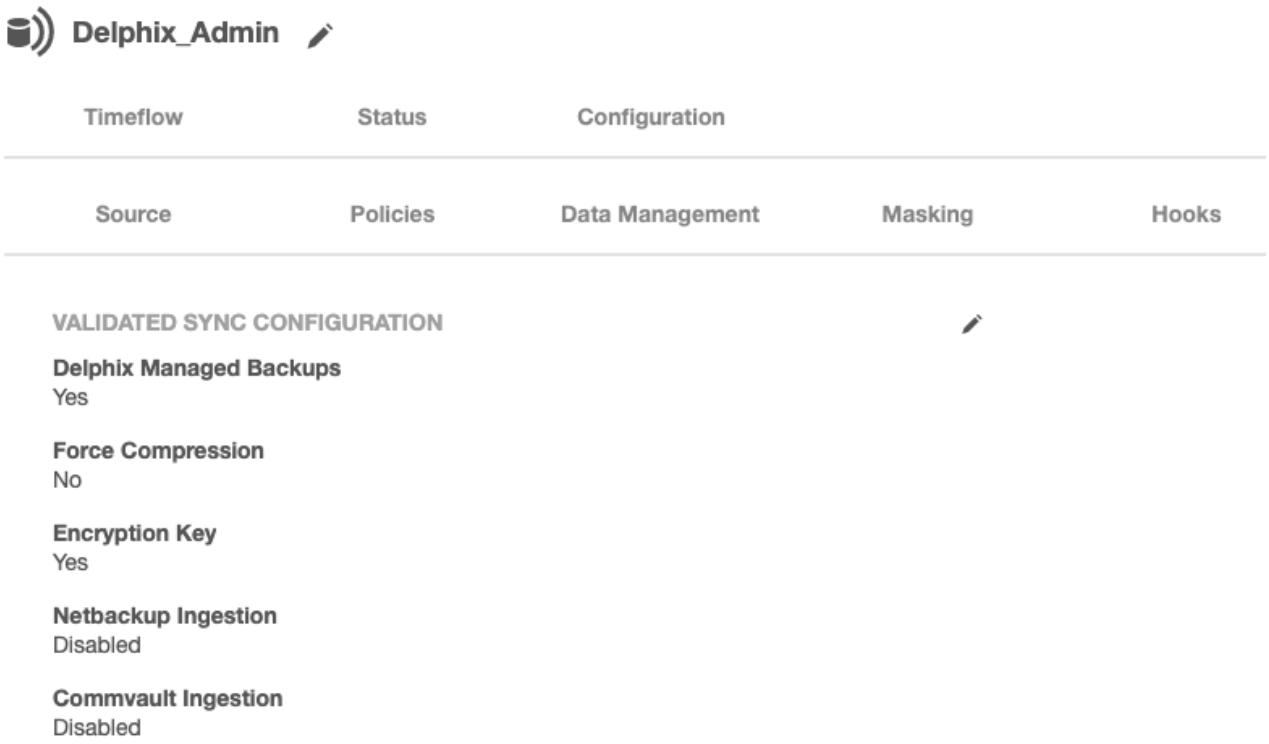
Linking SQL Server dSources with Delphix managed backups

The **Data Management** page of the link wizard for SQL Server dSources provides the option to enable Delphix Managed Backups.

It possible to enable this feature here at link time or toggle it on after the link for future syncs. If you enable this feature, the dSource can only use copy-only full backups taken by the Delphix Engine to stay in sync with its source; the Delphix Engine will prohibit syncing using existing backups. Checking the **Enabled** box results in the following changes to the **Data Management** page:

- The initial load option is set to a copy-only full backup taken by the Delphix Engine
- The ability to provide a backup path disappears
- A SnapSync policy selection screen appears in the **Policies** page

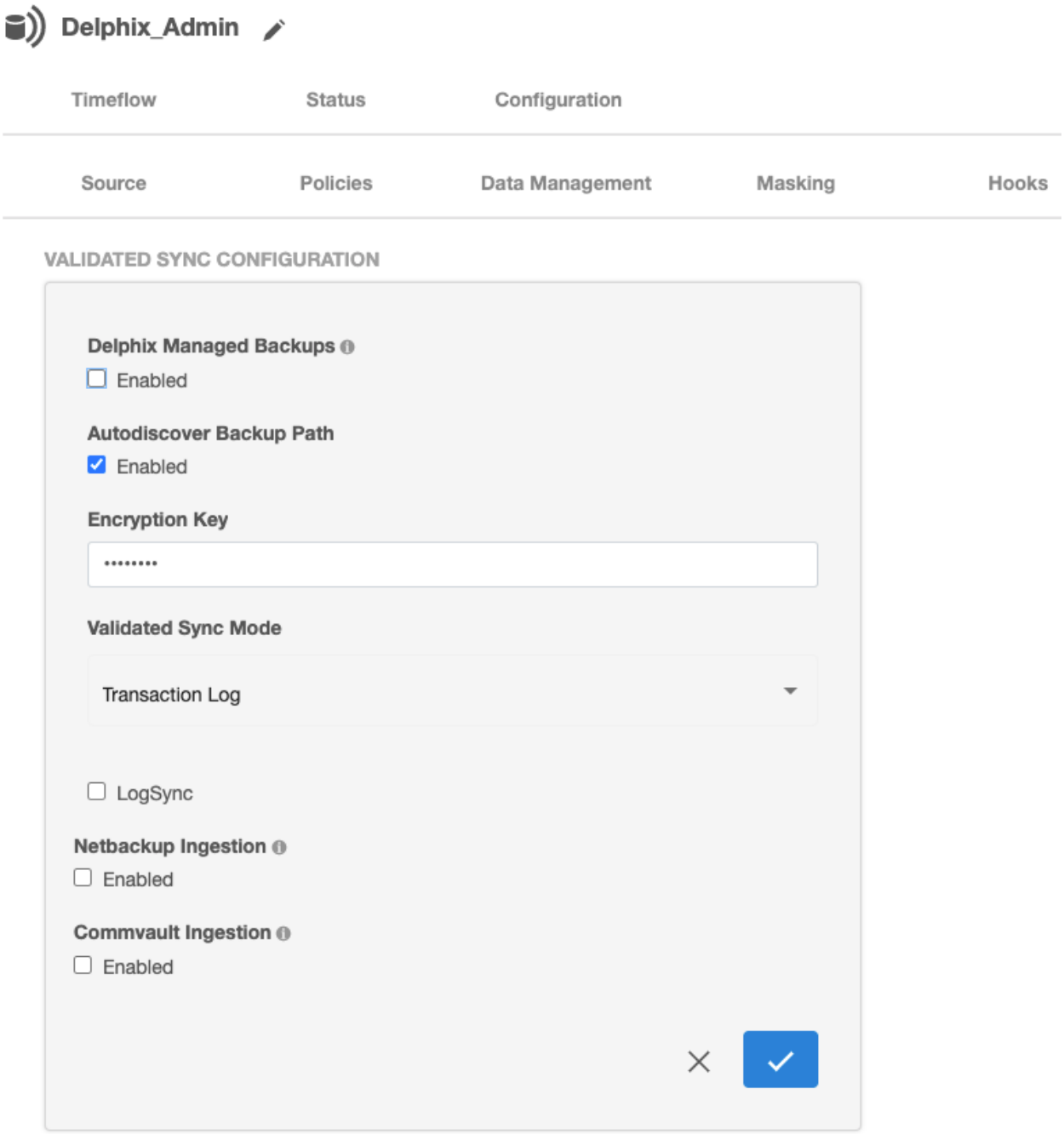
You can select from the list of existing SnapSync policies if the one you want doesn't exist you will need to create a new one on the **Policies** page under the **Manage** dropdown. Proceeding through the remainder of the link wizard will create a dSource with Delphix-managed backups enabled. You can confirm that a dSource has the feature by selecting the dSource and going to the **Configuration** > **Data Management** tab after creation and checking the **Delphix Managed Backups** section, as displayed below:



To disable/enable this feature after linking:

Toggling Delphix Managed backups is supported for SQL Server dSources. This means a dSource that was previously created with Delphix Managed Enabled can have regular validated sync and similarly a dSource that had external backups can use Delphix Managed Backups.

To disable Delphix Managed Backups for a specific dSource, select that specific dataset and go to **Data Management** tab under the **Configuration** tab.



Click the edit button and uncheck this feature. Then update all new inputs.

To enable this feature just go to the same edit menu.

Backup compression feature overview

Since Delphix Engine 5.2, the Delphix Engine has allowed compression to be enabled ("forced") for SQL Server backups which use Delphix Managed Backups. Delphix Managed Backups are used to synchronize SQL Server dSources when existing SQL Server backup files cannot be made available to the Staging server, or if a third-party backup vendor is used that is not yet supported by Delphix. For more information on this functionality see Delphix as a Backup Solution to SQL Server and Linking a dSource with SQL Server.

Backup compression is preferable in the following situations when:

- The default backup compression setting for the Source SQL Server instance is Disabled (0)
- The Source database does not use Transparent Data Encryption (TDE)
- The Source server has available CPU resources to perform compression

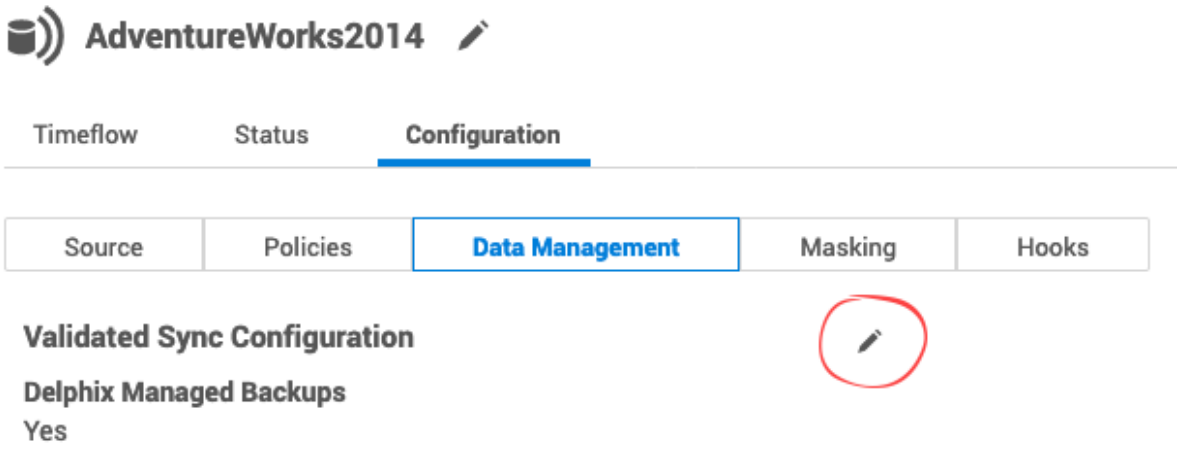
Where the compression ratio of a compressed backup exceeds 3:1, this will reduce the amount of data (by half) that must be transferred over the network to perform a backup, resulting in much faster SnapSync operations.

How to enable backup compression

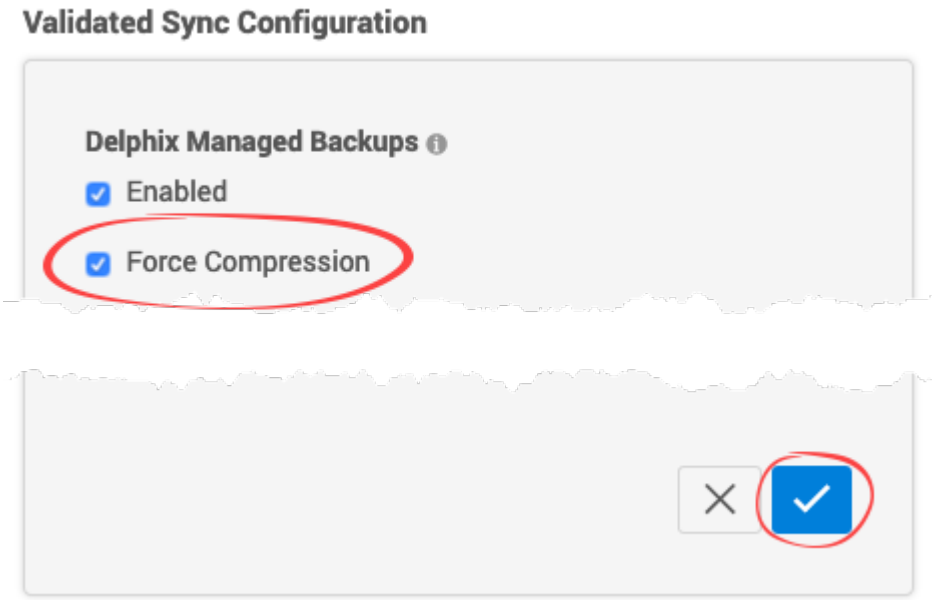
If the SQL Server instance's default backup compression setting is enabled (see Microsoft's document View or Configure the backup compression default Server Configuration Option), no specific action is required. Backups will automatically be compressed in accordance with this setting.

If backups are not automatically compressed, the Delphix Engine (5.2 and later) can be configured to force compressed backups using the following steps:

- Login to the Delphix Engine's Management interface.
- Open the **Manage > Datasets** screen.
- Locate and select the dSource (using the tree on the left).
- Navigate to the **Configuration > Data Management** pane.
- Click the **Edit** (pencil) icon to modify the Validated Sync settings.



- Select the **Force Compression** checkbox, then the select the Tick icon to save the changes



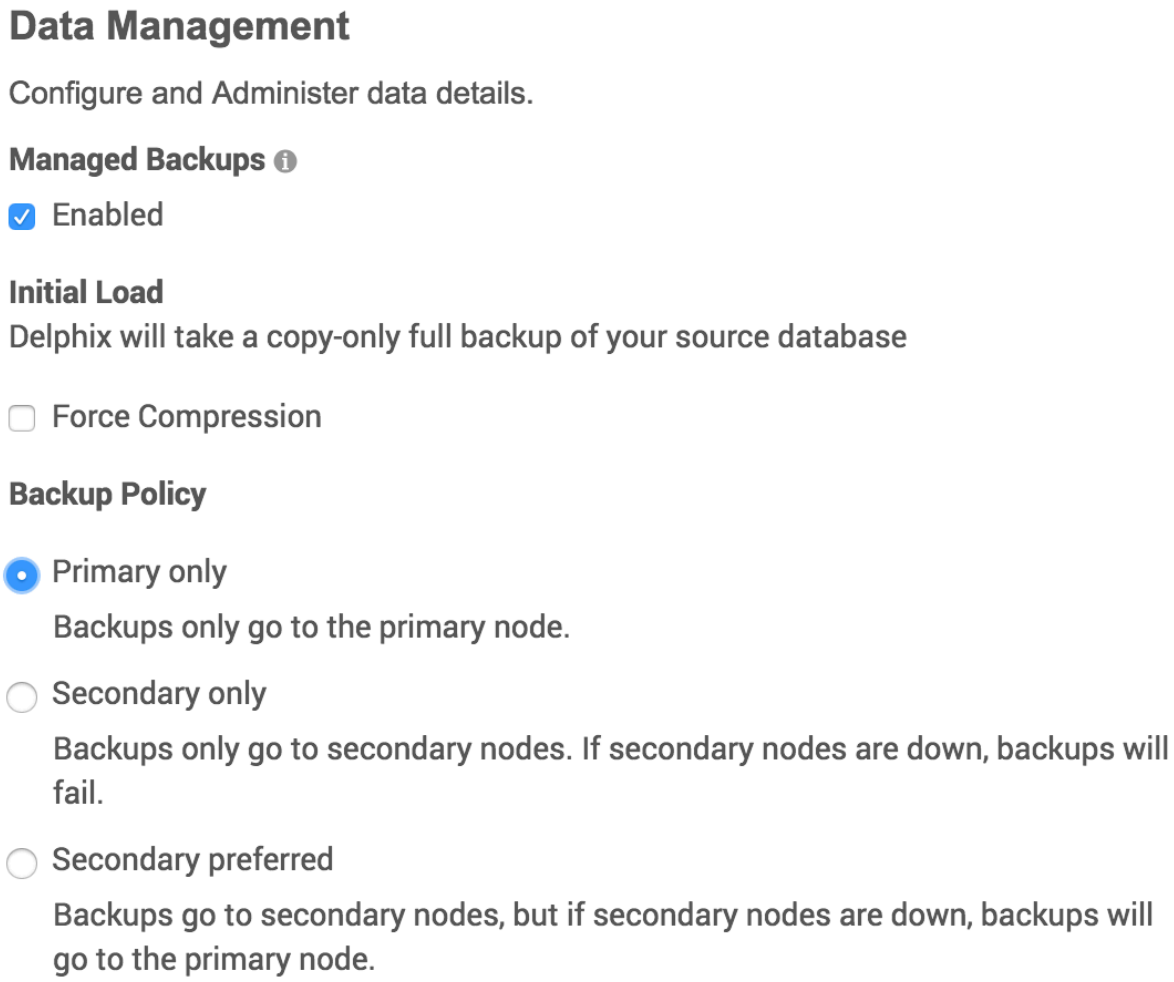
The change will automatically take effect during the next scheduled Snapshot/SnapSync operation.

Delphix managed backups on secondary nodes of alwaysOn availability group cluster

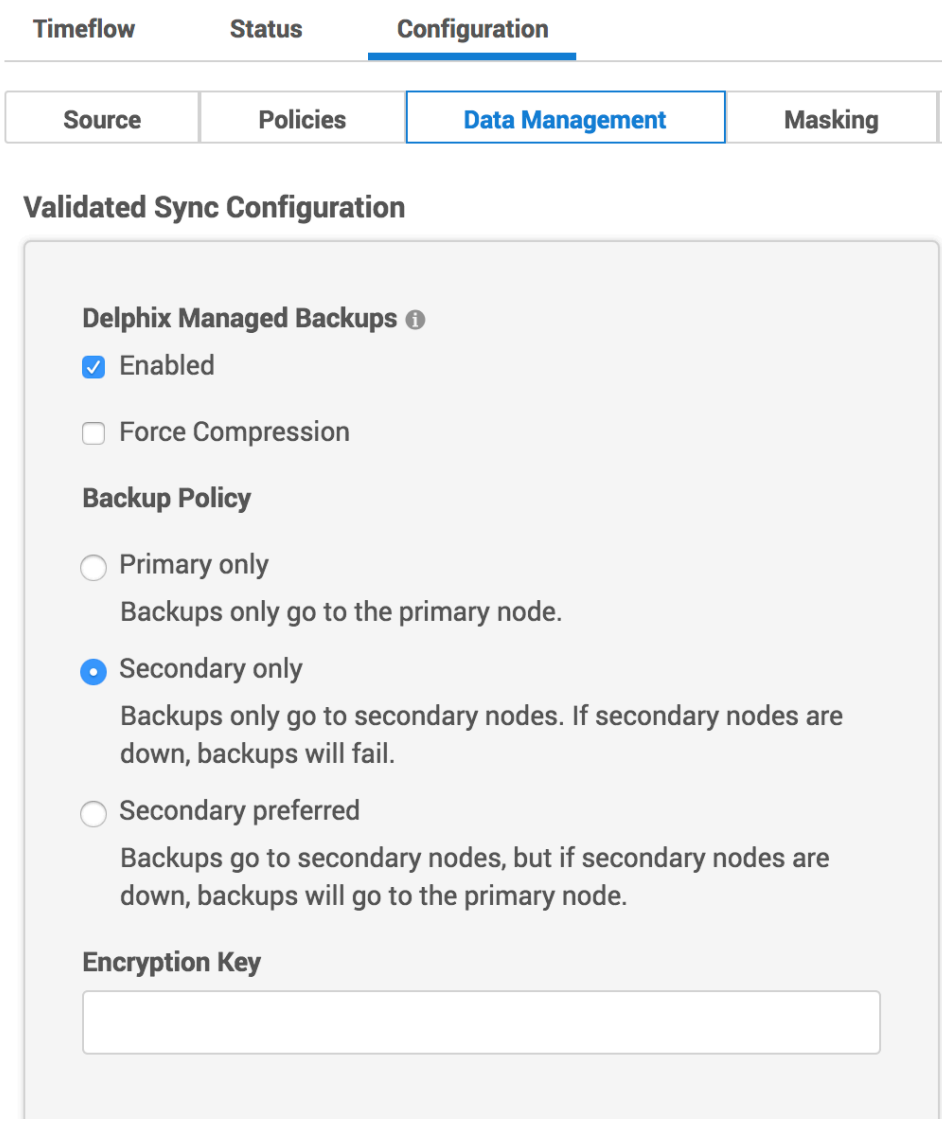
The Always On availability groups feature is a high-availability and disaster-recovery solution that provides an enterprise-level alternative to database mirroring. Introduced in SQL Server 2012 (11.x), Always On availability groups maximizes the availability of a set of user databases for an enterprise. An *availability group* supports a failover environment for a discrete set of user databases, known as *availability databases*, that failover together. An availability group supports a set of read-write primary databases and one to eight sets of corresponding secondary databases. Optionally, secondary databases can be made available for read-only access and/or some backup operations. Some customers prefer using secondary nodes for backup to reduce the load on the primary node while some customers are OK with using the primary node. To provide more flexibility for customers using Always On availability groups with Delphix Managed backups, delphix provides an option to choose backup policy during linking of a dSource.

They can pick one of the following options that describe their backup policy as shown below.

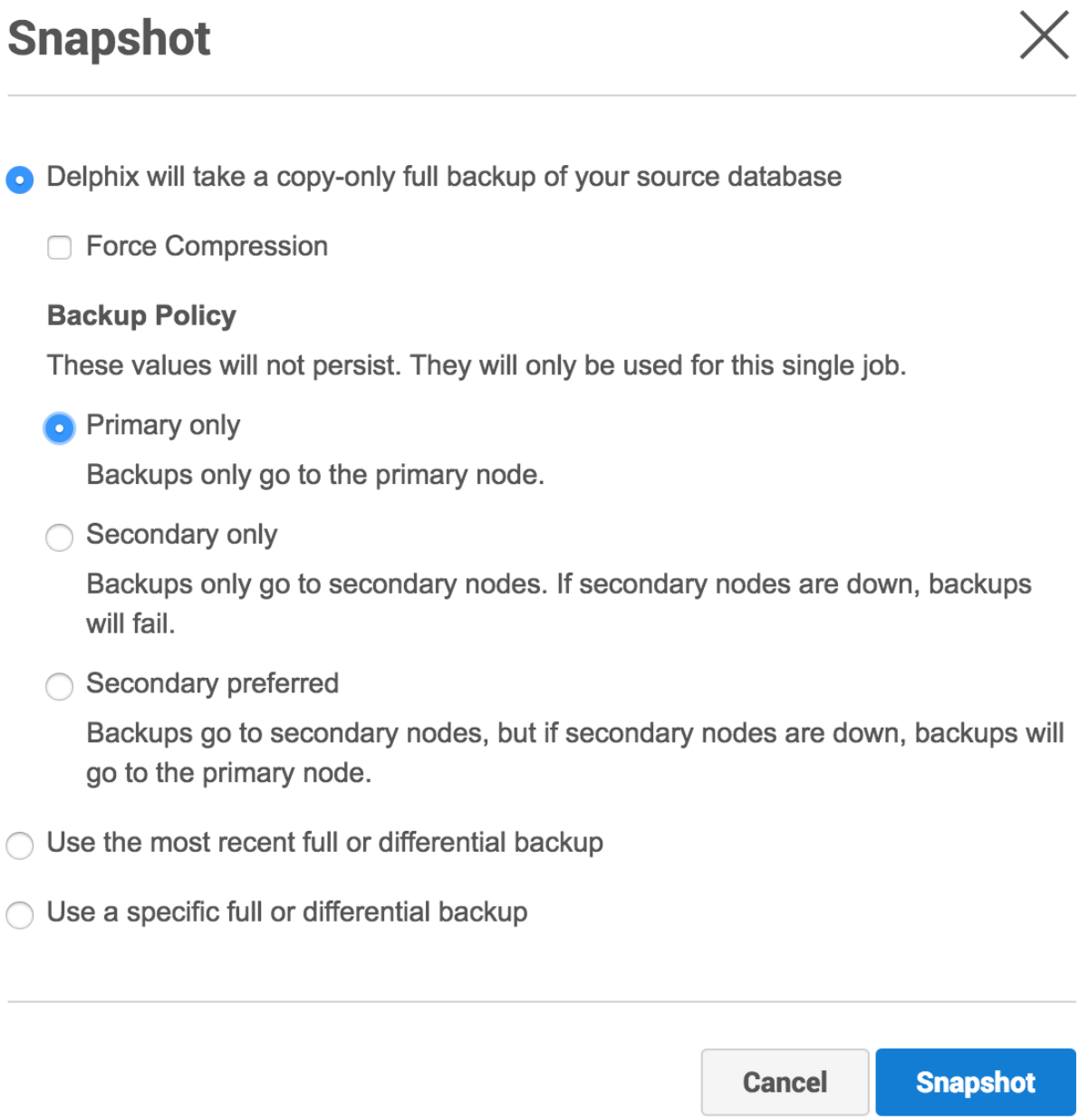
- Primary (**default option**): Backups are taken only on the primary node.
- Secondary Only: Customers who never want to use primarily for their backups must choose this option as it ensures backups never go to the primary node. If the secondary nodes are down, the backups fail but do not use primary at all.
- Secondary preferred: As the name suggests, one of the secondary nodes is used for backup. The backup will be taken on primary if none of the secondary nodes can be used for backup. So, customers who want reliable backups even if they are taken on the primary node if required should use this option.



The backup policy can be changed after dSource creation too as shown below.



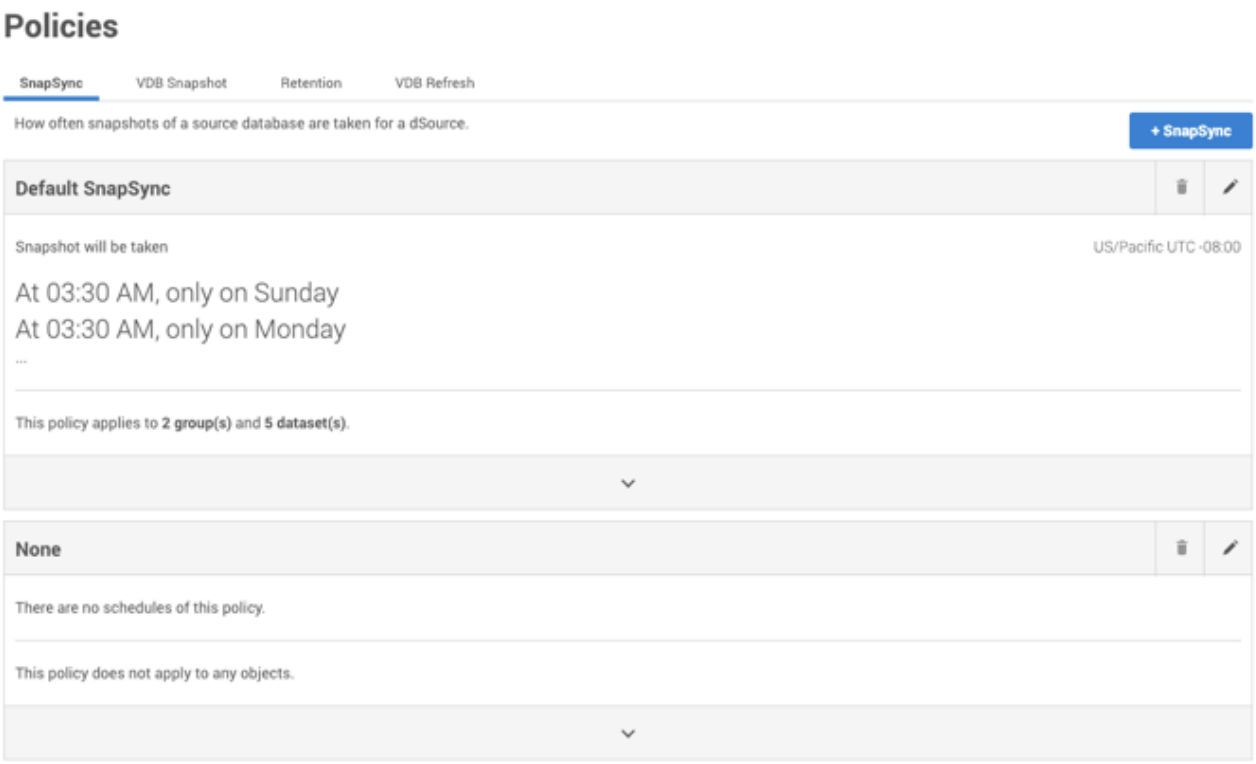
The backup policy is not only provided during the linking phase of a dsource but also during the manual snapsync phase too to provide more flexibility to customers as shown below. Default option is the Backup Policy selected during linking phase.



Understanding snapSync policies

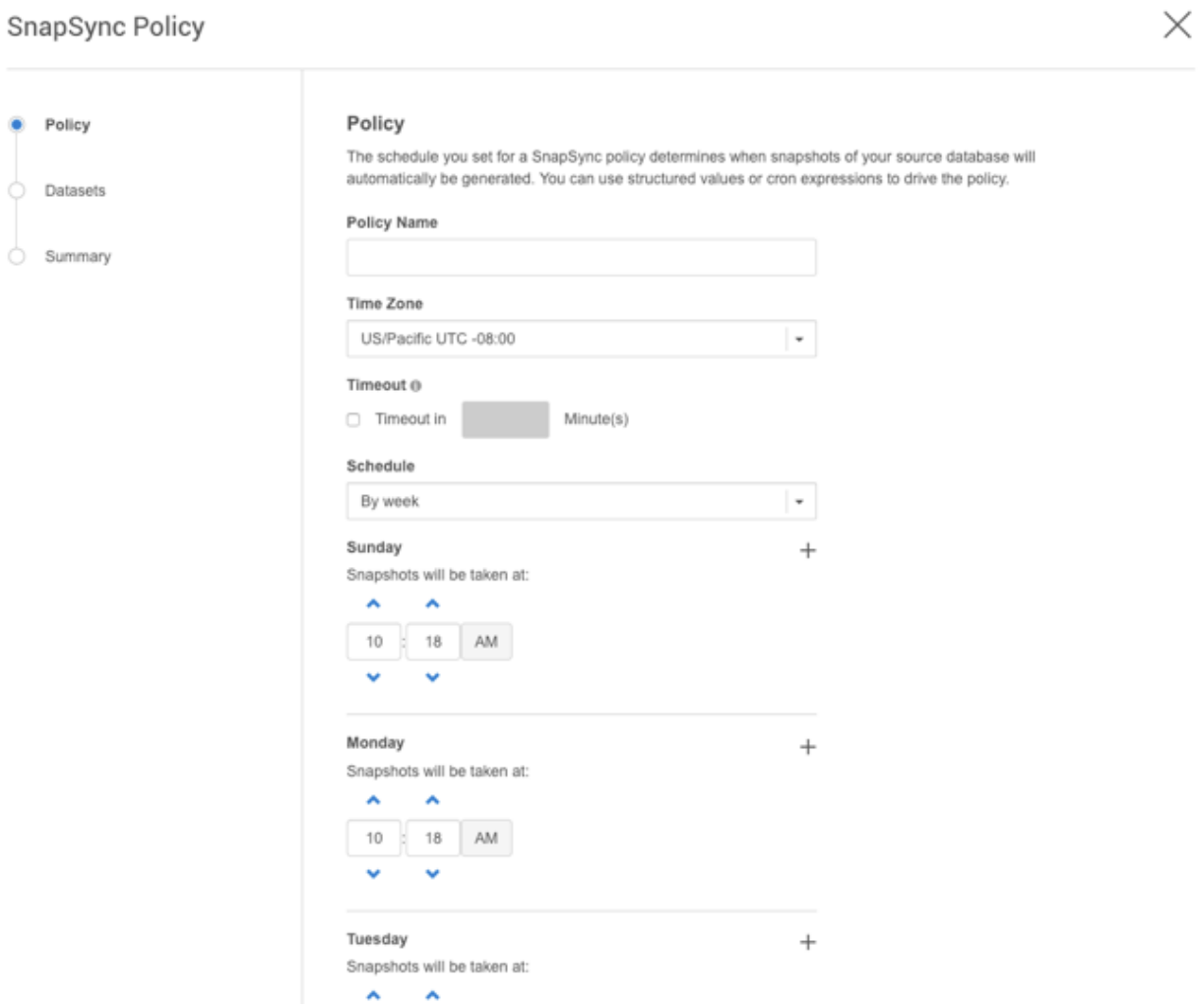
SnapSync policies provide you the ability to specify how frequently the Delphix Engine takes a copy-only full backup of a source database when Delphix Managed Backups are enabled. Selecting an initial SnapSync policy is mandatory at dSource link time. However, you can change the SnapSync policy that the Delphix Engine applies to a dSource at any time by visiting the policy management screen:

1. Click **Manage**.
2. Click **Policies**.



Check SnapSync Policy

For dSources that have Delphix-managed backups enabled, the current SnapSync policy will be displayed under the **SnapSync** column. The rows corresponding to dSources that do not use Delphix Managed Backups will be grayed out. Clicking the **current SnapSync policy** for a dSource will display a drop-down menu of existing SnapSync policies along with the option to create a new SnapSync policy. Selecting a SnapSync policy from this list will change the current SnapSync policy for the dSource. When creating a new policy, you will see the following screen:



Create New SnapSync Policy

Here, you can configure the frequency with which the Delphix Engine takes backups of your source database. You can modify these schedules at any time by clicking the **Modify Policy Templates** button in the upper right-hand corner of the policy management screen.

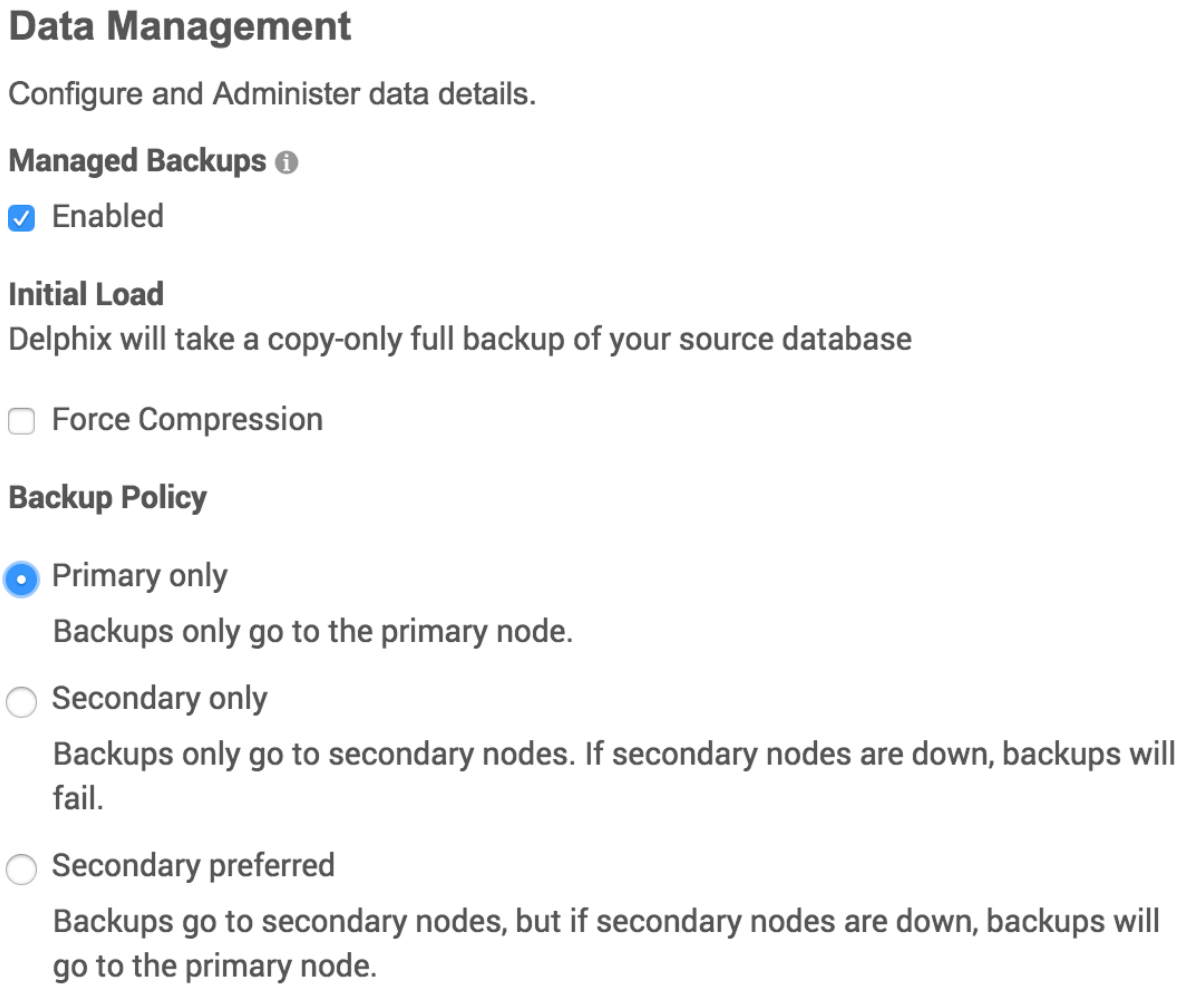
The **Timeout** field above specifies how long a SnapSync job is allowed to run before it is terminated. If a SnapSync job exceeds its timeout window, the Delphix Engine discards the new backup and rolls back the dSource to the most recent snapshot.

Delphix managed backups on secondary nodes of SQL Server alwaysOn availability group cluster

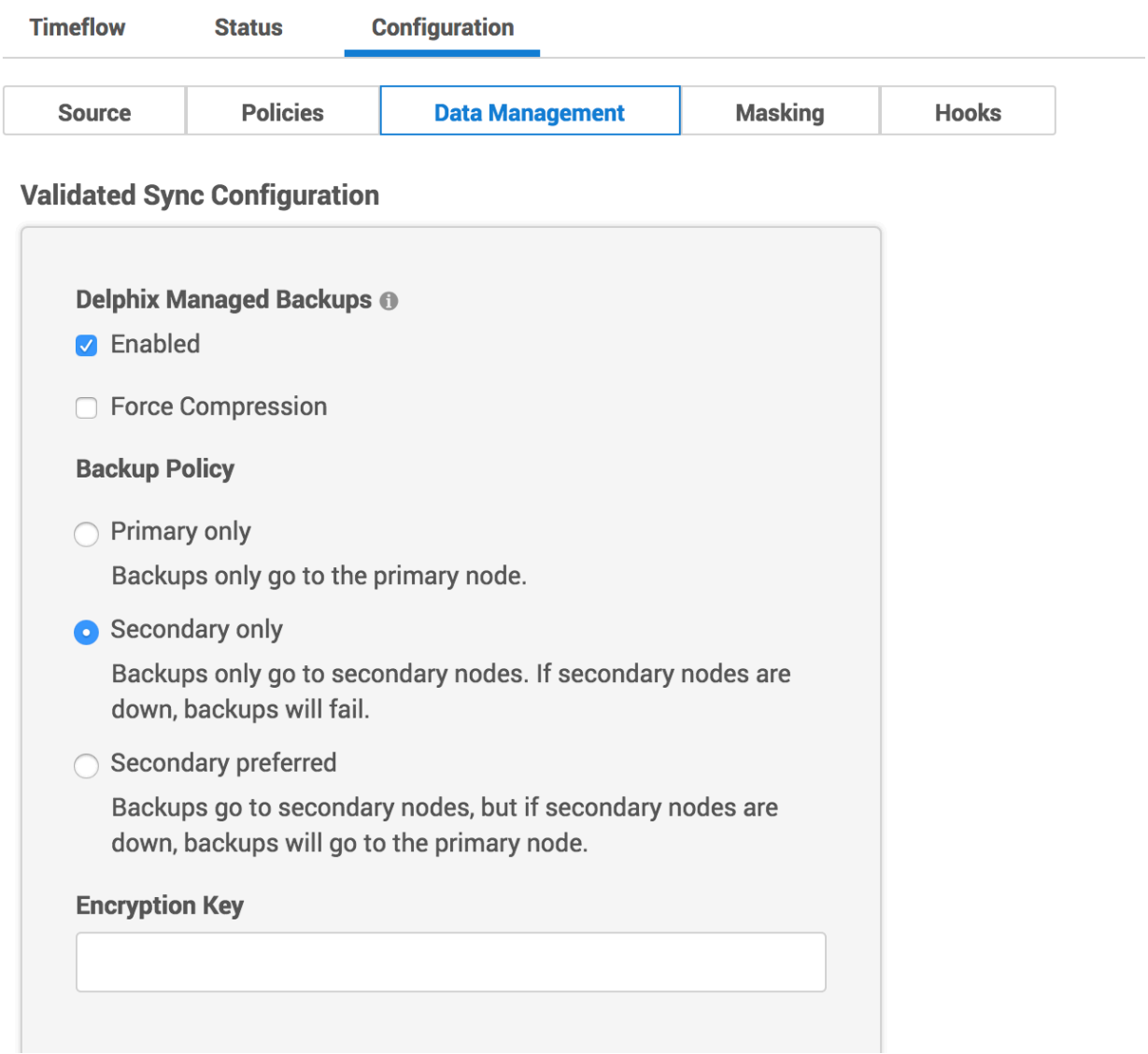
The Always On availability groups feature is a high-availability and disaster-recovery solution that provides an enterprise-level alternative to database mirroring. Introduced in SQL Server 2012 (11.x), Always On availability groups maximizes the availability of a set of user databases for an enterprise. An *availability group* supports a failover environment for a discrete set of user databases, known as *availability databases*, that fail over together. An availability group supports a set of read-write primary databases and one to eight sets of corresponding secondary databases. Optionally, secondary databases can be made available for read-only access and/or some backup operations. Some customers prefer using secondary nodes for backup to reduce load on the primary node while some customers are OK with using primary node. To provide more flexibility for customers using Always On availability groups with Delphix Managed backups, delphix provides an option to choose backup policy during linking of dSource.

They can pick one of the following options that describe their backup policy as shown below.

- Primary (**default option**): Backups are taken only on the primary node.
- Secondary Only: Customers who never want to use primarily for their backups must choose this option as it ensures backups never go to the primary node. If the secondary nodes are down, the backups fail but do not use primary at all.
- Secondary preferred: As the name suggests, one of the secondary nodes is used for backup. The backup will be taken on primary if none of the secondary nodes can be used for backup. So, customers who want reliable backups even if they are taken on the primary node if required should use this option.



The backup policy can be changed after dSource creation too as shown below.



The backup policy is not only provided during the linking phase of a dsource but also during the manual snapsync phase too to provide more flexibility to customers as shown below. The default option is the Backup Policy selected during the linking phase.

Snapshot

- (•) Delphix will take a copy-only full backup of your source database
  - [ ] Force Compression

  Backup Policy

  These values will not persist. They will only be used for this single job.

  - (•) Primary only

    Backups only go to the primary node.
  - ( ) Secondary only

    Backups only go to secondary nodes. If secondary nodes are down, backups will fail.
  - ( ) Secondary preferred

    Backups go to secondary nodes, but if secondary nodes are down, backups will go to the primary node.
- ( ) Use the most recent full or differential backup
- ( ) Use a specific full or differential backup

Cancel | Snapshot

Receive side scaling for windows staging target and targets

Enabling Receive Side Scaling (RSS) on a Windows Target and Staging Target can have a significant improvement in the overall IO throughput to the Delphix Engine and is a best practice. RSS enables network adapters to distribute the kernel-mode network processing load across multiple processor cores in multi-core computers. The distribution of this processing makes it possible to support higher network traffic loads than would be possible if only a single core were to be used.

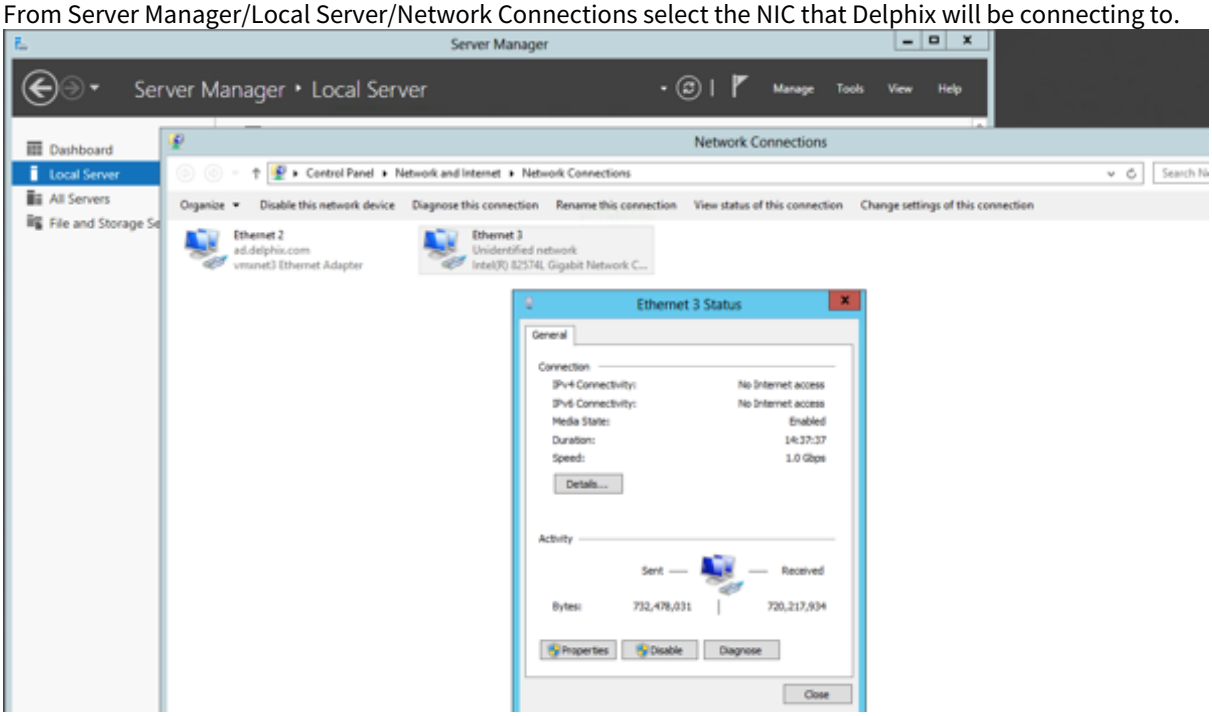
More information on RSS can be found here.

Enabling RSS on the network interface will force the network service to restart and will cause a momentary loss of connectivity on that network interface

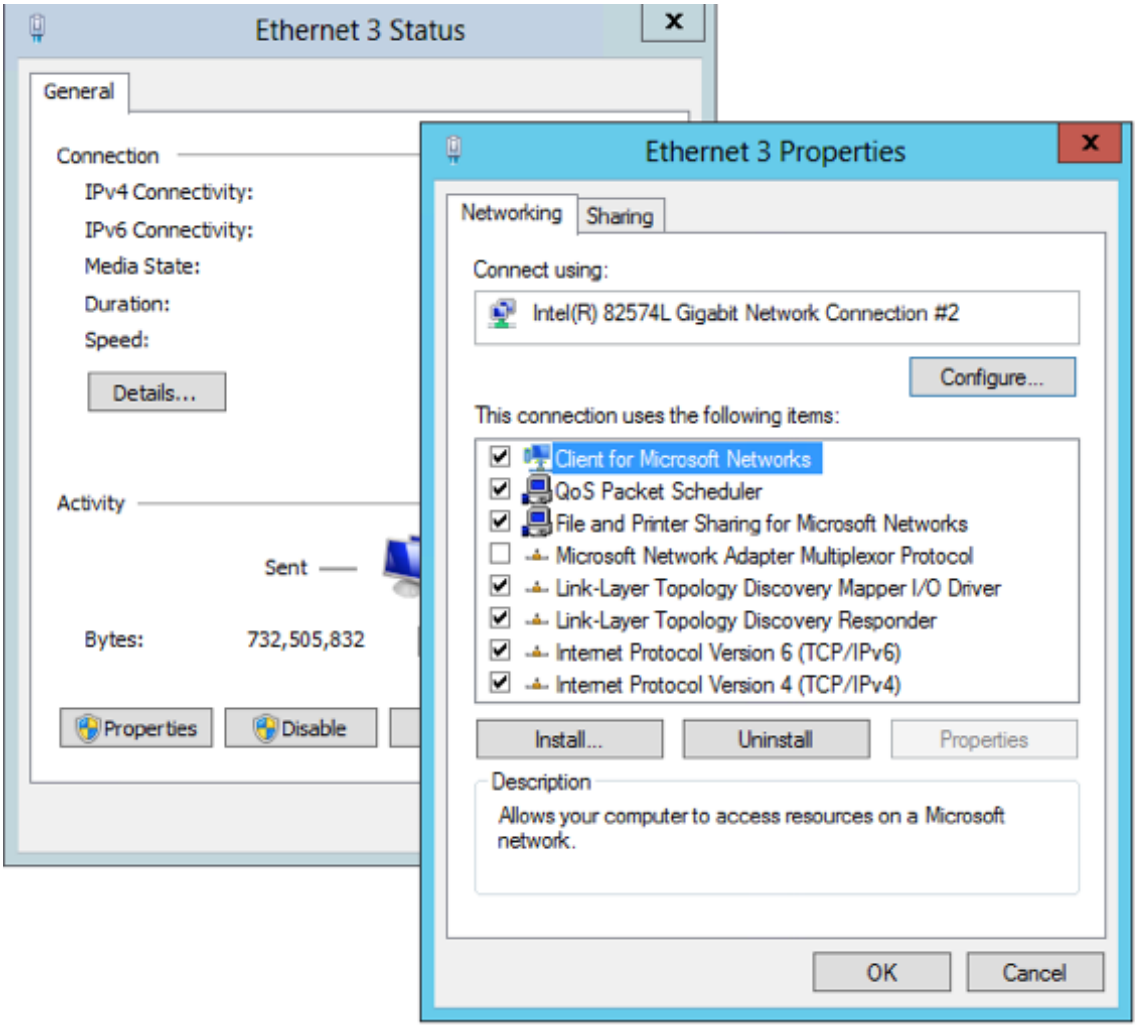
Because hyper-threaded CPUs on the same core processor share the same execution engine, the effect is not the same as having multiple core processors. For this reason, RSS does not use hyper-threaded processors.

Steps to implement RSS on Windows

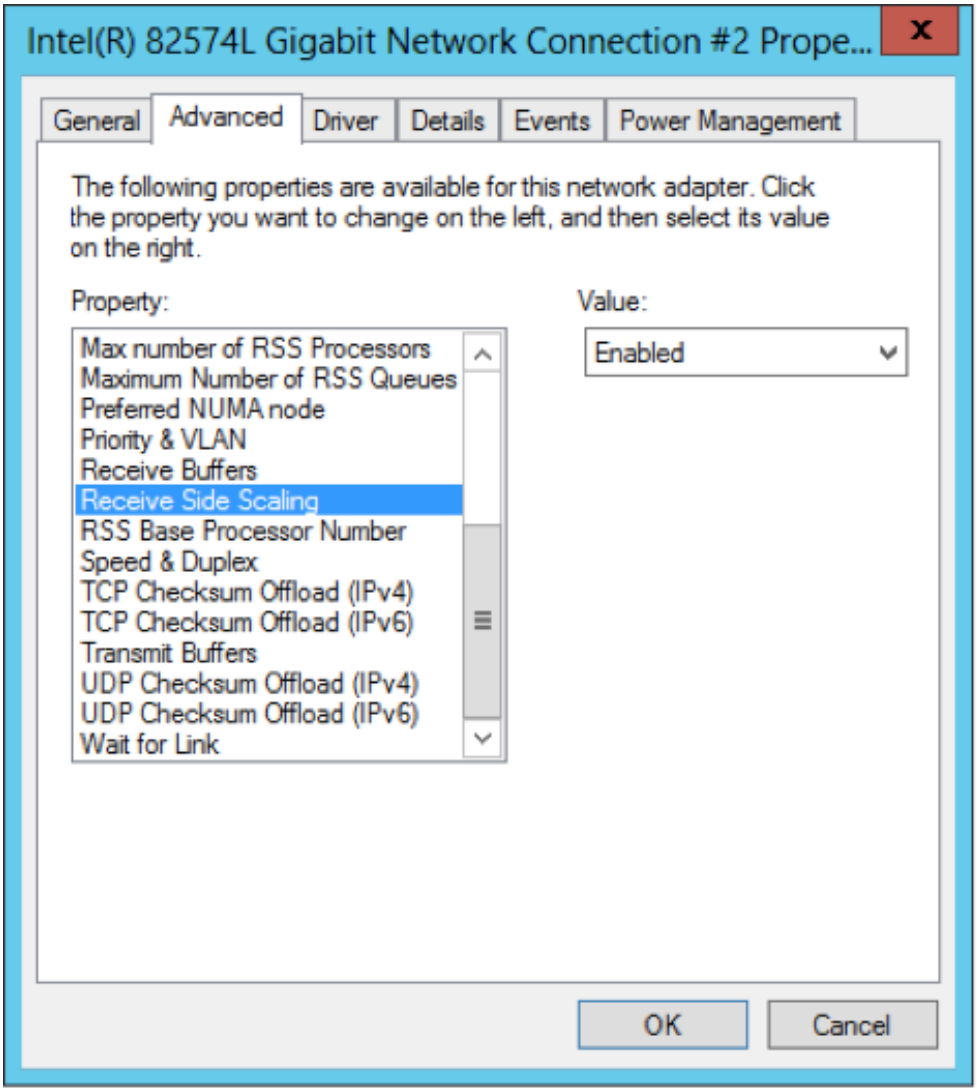
1. From Server Manager/Local Server/Network Connections select the NIC that Delphix will be connecting to.



2. Select Properties and then Configure.



3. From the Property menu on the left, select **Receive Side Scaling**, confirm that it is set to **Enabled**, and select **OK** to close each of the open windows.


Intel(R) 82574L Gigabit Network Connection #2 Prope...
General
Advanced
Driver
Details
Events
Power Management
The following properties are available for this network adapter. Click the property you want to change on the left, and then select its value on the right.
Property:
Max number of RSS Processors
Maximum Number of RSS Queues
Preferred NUMA node
Priority & VLAN
Receive Buffers
Receive Side Scaling
RSS Base Processor Number
Speed & Duplex
TCP Checksum Offload (IPv4)
TCP Checksum Offload (IPv6)
Transmit Buffers
UDP Checksum Offload (IPv4)
UDP Checksum Offload (IPv6)
Wait for Link
Value:
Enabled
OK
Cancel

Validated sync environment

This topic describes additional requirements for SQL Server environments that will be used as targets for validated sync. You must configure a staging (Validated Sync) environment as a target, with a few additional requirements.

Requirements for SQL server validated sync target environments

Each SQL Server target environment used for validated sync must meet these requirements:

- Only standalone target environments can be used as validated sync target environments. Windows Failover Cluster target environments and SQL Server Failover Cluster instances cannot be used.
- The SQL Server instance must be the same version as the instance hosting the source database. For more information about compatibility between different versions of SQL Server, see SQL Server Support Matrix
- The owner of the SQL Server instances on the target environment that are used for the staging databases must have SMB read access to the location containing the backup images of the source databases
- If the source database is backed up with third-party backup software like LiteSpeed or Red Gate SQL Backup Pro, you must install the backup software on both the source and the validated sync environment. For backup software compatibility requirements, see SQL Server Support Matrix

Add the validated sync environment

The order is important. Add the validated sync environment as the first step in setting up the SQL Server topology.

1. From the machine that you want to use as a target, start a browser session and connect to the **Delphix Management** application**.**
2. Click **Manage**.
3. Select **Environments**.
4. Next to **Environments**, click the **Actions** menu and select **Add Environment**.
5. In the **Add Environment**wizard Host and Server tab, select:
    a. Host OS: **Windows**
    b. Host Type: **Target**.
    c. Server Type: **Standalone**.
6. Click **Next**.
7. In the Environment Settings tab click the download link for the **Delphix Connector Installer**.
   The Delphix Connector will download to your local machine.
8. On the Windows machine that you want to want to use as a target, run the Delphix Connector installer. Click **Next**to advance through each of the installation wizard screens.
   The installer will only run on 64-bit Windows systems. 32-bit systems are not supported.
    a. For **Connector Configuration**, make sure there is no firewall in your environment blocking traffic to the port on the target environment that the Delphix Connector service will listen to.
    b. For **Select Installation Folder**, either accept the default folder, or click **Browse** to select another.
    c. Click **Next** on the installer final 'Confirm Installation' dialog to complete the installation process and then **Close** to exit the Delphix Connector Install Program.
9. Return to the Delphix Management application.
10. Enter the **Environment Name**, **Host Address**, **Delphix Connector Port**, **OS Username**, and **OS Password** for the target environment.
11. To provide your own Oracle Java select the **Provide my own JDK** checkbox and click **Next**.
12. In the Java Development Kit tab enter the absolute path to your Oracle JDK and click **Next**.
13. Click **Submit.**

As the new environment is added, you will see two jobs running in the **Delphix Admin Job History**, one to **Create and Discover** an environment, and another to **Create** an environment. When the jobs are complete, you will see the new environment added to the list in the **Environments** panel.

Changing the hostname (IP address) or IQN of a SQL server target or staging host

This topic describes how to change the hostname (IP address) or iSCSI Qualified Name (IQN) of a Windows Target or Staging host.

By default, Windows servers generate an IQN based on the hostname assigned to it. Changing the hostname will change the host IQN as well. Because the Delphix Engine exports storage for dSources and VDBs to Windows hosts using iSCSI, changes to the Windows hostname must be made according to the following procedure. If you have set a non-default IQN on a Windows Target or Staging host, and want to change that IQN, you must follow these procedures.

> Changing the hostname or IQN of a Windows target or staging server requires that you modify the iSCSI Initiator configuration on the Windows host. Doing so incorrectly can cause failures in dSources, VDBs, or non-Delphix users of iSCSI on the Windows host.
>
> The instructions in this topic describe how to change the IQN using the iscsicli command-line utility. Because many people are less familiar with the iscsicli utility, the instructions also include information for using the iSCSI Initiator graphical user interface.
>
> Failing to carefully follow the steps below in sequence can cause availability issues for your dSources and VDBs. If you have questions about the following instructions, please contact Delphix Support for help.

1. Disable the dSources.
2. Disable the VDB's.

> If your Windows server has dSources or VDBs from more than one Delphix Engine, you will need to disable the dSources and VDBs on each Delphix Engine.

3. Remove any remaining persistent volumes from the Windows server. From the Server Manager\Tools**iSCSI Initiator** configuration tool, use the options available in the **Volumes and Devices** tab. or, Follow these steps to use the iscsicli command-line utility: List the persistent volumes.

```
PS C:\> iscsicli reportpersistentdevices
Microsoft iSCSI Initiator Version 6.1 Build 7601
Persistent Volumes
"\\?\storage#volume#{bb38add1-d03f-11e1-8767-005056b37fe6}
#0000000008010000#{53f5630d-b6bf-11d0-94f2-00a0c91efb8b}"
"C:\Program Files\Delphix\DelphixConnector\564d6fbb-df9d-e90b-00f1-da37b17011d3-
staging-15\ARCHIVE\"
[...]

The operation completed successfully.
```

a. Volumes with a "normal" path correspond to mounted volumes. For example, `C:\Program Files\Delphix` is a normal path. If you see any normal paths in the output, be sure you have disabled all of the VDBs and dSources.

b. Volumes with a path beginning `\\?\` correspond to unmounted persistent volumes. Remove each of them:

```
PS C:\> iscsicli RemovePersistentDevice
"\\?\storage#volume#{bb38add1-d03f-11e1-8767-005056b37fe6}
#0000000008010000#{53f5630d-b6bf-11d0-94f2-00a0c91efb8b}"
```

c. Alternatively, if all of the persistent devices are for unmounted volumes, you can remove them all at once with this command:

```
PS C:\> iscsicli clearpersistentdevices
```

4. Remove all of the persistent targets. From the Server Manager\Tools**iSCSI Initiator** configuration tool, use the options available in the **Favorite Targets** tab. or,
   Follow these steps to use iscsicli command-line utility:

a. List persistent targets:

```
PS C:\> iscsicli ListPersistentTargets
```

Remove the appropriate persistent targets. Below is sample output listing the persistent targets:

```
PS C:\> iscsicli ListPersistentTargets
    [...]
    Target Name           : iqn.2008-07.com.delphix:02:02843619-12c4-e4d2-8041-
f5c56a647bc2
    Address and Socket    : 10.43.5.45 3260
    Session Type          : Data
    Initiator Name        : Root\ISCSIPRT\0000_0
    Port Number           : <Any Port>
    Security Flags        : 0x0
    Version              : 0
    Information Specified: 0x20
    Login Flags          : 0x0
    Username             :
    [...]
```

⚠ Misleading Help for RemovePersistentTarget Command
The help for iscsicli `RemovePersistentTarget` is misleading:

```
iscsicli RemovePersistentTarget <Initiator Name> <TargetName>
                                <Port Number>
                                <Target Portal Address>
                                <Target Portal Socket>
```

⚠ `<Initiator Name>` and `<Target Name>` show up in the listing and should be taken directly from there. `<Port Number>` can be taken from the listing output, but a `*` should be used if `<Any Port>` is listed. `<Target Portal Address>` and `<Target Portal Socket>` are shortened to `Address` and `Socket` in the `ListPersistentTargets` output. The term `Socket` in both places is what is more typically referred to as a `port`.

c. Use the `RemovePersistentTarget` command to remove the target, as shown in this example:

```
PS C:\> iscsicli RemovePersistentTarget Root\ISCSIPRT\0000_0 iqn.2008-07.com.delphix:02:02843619-12c4-e4d2-8041-f5c56a647bc2 * 10.43.5.45 3260
```

5. Log out of any sessions.
   From the Server Manager\Tools**iSCSI Initiator** configuration tool, use the options available in the **Targets** tab to log out. Selected a connected session under **Discovered Targets**, and then click **Disconnect**.
   or, Follow these steps to use the iscsicli command-line utility: List the sessions.

```
PS C:\> iscsicli sessionlist
Session Id : fffffa8003fb0018-4000013700000001
Initiator Node Name : iqn.1991-05.com.microsoft:10-43-1-200.ad.delphix.com
Target Node Name : (null)
Target Name : iqn.2008-07.com.delphix:02:02843619-12c4-e4d2-8041-f5c56a647bc2
[...]
```

b. Log out from the target.

```
PS C:\> iscsicli logouttarget fffffa8003fb0018-4000013700000001
```

6. Change the hostname (IP address) or IQN in the Delphix engine.

   a. If you are changing the hostname, follow the instructions in the Microsoft TechNet article "Rename the Computer."

   Note that if the computer is on a domain, you will need a domain administrator to perform the rename or re-add the computer to the domain depending on the version of Windows it is running.

   b. To use the Delphix Setup application, go to **Manage** → **Environments** screen and select the Target host. To modify the Host Address for the environment, use the Edit (Pencil) icon next to the **Attributes** panel. Update the Host Address to reflect the new IP address, and then use the **Save (Tick)** icon to confirm the change. This will automatically trigger an action to Refresh the Environment. The refresh process includes steps to configure the iSCSI Target and verify that it is reachable.

   c. If you are changing the IQN only, change it through the Microsoft iSCSI Initiator GUI following the instructions in the Microsoft iSCSI User Guide.

7. Wait for the computer to finish rebooting.
8. Verify the new IQN in the iSCSI initiator.

If you are using the default IQN and have changed the hostname (IP address), the IQN should include the new hostname.

9. Refresh the environment on the Delphix Engine.Re-enable the dSources.
10. Re-enable the dSources.
11. Re-enable the VDBs.
12. Using the iscsi cli command-line utility, verify that the sessions on the Windows server are using the new IQN.

```
PS C:\> iscsicli sessionlist
Microsoft iSCSI Initiator Version 6.1 Build 7601

Total of 1 sessions

Session Id : fffffa8003f77018-4000013700000004
Initiator Node Name : <NEW IQN>
[...]
```

Linking data sources and syncing data with SQL Server

Creating a dSource will ingest data from the source and create a dSource on the engine. The dSource is an object that the Continuous Data Engine uses to create and update virtual copies of your database. As a virtualized representation of your source data, it cannot be managed, manipulated, or examined by database tools.

This section covers the following topics:

Linking a dSource from a SQL server: An overview

When linking a dSource from a SQL Server source database, Delphix offers several different methods of capturing backup information:

- Delphix Managed Backups, where the Delphix Engine schedules and initiates the backups from the source database, and captures them
- SQL Server Managed Backups, where the SQL Server source database schedules and initiates backups and the Delphix Engine captures them
    - Full backups
    - Full or differential backups
    - Transaction log backups (with LogSync disabled)
    - Transaction log backups (with LogSync enabled)

Delphix Managed Backups are conceptually a lot simpler to explain, but they can be considered less desirable because they might be viewed as intrusive. SQL Server Managed Backups are explained in two sections:

- the initial load of the dSource from the source database
- subsequently keeping the dSource synchronized with the source database.

Below is a brief explanation of how these three different modes of operation work.

Delphix managed backup mode

When the **Enabled** checkbox for Delphix Managed Backups is selected, the Delphix Engine will initiate a COPY_ONLY full backup of the source database for the initial load of the dSource. Thereafter, the Delphix Engine will initiate COPY_ONLY full backups of the source database using the schedule specified by the selected SnapSync Policy. If SnapSync Policy is set to None, the Delphix Engine will not automatically initiate COPY_ONLY backups, but you can initiate them manually using the snapshot (camera) icon.

When the **Force Compression** checkbox for Delphix Managed Backups is selected and the backups are not compressed, the Delphix Engine will take a compressed copy-only full backup of the source database and this will take effect during the next scheduled Snapshot or SnapSync operation.

> COPY_ONLY backup files are written to the Delphix storage that has been mounted on the staging server. Delphix does not require space on the source or staging servers to hold the backup files.

You must also select the Staging Environment and the SQL Server instance onto which the backups will be restored.

SQL server managed backup modes

Initial load of the dSource

When the Delphix Managed Backups option is left unchecked (*default*), the Delphix Engine will initiate a backup only when the user selects to initiate a COPY_ONLY full backup that the Delphix Engine will use to keep the dSource in sync with the source database.

For the initial load of the dSource, you can choose one of the following:

- have the Delphix Engine initiate a COPY_ONLY full backup
- use the most-recent existing full or differential backup (*default*)
- use a specific existing full or differential backup identified by its `backup_set_uuid`

> Simple Recovery Model

If the source database is using a simple recovery model, using a new full COPY_ONLY backup initiated by the Delphix Engine is not supported for the initial load of a dSource.

After the initial load, you need to select the Backup Paths and tell the Delphix Engine where to look for backups of the source database.

- If **Autodiscover** is selected, the exact path used to take the backup from the source database will be determined by querying by the source database instance. If this option is used, the Delphix Engine should take backups to a UNC path (Windows file share) so that they are accessible to the Staging Server.
- If custom paths are specified, the Delphix Engine will query the source database instance, identify the filename of the source backup, and then recursively search the specified Backup Paths for this filename. These paths should also be UNC paths (Windows file share) which are accessible to the Staging Server.

The path used to restore backups must be readable by the Windows server hosting the staging instance**,** using the staging environment's configured Environment User**.**

The Delphix Engine supports source database backups that SQL Server creates natively, as well as backups created by Quest/Netvault LiteSpeed, Red Gate SQL Backup Pro, Veritas NetBackup, and Commvault. For more information, see the topic SQL Server Support Matrix.

Once you have decided how the dSource will be initially loaded, select the staging environment and the SQL Server instance onto which the backups will be restored. NetBackup and Commvault backups are not on local storage and therefore the Backup Paths will just be ignored. See Linking a dSource from a NetBackup SQL Server Backup and Linking a dSource from a Commvault SQL Server Backup for more information.

The staging instance opens the backup file for reading and may hold a lock on it when restoring the backup on the staging database. A new source database backup, initiated with the **Append to the existing backup set** option, may fail as SQL Server will not be able to open the locked backup file to append a new backup to it.

Keeping the dSource synchronized with the source database

Next, specify how the Delphix Engine will capture subsequent backups of the source database.

The selected Validated Sync mode determines how often the Delphix Engine will check for new backups, and which type of backups it will check for. You can always force synchronization with the source database by enabling **Validated Sync Mode** from the **Data Management** tab available under the **Configuration** tab for the selected dSource.

Validated sync and logSync

When you link a source database into the Delphix Engine, a staging database will still be required if the Validated Sync is not enabled, as described in Overview of Setting Up SQL Server Environments. In this process, the Delphix Engine continuously monitors the source database for new full and differential backups if the source database is using a **simple** recovery model, or transaction log backups if using a **full** recovery model. This will also depend on the selected backup mode. When it detects a new backup, it restores that backup to the staging database with the storage residing in Delphix. The result is a TimeFlow with consistent points from which you can provision a VDB, also known as snapshots.

Snapshots accumulate over time. To view a snapshot:

1. From the **Datasets** panel, click the **group** containing the dSource.
2. Select **dSource**.
3. Click the **TimeFlow** tab.

Each snapshot is displayed and includes some information about the captured database along with Snapshot database change number (SCN for Oracle and LSN for SQL Server). You can scroll through these cards to select the one you want, or you can enter a date and time to search for a specific snapshot.

Summary of validated sync modes

This table summarizes each mode of Validated Sync, displaying how often the Delphix Engine will poll to check for new backup files when it creates snapshots for the dSource, and whether point-in-time restores for provisioning and refreshing virtual databases (VDBs) is possible or not.

Validated Sync Mode	Polling Interval	Snapshot for each FULL backup	Snapshot for each DIFF backup	Snapshot for each TLOG backup	Allows Point-in-time Restores	Notes
Delphix Managed Backups	N/A	N	N	N	N	Takes COPY_ONLY full backups according to SnapSync schedule. For more information on this option, see Delphix as a Backup Solution to SQL Server.
Transaction log backups (LogSync DISABLED)	1-minute	N	N	Y	N	Log backups are not collected if: • There are gaps in the sequence of log backups (a break in the "log chain") • The available log backups do not include any changes since the last successful Delphix snapshot
Transaction log backups (LogSync ENABLED)	1-minute	N	N	Y	Y	Log files consume additional space on the Delphix Engine and are managed according to the defined retention policy for logs. For NetBackup and Commvault backups, Point-in-time restores are not supported.

Full or differential backups	1-minute	Y	Y	N	N	
Full backups	1-minute	Y	N	N	N	
None	Manual only	N	N	N	N	Only retrieves backups when you initiate a manual snapshot.

Timeflow cards
The Delphix Engine will create a Timeflow card for each backup it restores to the staging server. For example:

- A database in Full backups Validated Sync mode, and daily backup configured on the dSource, would receive one Timeflow card per day
- A database in Full or differential backups sync mode, with one daily backup and two differential backups per day, would receive three Timeflow cards per day
- A database in Transaction log backups sync mode, with a log backup every 15 minutes, would receive 96 TimeFlow cards per day

Linking data sources with SQL server

Linking a dSource will ingest data from the source and create a dSource object on the engine. The dSource is an object that the Continuous Data Engine uses to create and update virtual copies of your database. As a virtualized representation of your source data, it cannot be managed, manipulated, or examined by database tools.

For an overview of all dSource related actions, please Managing Data Sources and Syncing Data.

When linking a dSource from a SQL Server source database, Delphix offers several different methods of capturing backup information:

- SQL Server Managed Backups, where the SQL Server source database schedules and initiates backups and the Delphix Engine captures them
    - Full backups
    - Full or differential backups
    - Transaction log backups (with LogSync disabled)
    - Transaction log backups (with LogSync enabled)
- Delphix Managed Backups, where the Delphix Engine schedules and initiates the backups from the source database, and captures them

Procedure

1. Login to the **Delphix Management** application.
2. Navigate to **Manage > Datasets.**
3. Click the plus icon and select **Add dSource**.
4. In the **Add dSource** wizard, select the source database with the correct environment user-specified.
5. Select user type for source database authentication and enter the login credentials. Enter username and password for Database user or Domain (Windows) user. For Environment User, select a source environment user from the dropdown list and click **Next**.
6. Enter a name and select a group for your dSource.
   Adding a dSource to a dataset group lets you set Delphix Domain user permissions for that database and its objects, such as snapshots. See the topics under Users and Groups for more information.
7. Select the **Data Management** settings needed. For more information, Data Management Settings for SQL Server Data Sources.
8. Select the Staging environment and SQL Instance that will be used to manage the staging database used for validated sync of the dSource.
9. Select any policies for the new dSource.
10. Enter any scripts that should be run on the **Hooks** page.
11. Review the dSource Configuration and Data Management information, and then click **Submit**.

Staging push implementation for SQL server

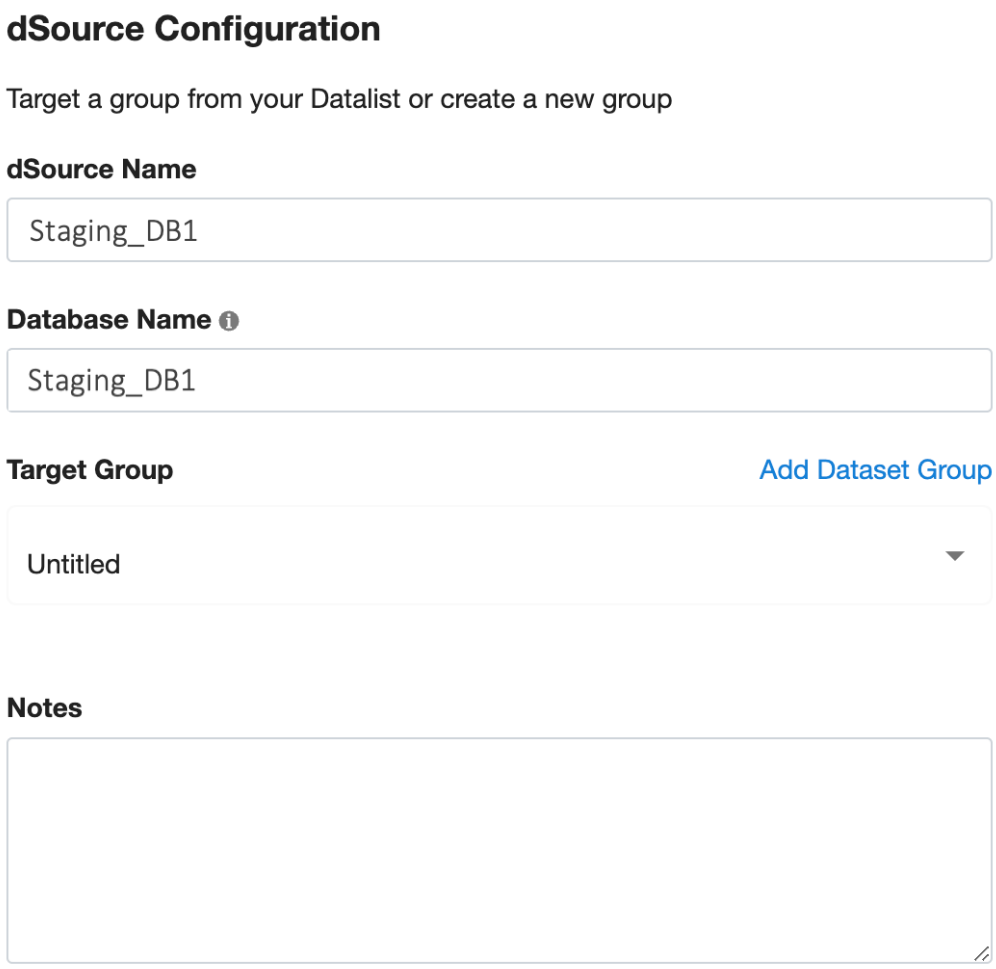
This topic provides implementation details of Staging Push for SQL Server. Staging Push eliminates the need of accessing the customer's production environment and hence the dependency on it. For end-users with a wide variety of architectural requirements such as an unsupported backup appliance, complex ingestion requirements, and supporting alternate replication methods, Staging Push increases flexibility by ensuring that Delphix can work with most of these unique requirements while maintaining Delphix standard product support.

Staging Push gives end-users control over some Staging DB processes so that the nuanced, staging-based ingestion can be orchestrated externally. It will give control of the staging database to the end-user to pull from any backup provider (as a part of this, you'll be responsible for keeping the Staging DB in sync with the production database). Staging database files will be stored on Delphix Storage. Delphix will still be the one snapshotting the underlying data files, and gathering the metadata required to provision from the snapshot.

The below steps show how to create a dSource using the Staging Push mechanism.

Procedure

1. Login to the **Delphix Management** application.
2. Navigate to **Manage > Datasets.**
3. Click the plus icon and select **Add dSource**.
4. On the Preparation tab, click **Next**.
5. From the **dSource Type** tab, select the **Staging Push** option and click **Next**.
6. From the **dSource Configuration** tab, enter the following dSource configurations, also shown in the screenshot below, and click **Next**. **dSource Name** - This name will be available on the Delphix Engine interface. **Database Name** - This is the name of the staging database that is created on the staging host after linking. It is suggested to use the word "staging" while assigning a name to the staging database as this will help to distinguish it from other end-user databases.



7. From the **Data Management** tab, select the staging environment and repository details. The staging environment is where the staging database will be hosted and a repository is a container for the SourceConfigs objects. Each environment in Delphix can contain any number of SQL Server instances and each SQL Server instance can contain any number of databases.
8. Select any policy for the new dSource. SnapSync policy is used as a default policy for taking snapshots. For more details on SnapSync policy, see Policies for Scheduled Jobs.
9. Enter any script that should be run on the **Hooks** page. For staging push dSources, a restore backup script can be added as a part of the hooks script to automate backup restore. Refer to the below section for more details.
10. Review the Staging Push dSource Configuration and Data Management information, and then click **Submit**.

Restoring backups

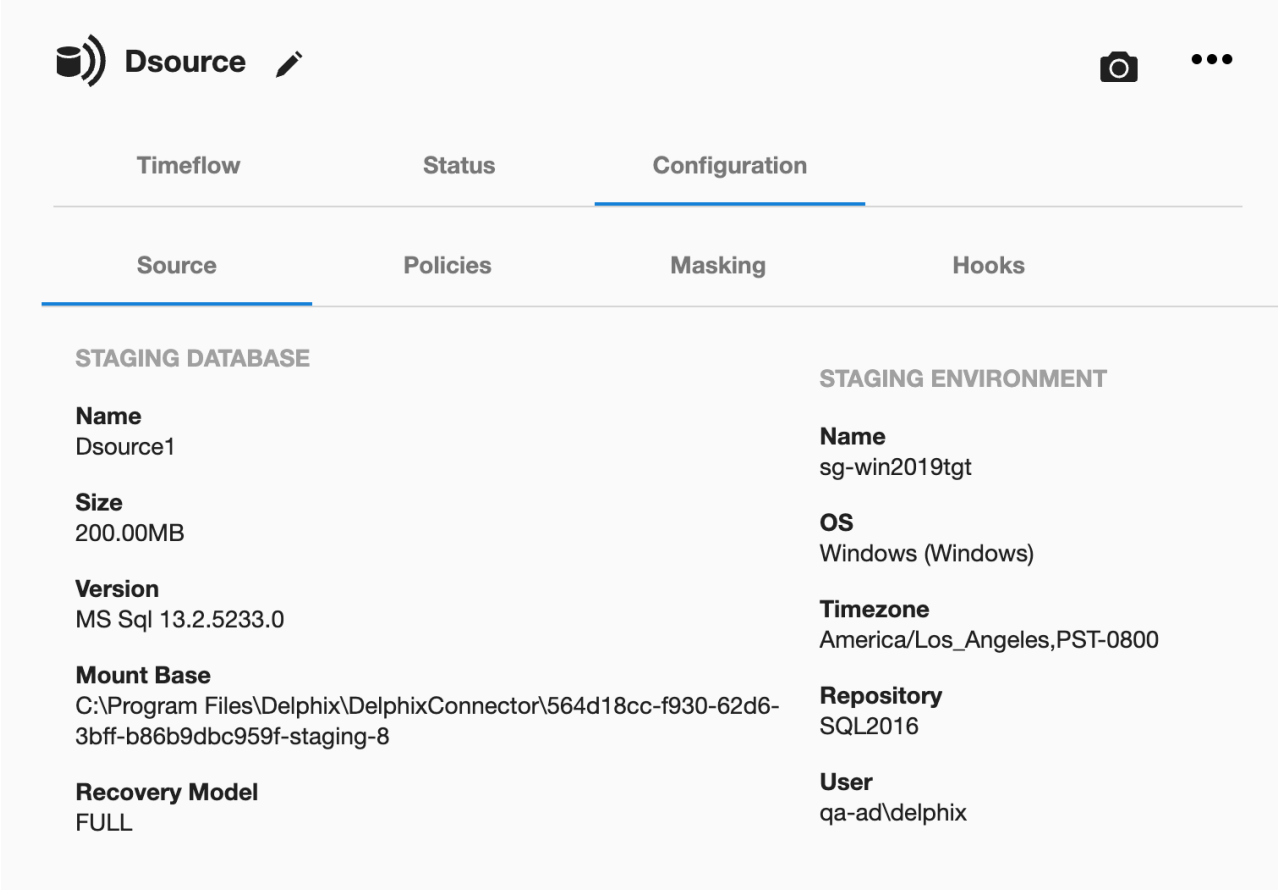
After linking and before taking a snapshot, if the user wants to align the staging database with the source database automatically they can do so by providing the restore script in the hook, as shown below.
They can also manually restore a database backup on the staging host.

First backup restoration needs to be a full backup with the REPLACE keyword.

Prerequisites

When restoring the backup, make sure that all the database files are present at the mounting location.

The mounting location is available as **Mount Base** under the **Configuration** tab of the newly created dSource.



Procedure

Manually restore a backup on the staging host or provide the restore script in the hook script. A sample pre-sync hook for native backup is shown below. It also lists the following keywords.

- NORECOVERY - Keeps the database in restoring state.
- REPLACE - Overwrites existing database created during linking with whichever source database that is in the backup set and is getting restored.
- MOVE - Restores the data and log file to the specified locations.
- BACKUP_FILE_LOCATION - The location of the backup file of the source database from where the backup is restored and it should be accessible from the staging host. For example, \\10-43-89-18\Backup\sp.bak.

Sample hook to illustrate restore script usage for restoring backups

```
#
# Copyright (c) 2022 by Delphix. All rights reserved.
#
#Set-Variable UTF8_CODEPAGE 65001 -option readonly
```

```
#Set-Variable UTF8_ENCODING "System.Text.UTF8Encoding" -option readonly
# Uncomment the following log to turn on debugging
# Set-PSDebug -Trace 2;
$DSOURCE_HOST = $env:STAGING_INSTANCE_HOST
$DSOURCE_PORT = $env:STAGING_INSTANCE_PORT
$DSOURCE_INSTANCE = $env:STAGING_INSTANCE_NAME
$CONNECT_STRING = "$DSOURCE_HOST\$DSOURCE_INSTANCE,$DSOURCE_PORT"
$DSOURCE_NAME = "$env:STAGING_DATABASE_NAME"
$DataDbFilePath = "$env:STAGING_MOUNT_BASE"
$SQL_SCRIPT= "RESTORE DATABASE $DSOURCE_NAME FROM DISK = '\\BACKUP_FILE_LOCATION'
WITH NORECOVERY, REPLACE, MOVE N'<Source_db_name>' TO '$DataDbFilePath\DATA\db\
$DSOURCE_NAME.mdf', MOVE N'<Source_db_name>_log' TO '$DataDbFilePath\DATA\db\" +
$DSOURCE_NAME + "_log.ldf'"
function die {
    Write-Error "Error: $($args[0])"

    # run exit handler, if defined
    if (Get-Command -type Function -name atExit 2> $null) {
        atExit
    }
    exit 1
}

function verifySuccess {
    if (!$?) {
        die "$($args[0])"
    }
}

### Restore database.

echo $SQL_SCRIPT
Sqlcmd  -b -S $CONNECT_STRING -U sa -P <PWD>  -Q $SQL_SCRIPT
verifySuccess "Failed to restore backup"

# Uncomment the following line to turn off debugging
# Set-PSDebug -Trace 0;

exit 0
```

Snapshot

The first snapshot is created as a part of dSource creation and contains data and log files within the Delphix created mount point.

> The first snapshot created is of an empty database and does not contain any source database's data. (Unless a source backup was restored via a pre-sync hook. In that case, the initial snapshot will be that of source backup and won't be empty).

Prerequisites

- The database should be present on the staging host.

- The DATA directory should be mounted.
- The DB files should be present on Delphix mounted DB directory. For example, `C:\Program Files\Delphix\DelphixConnector\ec2197b2-e0c6-48d2-bd14-265e6fa9b5ab-staging-1\DATA\db`
- The database should be in Restoring state.
- No other restore operation should be in progress on the staging database.

SnapSync criteria

- Delphix tries to fetch the last restored backup. If no backup is found, the snapshot is skipped with a warning.
- If a snapshot already exists for the backup in the current timeflow, the snapshot is skipped with a warning.

> Snapshots display the Staging Host timezone, as opposed to Linked dSources, where snapshots display the Source Host timezone. Here, the Staging timezone is displayed for Staging Push as we don't have the Source host to fetch the timezone information. This functioning might change in the future.

Perform the following steps to take a snapshot:

1. Login to the **Delphix Management** application.
2. Click **Manage** and select **Datasets** from the dropdown list.
3. Select the dSource created using the Staging Push mechanism to Snapshot.
4. Click the **Camera** icon.
5. From the Snapshot dialog box, select **Yes**.
6. Navigate to the **Timeflow** tab and click **View: All snapshots** to verify the Snapshot you just created.
   You can now proceed to provision the VDB using the snapshot.

Disabling and enabling the dSource

When a disabled dSource is later enabled, it will resume communication and incremental data updates from the staging database according to the original policies and data management configurations that you set.

Procedure

Disabling a dSource will stop further operations on the Delphix Engine related to the staging dSource.

1. Login to the Delphix Management application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **dSource** you want to disable.
5. In the upper right-hand corner, from the **Actions** menu (...) select **Disable.**
6. In the Disable dialog select **Disable**.

When you are ready to enable the dSource again, from the Actions menu (...) select **Enable**, and the dSource will continue to function as it did previously.

Keep a note of the following:

- During enable, if there exists a database with the same staging database name as the user entered and it is not Delphix managed, or if it is but Delphix Engine is unable to drop it, the enable operation will not be terminated.
- Storage mounting will be attempted for the dSource and a new job event `SKIP_ATTACH_OPERATION` wil l be displayed. This means the job is updated with a warning and attach operation is skipped. The enable operation will be marked as successful.
  Note : Customers can drop the database that is outside Delphix or that wasn’t dropped and try again the disable/enable option.

- Delphix will attempt to mount the staging database's storage during the enable job. Even if the attach job fails, the storage will remain mounted (if Delphix was able to successfully mount it) and the enable job will be marked as successful.

Unlink(Detach)/link(attach) a dSource

Each dSource contains metadata that associates it with the staging database, as well as the data it has ingested from the staging database in the form of snapshots up to that point. It is possible to detach, or unlink, a dSource from its staging database.

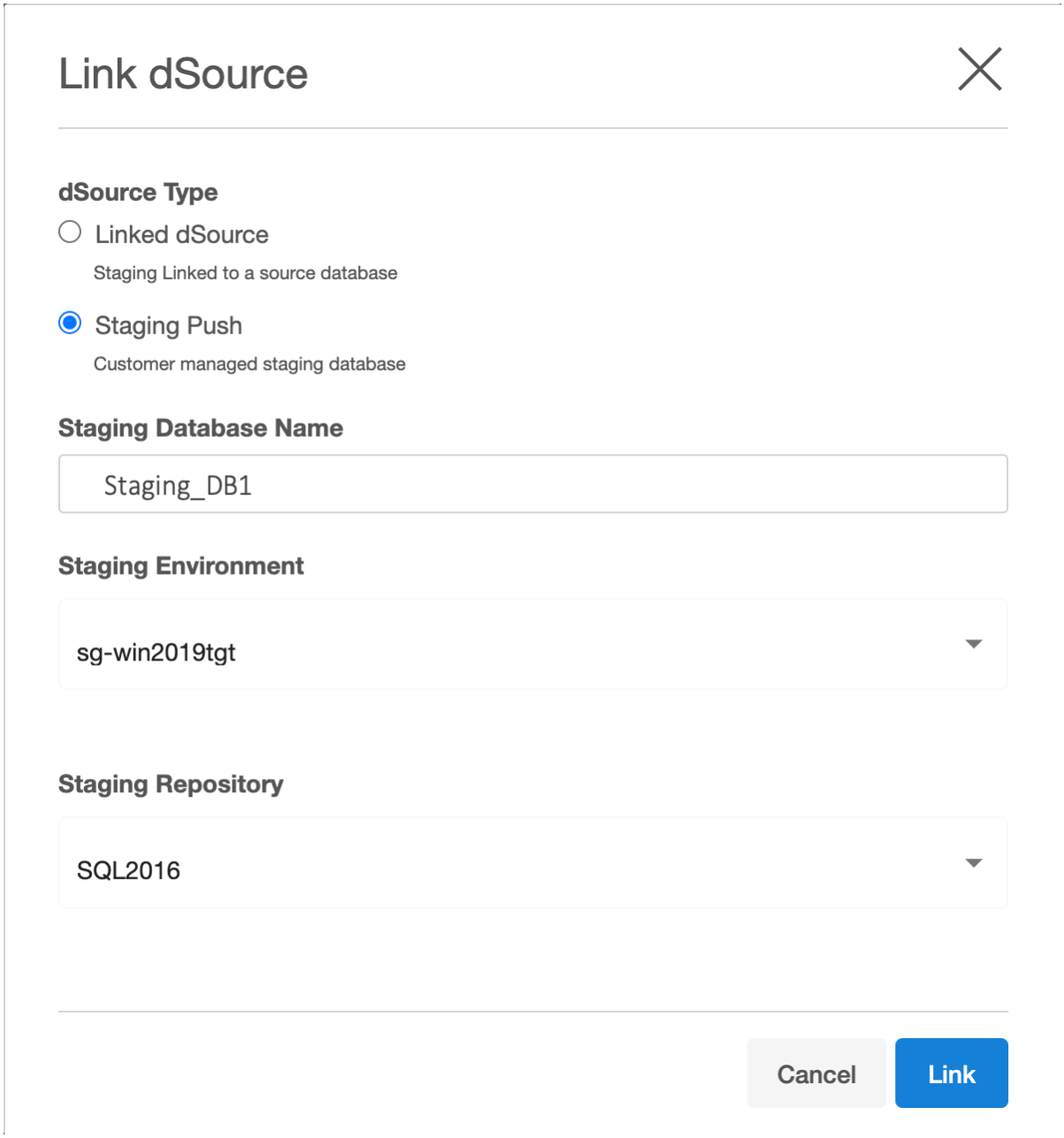
- A detached dSources can still be used to provision a virtual database (VDB).
- You can re-link the staging push dSource with a different staging database name than before. In that case, the staging database will be created with the new name provided. However, DB file names will remain the same as before.
- Delphix Engine supports converting existing Linked dSource to Staging push dSource and vice versa.

Unlinking or detaching a dSource

1. Login to the **Delphix Admin** application.
2. Click **Manage**.
3. Select **My Datasets**.
4. Select the **database** you want to unlink.
5. From the Actions ... tab, click **Unlink.**
6. Click **Yes** to confirm.

Attaching a previously detached dSource

1. Login to the **Delphix Admin** application.
2. Click **Manage**.
3. Select **My Datasets**.
4. Select the **database** you want to link.
5. From the Actions ... tab, click **Link dSource**. Select **Staging Push** and enter **staging database Name**, select **Staging Environment** and **Staging Repository**.



6. Click **Link to confirm.**

Limitations

1. Restore backups fails on staging database due to mount issues. When a critical threshold is enforced due to Delphix storage and later resumed successfully, a transactional log will not be applicable on the staging database directly. A generic warning will be displayed for the resume job only for staging push dSources on the Delphix interface that after the mount, the user might face issues in restoring a transaction log. SQL server error when restoring the Tlog after resume:
   `Msg 4320, Level 16, State 3, Line 13`
   The file " `SourceDB4_log` " was not fully restored by a database or file restore. The entire file must be successfully restored before applying this backup set.

`Msg 3013, Level 16, State 1, Line 13`

RESTORE LOG is terminating abnormally.

RESOLUTION: Full or differential backup will be successfully restored.

2. Attach operation fails during the Enable operation.
   If the attach operation fails for staging push dSource, any exception is consumed and the enable operation is marked as a success.
   For example, if DB files were changed for the staging database but a snapshot was not taken on the Delphix engine and disable was performed directly, the enable will succeed with an attach failure that will be shown as a warning. The previous DB files will be available on the mount path.
   RESOLUTION: The customer can apply a full restore at this point to bring the database in restoring mode before trying to run a sync.
3. Point-in-Time provisioning is not supported.
4. The sub-directory structure for DB files is not supported.
   For example, If a staging database is restored with DB files inside another folder within the DB directory (`C:\Program Files\Delphix\DelphixConnector\ec2197b2-e0c6-48d2-bd14-265e6fa9b5ab-staging-1\DATA\db\folder1\sp.mdf, C:\Program Files\Delphix\DelphixConnector\ec2197b2-e0c6-48d2-bd14-265e6fa9b5ab-staging-1\DATA\db\folder1\folder2\sp_log.ldf`), then operations such as to enable, relink, VDB provisioning, and export will fail.
5. The validated sync process is not supported for Staging Push dSources.
6. Staging database states other than Restoring such as ReadOnly, StandBy, and Online are not supported. If a database is in any of these unsupported states, dSource will go into the Cannot Monitor state on the Delphix Engine interface, and Sync operation will fail.

Data management settings for SQL server data sources

Each dSource has its own data management settings, which can be configured during the linking workflow as well as in the configuration page for that dSource.

You can configure data management settings to improve overall performance and match the needs of your specific server and data environment.

The following settings are available for SQL Server data sources:

Setting	**Explanation**
Managed Backups	When enabled, the Delphix engine will take full backups of the database, per the dSource's SnapSync policy, and validated sync will be disabled. Existing backups cannot be used to synchronize the dSource when backups are managed by Delphix.
Recovery Model	The current recovery model of the source database. Three recovery models exist simple, full, and bulk-logged. This is not configurable from Delphix but can be changed within the source's database settings.
Initial Load	1. Delphix will take a copy-only full backup of your source database. 2. Use the most recent full or differential backup (default). 3. Use a specific full or differential backup.
Backup Paths	These are the locations where Delphix will be looking for backups for ingestion. Select Autodiscover to have the Delphix Engine automatically locate the backups by querying the msdb database in the SQL instance. Otherwise, for each path, please specify the top level of a particular backup path since the Delphix Engine will recursively search for the backup file in all subdirectories beneath the specified path.
Validated Sync Mode	Determines the types of backups validated sync will use to generate snapshots. 1. Transaction log backups. a. LogSync adds log files from the source database to the dSource, allowing you to provision a virtual database (VDB) from a specific point in time or LSN for SQL Server databases. 2. Full or Differential backups. 3. Full backups. 4. None.
Staging Environment	This environment will host the staging database used for validated sync.

Setting	Explanation
Repository	A repository is a container for the SourceConfigs objects. Each environment can contain any number of repositories, and repositories can contain any number of source configurations. A repository typically corresponds to a database installation. Whereas source configurations typically correspond to the databases.
Encryption Key	The encryption key to be used when restoring encrypted backups. If the source database is backed up using LiteSpeed or RedGate with password-protected encryption, you can supply the encryption key that the Delphix Engine should use to restore those backups.
NetBackup Ingestion	Enables ingestion from a NetBackup source.
Commvault Ingestion	Enables ingestion from a Commvault source.

Additional SQL Server dSource topics

This section covers the following topics:

- Changing the staging target environment for a SQL Server dSource
- Upgrading a dSource after a SQL Server upgrade
- Linking a dSource from a commvault SQL Server backup
- Linking a dSource from a netBackup SQL Server backup
- Restoring SQL backups stored in Azure cloud storage
- Working with SQL Server snapshots
- Detaching and re-attaching SQL Server dSources

Changing the staging target environment for a SQL Server dSource

This topic describes how to change the staging target environment for a SQL Server dSource.

Prerequisites

The dSource for the staging database must be disabled before you can change the staging target environment. Follow the steps in *Enabling and Disabling SQL Server dSources* in Managing Data Sources and Syncing Data to disable the dSource.

Procedure

1. Click **Manage**.
2. Select **Datasets**.
3. Select the **dSource** for which you want to change the staging target environment.
4. Click the **Configuration** tab to view the **Staging Environment**.
5. Click the **Pencil** icon next to **Staging Environment**.
6. Edit the target server and the SQL Server instance on the server to use for staging.
7. Click the **Check** icon to save your changes.

Upgrading a dSource after a SQL Server upgrade

This topic describes how to upgrade dSources after a SQL Server database upgrade.

There are two ways to upgrade a Source database:

- Perform an Upgrade installation of SQL Server, upgrading the Source and Staging SQL Server instances in-place
- Perform a fresh installation of the new SQL Server version, and migrating the databases to the new instance

The steps required to support this in the Delphix Engine are different depending on method chosen. The required steps are outlined in the sections below.

In-place SQL server upgrade

1. **Disable** all dSources on the instance being upgraded
2. Following Microsoft's procedures, perform an upgrade of the Source SQL Server instance
3. Following Microsoft's procedures, perform an upgrade of the Staging SQL Server instance to the same version as the Source
4. Refresh the Source and Staging environments in the GUI
5. **Enable** the dSources being upgraded

Migrate databases to newer instance

Prerequisites for database migration

- Record the configuration data for each dSource being upgraded, including the Database User, Database Password (if applicable), and Validated Sync configuration. This will be needed to re-link the dSource.
- Following Microsoft's procedures, install a new Source SQL Server instance with the new SQL Server version.
- Ensure that a Staging SQL Server instance is available, running the same SQL Server version as the new Source SQL Server instance.
- **Add** or **Refresh** the Environment containing the new SQL Server instance(s), using the steps in Adding a SQL Server Source Environment and Adding a SQL Server Standalone Target Environment.

> ⓘ The refresh/rediscover operations do not affect the operations of any dSources or VDBs on the environment.

Migration steps

1. Navigate to the **Manage** → **Datasets** screen.
2. For each dSource being upgraded:
    a. Select **Unlink dSource** from the **Actions menu (...)**
3. Following Microsoft's procedures, migrate the Source database to the new Source SQL Server instance.
4. Commence a new **Full** database backup of each upgraded Source database
5. From the **Manage** → **Environments** screen, **Refresh** the new Source Environment. The migrated databases should be detected and visible from the Environment's **Databases** tab.
6. Navigate to the **Manage** → **Datasets** screen.
7. For each dSource that was upgraded:
    a. Select **Link dSource** from the **Actions menu (...)**
    b. Locate the upgraded Source database using the Source Environment, **Installation and Database** drop-down boxes
    c. Select a compatible Staging Environment and Staging Repository
    d. Configure the **Database Authentication** using values that are appropriate for the new Source database
    e. Click **Link** to begin linking the dSource

f. Reconfigure any dSource settings using the dataset's **Configuration** tab
g. Use the **Snapshot** button to take a snapshot using the upgraded database backup

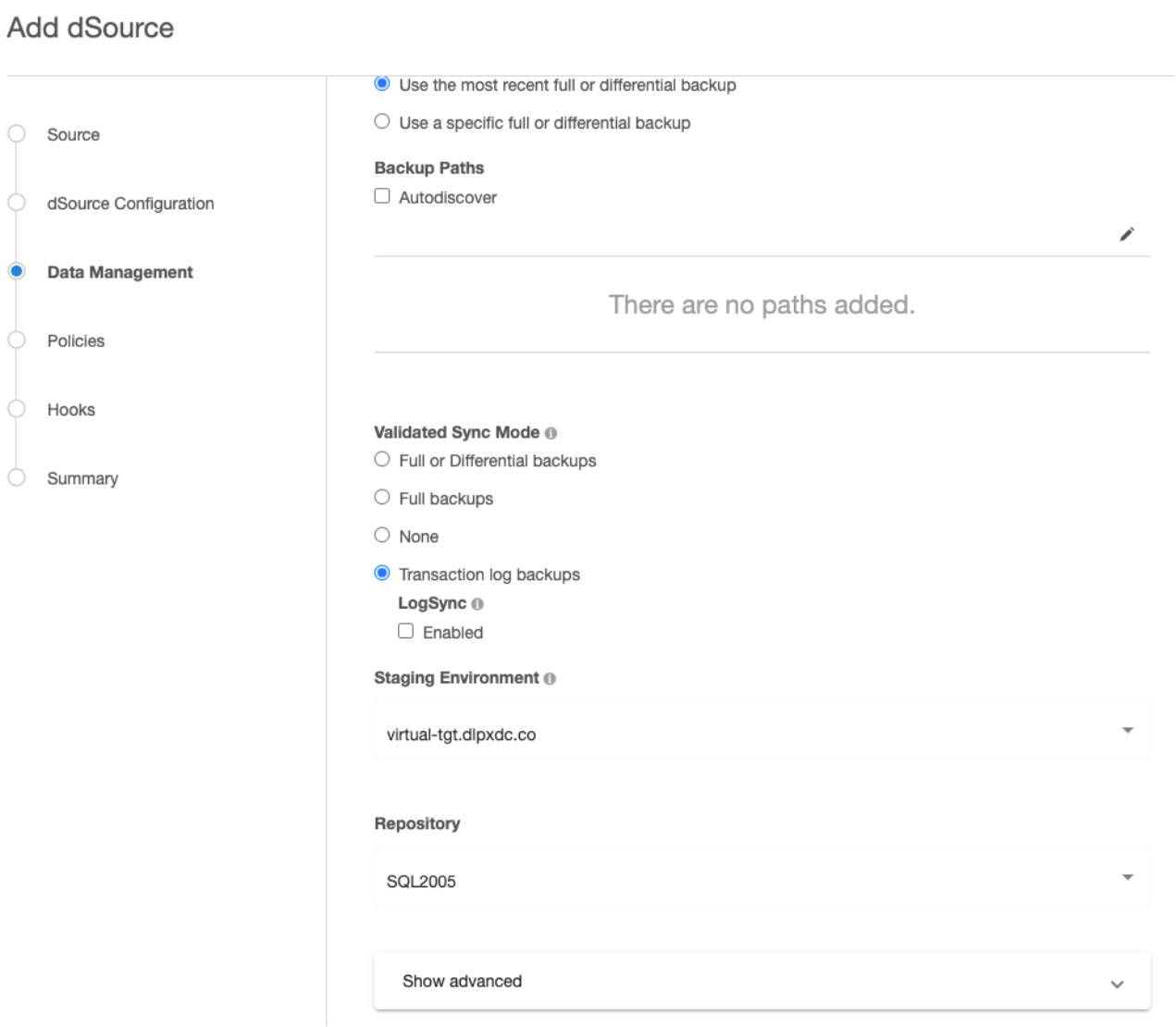
Linking a dSource from a commvault SQL Server backup

Customer requirements:

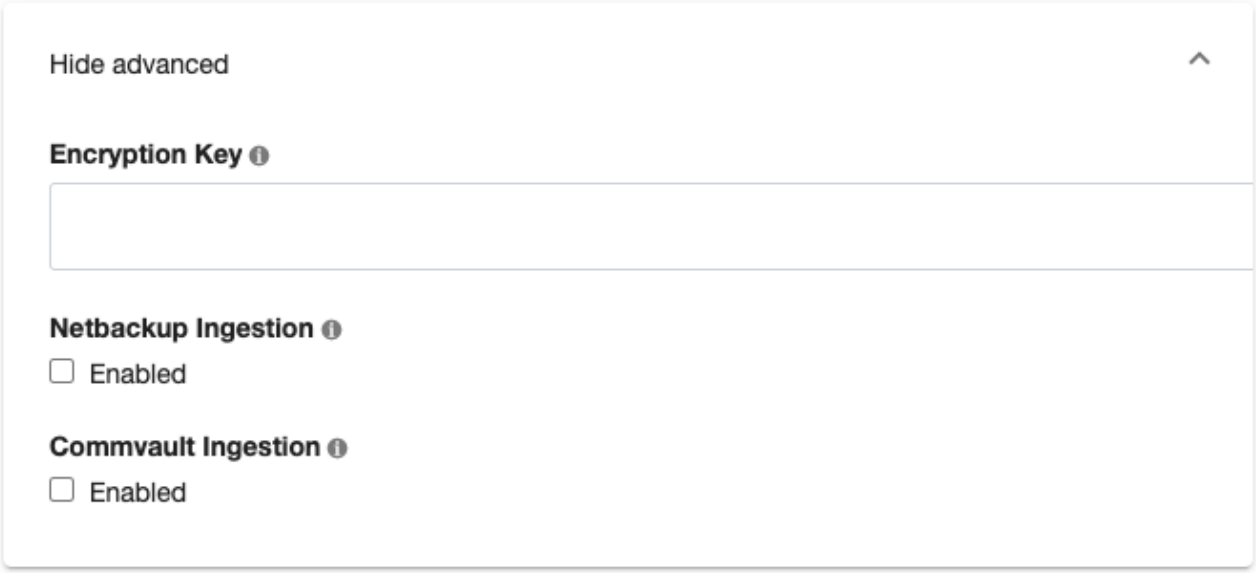
- Delphix currently supports Commvault v11. The version of Commvault SQL Agent on the staging environment must be the same as that on the source.
- The TCP port 8415 must be open from the staging host to the Delphix Engine.
- If the dSource is backed up with Commvault, the source and staging environments must each have the SQL Agent installed.
- Both SQL Agents (on source and staging) should be registered with Commvault Server mentioned during linking.
- The install path's 'Base' directory of Commvault SQL Server Agent on the staging host, must be part of the PATH environment variable as we need to access the Commvault CLI. This is typically located at <Commvault install path>\Base.
    - There are two PATH environment variables. One is for the current logged in user and the other is a global System variable. Since the Delphix Connector runs under the "Network Service" account and spawns the Commvault commands, changes need to be done to the System variable as well.
    - After the Commvault binaries are added to both the system and user PATH environment variables, the Delphix Connector service must be restarted in order for the new process to reflect the changes made to the PATH environment variable.
    - After making the changes to the PATH, login as the Delphix operating system user and try running "qlogin -sso -gt" to confirm the user can execute Commvault commands and authenticate to the Commvault server.
- Since SSO is used from Commvault CLI to login to the CommServe server from the staging environment, SSO should be enabled.
- Active Directory domain should be configured and configured staging environment user should be given required permissions to restore the database on staging client.
- In the CommServe server, the staging client should be configured with a user who has required permissions to restore the database on the staging environment. User account configuration can be done as explained in Commvault documentation at User Account Configuration for the SQL Server Agent.
- Only transaction logs, incremental, and database full backups are currently supported.

Linking with commvault backups:

To link a dSource and use Commvault, follow the steps to add the environments as earlier, and make sure all the requirements listed above are met. Going through the linking wizard on the **Data Management** page, select **Show advanced**.



Enable **Commvault Ingestion** and input the CommServe hostname and, source and staging client names.



We validate whether the source and staging clients exist in the CommServe server and the Commvault SQL Agent on staging client is registered to the CommServe server. To run the validation commands following login command with SSO option is used to login into CommServe Server:

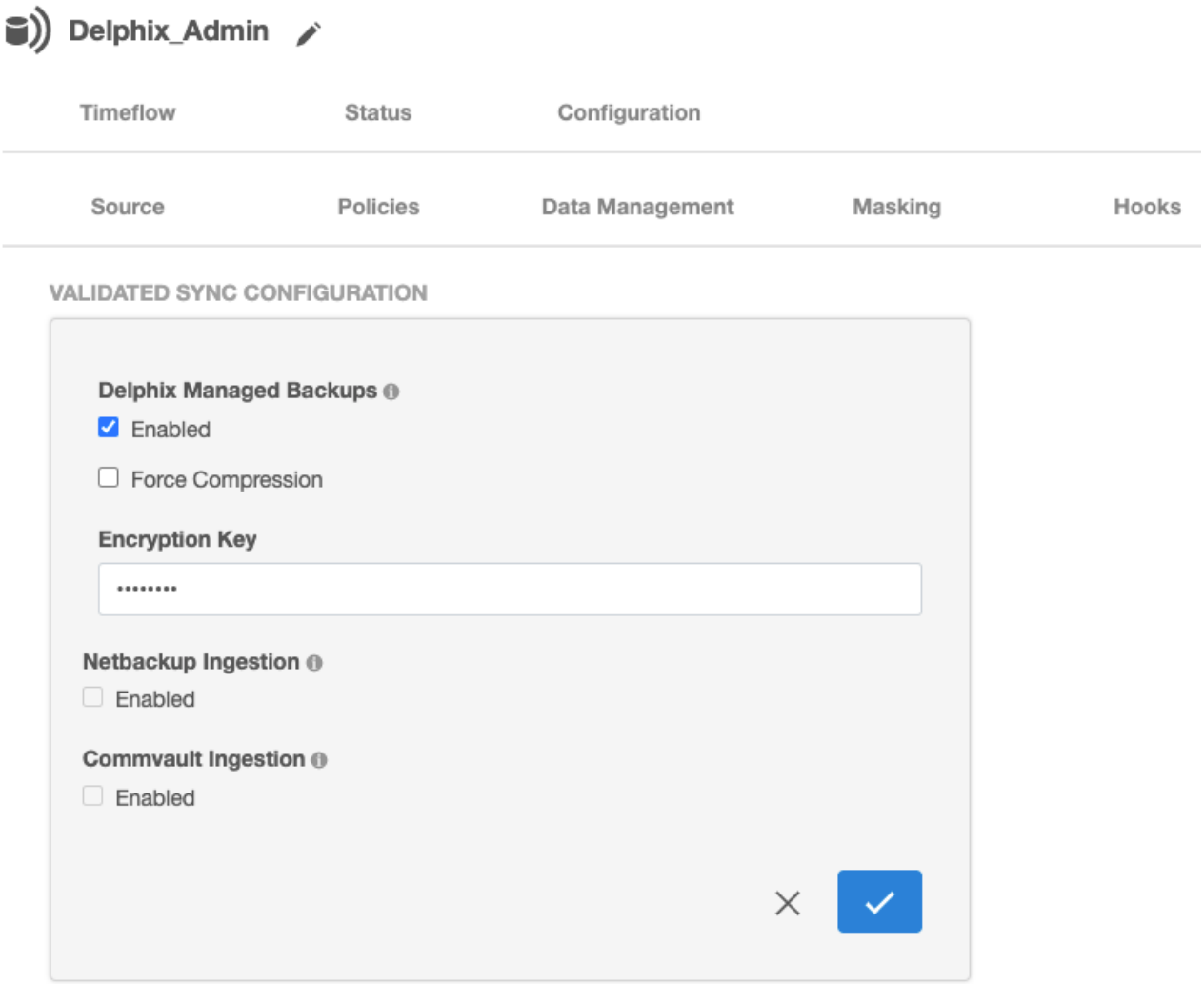
```
qlogin -sso -gt
```

LogSync for SQL server dSources

Logsync (point-in-time provisioning) is currently not supported for Commvault transaction logs. However, LogSync can still be enabled if Commvault ingestion is enabled. LogSync for backups taken with other backup providers that support LogSync will work as before.

Enabling commvault for previously created dSources

On an already created dSource go to **Configure** > **Data Management** and click edit (pencil button). Enable Commvault Ingestion and add all the configuration as necessary.



Linking an availability group database with commvault

Linking with an Availability Group (AG) source works similar to as described above, however, the source client name provided in the Commvault configuration should be the MSSQL AG Client name as in the CommServe server.

General notes/troubleshooting

- Commvault backups are not on local storage and therefore while ingesting backups taken using Commvault, Backup Paths provided in configuration will be ignored.

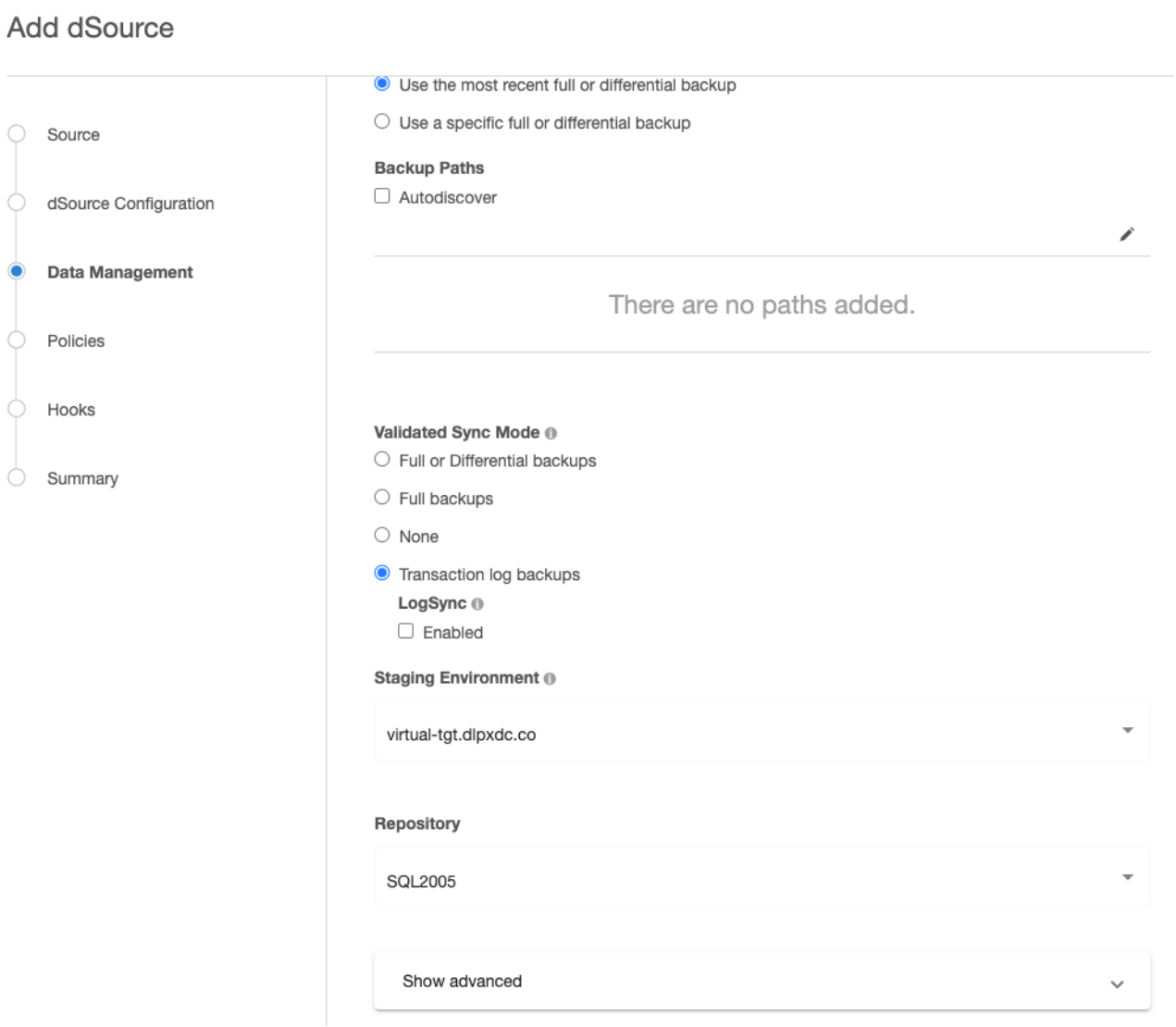
Linking a dSource from a netBackup SQL Server backup

Customer requirements

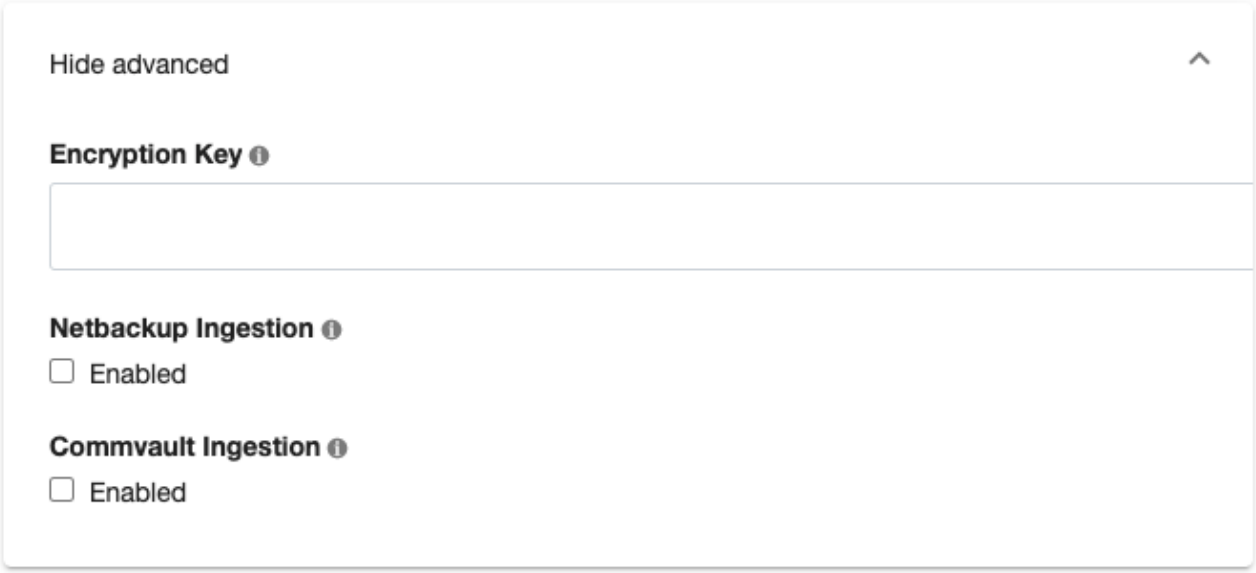
- The version of NetBackup client on the staging environment must be the same as that on the source.
- The TCP port 8415 must be open from the staging host to the Delphix Engine
- Backups must be taken via an MS-SQL-Server type policy with an INSTANCE client list type. (value is 15, and 1 respectively) Only transaction logs, incrementals, and database full backups are currently supported.
- The master server and source client servers' clocks must be within a minute of each other (timezones can be different).
- Any backups that Delphix needs to ingest for a dSource must be taken to one NetBackup master server using one NetBackup SQL Server client. A multiple master server setup for one dSource is not supported.
- If the dSource is backed up with NetBackup, the source and staging environments must each have the NetBackup client installed.
- Both clients (on source and staging) during installation must be setup with the master server and the source and staging instances must be registered.
- The install path’s bin directory of the SQL Server NetBackup client on the staging host must be part of the system PATH as we need access to dbbackex.exe and bplist.exe. This is typically located at
- If you have modified the PATH, then please restart the Delphix connector service otherwise the linking process won't be able to pick the changed system PATH environment variable.
- Configure redirected restores on the master server between the source NetBackup Client and the staging NetBackup Client
    - https://www.veritas.com/support/en_US/doc/17221771-126559330-0/v113535700-126559330

Linking with netBackup backups

To link a dSource and use NetBackup, follow the steps to add the environments as before, and make sure all requirements listed above are met. Going through the linking wizard on the **Data Management** page, select **Show advanced**.



Enable **Netbackup Ingestion** and input the master and source client names.



We use these names to query bplist using these options:

When validating the master and client servers we expect that at least one NetBackup MS-SQL-Server type backup had been taken with this pair in the last two days. Validation will fail otherwise.

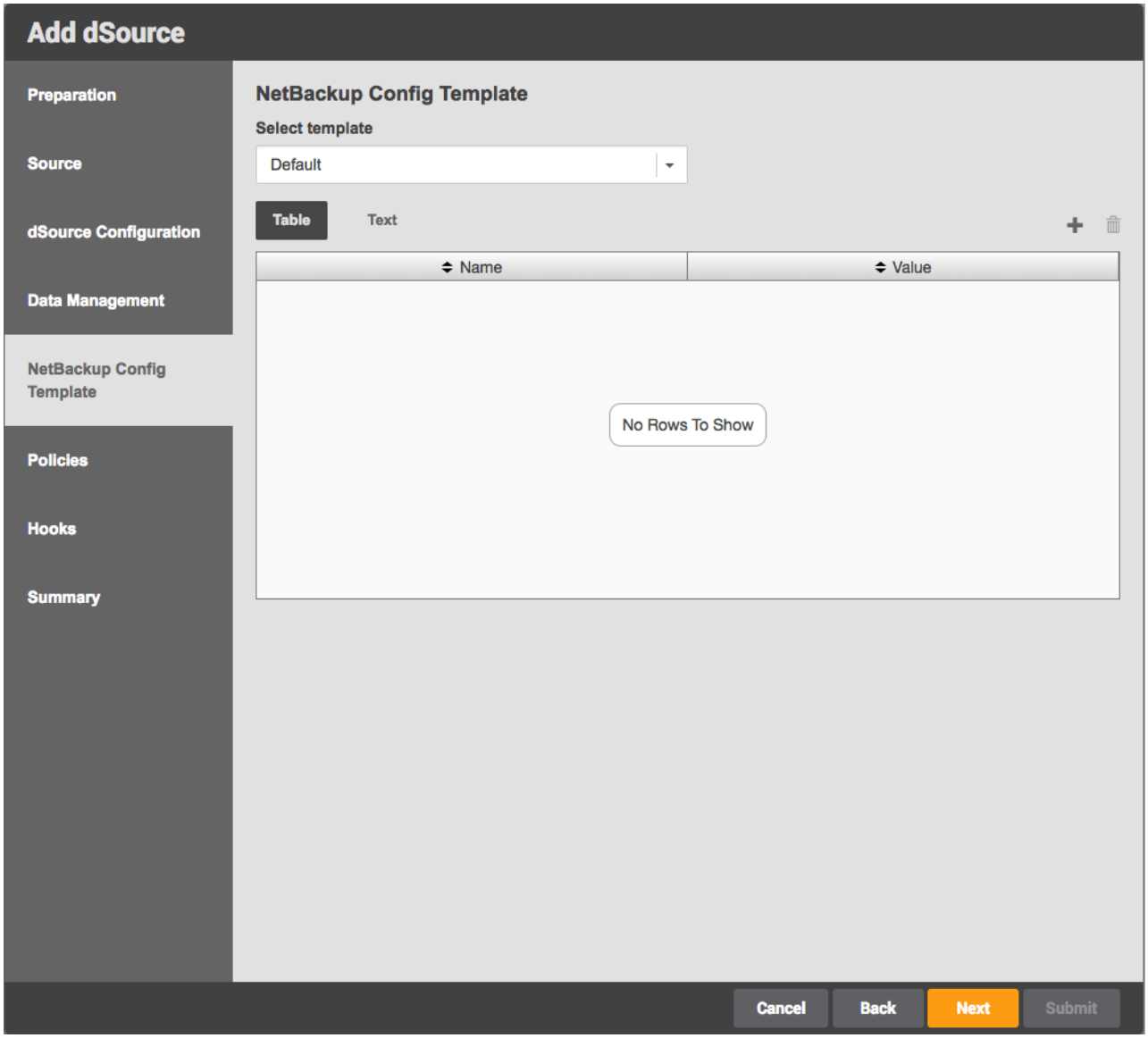
Using NetBackup config parameters or templates

When restoring we create a batch file based on information we find on the backup. Batch files can be customized with non-default and additional options using config templates.

https://www.veritas.com/content/support/en_US/doc/123947690-126579517-0/id-SF930853806-126579517 explains which parameters in the batch file can be edited outside of these blacklisted keywords:

- ALTCLIENT
- BROWSECLIENT
- DATABASE
- DUMPOPTION
- ENABLESERVICEBROKER
- ENDOPER
- MOVE
- NBIMAGE
- NBSERVER
- OBJECTNAME
- OBJECTYPE
- RECOVERED STATE
- RESTOREBEFOREMARK
- RESTOREBEFOREMARK AFTERTIME
- RESTOREOPTION
- RESTOREPAGES
- PARTIAL
- PAGE Any key that is used outside of what is documented may cause the restore to fail with unknown errors.

If the **Configure NetBackup Config Template** is checked, the page after **Data Management** becomes **NetBackup Config Template**.



This page allows this dSource's specific config parameters to be updated, either by adding rows to the table or inputting them as text input (key=value).

For config params/templates, the 'value' will be injected into the script as is so if the value of the parameter needs to be a String, quotes should be included with 'value'.

If creating a new set of config parameters while linking each dSource seems unnecessary, a config template can be specified. However, that is only possible via the CLI. To do that, log onto the CLI using the Delphix admin user/password and go to **database** > **templates**. Make sure to create the new template with sourceType set to MSSqlLinkedSource.

Example:

Then when linking, select the template created. (In this screenshot it would be **NetBackup**) If any edits are made to a selected template while linking, this will create a new set of config parameters for the specific dSource and will

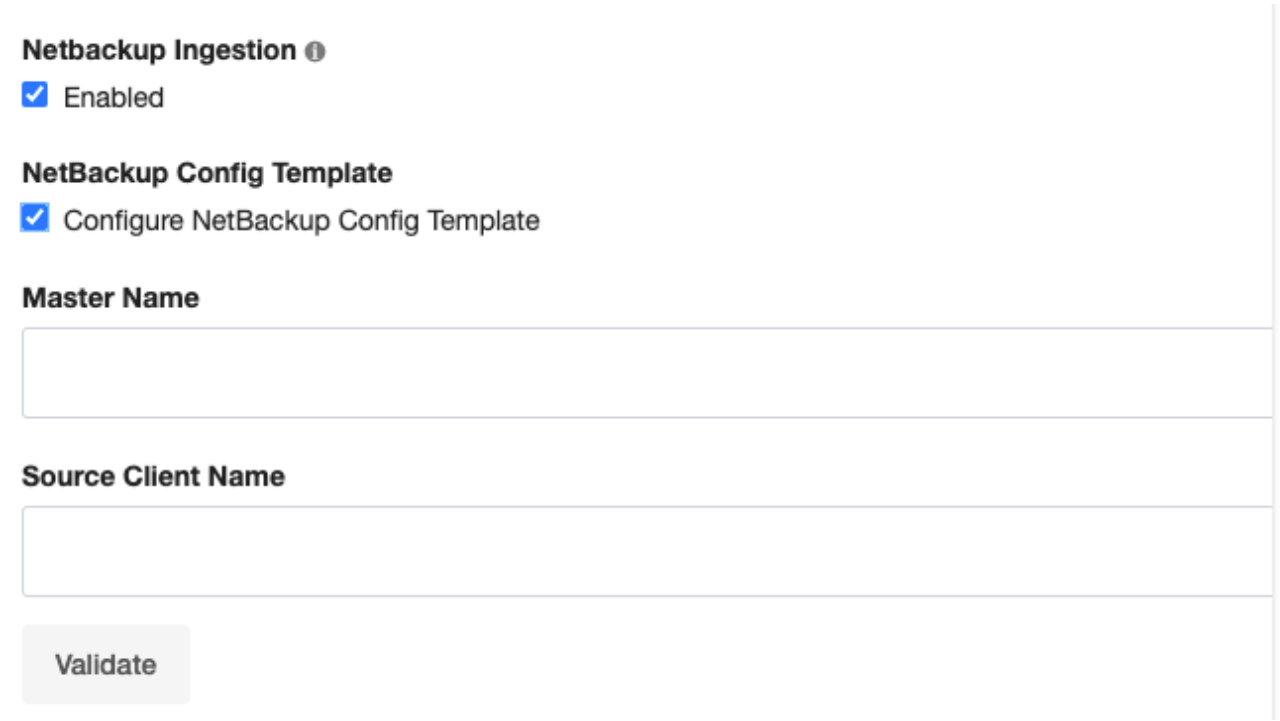
not edit the actual template. Config parameters added to a specific dSource will be ignored if the dSource has a config template selected.

LogSync for SQL server dSources

LogSync can still be enabled if NetBackup ingestion is enabled however Point-in-Time provisioning is currently not supported for NetBackup transaction logs. LogSync for backups taken with other backup providers that support LogSync will work as before.

Enabling NetBackup for previously created dSources

On an already created dSource go to **Configure** > **Data Management** and click edit (pencil button). Enable NetBackup Ingestion and add all the configuration as necessary.

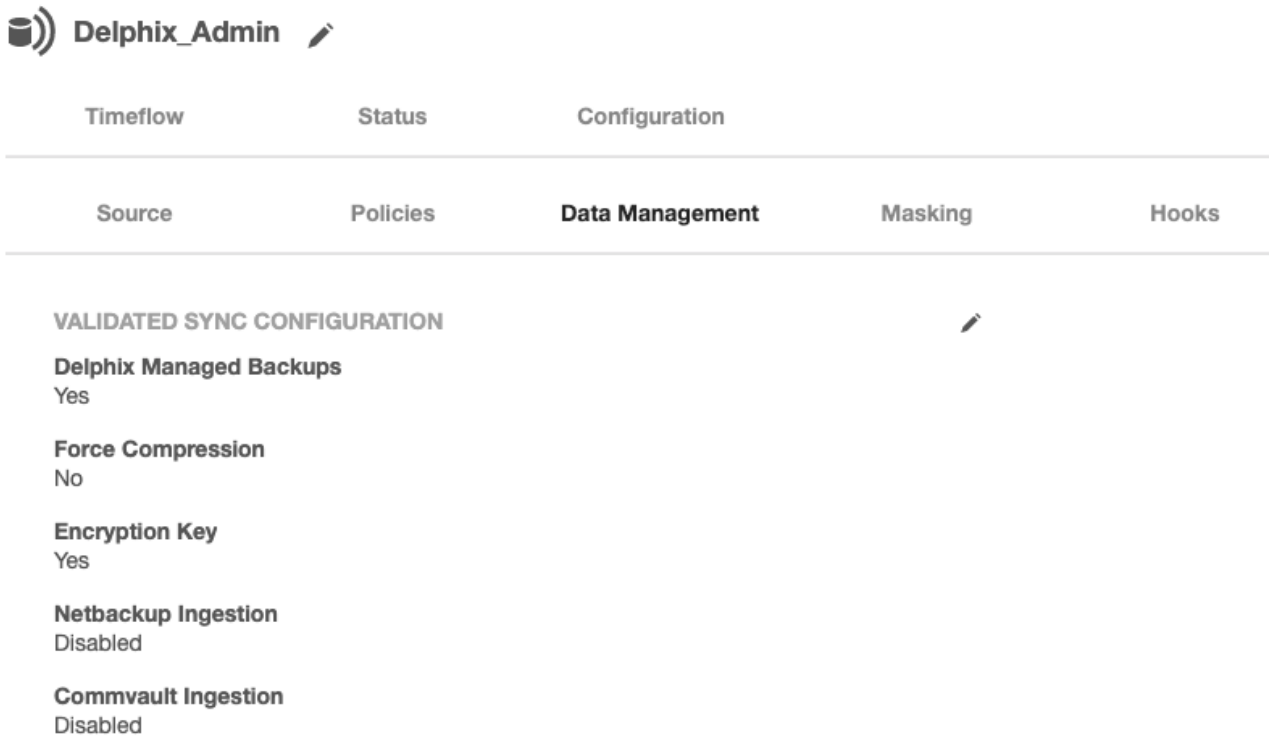


If **NetBackup Config Template** is set to **Default** then the params will be empty. Go to the CLI to update the config params.

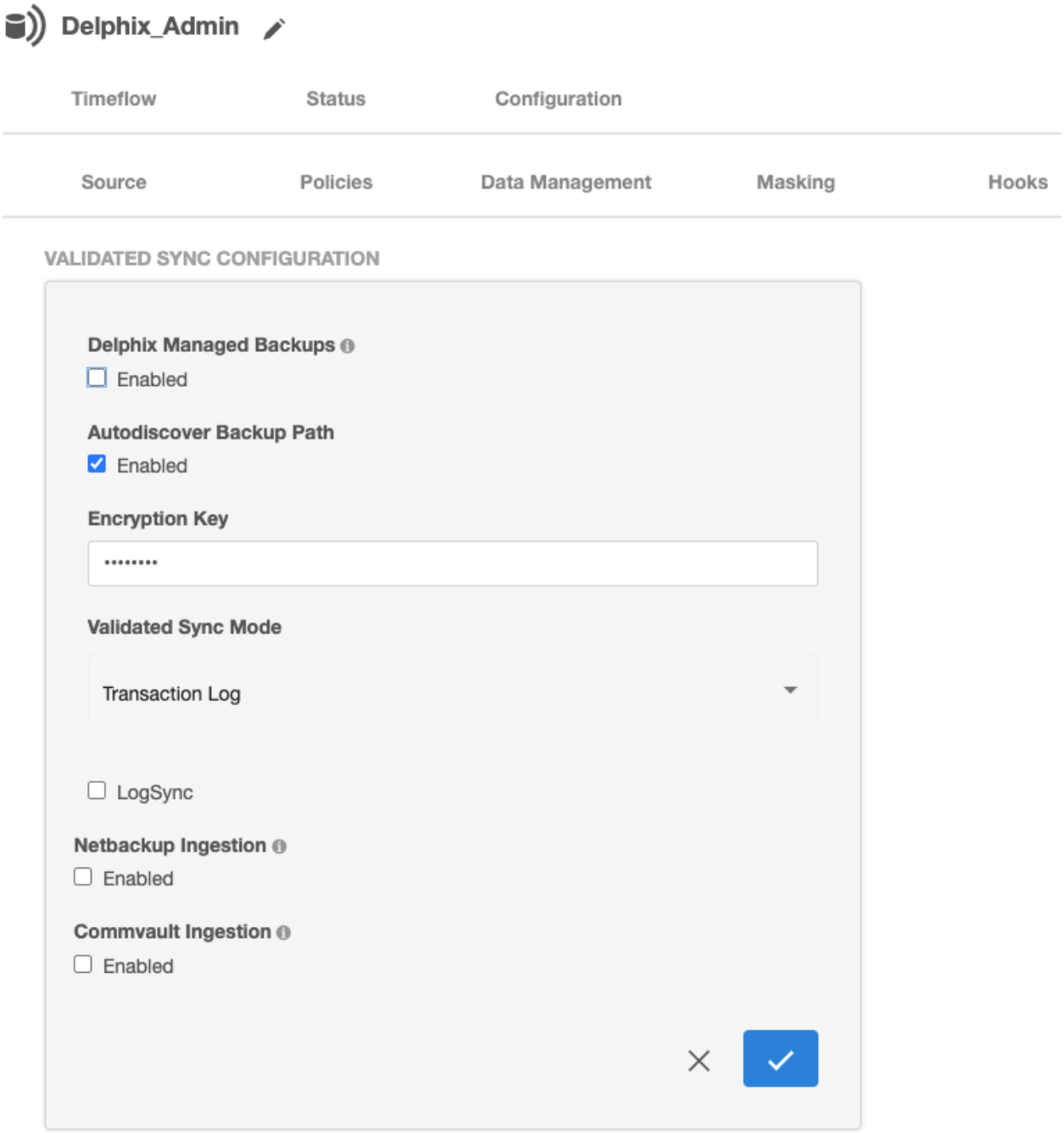
```
Password:
ip-10-110-238-136> cd source
ip-10-110-238-136 source> select Biscuit
ip-10-110-238-136 source 'Biscuit'> update
ip-10-110-238-136 source 'Biscuit' update *> edit mssqlNetbackupConfig
ip-10-110-238-136 source 'Biscuit' update mssqlNetbackupConfig *> set configParams.BLOCK_SIZE=6
ip-10-110-238-136 source 'Biscuit' update mssqlNetbackupConfig *> ls
Properties
    type: MSSqlNetbackupConfig
    configParams:
        BLOCK_SIZE: 6 (*)
    configTemplate: (unset)
    masterName: nbu-80-master.delphix.com
    sourceClientName: ln-win2012src.dlpxdc.co
ip-10-110-238-136 source 'Biscuit' update mssqlNetbackupConfig *> commit
```

Like above, to create a config template, create the template first so that during this update that specific template can be selected in the drop-down.

Delphix Managed Backups must be disabled before NetBackup can be enabled.
To disable Delphix Managed Backups for a specific dSource, select that specific dataset and go to **Data Management** tab under the **Configuration** tab.



Click the edit button and uncheck this feature. Then update all new inputs.



Linking an availability group database with NetBackup

Linking with an AG source works similarly however the source client name inputted to the NetBackup config should be the windows cluster name rather than the client name that takes the backup.

General notes/troubleshooting

- If a restore fails and the staging environment's NetBackup client job log shows no attempted restores, it is likely that you need to log onto the environment using the staging user provided to Delphix. This article

explains that the user profile is created on the first login and that NetBackup requires it to restore. https://www.veritas.com/support/en_US/article.100032299

- Through observation, we've noticed that the block size of the restore batch file must be equal to or less than what was used when backing up the dump. We default to the blocksize to what the backup was taken with but the config param/template can include BLOCK_SIZE to change. We won't prevent this but it's up to the user to set it correctly. Editing this field is generally not recommended for this reason.
- NetBackup backups are not on local storage and therefore the Backup Paths will just be ignored.

Restoring SQL backups stored in Azure cloud storage

Need for supporting SQL backups stored in Azure cloud storage

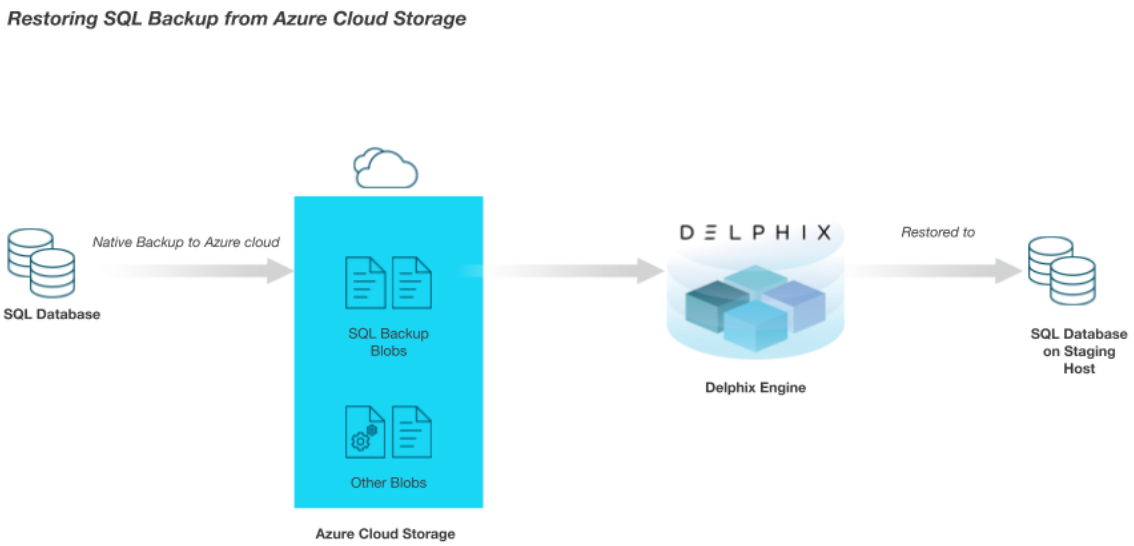
Currently, you can restore the SQL backups that are stored locally on the staging host, or on the network path, or on the backup servers with third-party vendors such as Commvault or Netbackup. At the same time, there are users that are having their native SQL backups on Azure Cloud Storage.

Delphix now supports restoring native SQL backups from Azure Cloud Storage. This enables users who are moving to Azure to use direct backups from the Azure Storage containers instead of third-party vendors.

These backups support the following.

- This is supported by SQL Server 2016 and above
- All backup modes
    - Full backups
    - Differential
    - Transaction log backups Logsync (point-in-time provisioning) is currently not supported. However, LogSync can still be enabled if Azure backup is being ingested. LogSync for SQL native backups that are present on Disk will work as before.
- Striped backups All backup files are to be entirely present on the Azure Cloud and no part of a backup should be present outside of Azure Cloud.
- Validated Sync
- Azure Storage authentication mode This backup solution uses a shared access signature token authentication method to authorize access to the blob data.

Workflow architecture



Workflow Steps

- **User takes backup**: You need to take the SQL native backup directly to the Azure Cloud using SQL Backup to URL. Performance can be improved by enabling COMPRESSION while taking the backup. If the backup size is large, it should be striped into multiple files.

```
BACKUP DATABASE [SourceDB4]
to URL = 'https://idea1201.blob.core.windows.net/ideal1201container/source.bak'
```

- **Authentication to authorize access to the blob data**:
  You must create a SQL Credential on the staging host for the following cases.
  - Create using SAS authentication token as Access Key authentication is currently not supported.
  - Create for each Azure container where the backup files are expected to be stored.
  - Create on each SQL instance where restore has to be performed.

```
CREATE CREDENTIAL [ https://idea1201.blob.core.windows.net/
ideal1201container] (this is the Azure Container URL)
WITH IDENTITY = 'SHARED ACCESS SIGNATURE',
SECRET =
'sv=2019-12-12&ss=bfqt&srt=c&sp=rwdlacupx&se=2021-01-19T14:18:04Z&st=2021-
01-19T06:18:04Z&spr=https&sig=DBlZnu0VQTaXUwY9IgBEqNbSk';
(provide the SAS token here)
```

- **Run queries on Staging host**: Delphix Engine uses this credential to run the following queries Restore, Restore Headeronly, and Restore Filelistonly on the staging host.
- Irrespective of whether auto-discovery is selected or not, if an Azure backup is being ingested, its location will be fetched from the `msdb.dbo.backupmediafamily` table on the source host and it will be restored on the staging host. Hence, it is necessary that the backup files are present on the same blob URL where the backup was originally taken. This is different from the usual native backups to the disk.

ⓘ In SQL native disk backup,
- If auto-discovery is on, the backup path is fetched from `msdb.dbo.backupmediafamily` table on the source host.
- If auto-discovery is off, Delphix Engine uses the custom path(s) specified by the user.

If any native Azure backup is found, Delphix Engine will always try to restore it.
- This is similar to the user taking backups using multiple backup vendors.
- In that case, Delphix Engine tries to restore all backups, irrespective of the backup vendor.

Unsupported features

The following features and functionalities are currently not supported.

- Third-party backup vendors
- Point-in-time provisioning
- Access key authentication method for Azure backups
- Support for SQL Server versions below SQL 2016
- Moving backup files across Azure containers
- Moving a backup from disk to cloud
- Striped backups - Backup files that are partially present on Azure Cloud and partially on another device.
- Managed identity

Working with SQL Server snapshots

This section lists the steps to take a snapshot and delete the same.

Taking a snapshot creates a new snapshot entry in the Oracle dSource's Timeflow. You can use either **Snapshot (Default)** or **Snapshot with Parameters** option for taking the snapshot.

Snapshot (Default)

Perform the following steps to take a snapshot:

1. Login to the **Delphix Management** application.
2. Click **Manage** and select **Datasets** from the dropdown list.
3. Select the dSource you want to Snapshot.
4. Click the **Camera** icon. Alternatively, click the arrow next to the Camera icon and select **Snapshot (default)**.
5. From the Snapshot dialog box, select **Yes**.
6. Navigate to the **Timeflow** tab and click **View: All snapshots** to verify the Snapshot you just created.
7. To delete the snapshot, select the snapshot you just created, and from the Actions menu (...), select **Delete Snapshot**.
8. From the **Delete Snapshot** dialog box, select **Delete**.
9. Navigate to the **Timeflow** tab and click **View: All snapshots** to verify the snapshot you just deleted.

Snapshot with Parameters

Perform the following steps to take a snapshot:

1. Login to the **Delphix Management** application.
2. Click **Manage** and select **Datasets** from the dropdown list.
3. Select the dSource you want to Snapshot.
4. Click the arrow next to the Camera icon and select the **Snapshot with Params...** option.
5. From the Snapshot dialog box, select one of the following:
    a. **Force Full Backup** - If you select this option, then the Delphix Engine will perform an incremental backup by default. You must select this option only when a full backup is required. Full and Incremental backups consume the same space on the Delphix Engine.
    b. **Double Sync** - Selecting this option will perform a SnapSync operation as normal. After the first SnapSync is successful, the Engine will immediately perform a second SnapSync without waiting for the Log Files required for the first SnapSync to be made consistent. This is most useful when performing the initial SnapSync (or when "Force Full Backup" is selected) on a very large database that would lead to a large number of archive logs being required to make the SnapSync consistent. Provisioning from a SnapSync that requires excessive recovery is typically time-consuming.
    c. **Do Not Resume** - If a failure is encountered during the initial SnapSync, the Delphix Engine can resume the SnapSync at a later date. This option will cause the engine to not resume, but rather to start the initial SnapSync over again.
6. Navigate to the **Timeflow** tab and click **View: All snapshots** to verify the Snapshot you just created.
7. To delete the snapshot, select the snapshot you just created, and from the Actions menu (...), select **Delete Snapshot**.
8. From the **Delete Snapshot** dialog box, select **Delete**.
9. Navigate to the **Timeflow** tab and click **View: All snapshots** to verify the snapshot you just deleted.

Detaching and re-attaching SQL Server dSources

Each dSource contains metadata that associates it with the source database, as well as the data it has ingested from the source database in the form of snapshots up to that point. It is possible to detach, or unlink, a dSource from its source database. This breaks the association with the source database without affecting the data within the Delphix Engine. Detached dSources and their source databases have these properties:

- A detached dSources can still be used to provision a virtual database (VDB).
- You can re-link the source database as a different dSource.

Prerequisites

A dSource can only be attached to a new data source once it has been unlinked.

Detaching a dSource

1. Login to the **Delphix Management** application as a user with **OWNER** privileges on the dSource, group, or domain.
2. Click **Manage**.
3. Select **Databases**.
4. Select the **database** you want to unlink or delete.
5. From the **Actions** menu (...) select **Unlink.** A warning message will appear.
6. Click **Unlink** to confirm.

> Rebuilding Source Databases and Using VDBs
> In situations where you want to rebuild a source database but retain the existing dSource, you will need to detach the original dSource and create a new one from the rebuilt data source.
>
> 1. Detach the dSource as described in the procedure on this page.
> 2. You cannot attach a dSource with the same name as a dSource that is already attached. If you intend to give the new dSource the same name as the original one, rename the detached dSource.
>   a. At the top of the **Configuration** tab, next to the dSource's name, click the **Edit** (pencil) icon.
>   b. After renaming the dSource, click the green **checkmark**.
> 3. Create the new dSource from the rebuilt database.
>
> You will now be able to provision VDBs from both the detached dSource and the newly created one, but the detached dSource will only represent the state of the source database prior to being detached.

Attaching a previously detached dSource

You can only re-attach databases that represent the same physical database.

1. Login to the **Delphix Management** application as a user with **OWNER** privileges on the dSource, group, or domain.
2. Click **Manage**.
3. Select **Databases**.
4. Select the database you want to re-attach.
5. From the **Actions** menu (...) select **Link dSource**.
6. Select the environment and database that represents the source to which you would like to re-attach.
7. Select an appropriate staging environment, and choose the credentials that should be used to authenticate to the source database.
8. Click **Link** to link the dSource to the new source database.
9. Once linking is complete, use the **Configuration > Data Managemen**t tab to set up an appropriate **Validated Sync** configuration.

Attaching a previously detached dSource using CLI

You can only re-attach databases that represent the same physical database.

1. Login to the **Delphix CLI** as delphix_admin or a user with OWNER privileges on the dSource, group, or domain.
2. Select the dSource by name using `database select <dSource Name>.` .
3. Run the `attachSource` command.
4. Set the source config you want to attach to, using `set source.config=<Source Database Unique Name>` . Source configs are named by their database unique name.
5. Set any other source configuration operations as you would for a normal link operation.
6. Run the `commit` command.

Provisioning and managing virtual databases with SQL Server

Virtual databases are a key data management concept for Delphix. In order to create or provision a virtual database, you will need a linked dSource from a source host and a compatible target environment. From a dSource, you can select a snapshot or point in time to create a VDB. SQL Server VDBs each have their own configuration settings.

This section covers the following topics:

Overview of SQL Server virtual databases

Virtual databases are a key data management concept for Delphix, explained in Provisioning and Managing Virtual Databases. In order to create or provision a virtual database, you will need a linked dSource from a source host and a compatible target environment, as described in the overview for Managing Environments and Hosts and Overview of Requirements for SQL Server.

From a dSource, you can select a snapshot or point in time to create a VDB. SQL Server VDBs each have their own configuration settings as described in Configuration Settings for SQL Server Virtual Databases below. This document describes the steps to provisioning VDBs with SQL Server.

Procedure

1. In the Datasets panel on the left-hand side, click the group containing the dSource or VDB from which you want to provision and select the dSource or VDB from the provided list.
2. From the TimeFlow tab, select a snapshot to provision from. To provision from a specific point in time from dSources with LogSync enabled, use the open LogSync button.
3. Click to open the Provision VDB wizard, and select a compatible Target Environment for the new SQL Server VDB
4. On the Target Configuration page, you may customize the VDB. For a list of available configuration options, see Configuration Settings for SQL Server Virtual Databases below.
5. Select a Snapshot Policy for the VDB.
6. If the VDB should be masked during provisioning, enable Masked Provisioning by selecting an option on the Masking page
7. Enter any operations that should be run on the Hooks page.
8. Review the VDB Configuration and Summary, and then click Submit.

When provisioning starts, you can review the progress of the job by selecting the VDB and clicking on the Status tab, or by selecting System and viewing the Jobs page.

Alternatively, you can see this in the Actions Sidebar. When provisioning is complete, the VDB will be included in the group you designated and listed in the Datasets panel.

Configuration settings for SQL server virtual databases

Each VDB has its own data management settings, found during the provisioning workflow as well as in the configuration page for that VDB. When you create a SQL ServerVDB, Delphix copies most configuration settings from the dSource and uses them to create the VDB. However, you can customize these with the following settings:

Setting	Explanation
Recovery Model	The current recovery model of the source database. By default, this value is set to SIMPLE. You must set it explicitly during provisioning.
Auto VDB Restart	Enabling this option will automatically restart this VDB whenever its target host is rebooted.
Change Data Capture (CDC)	Indicate whether this virtual source should be enabled for CDC.

Configuration settings for SQL Server virtual databases

Database configuration settings for a SQL Server virtual database can be provided as a key-value pair. Currently, there is support for the below-mentioned configuration parameter(s). These parameter(s) can be added in a configuration template and then applied to virtual databases.

The following parameters are available for SQL Server VDBs:

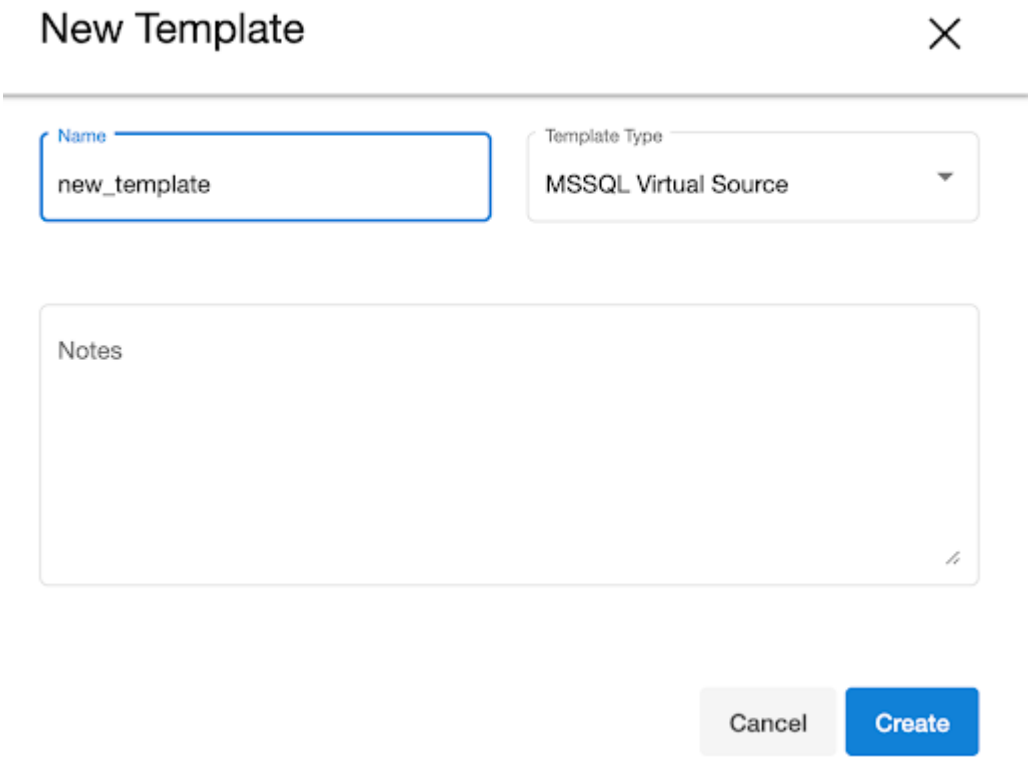
VDB Configuration Parameters	Value	Explanation
READ_COMMITED_SNAPSHOT	ON/OFF	Enables the READ_COMMITTED_SNAPSHOT isolation database option on the virtual database. By default, this option is set to OFF.

VDB config templates

A VDB config template is a list of database configuration parameter names and values that you can save on the Delphix Engine to use at a later time.

Creating a VDB config template via GUI

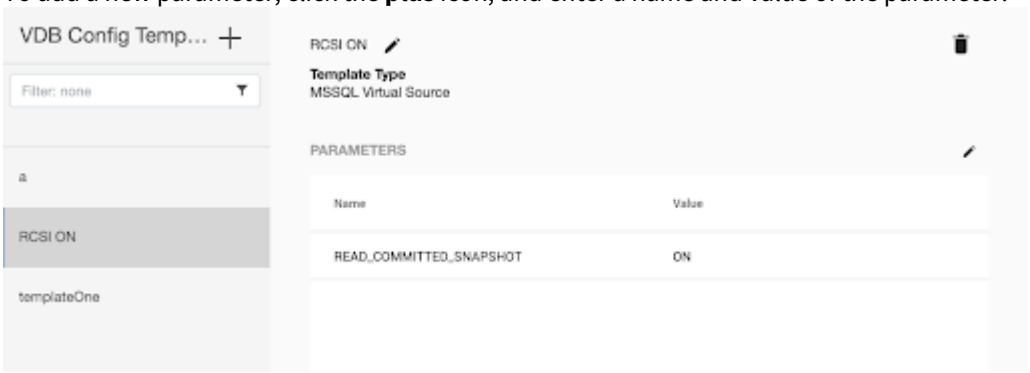
1. Log into the **Delphix Management** application as an engine administrator.
2. Click **Manage**.
3. Select **VDB Config Templates**.
4. Click the icon next to the **VDB Config Temp...** and select **New Template** to create a new template.
5. In the **New Template** dialog window, enter the name for the new template, the parameters that you want to provide, and select the template type from the available options.



6. Click **Create**.

Updating a VDB config template via GUI

1. Log into the **Delphix Management** application as an engine administrator.
2. Click **Manage**.
3. Select **VDB Config Templates**.
4. Select the template from the left-side pane that you need to update.
5. Click on the **pencil** icons next to the parameters to edit an existing VDB template.
6. To add a new parameter, click the **plus** icon, and enter a name and value of the parameter.



7. Click the button to save the changes or click the button to discard the changes that you made.

You can apply a VDB Config Template to a VDB during the provisioning process, which copies the values from the template. Any subsequent changes to the template will be reflected in the VDB when that VDB is refreshed/ rewinded. During provisioning, you can specify configuration parameters directly or copy them from a VDB Configuration Template. Once set, the Delphix Engine will use these parameters whenever the VDB is refreshed/ rewinded, even if you change the original template.

Provisioning a SQL Server VDB

This topic describes how to provision a virtual database (VDB) from a SQL Server dSource.

Prerequisites

- You must have already linked a dSource from a source database, as described in Linking a SQL Server dSource or have already created a VDB from which you want to provision another VDB.
- You must have already set up Windows target environments and installed the Delphix Connector on them, as described in Adding a SQL Server Standalone Target Environment.
- Make sure that you have the required privileges on the target environment.
- If you are provisioning to a different target environment than the one where the staging database has been set up, make sure that the two environments have compatible operating systems. For more information on the staging database and the validated sync process.
- If using Change Data Capture (CDC):
  - The SQL Server instance on which the VDB is being provisioned or exported must support CDC.
  - The *SQL Server Agent* for the instance must be running otherwise adding a CDC capture and cleanup jobs will fail.

Procedure

⚠ When provisioning or refreshing a SQL Server VDBs, both the DB_CHAINING and TRUSTWORTHY database parameters will be disabled (even if they were enabled on the dSource).

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select a **dSource**.
5. Click **Timeflow** tab.
6. Next to a snapshot select the  Provision VDB icon. The **Provision VDB** panel opens.
7. Select a **target environment**.
8. On the **Target Configuration** page, specify a **Mount Path.**
9. On the **Configuration**page, do the following:
   a. Under **Target Group**, select a Target Group for the VDB.
   b. Enter a database name.
   c. Under Recovery Model, click the drop-down list to select a recovery model. You can choose one of the following:
      i. Simple - This option is recommended and selected by default. It allows SQL Server to automatically mark parts of its transaction log file for re-use if they are not in use.
      ii. Full - This option lets you take the responsibility for taking backups and log backups of the VDB to an external location.
      iii. Bulk Logged - This option lets you take the responsibility for taking backups and log backups of the VDB to an external location.

   d. Specify any **Pre Scripts** or **Post Scripts** that should be used during the provisioning process. For CDC users, it is recommended that VDB Post Start hooks be configured to automatically start these jobs on successful provisioning

It is advised to use a Post Start hook as it is executed in all VDB operations like provision, refresh, rewind, disable/enable. Also, CDC jobs get deleted whenever we disable a VDB. The Post Start hook will add the CDC jobs again on enabling the VDB.

e. Under VDB configuration, enable **Auto VDB Restart** to allow the Delphix Engine to automatically restart the VDB when it detects the target host reboot.

f. To enable **Change Data Capture (CDC)**, select the Enable checkbox.

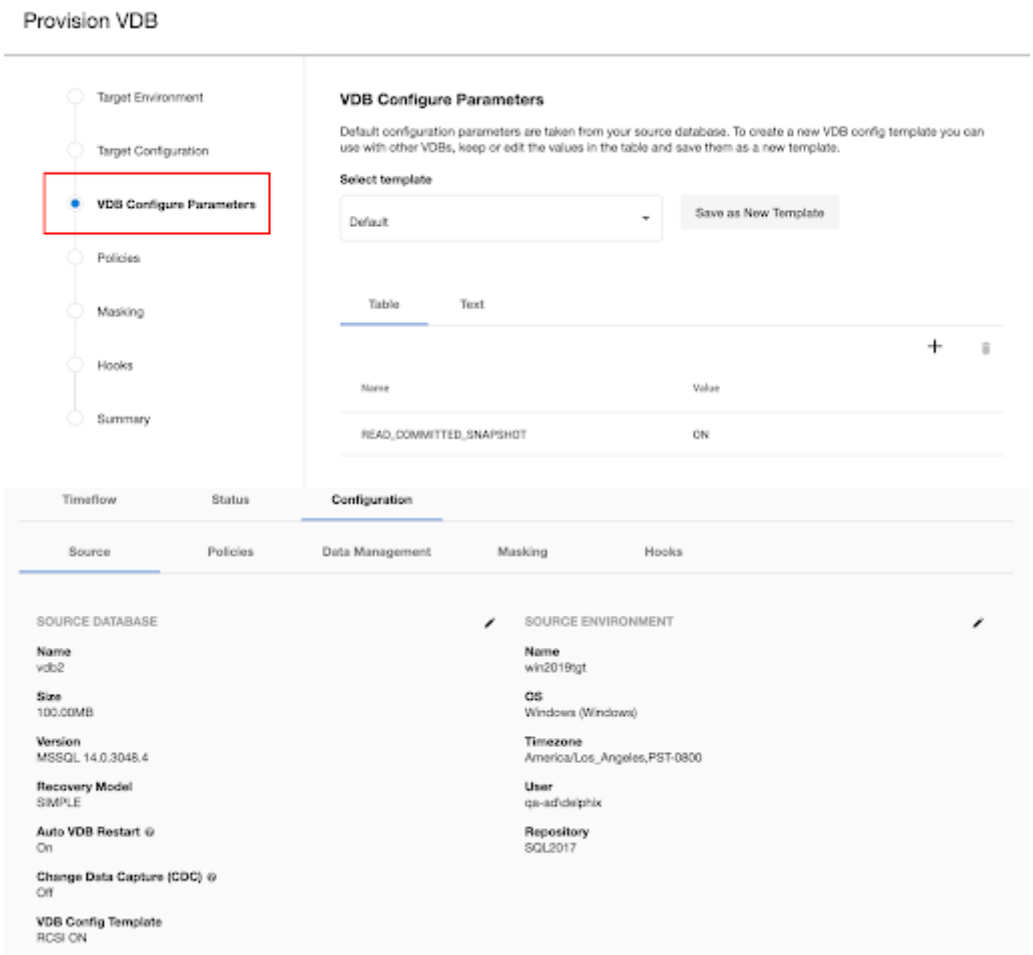
g. To enable VDB configuration parameters, select the **VDB Configuration Parameters** checkbox. This step displays a new page to either select an existing template or set configuration parameters.

READ_COMMITTED_SNAPSHOT is the only parameter that can be defined here, and the allowed values are OFF and ON.

To do so, perform the following steps on the **VDB Configure Parameters** page:

a. Click the plus icon to add a new key-value pair as a new template for the configuration parameters.

b. OR, From the **Select Template** dropdown list, select an existing configuration template to be applied on the VDB. Applied VDB Config Template will be displayed under the **Configuration** tab in the dataset and will remain editable.



h. Click **Next**.

CDC documentation

For more information see:

- About Change Data Capture

- Enable and Disable Change Data Capture

10. Under **Policies**, select a **Snapshot Policy** for the VDB and click **Next**.
11. Under **Masking**, select **Mask this VDB** checkbox to mask your data during provisioning and then select one of the following masking options:
    a. Select an existing masking job
    b. Masking Job is not currently available for the selected data type, please mask using script(s) instead. If you select this option, you should define a Configure Clone script in the Hooks step to mask the dataset.
12. Under **Hooks**, specify any Hooks to be used during the provisioning process. For more information, see Hooks for SQL Server.
13. Click **Next**.
14. The final summary tab will enable you to review your configurations.
15. Click **Submit**.

When provisioning starts, the VDB will appear in the **Datasets** panel. Select the VDB and navigate to the **Status** tab to see the progress of the job. When provisioning is complete, you can see more information on the **Configuration** tab.

> You can select a SQL Server instance that has a higher version than the source database and the VDB will be automatically upgraded. For more information about compatibility between different versions of SQL Server, see SQL Server Support Matrix.

Provisioning by snapshot or logSync

When provisioning by snapshot, you can provision to the start of any particular snapshot, either by time or LSN.

> You can take a new snapshot of the dSource and provision from it by clicking the **Camera** icon.
> Provisioning By Snapshot

Provisioning By Snapshot	Description
Provision by Time	You can provision to the start of any snapshot by selecting that snapshot card from the **TimeFlow** tab, or by selecting and entering a value in the time entry fields. The values you enter will snap to the beginning of the nearest snapshot.
Provision by LSN	You can use **Provision by LSN** control to open the LSN entry field. Here, you can type or paste in the LSN to which you want to provision. After entering a value, it will "snap" to the start of the closest appropriate snapshot.

V2P with a SQL Server VDB

This topic describes how to perform the Virtual to Physical (V2P) process with a SQL Server virtual database (VDB).

Procedure

1. Log into the **Delphix Management** application.
2. Click **Manage** and select **Datasets**.
3. Select the dSource or VDB you want to export.
4. Select the snapshot of the dSource or VDB state you want to export.
5. If you want to export the state of the database from a specific point in time, select the **LogSync** icon and then select the point in time from which you want to create the export.

6. From the actions menu (...) select **Virtual to Physical**.
7. Select the target environment.
8. Enter the **Target Directory** for the export. The target environment should have been added to Delphix previously and should meet all target host requirements, see Overview of Requirements for SQL Server Environments for more information on user requirements for target environments.
   The target directory you enter here must exist in the target environment and the Delphix operating system user listed under the environment must have permission to write to it.
9. Select the checkbox to enable or disable the option **Open database after Recovery**.
   If you select to **disable** this option, you can use the scripts that are created in the target environment to manually recover the database at a later time. See Manually Recovering a Database after V2P for more information.
10. Click **Show Advanced** to customize the target directory layout or to enable Change Data Capture (CDC) on the exported database. See Customizing Target Directory Structure for Database Export for more information.
11. Click **Next**.
12. Select whether you want to have an email sent to you when the export process completes, and then click **Submit**.

Post-requisites

If you have selected disable for **Open database after Recovery**, then follow the instructions in Manually Recovering a Database after V2P to complete the V2P process.

Additional SQL Server VDB topics

Renaming a SQL server VDB via CLI

This topic describes how to change the database name on the SQL Server Instance for VDB through the Delphix CLI. Database name on SQL Server vs. VDB name on Delphix The database name is what you would see the SQL Server instance on the Target in...

Updated on : 25 May 2023

Upgrading SQL server VDBs

This topic describes how to upgrade a SQL Server VDB to a higher version of SQL Server instance. Procedure for VDB In-Place Upgrade Remove any VDB Refresh Policy assigned to the VDB. Upgrade the target SQL Server instance. Refresh the target ...

Updated on : 25 May 2023

Extended properties for SQL server VDBs

Extended Properties and How to View Them This topic describes extended properties on VDBs that you can use to track the origin of VDBs through SQL Server tools on target servers. These are the extended properties: You can find these properties ...

Updated on : 25 May 2023

File permissions for SQL server VDBs

When provisioning a VDB, the Delphix Engine modifies the "access control lists" (ACLs) of database and log files to help prevent unintentional data loss through file deletion. Files could be deleted, for example, if there is an attempt to DROP a VD...

Updated on : 25 May 2023

Deleting a SQL server VDB

Procedure Deleting a VDB is an unrecoverable operation. Proceed only if you want to permanently destroy the unique data that was created in the VDB. Login to the Delphix Management application. Click Manage . Select Datasets . Click th...

Updated on : 25 May 2023

Renaming a SQL Server VDB via CLI

This topic describes how to change the database name on the SQL Server Instance for VDB through the Delphix CLI.

> Database name on SQL Server vs. VDB name on Delphix
> The database name is what you would see the SQL Server instance on the Target in SQL Server Management Studio or sys.databases. It is also the database name in the **Configuration** tab in Delphix. The name of the VDB is an internal name within Delphix Engine and does not need to be the same as the database name on Target. This is found on the **Status** tab of Delphx.

Prerequisites

- The VDB must be running on the target environment.
- The SQL Server instance on the target environment where the VDB is located must be up and reachable.

Procedure

1. Select the **source** associated with the VDB and disable the VDB.

```
delphix> source "vexample"
delphix source 'vexample'> disable
delphix source 'vexample' disable *> commit
```

2. Select the source **config** associated with the source.

```
delphix source "vexample" > get config
  vexample
delphix source "vexample" > /sourceconfig "vexample"
delphix sourceconfig "vexample" >
```

3. Update the **databaseName** to the new name.

```
delphix sourceconfig "vexample" > update
delphix sourceconfig "vexample" update *> set databaseName=newDBName
delphix sourceconfig "vexample" update *> commit
delphix sourceconfig "vexample" >cd
```

4. Enable the VDB.

```
delphix> source "vexample"
delphix source 'vexample'> enable
delphix source 'vexample' enable *> commit
```

Upgrading SQL Server VDBs

This topic describes how to upgrade a SQL Server VDB to a higher version of SQL Server instance.

Procedure for VDB in-place upgrade

1. Remove any VDB Refresh Policy assigned to the VDB.
2. Upgrade the target SQL Server instance.
3. Refresh the target environment.

Procedure to upgrade a VDB to a new SQL instance

1. Refresh all environments.
2. Login to the **Delphix Management** application.
3. Click **Manage**.
4. Select **Datasets**.
5. Select the **VDB** to be upgraded.
6. From the **Actions** menu (...) select **Disable**.
7. Click **Disable** to confirm.
8. From the **Actions** menu (...) select **upgrade**. The **Upgrade Database** window will open.
9. Select the **SQL Server instance** to which you want the VDB to upgrade.
10. Click **OK**.
11. Enable the VDB. (See the *Enabling and Disabling SQL Server VDBs section in Provisioning and Managing Virtual Databases*.)
12. Repeat steps 5 to 12 for each VDB you want to upgrade.

Extended properties for SQL Server VDBs

Extended properties and how to view them

This topic describes extended properties on VDBs that you can use to track the origin of VDBs through SQL Server tools on target servers.

These are the extended properties:

Property	Description
`dlpx_server_name`	Address of the Delphix Engine hosting the VDB
`dlpx_server_uuid`	UUID of the Delphix Engine hosting the VDB
`dlpx_source_id`	Internal reference of the VDB

You can find these properties in the following locations:

- For a VDB using the SQL Server Management Studio tool: under the Extended Properties page of the Properties window
- Using the `sp_dlpx_vdbinfo` stored procedure.
    - To install and run this stored procedure, run the SQL code contained in `<Delphix Connector install path>\etc\sp_dlpx_vdbinfo.sql.`

File permissions for SQL Server VDBs

When provisioning a VDB, the Delphix Engine modifies the "access control lists" (ACLs) of database and log files to help prevent unintentional data loss through file deletion. Files could be deleted, for example, if there is an attempt to DROP a VDB directly through SQL Server management studio or other native SQL Server tools.

The Delphix Engine updates each database and log file ACL to include a deny-delete "access control entry" (ACE) for the user account running the SQL Server instance.

You can still drop VDBs directly through SQL Server tools. However, a warning message will be displayed, and the files will remain on the volume that the Delphix Engine exports. This file deletion prevention also applies to attempts to remove files from a database using the ALTER DATABASE .. REMOVE FILE command.

If a VDB is inadvertently dropped, you can reattach the database using SQL Server tools.

If you attempt to delete a database or log file and then try to add a file of the same name, this may fail because the original file was prevented from being deleted by the deny-delete ACE.

If you intend to delete the files from the volume that the Delphix Engine provides, you can change the ACLs on the files using the icacls command:

```
icacls <file> /remove <SQL Server instance account>:deny(D)
```

Accounts other than the SQL Server instance account will not be prevented from deleting the VDB database and log files.

Deleting a SQL Server VDB

Procedure

Deleting a VDB is an unrecoverable operation. Proceed only if you want to permanently destroy the unique data that was created in the VDB.

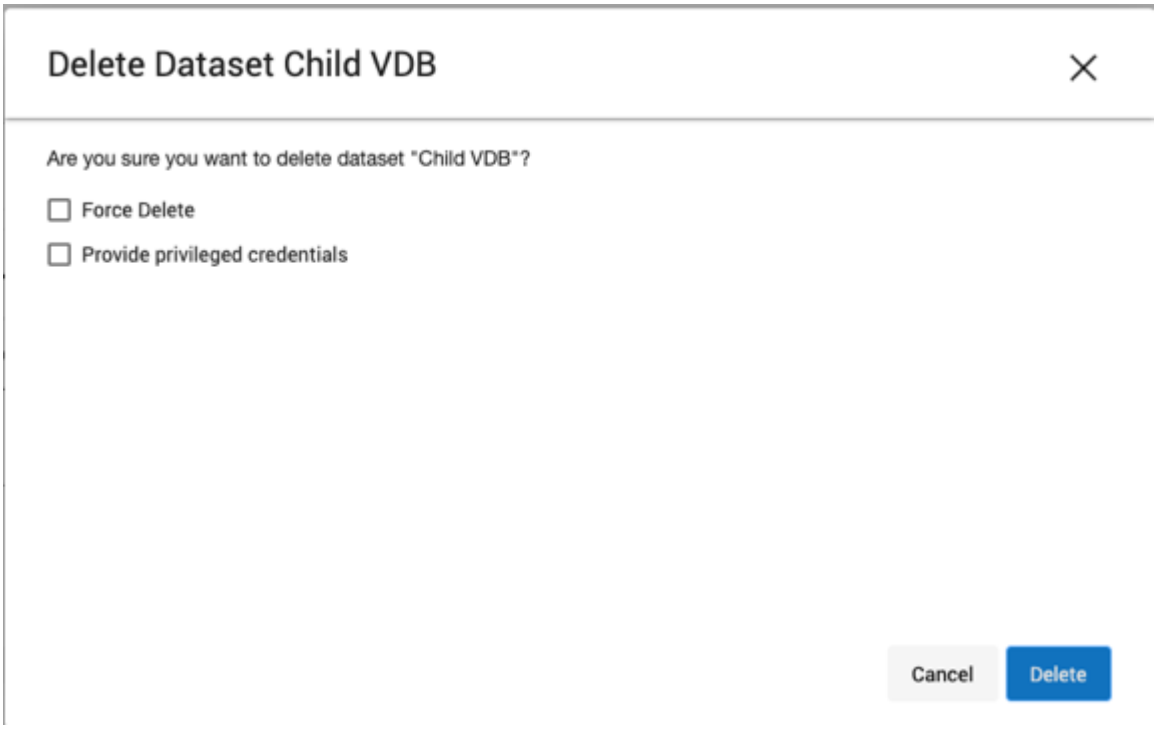
1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Click the **VDB** that you want to delete.
5. From the Actions menu (...) select **Delete.**
6. If stopping or starting the VDB requires particular credentials for the target environment other than those of the default environment user:
    a. Check **Provide Privileged Credentials**.
    b. Enter the **username** and **password**.
    c. Click **Validate Credentials**.
7. Click **Delete** to confirm that you want to delete the VDB.

If the VDB was currently active, the Delphix Engine will shut it down, unmount all filesystems from the target environment, and finally delete the VDB itself.

Using force delete

Deleting unused or outdated objects should be a regular part of Delphix Engine administration. This is especially important to prevent low space errors, which can cause the Delphix Engine to stop. The Delphix Engine holds a maximum of 400 objects.

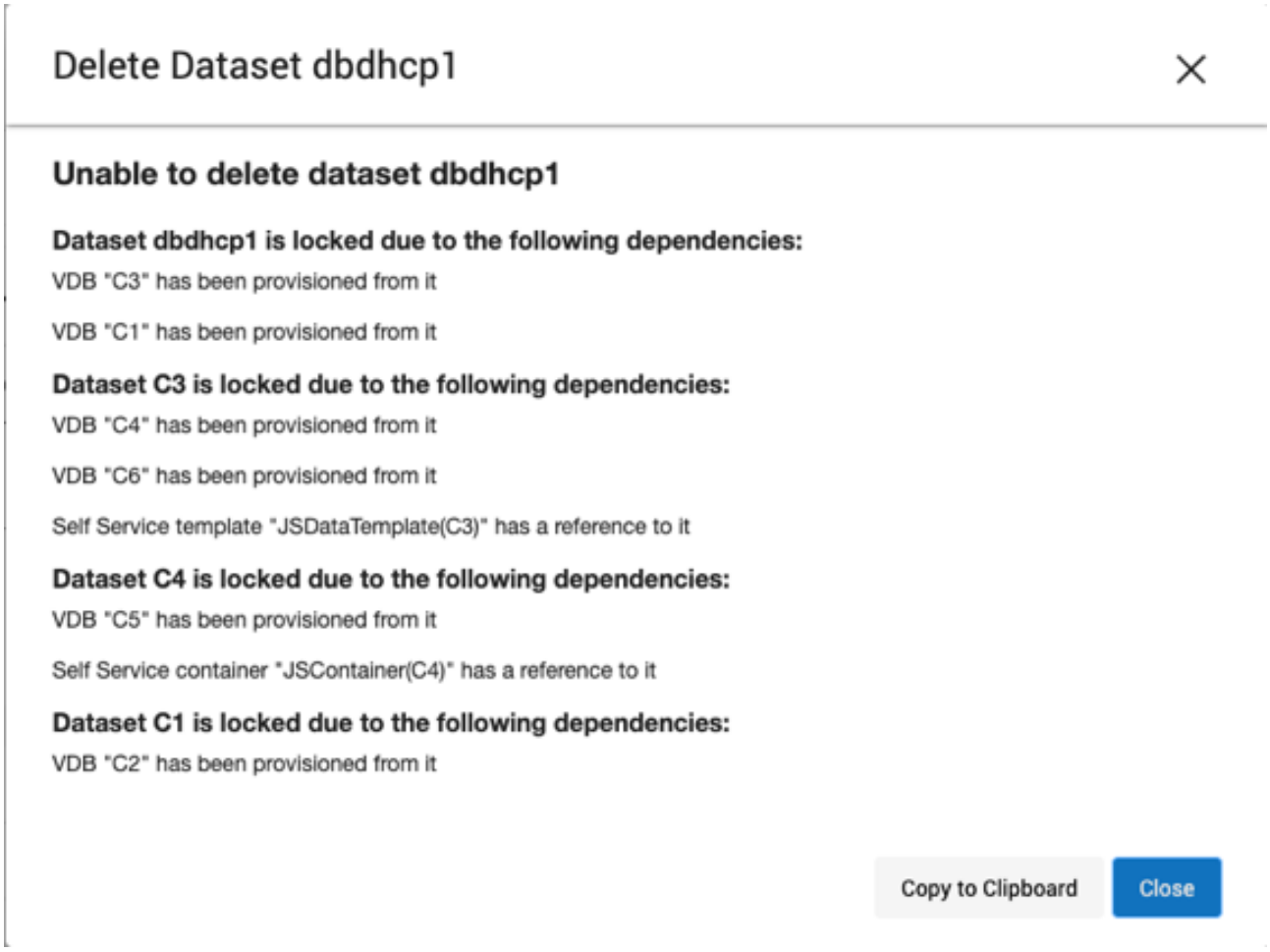
1. Log into the **Delphix Management** application.
2. Select **Resources > Storage Capacity**.
3. Next to the object you want to delete select the **Trashcan**.
4. In the Delete dialog select **Force Delete**. Oracle users will have the option to provide additional credentials.



5. Click Delete.

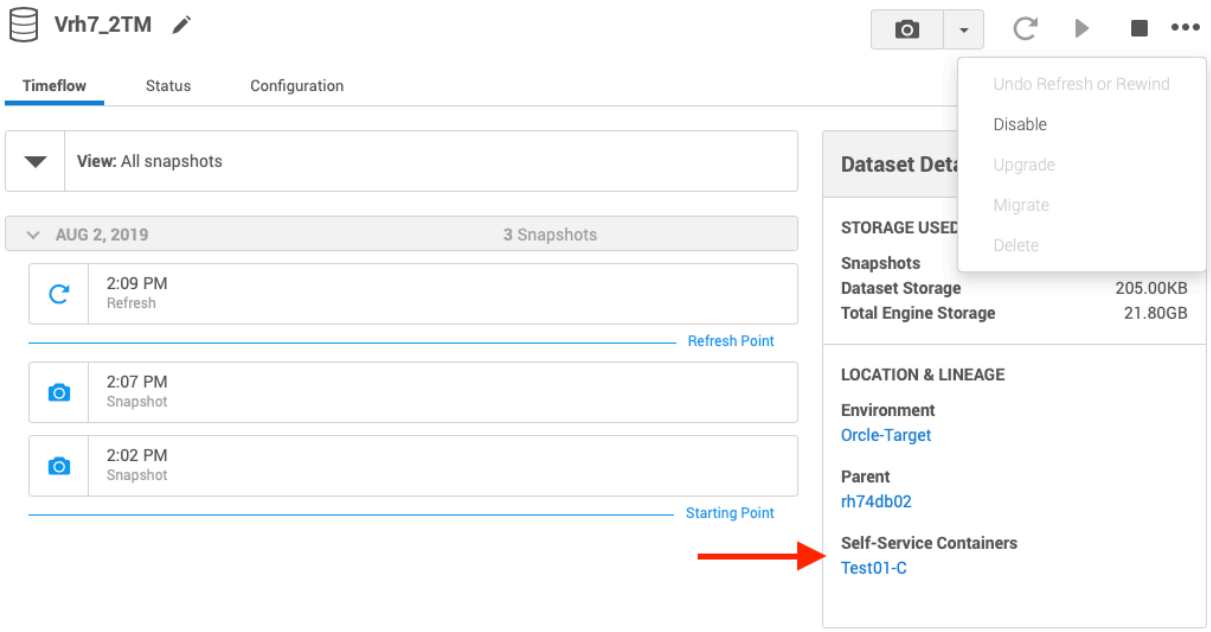
Dependencies

If there are dependencies on the snapshot, you will not be able to delete the snapshot free space; the dependencies rely on the data associated with the snapshot. These items are displayed with a lock icon next to the name.



Deleting a VDB associated with a self-service container

As shown below if a VDB is associated with a Self-Service container, the delete option will not show.



Users will need to first delete the container and then delete the VDB. Refer to Delphix Self-Service Data Container Activities.

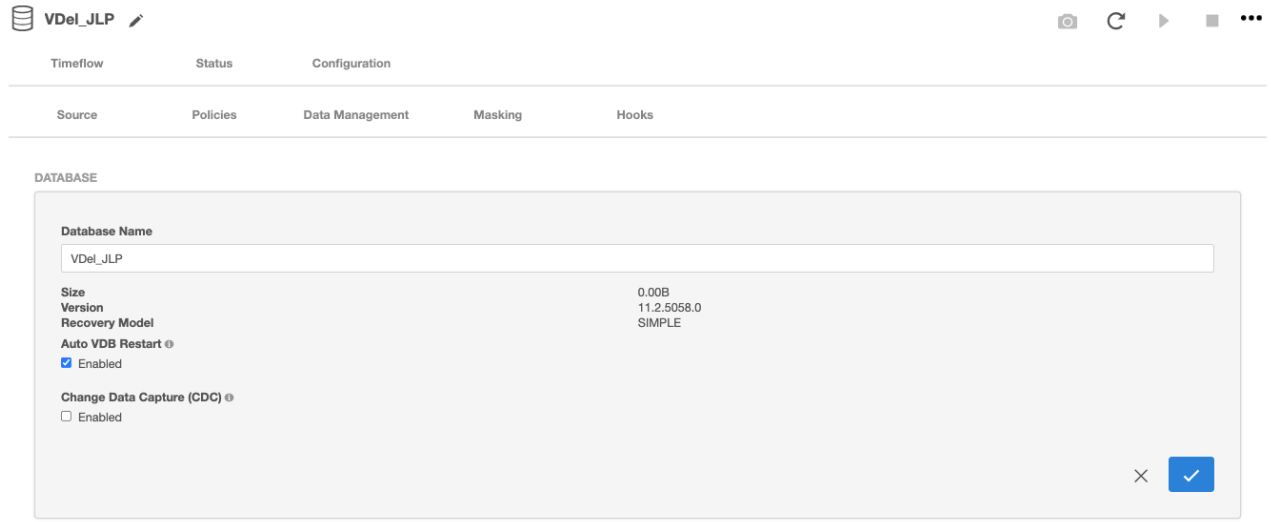
CDC support in SQL Server

Provisioning CDC enabled virtual databases

Please see Provisioning a SQL Server VDB.

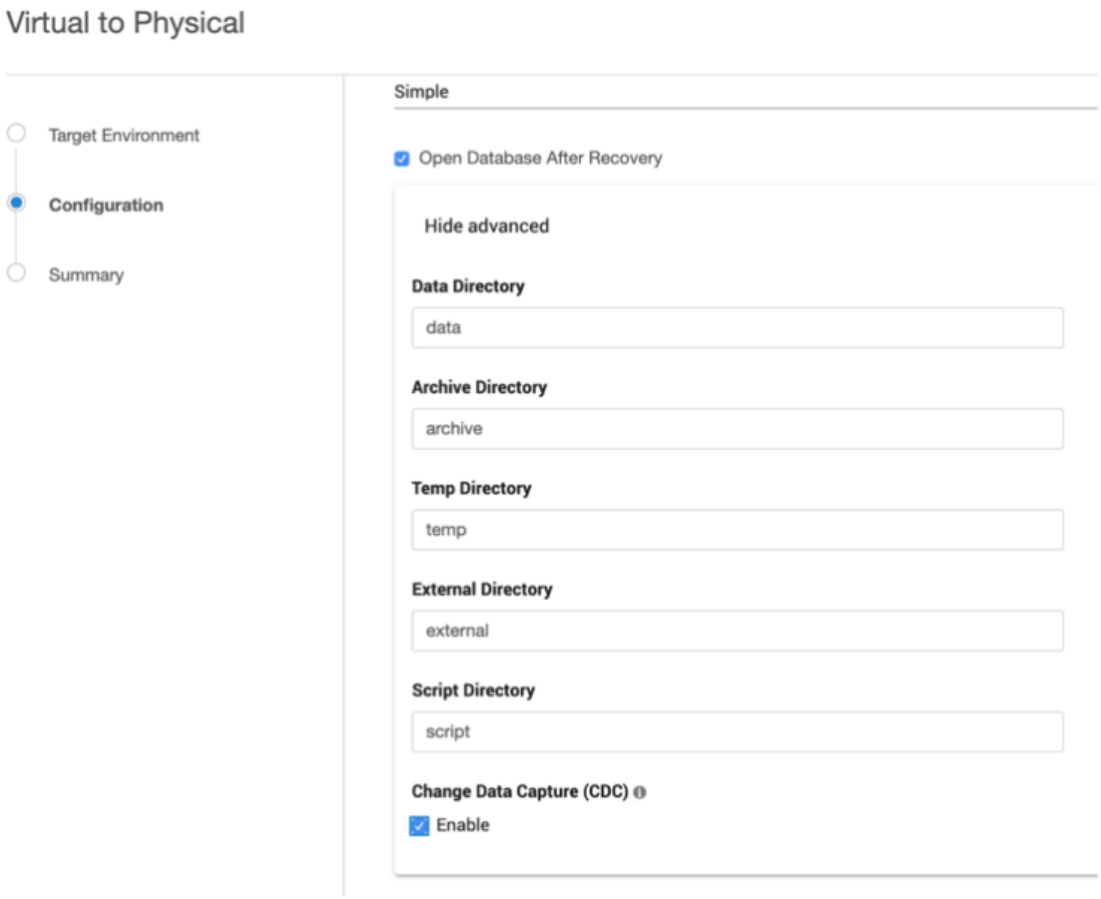
Viewing or updating CDC your configuration

CDC configuration used while provisioning would be saved and used for subsequent Refresh and Rewind operations on the VDB. This configuration can be viewed/updated after the VDB has provisioned, in the 'Source' tab under the **Configuration'**section of the VDB.



Exporting CDC enabled physical databases

1. In the Configuration page of the wizard, open the Advanced section and Select the **'Enable'** option under 'Change Data Capture (CDC)



2. CDC capture and cleanup jobs have to be added and CDC metadata has to be upgraded (if exporting is done from a lower database version to SQL2016 and above), manually on exported databases.

General notes/troubleshooting

- Make sure the SQL Server instance on which the VDB is being provisioned or exported, supports CDC.
- Make sure 'SQL Server Agent' for the instance is running otherwise adding CDC capture and cleanup jobs will fail.

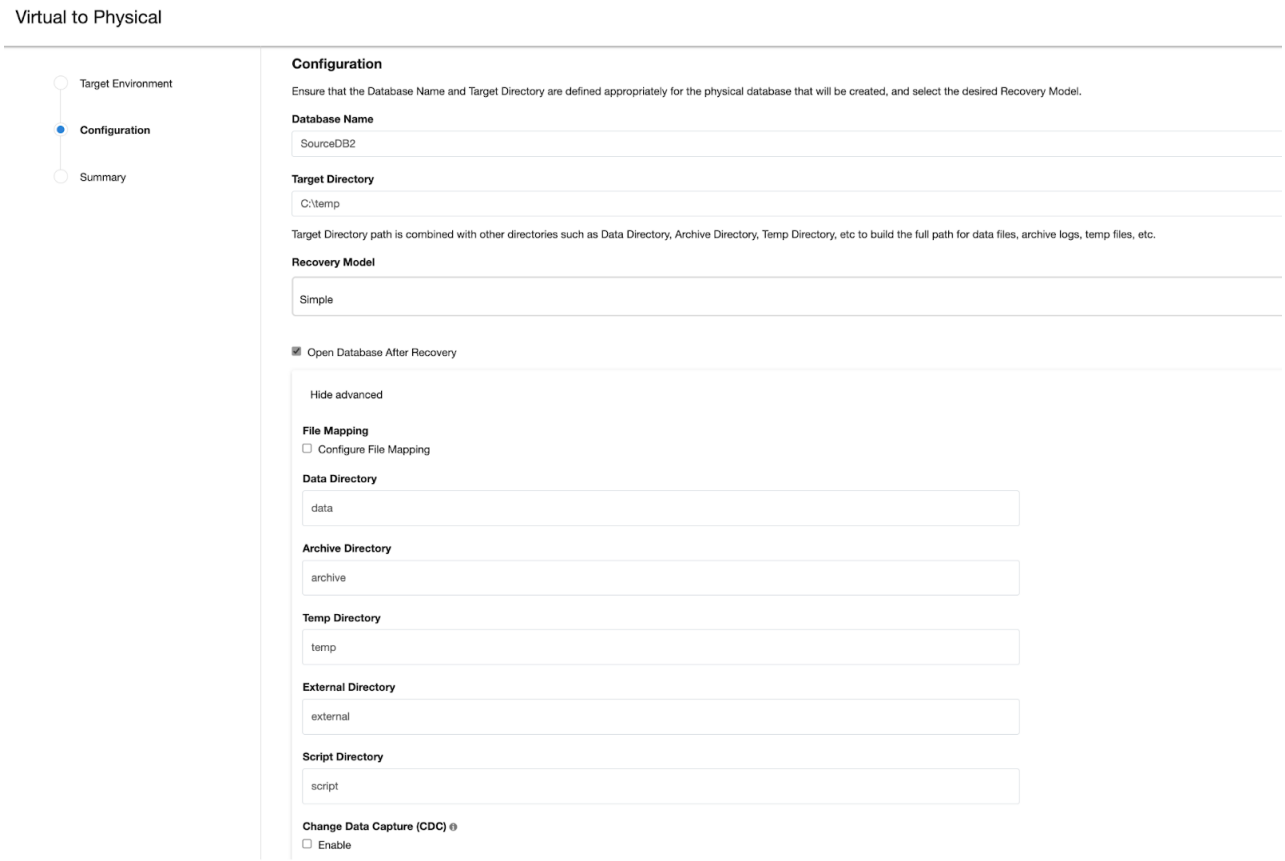
MSSQL V2P file mapping

Introduction

This article describes how to customize file path mappings when performing a Virtual to Physical (V2P) operation for MSSQL databases. During the V2P process, it could be required to create mappings between the files and directories existing on the staging host, as well as files and directories created on the target. For example, putting all of the transaction log files that exist on the staging environment into a folder on the target machine.

The **Configuration** section of the process has name fields that can be specified for the database and directories, as shown in the screenshot below.

- Database Name: V2PDatabase
- Target Directory: C:\temp
- Data Directory: data
- Archive Directory: archive
- Script Directory: script
- Temp Directory: temp
- External Directory: external



When working with File Mappings, the data file names affect everything that follows. These data files, including log files and File Streams folders, inherit file names from the dSources and VDBs by default (e.g., SourceDB2.mdf, SourceDB2_log.ldf, File_Stream, etc.). Users have control over the data files that are part of the dSource and VDB snapshots.

Archive log files go directly into the **C:\temp\archive** directory. If provided, the V2P process automatically appends the target directory and data directory to the data file names, as shown in an example list below.

- C:\temp\data\SourceDB2.mdf
- C:\temp\data\SourceDB2_log.ldf
- C:\temp\data\File_Stream

Pattern matching

Pattern Matching rules can be used to create full path names for data files and control files. These rules have take this format: **source-regex-expression-KEY ? target-replacement-VALUE**. Multiple rules can be used and are applied successively. In addition, multiple rules with the same source key are allowed.

File mapping options

Example 1

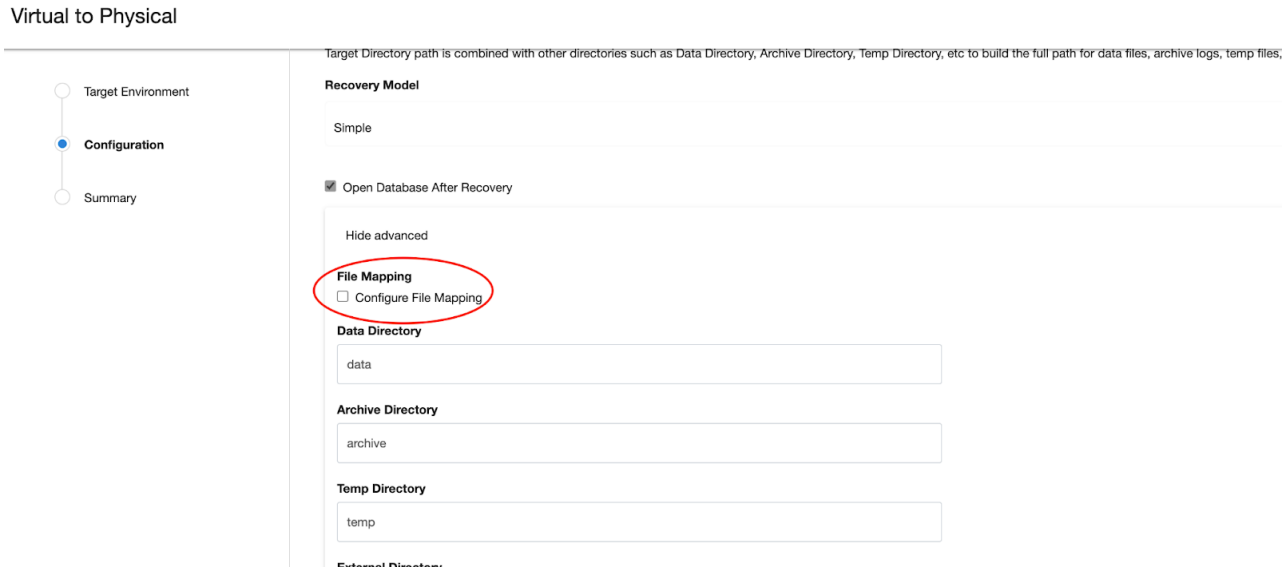
For this example, the ultimate goal is to perform a V2P operation that has the following data file File Mappings:

- C:\temp\SourceDB2\data\SourceDB2.mdf
- D:\temp\SourceDB2\logFiles\SourceDB2_log.ldf
- E:\temp\SourceDB2\fileStream\SourceDB2_File_Stream

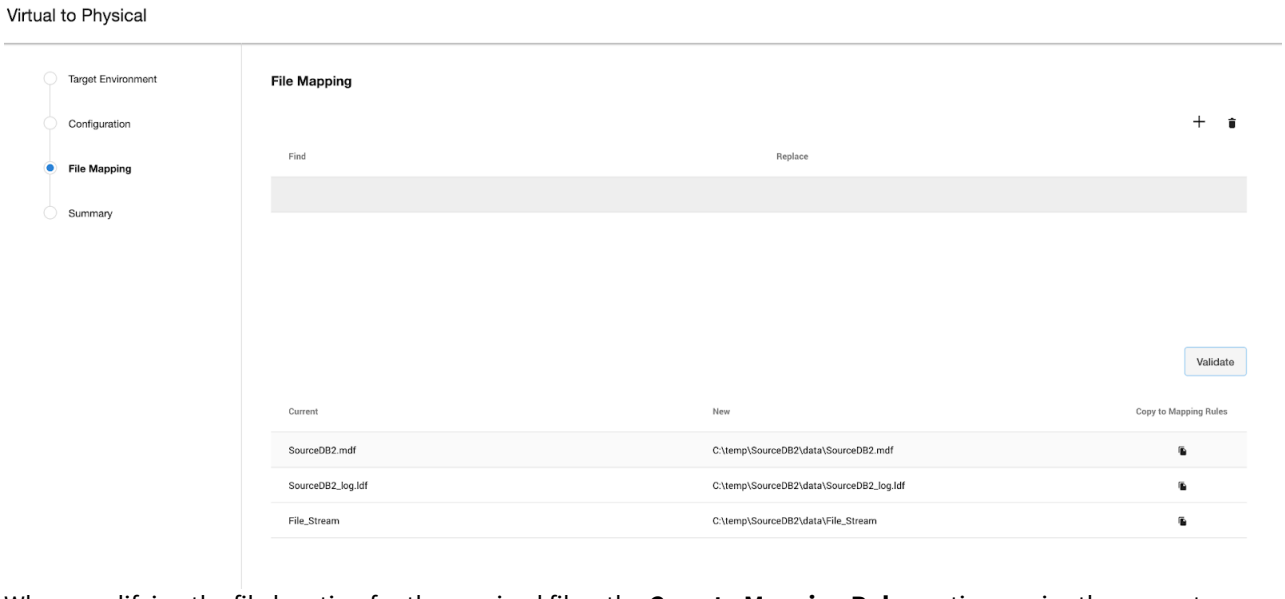
The default behavior can be changed to modify the target directory, modify the data directory, or modify the individual file location

- Modify the target directory.
- Modify the data directory.
- Modify the individual file location by using File Mapping.

To modify the location of each file, select **Configure File Mapping** in the **Advanced** tab of the Configuration page.

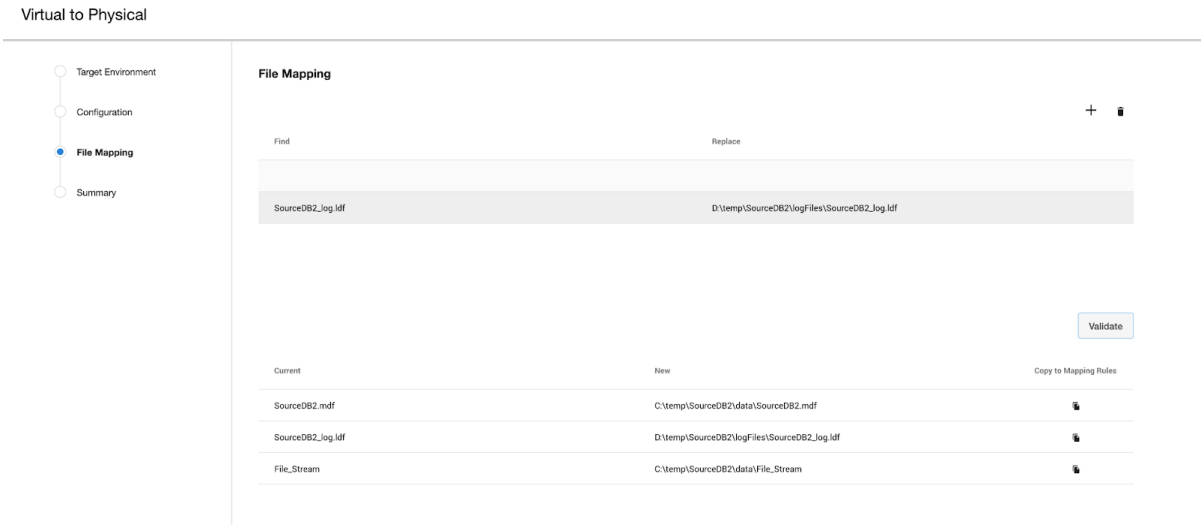


Click the + button to add a new File Mapping or fetch all of the available data files that can be modified by simply clicking **Validate** with an **empty** File Mapping list, as shown in the screenshot below.

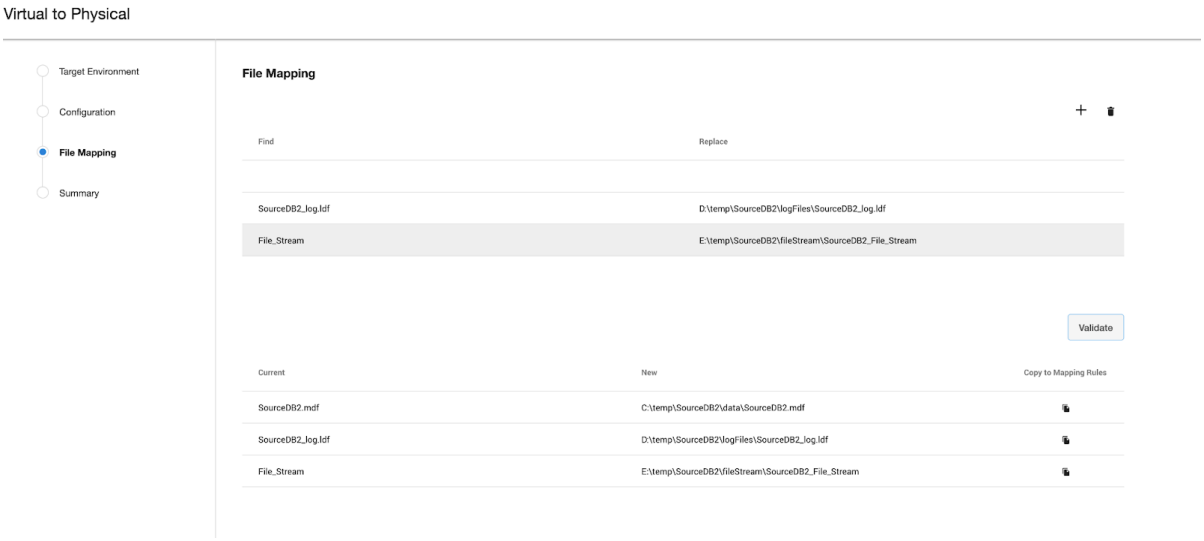


When modifying the file location for the received files, the **Copy to Mapping Rules** option copies the current (source location) and new (target location) to File Mapping, or the values can be entered manually to Find and Replace. The steps below are to follow.

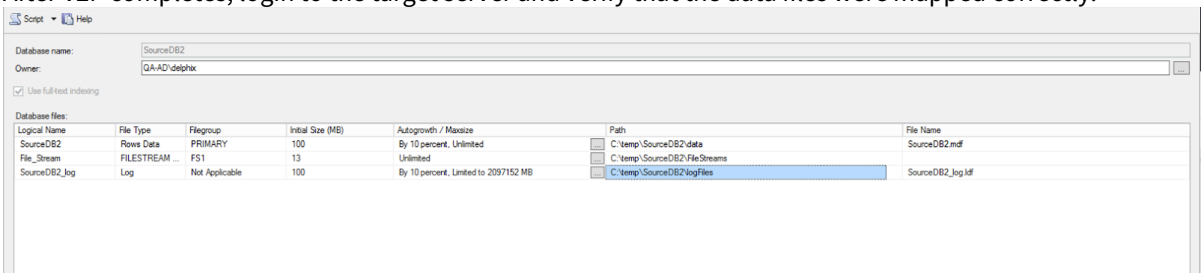
1. Since the new (target location) of the SourceDB2.mdf would appended to `C:\temp\SourceDB2\data\SourceDB2.mdf` (a combination of **C:\temp\SourceDB2** (target directory), **data** (data directory), and **SourceDB2.mdf** (file name)), applying File Mapping is not necessary.
2. Copy the SourceDB2_log.ldf file using the **Copy to Mapping Rules** option.
3. Place the SourceDB2_log.ldf file into the `D:\temp\SourceDB2\logFiles\SourceDB2_log.ldf` location.
4. Configure **Replace** to `D:\temp\SourceDB2\logFiles\SourceDB2_log.ldf` and then select **Validate** to see the results.



5. Similarly, to move **File_Stream** to `E:\temp\SourceDB2\fileStream\SourceDB2_File_Stream`, provide the mapping as shown below.



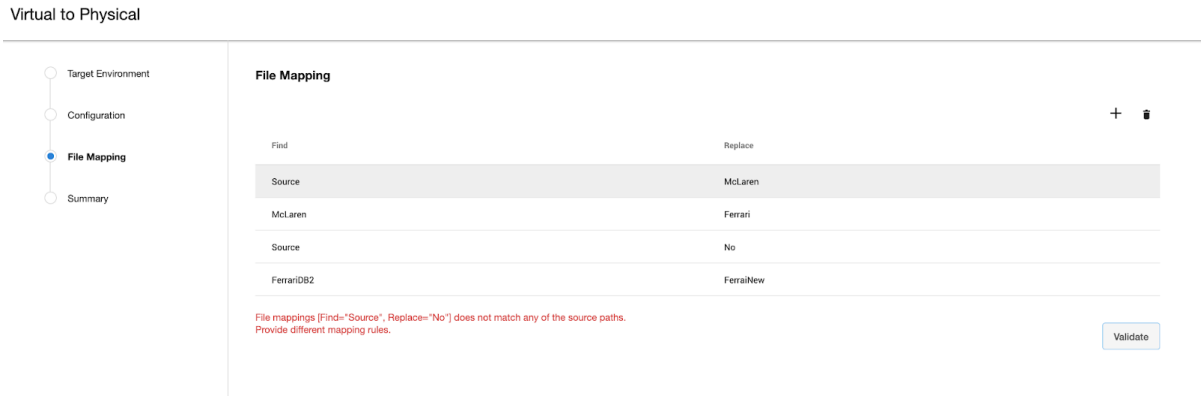
6. Select **Validate** between each new entry, in order to verify that data files are being mapped as expected.
7. The File Mappings build upon one another, so all the provided File Mapping Rules are applied sequentially to each source file in order to generate a target file location.
8. Once all the files are located as desired, select **Next** to continue the provision process.
9. The **Summary** page will show the modifications to **Target Directory** and **Database Name** directories, and will show that **Customized File Mapping** was defined.
10. After V2P completes, login to the target server and verify that the data files were mapped correctly.



Example 2

In this example, several rules are applied to the source file path for **SourceDB2.mdf**. Note that the rules are applied in sequential order, as shown in the screenshot below.

1. Applying the rule **Source?McLaren** results in: **C:\temp\data\McLarenDB2.mdf**
2. Applying the rule **McLaren?Ferrari** results in: **C:\temp\data\FerrariDB2.mdf**
3. Applying the rule **FerrariDB2?FerraiNew** results in: **C:\temp\data\FerraiNew.mdf**
4. Applying the rule **Source?no** results in an error, because **Source** is no longer found in the pathname.



During the pattern matching process, various errors can be generated. Some of these errors are described below.

1. **No match for specified mapping rules** When none of the rules match a source file
2. **Invalid regex pattern specified for path mapping** An invalid regex rule mapping error
3. **File Extension Mismatch** A modified file extension or folder to file conversion error
4. **Duplicate target Paths** When the same target path is created for multiple source files

The java.regex.util class article (redirect to docs.oracle.com) shows the regular expression syntax and constructs recognized by the Delphix Engine pattern-matching operations.

Hooks for SQL Server

Overview

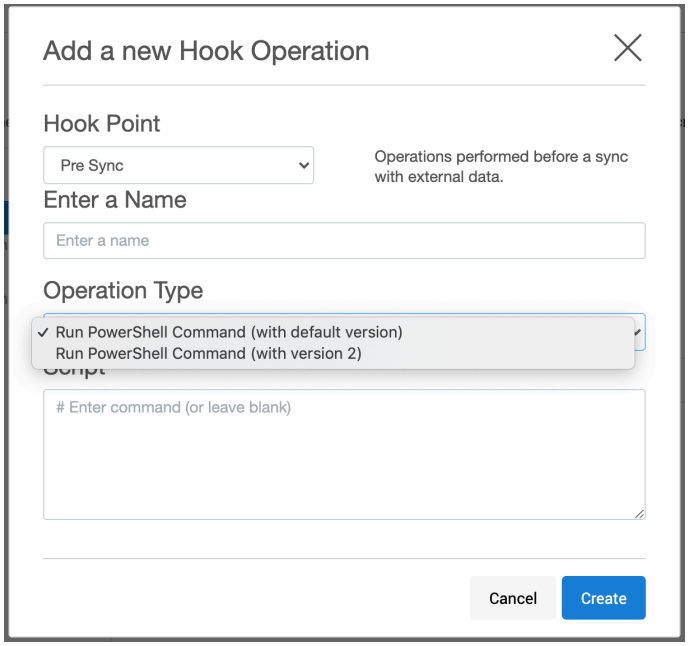
This topic describes the use of hook operations with dSources created from SQL Server source databases and virtual databases (VDBs) that are created from SQL Server dSources or other VDBs.

Hooks are Windows Powershell code executed on:

1. The staging target host before or after the manual snapshot of a dSource.
2. The VDB target host before and after the provision, refresh, rewind, snapshot, start or stop of a VDB.

Powershell version

While creating a hook, a user can provide the PowerShell version (default version or version 2) in the field "Operation Type" and this PowerShell version will be used to execute the hook script. Here, the default version is the version of PowerShell installed on the target host.



Hooks can be specified in the wizard used during the creation of a VDB, or modified afterward by navigating to the Configuration > Hooks tab. Hooks can also be set using the Delphix command-line interface (CLI) or REST Web API.

Each hook operation represents a user-configurable action that the Delphix virtualization engine will execute. You can configure the custom hook code to fail if they encounter an unexpected error. The failure of a hook operation will cause the enclosing operation to fail.

The Windows environment user for the dSource or VDB runs the "Powershell" executable, which runs the specified PowerShell script on the Staging or VDB Target host. The Delphix Engine captures and logs all output of the script and displays it if a failure occurs.

The intent of hook operations are customization of the data contents or configuration of a dataset while it is being manipulated. Actions performed by hooks effectively become an integrated part of the sync operations of a dSource or the provision, refresh, rewind, snapshot, start, or stop actions for that VDB.

Hooks are mainly used for pre- and post-provisioning operations. For example, you can use hooks to:

- Back up test data before refresh and rewind
- Back up data after provisioning
- Reset configuration settings from production to non-production settings after provisioning
- Create logins for dev/qa users who do not have privileges on production databases
- Sync logins on the target that are cloned from the production database
- Back up configuration data from the database

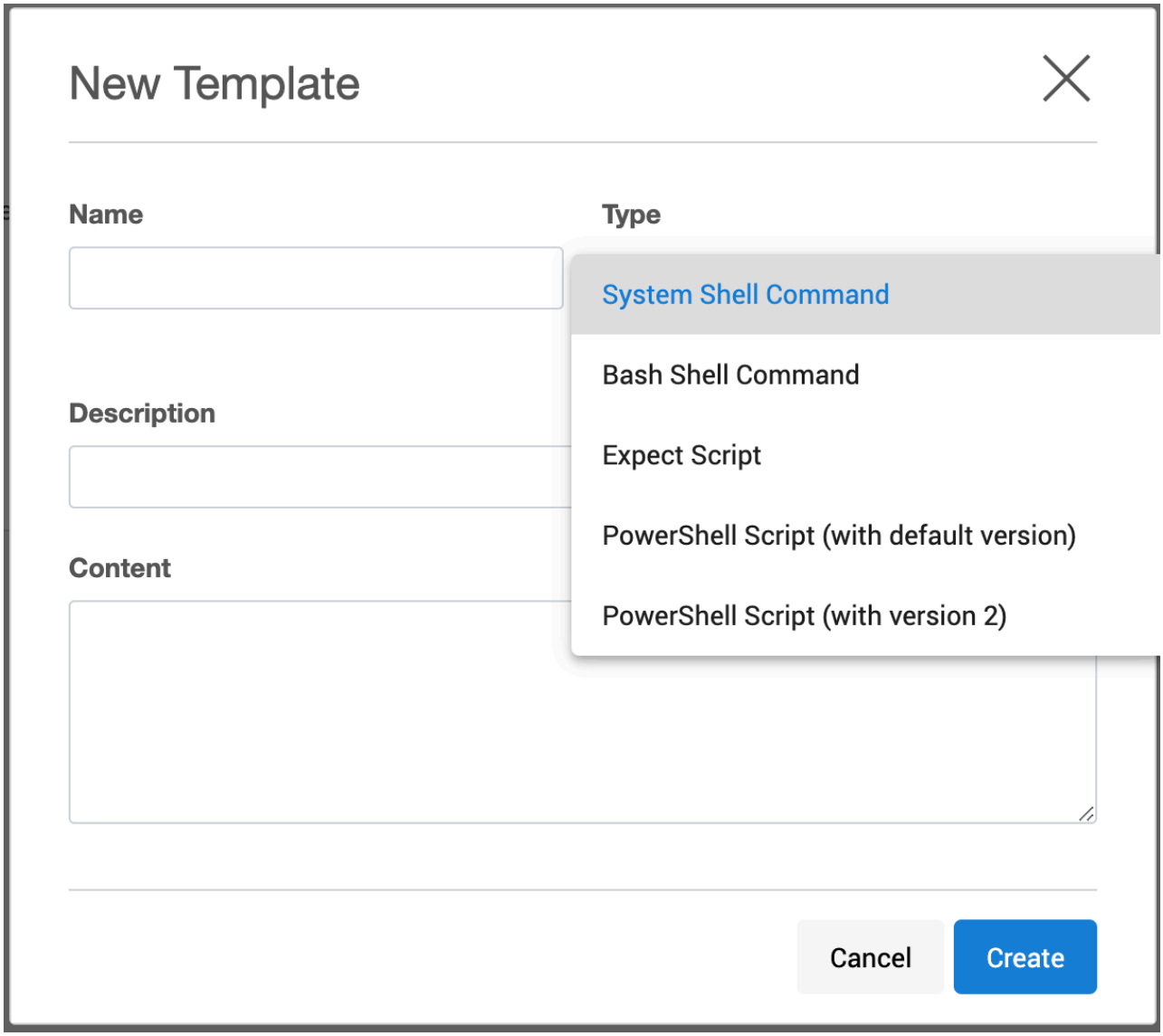
For more information on Hook Operations, see SQL server hook operation notes

Hook operation templates

You can use operation templates to store commonly used operations, which allows you to avoid repeated code entry when an operation is applicable to more than a single hook, dSource, or virtual dataset. You can manage templates through the Delphix Management application.

You can also create templates from existing hooks by exporting the hooks in the Delphix Management application.

While creating a hook template, the user can provide the PowerShell version (default version or version 2) in the field "Type" and this PowerShell version will be used to execute the hook's script created from this template. Here, the default version is the version of PowerShell installed on the target host.



The existing template's PowerShell version can be changed by using Delphix CLI only because UI currently does not support this feature.

Windows environment variables

When a hook is executed, Delphix will set specific Windows environment variables to provide context, such as the name of the current host, the name of the SQL Server instance and port, and the name of the database. For more information, see SQL server hook operation notes

Python script to migrate hooks from PowerShell version 2 to host's default powerShell version.

Overview

With 6.0.3.0, the Delphix Engine will use the default PowerShell version installed on the host (hereinafter referred as default PowerShell) to perform all its operations, also the new hooks and hook templates can be created using the default PowerShell. The existing hooks and hook templates on the engine will continue functioning using PowerShell version 2, and there will be an option to migrate them to use default PowerShell.

The article intends to introduce a Python script that can be used to migrate all the hooks and hook templates on the engine to default PowerShell. The motivation behind writing the script is to save the manual effort required in doing the migration via the UI or CLI.
Link to download the script.

Requirements for running the script

The requirements for running the script are as follows, they are the same as running any Python script in general.

1. The machine where the script is run should have Python installed.
    a. Relevant link: https://www.python.org/downloads/
    b. The script is supported for both Python 2 and Python 3 release.
2. The machine where the script is run should have the `delphixpy` Python package installed.
    a. Relevant link: https://pypi.org/project/delphixpy/
    b. python.org documentation on installing packages and creating Python environments: Installing packages using pip and virtual environments.
    c. If the package is already installed, it should be upgraded to the latest version, the minimum `delphixpy` version required to run the script is 1.11.3.0.
3. The Delphix Engine should be accessible from the machine where the script is run since the script makes API requests to the Delphix Engine to perform the migration,
    a. The easiest way to verify the same is to use the **ping** command.

Functionalities of the script

1. The script can migrate the hooks and hook templates in the Delphix Engine to run with default PowerShell.
2. The script can also migrate the hooks and hook templates back to PowerShell version 2.
3. It is possible to migrate only the hooks or only the hook templates.
4. If **INSTALLEDPOWERSHELL** feature flag is enabled on the Delphix Engine, the script will disable the same.

The script parameters and usage

Script parameter	Description	Type	Possible values	Default value
`--help`	Displays the description and usage details of the parameters for the script on the terminal	Optional	Not applicable	Not applicable
`--engine-addr`	The Delphix Engine host address.	Required	Not applicable	Not applicable

Script parameter	Description	Type	Possible values	Default value
`--sys-admin-usr`	The username for System Administrator user to log into the Delphix Engine.	Required	Not applicable	Not applicable
`--sys-admin-pwd`	The password for System Administrator user to log into the Delphix Engine.	Required	Not applicable	Not applicable
`--admin-usr`	The username for Engine Administrator user to log into the Delphix Engine.	Required	Not applicable	Not applicable
`--admin-pwd`	The password for Engine Administrator user to log into the Delphix Engine.	Required	Not applicable	Not applicable
`--hook-ps-version`	Migrates all the hooks to PowerShell Version two.	Optional	default, ps2	default
`--hook-templ-ps-version`	Migrates all the hook templates to PowerShell Version two.	Optional	default, ps2	default
`--migrate-only`	Migrates only the hook or only the hook templates. Input "hooks" for migrating only the hooks, whereas input "templates" for migrating only the hook templates	Optional	hooks, templates, hooks-and-templates	hooks-and-templates
`--debug`	In case there is an error executing the script, prints the Python stack trace required for debugging.	Optional	Not applicable	Not applicable

Command examples

1. Displaying the description and usage details of the parameters for the script on the terminal

```
python <path to the script> --help
```

2. An example command for migrating all the hook and hook templates to default PowerShell version:

```
python <path to the script> --engine-addr engine.delphix.com --sys-admin-usr
sysadmin --sys-admin-pwd sysadmin --admin-usr admin --admin-pwd delphix
```

3. An example command for migrating all the hook and hook templates to PowerShell version 2:

```
python <path to the script> --engine-addr engine.delphix.com --sys-admin-usr
sysadmin --sys-admin-pwd sysadmin --admin-usr admin --admin-pwd delphix --hook-
ps-version ps2 --hook-templ-ps-version ps2
```

Please note that this will migrate all the hooks and hook templates in the Delphix Engine to PowerShell version two, the script does not have the functionality to perform the migration for specific hooks or hook templates.

4. An example command for migrating only the hook templates:

```
python <path to the script> --engine-addr engine.delphix.com --sys-admin-usr
sysadmin --sys-admin-pwd sysadmin --admin-usr admin --admin-pwd delphix --
migrate-only templates
```

5. An example command for running the script in debug mode:

```
python <path to the script> --engine-addr engine.delphix.com --sys-admin-usr
sysadmin --sys-admin-pwd sysadmin --admin-usr admin --admin-pwd delphix --debug
```

Common errors while running the script:

1. If the Delphix engine is not accessible from the machine where the script is run, the connectivity test run by the script prior to running the migration will fail, and the following error will be thrown:

```
Error occurred while connecting to the engine via the given Delphix System
Administrator user:
[Errno 8] nodename nor servname provided, or not known

Error occurred while connecting to the engine via the given Engine
Administrator user:
[Errno 8] nodename nor servname provided, or not known
```

The connectivity test run by the script prior to running the migration will fail if the value of the required parameters provided is not correct, and the following error will be thrown:

```
Error occurred while connecting to the engine via the given Engine
Administrator user:
HTTP status was 401 when doing POST '{"username": "admin", "password": "blah",
"type": "LoginRequest", "target": "DOMAIN"}' to '/resources/json/delphix/login':
 {"type":"ErrorResult","status":"ERROR","error":{"type":"APIError","details":"I
nvalid username or password.","action":"Try with a different set of
```

```
credentials.","id":"exception.webservices.login.failed","commandOutput":null,
"diagnoses":[]}}
```

2. If for some reason, the script execution stops in between before migrating all the hooks and hook templates, it's perfectly fine to run the script again; each time the script is run, the hooks and hook templates will be migrated to the specified PowerShell version.

SQL Server hook operation notes

SQL server clusters

When linking from, or provisioning to cluster environments, hook operations will not run once on each node in the cluster. Instead, the Delphix Engine always runs all hooks on the instance primary node.

Run powershell operation

The RunPowershell operation executes a PowerShell script on a Windows environment. The environment user runs this shell command from their home directory. The Delphix Engine captures and logs all output of the script. If the script fails, the output is displayed in the Delphix Management application and command-line interface (CLI) to aid in debugging.

If successful, the script must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Example of a run powershell Operation

You can input the full command contents into the Run powershell operation.

```
$removedir = $Env:DIRECTORY_TO_REMOVE

if ((Test-Path $removedir) -And (Get-Item $removedir) -is [System.IO.DirectoryInfo])
{
    Remove-Item -Recurse -Force $removedir
} else {
    exit 1
}
exit 0
```

SQL server environment variables

Operations that run user-provided scripts have access to environment variables. For operations associated with specific dSources or virtual databases (VDBs), the Delphix Engine will always set environment variables so that the user-provided operations can use them to access the dSource or VDB.

dSource environment variables

Environment Variables	Description
`SOURCE_INSTANCE_HOST`	The hostname of linked instance for the dSource
`SOURCE_INSTANCE_PORT`	Port of linked instance for the dSource
`SOURCE_INSTANCE_NAME`	Name of linked instance for the dSource
`SOURCE_DATABASE_NAME`	Name of database linked for the dSource

Staging variables

We have the following environment variables applicable to Staging Push dSources.

Environment Variables	Description
`STAGING_INSTANCE_HOST`	The hostname of the staging instance
`STAGING_INSTANCE_PORT`	Port number of the staging instance
`STAGING_INSTANCE_NAME`	Name of the staging instance
`STAGING_DATABASE_NAME`	Name of the staging database
`STAGING_MOUNT_BASE`	Mount path for the staging push dSource
`STAGING_DATA_DB_FILE_PATH`	Filepath of the staging database

VDB environment variables

Environment Variables	Description
`VDB_INSTANCE_HOST`	The hostname of linked instance for the VDB
`VDB_INSTANCE_PORT`	Port of linked instance for the VDB
`VDB_INSTANCE_NAME`	Name of linked instance for the VDB
`VDB_DATABASE_NAME`	Name of database linked for the VDB

Error handling for SQL server powerShell scripts

If a pre-script or post-script encounters an unrecoverable error during execution, the Delphix Engine expects the script to return with a non-zero exit code or the error will not be detected. The **Powershell -File** prefix and **exit $LASTEXITCODE** suffix are required to pass the script's exit code up to the layer calling the script.

Delphix does not perform error checking on PowerShell hook scripts. The script should perform error checking and logging, and return a non-zero exit code to indicate the script's failure. Failure to return a non-zero exit code when appropriate means that Delphix will think the hook script succeeded and mark

the VDB provision/refresh/rewind job as a success, when it should be seen as a failure. This is especially important when masking data is part of the hook – the VDB should not be released to users when the hook failed to mask data.

PowerShell gives you a few ways to handle errors in your scripts:

- Set `$ ErrorActionPreference`. This only applies to PowerShell Cmdlets. For scripts or other executables such as `sqlcmd`, PowerShell will return with exit code 0 even if there is an error, regardless of the value of `$ErrorActionPreference`. The allowable values for `$ErrorActionPreference` are :
  - `Continue` (default) – Continue even if there is an error
  - `SilentlyContinue` – Acts like Continue with the exception that errors are not displayed
  - `Inquire` – Prompts the user in case of error
  - `Stop` : Stops execution after the first error
- Use exception handling by using traps and try/catch blocks to detect errors and return with non-zero exit codes
- Use custom error handling that can be invoked after launching each command in the script to correctly detect errors. The following example shows how you can use the function verifySuccess to detect whether the previous command failed, and if it did print, print an error message and return with an exit code of **1**.

```
function die {
    Write-Error "Error: $($args[0])"
    exit 1
}
function verifySuccess {
    if (!$?) {
        die "$($args[0])"
    }
}
Write-Output "I'd rather be in Hawaii"
verifySuccess "WRITE_OUTPUT_FAILED"
& C:\Program Files\Delphix\scripts\myscript.ps1
verifySuccess "MY_SCRIPT_FAILED"
```

Using pre- and post-scripts with SQL Server dSources

Overview

This topic describes the use of pre- and post-scripts with dSources that are created from SQL Server source databases.

Pre-scripts and post-scripts are Windows PowerShell code executed on the Staging Target host before and after a SnapSync of a dSource. You can specify pre- and post-scripts in the wizard for creating a dSource, or you can modify them afterward by navigating to the Configuration > Standard tab. You can also set pre- and post-scripts using the Delphix command-line interface (CLI) or REST Web API.

The Delphix Engine executes a pre-script on the Staging Target host prior to the SnapSync of a dSource. If it is an initial snapshot, or manual snapshot by selecting the snapshot button on the GUI, the pre/post scripts do not get executed for dSources. Since the dSource resides within a restoring database in the SQL instance on the Staging Target host, the script can perform queries on the instance if a database account is available.

Hooks do allow for a pre/post snapshot hook, but the pre/post scripts do not

A post-script is executed after the SnapSync on a dSource completes. If the post-script fails, the provision, refresh, or rewind operation will also fail with an error message, and a fault will be created on the dSource.

> ⚠ Pre/Post hooks allow for pre/post snapshot hooks for which there is no equivalent in the old pre/post scripts. Pre/Post scripts are run during validated sync, for which there is no equivalent hook operation today.
> Pre- and post-scripts are supported for backward compatibility with older versions of the Delphix Engine.

Associating scripts with a dSource

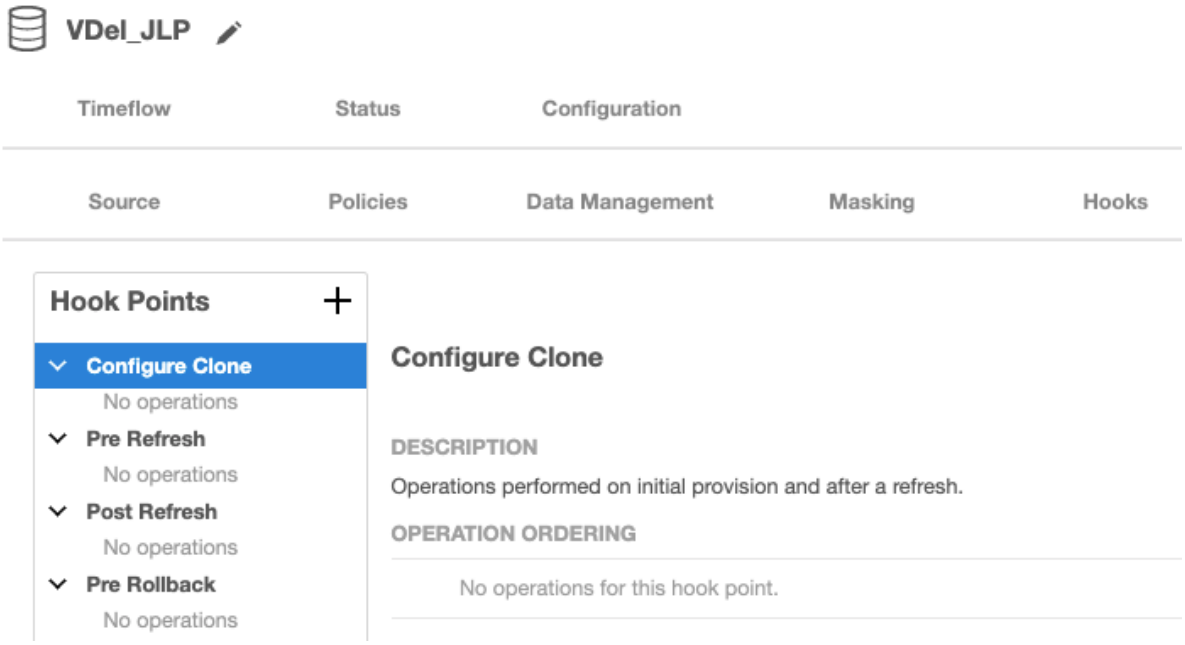
Pre- and Post-scripts can be associated with a dSource in one of two ways:

During the dSource linking process

1. Login to the **Delphix Management** application.
2. In the top menu bar, click **Manage**.
3. Select **Datasets**.
4. Select the **Plus** icon and then select **Add dSource**.
5. In the **Hooks** tab of the **Add dSource** wizard, select the Plus icon to add new **Pre Script** and **PostScript** hooks.
6. Enter the calling syntax of the Windows PowerShell script into either or both of the appropriate fields.
    a. The calling environment is used during linking, as shown in the **Environment Details** panel of the Delphix environment (**Manage** > **Environments**)
    b. Four (4) environment variables will be populated with the name of the Delphix dSource, the SQL Server instance name and port, and the SQL Server database
    c. name.

After linking, using the configuration tab of the datasets details page

1. Login to the **Delphix Management** application.
2. In the top menu bar, click **Manage**.
3. Select **Datasets** to display the SQL Server dSources and VDBs.
4. Select a SQL Server **dSource** from the listed **Datasets** in the left-hand navigation bar.
5. Click the **Configuration** panel, select the Hooks sub-tab.
6. By selecting a Plus icon you can create hooks from a template or create a new hook.



7. Enter the calling syntax of the Windows Powershell script into either or both of the appropriate fields
   a. The calling environment is that of the **primary Environment User account**, as shown in the **Environment Details** panel of the Delphix environment (**Manage** > **Environments**)
   b. Four (4) environment variables will be populated with the name of the VDB, the SQL Server instance name and port, and the SQL Server database name.
   c. Select the **Create** to accept the change.

Execution context for SQL server scripts

For dSources, pre- and post-scripts are executed in the context of the staging host user that was provided when linking.

Using pre- and post-scripts with SQL Server VDBs

Overview

This topic describes the use of pre- and post-scripts with virtual databases (VDBs) that are created from SQL Server dSources.

Pre-scripts and post-scripts are Windows PowerShell code executed on the VDB target host before and after the provision, refresh, or rewind of a VDB. You can specify pre- and post-scripts in the wizard for creating a VDB, or you can modify them afterward by navigating to the Configuration > Standard tab. You can also set pre- and post-scripts using the Delphix command-line interface (CLI) or REST Web API.

The intent of these scripts is a customization of the data contents or configuration of a VDB while it is being manipulated. Actions performed by pre-scripts and post-scripts effectively become an integrated part of the provision, refresh, or rewind actions for that VDB.

The pre-script executes during the initial provision of a VDB. During refresh and rewind, the PowerShell script referenced in a pre-script is executed after the VDB has been stopped and unmounted, but before the new VDB is mounted. If the pre-script fails, the refresh or rewind operation will also fail with an error message.

During provision, refresh, and rewind, the PowerShell script referenced in a post-script is executed after the Delphix engine has mounted and started the VDB. If the post-script fails, the provision, refresh, or rewind operation will also fail with an error message, and a fault will be created on the VDB.

You can use a pre-script to capture configuration file settings, but not the contents of the soon-to-be recreated VDB; a pre-script executes too late to access the VDB which has already been shut down and unmounted. This makes pre-script functionality much less useful than hook operations like Pre-Refresh.

You can use a post-script to run data transformation operations on newly-provisioned, newly-refreshed, or newly-rewound VDBs. These operations include data masking and setting non-production account passwords in place of cloned production passwords.

> Pre- and post-scripts are an older customization mechanism for SQL Server virtual databases. They have been replaced by hook operations, which have been the standard customization mechanism on all other data platforms.
>
> Pre- and post-scripts are supported for backward compatibility with older versions of the Delphix Engine. Delphix encourages everyone to use Hooks for customizing SQL Server VDBs for future implementations, if possible.

Associating scripts with a VDB

Pre- and Post-scripts can be associated with a VDB in one of two ways:

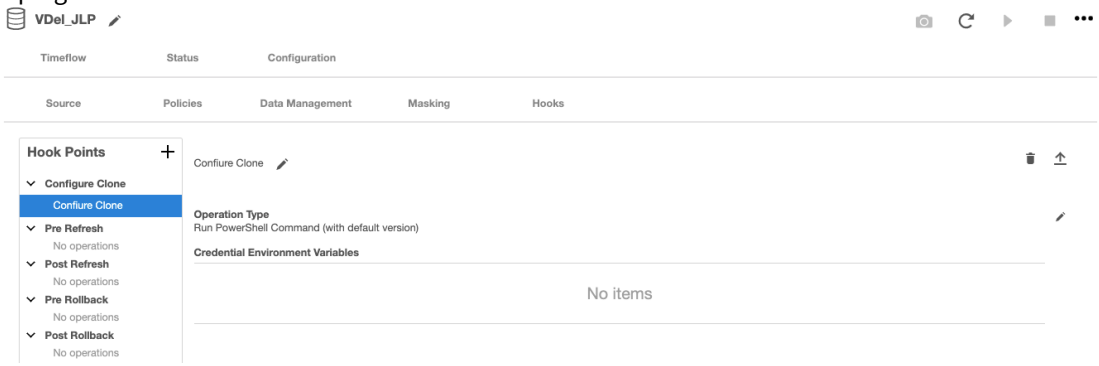
During the VDB provisioning process

1. Login to the **Delphix Management** application/
2. In the top menu bar, click **Manage**.
3. Select **Datasets** to display the SQL Server dSources and VDBs.
4. From the listed **Datasets** in the left-hand navigation bar, select a SQL Server **dSource** or **VDB**.
5. Click the **TimeFlow**.
6. Click **Provision**.
7. In the first **Target Environment** step of the **Provision VDB** wizard, there are fields for **Pre Script** and **Post Script**.
8. Enter the calling syntax of the Windows PowerShell script into either or both of the appropriate fields.

a. The calling environment is that of the **primary Environment User account**, as shown in the **Environment Details** panel of the Delphix environment (**Manage** > **Environments**)
b. Four (4) environment variables will be populated with the name of the VDB, the SQL Server instance name and port, and the SQL Server database name.

After provisioning, using the configuration tab of the Datasets details page

1. In the **Datasets** panel, click the virtual dataset.
2. Click the **Configuration** tab.
3. Within the **Configuration** tab, click the **Hooks** tab.
4. Select the **hook** to edit.
    a. Click the **Plus** icon to add a new operation.
    b. Select the **type of operation** or click to load a hook operation template.
    c. Click the **text area** and edit the contents of the operation.
    d. To remove an operation from the list, click the **Trash** icon on the operation.
    e. When you have set all hook operations, click the **checkmark** to save the changes.The current operations at this hook will be displayed. To edit this list of operations, click the **Pencil** icon in the top right-hand corner of the tab.



Execution context for SQL server scripts

For VDBs, pre- and post-scripts are executed in the context of the environment user that was selected during the VDB provision and not the primary environment user. The primary environment user can change over time, all VDB operations are done using the user that was initially selected.

Error handling for SQL server powerShell scripts

If a pre-script or post-script encounters an unrecoverable error during execution, the Delphix Engine expects the script to return with a non-zero exit code. Otherwise, the error will not be detected.

PowerShell gives you a few ways to handle errors in your scripts:

- Set `undefinedErrorActionPreference`. The allowable values for `$ErrorActionPreference` a re:
    - `Continue` (default) – Continue even if there is an error
    - `SilentlyContinue` – Acts like Continue with the exception that errors are not displayed
    - `Inquire` – Prompts the user in case of error
    - `Stop` : Stops execution after the first error
- Use exception handling by using traps and try/catch blocks to detect errors and return with non-zero exit codes

- Use custom error handling that can be invoked after launching each command in the script to correctly detect errors. The following example shows how you can use the function verifySuccess to detect whether the previous command failed, and if it did print, print an error message and return with an exit code of **1**.

```
function die {
    Write-Error "Error: $($args[0])"
    exit 1
}
function verifySuccess {
    if (!$?) {
        die "$($args[0])"
    }
}
Write-Output "I'd rather be in Hawaii"
verifySuccess "WRITE_OUTPUT_FAILED"
& C:\Program Files\Delphix\scripts\myscript.ps1
verifySuccess "MY_SCRIPT_FAILED"
```

Using hostChecker to validate target database servers

Prerequisites

Make sure that your source and target environments meet the requirements.

Procedure to validate target environments

1. Verify with your System Administrator that the Delphix Connector has been installed in all Target environments.
2. Login to the Windows target host using the **Windows user account** that the System Administrator configured as a Delphix target user.
3. Open **Windows Powershell** using the **Run as Administrator** option.
4. Execute the **host checker script**by running:

   ```
   <Delphix Connector installation folder>\etc\dlpx-host-checker.ps1
   ```

5. Select a path where a report file will be saved, such as **C:\temp\delphix-host-checker-report.txt**.
6. Select the **default** option of **Target Host**.
7. Read the output of the checks.
8. The error or warning messages will explain any possible problems and how to address them. Resolve the issues that the HostChecker describes. Do not be surprised or undo your work if more errors appear the next time you run HostChecker; the error you just fixed may have been masking other problems.
9. Repeat steps 4–7 until all the checks return no errors or warnings.

Tests run

Test	SQL Server Source	SQL Server Target	Description
Check Powershell Version	X	X	Verifies that Powershell 2.0 or greater is installed
Check OS User Privileges	X	X	For target hosts, verifies that the operating system (OS) user has administrative rights. For source hosts, verifies that the OS user can successfully perform remote registry access from the target host to the source host.
Check host settings	X	X	Verifies that the Delphix Engine can discover host environment details from the Windows registry.
Check SQL Server instance discovery	X	X	Verifies that the Delphix Engine can discover SQL Server instances.

Test	SQL Server Source	SQL Server Target	Description
Check SQL Server instance login permission	X	X	For target hosts, verifies that the Windows OS user can be used to log in to the SQL Server instances. For source hosts, verifies that the supplied SQL Server login credentials can be used to log in to the SQL Server instances.
Check database discovery	X	X	Verifies that the Delphix Engine can discover SQL Server databases.

Additional options

Run the following to view additional HostChecker options:

```
dlpx-host-checker.ps1 -?
```

Unstructured files and app data

This section contains the following topics:

Getting started with unstructured files

The term “unstructured files" refers to data stored in a filesystem that is NOT usually accessed by a DBMS or similar software. Unstructured files can consist of anything from a simple directory to the root of a complex application like Oracle Enterprise Business Suite. Like with other data types, you can configure a dSource to sync periodically with a set of unstructured files external to the Delphix Engine. The dSource is a copy of these physical files stored on the Delphix Engine. On Unix platforms, dSources are created and periodically synced by an implementation of the rsync utility. On Windows, files are synced using the robocopy utility, which is distributed with Windows. dSources enable you to provision “vFiles,” which are virtual copies of data that are fully functional read-write copies of the source of the original file. You can mount vFiles across one target environment or many.

Unstructured files environment requirements

This section contains the following topics:

Unstructured files on unix environments

This section contains the following topics:

Requirements for Unix environments

This topic outlines the supported operating systems (OSs) for use on UNIX source and target environments.

Supported operating systems

Operating System	Version	Processor Family
Solaris	10, 11	SPARC x86_64
Red Hat Enterprise Linux	5.5 - 5.11 6.1 - 6.10 7.0 - 7.8 8.0	x86_64
Oracle Enterprise Linux	5.5 - 5.11 6.1 - 6.10 7.0 - 7.8 8.0	x86_64
SUSE Linux Enterprise Server	11, 11SP1	x86_64
AIX	6.1, 7.1, 7.2	Power
HP-UX	11i v3 (11.31)	IA64
Ubuntu	18, 20	X86_64

Delphix supports all 64-bit OS environments for source and target.

PHNE_37851 - resolves a known bug in HP-UX NFS client prior to HP-UX 11.31.

Additional source environment requirements

- There must be an operating system user with these privileges. For example, here in this section delphix_os is the primary user for the environment:
  - Ability to login to the source environment via SSH
- There must be a directory on the source environment where you can install the Delphix platform Toolkit – for example, `/var/opt/delphix/toolkit`.
  - The **delphix_os**user must own the directory
  - The directory must have permissions -rwxrwx--- (0770), but you can also use more permissive settings
  - The **delphix_os** user must have read and execute permissions on each directory in the path leading to the toolkit directory. For example, when the toolkit is stored in `/var/opt/delphix/toolkit`,

the permissions on `/var`, `/var/opt`, and `/var/opt/delphix` should allow read and execute for "others," such as -rwxr-xr-x.
    - The directory should have 1.5GB of available storage: 400MB for the toolkit and 400MB for the set of logs generated by each client that runs out of the toolkit
- On a Solaris host, `gtar` must be installed. Delphix uses `gtar` to handle long file names when extracting the toolkit files into the toolkit directory on a Solaris host. The `gtar` binary should be installed in one of the following directories:
    - `/bin:/usr`
    - `/bin:/sbin:/usr`
    - `/sbin:/usr/contrib`
    - `/bin:/usr/sf`
    - `/bin:/opt/sfw`
    - `/bin:/opt/csw/bin`
- The Delphix Engine must be able to initiate an SSH connection to the source environment

Additional target environment requirements

- There must be an operating system user with these privileges. Here we use delphix_os as an example for the primary user for the environment:
    - Ability to login to the target environment via SSH
    - The following permissions are usually granted via sudo authorization of commands.
    - See Sudo Privilege Requirements for further explanation of the commands and Sudo File Configuration Examples for Unstructured Files on Unix for examples of the /etc/sudoers file on different operating systems.
        - The primary user for the target environment must have the ability to run mount, unmount, mkdir, and rmdir as a super-user on any directory mounted to by the Delphix Engine.
        - If the target host is an AIX system, permission to run the nfso command as a super-user
    - There must be a directory on the target environment where you can install the Delphix Engine Toolkit - for example, /var/opt/delphix/toolkit.
        - The **delphix_os**user must own the directory
        - The directory must have permissions -rwxrwx--- (0770), but you can also use more permissive settings
        - The **delphix_os** user must have read and execute permissions on each directory in the path leading to the toolkit directory. For example, when the toolkit is stored in `/var/opt/delphix/toolkit`, the permissions on `/var`, `/var/opt`, and `/var/opt/delphix` should allow read and execute for "others," such as -rwxr-xr-x.
        - The directory should have a total of at least 800MB of storage, plus 1MB of storage per vFile that will be provisioned to the target
- On a Solaris host, `gtar` must be installed. Delphix uses `gtar` to handle long file names when extracting the toolkit files into the toolkit directory on a Solaris host. The `gtar` binary should be installed in one of the following directories:
    - `/bin:/usr`
    - `/bin:/sbin:/usr`
    - `/sbin:/usr/contrib`
    - `/bin:/usr/sf`
    - `/bin:/opt/sfw`

    - `/bin:/opt/csw/bin`
- There must be a directory under which the mount points are created when provisioning a vFile to the target environment. The group associated with the directory must be the primary group of the delphix-os user. Group permissions for the directory should allow read, write, and execute by members of the group.
- The Delphix Engine must be able to initiate an SSH connection to the target environment.
- NFS client services must be running on the target environment.
    - Required packages on target hosts:
        i. Portmapper (rpcbind)
        ii. status daemon (rpc.statd)
        iii. NFS lock manager (rpc.lockd/lockmgr)
    - The Delphix Engine enables the use of NFSv3 by default for mounting target host filesystems, thus, the prerequisite packages to support NFSv3 client communication are required for normal operation. In addition, the required services to support NFS client communications (including file locking) must be running. These services are shown in the left column.
        - To enable NFSv4, which does not need to interact with those discrete services, See NFSv4 configuration.

Network and connectivity requirements for Unix environments

Port allocations specific to unstructured files

The Delphix Engine makes use of the following network ports for unstructured files dSources and vFiles:

Inbound to the Delphix Engine port allocation

Protocol	Port Number	Use
TCP	873	Rsync connections used for communication to `rsyncd` during SnapSync
TCP/UDP	111	Remote Procedure Call (RPC) port mapper used for NFSv3 mounts Note: RPC calls in NFSv3 use additional fixed ports for supporting services (lockd, mountd and statd) seen below.
TCP	1110	NFS Server daemon status and NFS server daemon keep-alive (client info)
TCP	2049	NFS Server daemon from vFiles to the Delphix Engine
TCP	54043	NFSv3 mount daemon
TCP	54044	NFSv3 stat daemon (lock state notification service)
TCP	54045	NFSv3 lock daemon/manager
UDP	33434 - 33464	Traceroute from source and target database servers to the Delphix Engine (optional)

Outbound from a source environment port allocation

Protocol	Port Numbers	Use
TCP	873	Rsync connections used during SnapSync
TCP	xxxx	DSP connections used for monitoring and script management during SnapSync. Typically DSP runs on port 8415.

Inbound to a source environment port allocation

Protocol	Port Numbers	Use
TCP	22	SSH connections to the source environment

Outbound from a target environment port allocation

Protocol	Port Numbers	Use
TCP	873	Rsync connections used during V2P
TCP	xxxx	DSP connections used for monitoring and script management. Typically DSP runs on port 8415.

Inbound to a target environment port allocation

Protocol	Port Numbers	Use
TCP	22	SSH connections to the target environment

General outbound from the Delphix Engine port allocation

Protocol	Port Numbers	Use
TCP	25	Connection to a local SMTP server for sending email
TCP/UDP	53	Connections to local DNS servers
UDP	123	Connection to an NTP server
UDP	162	Sending SNMP TRAP messages to an SNMP Manager
TCP	443	HTTPS connections from the Delphix Engine to the Delphix Support upload server
TCP/UDP	636	Secure connections to an LDAP server
TCP	8415	Connections to a Delphix replication target. See Configuring Replication.
TCP	50001	Connections to source and target environments for network performance tests.

General inbound to the Delphix Engine port allocation

Protocol	Port Number	Use
TCP	22	SSH connections to the Delphix Engine
TCP	80	HTTP connections to the Delphix GUI
UDP	161	Messages from an SNMP Manager to the Delphix Engine
TCP	443	HTTPS connections to the Delphix Management Application
TCP	8415	Delphix Session Protocol connections from all DSP-based network services including Replication, SnapSync for Oracle, V2P, and the Delphix Connector.
TCP	50001	Connections from source and target environments for network performance tests via the Delphix CLI.

Firewalls and Intrusion Detection Systems (IDS)

Production databases on source environments (for dSources) are often separated from the non-production environment by firewalls. Firewalls can add milliseconds to the latency between servers. Accordingly, for best performance, there should be no firewalls between the Delphix Engine and the virtual database (VDB) target environments. If the Delphix Engine is separated from a source environment by a firewall, the firewall must be configured to permit network connections between the Delphix Engine and the source environments for the application protocols (ports) listed above.

Intrusion detection systems (IDSs) should also be made permissive to the Delphix Engine deployment. IDSs should be made aware of the anticipated high volumes of data transfer between dSources and the Delphix Engine.

Sudo privilege requirements for unstructured files on Unix

This topic describes `sudo` file privilege configurations necessary for interacting with the Delphix Engine when virtualizing unstructured files on Unix Environments.

Sudo privilege rationale

Privilege	Sources	Targets	Rationale
`mkdir/rmdir`	Not Required	Optional	Delphix dynamically makes and removes directories under the provisioning directory during vFiles operations. This privilege is optional, provided the provisioning directory permissions allow the **delphix_os** user to make and remove directories.
`mount/ umount`	Not Required	**Required**	Delphix dynamically mounts and unmounts directories under the provisioning directory during vFiles operations. This privilege is required because `mount` and `umount` ar e typically reserved for a super-user.
`nfso` (AIX only)	Not Required	**Required**	Delphix monitors NFS read and write sizes on an AIX target host. It uses the `nfso` command to query the sizes in order to optimize NFS performance for vFiles running on the target host. Only a super-user can issue the `nfso` comman d.

Sudo file configuration examples for unstructured files on Unix

This topic describes `sudo` file privilege configurations necessary for interacting with the Delphix Engine when virtualizing unstructured files on Unix Environments.

> **Considerations for sudo access and account locking**
> The Delphix Engine tests its ability to run the `mount` command using `sudo` on the target environment by issuing the `sudo mount` command with no arguments. Many of the examples shown in this topic do not allow that. This causes a warning during environment discovery and monitoring but otherwise does not cause a problem. If your vFiles operations succeed, it is safe to ignore this warning.
> However, some users configure the security on the target environments to monitor `sudo` failures and lockout the offending account after some threshold. In those situations, the failure of the sudo commands might cause the **delphix_os** account to become locked. One workaround for this situation is to increase the threshold for locking out the user account. Another option is to modify `/etc/sudoers` to permit the **delphix_os** user to run the `mount` command without parameters.

Configuring `sudo` access on Solaris for unstructured files

On a Solaris SPARC target, `sudo` access to `mount`, `umount`, `mkdir`, and `rmdir` is required. In this customer example, super-user privileges are restricted to the virtual dataset mount directory /delphix and are further restricted to commands which mount data from a single Delphix Engine with IP address 100.245.235.12.

Delphix requires `umount -f` for emergency force unmounts on Solaris.

Additionally, sudo access to `ps` may be added to avoid warnings during discovery but is not required.

Example: Solaris /etc/sudoers entries for a Delphix Target for Unstructured Files

```
User_Alias DELPHIX_USER=delphix_os

Cmnd_Alias DELPHIX_CMDS= \
/usr/sbin/mount     100.245.235.12\:* /delphix/*, \
/usr/sbin/mount * 100.245.235.12\:* /delphix/*, \
/usr/sbin/umount    /delphix/*, \
/usr/sbin/umount *  /delphix/*, \
/usr/sbin/umount -f /delphix/*, \
/usr/bin/mkdir      /delphix/*, \
/usr/bin/mkdir -p   /delphix/*, \
/usr/bin/rmdir      /delphix/*
/usr/bin/ps
DELPHIX_USER ALL=(ALL) NOPASSWD: DELPHIX_CMDS
```

Configuring `sudo` access on Linux for unstructured files

On a Linux target, sudo access to `mount`, `umount`, `mkdir`, and `rmdir` is required. In this customer example, super-user privilege is restricted to the virtual database mount directory `/delphix`. Aliases are used to restrict the Delphix Engines which are allowed to run these commands.

Delphix requires `umount -lf` for emergency force unmounts on Linux.

Example: Linux /etc/sudoers file for a Delphix Target for Unstructured Files

```
Defaults:delphix_os !requiretty

Cmnd_Alias DELPHIX_ADMIN_CMDS= \
/bin/mount           /delphix/*, \
/bin/mount   *       /delphix/*, \
/bin/umount          /delphix/*, \
/bin/umount  *       /delphix/*, \
/bin/umount  -lf     /delphix/*, \
/bin/mkdir -p -m 755 /delphix/*, \
/bin/mkdir -p        /delphix/*, \
/bin/mkdir           /delphix/*, \
/bin/rmdir           /delphix/*
/bin/ps
Host_Alias DELPHIX_HOSTS=delphix001, delphix002
delphix_os DELPHIX_HOSTS=NOPASSWD:DELPHIX_ADMIN_CMDS
```

Configuring `sudo` access on AIX for unstructured files

In addition to sudo access to the `mount`, `umount`, `mkdir`, and `rmdir` commands on AIX target hosts, Delphix also requires sudo access to `nfso`. This is required on target hosts for Delphix to monitor the NFS read/write sizes configured on the AIX system. The super-user access level is needed to run the `nfso` command. This example does not restrict the Delphix Engine which is allowed to run these commands.

Delphix requires `umount -f` for emergency force unmounts on AIX.

Example: AIX /etc/sudoers File for a Delphix Target for Unstructured Files

```
Defaults:delphix_os !requiretty
delphix_os ALL=NOPASSWD: \
/bin/mount, \
/bin/umount, \
/bin/mkdir, \
/bin/rmdir, \
/usr/sbin/nfso, \
/usr/bin/ps
```

Configuring `sudo` access on HP-UX for unstructured files

On the HP-UX target, as with other operating systems, `sudo` access to `mount`, `umount`, `mkdir`, and `rmdir` is required. This example does not restrict the Delphix Engine which are allowed to run these commands.

Example: HP-UX /etc/sudoers file for a Delphix Target for Unstructured Files

```
Defaults:delphix_os !requiretty
```

```
delphix_os ALL=NOPASSWD:/sbin/mount, /sbin/umount, /bin/mkdir, /bin/rmdir, /bin/ps
```

Adding a Unix environment

This topic describes how to add a new Unix environment.

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Click the **Plus** icon next to **Environments**.
5. In the **Add Environment** dialog, select **Unix/Linux**.
6. Select **Standalone Host**.
7. Enter the **Host IP address**.
8. Enter an optional **Name** for the environment.
9. Enter the **SSH** port. The default value is **22**.
10. Enter a **Username** for the environment.
11. Select **Login Type**.
    a. Password - enter the OS password associated with the user in Step 10, or
    b. Public Key, or
    c. Password Vault - select from an existing Enterprise Password Vault
    Note:
    Using Public Key Authentication
    If you want to use public-key authentication for logging into your Unix-based environment, there are two options: use the engine's key pair or provide a key pair for this environment.
    To use the engine's key pair:
        i. Select **Public Key** for the **Login Type.**
        ii. Click **View Public Key**.
        iii. Copy the public key that is displayed, and append it to the end of your `~/.ssh/authorized_keys` file. If this file does not exist, you will need to create it.
            1. Run chmod 600 ~/.ssh/authorized_keys to allow only the file's owner to read and write to it (make sure the file is owned by the user).
            2. Run chmod 755 ~ to restrict access to the user's home directory so no other user may write to it.
            3. Run chmod 700 ~/.ssh so that others cannot write to it. The ~/.ssh directory cannot be writable by group or other users. Otherwise, authentication will fail.
    As an alternative, you can provide a key pair specific for this environment via the API or CLI.
12. For **Password Login**, click **Verify Credentials** to test the username and password.
13. Enter a **Toolkit Path**. The toolkit directory stores scripts used for Delphix Engine operations. It should have a persistent working directory rather than a temporary one.
14. Click **Submit**.

Post-Requisites

After you create the environment, you can view information about it by doing the following:

1. Click **Manage**.
2. Select **Environments**.
3. Select the **environment name**.

Unstructured files on windows environments

This section contains the following topics:

Requirements for Windows environments

Supported operating systems

- Windows Server 2012, 2012 R2
- Windows Server 2016
- Windows Server 2019
- Windows Server 2022

Requires 64-Bit Windows

Delphix must install the Delphix Connector on all Windows hosts that Delphix will directly communicate with. This means all target hosts, and source or staging hosts. The Delphix Connector only supports 64-bit versions of Windows.

See Options for linking unstructured files on Windows environments for more information about source vs. staging hosts.

Additional source or staging environment requirements

- The Delphix Connector must be installed on the source or staging environment. You must have used the Delphix Connector to register this environment with the Delphix Engine.
- The `robocopy` utility must be installed on the source or staging Windows environment. `robocopy` is installed by default on Windows Server 2008, Windows Vista, Windows 7, and Windows 8. For other versions of Windows, it is available by downloading a resource kit from Microsoft.
- If using a staging environment, the source's files must be made available and readable to the environment user from the staging environment via a UNC path. For example, use Windows Sharing.

Additional target environment requirements

- The Delphix Connector must be installed on the target environment. You must have used the Delphix Connector to register this environment with the Delphix Engine.

Procedure for adding and installing the Delphix connector for Windows

All Windows environments that will communicate with Delphix must have the Delphix Connector installed. The instructions in this topic cover downloading Delphix Connector, running the Delphix Connector installer on the Windows machine, and then registering the environment with the Delphix Engine.

Procedure

Downloading the Delphix connector

Delphix Connector software supplied by Delphix Engine versions before 4.2.4.0 required that the Windows machine had SQL Server installed. If you are using a Windows machine that does not have SQL Server installed, you must download the Delphix Connector from a Delphix Engine of version 4.2.4.0 or higher.

The Delphix Connector can be downloaded through the Delphix Engine Interface, or by directly accessing its URL.

Using the Delphix Engine interface

A Flash player must be available on the Windows host in order to download Delphix Connector using the Delphix GUI.

1. From the Windows machine that you want to use, start a browser session and connect to the **Delphix Management** application using the delphix_admin login.
2. Click **Manage**.
3. Select **Environments**.
4. Next to **Environments**, click the **Plus** icon.
5. In the **Add Environment** dialog, select **Windows** in the operating system menu.
6. Select **Target**.
7. Select **Standalone**.
8. Click the download link for the **Delphix Connector Installer**. The Delphix Connector will download to your local machine.

Direct download

1. You can download the Delphix Connector directly by navigating to this URL: `http://<name of your Delphix Engine>/connector/DelphixConnectorInstaller.exe`

Installing Delphix connector

On the Windows machine that you want to want to use, run the Delphix Connector installer. Click **Next** to advance through each of the installation wizard screens.

The installer will only run on 64-bit Windows systems. 32-bit systems are not supported.

1. For **Connector Configuration**, make sure there is no firewall in your environment blocking traffic to the port on the Windows environment that the Delphix Connector service will listen to.
2. For **Select Installation Folder**, either accept the default folder or click **Browse** to select another.
3. Click **Next** on the installer final **Confirm Installation** dialog to complete the installation process and then **Close** to exit the Delphix Connector Install Program.
4. Note: At this point, you can close the Delphix GUI dialog by clicking **Cancel**.

Registering environment with Delphix Engine

1. Return to the Delphix Management application.
2. Enter the **Environment Name**, **Host Address**, **Delphix Connector Port**, **OS Username**, and **OS Password** for the target environment.
3. To provide your own Oracle Java select the **Provide my own JDK** checkbox and click **Next**.
4. In the Java Development Kit tab, enter the absolute path to your Oracle JDK and click **Next**.
5. Click **Submit.**

As the new environment is added, you will see two jobs running in the **Delphix Admin Job History**, one to **Create and Discover** an environment, and another to **create** an environment. When the jobs are complete, you will see the new environment added to the list in the **Environments** panel.

Post-Requisites

- On the Windows machine, in the **Windows Start Menu**, go to **Services > Extended Services**, and make sure that the **Delphix Connector** service has a **Status** of **Started**, and that the **Startup Type** is **Automatic**.

Windows iSCSI configuration requirements

Windows iSCSI configuration requirements are split into two types. These requirements are needed on both staging and target servers.

1. iSCSI configuration required for operational stability.
2. Optional iSCSI parameters for performance improvement.

> When target environments are discovered, Delphix will configure the Microsoft iSCSI Initiator Service for Automatic startup.

iSCSI configuration required for operational stability

The following Microsoft iSCSI Initiator configuration parameters are required for the target and staging Hosts. For details about configuring registry settings, see How to Modify the Windows Registry.

> You must reboot the Windows server reboot after changing the iSCSI configuration parameters.

Registry Key	Registry Value	Type	Data
`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\iSCSI\Discovery`	MaxRequestHoldTime	REG_DWORD	0x384 (900)
`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Disk`	TimeOutValue	REG_DWORD	0x384 (900)
`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Class\{4D36E97B-E325-11CE-BFC1-08002BE10318}\<Instance Number>\Parameters`	MaxRequestHoldTime	REG_DWORD	0x12C (300)

These settings will improve operational stability for VDBs and staging databases. If these settings are not adjusted, SQL Server may raise errors if VDBs are accessed during a temporary infrastructure outage. Affected VDBs may need to be manually restarted using the Continuous Data Engine.

Delphix Knowledge Base article KB1251 includes scripts to validate or set registry parameters so that they meet current Delphix recommendations.

Optional iSCSI parameters for performance improvement

The following iSCSI Registry setting may improve SQL Server dSource and VDB performance on the staging and target hosts.

Registry Key	Registry Value	Type	Data
`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\<Interface GUID>`	TcpAckFrequency	REG_DWORD	0x1 (1)

This setting is recommended for storage networks in Microsoft's TechNet article iSCSI and the Nagle Algorithm, described in Microsoft's document TcpAckFrequency to control the TCP ACK behavior.

In some environments, adjusting this setting may not improve performance compared to Windows defaults. Modifications to this registry parameter should be tested in each environment, to confirm that this provides a performance improvement.

Delphix Engine validation for Windows iSCSI configuration

Delphix Engine validates the Windows iSCSI Configurations that are set on any supported windows staging and target host with the Delphix recommended configurations while performing the following operations:

1. Add environment operation
2. Refresh environment operation
3. Enable environment operation

Prerequisites

1. Supported if you are using Powershell 3.0 or above - If you are on Powershell version below 3.0, then the job will be updated with a warning that the Powershell version on your host is not supported for validating iSCSI parameters.
2. The below alerts are applicable only for staging or target Windows hosts.

Additional Information

1. Delphix Engine will only validate and will not alter any configuration in the user environment.
2. On update of registry values on the target host to match Delphix recommendations, the faults from the Delphix engine will only be resolved if any of the operations (environment add, refresh or enable) is performed. Delphix engine will not monitor the state of the target host in the background and hence any change will not be picked up unless an operation is triggered. So, the user needs to take action for the change to reflect in faults.
3. On a successful Delphix Engine upgrade, the latest default iSCSI recommendations will be used for validations.

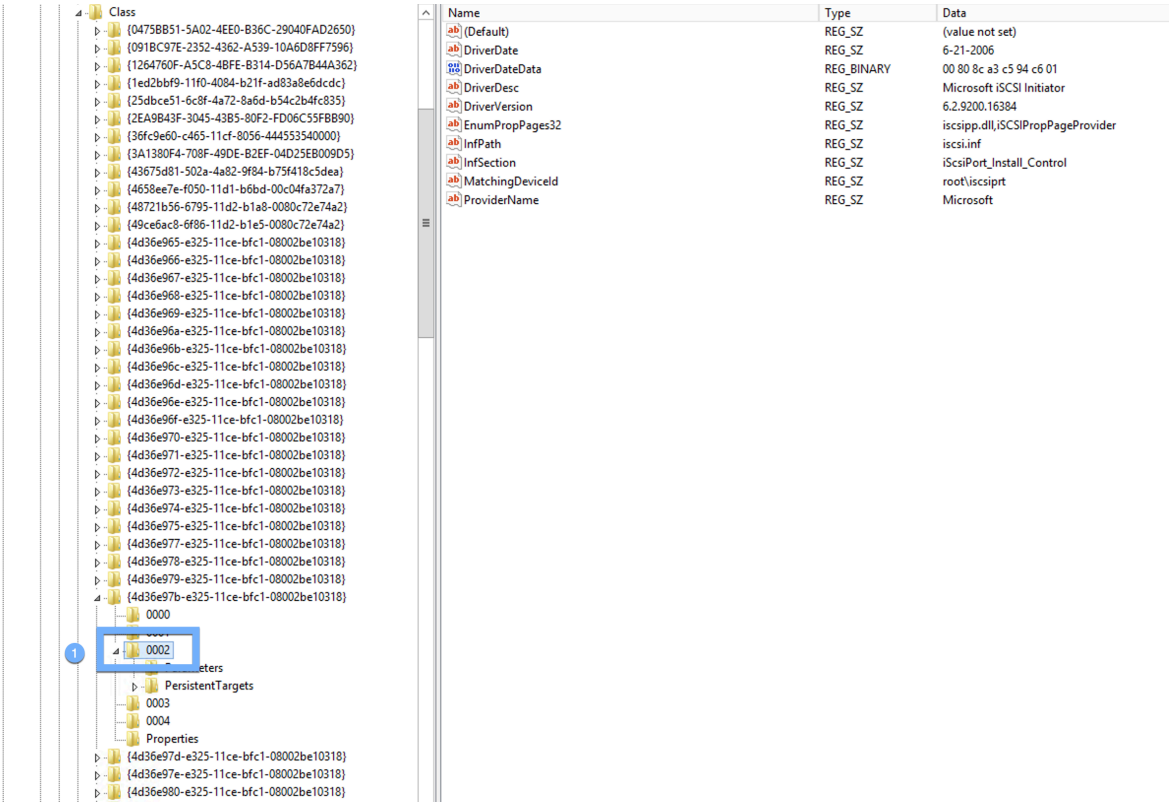
Troubleshooting

Type		Description
Fault(Severity = Warning)	`ENVIRONMENT_ISCSI_CONFIG_MISMATCH`	The single fault is thrown for all mismatched parameters
	`ISCSI_FETCH_CONFIG_PARAM_FAILURE`	The single fault is thrown for all parameters where we failed to fetch the value at the target host

Type		Description
Warning	`ISCSI_CONFIG_PARAM_TIMEOUT`	Job warning raised if we are unable to get the iSCSI parameters on the host within 5 minutes. No fault thrown at this point.
	`ISCSI_PS_VERSION_NOT_SUPPORTED`	Job warning raised if the PowerShell version is below 3 on the host side during the validation of the iSCSI parameters. No fault thrown at this point. The validation is skipped.
	`ENVIRONMENT_ISCSI_CONFIG_MISMATCH`	Job warning added for all mismatched parameters
	`ISCSI_FETCH_CONFIG_PARAM_FAILURE`	Job warning for all parameters where we failed to fetch the value at the target host

Identifying the instance number for iSCSI control class initiator drivers

1. From the Windows toolbar, click **Start** and select **Run** from the menu.
2. Type `regedit` in the **Open** field and click **OK.**
3. Go to the following registry key: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Class\{4D36E97B-E325-11CE-BFC1-08002BE10318}\<Instance Number> where the value of <Instance Number> is the one that shows a DriverDesc value of Microsoft iSCSI Initiator. Under the registry key, locate and expand the plus (+) sign next to the instance number. In the example below, the value of the instance number is 0002.



Value of Instance Number

Network and connectivity requirements for Windows environment

Port allocations specific to unstructured files

The Delphix Engine makes use of the following network ports for unstructured files dSources and VDBs:

Outbound from the Delphix Engine

Protocol	Port Numbers	Use
TCP	xxxx	Delphix Connector connections to source and target environments. Typically, the Delphix Connector runs on port 9100.

Inbound to the Delphix Engine

Protocol	Port Number	Use
TCP	3260	iSCSI target daemon for connections from iSCSI initiators on the target environments to the Delphix Engine

Outbound from a source, staging, or target environment

Protocol	Port Numbers	Use
TCP	80	The Delphix Connector registers environments over HTTP
TCP	xxxx	DSP connections used for monitoring and script management. Typically, DSP runs on port 8415.

Inbound to a source, staging, or target environment

Protocol	Port Numbers	Use
TCP	xxxx	Delphix Connector connections to source environments. Typically, the Delphix Connector runs on port 9100.
TCP	445	If a staging environment is used, the staging server must be able to access SMB (Windows File Sharing) files on the source server.

General outbound from the Delphix Engine port allocation

Protocol	Port Numbers	Use
TCP	25	Connection to a local SMTP server for sending email

Protocol	Port Numbers	Use
TCP/UDP	53	Connections to local DNS servers
UDP	123	Connection to an NTP server
UDP	162	Sending SNMP TRAP messages to an SNMP Manager
TCP	443	HTTPS connections from the Delphix Engine to the Delphix Support upload server
TCP/UDP	636	Secure connections to an LDAP server
TCP	8415	Connections to a Delphix replication target. See Configuring Replication.
TCP	50001	Connections to source and target environments for network performance tests.

General inbound to the Delphix Engine port allocation

Protocol	Port Number	Use
TCP	22	SSH connections to the Delphix Engine
TCP	80	HTTP connections to the Delphix GUI
UDP	161	Messages from an SNMP Manager to the Delphix Engine
TCP	443	HTTPS connections to the Delphix Management Application
TCP	8415	Delphix Session Protocol connections from all DSP-based network services including Replication, SnapSync for Oracle, V2P, and the Delphix Connector.
TCP	50001	Connections from source and target environments for network performance tests via the Delphix CLI.

Firewalls and Intrusion Detection Systems (IDS)

Production databases on source environments (for dSources) are often separated from the non-production environment by firewalls. Firewalls can add milliseconds to the latency between servers. Accordingly, for best performance, there should be no firewalls between the Delphix Engine and the virtual database (VDB) target environments. If the Delphix Engine is separated from a source environment by a firewall, the firewall must be

configured to permit network connections between the Delphix Engine and the source environments for the application protocols (ports) listed above.

Intrusion detection systems (IDSs) should also be made permissive to the Delphix Engine deployment. IDSs should be made aware of the anticipated high volumes of data transfer between dSources and the Delphix Engine.

Adding a windows environment

This topic describes how to add a Windows environment to the Delphix Engine for use with unstructured files.

All Windows source and target environments containing unstructured files must have the Delphix Connector installed to enable communication between the environment and the Delphix Engine. The instructions in this topic cover initiating the Add Target process in the Delphix Management application, running the Delphix Connector installer on the environment, and verifying that the environment has been added to the Delphix Engine.

Prerequisites

- Make sure your source and target environment meet the requirements described in Requirements for Windows Environments.

Procedure

1. From the machine that you want to use, log in to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Next to **Environments**, click the **Actions** menu and select **Add Environment**.
5. In the **Add Environment** wizard Host and Server tab, select:
    a. Host OS: **Windows**
    b. Host Type: **Target**.
    c. Server Type: **Standalone**.
6. Click **Next**.
7. In the Environment Settings tab click the download link for the **Delphix Connector Installer**. The Delphix Connector will download to your local machine.
8. On the Windows machine that you want to want to use as a target, run the Delphix Connector installer. Click **Next** to advance through each of the installation wizard screens.
   The installer will only run on 64-bit Windows systems. 32-bit systems are not supported.
    a. For **Connector Configuration**, make sure there is no firewall in your environment blocking traffic to the port on the target environment that the Delphix Connector service will listen to.
    b. For **Select Installation Folder**, either accept the default folder or click **Browse** to select another.
    c. Click **Next** on the installer's final 'Confirm Installation' dialog to complete the installation process and then **Close** to exit the Delphix Connector Install Program.
9. Return to the Delphix Management application.
10. Enter the **Environment Name**, **Host Address**, **Delphix Connector Port**, **OS Username**, and **OS Password** for the target environment.
11. To provide your own Oracle Java select the **Provide my own JDK** checkbox and click **Next**.
12. In the Java Development Kit tab enter the absolute path to your Oracle JDK and click **Next**.
13. Click **Submit.**

As the new environment is added, you will see two jobs running in the **Delphix Admin Job History**, one to **Create and Discover** an environment, and another to **Create** an environment. When the jobs are complete, you will see the new environment added to the list in the **Environments** panel.

Post-requisites

1. On the Windows environment, in the **Windows Start Menu**, select **Services**.
2. Select **Extended Services**.
3. Make sure that the **Delphix Connector** service has a Status of **Started**.
4. Make sure that the **Startup Type** is **Automatic**.

Options for linking unstructured files on Windows environments

There are two techniques for linking a new dSource from files on a Windows source.

Direct communication with the source environment

The simplest technique is to have the Delphix Engine communicate directly with the source environment. This requires installing the Delphix Connector on the source machine. When linking, specify a local path on the source machine, such as C:\Files\MyData.

Using a staging environment

In some cases, it is not possible or desirable to install the Delphix Connector on the source environment. In those cases, you can install the Delphix Connector on a "staging environment." This is another Windows machine that will act as an intermediary between the Delphix Engine and the source environment. Files on the source must be accessible by the environment user from the staging environment via a UNC path. Specifically, the environment user is only required to have READ access to the path, its directories, and files so that the robocopy utility called from the staging host can function properly. For example, use Windows Sharing on the source machine. When linking, specify the UNC path to the files on the source – for example, \\MySource\MyData\

For more information on installing the Delphix connector, refer to Installing the Delphix Connector Service on the Target Database Servers.

Linking unstructured files

Prerequisites

- The source environment must meet the requirements outlined in Unstructured Files Environment Requirements.
- The Delphix Engine must have access to an environment user. This user should have read permissions on all files to be cloned.

> **Unstructured Files on Cluster Environments**
> Unstructured files cannot be linked from, or provisioned to, any form of a cluster environment, such as an Oracle RAC environment. To link or provision unstructured files from a host that is part of a cluster, add the host as a standalone environment. Then link from, or provision to, this standalone host.

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Environments**.
4. Select the **environment** containing the unstructured files you want to link.
5. Click the **Environment Details** tab.
6. If the environment user described in the Prerequisites section is not listed under **Environment Users**, add the user.
7. Click the **Databases** tab.
8. Scroll to the bottom of the page to view the **Unstructured Files** section.
9. Click the **Plus** icon on the right. This action displays a dialog box prompting for the **Database Name** and **Path**. Enter a name to help identify the files. The path is the absolute path to the directory on the environment server.
10. Click **Add** to save the configuration. After saving this configuration, add the dSource.
11. Click **Manage > Datasets**.
12. On the left-hand side, click the **plus** sign.
13. Select **Add dSource**. Alternatively, on the **Environment Management** screen, you can click **Add dSource** next to a dataset name to start the dSource creation process.
14. In the **Add dSource** wizard, select the source of the files.
15. Select the **Environment User** outlined in the Prerequisites section.
16. Click **Advanced**.
17. Enter **Paths to Exclude**. These paths are relative to the root path of the dataset home path and will not be linked by the Delphix Engine. This feature is most commonly used to exclude directories containing log files. Wildcard (*) pattern matching is supported to exclude all the contents of a directory, without excluding the directory itself. For example, specifying `/dir/` will exclude all contents of `/dir` but still link `/dir` as an empty directory. For PowerShell to escape a $ sign in a directory path please use the following / `before the dollar sign when adding it to the exclude paths, for example: /` $RECYCLE.BIN.
    Info:
    Retroactive Edits to Exclude Paths on Windows
    After creating a dSource, you can edit the set of **Paths to Exclude** from syncing at any time on the dSource's **Configuration** tab. For Unix environments, retroactively adding a path to exclude will result in the next SnapSync deleting the newly-excluded files. However, for Windows environments, retroactively adding a path to exclude will result in the next SnapSync ignoring newly-excluded files. Stale versions of these files will still exist in all future snapshots.

18. If you are linking files from a Unix environment, enter **Paths of Symlinks to Follow**. These paths are relative to the root path of the dataset home path and will be followed to gather additional files to copy.
    Info:
    Paths of Symlinks to Follow - Caveats
    - This feature can only be used to follow symlinks to directories. Symlinks to files will be ignored.
    - This feature is not available for files on Windows environments.
19. Click **Next**.
20. Enter a **dSource Name**.
21. Select a **Database Group** for the dSource.
22. Click **Next**. Adding a dSource to a database group enables you to set Delphix Domain user permissions for that dSource's objects, such as snapshots.
23. Select a **SnapSync** policy.
24. Click **Advanced** to edit retention policies.
25. Click **Next**.
26. Enter any operations that should be run at **Hooks** during the sync process (or any future sync processes).
27. Click **Next**.
28. Review the **dSource Configuration** and **Data Management** information.
29. Click **Submit**.

The Delphix Engine will initiate two jobs to create the dSource, **DB_Link,** and **DB_Sync**. You can monitor these jobs by clicking **Active Jobs** in the top menu bar, or by selecting **System** > **Event Viewer**. When the jobs have been completed successfully, the file's icon will change to a **dSource** icon on the **Environments** > **Databases** screen, and the dSource will be added to the list of **Datasets** under its assigned group.

> **dSource Information**
> After you have created a dSource, you can view information about it and make modifications to its policies and permissions by selecting it in the **Datasets** panel.

Create an empty VDB for unstructured files in the Delphix Engine

This topic describes the procedure for creating an empty VDB, used for unstructured files. The term "unstructured files" in Delphix refers to a dataset that acts as a directory of different files. An empty VDB for unstructured files is not a database and does not receive special treatment or processing by Delphix, it exists as a place for the files to be generated, tracked, and copied. Creating a VDB can be done with provisioning from an existing dataset (a dSource or another VDB) or as an empty VDB created and filled with data.

Creating an empty VDB places an initially-empty mount on target environments. It functions similar to a VDB created via provisioning except it cannot refresh. Refreshing a dataset means overwriting the dataset content with new data pulled from the dataset parent. If a new VDB is created from scratch, the newly-created dataset will not have a parent and cannot be refreshed. All other functions are identical, meaning that the new VDB for unstructured files can be provisioned from, rewound, take snapshots, and so on.

Prerequisites

The target environment must meet the requirements outlined in Unstructured Files Environment Requirements.

> **Unstructured Files on Cluster Environments**
> You cannot create an empty VDB on any form of the cluster environment, such as an Oracle RAC environment. To create an empty VDB on a host that is part of a cluster, add the host as a standalone environment, then create the empty VDB on the standalone host.

Procedure

To create an empty VDB without provisioning:

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select**Datasets**.
4. Click the **plus** icon.
5. Select **Create Empty VDB**.
6. Click **Next** to go to the Target environment tab and do the following:
    a. Under Environment, select a target environment where your empty VDB will be placed.
    b. In the **Mount Path** field, enter the absolute path where the empty VDB/dataset will be mounted.
    c. Click **Next**.
7. On the **Configuration** tab, do the following:
    a. In the **Empty VDB name** field, enter a name for the VDB.
    b. Select a target group for the empty VDB.
    c. Select the **Enable auto-restart of the empty VDB** checkbox to allow VDB to be automatically restarted when the target host is rebooted.
    d. Click **Next**.
8. On the **Policies** tab, select a Snapshot Policy for the empty VDB.
9. (Optional) On the **Hooks** tab, select a hook point and then click + to add a script to run at that point. Click **Next**.
10. The **Summary** tab will enable you to review your configurations. Click **Submit**

After the operation completes, the empty VDB will appear in the Datasets panel.

Provisioning unstructured files as vFiles

Overview

This topic describes the process of provisioning to a set of unstructured files as vFiles.

Prerequisites

- You will need an unstructured files dSource, as described in Linking Unstructured Files, or an existing vFiles from which you want to provision another.
- The target environment must meet the requirements outlined in Unstructured Files Environment Requirements.

> **Unstructured Files on Cluster Environments**
> Unstructured files cannot be linked from, or provisioned to, any form of a cluster environment, such as an Oracle RAC environment. To link or provision unstructured files from a host that is part of a cluster, add the host as a standalone environment. Then, link from or provision to this standalone host.

Post provision/migration ownership rules

When a new vFile VDB is provisioned, the ownership is changed to match the Environment User anytime a VDB is enabled. This could cause conflict in the ownership of existing files in a case where the VDB has just been migrated to a new host – the VDB files on the new host will now be owned by the new Environment User.

Procedure

1. Login to the **Delphix Management** application.
2. Click **Manage**.
3. Select **Datasets**.
4. Select a **dSource** or **vFiles**.
5. Click the **TimeFlow** tab.
6. Select a **snapshot**.
7. Click **Provision**. The **Provision vFiles** panel will open, and the field **Mount Path** will auto-populate with the path to the files on the source environment.
8. Select a target environment. If you need to add a new target environment for the vFiles, click the **Plus** icon next to **Filter Target**to add an environment.
   Note: You can only target a Unix environment when provisioning from a Unix dSources or vFiles. You can only target a Windows environment when provisioning from a Windows dSources and vFiles.
9. If necessary, modify the **Mount Path**.
   - On Windows, this mount path must not be a UNC path. It must be a local drive letter and folder path. The UNC path will operate after the provisioning completes.
   - On Linux and Unix hosts, this mount path must be the full path and not include symlinks.
10. Click **Advanced**.
11. Enter **Additional Mount Points**. When it is mounted to the target environment, the vFiles will be mounted to any additional mount points you provide.
    Note: The **Shared Path** is a relative path dictating which portion of the vFiles should be available on the additional environments. To share the entirety of the vFiles, specify a **Shared Path** of /.
12. Select an Environment User to own the mounted files. If the files are being mounted to multiple environments, ensure this user is available across all environments.
13. Click **Next**.

14. Enter a vFiles **Name**.
15. Select a **Target Group** for the vFiles. If necessary, click the **Plus** icon to add a new group.
16. Select a **Snapshot Policy** for the vFiles. if necessary create a new policy.
17. Click **Next**.
18. Enable Auto VDB Restart to allow the VDB to be automatically restarted when the target host reboot is detected by Delphix.
19. Enter any operations that should be run as Hooks during the lifetime of the vFiles.
20. Click **Next**.
21. Click **Submit**.

When provision starts, the vFiles will appear in the Datasets panel. Select the vFile and navigate to the Status tab to see the progress of the job. When provisioning is complete, you can see more information on the Configuration tab.

Managing vFiles

Overview

This article is used to cover steps on adding an additional mount to an existing vFile and outlining post provision/ migration ownership rules.

Adding an additional mount

1. Login to the **Delphix Management** application.
2. Select the **vFile** you want to edit.
3. From the Actions menu (**...**) select **Disable**.
4. In the **Configuration** tab select the **Source** sub-tab, a Pencil icon will appear on the right of the Additional Mount Points.
5. Click on the **Pencil** to edit the mount points.
6. In the **Additional Mount Points** window select the **Plus** icon to add additional mount points.
7. Click the **checkmark** to save.
8. From the Actions menu (**...**) **Enable** the vFile.

Post provision/migration ownership rules

When a new vFile VDB is provisioned, the ownership is changed to match the Environment User anytime a VDB is enabled. This could cause conflict in the ownership of existing files in a case where the VDB has just been migrated to a new host – the VDB files on the new host will now be owned by the new Environment User.

vFiles best practices and common pitfalls

Overview

This document is the implementation guide for the best practices of implementing data source integrations using the Delphix vFile functionality provided through the AppData toolkit. Since "unstructured files", or vFiles, typically is implemented for non-DBMS file types, this implementation will require a certain degree of configuration and scripting in order to function. This means that while you do get some of the main functionality of the Delphix Engine, there are also quite a few pitfalls that may deter customers from wanting this type of configuration. Using vFiles and scripts to implement data source ingestion is a workaround that should only be used if you and the customer are fully aware of the best practices and limitations as called out below.

Best practices and implementation

In order to be successful with this workaround, please keep in mind these best practices which will enable you to be successful in implementing a vFile configuration for data sources.

- Write a script to put the RDBMS into 'Backup' mode to prepare it for the next set of scripts Delphix will perform
- Create tablespaces in the RDBMS which dxtoolkit will take snapshots of while the database is in backup mode
- After any vFile provisioning job, script a recovery of the RDBMS instance

Common use cases

These use cases are the typical scenarios for which you would use this workaround.

Unsupported DBMS configuration

This functionality is most appropriate when a customer is using an unsupported data source and wants a quick MVP to demonstrate the value of Delphix. It is critical to understand the pitfalls of this implementation and what value propositions that this excludes so that the customer is aware of what capabilities our core integrations provide.

Common pitfalls

These pitfalls of functionality should be considered whenever implementing vFile scripted ingestion. Refer to the table for a quick reference of pitfalls.

Pitfall	**Description**
LogSync	Not Available
Point in Time Recovery	Not Available
Clustering	Not Available
User Interface	Not Available

Pitfall	Description
Production Downtime	Yes
Use of dxtoolkit	Requires Training

LogSync and Point in Time provisioning

Since these scripts are taking full backups of the database, there will not be logs available for us to sync with. Therefore, point in time provisioning is not available to any vFile implementation.

Manual operations

Currently, customers using this workaround have to perform the scripts manually. This creates operational overhead for customers, particularly their DBA team.

Clustering

Using vFiles is not available for clustered instances, and there is currently no workaround for this type of configuration.

Lack of Graphic User Interface (GUI)

Scripts perform all the core functionality that is typically carried out in the Delphix GUI. This implementation precludes the usage of the main GUI and is only accessible via the Command Line Interface (CLI).

Production downtime

As the database enters backup mode during the ingestion process, this workaround causes the production source to go down while Delphix ingests their data.

Understanding and implementing scripts and dxtoolkit

Implementing these scripts requires an understanding of dxtoolkit and the data source which you are trying to integrate with. Typically, we estimate the learning period of this to be about 1 - 2 months to fully understand how to use these tools and how to best implement the scripts for any given source.

Conclusion

Using vFile functionality with scripts is a quick way to demonstrate the value of the Delphix Engine easily. With a little scripting and Delphix knowledge, you can engineer ingestion with dxtoolkit to take snapshots of the entire database. However, there are several pitfalls that every customer should know about and understand before moving forward with this type of implementation. If you have any questions, please reach out to Product Management.

Delphix Engine plugin management

Plugin management

This section provides information on the data management plugin deployed on the Delphix Engine. Users can directly upload their Plugins to the Delphix Management application with a single click, thus eliminating the process of uploading the Plugin using the command. Plugin upload via the Delphix Management application helps users to build their Plugin library in a simple and easy-to-use manner.

The Plugin screen allows users to view a structured or formatted view of their selected Plugin and it provides a single click option for users to delete a Plugin.

> Only Admin users are able to upload or delete Plugins. Standard users are only able to view Plugin information.

Plugin types

There are two types of plugins the Delphix engine supports Lua/Python plugins and platform plugins. Because support for these types of plugins was written at different times, the upgrade and replication workflows will differ depending on what the plugin type is. To figure out what type of plugin this is, follow the instructions below:

1. Login to the **Delphix Management** application.
2. Select **Manage > Plugins**.
3. Select the plugin you want to check and look for the type field in the details section on the right.
4. If the type is Toolkit this means the plugin is a Lua plugin. If the type is Plugin, then the plugin is either a platform plugin or a python plugin.

Installation of plugin

Plugins can be uploaded onto the Delphix Engine only by an admin.

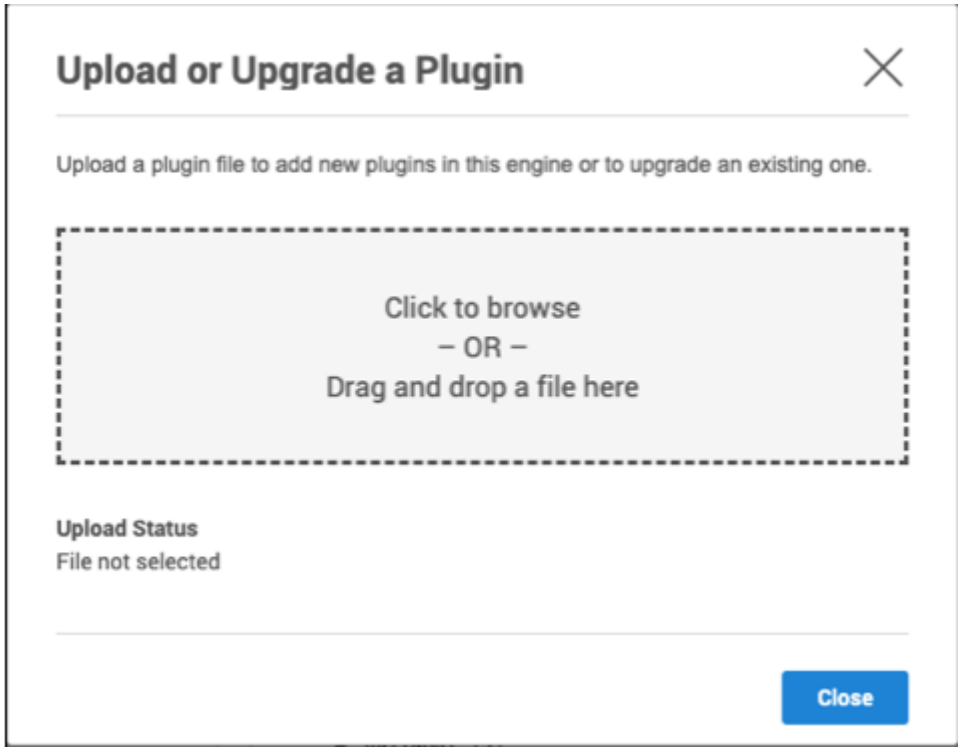
Prerequisites

- Make sure you have access to a copy of the plugin from the computer you will be installing.
- Verify with the plugin author that the version of the plugin you have is compatible with your Delphix Engine version.

Procedure

To add or upload a new Plugin complete the instructions below:

1. Login to the **Delphix Management** application.
2. Select **Manage > Plugins**.
3. To add or upload a Plugin, click the icon.
   This opens the Upload or Upgrade Plugin dialog.



4. To start using the plugin, refresh target and source environments to discover any repositories.

Upgrading a plugin

When a new version of a plugin is released, the new plugin can only be uploaded onto the Delphix Engine by an Admin.

Prerequisites

- Make sure you have access to a copy of the plugin from the computer you will be installing.
- Verify with the plugin author that the new version of the plugin is compatible with your current version, and with your Delphix Engine version.
- If the already-installed plugin does not use the Lua language, then you don't have to disable any dSources or VDBs. However, be aware that you will not be able to perform any Delphix operations on them while the plugin is being upgraded.

Procedure

To upgrade a Plugin just follow the instructions above on how to upload a Plugin. The newly-uploaded plugin will be used to upgrade any already-present version(s) of the same plugin.

Notes on replication with objects administered by a plugin

Replication works the same way for objects created with plugins and objects created using natively supported data types. But, complications and problems can arise when a replicated plugin's version does not match the version of the installed plugin.

Replication Recommendation

> As discussed in the section on Delphix Replication Overview, we highly recommend that replication target engines be used solely to hold replicated objects, and that failover is only done in disaster recovery situations. Or, at the least, make sure that the versions of replicated plugins match the versions of plugins that are already installed on the replication target engine.

There are specifically two operations done after replication that will be listed below.

Provisioning from replicated data sources or VDBs

Replica provisioning is only guaranteed to work when the source object's plugin is already also installed on the target engine with the identical version.

Installed Plugin Version	**Result**
None	Replica provisioning will fail.
Lower than replicated version	Replica provisioning will fail.
Same as replicated version	Replica provisioning will succeed.
Higher than replicated version	If the replicated plugin uses the Lua language, replica provisioning will fail. If the replicated plugin does not use the Lua language, replica provisioning may or may not succeed, depending on whether the installed plugin is compatible with the replicated version.

More general information on how to perform a replica provision can be found here.

Failing over a replica

As mentioned above, we strongly recommend that replication target engines do not have any non-replica objects. Although the replica failover process will always work, the failed-over objects may or may not be fully functional if there are non-replica plugin-administered objects already on the same engine. In some cases, you may have to delete some objects, and their associated plugins.

Installed Plugin Version	**Result**
None	The replicated plugin and its objects will be moved into the live namespace. All failed-over objects will be fully functional.

Installed Plugin Version	Result
Lower than replicated version	The failed-over objects may become "inactive", and any Delphix operations run on them will fail. In this case, you can try to upgrade the existing plugin so that the version matches the replicated inactive plugin. This may or may not work, depending on whether there are any conflicts between any objects.
Same as replicated version	The failed-over objects will be fully functional.
Higher than replicated version	The failed-over objects will become "inactive", and any Delphix operations run on them will fail. In this case, you can try to upgrade the inactive plugin (see instructions below). Again, this may or may not work, depending on whether there are conflicts between any objects, and on whether the installed plugin knows how to interact with the inactive plugin.

More generic information on how to perform an actual failover can be found here.

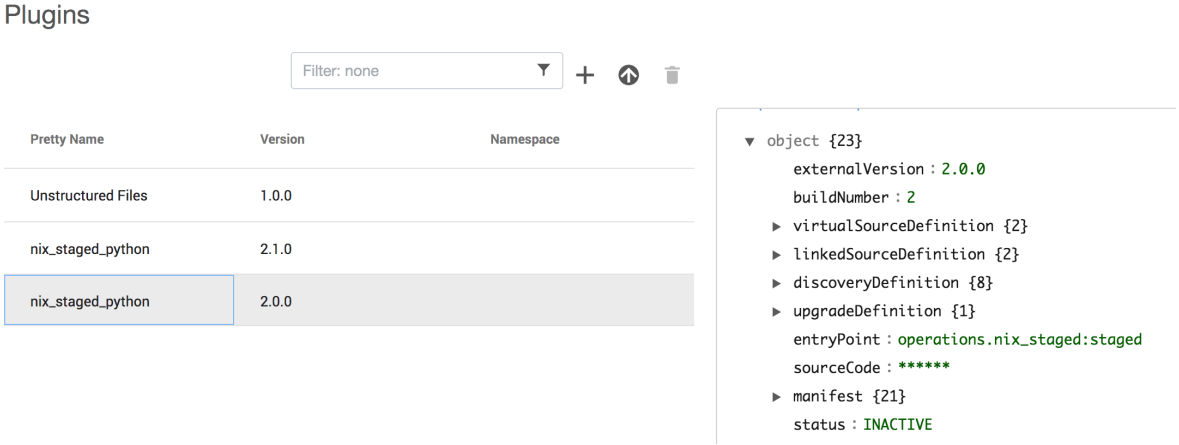
Upgrading an inactive plugin

If a failed-over plugin becomes "inactive" (see details above), you may be able to activate the plugin by upgrading it to match the version of the installed plugin.

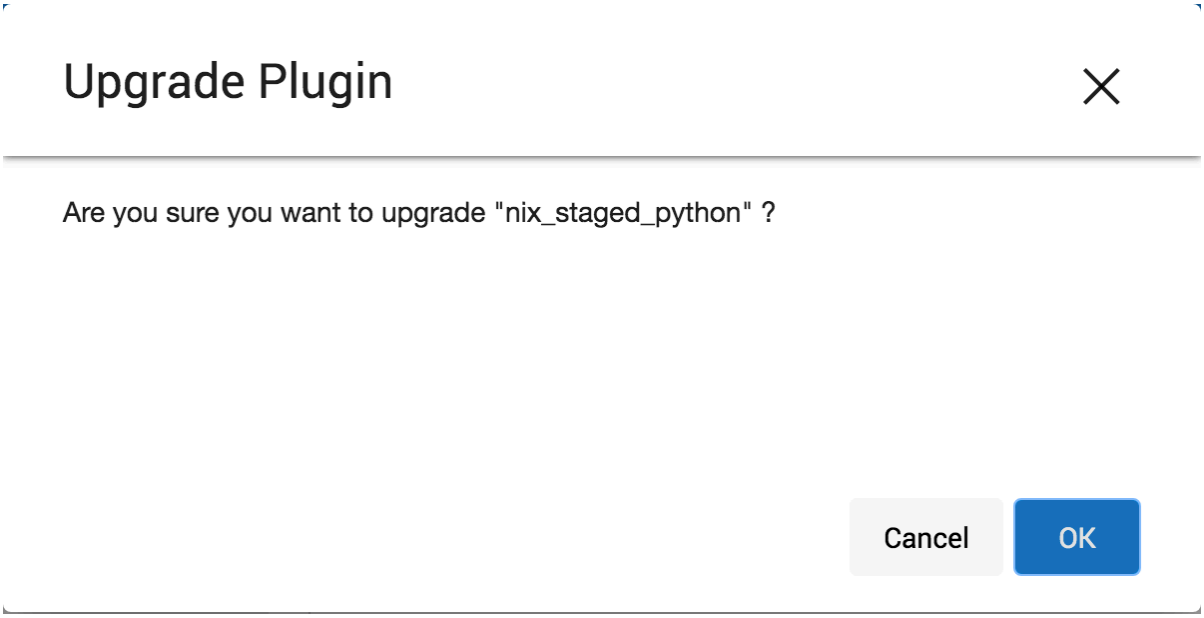
Procedure

To upgrade an inactive Plugin complete the instructions below:

1. Login to the **Delphix Management** application.
2. Select **Manage > Plugins**.
3. In the drop-down menu on the left, select the plugin that is inactive.



4. Click the upgrade button  which should be clickable now to activate the inactive plugins. Make sure to confirm the upgrade by selecting OK.



5. After the upgrade is successful the inactive plugin of the lower version will not exist in the plugin list.

> If the inactive plugin is a higher version, the active plugin needs to be upgraded using the normal upgrade procedure instead.

Unstructured files hook operation notes

Shell Operations

RunCommand operation

The RunCommand operation runs a shell command on a Unix environment using whatever binary is available at `/bin/sh`. The environment user runs this shell command from their home directory. The Delphix Engine captures and logs all output from this command. If the script fails, the output is displayed in the Delphix Management application and command line interface (CLI) to aid in debugging.

If successful, the shell command must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Examples of RunCommand operations

You can input the full command contents into the RunCommand operation.

```
remove_dir="$DIRECTORY_TO_REMOVE_ENVIRONMENT_VARIABLE"

if test -d "$remove_dir"; then
    rm -rf "$remove_dir" || exit 1
fi

exit 0
```

If a script already exists on the remote environment and is executable by the environment user, the RunCommand operation can execute this script directly.

```
/opt/app/oracle/product/10.2.0.5/db_1/dbs/myscript.sh "$ARG_ENVIRONMENT_VARIABLE"
"second argument in double quotes" 'third argument in single quotes'
```

RunBash operation

The RunBash operation runs a Bash command on a Unix environment using a `bash` binary provided by the Delphix Engine.The environment user runs this Bash command from their home directory. The Delphix Engine captures and logs all output from this command. If the script fails, the output is displayed in the Delphix Management application and command line interface (CLI) to aid in debugging.

If successful, the Bash command must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Example of RunBash operations

You can input the full command contents into the RunBash operation.

```
remove_dir="$DIRECTORY_TO_REMOVE_ENVIRONMENT_VARIABLE"
```

```
# Bashisms are safe here!
if [[ -d "$remove_dir" ]]; then
    rm -rf "$remove_dir" || exit 1
fi

exit 0
```

Shell operation tips

Using `nohup`

You can use the `nohup` command and process backgrounding from resource in order to "detach" a process from the Delphix Engine. However, if you use `nohup` and process backgrounding, you MUST redirect `stdout` and `stderr.`

Unless you explicitly tell the shell to redirect `stdout` and `stderr` in your command or script, the Delphix Engine will keep its connection to the remote environment open while the process is writing to either `stdout` or `stderr` . Redirection ensures that the Delphix Engine will see no more output and thus not block waiting for the process to finish.

For example, imagine having your `RunCommand` operation background a long-running Python process. Below are the bad and good ways to do this.

Bad Examples

- `nohup python file.py & # no redirection`
- `nohup python file.py 2>&1 & # stdout is not redirected`
- `nohup python file.py 1>/dev/null & # stderr is not redirected`
- `nohup python file.py 2>/dev/null & # stdout is not redirected`

Good Examples

- `nohup python file.py 1>/dev/null 2>&1 & # both stdout and stderr redirected, Delphix Engine will not block`

Other operations

RunExpect operation

The RunExpect operation executes an Expect script on a Unix environment. The Expect utility provides a scripting language that makes it easy to automate interactions with programs which normally can only be used interactively, such as `ssh` . The Delphix Engine includes a platform-independent implementation of a subset of the full Expect functionality.

The script is run on the remote environment as the environment user from their home directory. The Delphix Engine captures and logs all output of the script. If the operation fails, the output is displayed in the Delphix Management application and CLI to aid in debugging.

If successful, the script must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Example of a RunExpect Operation

Start an `ssh` session while interactively providing the user's password.

```
spawn ssh user@delphix.com
expect {
    -re {Password: } {
        send "${env(PASSWORD_ENVIRONMENT_VARIABLE)}\n"
    }
    timeout {
        puts "Timed out waiting for password prompt."
        exit 1
    }
}
exit 0
```

RunPowershell operation

The RunPowershell operation executes a PowerShell script on a Windows environment. The environment user runs this shell command from their home directory. The Delphix Engine captures and logs all output of the script. If the script fails, the output is displayed in the Delphix Management application and command line interface (CLI) to aid in debugging.

If successful, the script must exit with an exit code of `0`. All other exit codes will be treated as an operation failure.

Example of a RunPowershell operation

You can input the full command contents into the RunPowershell operation.

```
$removedir = $Env:DIRECTORY_TO_REMOVE

if ((Test-Path $removedir) -And (Get-Item $removedir) -is [System.IO.DirectoryInfo])
{
    Remove-Item -Recurse -Force $removedir
} else {
    exit 1
}
exit 0
```

Unstructured files environment variables

Operations that run user-provided scripts have access to environment variables. For operations associated with specific dSources or virtual databases (VDBs), the Delphix Engine will always set environment variables so that the user-provided operations can use them to access the dSource or VDB.

dSource environment variables

Environment Variable	Description
`DLPX_DATA_DIRECTORY`	Path where linked-staged database is mounted

VDB environment variables

Environment Variable	Description
`DLPX_DATA_DIRECTORY`	Path where virtual database is mounted

Best practices

For optimal usage of Delphix, there are many different configurations and best practices you'll need to note to tune your engines for peak performance. This ranges from settings on the hypervisor that runs Delphix to the setting on your sources, virtual databases, and the operating systems on which they run. Find out more information based on each category below.

These pages are the starting point to begin your architecture planning for the best possible Delphix deployment. To function as a high-performance virtual appliance, we need to ensure the underlying infrastructure components are deployed in an ideal manner consistent with our best practices - these pages are to help you achieve that goal.

Introduction to Delphix architecture

Database virtualization with Delphix

Hypervisor and host

Best practices for hypervisor host and VM guest

Network

Best practices for network configuration

CLI Cookbook: network performance

Optimal network architecture for the Delphix engine

Network operations using the Delphix session protocol

Network performance expectations and troubleshooting

Storage

Best practices for storage

Optimal storage configuration parameters for the Delphix engine

Storage performance expectations and troubleshooting

Data protection

Best practices for Delphix engine data protection

Source DB and OS settings

Best practices for source DB and OS settings

Target DB and OS settings

Best practices for target DB and OS settings

Staging target

Best practice for staging targets

Architecture checklist (consolidated best practices)

Architecture Checklist

Architecture checklist FAQ

Architecture checklist FAQ

Architecture checklist FAQ

Architecture best practices overview FAQ

These questions relate to the standard processes our Solutions Architecture team executes when engaged for Architecture review of a customer's environment.

Why go through an architecture or sizing process?

To measure - when possible - the workload in your environment and calculate the number of engines needed

- To ensure best practices are applied so your engine(s) perform to the level expected
- To find and avoid or work around potential challenges in an environment
- To maximize your license value

Why does Delphix collect IO data?

The collection of IO data allows us to more accurately size the Delphix Engine rather than making recommendations based on assumptions or synthetic data. The data you provide for analyses is from the SOURCE and TARGET DB servers.

Why is the customer database and inventory (CDI) important?

The database information is typically the fundamental building block upon which we build our understanding of your future Delphix environment. We ask questions related to the database names, locations, platforms, and versions, as well as estimates of size, throughput, and changes.

The answers you provide to our infrastructure questions give us insight into potential challenges that may arise as we seek to integrate our application and best practices into your environment. Processes and software you use normally may not integrate optimally into our engine, and it’s helpful to know this as soon as possible.

Why does Delphix use a virtual machine? Shouldn’t all high-performance applications use physical infrastructure?

Using virtualization not only allows our customers to use their choice of commodity hardware, but it also allows us to focus on core features rather than hardware support.

Architecture best Practices for hypervisor host and VM guest - ESX FAQ

Why does Delphix recommend 8 vCPUs and 128GB of memory per 8 vCPUs?

8 vCPUs are not only a standard licensing block, but they are also key to meeting our 10Gbps single-engine throughput potential and help to sustain low latency for VDBs.

As with CPU, cache memory is required to drive peak loads on the Delphix Engine. More memory allows for more blocks to be read from the cache rather than going to less performant disks. Delphix stores cached data in a compressed format and only keeps a single copy of unique blocks in memory. These features give read performance across multiple VDBs provisioned from a single source dramatic improvements in speed, scalability, and memory utilization.

Why does Delphix request reservations for CPU and memory?

Delphix performance can be greatly impacted there is contention over CPU or RAM. Reservations allow the engine to explicitly control those resources and avoids the possibility of contention with other VMs, even when resources are overcommitted.

Why does Delphix request Hyper-Threading (HT) be disabled?

Hyper-Threading can have a positive impact on many, but not all applications. Delphix has a different execution profile that does not benefit from Hyper-Threading. Factors such as significant memory bandwidth requirements and a high level of parallelism which requires a high number of shared locks mean that HT and Delphix do not generally work well together.

Why does Delphix request 4 controllers, and why must the storage be identical between them?

To provide optimal storage performance, you must spread data equally over the maximum (4) virtual SCSI controllers. To provide consistent performance between each of the four controllers, you need to ensure storage is identical between them.

Why must virtual disks (VMDKs) be thick provisioned and eager-zeroed?

Thick provisioning and eager zero ensure performance is top-notch from the start with no hiccups from expanding virtual resources.

Why is 20% free space required?

While the ZFS file system has a lot of features leveraged by Delphix, it loses efficiency as space decreases. 20% is the minimum that must be available for best performance.

Why does Delphix request you reserve CPU and RAM for Hypervisor overhead?

Based on VMWare's resource management guide and our own experiences with high IO throughput. Note there is no specific mechanism to assign resources to the hypervisor, the only way to preserve overhead is by not allocating resources to guests.

Why does Delphix generally want VMWare HA enabled, but DRS disabled?

VMware HA (High Availability) addresses outages that occur when a physical host goes down or is completely offline, by migrating the guest(s) to another physical host and restarting them. There is no real downside, it simply brings unavailable servers back online.

VMware DRS (Distributed Resource Scheduler) is for load balancing host resources in a cluster. Because of high IO and best practices configuration for optimal performance, our engine is typically not a good candidate for relocation.

Why does Delphix request you set power management to High-Performance Mode?

This will ensure power management will never impact performance by entering into a lower power state (also known as c-state).

Architecture best practices for storage FAQ

Why does Delphix require 127GB of storage for the OS?

The system partition requires space to store and operate the OS, as well as application logs, upgrade and rollback images, and enough free space to store a kernel or application core dump should it be required.

Why does Delphix require our LUNS to be uniform and contain an equal quantity and capacity of VMDKs, yet thin provisioning is OK?

Because our engine leverages parallel reads, we need the storage capacity and quantity of disks they hold to be consistent. This allows the reads and writes to be evenly distributed, and minimizes the impact of potential utilization imbalances which would create a “long tail” of higher latency on a single controller, impacting the entire engine.

Data storage LUNS are generally formatted with a VMFS file system and have placed upon them a virtual disk (VMDK) which is thick provisioned and eager zeroed, so it would be a waste of time to thick provision the LUN also.

Why does Delphix require < 10ms latency (95th percentile) storage?

Storage latency is especially important in database environments. Average latency doesn’t give a complete picture of responsiveness, especially because Delphix leverages parallel reads; so inconsistent performance (e.g. good average latency but a “long tail”) can impact multiple operations. This is why Delphix has a focus on 95th percentile latency, and why we validate storage performance as the first step when a new engine is deployed.

For more information, you may find the following article helpful: The Cost of Latency.

Why does Delphix prefer to extend existing storage rather than simply add more while maintaining equal distribution?

While it is possible to add more storage and maintain the practice that "storage should be equal across controllers" – extending LUNS (then virtual disks) ensures that:

- We do not continue to fill disks which may be full
- Existing disks do not suffer a write performance penalty from low capacity
- Storage performance is consistent

Architecture best practices for network FAQ

Why does Delphix request 10GE Ethernet?

As a matter of physics and standards - 10 gigabit (Gb) Ethernet can sustain approximately 1 gigabyte (GB) per second of throughput. With all our best practices applied, a Delphix Engine can achieve very close to that line speed, allowing for optimal load, engine, and license utilization. Lower network speeds may be acceptable for low loads, while in some environments NIC teaming (e.g. LACP) may be required for top speeds.

Why does Delphix require < 1ms latency to TARGET servers and < 50ms to SOURCE servers?

Delphix leverages NFS and iSCSI (depending on platform) for live TARGET DB mounting over the network, so it’s imperative that latency is as low as possible. Data coming from SOURCE servers is not generally as time-sensitive, so you need a minimum latency of < 50ms to ensure operational integrity.

Why does Delphix request Jumbo Frames?

Jumbo frames increase the Ethernet maximum transmission unit (MTU) from the default 1500 bytes to 9000 bytes. This has several effects such as decreasing CPU cycles by transferring fewer packets and increasing the engine throughput. You will find jumbo frames have a 10-20% real-world impact and are required (along with all other best practices) to handle peak loads of 800 - 1000MB/s on an 8 vCPU engine with a 10Gb network.

How does Delphix avoid communication impact with non-jumbo frame hosts when Jumbo Frames are enabled on the Delphix Host?

Path MTU Discovery is the mechanism by which two hosts agree on the MTU leveraged for communication between them. This mechanism will ensure communication between both standard and Jumbo Frame enabled hosts works as expected.

When does Delphix recommend NIC teaming?

The Delphix Engine is capable of high throughput, but not every enterprise has sufficient network bandwidth to support it. Teaming is a less expensive way of increasing the bandwidth when compared to new hardware.

Why does Delphix recommend logical and physical and co-location?

The Delphix Engine leverages network connections extensively, so optimizing the latency whenever possible is very important - sometimes critical.

Architecture best practices for Delphix engine data protection FAQ

Why does Delphix recommend SAN snapshots or Delphix replication for backup?

There are a few possible methods for data protection of the Delphix Engine. Those methods are SAN snapshots, Delphix replication, and virtual machine snapshots (for very small engines only). Because the Delphix Engine is itself a backup of source environments, many customers simply plan to rebuild in the event of a disaster.

What is the DXToolkit, and how can it help?

The professional services team has created a Perl-based “DXToolkit” which can help export and import certain configuration data over web services. This toolkit can be leveraged to assist with would normally be a manual re-install outside of the above methods.

For further detail around data protection, please speak with your Delphix contact.

Can I use a VMware-based backup solution such as VEEAM to backup my Delphix Engine?

Yes, VMWare backup solutions are useful for backing up guest VMs. However, Delphix suggests that you only use this approach for Delphix Engines which have a smaller storage footprint (perhaps < 2 TB) and are less active.

Running this type of backup puts a load on the environment, which might adversely impact Delphix VM performance.

Can I use a VMware snapshot for backing up Delphix for a small window – for example, during an engine upgrade?

Yes. However, even though snapshots are instantaneous, they track changes separately from the base disks and can grow to consume as much space as the original.

Upgrades, in particular, can change substantial amounts of data.

If you lose physical disks, snapshots are useless because it needs them to make up the current state of VM.

A Delphix Engine is often allocated multiple terabytes of storage and is often very busy due to load aggregation from virtual databases on multiple target servers, so this approach may be challenging.

Snapshots cannot detect storage corruption.

Can I use a Storage snapshot solution to protect Delphix against Storage and Delphix corruption?

Yes. However, please note that the caveats which apply to VMWare snapshots will also apply here.

A specific concern related to storage layer snapshots is that you must create a consistency group that contains both the OS and Data disks.

Can I use RMAN to backup my VDBs just like a Physical database to provide extra protection?

You can backup Delphix VDBs using Oracle RMAN tools, but the recovery database would first require re-hydration of that VDB, which might take up equivalent production storage space.

Furthermore, that re-hydrated database needs to be brought into the Delphix framework as a dSource, after which you can provision a VDB to complete recovery. The whole process might take hours or days to recover.

The best approach is to use the VDB Snapshot capability to backup VDB frequently and then leverage Delphix Replication capability to protect underlying Delphix storage, which holds that VDB snapshot.

Architecture checklist

Overview

The Architecture Checklist is an overview combining the more important Architectural best practices into a single list. For more detail regarding some of the reasons for our best practices, review the Architecture Checklist FAQ.

Architecture best practices for hypervisor host and VM guest - ESX

Hypervisor

1. ESXi 6.x or 7.0 is recommended. ESXi 5.5 or earlier is no longer supported.
2. HyperThreading (HT) for Intel®-based servers (no HT on AMD CPUs).
    a. Disable HT in BIOS, on the ESXi Host, **and** disable HT Sharing on the Delphix VM for consistency. This is our best practice, disable at all levels. Any other combination may result in non-deterministic performance.
    b. When HT cannot be turned off for both the Host and Delphix VM, it should be turned on at all levels, not run in a "mixed-mode".
        i. HT disablement at a guest level only can result in non-deterministic performance.
        ii. A dedicated ESXi host, cluster or DRA is recommended where consistent VDB performance is paramount.
    c. VM migration to a new host (e.g. VMware HA or vMotion) can create mismatched HT settings.
3. ESXi overhead (resources required for hypervisor cannot be reserved, they must be left unallocated).
    a. Memory Overhead - 10% of available RAM must not be allocated to guest VMs.
        - *Example: 256GB RAM, allocate 230GB to Delphix VM, leave 26GB for ESX*
    b. CPU Overhead - At least 2 cores (ideally 4) must not be allocated to guest VMs.
        - *Example: If 16 physical cores are available, allocate 12 to the virtual machines, leaving 4 for the hypervisor*
        - Why 4? Certain hypervisor functions require precedence over any virtualized system. If a hypervisor needs more CPU than the amount currently available, it can de-schedule all other virtual processes to ensure adequate CPU resources for the hypervisor. Ensuring the hypervisor will not have to de-schedule any running virtual processes (worlds) by setting aside and not over-subscribing CPUs for virtual functions will leave them available for hypervisor use.
    c. Even if the Delphix VM is the only VM on a host, the hypervisor is still active and essential; and still needs resources.
4. BIOS Power Management should be set to High Performance where ESXi controls power management.
    a. Can be impacted by VMware KB 1018206 - poor VM application performance caused by power management settings.
    b. Ensure that all BIOS managed C-States other than C0 are disabled if power management is hardware controlled.
    c. Ensure that all ACPI sleep states above S0 are disabled in the BIOS.
    d. Examples for popular server lines from Cisco, HP, Dell below. *Specific models will vary, use the appropriate spec sheet.*
        i. UCS: disable the Processor Power States, disable Power Technology, set Energy Performance to "Performance"
        ii. HP Proliant: set HP Power Regulator to HP "Static High Performance" mode
        iii. Dell: set BIOS System Profile to "Performance Optimized" mode
5. VMware HA can be enabled; VMware DRS is generally disabled.
6. Blade/Rack Server Firmware and ESXi Drivers should be updated to latest versions.

7. For Intel®-based servers with E5-2600 v2 processors.
8. Two typical server configurations:
    a. Blade Farm
    b. Rack Server Hyper-Converged configurations are possible for high performance.

Virtual Machine Guest

1. For VM machine settings, see Virtual Machine Requirements for VMware Platform.
2. VMWare Guest Specifications:
    a. Minimum: 8 vCPU x 64 GB Small: 8 vCPU x 128GB Medium: 16 vCPU x 256 GB Large: 24 vCPU x 512 GB
    b. Reserve 100% of RAM and CPU:
        - If the ESX host is dedicated to Delphix, CPU and RAM reservations are advised but not necessary, however, swap space will be required on the hypervisor to compensate for the lack of reserved RAM.
    c. Hyperthreading - See the ESX host section at the top. Disable HT Sharing on VM, disable HT on ESX Host.
3. Assign single-core sockets for vCPUs in all cases. If there is a compelling reason to use multi-core CPUs, reference the following article from VMware which describes matching virtual multi-core sockets to the hardware ESX is running on. VMware Article on CoresPerSocket Example: *ESXi Host has 2 socket x 18 core Intel Xeon, Delphix Engine wants 16 vCPU.* Configure Delphix VM with 2 Virtual Sockets, 8 Cores Per Socket to utilize hardware architecture.
4. Avoid placing other extremely active VMs on the same ESX host.
5. Monitoring - vSphere Threshold Alerts for CPU, Network, Memory, Capacity.
6. To set the number of vCPUs per virtual machine via the vSphere client, please see "*Virtual CPU Configuration"*in the Administration guide:
   ESXi 6.0
    a. Delphix VM CPU Utilization - Delphix KB article on what makes Delphix VMs similar to other resource-intensive applications
    b. Exchange on VMware Best Practices - VMworld 2013 session
    c. ESXTOP Reference, Blog
7. Ensure that the latest available VMware drivers and firmware versions are installed for HBAs, NICs and any other hardware components configured on the Delphix virtual machine. This is a critical step that can have a massive impact on the performance and robustness of our solution.

Architecture best practices for network configuration

1. For more information about network configuration refer to Network Performance Configuration Options.
2. Delphix Engine <===> Target Host:*Implement standard requirements for optimal NFS/ISCSI performance:*
    a. Optimal physical network topology:
        - Low latency: < 1ms for 8K packets.
        - Network adjacency: minimize network hops, co-locate in the same blade enclosure, co-locate on the same physical host.
        - Eliminate all Layer 3+ devices - firewalls, IDS, packet filters (Deep Packet Inspection - DPI).
        - Multiple switches can add latency and fragmentation and reordering issues will add significant latency.
    b. Optimal throughput:
        - 10GbE physical uplinks or higher.
        - Jumbo frames (typically MTU 9000) improves network efficiency for 8K packets: lower CPU, lower latency, greater throughput.
        - All devices end-to-end *must* be configured for the larger frame size including switches, routers, fabric interconnects, hypervisors and servers.
        - Traffic between two VMs on the same host is limited to ~16Gbps when leveraging built-in virtual networking.

c. Optimal logical flow:
   - Disable QoS throttles limiting network utilization below line rate (e.g. HP Virtual Connect FlexFabric).
   - Consider a dedicated VLAN (with jumbo frames) for NFS/iSCSI traffic.
     - VMware KB-Configuring iSCSI port binding with multiple NICs in one vSwitch for VMware ESXi 6.0.x (2045040)
     - VMware KB-Considerations for using software iSCSI port binding in ESX/ESXi (2038869)

d. NIC Teaming (at ESX layer) of multiple physical uplinks can provide additional throughput for higher workloads.
   i. Examples: 4 x 1Gb NICs support up to 400 MBPS IO, 2 x 10Gb NICs support up to 2 GBPS IO.
   ii. VMware KB-1004088 has NIC teaming recommendations, including `route-based-on-IP-hash` policy.
   iii. VMware KB-1001938 has host requirements for physical link aggregation (LACP, EtherChannel).
   iv. VMware KB-1007371, popular blog post details problems with NIC selection using `dest-IP` hash.

e. Fragmentation and dropped packets can result in excessive retransmissions of data, reducing throughput.
   i. On AIX, LSO (Large Send Offload) and LRO (Large Receive Offload) network features have caused many problems. The virtualization engine employs LSO only, but no LRO, AIX employs both. These features are enabled by default. The best practice is to disable LSO on the VE when the target is AIX.
      - LSO enabled on the Virtualization engine can result in dramatically limited or blocked transmit throughput (and increase re-transmission traffic).
      - When wishing to disable all, LSO should be disabled on the Delphix Virtualization Engine, while both LRO and LSO should be disabled on guest VM of AIX Target. The specific commands to disable LSO and LRO on AIX are not captured here and appear to be context/version-specific.

f. **Jumbo frames check via ping**

```
Delphix Engine
$ ping -D -s [Target_IP] 8000
"ICMP Fragmentation needed and DF set from gateway" indicates MTU < 8028

Linux
$ ping -M do -s 8000 [Delphix_Engine_IP]
"Frag needed and DF set (mtu = xxxx)" indicates MTU < 8028

MacOS
ping -D -s 8000 [Delphix_Engine_IP]

Note: "sudo sysctl -w net.inet.raw.maxdgram=16384" will increase the max
ICMP datagram size on Mac, allowing you to use -s 9000 on MacO

Windows
ping -f -l 8000 [Delphix_Engine_IP]

http://www.mylesgray.com/hardware/test-jumbo-frames-working/
```

g. Measure Network Bandwidth and Latency:
   i. Latency in both directions should be < 1ms for an 8KB payload.

ii. Network Hops should be minimized: `traceroute (Unix/Linux) / tracert (windows).`
iii. Throughput in both directions: 50-100 MB/s on 1 GbE, 500-1000 MB/s on 10 GbE physical link.
iv. Always use the CLI when possible.

h. NIC should use Auto-negotiate on Ethernet with a minimum of 1000Mbps.
   - Hard setting speed/duplex will limit network throughput below the line rate.

3. **Delphix <===> Staging Server**(SQL Server, Sybase):
Measure latency, bandwidth for transaction log restore performance
4. **Source <===> Delphix:** Measure latency, bandwidth to explain snapsync performance
5. **ESX host <===> ESX host**(ESX Cluster):
   a. This is because the entire memory footprint of the Delphix VM (more precisely, the entire address space of the ESX processes that comprise the VM) must be copied to the receiving ESX host, along with all changes to that address space as they happen. Our ZFS cache comprises the bulk of that memory and changes as I/O occurs, and that rate of change is governed by the network bandwidth available to the Delphix Engine.
   b. Attempting to live vMotion a DE with a 10Gb NIC over a 1Gb vMotion network is thus likely to fail.

Architecture best practices for storage

1. Configuration:
   a. Virtual Disks (VMDK) with spinning or tiered media must be thick provisioned + lazy zeroed.
      - For storage which is 100% SSD/EFD/Flash-based, continue to thick provision, however, eager zero is not necessary.
   b. VMDKs may be homed on VMFS or RDM storage, however, VMFS is much more common and generally preferred.
   c. Regardless of VMFS/RDM selection, physical LUNs must have uniform characteristics (RAID, spindle count, tier) and should be thin provisioned to save time.
   d. Storage allocated must be identical between each vSCSI controller.
   e. The supported maximum of four virtual SCSI controllers (PVSCSI (default) or LSI Logic Parallel) must be enabled. A mix of different types of SCSI controllers is not supported within the engine.
   f. Virtual Disks must be evenly distributed across the 4 virtual SCSI controllers. For example, 8 virtual disks should be configured as 2 disks per controller: SCSI (0:0), SCSI (0:1), SCSI (1:0), SCSI (1:1), SCSI (2:0), SCSI (2:1), SCSI (3:0), SCSI (3:1).
      - You don't need to account for the OS in the even distribution of disks across controllers, just pick one; the OS doesn't place a substantial load on the controller.
   g. To provision VMDK disks over 16TB in size, the vSphere web client must be used, the win32 client will return an error.
   h. As of 5.1.3, we require 127GB for the system disk. older versions defaulted to 150GB.
      - Due to our unified OVA for masking, the 127GB requirement also applies to Masking Engines
   i. VMDK for the Delphix Engine OS is often stored on the same VMFS volume as the Delphix VM definition file (aka VMX). In that case, the VMFS volume must have sufficient space to hold the Delphix VMX Configuration, the VDMK for the system disk, a swap/paging area if the memory reservation was not enabled for the Delphix Engine (or to suspend the system), and any related VMware logging.
   j. Set ESX Storage Multipathing IO optimization (KB 2072070).
      i. Set multipathing policy to round-robin.
      ii. Set path switching IO operation limit to 1 (default 1000).
   k. Verify that the queue depth setting on the virtual disks used for VMware is appropriate based on the minimum of the HBA type or the underlying storage's combined queue depth for all hosts attached to the same controller
      - See this VMware KB article for how to check/set the queue depth
2. Testing:

   a. Run Storage Performance Tool on the raw storage **before any engine configuration**. This is a one-time opportunity for each engine upon installation
   b. Required maximum storage latency is < 2ms for writes and < 10ms (95th percentile) for small random reads. Minimum passing grades: 4KB/8KB reads (B-), 1MB reads (A), 1KB/128KB writes (A).
   c. If working with the Delphix Professional Services team, we would expect to run additional baseline performance measurements via composite tools and scripts.
      - e.g. "Sanity Check" (Oracle) or "DiskSpd" (MSSQL)
3. Detail Discussion:
   a. Before beginning any discussion on storage performance, or at the beginning of the installation, collecting the following specs from your storage administrator will assist in understanding
   b. Vendor, Model (For example: EMC VMAX 40k, HP 3PAR StoreServ 7000)
   c. IO latency SLO (For example 5ms 99.999%)
   d. IOPS/GB SLO (For example: 0.68 IOPS/GB for EMC Gold-1)
   e. Cache type and size (For example FAST cache 768GB)
   f. Tier, #Pools; if auto – tiering; relocation schedule (For example: Gold/Silver/Auto/3 pools/etc)
   g. Pool detail: (#) drives, RPM, Type (For example: Pool1: (20) EFD, (30) 15k SAS, Pool 2: (40) 10k SATA)
   h. Connection (For example: 16Gb Fibre Channel, 10Gb iSCSI, 20Gb N

Architecture best practices for Delphix engine data protection

1. For protection against physical host failure - leverage VMware HA.
2. For protection against storage failure - leverage Delphix replication.
3. For protection against administrative error - leverage storage snapshots.
4. For protection against site failure - leverage Delphix replication and/or Delphix Live Archive.
5. Infrastructure Backup of the Delphix VM: Must take a consistent group snapshot of all Delphix VM storage (system disk, VM configuration, database VMDK/RDMs) RTO, RPO are inferior compared to Delphix Replication. The granularity of restore is at the VM level: ***all-or-nothing.***
   a. Virtual Machine Backup: Create VM snapshot, backup (proxy server), remove the snapshot. Products use VMware APIs: NBU for VMware, TSM, Networker Limited to < 2TB because of time to backup, impact on running VM
   b. Storage Array Backup: Take a consistent storage snapshot, replicate to tape/VTL media server, remove the snapshot. Products include Hitachi Shadow Copy, EMC SnapCopy, HP Business/Snap Copy.

Architecture best practices for Masking

1. As of release 5.0, the virtualization and masking functions are combined into a single OVA and require additional consideration for installation and configuration. Additionally, support for remote Continuous Compliance Engine calls has been implemented and is supported in 5.0.4 and above.
2. Continuous Compliance Engines should continue to be deployed to hosts dedicated to that function.
   - Possible exceptions to this would be when any virtualization needed is extremely low and unlikely to be heavily impacted by masking requirements.
3. The Masking Engine is disabled by default.
   - The Delphix Engine will continue to remain running (but unused) on a Masking-only VM.
4. The standard configuration for a dedicated Continuous Compliance Engine:
   a. 8 vCPUs
   b. 16GB RAM minimum, 32 GB RAM or more recommended.
   c. 300 GB storage for the OS/system root disk is required for the OS (5.1.4 and greater).
   d. 50 GB storage for the data disk must be added during the initial configuration via the Engine Setup wizard. (the engine will not complete its setup without a separate data disk).
   e. If a bulk operation is used, allocate extra space equivalent to the size of all datasets (tables) that will be masked concurrently.

i. As a rule of thumb: *Disk Space Required for Bulk = ((Total Database Size * .66) * .10) where Raw data = Total Database Size * .66*
ii. 10% change is an estimate based on our experience for data we mask. Often it is lower but there are exceptions such as masking a data warehouse with a large fact table and a bunch of much smaller tables.
iii. The VMDK for the engine OS is often stored on the same VMFS volume as the VM definition file (aka VMX). In that case, the VMFS volume must have sufficient space to hold the VMX Configuration, the VDMK for the system disk, and any VMWare logging.
iv. Additional VMFS space for swap/paging is required if RAM reservations are not enabled. (The VM will not start if reservations are lacking and disk space is not available for swap)

CPU utilization

1. One vCPU per concurrent masking job is considered a best practice.
2. Dependent on the algorithms used: Some are calculations such as ones using AES encryption and others are lookups and tend to do more I/O.

Memory utilization

1. The Continuous Compliance Engine uses its memory to cache data. More memory will provide better performance. 1GB per masking job is considered a best practice.
2. Dependent on memory settings in the Masking Engine and JVMs. An increase in parallel workloads will require more memory. Data is either cached directly or using Kettle so the larger the lookups for algorithms the more memory required. This is the first thing to look at for performance issues.

Network and I/O

Continuous Compliance leverages the Target DB server and VDB for most of the workload. This means the masking engine can be I/O bound waiting for the DB server. As long as the masking engine can read the data faster than it can process it this is not an issue. Slow networks with numerous hops between the DB server and the Masking server can cause performance problems. Co-locating the masking server with the DB server is recommended in these cases.

Masking VDB tuning

1. Always start with the tuning recommendations for Target servers and VDBs first. If the VDB is not performing well, the performance of masking will suffer.
2. For Oracle, it is critical to select no archivelog mode and tune online redo log size at provision time.
3. For SQL Server, the VDB should be in SIMPLE recovery with an appropriate log file and TempDB sizes.

Backup of a continuous compliance engine

1. Virtual machine backups are recommended for versions of software in which masking runs in its own VM – in other words, the masking VM is separate from the VM(s) where virtualization takes place.
    a. If an engine is supporting both masking and virtualization, review data protection best practices.
    b. Although XML exports of inventories and environments do exist, they are incomplete. Do not rely on them.
2. In-Place (not On-the-Fly) Masking is the primary use case.

The following recommendations apply to the Continuous Compliance Engine versions 5.0.2 and earlier:

1. Jobs vs. Streams:
    a. If there are multiple tables to be masked concurrently, use multiple, separate jobs – one per table.
    b. Avoid multiple streams due to internal limitations.

⚠ 1. The default setting for streams is 20; set it to 1. This will force serialization (one table at a time) if the job contains multiple tables.

2. Use one update thread per job; this avoids block collisions/contention during the UPDATE phas.

> ⚠ The default setting is 4; set it to 1.

a. Identify ALL indexes, constraints, and triggers on columns being masked (and only on columns being masked).

b. Evaluate whether it is better to drop/mask/recreate for indexes, or disable/mask/re-enable for triggers and constraints – as compared to leaving in-place during masking. The best choice depends on the situation

3. For Oracle VDBs, use ROWID for SQL UPDATE of masked row value(s).

1. Edit the ruleset, select Edit All Logical Keys, enter ROWID as the logical key value; see Managing Rule Sets. When a single, large, non-partitioned table must be masked by concurrent jobs (each masking a subset of the table) to shorten masking elapsed time, segregate jobs by database block/page to avoid contention and locking conflicts. In other words, each job masks a unique set of blocks; each block is masked exclusively by one job.
2. If a single table is being masked by multiple, concurrent Jobs, and Indexes/constraints/triggers must be dropped/recreated or disabled/enabled, these must be performed OUTSIDE of masking Jobs.
3. Pre-masking and post-masking steps must be created manually.
4. Scheduling of pre-script and post-script jobs must be devised. Plan to scheduled/execute externally.

Architecture best practices for source DB and OS settings

Oracle

1. ARCHIVELOG must be enabled: `select log_mode from v$database.`
2. FORCE LOGGING should be enabled to ensure VDBs are not missing data. When NOLOGGING redo is applied during provision, the resulting VDB will be missing changes. Tables with NOLOGGING changes will throw corruption errors when scanned.
3. Block Change Tracking should be enabled to minimize snapsync time.
4. Consult the documentation for Oracle Standby sources.
5. If the database is encrypted with Oracle TDE (Transparent Data Encryption) plan your Delphix storage requirements with the expectation of minimal compression. Customer Observation: space usage for a TDE dSource copy was 92% (2.44 TB) of the source database size (2.67 TB). A typical Oracle dSource copy for a non-TDE database consumes 40% of the source database size.

SQL server

1. FULL vs SIMPLE recovery mode trade-off.
2. The maximum size of an MSSQL database that can be linked is 256TB for Windows versions greater than 2003. The limit is 2TB for Windows 2003. (See Linking a SQL Server dSource)

Architecture best practices for target DB and OS settings

Target database application settings

1. Oracle:
    a. Provision with 3 x 5GB online redo logs (minimum) to avoid pause when transaction logs wraparound.
    b. A provision in NOARCHIVELOG mode to reduce transaction log IO. Masking, Test, QA VDBs rarely need point-in-time rewind

c. Always check initialization parameters inherited from a parent, remove any expensive or irrelevant parameters.
   i. `DB_CACHE_SIZE, SGA_TARGET`: set based on the target system being compared to.
   ii. `FILESYSTEMIO_OPTIONS` to `SETALL`. Any other setting inherited from the source is probably wrong.
   iii. `DB_BLOCK_CHECKSUM, DB_BLOCK_CHECKING, DB_LOST_WRITE_PROTECT, DB_ULTRA_SAFE`: set to default values to minimize the impact.
   iv. `PARALLEL_DEGREE_POLICY` to `AUTO`, `PARALLEL_MAX_SERVERS` default, `PARALLEL_EXECUTION_MESSAGE_SIZE` to 32768 (maximum): improve PQ performance.
   v. `FAST_START_MTTR_TARGET:` drives steady write activity. Set based on the target system being compared to.
   vi. Consider non-durable commits for Masking, Test, QA, UAT: set `COMMIT_WAIT = NOWAIT, COMMIT_LOGGING = BATCH`
d. Use Oracle Direct NFS (dNFS) for 11.2.0.4+ (**unstable** on older releases):
   i. Recommended documentation:
      - Configuration Examples and Troubleshooting blog from Helmut Hutzler
      - Sample oranfstab to leverage multiple network paths for Delphix VDB
   ii. Set DNFS_BATCH_SIZE = 128 (default is 4096). This is a good starting point and sufficient for most workloads.
   iii. Tune TCP stack: set `tcp_adv_win_scale` = 2 due to workaround hard-coded Oracle dNFS TCP buffer size.
   iv. Check Alert Log, V$DNFS_SERVERS, V$DNFS_FILES, V$DNFS_STATS to verify proper working (sample here).
e. Create AWR snapshots around a reference customer workload, generate an AWR report.
   i. AWR snap before/after workload: `SQL> exec DBMS_WORKLOAD_REPOSITORY.CREATE_SNAPSHOT();`
   ii. AWR report between the snaps: `SQL> @?/rdbms/admin/awrrpte`
f. Generate ASH report to diagnose bottlenecks while a workload is running.
   - `SQL> @?/rdbms/admin/ashrpt`
g. Run synthetic benchmark `sc-workload.`
h. Where `db file scattered read` (multiblock cached read) latency is high consult this Support KB: How to Mitigate Multi-Block Read Performance on Oracle 10g
i. Improve distributed query performance by modifying dblinks to use local IPs instead of SCAN IPs.
j. NFS recommended mount options for Oracle RAC/SI: Oracle Support Note 359515.1.

Target Host OS Settings

1. Existing documentation on Target OS practices: Target Host OS and Database Configuration Options
2. HP-UX 11.31+
   - Async NFS Direct I/O: HP-UX requires Oracle `disk_asynch_io` turned off for filesystems
3. IBM AIX:
   a. Consult IBM documentation on AIX TCP Tuning
   b. AIX TCP Tuning Prezo
4. Windows:

a. Anti-virus programs can impact both performance and operation. Delphix recommends anti-virus scanning exclude folders where Delphix files are maintained, in addition to the normal exclusions put in place for MSSQL operation.
b. Delphix Connector (aka DX Connector):
    i. Plan 3-5GB for the Delphix Connector installation.
    ii. Windows does not yet have ssh, so Delphix developed the "DX Connector for Windows target host communication.
    iii. The connector must be installed on all Target Windows hosts.
    iv. The connector supports two modes – v1 and v2 both use the same application binaries.
    v. The connector v1 process is used to bootstrap the v2 process on a target. This opens a DSP session back to the Delphix Engine (The same thing is done via SSH on U*nix Targets)
    vi. v2 mode is required to enable SQL hooks
    vii. The connector can always be downloaded from a local Delphix Engine at: http://<delphix_engine>/connector/DelphixConnectorInstaller.msi.
    viii. The connector is backward compatible, so it is not always necessary to upgrade it during a Delphix upgrade.
c. iSCSI connections:
    i. Read the following for general awareness of iSCSI limits
    ii. In addition to the hard limits on iSCSI connections, consideration must be given to the RAM, CPU, and Network to provide sufficient resources for the load on any Target or Staging host.
    iii. To increase the iSCSI timeout on both Target and Staging hosts.
    iv. In certain circumstances, it's possible that iSCSI startup will not complete before the SQL Service attempts to start a database. In such circumstances, it can be helpful to ensure the SQL service depends on the iSCSI service.
        - Example: c:\> sc config "MSSQLServer" depend="Microsoft iSCSI Initiator Service"
    v. **Note** that any changes to iSCSI are system-wide and could potentially impact other applications also leveraging that feature.
d. Enable Receive Side Scaling (RSS) on each network interface that Delphix will be connecting to.

Architecture best practices for Staging Targets

This host is called a "staging target" because it has much in common with other targets, such as the remote storage mount to the Delphix Virtualization Engine. Because it leverages remote storage over the network, the staging target only needs enough disk capacity for the OS, database application, and any relevant logs or tools. A staging target is a requirement on all platforms that Delphix supports except Oracle, but you can also use it for Oracle to replay logs leveraging validated sync.

Memory and CPU

1. 32 GB RAM minimum
2. 4 vCPU minimum

General guidance for staging servers (Multi-platform)

1. Delphix recommends dedicated Staging servers for role/architecture separation. However, any Target server can be used as Staging.
2. In cases where the same server is used as both Staging and Target, we strongly recommend a dedicated instance/install for staging to avoid confusion.
3. Delphix recommends at least one Staging server per Delphix Engine to avoid the possibility of a single point of failure across multiple engines.
4. If a staging server is shared among multiple Delphix Engines, please ensure that a dedicated SQL Server Instance is created for each Delphix Engine.
5. Configuration / performance factors:

a. Transaction log generation rate.
b. Number of VDBs.

⚠ Precise guidance on these items has not yet been defined. In general, if there is a heavy log generation rate and few VDBs, the ideal recommendation is to have at least 1 Staging Target per Delphix Engine.

Disk / Local storage

1. The only local storage needed is for the OS and application with default databases.
2. Storage for a staging database is provided from the Delphix Engine, which is mounted over the network similar to any Target host (NFS/iSCSI).
3. If the customer has a standard DB server build, their standard storage sizing is probably fine.
4. If a recommendation is still needed, suggest 30GB for OS and application and any tools needed.

Network requirements

1. The Staging Target engages in network data transfers between Staging and the Source backup shared location as well as between Staging and the Delphix Engine.
2. The Staging Target is also a Target server, and as such should have < 1ms latency to the Delphix Engine (and low latency to the Source backup, when possible).
3. If the change rate on the Source database(s) is > 1 Gb/sec, the recommended network bandwidth to support network transfers is 10 Gb/sec.
4. In cases where only 1 Gb/sec network bandwidth is available, segregation of each network is recommended to reduce network contention.
5. Ensure that the virtual NIC is using the standard vmxnet3 adapter and not Intel for VMWare based clients. Logical IO errors have been reported while using Intel instead of vmxnet3 adapter.

Windows and MSSQL Specific

1. An MSSQL Server Instance used for Staging should not be clustered.
2. Staging should not be hosted on Windows 2003 - extended support ended July 14, 2015. It is also the first Windows version with iSCSI support and is not ideal.
3. The SQL Server Instances hosted on the Staging Target should have a Maximum Memory set. Also ensure that at all times, at least 10% of total memory is available for OS operations.
4. Only system databases (Master/MSDB/Temp/MSDB) are kept on local storage. All other data is read/written to the Delphix Engine.
5. Windows iSCSI configuration and limits for v2p, target, and staging hosts.
6. Ensure that Receive Side Scaling (RSS) is enabled on every network interface that Delphix will be connecting to.

Best practices for hypervisor host and VM guest

Hypervisor

1. ESXi 6.x or 7.0 is recommended. ESXi 5.5 or earlier is no longer supported.
2. HyperThreading (HT) for Intel®-based servers (no HT on AMD CPUs).
    a. Disable HT in BIOS, on the ESXi Host, **and** disable HT sharing on the Delphix VM for consistency. This is our best practice, disable at all levels.
    Any other combination may result in a non-deterministic performance.
    b. When HT cannot be turned off for both the Host and Delphix VM, it should be turned on at all levels, not run in a "mixed-mode".
        i. HT disablement at a guest level only can result in non-deterministic performance.
        ii. A dedicated ESXi host, cluster or DRA is recommended where consistent VDB performance is paramount.
    c. VM migration to a new host (e.g. VMware HA or vMotion) can create mismatched HT settings.
3. ESXi overhead (resources required for hypervisor cannot be reserved, they must be left unallocated).
    a. Memory Overhead - 10% of available RAM must not be allocated to guest VMs.
        - *Example: 256GB RAM, allocate 230GB to Delphix VM, leave 26GB for ESX*
    b. CPU Overhead - At least 2 cores (ideally 4) must not be allocated to guest VMs.
        - *Example: If 16 physical cores are available, allocate 12 to the virtual machines, leaving 4 for the hypervisor*
        - Why 4? Certain hypervisor functions require precedence over any virtualized system. If a hypervisor needs more CPU than the amount currently available, it can de-schedule all other virtual processes to ensure adequate CPU resources for the hypervisor. Ensuring the hypervisor will not have to de-schedule any running virtual processes (worlds) by setting aside and not over-subscribing CPUs for virtual functions will leave them available for hypervisor use.
    c. Even if the Delphix VM is the only VM on a host, the hypervisor is still active and essential; and still needs resources.
4. BIOS Power Management should be set to High Performance where ESXi controls power management.
    a. It can be impacted by VMware KB 1018206 - poor VM application performance caused by power management settings.
    b. Ensure that all BIOS managed C-States other than C0 are disabled if power management is hardware controlled.
    c. Ensure that all ACPI sleep states above S0 are disabled in the BIOS.
    d. Examples for popular server lines from Cisco, HP, Dell below. *Specific models will vary, use the appropriate spec sheet.*
        i. UCS: disable the Processor Power States, disable Power Technology, set Energy Performance to "Performance"
        ii. HP Proliant: set HP Power Regulator to HP "Static High Performance" mode
        iii. Dell: set BIOS System Profile to "Performance Optimized" mode
5. VMware HA can be enabled; VMware DRS is generally disabled.
6. Blade/Rack Server Firmware and ESXi Drivers should be updated to the latest versions.
7. For Intel®-based servers with E5-2600 v2 processors.
8. Two typical server configurations:
    a. Blade Farm
    b. Rack Server
    Hyper-Converged configurations are possible for high performance.

Virtual machine guest

1. For VM machine settings, see Virtual machine requirements for VMware platform
2. VMWare Guest Specifications:
    a. Minimum: 8 vCPU x 64 GB
    Small: 8 vCPU x 128GB
    Medium: 16 vCPU x 256 GB
    Large: 24 vCPU x 512 GB
    b. Reserve 100% of RAM and CPU:
        - If the ESX host is dedicated to Delphix, CPU and RAM reservations are advised but not necessary, however, swap space will be required on the hypervisor to compensate for the lack of reserved RAM.
    c. Hyperthreading - See the ESX host section at the top. Disable HT Sharing on VM, disable HT on ESX Host.
3. Assign single-core sockets for vCPUs in all cases. If there is a compelling reason to use multi-core vCPUs, reference the following article from VMware which describes matching virtual multi-core sockets to the hardware ESX is running on.
    VMware article on CoresPerSocket
    Example: *ESXi Host has 2 socket x 18 core Intel Xeon, Delphix Engine wants 16 vCPU.*
    Configure Delphix VM with 2 Virtual Sockets, 8 Cores Per Socket to utilize hardware architecture.
4. Avoid placing other extremely active VMs on the same ESX host.
5. Monitoring - vSphere Threshold Alerts for CPU, Network, Memory, Capacity.
6. To set the number of vCPUs per virtual machine via the vSphere client, please see "*Virtual CPU Configuration"* in the Administration guide:
    ESXi 6.0
    a. Delphix VM CPU utilization - Delphix KB article on what makes Delphix VMs similar to other resource-intensive applications
    b. Exchange on VMware best practices - VMworld 2013 session
    c. ESXTOP reference, blog
7. Ensure that the latest available VMware drivers and firmware versions are installed for HBAs, NICs and any other hardware components configured on the Delphix virtual machine. This is a critical step that can have a massive impact on the performance and robustness of our solution.

Best practices hypervisor host and VM guest ESX FAQ

Frequently asked questions

Why does Delphix require a minimum of 8 vCPUs and recommend 128 GB per 8 vCPUs?

8 vCPUs are not only a standard licensing block, but they are also key to meeting our 10Gbps single-engine throughput potential and help to sustain low latency for VDBs.

As with CPU, cache memory is required to drive peak loads on the Delphix Engine. More memory allows for more blocks to be read from the cache rather than going to less performant disks. Delphix stores cached data in a compressed format and only keeps a single copy of unique blocks in memory. These features give read performance across multiple VDBs provisioned from a single source dramatic improvements in speed, scalability and memory utilization.

Why does Delphix request reservations for CPU and memory?

Delphix performance can be greatly impacted there is contention over CPU or RAM. Reservations allow the engine to explicitly control those resources and avoids the possibility of contention with other VMs, even when resources are overcommitted.

Why does Delphix request Hyper-Threading (HT) be disabled?

Hyper-Threading can have a positive impact on many, but not all applications. Delphix has a different execution profile that does not benefit from Hyper-Threading. Factors such as significant memory bandwidth requirements and a high level of parallelism which requires a high number of shared locks mean that HT and Delphix do not generally work well together.

Why does Delphix request 4 controllers, and why must the storage be identical between them?

To provide optimal storage performance, you must spread data equally over the maximum (4) virtual SCSI controllers. To provide consistent performance between each of the four controllers, you need to ensure storage is identical between them.

Why must virtual disks (VMDKs) be thick provisioned and eager-zeroed?

Thick provisioning and eager zero ensure performance is top-notch from the start with no hiccups from expanding virtual resources.

Why is 20% free space required?

While the ZFS file system has a lot of features leveraged by Delphix, it loses efficiency as space decreases. 20% is the minimum that must be available for best performance.

Why does Delphix request you reserve CPU and RAM for Hypervisor overhead?

Based on VMWare's resource management guide and our own experiences with high IO throughput. Note there is no specific mechanism to assign resources to the hypervisor, the only way to preserve overhead is by not allocating resources to guests.

Why does Delphix generally want VMWare HA enabled, but DRS disabled?

VMware HA (High Availability) addresses outages that occur when a physical host goes down or is completely offline, by migrating the guest(s) to another physical host and restarting them. There is no real downside, it simply brings unavailable servers back online.

VMware DRS (Distributed Resource Scheduler) is for load balancing host resources in a cluster. Because of high IO and best practices configuration for optimal performance, our engine is typically not a good candidate for relocation.

Why does Delphix request you set power management to High-Performance Mode?

This will ensure power management will never impact performance by entering into a lower power state (also known as c-state).

Best practices for network configuration

1. For more information about network configuration refer to Network performance configuration options
2. Delphix engine <===> Target Host:*Implement standard requirements for optimal NFS/ISCSI performance:*
    a. Optimal physical network topology:
        - Low latency: < 1ms for 8K packets.
        - Network adjacency: minimize network hops, co-locate in the same blade enclosure, co-locate on the same physical host.
        - Eliminate all Layer 3+ devices - firewalls, IDS, packet filters (Deep Packet Inspection - DPI).
        - Multiple switches can add latency and fragmentation and reordering issues will add significant latency.
    b. Optimal throughput:
        - 10GbE physical uplinks or higher.
        - Jumbo frames (typically MTU 9000) improves network efficiency for 8K packets: lower CPU, lower latency, greater throughput.
        - All devices end-to-end *must* be configured for the larger frame size including switches, routers, fabric interconnects, hypervisors and servers.
        - Traffic between two VMs on the same host is limited to ~16Gbps when leveraging built-in virtual networking.
    c. Optimal logical flow:
        - Disable QoS throttles limiting network utilization below line rate (e.g. HP Virtual Connect FlexFabric).
        - Consider a dedicated VLAN (with jumbo frames) for NFS/iSCSI traffic.
            - VMware KB-configuring iSCSI port binding with multiple NICs in one vSwitch for VMware ESXi 6.0.x (2045040)
            - VMware KB-considerations for using software iSCSI port binding in ESX/ESXi (2038869)
    d. NIC Teaming (at ESX layer) of multiple physical uplinks can provide additional throughput for higher workloads.
        i. Examples: 4 x 1Gb NICs support up to 400 MBPS IO, 2 x 10Gb NICs support up to 2 GBPS IO.
        ii. VMware KB-1004088 has NIC teaming recommendations, including `route-based-on-IP-hash` policy.
        iii. VMware KB-1001938 has host requirements for physical link aggregation (LACP, EtherChannel).
        iv. VMware KB-1007371, popular blog post details problems with NIC selection using `dest-IP` hash.
    e. **Jumbo frames check via ping**

```
Delphix Engine
$ ping -D -s [Target_IP] 8000
"ICMP Fragmentation needed and DF set from gateway" indicates MTU < 8028

Linux
$ ping -M do -s 8000 [Delphix_Engine_IP]
"Frag needed and DF set (mtu = xxxx)" indicates MTU < 8028

MacOS
ping -D -s 8000 [Delphix_Engine_IP]
```

```
Note: "sudo sysctl -w net.inet.raw.maxdgram=16384" will increase the max
ICMP datagram size on Mac, allowing you to use -s 9000 on MacO

Windows
ping -f -l 8000 [Delphix_Engine_IP]

http://www.mylesgray.com/hardware/test-jumbo-frames-working/
```

Copy

f. Measure network bandwidth and latency:
   i. Latency in both directions should be < 1ms for an 8KB payload.
   ii. Network Hops should be minimized: `traceroute (Unix/Linux) / tracert (windows).`
   iii. Throughput in both directions: 50-100 MB/s on 1 GbE, 500-1000 MB/s on 10 GbE physical link.
   iv. Always use the CLI when possible.

g. NIC should use Auto-negotiate on Ethernet with a minimum of 1000Mbps.
   - Hard setting speed/duplex will limit network throughput below the line rate.

3. **Delphix <===> Staging server**(SQL Server, Sybase):
   Measure latency, bandwidth for transaction log restore performance
4. **Source <===> Delphix:**
   Measure latency, bandwidth to explain snapsync performance
5. **ESX host <===> ESX host**(ESX Cluster):
   a. This is because the entire memory footprint of the Delphix VM (more precisely, the entire address space of the ESX processes that comprise the VM) must be copied to the receiving ESX host, along with all changes to that address space as they happen. Our ZFS cache comprises the bulk of that memory and changes as I/O occurs, and that rate of change is governed by the network bandwidth available to the Delphix Engine.
   b. Attempting to live vMotion a DE with a 10Gb NIC over a 1Gb vMotion network is thus likely to fail.

Best practices network FAQ

Frequently asked questions

Why does Delphix request 10GE Ethernet?

As a matter of physics and standards - 10 gigabit (Gb) Ethernet can sustain approximately 1 gigabyte (GB) per second of throughput. With all our best practices applied, a Delphix Engine can achieve very close to that line speed, allowing for optimal load, engine, and license utilization. Lower network speeds may be acceptable for low loads, while in some environments NIC teaming (e.g. LACP) may be required for top speeds.

Why does Delphix require < 1ms latency to TARGET servers and < 50ms to SOURCE servers?

Delphix leverages NFS and iSCSI (depending on platform) for live TARGET DB mounting over the network, so it's imperative that latency is as low as possible. Data coming from SOURCE servers is not generally as time-sensitive, so you need a minimum latency of < 50ms to ensure operational integrity.

Why does Delphix request Jumbo Frames?

Jumbo frames increase the Ethernet maximum transmission unit (MTU) from the default 1500 bytes to 9000 bytes. This has several effects such as decreasing CPU cycles by transferring fewer packets and increasing the engine throughput. You will find jumbo frames have a 10-20% real-world impact and are required (along with all other best practices) to handle peak loads of 800 - 1000MB/s on an 8 vCPU engine with a 10Gb network.

How does Delphix avoid communication impact with non-jumbo frame hosts when Jumbo Frames are enabled on the Delphix Host?

Path MTU Discovery is the mechanism by which two hosts agree on the MTU leveraged for communication between them. This mechanism will ensure communication between both standard and Jumbo Frame enabled hosts works as expected.

When does Delphix recommend NIC teaming?

The Delphix Engine is capable of high throughput, but not every enterprise has sufficient network bandwidth to support it. Teaming is a less expensive way of increasing the bandwidth when compared to new hardware.

Why does Delphix recommend logical and physical and co-location?

The Delphix Engine leverages network connections extensively, so optimizing the latency whenever possible is very important - sometimes critical.

Best practices for Delphix engine data protection

The following are recommended best practices for data protection in the Delphix Engine.

1. For protection against physical host failure - leverage VMware HA.
2. For protection against storage failure - leverage Delphix replication.
3. For protection against administrative error - leverage storage snapshots.
4. For protection against site failure - leverage Delphix replication and/or Delphix Live Archive.
5. Infrastructure Backup of the Delphix VM:
   Must take a consistent group snapshot of all Delphix VM storage (system disk, VM configuration, database VMDK/RDMs)
   RTO, RPO are inferior compared to Delphix Replication. The granularity of restore is at the VM level: ***all-or-nothing.***
   a. Virtual machine backup: Create VM snapshot, backup (proxy server), remove the snapshot.
      Products use VMware APIs: NBU for VMware, TSM, Networker
      Limited to < 2TB because of time to backup, impact on running VM
   b. Storage array backup: Take a consistent storage snapshot, replicate to tape/VTL media server, remove the snapshot.
      Products include Hitachi Shadow Copy, EMC SnapCopy, HP Business/Snap Copy.

Best practices Delphix engine data protection FAQ

Frequently Asked Questions

Why does Delphix recommend SAN snapshots or Delphix replication for backup?

There are a few possible methods for data protection of the Delphix Engine. Those methods are SAN snapshots, Delphix replication, and virtual machine snapshots (for very small engines only). Because the Delphix Engine is itself a backup of source environments, many customers simply plan to rebuild in the event of a disaster.

What is the DXToolkit, and how can it help?

The professional services team has created a Perl-based “DXToolkit” which can help export and import certain configuration data over web services. This toolkit can be leveraged to assist with would normally be a manual re-install outside of the above methods.

For further detail around data protection, please speak with your Delphix contact.

Can I use a VMware-based backup solution such as VEEAM to backup my Delphix Engine?

Yes, VMWare backup solutions are useful for backing up guest VMs. However, Delphix suggests that you only use this approach for Delphix Engines which have a smaller storage footprint (perhaps < 2 TB) and are less active.

Running this type of backup puts a load on the environment, which might adversely impact Delphix VM performance.

Can I use a VMware snapshot for backing up Delphix for a small window – for example, during an engine upgrade?

Yes. However, even though snapshots are instantaneous, they track changes separately from the base disks and can grow to consume as much space as the original.

> ⚠ Upgrades, in particular, can change substantial amounts of data.
> If you lose physical disks, snapshots are useless because it needs them to make up the current state of VM.
> A Delphix Engine is often allocated multiple terabytes of storage and is often very busy due to load aggregation from virtual databases on multiple target servers, so this approach may be challenging.
> Snapshots cannot detect storage corruption.

Can I use a Storage snapshot solution to protect Delphix against Storage and Delphix corruption?

Yes. However, please note that the caveats which apply to VMWare snapshots will also apply here.

> ⚠ A specific concern related to storage layer snapshots is that you must create a consistency group that contains both the OS and Data disks.

Can I use RMAN to backup my VDBs just like a Physical database to provide extra protection?

You can backup Delphix VDBs using Oracle RMAN tools, but the recovery database would first require re-hydration of that VDB, which might take up equivalent production storage space.

Furthermore, that re-hydrated database needs to be brought into the Delphix framework as a dSource, after which you can provision a VDB to complete recovery. The whole process might take hours or days to recover.

The best approach is to use the VDB Snapshot capability to backup VDB frequently and then leverage Delphix Replication capability to protect underlying Delphix storage, which holds that VDB snapshot.

Best Practices for source DB and OS settings

Best practices

Oracle

1. ARCHIVELOG must be enabled: `select log_mode from v$database.`
2. FORCE LOGGING should be enabled to ensure VDBs are not missing data. When NOLOGGING redo is applied during provision, the resulting VDB will be missing changes. Tables with NOLOGGING changes will throw corruption errors when scanned.
3. Block Change Tracking should be enabled to minimize snapsync time.
4. Consult the documentation for Oracle Standby sources.
5. If the database is encrypted with Oracle TDE (Transparent Data Encryption) plan your Delphix storage requirements with the expectation of minimal compression. Customer Observation: space usage for a TDE dSource copy was 92% (2.44 TB) of the source database size (2.67 TB). A typical Oracle dSource copy for a non-TDE database consumes 40% of the source database size.

SQL server

1. If using a FULL recovery model, configure your dSource to stay synchronized using Transaction Log backups. This will usually allow the dSource to stay in sync using much less network and disk IO than Full or Differential backups.
2. Ensure that the number of Virtual Log Files (VLFs) in the Source database is appropriate. Databases with hundreds or thousands of VLFs will experience slower provisioning times as SQL Server must do more work during recovery. The blog post A Busy/Accidental DBA's Guide to Managing VLFs has helpful information here.
3. Review index maintenance and defragmentation operations on your Source database, to ensure that they are not running more often than necessary. Many maintenance operations are logged, and can result in extremely large log backups without much benefit to the database server. See Stop Worrying About SQL Server Fragmentation.
4. If configuring a dSource to use Full or Differential backups, ensure that Source database backups do not run at the same time as database maintenance or large batch operations. This can significantly increase the amount of time required to perform database recovery and will slow down the provisioning of VDBs.
5. For best compatibility with the Delphix Engine, avoid taking log backups more frequently than every 10 minutes. The validated sync process requires time to detect and validate log backups before they can be applied to a dSource, and extremely high-frequency log backups can make it difficult for a dSource to stay in sync.
6. If you are using SQL Authentication for database logins, and you want Source database users to be mapped to database logins when provisioning VDBs, ensure that the SID of your database logins is the same between your Source and Target environments (the passwords can be different). For more details, see the Correcting Orphaned SQL Server Database Users (KBA1111) KB article.
7. To maximize dSource performance, ensure that backups are being taken to a disk drive or network location that can be accessed by the Staging Server at high speed.

Best practices for target DB and OS settings

Target database application settings

1. Oracle:
    a. Provision with 3 x 5GB online redo logs (minimum) to avoid pause when transaction logs wraparound.
    b. Provision in NOARCHIVELOG mode to reduce transaction log IO. Masking, Test, QA VDBs rarely need point-in-time rewind
    c. Always check initialization parameters inherited from parent, remove any expensive or irrelevant parameters.
        i. `DB_CACHE_SIZE, SGA_TARGET` : set based on target system being compared to.
        ii. `FILESYSTEMIO_OPTIONS` to `SETALL` . Any other setting inherited from source is probably wrong.
        iii. `DB_BLOCK_CHECKSUM, DB_BLOCK_CHECKING, DB_LOST_WRITE_PROTECT, DB_ULTRA_SAFE` : set to default values to minimize impact.
        iv. `PARALLEL_DEGREE_POLICY` to `AUTO` , `PARALLEL_MAX_SERVERS` default, `PARALLEL_EXECUTION_MESSAGE_SIZE` to 32768 (maximum): improve PQ performance.
        v. `FAST_START_MTTR_TARGET:` drives steady write activity. Set based on target system being compared to.
        vi. Consider non-durable commits for Masking, Test, QA, UAT: set `COMMIT_WAIT = NOWAIT, COMMIT_LOGGING = BATCH`
    d. Use Oracle Direct NFS (dNFS) for 11.2.0.4+ (**unstable**on older releases):
        i. Recommended documentation:
            - Oracle direct NFS configuration
            - Configuration examples and troubleshooting blog from Helmut Hutzler
            - Sample oranfstab to leverage multiple network paths for Delphix VDB
        ii. Set DNFS_batch_size = 128 (default is 4096). This is a good starting point and sufficient for most workloads.
        iii. Tune TCP stack: set `tcp_adv_win_scale` = 2 due to workaround hard-coded Oracle dNFS TCP buffer size.
        iv. Check Alert Log, V$DNFS_SERVERS, V$DNFS_FILES, V$DNFS_STATS to verify proper working (sample here).
    e. Create AWR snapshots around a reference customer workload, generate an AWR report.
        i. AWR snap before/after workload: `SQL> exec DBMS_WORKLOAD_REPOSITORY.CREATE_SNAPSHOT();`
        ii. AWR report between the snaps: `SQL> @?/rdbms/admin/awrrpte`
    f. Generate ASH report to diagnose bottlenecks while a workload is running.
        - `SQL> @?/rdbms/admin/ashrpt`
    g. Run synthetic benchmark `sc-workload.`
    h. Where `db file scattered read` (multiblock cached read) latency is high consult this Support KB: How to mitigate multi-block read performance on Oracle 10g
    i. Improve distributed query performance by modifying dblinks to use local IPs instead of SCAN IPs.

j. NFS recommended mount options for Oracle RAC/SI: Oracle support note 359515.1

Target Host OS Settings

1. Existing documentation on Target OS practices: Target host configuration options for improved performance
2. HP-UX 11.31+
   - Async NFS direct I/O: HP-UX requires Oracle `disk_asynch_io` turned off for filesystems
3. IBM AIX:
   a. Consult IBM documentation on AIX TCP Tuning
   b. AIX TCP tuning prezo
4. Windows:
   a. Anti-virus programs can impact both performance and operation. Delphix recommends anti-virus scanning exclude folders where Delphix files are maintained, in addition to the normal exclusions put in place for MSSQL operation.
   b. Delphix Connector (aka DX Connector):
      i. Plan 3-5GB for the Delphix Connector installation.
      ii. Windows does not yet have ssh, so Delphix developed the "DX Connector for Windows target host communication.
      iii. The connector must be installed on all Target Windows hosts.
      iv. The connector supports two modes – v1 and v2 both use the same application binaries.
      v. The connector v1 process is used to bootstrap the v2 process on a target. This opens a DSP session back to the Delphix Engine (The same thing is done via SSH on U*nix Targets)
      vi. v2 mode is required to enable SQL hooks
      vii. The connector can always be downloaded from a local Delphix Engine at: http://<delphix_engine>/connector/DelphixConnectorInstaller.exe.
      viii. The connector is backwards compatible, so it is not always necessary to upgrade it during a Delphix upgrade.
   c. iSCSI connections:
      i. Read the following for general awareness of iSCSI limits
      ii. In addition to the hard limits on iSCSI connections, consideration must be given to the RAM, CPU and Network to provide sufficient resources for the load on any Target or Staging host.
      iii. To increase the iSCSI timeout on both Target and Staging hosts.
      iv. In certain circumstances it's possible that iSCSI startup will not complete before the SQL Service attempts to start a database. In such circumstances, it can be helpful to ensure the SQL service depends on the iSCSI service.
         - Example: c:\> sc config "MSSQLServer" depend="Microsoft iSCSI Initiator Service"
      v. **Note** that any changes to iSCSI are system-wide and could potentially impact other applications also leveraging that feature.
   d. Enable receive side scaling (RSS) on each network interface that Delphix will be connecting to.

Receive side scaling (RSS) for windows staging target and targets

Enabling Receive Side Scaling (RSS) on a Windows Target and Staging Target can have a significant improvement in the overall IO throughput to the Delphix Engine and is a best practice. RSS enables network adapters to distribute the kernel-mode network processing load across multiple processor cores in multi-core computers. The distribution of this processing makes it possible to support higher network traffic loads than would be possible if only a single core were to be used.

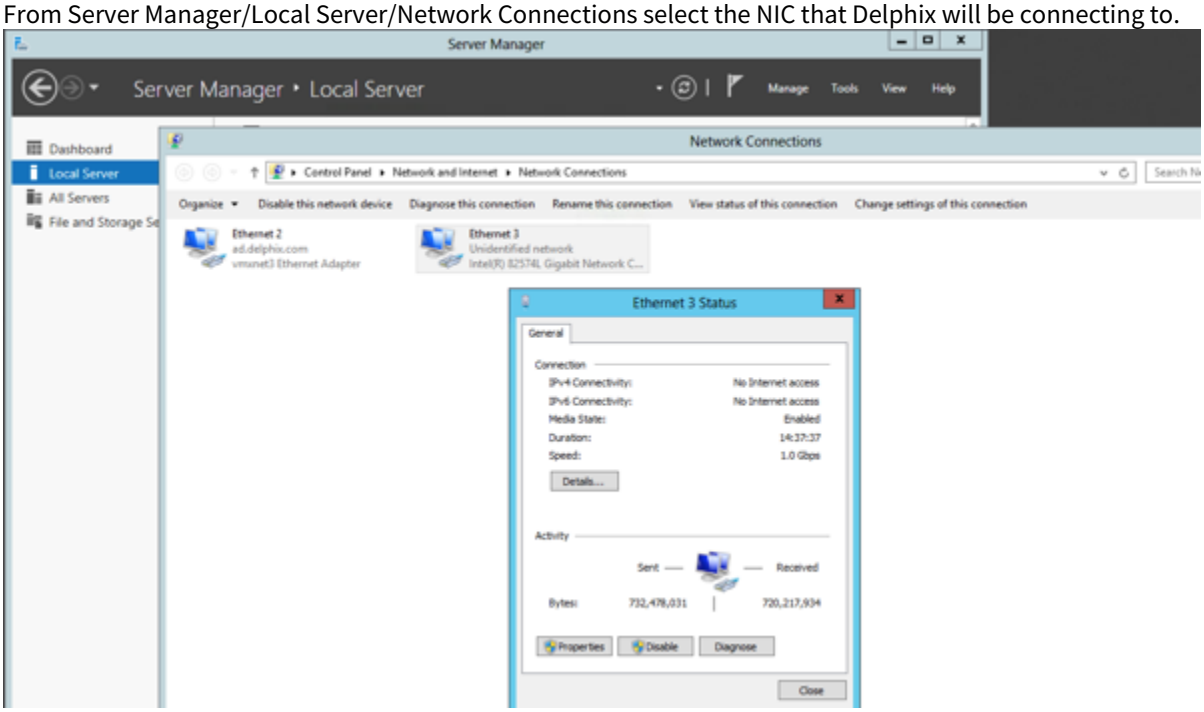
More information on RSS can be found here.

> ⚠ Enabling RSS on the network interface will force the network service to restart and will cause a momentary loss of connectivity on that network interface.

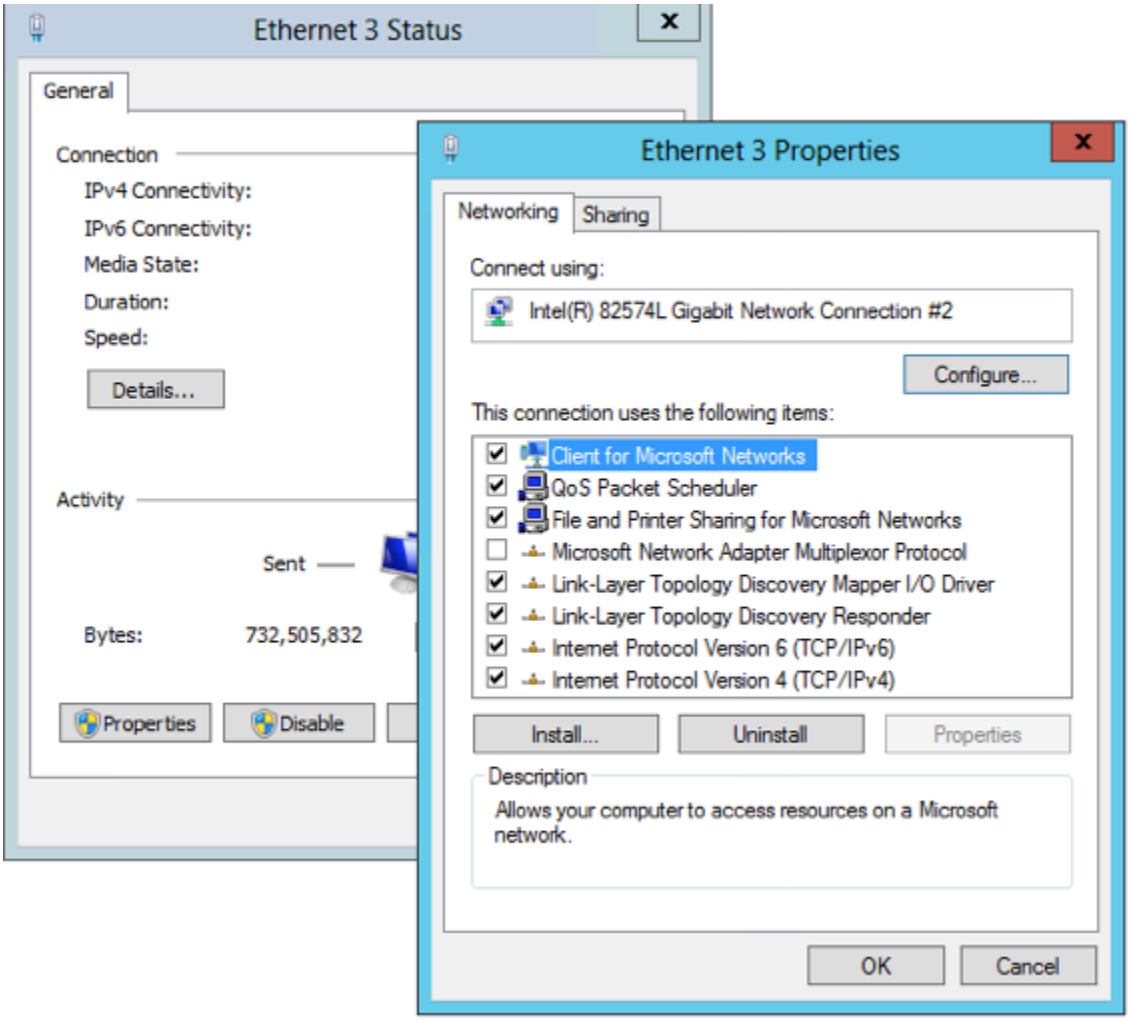
> ⓘ Because hyper-threaded CPUs on the same core processor share the same execution engine, the effect is not the same as having multiple core processors. For this reason, RSS does not use hyper-threaded processors.

Steps to implement RSS on windows

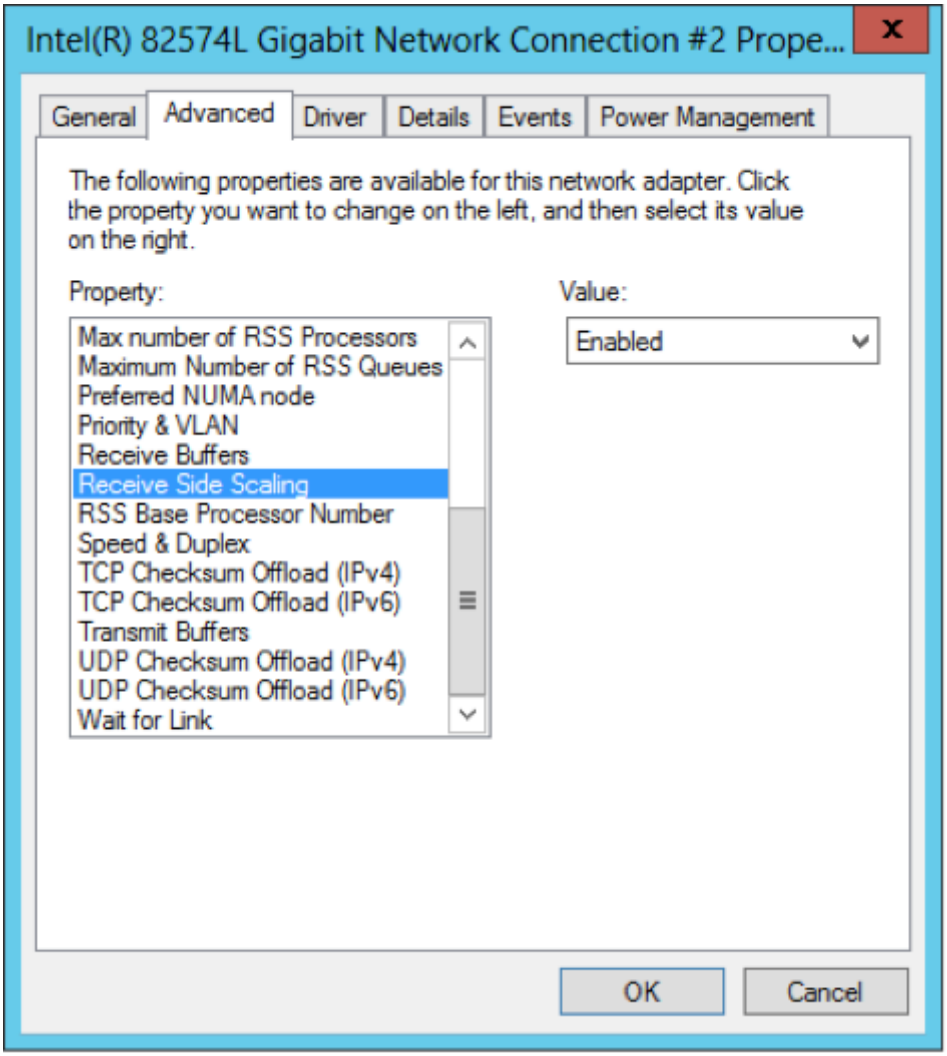
1. From Server Manager/Local Server/Network Connections select the NIC that Delphix will be connecting to.



2. Select Properties and then Configure.



3. From the Property menu on the left, select Receive Side Scaling and select 'OK' to close each of the open windows.


Intel(R) 82574L Gigabit Network Connection #2 Prope...
General
Advanced
Driver
Details
Events
Power Management
The following properties are available for this network adapter. Click the property you want to change on the left, and then select its value on the right.
Property:
Value:
Max number of RSS Processors
Maximum Number of RSS Queues
Preferred NUMA node
Priority & VLAN
Receive Buffers
Receive Side Scaling
RSS Base Processor Number
Speed & Duplex
TCP Checksum Offload (IPv4)
TCP Checksum Offload (IPv6)
Transmit Buffers
UDP Checksum Offload (IPv4)
UDP Checksum Offload (IPv6)
Wait for Link
Enabled
OK
Cancel

Best practices for staging targets

This host is called a "staging target" because it has much in common with other targets, such as the remote storage mount to the Continuous Data Engine. Because it leverages remote storage over the network, the staging target only needs enough disk capacity for the OS, database application, and any relevant logs or tools. A staging target is a requirement on all platforms that Delphix supports except Oracle, but you can also use it for Oracle to replay logs leveraging validated sync.

Memory and CPU

1. 32 GB RAM minimum
2. 4 vCPU minimum

General guidance for staging servers (Multi-platform)

1. Delphix recommends dedicated Staging servers for role/architecture separation. However, any Target server can be used as Staging.
2. In cases where the same server is used as both Staging and Target, we strongly recommend a dedicated instance/install for staging to avoid confusion.
3. Delphix recommends at least one Staging server per Delphix Engine to avoid the possibility of a single point of failure across multiple engines.
4. If a staging server is shared among multiple Delphix Engines, please ensure that a dedicated SQL Server Instance is created for each Delphix Engine.
5. Configuration / performance factors:
    a. Transaction log generation rate.
    b. Number of VDBs.

⚠ Precise guidance on these items has not yet been defined. In general, if there is a heavy log generation rate and few VDBs, the ideal recommendation is to have at least 1 Staging Target per Delphix Engine.

Disk / local storage

1. The only local storage needed is for the OS and application with default databases.
2. Storage for a staging database is provided from the Delphix Engine, which is mounted over the network similar to any Target host (NFS/iSCSI).
3. If the customer has a standard DB server build, their standard storage sizing is probably fine.
4. If a recommendation is still needed, suggest 30GB for OS and application and any tools needed.

Network requirements

1. The Staging Target engages in network data transfers between Staging and the Source backup shared location as well as between Staging and the Delphix Engine.
2. The Staging Target is also a Target server, and as such should have < 1ms latency to the Delphix Engine (and low latency to the Source backup, when possible).
3. If the change rate on the Source database(s) is > 1 Gb/sec, the recommended network bandwidth to support network transfers is 10 Gb/sec.
4. In cases where only 1 Gb/sec network bandwidth is available, segregation of each network is recommended to reduce network contention.
5. Ensure that the virtual NIC is using the standard vmxnet3 adapter and not Intel for VMWare based clients. Logical IO errors have been reported while using Intel instead of vmxnet3 adapter.

Windows and MSSQL specific

1. An MSSQL Server Instance used for Staging should not be clustered.
2. Staging should not be hosted on Windows 2003 - extended support ended July 14, 2015. It is also the first Windows version with iSCSI support and is not ideal.
3. The SQL Server Instances hosted on the Staging Target should have a Maximum Memory set. Also ensure that at all times, at least 10% of total memory is available for OS operations.
4. Only system databases (Master/MSDB/Temp/MSDB) are kept on local storage. All other data is read/written to the Delphix Engine.
5. Windows iSCSI configuration and limits for v2p, target and staging hosts.
6. Ensure that Receive Side Scaling (RSS) is enabled on every network interface that Delphix will be connecting to.

Delphix Engine controls the number of concurrent restore operations that can run on a staging environment by the validated sync workers, which means we throttle the number of restore operations done by validated sync workers running for different staging databases on the staging environment, with five executing at a time and others waiting for their turn as per First Come First Serve scheduling. This is done to improve overall system performance by reducing resource contention, disk I/O, and network traffic. Also, note that this limit is per Delphix Engine which is connecting to the staging environment.

Following are the limiting factors which will come into play when looking at the performance of staging databases on a staging environment when a validated sync worker runs to keep up with the production databases:

- Backup generation frequency: With higher backup frequency, increased restore time will be seen as the pre-provisioning worker will keep ingesting previous backups while new backups are being generated.
- Staging database count: When multiple staging databases are hosted on the same server, the backup ingestion load on the staging host will increase. Additionally, if the frequency of backups is high, there will be a greater number of candidates (pre-provisioning workers) waiting in the queue.
- Number of VDB hosted on the server
- Multiple Delphix Engines connecting to the same staging environment will increase the number of parallel restore operations running on the staging environment and contribute to the performance.

Below are the troubleshooting steps for improving performance:

- Have dedicated Staging servers for role/architecture separation from VDB
- Add CPU/Memory
- Decrease backup frequency
- Introduce dedicated networks

Sharing an example of performance in a given setup for reference purposes.

> **Note**
> The below findings are from a non-production setup

Environment details

- Staging Host: 64 GB Memory, 8 vCPUs, ESXi version: 7.0.3
- Backup File Size: 200MB
- User for linking: Database user

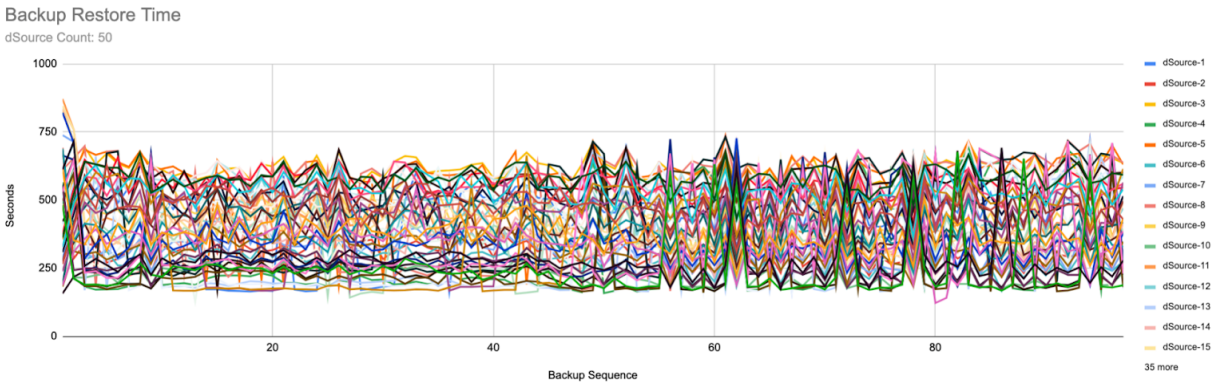
Setup notes:

- No other operations were executed on the Delphix Engine other than pre-provisioning worker running.
- No virtual database existed on the staging host.
- The source servers and the Delphix Engine are all on the same on-prem data center.

- Only one Delphix Engine was connecting to the staging host.

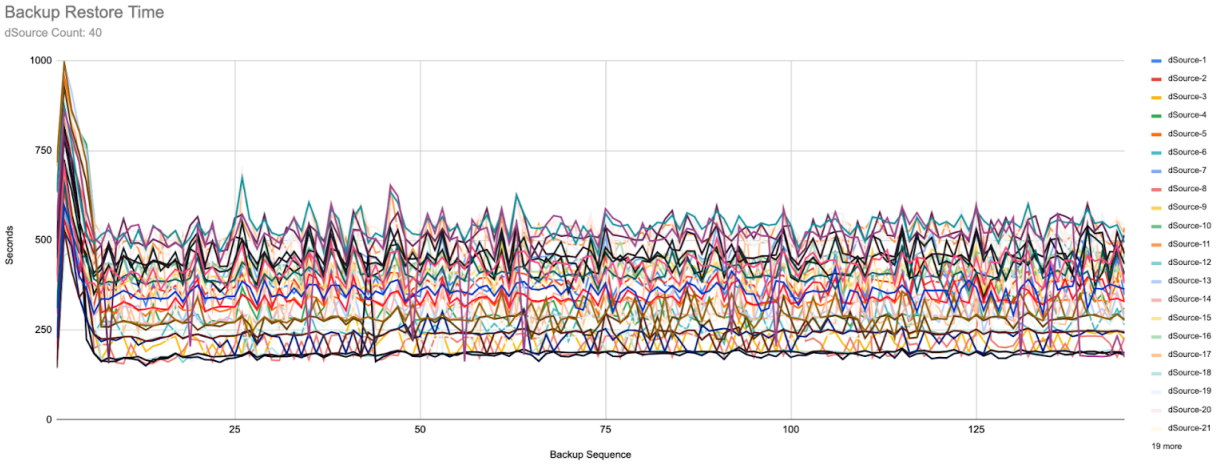
Performance Scenario 1

For a staging host with the above configuration supporting 50 staging databases on a single database instance, and with every dSource having a backup at the 15-minute interval, the time taken to restore these transaction logs stay under 13 minutes(< 750 seconds) on average and hence the staging databases keep up with the backups.



Performance Scenario 2

The same setup could support frequent backups, that is every 10 min, but required the staging databases to be reduced. For example, 40 staging databases on a single database instance could support backups every 10 minutes without causing any lag.



Best practices for storage

1. Configuration:
    a. Virtual Disks (VMDK) with spinning or tiered media must be thick provisioned + lazy zeroed.
        - For storage which is 100% SSD/EFD/Flash-based, continue to thick provision, however, eager zero is not necessary.
    b. VMDKs may be homed on VMFS or RDM storage, however, VMFS is much more common and generally preferred.
    c. Regardless of VMFS/RDM selection, physical LUNs must have uniform characteristics (RAID, spindle count, tier) and should be thin provisioned to save time.
    d. Storage allocated must be identical between each vSCSI controller.
    e. The supported maximum of four virtual SCSI controllers (PVSCSI (default) or LSI Logic Parallel) must be enabled. A mix of different types of SCSI controllers is not supported within the engine.
    f. Virtual Disks must be evenly distributed across the 4 virtual SCSI controllers. For example, 8 virtual disks should be configured as 2 disks per controller: SCSI (0:0), SCSI (0:1), SCSI (1:0), SCSI (1:1), SCSI (2:0), SCSI (2:1), SCSI (3:0), SCSI (3:1).
        - You don't need to account for the OS in the even distribution of disks across controllers, just pick one; the OS doesn't place a substantial load on the controller.
    g. To provision VMDK disks over 16TB in size, the vSphere web client must be used, the win32 client will return an error.
    h. Delphix requires 127GB for the system disk. older versions defaulted to 150GB.
        - Due to our unified OVA for masking, the 127GB requirement also applies to Masking Engines
    i. VMDK for the Delphix Engine OS is often stored on the same VMFS volume as the Delphix VM definition file (aka VMX). In that case, the VMFS volume must have sufficient space to hold the Delphix VMX Configuration, the VDMK for the system disk, a swap/paging area if the memory reservation was not enabled for the Delphix Engine (or to suspend the system), and any related VMware logging.
    j. Set ESX Storage Multipathing IO optimization (KB 2072070).
        i. Set multipathing policy to round-robin.
        ii. Set path switching IO operation limit to 1 (default 1000).
    k. Verify that the queue depth setting on the virtual disks used for VMware is appropriate based on the minimum of the HBA type or the underlying storage's combined queue depth for all hosts attached to the same controller
        - See this VMware KB article for how to check/set the queue depth
2. Testing:
    a. Run Storage Performance Tool on the raw storage **before any engine configuration**. This is a one-time opportunity for each engine upon installation
    b. Required maximum storage latency is < 2ms for writes and < 10ms (95th percentile) for small random reads. Minimum passing grades: 4KB/8KB reads (B-), 1MB reads (A), 1KB/128KB writes (A).
    c. If working with the Delphix Professional Services team, we would expect to run additional baseline performance measurements via composite tools and scripts.
        - e.g. "Sanity Check" (Oracle) or "DiskSpd" (MSSQL)
3. Detail Discussion:
    a. Before beginning any discussion on storage performance, or at the beginning of the installation, collecting the following specs from your storage administrator will assist in understanding
    b. Vendor, Model (For example: EMC VMAX 40k, HP 3PAR StoreServ 7000)
    c. IO latency SLO (For example 5ms 99.999%)
    d. IOPS/GB SLO (For example: 0.68 IOPS/GB for EMC Gold-1)
    e. Cache type and size (For example FAST cache 768GB)
    f. Tier, #Pools; if auto – tiering; relocation schedule (For example: Gold/Silver/Auto/3 pools/etc)
    g. Pool detail: (#) drives, RPM, Type (For example: Pool1: (20) EFD, (30) 15k SAS, Pool 2: (40) 10k SATA)
    h. Connection (For example: 16Gb Fibre Channel, 10Gb iSCSI, 20Gb NFS)
    i. Dedicated or Shared pool (how many applications/servers)

Data backup and recovery solutions

Cloning an existing Delphix Engine by using hypervisor VM cloning features or by copying existing Engine storage to a new virtual machine is currently not supported. You can choose from the following methods for backup or duplication of an Engine.

Data backup and recovery solutions

Learn about the suite of Delphix Backup and Recovery Strategies

Delphix Continuous Vault

Backup and Recovery Strategies for the Delphix Engine

Replication

Selective Data Distribution

Delphix continuous vault

Overview

The Delphix Engine has a base feature set that is compelling as a data protection solution. An enhancement is being introduced that can further prevent snapshot and database loss in the event of a ransomware attack.

Delphix continuous vault for ransomware protection allows organizations to recover their application data access much faster than traditional backup solutions, in case of malicious attacks.

The Continuous Vault solution provides built-in protection against ransomware attacks by frequently refreshing production application data to a virtual database (dSource), from which you can recover applications almost instantly to any valid point in time prior to the encryption attack by leveraging Delphix VDB provisioning capabilities. The refresh interval between the production application and the Delphix dSource can be configured to be up to seconds, for business-critical applications (not applicable to all DB technologies).

There are two variants of Delphix Continuous Vault:

- **Replica Continuous Vault**, which replicates critical business DB data stored on Delphix Engines to a new target Engine called Replica Continuous Vault.
- **Single Engine Continuous Vault**, which protects critical business DB data stored on a Delphix Engine by preventing manual deletion of protected sources or snapshots.

Once securely stored on the Continuous Vault, the DB data can be used to recover business applications with very low RTO and RPO.

> Requires Technical Services Consult
> Delphix **requires** a Technical Services assessment prior to deployment and configuration of Continuous Vault. The process of configuring a Delphix Continuous Vault replication profile is simple; the assessment is required because each application has specific data protection and recovery requirements and we must ensure that Delphix can respond to them appropriately. To schedule an assessment, please contact your Customer Success Manager.

Replica continuous vault

The Replica Continuous Vault feature is available via CLI or via the Continuous Vault UI section of the Replication page. This UI provides functions for creating Continuous Vault replication profiles from scratch and converting existing profiles to be locked.

Advantages

The Replica Continuous Vault variant provides the following advantages:

- Creates a separation of responsibilities between the two Delphix Engines
  - One engine is used for regular Virtualization cases (ingestion, VDBs, SDD)
  - Another engine is used for ransomware protection.
- Creates a physical separation by allowing the admin to isolate and secure the locked Delphix Engine.
  - Only the DSP port has to be open for replication.
  - No ports are needed for JDBC, NFS, or SSH until VDBs need to be created to export data. This also prevents attack vectors related to any of those protocols.
- Can assist with making deployments and security reviews easier to pass since the locked Delphix Engine is isolated and has a single purpose to reduce potential attack vectors.

In the event of a ransomware attack on a primary engine source being compromised or corrupted, you can provision a VDB on the replica in the locked namespace of the replication target – similar to the normal replication namespace. This process can be further outlined in the Provisioning From Replicated Data Sources or VDBs article. If a complete recovery of the primary engine is needed, please contact Delphix Support.

Implementation

This feature adds a property to the replication namespace and specifications called “locked”. Additional dSources, VDBs, groups, and domains can be added to locked replication specs, but data sources cannot be removed after doing so. Failover on the target Delphix Engine is not allowed if the namespace is part of the locked replication spec. The retention policy duration on a locked namespace can be modified as long as the duration is either being increased, or it is being decreased to a minimum of 100 days.

The time configuration on Delphix engines with Continuous Vault enabled cannot be changed. This is to prevent attempts at bypassing retention policies in order to try and delete snapshots on the target. Also, the factory reset operation is forbdiden when at least one locked replication specification or namespace is present.

A fault is now generated on the Continuous Vault target for a locked namespace that has not received a successful replication update in 12 hours. Upon request, Delphix Support can change this value. New replication specs must also have automatic replication enabled and a satisfactory replication schedule.

CLI functions

Create a locked replication profile.

```
[user.hostname]> replication spec
[user.hostname] replication spec> create
[user.hostname] replication spec create *> set name=locked-spec-1
[user.hostname] replication spec create *> set objectSpecification.objects=Untitled/
dbname
[user.hostname] replication spec create *> set targetHost=example.delphix.com
[user.hostname] replication spec create *> set targetPrincipal=admin
[user.hostname] replication spec create *> set targetCredential.password=delphix
[user.hostname] replication spec create *> set lockedProfile=true
[user.hostname] replication spec create *> commit
    `REPLICATION_SPEC-3
[user.hostname] replication spec> select locked-spec-1
[user.hostname] replication spec 'locked-spec-1'> get
    type: ReplicationSpec
    name: locked-spec-1
    automaticReplication: false
    bandwidthLimit: 0
    description: (unset)
    encrypted: false
    lockedProfile: true <------------------------------ LOCKED
    numberOfConnections: 1
    objectSpecification:
        type: ReplicationList
        name: (unset)
        objects: Untitled/dbname
    reference: REPLICATION_SPEC-3
    runtime:
        type: ReplicationSpecRuntime
```

```
    schedule: (unset)
    tag: 5570be25-dbcf-48c3-b2d2-dd2c65eb98b7
    targetCredential:
        type: PasswordCredential
        password: ********
    targetHost: example.delphix.com
    targetPort: 8415
    targetPrincipal: admin
    useSystemSocksSetting: false
[user.hostname] replication spec 'locked-spec-1'> cd ..
[user.hostname] replication spec>
```

Lock an unlocked replication profile.

```
[user.hostname]> replication spec create
[user.hostname] replication spec create *> set name=locked-spec-2
[user.hostname] replication spec create *> set objectSpecification.objects=Untitled/
dbname
[user.hostname] replication spec create *> set targetHost=example.delphix.com
[user.hostname] replication spec create *> set targetPrincipal=admin
[user.hostname] replication spec create *> set targetCredential.password=delphix
[user.hostname] replication spec create *> commit
    `REPLICATION_SPEC-4
[user.hostname]> replication spec select locked-spec-2
[user.hostname] replication spec 'locked-spec-2'> get
    type: ReplicationSpec
    name: locked-spec-2
    automaticReplication: false
    bandwidthLimit: 0
    description: (unset)
    encrypted: false
    lockedProfile: false
    numberOfConnections: 1
    objectSpecification:
        type: ReplicationList
        name: (unset)
        objects: Untitled/dbname
    reference: REPLICATION_SPEC-4
    runtime:
        type: ReplicationSpecRuntime
    schedule: (unset)
    tag: e8608d05-0693-440d-8a2b-8c6cbfe06a62
    targetCredential:
        type: PasswordCredential
        password: ********
    targetHost: example.delphix.com
    targetPort: 8415
    targetPrincipal: admin
    useSystemSocksSetting: false
[user.hostname] replication spec 'locked-spec-2'> update
[user.hostname] replication spec 'locked-spec-2' update *> set lockedProfile=true
[user.hostname] replication spec 'locked-spec-2' update *> commit
```

```
[user.hostname] replication spec 'locked-spec-2'> get lockedProfile
    true
[user.hostname] replication spec 'locked-spec-2'>
```

Verify the locked status of a namespace.

```
[user.hostname]> namespace
[user.hostname] namespace> list
NAME
[user.hostname]-1
[user.hostname]-3
[user.hostname] namespace> select [user.hostname]-3
[user.hostname] namespace '[user.hostname]-3'> get
    type: Namespace
    name: [user.hostname]-3
    description: (unset)
    failedOver: false
    locked: true   <------------------------------ LOCKED
    namespaceType: REPLICATION
    reference: NAMESPACE-4
    secureNamespace: false
    tag: 5570be25-dbcf-48c3-b2d2-dd2c65eb98b7
[user.hostname] namespace '[user.hostname]-3'>
```

Verify that the namespace cannot be deleted or failed over.

```
[user.hostname] namespace '[user.hostname]-1'> delete
[user.hostname] namespace '[user.hostname]-1' delete *> commit
   Error: Namespace "[user.hostname]-1" is locked and cannot be deleted.
  Action: Cannot delete a locked namespace.
[user.hostname] namespace '[user.hostname]-1' delete *> discard
[user.hostname] namespace '[user.hostname]-1'> failover
[user.hostname] namespace '[user.hostname]-1' failover *> commit
   Error: Namespace "[user.hostname]-1" is locked and cannot be failed over.
  Action: Cannot failover a locked namespace.
[user.hostname] namespace '[user.hostname]-1' failover *> discard
```

Verify that the replication profile cannot be deleted or modified. Objects can still be added to the replication profile.

```
[user.hostname]> replication spec
[user.hostname] replication spec> select locked-spec-1
[user.hostname] replication spec 'locked-spec-1'> delete
[user.hostname] replication spec 'locked-spec-1' delete *> commit
   Error: The replication profile is locked and cannot be deleted.
  Action: Select an unlocked profile to delete.
[user.hostname] replication spec 'locked-spec-1' delete *> discard
[user.hostname] replication spec 'locked-spec-1'> update
[user.hostname] replication spec 'locked-spec-1' update *> set automaticReplication=t
rue
[user.hostname] replication spec 'locked-spec-1' update *> commit
   Error: The replication profile is locked and cannot be updated.
```

```
  Action: Select an unlocked profile to update.
[user.hostname] replication spec 'locked-spec-1' update *> discard
[user.hostname] replication spec 'locked-spec-1'> update
[user.hostname] replication spec 'locked-spec-1' update *> set
objectSpecification.objects=Untitled/dbname,Group:/Untitled
[user.hostname] replication spec 'locked-spec-1' update *> commit
[user.hostname] replication spec 'locked-spec-1'> update
[user.hostname] replication spec 'locked-spec-1' update *> set
objectSpecification.objects=Untitled/dbname
[user.hostname] replication spec 'locked-spec-1' update *> commit
   Error: Objects cannot be removed from a locked replication profile.
  Action: Select an unlocked profile to update.
[user.hostname] replication spec 'locked-spec-1' update *> discard
[user.hostname] replication spec 'locked-spec-1'>
```

Create a replica retention policy and apply it to the locked namespace.

```
[user.hostname]> policy
[user.hostname] policy> createAndApply
[user.hostname] policy createAndApply *> set policy.type=ReplicaRetentionPolicy
[user.hostname] policy createAndApply *> set policy.duration=6
[user.hostname] policy createAndApply *> set policy.durationUnit=YEAR
[user.hostname] policy createAndApply *> set target=Namespace:/[user.hostname]-1
[user.hostname] policy createAndApply *> set policy.name="Six Years"
[user.hostname] policy createAndApply *> get
    type: PolicyCreateAndApplyParameters
    policy:
        type: ReplicaRetentionPolicy (*)
        name: Six Years (*)
        customized: false
        duration: 6 (*)
        durationUnit: YEAR (*)
    target: Namespace:/[user.hostname]-1 (*)
[user.hostname] policy createAndApply *> commit
    `POLICY_REPLICA_RETENTION-30
[user.hostname] policy>
```

Verify that the replica retention policy cannot be deleted or modified.

```
[user.hostname] policy> select POLICY_REPLICA_RETENTION-30
[user.hostname] policy 'Six Years'> delete
[user.hostname] policy 'Six Years' delete *> commit
   Error: The replica retention policy "Six Years" could not be removed because the
target namespace "[user.hostname]-1" is locked.
[user.hostname] policy 'Six Years' delete *> discard
[user.hostname] policy 'Six Years'> update
[user.hostname] policy 'Six Years' update *> set duration=4
[user.hostname] policy 'Six Years' update *> commit
   Error: The replica retention policy "Six Years" could not be modified because the
target namespace "[user.hostname]-1" is locked.
[user.hostname] policy 'Six Years' update *> discard
```

```
[user.hostname] policy 'Six Years'> unapply
[user.hostname] policy 'Six Years' unapply *> set target=Namespace:/[user.hostname]-1
[user.hostname] policy 'Six Years' unapply *> commit
   Error: The replica retention policy "Six Years" could not be removed because the
target namespace "[user.hostname]-1" is locked.
[user.hostname] policy 'Six Years' unapply *> discard
[user.hostname] policy 'Six Years'>
```

Single engine continuous vault

This feature is available in versions 6.0.14.0 and above. CLI and UI functions are available for locking dSources.

Advantages

The Single Engine Continuous Vault provides effective protection against ransomware attacks in a standalone Delphix Engine. This option may be preferable for deployments where maintaining two separate engines is not architecturally necessary.

Implementation

This feature adds a “locked” property to sources. Once the locked property is enabled, the source cannot be removed. Locked sources are required to have a SnapSync policy defined that refreshes data at least once daily. Furthermore, an alert is raised if no new snapshot or log data is received in the last 12 hours. Upon request, Delphix Support can change this value.

To protect Continuous Vault application data from accidental deletion or malicious attack, snapshots of locked sources may not be manually deleted. These snapshots are managed by a retention policy that must be configured for locked sources. The retention policy duration can be modified as long as retention satisfies the minimum duration (100 days).

As with the Continuous Vault Replication implementation, the time configuration of a Single Engine Continuous Vault cannot be changed. This is to prevent attempts at bypassing retention policies in order to try and delete snapshots on the Continuous Vault. Also, the factory reset operation is forbidden when at least one locked source is present.

By default, locked sources are not required to have LogSync enabled. However, upon request, Delphix Support can configure an engine-wide setting that prevents LogSync from being disabled on a locked source and, optionally, requires LogSync to be enabled from the very beginning—before locking a source.

CLI functions

Locking a source.

```
sedv> source
sedv source> select src10
sedv source 'src10'> lock
sedv source 'src10' lock *> commit
sedv source 'src10'>
```

Verifying the locked status of a source.

```
sedv source 'src10'> ls
```

```
Properties
    type: OracleLinkedSource
    name: src10
    container: src10
    externalFilePath: (unset)
    linked: true
    locked: true     <--------------- LOCKED
    logCollectionEnabled: false
    operations:
...
```

Verify that a locked source cannot be disabled.

```
sedv source 'src10'> disable
sedv source 'src10' disable *> commit
   Error: The source "src10" cannot be disabled because it is locked.
  Action: Contact Delphix support.
sedv source 'src10' disable *> discard
```

Verify that source locking requires a SnapSync policy that refreshes data at least once daily.

```
# An insufficiently-frequent SnapSync policy: runs at 03:00 on Sundays
sedv policy 'snapsync_weekly'> ls
Properties
    type: SyncPolicy
    name: snapsync_weekly
    customized: false
    default: false
    effectiveType: DIRECT_APPLIED
    reference: POLICY_SYNC-7
    scheduleList:
        0:
            type: Schedule
            cronString: 0 0 3 ? * 1
            cutoffTime: 14400sec
    timezone:
        type: TimeZone
        id: America/New_York
        offset: 240
        offsetString: UTC -04:00

Operations
delete
update
apply
unapply


sedv> source
sedv source> select src10
sedv source 'src10'> lock
sedv source 'src10' lock *> commit
```

```
   Error: Insufficient or unrecognized day coverage in schedule that affects locked
sources.
  Action: Cover either all days of the month or all days of the week when locked
sources are affected. Check the documentation for examples.
```

Verify that source locking requires a retention policy that retains 100 days of snapshots.

```
# A one-month retention policy
sedv policy 'retention_one_month'> ls
Properties
    type: RetentionPolicy
    name: retention_one_month
    customized: false
    dataDuration: 1
    dataUnit: MONTH
    dayOfMonth: 1
    dayOfWeek: MONDAY
    dayOfYear: Jan 1
    default: false
    effectiveType: DIRECT_APPLIED
    logDuration: 1
    logUnit: MONTH
    numOfDaily: 0
    numOfMonthly: 0
    numOfWeekly: 0
    numOfYearly: 0
    reference: POLICY_RETENTION-8

Operations
delete
update
apply
unapply

sedv> source
sedv source> select src10
sedv source 'src10'> lock
sedv source 'src10' lock *> commit
   Error: The retention policy is less than the minimum "100" days required when
applied to locked sources.
  Action: Set retention parameters to preserve at least "100" days of data, and try
 again.
```

Verify that a snapshot from a locked source cannot be manually deleted.

```
sedv snapshot ''@2022-05-05T22:41:24.045Z''> delete
sedv snapshot ''@2022-05-05T22:41:24.045Z'' delete *> commit
   Error: The selected snapshot cannot be deleted because the source associated with
its container "Untitled/src10" is locked.
  Action: Wait for the snapshot to be automatically deleted based on its retention
policy, or Contact Delphix support.
```

Continuous vault alert system

In addition to the data-protection rules described in the previous sections, Continuous Vaults have a special alert system that notifies administrators about all events that can affect the ability to ingest and replicate locked data or even the ability to send such alerts.

There are two categories of Continuous Vault alerts: domain alerts and system alerts which are emailed to Engine Administrators and System administrators, respectively. To receive alerts, an administrator must have an email address configured and SMTP must be enabled.

All Continuous Vault alerts are sent also via SNMP, Syslog and Splunk/Fluentd when those services are enabled and their configured severity levels allow for each alert level.

Continuous Vault domain alerts are generated for the following events:

1. Locking a source, replication specification or namespace (informational level).
2. All actions on locked replication specifications and namespaces, such as changes in replication schedules, and all actions on locked sources and related objects that can affect them, such as changes in environment settings. All action alerts are audit-level alerts.
3. Deleting a Engine Administrator or changing their email address. These are warning-level alerts emailed to the addresses before the change takes effect.

Continuous Vault system alerts are generated for the following events:

1. Modifying, disabling or deleting services used for delivering alerts: SMTP, SNMP, Syslog and Splunk/Fluentd. These are warning-level alerts sent using the service configurations before the change takes effect.
2. Creating or enabling such a service (informational level).
3. Deleting a System administrator or changing their email address. These are warning-level alerts emailed to the addresses before the change takes effect.

By default, system administrators are allowed to change these service configurations. Continuous Vaults only notify when those changes happen. However, upon request, Delphix Support can enable locking these services such that, once an administrator enables a service, that service cannot be changed (except for subsequent changes requested to Delphix Support).

Backup and recovery strategies for the Delphix engine

As a software virtual appliance, Delphix leverages features of the storage, hypervisor, and appliance infrastructure to provide for recovery in the event of failure. These topics walk through the process of evaluating requirements and defining a solution. This process depends on the requirements and features of the environment in which the Delphix Engine is deployed.

Backup and recovery requirements

This topic describes determining requirements for infrastructure failure modes and recovery.

Before devising a strategy, you must first have a set of requirements to evaluate possible solutions. What failures are you trying to protect against, and what are your recovery goals in the event of failure?

Failure points

Before devising a strategy, you must first have a set of requirements by which the resulting solution can be evaluated. What failures are you trying to protect against, and what are your recovery goals in the event of failure?

Physical server failure

The Delphix Engine runs within the VMware ESX hypervisor, which itself is running on a physical machine. Failure of that physical machine will affect the Delphix Engine, as well as any other virtual machines running on that server. The failure is isolated to that particular server and is not the result of a larger, site-wide failure.

- **Recommendation**: ESX Clustering

Storage failure

The Delphix Engine uses LUNs from a storage array provided through the VMware hypervisor. The storage array may have redundant disks and/or controllers to protect against single points of failure within the array. However, the Delphix Engine can still be affected by a failure of the entire array, the SAN path between the Delphix Engine and the array, or by a failure of the LUNs in the array that are assigned to the Delphix Engine.

- **Recommendation**: Replication

Site failure

When an entire site or datacenter goes down, all servers, storage, and infrastructure are lost. This will affect not only the Delphix Engine but any production databases and target servers in the datacenter.

- **Recommendation**: Replication

Administrative error

If an administrator mistakenly deletes a VDB or takes some other irreversible action, there is no method of recovery built into the Delphix Engine.

- **Recommendation**: Snapshots

Recovery objectives

Once infrastructure fails, some amount of work is required to restore the Delphix Engine to an operational state. Clients won't have access to the Delphix Engine during this time, and the point to which the system is recovered is dependent on the mechanism being used. These qualitative aspects of recovery can be captured by the following metrics. As these metrics are often directly associated with cost, it is important to think not just about the desired metrics, but also the minimum viable goals.

Recovery point objective (RPO)

The RPO is the acceptable amount of data that can be lost in the event of a failure. For example, if backups are taken once a day, then at most 24 hours of data will be lost if the system fails immediately before a regularly scheduled backup.

Recovery time objective (RTO)

The RTO is the time required to restore the system to an operational state after a failure. For example, recovery may require restoring data from a backup, followed by some number of manual steps to recreate the configuration in the new system. RTO is equivalent to the downtime experienced by clients.

Recovery time granularity (RTG)

The granularity of the recovery time is the specificity by which you can select a particular point in time from the past to restore the system. For example, VM snapshots taken every hour provide no way to restore to a point in time between those snapshots.

Backup solution implementation

This topic describes the tradeoffs involved with backup and recovery solutions.

With the exception of clustering, solutions can be implemented using features at both the storage and hypervisor layer. Choosing the right technology requires understanding both your requirements and what infrastructure is in use in your environment. The following sections outline some basic choices and the tradeoffs involved.

Clustering

VMware vSphere's high availability provides the ability to have a VM configuration shared between multiple physical ESX servers. Once the storage has been configured on all physical servers, any server can run the Delphix Engine VM. This allows ESX clusters to survive physical server failure. In the event of failure, the VM is started on a different server and appears to clients as an unexpected reboot with non-zero but minimal downtime. Depending on the length of the outage, this may cause a short pause in I/O and database activity, but longer outages can trigger timeouts at the protocol and database layers that result in I/O and query errors. Such long outages are unlikely to occur in a properly configured environment.

Automatic detection of a failure in an HA environment does not work in all circumstances, and there are cases where the host, storage, or network can hang such that clients are deprived of access, but the systems continue to appear functional. In these cases, a manual failover of the systems may be required.

When configuring a cluster, it is important to provide standby infrastructure with equivalent resources and performance characteristics. Asymmetric performance capabilities can lead to poor performance in the event of a failover. In the worst case of an over-provisioned server, this can cause widespread workload failure and the inability to meet performance SLAs.

Snapshots

VMware provides storage-agnostic snapshots that are managed through the VMware Snapshot Manager. The use of VMware snapshots can, however, cause debilitating performance problems for write-heavy workloads due to the need to manage snapshot redo-log metadata. In order to provide an alternative snapshot implementation, while retaining the existing management infrastructure, VMware has created an API to allow storage vendors to supply their own snapshot implementation. This is only supported in ESX 5.1. Furthermore, the array must support the vStorage APIs. Consult the VMware documentation for supported storage solutions and the performance and management implications.

Storage-based snapshots, by virtue of being implemented natively in the storage array, typically do not suffer from such performance problems and are preferred over VMware snapshots when available. When managing storage-based snapshots, it is critical that all LUNs backing a single VM be part of the same consistency group. Consistency groups provide write order consistency across multiple LUNs and allow snapshots to be taken at the same point in time across the LUNs. This must include all VM configuration, system VMDKs, and VMDKs that hold the dSources and VDBs. Each storage vendor presents consistency groups in a different fashion; consult your storage vendor documentation for information on how to configure and manage snapshots across multiple LUNs.

In the event of a snapshot recovery becoming required, ensure that the Delphix Engine VM is powered off for the duration of the snapshot recovery. Failure to do so can lead to filesystem corruption as you're changing blocks underneath a running system.

Replication

Site Recovery Manager (SRM) is a VMware product that provides replication and failover of virtual machines within a vSphere environment. It is primarily an orchestration framework, with the actual data replication performed by a native VMware implementation, or by the storage array through a storage replication adapter (SRA). A list of

supported SRAs can be found in the VMware documentation. There is some performance overhead in the native solution, but not of the same magnitude as the VMware snapshot impact. SRAs provide better performance but require that the same storage vendor be used as both source and target, and require resynchronization when migrating between storage vendors.

Storage-based replication can also be used in the absence of SRM, though this will require manual coordination when re-configuring and starting up VMs after failover. The VM configuration, as well as the storage configuration within ESX, will have to be recreated using the replicated storage.

The Delphix Engine also provides native replication within Delphix. This has the following benefits:

- The target system is online and active
- VDBs can be provisioned on the target from replicated objects
- A subset of objects can be replicated
- On failover, the objects are started in a disabled state. This allows configuration to be adjusted to reflect the target environment prior to triggering policy-driven actions.
- Multiple sources can be replicated to a single target

Note that the Delphix Engine currently only replicates data objects (dSources and VDBs) and environments (source and target services). It does not replicate system configuration, such as users and policies. This provides more flexibility when mapping between disparate environments, but requires additional work when instantiating an identical copy of a system after failover.

Backup

There is a large ecosystem of storage and VM-based backup tools, each with its own particular advantages and limitations. VMware provides Data Protector, but there are size limitations (linked to a maximum of 2TB of deduped data) that make it impractical for most Delphix Engine deployments. Most third-party backup products, such as Symantec NetBackup, EMC Networker, and IBM Tivoli Storage Manager, have solutions designed specifically for the backup of virtual machines. Because the Delphix Engine is packaged as an appliance, it is not possible to install third-party backup agents. However, any existing solution that can back up virtual machines without the need for an agent on the system should be applicable to Delphix as well. Check with your preferred backup vendor to understand what capabilities exist.

Some storage vendors also provide a native backup of LUNs. Backup at the storage layer reduces overhead by avoiding data movement across the network but loses some flexibility by not operating within the VMware infrastructure. For example, recreating the VM storage configuration from restored LUNs is a manual process when using storage-based recovery.

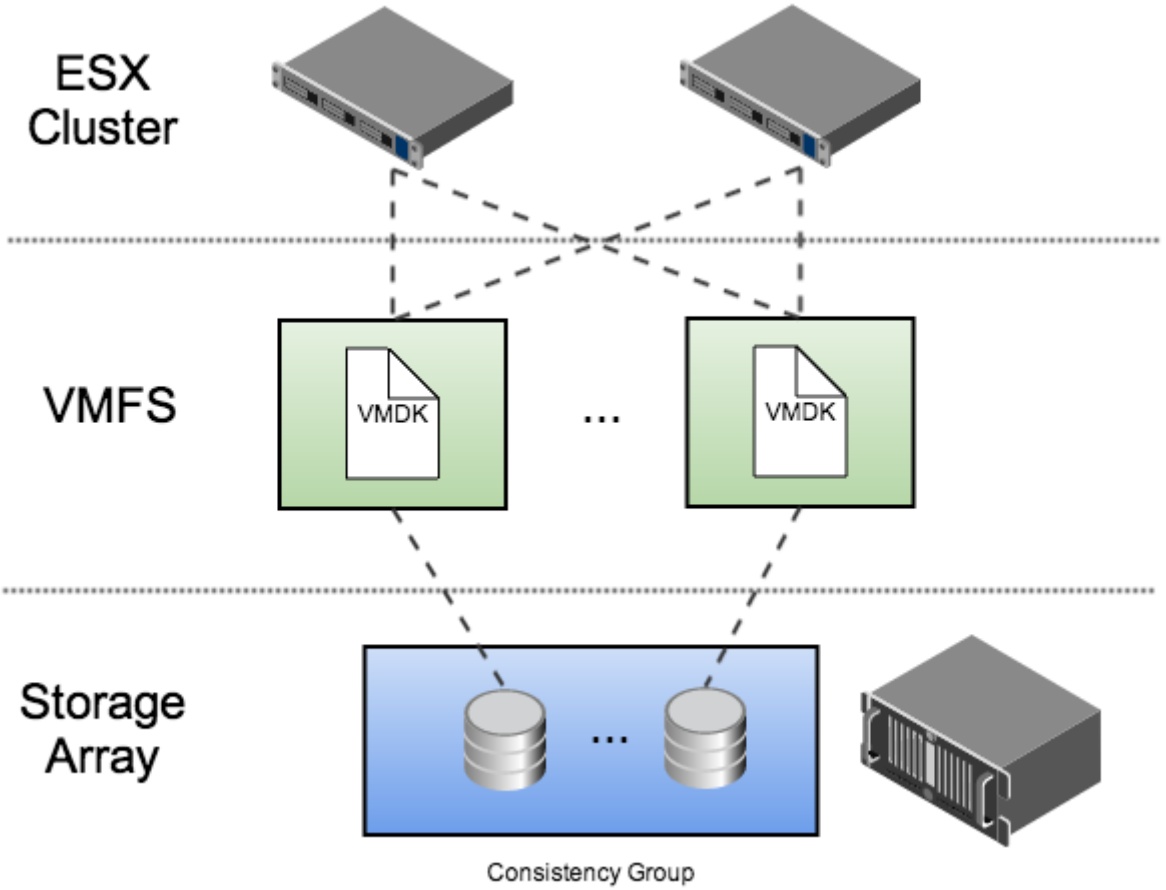
Deployment architecture

This topic describes components of the Delphix deployment architecture.

Delphix operates in a virtual environment with several core systems working in concert, each with its own set of capabilities. Understanding this architecture is critical in evaluating how solutions can be applied across the components, and the tradeoffs involved.

Architectural components

This diagram illustrates Delphix's recommended best practices for deploying the Delphix Engine in a VMware environment:



This architecture is designed to isolate I/O traffic to individual LUNs while using the most commonly deployed VMware components. In this example, each VMDK file is placed in a separate VMFS volume. Each volume is exported to every node in the ESX cluster, allowing the Delphix Engine to run on any physical host in the cluster.

Fault Recovery Features

Across the recommended deployment architecture there are three key components in play: Delphix Engine, VMware, and storage. Each of these provides different failure handling capabilities, which can be roughly grouped into the following areas.

Server Clustering

Clustering provides a standby server that can take over in the event of failure. A given clustering solution may or may not provide high availability guarantees, though all provide failover capabilities, provided that an identical passive system is available.

Snapshots

Snapshots preserve a point-in-time copy of data that can be used later for rollback or to create writable copies. Creating a snapshot is typically low cost in terms of space and time. Because they use the storage allocated to the array, snapshots restore quickly, but they do not protect against failures of the array.

Replication

Data replication works by sending a series of updates from one system to another in order to recreate the same data remotely. This stream can be synchronous, but due to performance considerations is typically asynchronous, where some data loss is acceptable. Replication has many of the same benefits of backup, in that the data is transferred to a different fault domain, but has superior recovery time given that the data is maintained within an online system. The main drawback of replication is that the data is always current - any logical data error in the primary system is also propagated to the remote target. The impact of such a failure is less when replication is combined with snapshots, as is often the case with continuous data protection (CDP) solutions.

Backup

Like snapshots, backup technologies preserve a point-in-time copy of a storage dataset, but then move that copy to offline storage. Depending on the system, both full and incremental backups may be supported, and the backup images may or may not be consistent. Backup has the advantage that the data itself is stored outside the original fault domain, but comes at high cost in terms of complexity, additional infrastructure, and recovery time.

Mapping requirements to solutions

This topic describes how to map from backup and recovery requirements to solutions.

With requirements and detailed knowledge of the deployment architecture, we can map to solutions tailored for the features provided by the underlying infrastructure.

Feature capabilities

Based on these failure points and recovery features, you can use the following table to map requirements to architectural components: VMware (V), Delphix (D), or storage (S). This can drive implementation based on infrastructure capabilities and recovery objectives.

	Fault Recovery Features			
Failure Point	**Clustering**	**Snapshots**	**Replication**	**Backup**
Server Failure	V	-	V S D	-
Storage Failure	-	-	V S D	V S
Site Failure	-	-	V S D	V S
Administrative Error	-	V S	-	V S

Recovery Point Objective (RPO)

Feature	**Time**	**Description**
Clustering	Zero	All changes committed to disk are automatically propagated to the new server. Any pending changes in memory are lost.
Replication	Near zero	Most solutions offer scheduled replication, but many can offer continuous replication with near-zero data loss.
Snapshots	Snapshot period (for example, one hour)	Given their relatively low cost, snapshots tend to be taken at a higher frequency than a traditional backup schedule.
Backup	Backup period (for example, one day)	Backup policies can be configured in a variety of ways, but even with incremental backups, most deployments operate no more frequently than once a day because of the cost of full backups, and the impact of incremental backups on recovery time.

Recovery time objective (RTO)

Feature	Time	Description
Clustering	Near zero	VM clustering with the Delphix Engine provides near-zero downtime in the event of failure, but clients may be briefly paused or interrupted.
Replication	15 minutes	The target side environment is kept in hot standby mode, so it is relatively quick to switch over to the target environment. Depending on the scope of the failure, however, some configuration information may need to be changed on the target side prior to enabling operation.
Snapshots	15 minutes	The Delphix Engine can be rolled back to a previous state. Changes made to systems external to the Delphix Engine (for example, deleting a VDB) can cause inconsistencies after rollback.
Backup	Hours or days	Restoring a full backup can be very time-consuming. In addition to having to read, transfer, and write all of the data, the same process will need to be run for each incremental backup to reach the objective point.

Recovery time granularity

Feature	Granularity	Description
Clustering	None	Only the current system state can be recovered.
Replication	None	Only the nearest replicated state can be recovered unless combined with snapshots.
Snapshots	Snapshot period (for example, one hour)	Determined by the snapshot schedule.
Backup	Backup period (for example, one day)	Determined by the backup schedule.

Delphix replication

This section covers the following topics:

Replication overview

Delphix allows data objects to be replicated between Delphix Engines. Prior to version 5.3.3, these Delphix Engines had to be running identical Delphix versions, but could otherwise vary in terms of other configurations. With version 5.3.3 and above, engines can now be on different versions, which is further detailed in the Forward Compatible Replication (FCR) section. In the event of a failure that destroys the source engine, you can bring up the target engine in a state that matches the source. In addition, you can provision VDBs from replicated objects, allowing for the geographical distribution of data and remote provisioning.

You can run replication ad hoc, but it is typically run according to a predefined schedule. After the initial update, each subsequent update sends only the changes incurred since the previous update. Replication does not provide synchronous semantics, which would otherwise guarantee that all data is preserved on the target engine. When there is a failover to a replication target, some data is lost, equivalent to the last time a replication update was sent.

Replication is generally not suited for high-availability configurations where rapid failover (and failback) is a requirement. Failing over a replication target requires a non-trivial amount of time and is a one-way operation; failback requires replicating all data back to the original source. For cases where high availability is necessary, it is best to leverage features of the underlying hypervisor or storage platform. For more information on how to evaluate the use of Delphix Engine replication for your data recovery requirements, see the topics under Backup and Recovery Strategies for the Delphix Engine.

Forward compatible replication (FCR)

Delphix Virtualization supports the ability to replicate to a Delphix Target Engine running on a higher version. To do so, FCR has a few requirements and restrictions to consider:

- FCR is supported for replication jobs starting from a **source engine** running 5.3.0.0 and beyond.
- The **target engine** must be running 5.3.3.0 and beyond.
- As of 6.0.10.0, the FCR replication operation can not go beyond engine versions released more than 12 months apart.
- FCR supports replicating between major versions as long as those specific versions are no more than a year apart.

Examples of supported and not supported FCR configurations:

Supported

- 6.0.3.0 to 6.0.3.0 (same version)
- 6.0.3.1 to 6.0.3.1 (same version)
- 6.0.3.0 to 6.0.3.1 (higher patch version)
- 6.0.3.1 to 6.0.5.0
- 6.0.3.0 to 6.0.9.0
- 6.0.4.0 to 6.0.10.0

Not supported

- 5.2.5.0 to 6.4.0.0 (source version not compatible with FCR)
- 5.3.0.0 to 5.3.2.0 (target version not compatible with FCR)
- 6.0.9.0 to 6.0.8.0 (source version higher than the target)
- 6.0.0.0 to 6.0.10.0 (there is more than a year between release dates for those versions)

Exceptions

Some of the newer versions of 5.3.x are not compatible with the early versions of 6.0.x.0 (6.0.0.0 and 6.0.1.0).

- 5.3.7.0 and 5.3.7.1 are not compatible with 6.0.0.0

- 5.3.8.0 and 5.3.8.1 are not compatible with 6.0.0.0
- 5.3.9.0 is not compatible with 6.0.0.0
- 5.3.9.0 is not compatible with 6.0.1.0 and 6.0.1.1

Replication matrix

The following table lists all the Delphix Engine versions that a user can replicate to the required or the latest version.

When replicating from X.Y.X.* to X.Y.Z.* the target version has to be greater than or equal to the source version. We are using the * convention to refer to patch releases and reduce the table size.

From Source Version	**To Target Version**
Replication Supported	
5.3.0.*	5.3.0.* 5.3.3.* - 6.0.9.*
5.3.2.*	5.3.2.* - 6.0.9.*
5.3.3.*	5.3.3.* - 6.0.9.*
5.3.4.*	5.3.4.* - 6.0.9.*
5.3.5.*	5.3.5.* - 6.0.9.*
5.3.6.*	5.3.6.* - 6.0.9.*
5.3.7.*	5.3.7.* - 5.3.9.* 6.0.1.* - 6.0.9.*
5.3.8.*	5.3.8.* - 5.3.9.* 6.0.1.* - 6.0.9.*
5.3.9.*	5.3.9.* 6.0.2.* - 6.0.9.*
6.0.0.*	6.0.0.* - 6.0.7.*
6.0.1.*	6.0.1.* - 6.0.7.*
6.0.2.*	6.0.2.* - 6.0.7.*

From Source Version	To Target Version
6.0.3.*	6.0.3.* - 6.0.7.*
6.0.4.*	6.0.4.* - 6.0.7.*
6.0.5.*	6.0.5.* - 6.0.11.*
6.0.6.*	6.0.6.* - 6.0.12.*
6.0.7.*	6.0.7.* - 6.0.13.*
6.0.8.*	6.0.8.* - 6.0.14.*
6.0.9.*	6.0.9.* - 6.0.15.*
6.0.10.*	6.0.10.* - 6.0.16.*
6.0.11.*	6.0.11.* - 6.0.17.*
6.0.12.*	6.0.12.* - 7.0.0.*
6.0.13.*	6.0.13.* - 9.0.0.*
6.0.14.*	6.0.14.* - 11.0.0.*
6.0.15.*	6.0.15.* - 13.0.0.*
6.0.16.*	6.0.16.* -14.0.0.*
6.0.17.*	6.0.17.* -14.0.0.*
7.0.0.*	7.0.0.* - 14.0.0.*
8.0.0.*	8.0.0.* - 14.0.0.*
9.0.0.*	9.0.0.* - 14.0.0.*
10.0.0.*	10.0.0.* - 14.0.0.*

From Source Version	To Target Version
11.0.0.*	11.0.0.* - 14.0.0.*
12.0.0.*	12.0.0.* - 14.0.0.*
13.0.0.*	13.0.0.* - 14.0.0.*
14.0.0.*	14.0.0.*

Replication features

As virtual appliances, it is possible to backup, restore, replicate, and migrate data objects between Delphix Engines using features of VMWare and the underlying storage infrastructure. Data objects include groups, dSources, VDBs, Self-Service (Jet Stream) data templates and data containers, and associated dependencies. In addition to the replication capabilities provided by this infrastructure, native Delphix Engine replication provides further capabilities, such as the ability to replicate a subset of objects, replicate multiple sources to a single target, and provision VDBs from replicated dSources and VDBs without affecting ongoing updates. The topics under Backup and Recovery Strategies for the Delphix Engine provide more information on how to evaluate features of the Delphix Engine in relation to your backup and recovery requirements.

Replication is configured on the source Delphix Engine and copies a subset of dSources and VDBs to a target Delphix Engine. It then sends incremental updates manually or according to a schedule. For more information on configuring replication, see Configuring Replication.

You can use replicated dSources and VDBs to provision new VDBs on the target side. You can refresh these VDBs to data sent as part of an incremental replication update, as long as you do not destroy the parent object on the replication source. For more information, see Provisioning from Replicated Data Sources or VDBs.

During replication, replicated dSources and VDBs are maintained in an alternate replica and are not active on the target side. In the event of a disaster, a failover operation can break the replication relationship. For more information on how to activate replicated objects, see Replicas and Failover.

Replication details

When you select objects for replication, the engine will automatically include any dependencies, including parent objects, such as groups, and data dependencies such as VDB sources. This means that replicating a VDB will automatically include its group, the parent dSource, and the group of the dSource, as well as any environments associated with those databases. When replicating an entire engine, all environments will be included. When replicating a database or group, only those environments with the replicated databases are included.

Only database objects and their dependencies, as well as certain non-database objects, are copied as part of a backup or replication operation, including:

- dSources
- VDBs
- Groups
- Self-service (Jet Stream) Data Templates and Data Containers
- Environments
- Environment configuration (users, database instances, and installations)
- Delphix users, roles, permissions, and authorizations
- Policies

- Database configuration templates

The following objects are NOT copied as part of a backup or replication operation:

- Events and faults
- Job history
- System services settings, such as SMTP
- Hook templates
- Alert profiles

After failover, you must recreate these settings on the target.

> On-Premises Replication To Azure/OCI/GCP/Hyper-V
> Replicating from on-premises engines with an underlying storage block size of 512B will experience disk usage inflation when replicating to target engines with different underlying block sizes. Azure, GCP, Hyper-V, and OCI are known to have 4K block sizes and therefore will require extra disk capacity when receiving replication from an on-premises engine. This behavior is due to the fact that the underlying storage block size is different (512B vs. 4K) between the two Delphix Engines (one on Prem, one on Azure/OCI/GCP/Hyper-V), resulting in a lower compression rate on the replication target. It is expected that 1.5-1.6x the amount of space is taken from objects on-premises in these cases.

Resumable replication

Resumable replication enhances the current replication feature by allowing you to restart large, time-consuming initial replication or incremental updates from an intermediate point. A single replication instance can fail for a number of environmental and internal reasons. Previously, when you restarted a failed replication instance, replication required a full resend of all data transmitted prior to the failure. With resumable replication, no data is retransmitted. Replication is resumable across machine reboot, stack restart, and network partitions. The resumable replication feature is fully automated and does not require or allow any user intervention.

For example, suppose a replication profile has already been configured from a source to a target. A large, full send begins between the two that is expected to take weeks to complete. Halfway through, a power outage at the data center that houses the source causes the source machine to go down and only come back up after a few hours. On startup, the source will detect a replication was ongoing, automatically re-contact the target, and resume the replication where it left off. In the user interface (UI) on the source, the same replication sends job will appear as active and continue to update its progress. However, in the UI of the target, a new replication receives job will appear but will track its progress as a percentage of the entire replication.

In 4.1 and earlier releases, the replication component would always clean up after failed jobs to ensure that the Delphix Engine was kept in a consistent state and that no storage was wasted on unused data. With the addition of reusability, the target and source can choose to retain a partial replication state following a failure to allow future replications to complete from that intermediate point. In the current release, the target and source will only choose to retain partial replication state following failures that leave them out of network contact with each other – for example, source restart, target restart, or network partition. Once network contact is re-established, the ongoing replication will be automatically detected and resumed.

Replication will not resume after failures that leave the source and target connected. For example, if a storage failure on the target, such as out-of-space errors, causes a replication to fail, the source and target remain connected. As a result, the Engine will discard state data associated with the failed Replication operation. Nonetheless, resumable replication would begin during a source reboot, a target reboot, and a network partition.

Replicating Delphix self-service templates

Templates can now be replicated and accessed on the target engine via Delphix Self-Service (Jet Stream). Replicated templates can be replicated into the target space with or without their containers. On the new target engine, the newly created replicated template can be used to create new containers that are assigned to users. You cannot change the replicated template's name or the names of the containers from which it was replicated.

Any containers that were replicated over with the template cannot be used to start, stop, etc until they are disconnected to their parent containers in the source engine during the failover operation.

Replication of non-data objects

As of 6.0.5.0, the replication of non-data objects is supported.

Non-data objects refer to:

- Delphix users, roles, permissions, and authorizations
- Policies
- Database configuration templates

These objects will not be presented as selectable objects when creating a replication spec, but will instead be passively included by association, the same way environments are. Replication of non-data objects will follow these rules:

- If the entire engine is replicated, all non-data objects will be included
- If specific data objects are replicated, all associated non-data objects will be included

For example, when a container is replicated, then policies that apply to that container, as well as users who have authorizations on that container, will all be included for replication.

Replicated users will be shown on the Received Replicas page on the Delphix Engine UI.

Also as of 6.0.5.0, the automatic conflict resolution option is chosen by default; it is the recommended way of handling the failover of non-data objects, as non-data objects are expected to cause collisions. Automatic conflict resolution will resolve non-data object conflicts by the following rules:

- Users will be consolidated if both username and email match. Otherwise, the replicated user will be renamed.
- Roles, permissions, and authorizations will be consolidated
- Policies and database configuration templates will be renamed, except for "None" type policies, which will be consolidated. Following failover, replicated policies will continue to apply to the same replicated data objects.

Replication concepts

Delphix replication allows you to copy objects from one Engine (referred to as a source Engine) to another Engine (referred to as a target Engine).

Replication recreates objects from the source Engine onto the target Engine in a **replica**, also known as a **namespace**, that preserves object relationships and naming on the target Engine without interfering with its active objects. Objects within a replica are read-only and disabled until the replica is failed over, at which point they can be activated. VDBs and dSources within a replica can be used as the source for provisioning new VDBs. Below are key concepts for replication that will be explained in detail:

- **Received Replicas or Namespaces**: Once replication is complete, the target Engine will create a received replica, also known as a namespace. This is a copy of objects that are related via a grouping.
- **Failover and Conflict Resolution**: Certain objects may require changes to resolve conflicts prior to completing a replication failover. Names of replicated objects should be unique for replication to complete successfully
- **Enabling Databases and Environments**: Once replication takes place, you must enable the databases and environments on the target engine. This step ensures the configuration of these objects is correct and will ensure that replication has correctly copied the necessary object relationships from the source engine.

Received replicas or namespaces

A replica contains a set of replicated objects, such as dSources and VDBs. These objects are read-only and disabled while replication is ongoing. You may view replicated objects both in the Delphix Engine UI as well as the CLI. To view received replicas:

1. On the target Engine, navigate to ‘System’. Then select Replication.
2. Under Received Replicas, select the replica.

On this screen, you can browse the contents of replicas, as well as failover or delete individual replicas.

Deleting or failing over a replica will break any link with the replication source. Subsequent incremental updates will fail, requiring the source to re-establish replication. Failover should only be triggered when no further updates from the source are possible, as in a disaster scenario. Various replication use cases are also described in Replication Use Cases.

Multiple replicas can exist on the system at the same time. Active objects can exist in the system alongside replicas without interfering with replication updates. You can also use VDBs and dSources within a replica as a source when provisioning. For more information, see Provisioning from Replicated Data Sources or VDBs.

Failover and conflict resolution

To activate the objects in a replica, you must first fail over the replica. This will disconnect the replication connection and move the objects to the live system, where they can be actively used.

During the replication failover, there may be objects that conflict between the source Engine and the target Engine. For example, ‘groups’ will conflict if the failing over group has the same name as a group in the live system, as well as environments, dSources, and VDBs. By default, active objects with conflicting names will cause an error at the time of failover. In these scenarios, you must rename the active objects before the failover operation can complete successfully.

Given that conflicting names prevent failover from succeeding, best practices in a disaster recovery situation are to leave the target system completely passive with no active objects until the time that failover is required.

Once a replica is failed over, the objects are active but will be automatically disabled. The next section describes enabling these objects after replication is complete.

Enabling databases and environments

Objects may refer to states (IP addresses, mount paths, etc.) that differ between the source and target Engines. Because of this, all databases and objects within a replica are automatically disabled after a failover. This allows the administrator to alter configuration prior to enabling databases and environments, without the system inadvertently connecting to invalid systems.

After failover is complete, you must explicitly enable all dSources, VDBs, and environments. If you need to change any configuration for the target environment, you can do so prior to enabling the objects. In the event that a failing-over environment is consolidated with a live system environment, it must be refreshed before all of its databases can be used.

Replication use cases

Replication allows you to move Delphix objects such as dSources and virtual databases (VDBs) between Delphix Engines. These topics describe how you can use replication to meet different use cases:

- **Replicating to the Public Cloud**: With Delphix replication you can send data from engines deployed on-premise to the public cloud. In this case, you may have a Delphix engine in the production zone to ingest data securely with replication set up to the public cloud for development access and rapid, on-demand testing.
- **Disaster Recovery**: In the event of a disaster, you may need to failover your engine. Delphix replication enables you to configure a failover Delphix Engine to preserve the data and Delphix objects from the source engine for disaster recovery.
- **Geographically Distributed Development**: Often, development teams access data from all over the world. In this scenario, you may want to replicate data to be local to the developers who require access. With Delphix replication, you can provide developers with local access to Delphix datasets.
- **Data Migration**: Delphix supports simple migration of data and resources between Delphix Engines. There may be cases when you need to consolidate workloads between different Delphix Engines. With replication, you can easily migrate data as needed.

With Delphix replication, you can achieve the ideal Delphix deployment:

1. Ingesting data in production with a Delphix Engine deployed within the production zone.
2. Masking the data in production using Delphix Masking.
3. Replicating the masked data to a non-production environment, using Selective Data Distribution (SDD). SDD enables you to securely replicate masked data without compromising sensitive data from the source engine. For more information, read Selective Data Distribution Overview.

For more information on a few examples of Replication use cases, view the sections below.

Replicating to the public cloud

Enterprises will usually have the separate infrastructure for production systems and non-production development environments. For example, in the hybrid-cloud model, on-premise data centers are used to allow the company maximum ownership of these systems, while the public cloud is leveraged by development and automated testing teams to accelerate software development.

Delphix engine deployed in the public cloud

- Once a Delphix Engine has been deployed in the public cloud (using a supported cloud platform as described in Deployment) you can begin replicating from any source engine to that target engine.
- This enables you to provide access to production data from a Delphix Engine deployed on-premise to a Delphine Engine deployed in the cloud platform of your choice.

Disaster recovery

Replication is often used to provide recovery in the event of a disaster, where a data center or site becomes completely unavailable. To prepare for this, you may configure a failover Delphix Engine, which we will refer to as the *replication target Engine.* This target Engine will regularly receive replication updates from the original, or source, Engine so that if the source ever becomes unavailable, the target can be activated immediately.

Configuration steps

- **Passive target engine**

    - For disaster recovery, the target engine should be kept in a passive state until the source engine is lost. This prevents failover conflicts that may occur during the replication process.
    - At this point, a failover is performed that breaks subsequent replication updates and activates objects so that you can manage them on the target side.
- **Same configuration for source and target engines**
    - Target hosts and systems should exist at the target that matches the configuration of those at the primary engine. For example, if the source engine has two Red Hat environments discovered with four Oracle databases, the target engine should have exactly two Red Hat environments and four Oracle databases as well.
    - The failover engine should be provisioned with identical resources as the primary engine. For example, both engines should have the same number and types of disks attached as storage.
    - Finally, both engines should have the same network and storage topologies.

Failover object management

Once a failover occurs, there are two scenarios that will affect how you manage replicated objects, which include dSources, VDBs, and Environments.

- **Failure of Infrastructure Running the Delphix Engine Only**
    - You can enable dSources and VDBs and reconnect to the original environments that existed on the Source Engine.
    - You can reconfigure environments on the target Engine prior to failover as well.
- **Failure of Infrastructure Running the Delphix Engine and Delphix-connected Environments**
    - Environments will then have to be adjusted to point to the new systems on the target side.
    - If there is not a 1:1 mapping, then you can migrate the VDBs to new systems on the target, and you can detach dSources and attach them to the standby system in the target environment.

Follow the best practices below to simplify failover and meet performance expectations in the event of a disaster:

- To the extent possible, the failover Engine should mirror the primary Engine
    - Maintain a 1:1 relationship between source and targets. All data-related objects such as dSources, VDBs, and Environments as well as their configurations such as users and policies.
- The target Engine should remain passive and not be actively used for other workloads

Geographically distributed development

Organizations often have development teams distributed across different networks as well as different geographical locations. To improve performance or even meet security requirements, it may be necessary to replicate data from one location to another. The Delphix Engine allows for VDBs to be provisioned from replicated dSources and VDBs, as described in Provisioning from Replicated Data Sources or VDBs.

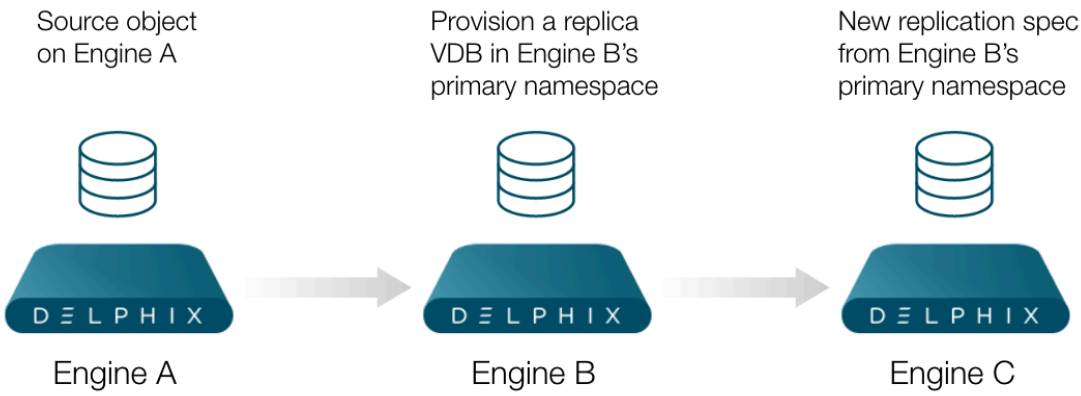
This use case differs from **Disaster Recovery** as replication is never broken and failover is never performed. You can refresh remote VDBs as long as the parent objects continue to exist on the source. If they are deleted, then remote VDBs will continue to exist but cannot be refreshed or updated from their original source VDB.

Configuration steps

- **Avoid Heavy VDB Workloads on the Source**: Because each replication stream induces load on the source system:
    - Minimize the number of simultaneous replication updates. Each source engine can support replicating to multiple target engines, but you should try to keep simultaneous updates to under three target engines per source.
    - If possible, avoid heavy VDB workloads on the Source Engine
- **The permanence of Source Objects:** Provision only from sources that are effectively permanent, since replicated VDBs cannot be refreshed once the source is deleted. If you delete a source object you will need to re-replicate the VDBs if they need to be refreshed.

- **Additional Storage Capacity on the Target Engine:** Provision additional storage capacity on the target Engine
    - Remotely provisioned VDBs can consume shared storage (via NFS mounts) on the target even when the parent is deleted on the source

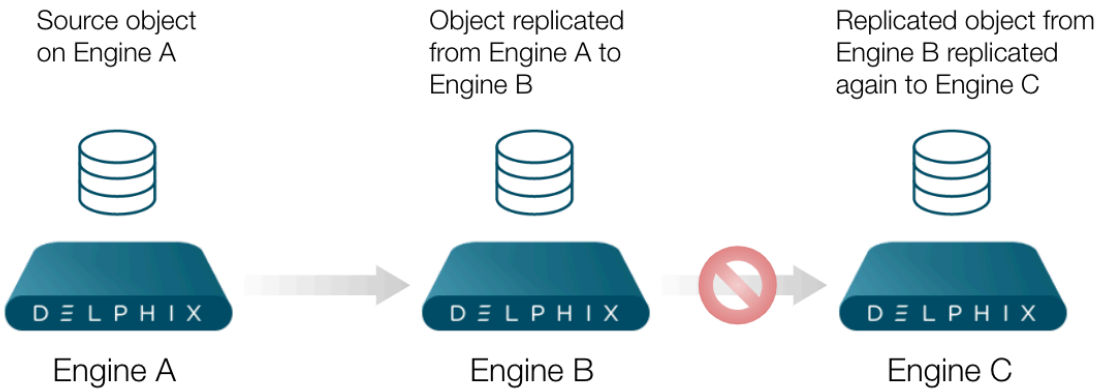
This use case supports more complex models such as 1-to-many and many-to-1. However, replication can only replicate objects that exist in the primary namespace. Consider a Delphix deployment with three Engines: Engine A, Engine B, and Engine C. The following workflow **is supported**:



1. **Engine A Replicates to Engine B**
2. **Provision a Replica VDB on Engine B**
3. **Engine B Replicates to Engine C**

In this scenario, you can replicate from Engine A to Engine B. Then, on Engine B provision a replica VDB into the primary namespace. Finally, add that object into a replication specification to replicate to Engine C. **Note**: Engine B can only replicate objects that exist in the primary (live) namespace to Engine C.

It is important to note the interim step of provisioning an on Engine B is required. For example, the following usage **is not allowed**:



1. **Engine A Replicates to Engine B**
2. **Engine B Replicates to Engine C**

You can replicate from Engine A to Engine B. But on Engine B, it is not possible, nor supported, to create a replication specification with objects in the replication namespace that are desired to be replicated to Engine C.

Data migration

You can use replication to perform a one-time migration of resources, such as virtual databases or environments, from one Delphix Engine to another. While the hypervisor provides tools to move virtual appliances between physical systems, there are times when migration is necessary, such as:

- Migrating between different physical storage
- Consolidating or distributing workloads across Delphix Engines

In these cases, you can use replication to copy a subset of objects across different topologies.

For migration, follow these best practices:

- Send full updates, followed by incremental updates, until the time required for incremental updates meets your downtime window
- Disable all objects to be migrated on the source, to ensure that they are not actively changing
- Send a final incremental update before failing over the target Engine
- After failover, delete any migrated objects on the source, or the entire Engine

Configuring replication

This topic describes how to configure data replication between Delphix Engines. Replication is configured with *Replication Profiles* that contain options such as the replication schedule, the hostname of the target Engine, and the selected objects that will be replicated.

Requirements

- **Version requirements**: the replication target can be on the same or newer version than the replication source.
- **Engine communication**: The target Delphix Engine must be reachable from the source Engine.
- **Storage allocation**: The target Delphix Engine must have sufficient free storage to receive the replicated data.
- **Privilege requirements**: The user in the replication profile must have administrative privileges on the source and the target engines.

Configuring the network

Delphix Replication uses a private network protocol to communicate between two Delphix Engines. You may specify a network interface to run replication by configuring routing to direct traffic over a particular interface.

The replication network protocol uses TCP port 8415. If there is a firewall between the source and target that is blocking this port, then there are two possible solutions:

1. Enable port 8415 on the firewall in order to allow connections to this port from the source to the target.
2. Replication can connect through a SOCKS proxy if one exists. Configure the SOCKS proxy address and port by connecting to the command-line interface (CLI) as a system administrator user and navigating to "service proxy" to update the SOCKS configuration.

Port 1080
SOCKS port 1080 is used by default but can be overridden

Replication can connect through a SOCKS proxy if one exists. Configure the SOCKS proxy address and port by connecting to the command-line interface (CLI) as a system administrator user and navigating to "service proxy" to update the SOCKS configuration. Example:

Example of a SOCKS Proxy

```
dlpx-engine> service proxy
dlpx-engine service proxy> update
dlpx-engine service proxy update *> set socks.enabled=true
dlpx-engine service proxy update *> set socks.host=10.2.3.4
dlpx-engine service proxy update *> set socks.username=someuser
dlpx-engine service proxy update *> set socks.password=somepassword
dlpx-engine service proxy update *> commit
dlpx-engine service proxy> get
    type: ProxyService
    https:
        type: ProxyConfiguration
        enabled: false
        host: (unset)
```

```
        password: (unset)
        port: 8080
        username: (unset)
    socks:
        type: ProxyConfiguration
        enabled: true
        host: 10.2.3.4
        password: ********
        port: 1080
        username: someuser
```

Configuring the replication source Delphix engine

1. On the source Delphix Engine, click **System**, then **Replication**.
2. In the left-hand navigation section, click **Create Profile**.
3. Enter the following required fields:
    a. Name of the replication profile
    b. The hostname or IP address for the target Delphix Engine.

Replication profile options

There are several configuration options for your replication profiles. These give you more granular control on options such as when replication will run, how much bandwidth it may use, and which objects are replicated. Details for each option are described below.

The following configurable options are static and can not be configured at run-time. You can set these configurations while a replication spec is being executed, but the values will be applied only after the next execution.

- **Automatic replication**: A policy to automatically run replication. With this option, you can set up replication based on the schedule you need. Automatic replication allows you to define a policy to automatically run replication. By default, automatic replication is disabled, meaning that you must trigger replication updates manually. To enable automatic replication, click the Enabled checkbox. With this setting, you can enter the frequency and time for replication updates to the target Delphix Engine. Automatic replication uses Quartz, a job scheduling tool (http://www.quartz-scheduler.org/), for scheduling, which can be configured via the Advanced option.
- **Traffic Options**: Various traffic and bandwidth options are available. For example, you may want to enable encrypted traffic or limited bandwidth during replication updates.
    - **Encrypting traffic:** By default, replication streams are unencrypted, which provides maximum performance on a secure network. However, this setting allows you to encrypt traffic during replication.
      Note:
      Encrypting Replication Encrypting the replication stream will consume additional CPU resources and may limit the maximum bandwidth that can be achieved. During replication, the Delphix Engine will negotiate an SSL connection with its server peer to use TLS_AES_128_GCM_SHA256 as the cipher suite and TLSv1.3 as the protocol.
    - **Network connections**: Allows setting the number of underlying network connections that can be used by replication.
    - **Limiting bandwidth:** By default, replication will run at the maximum speed permitted by the underlying infrastructure. In some cases, particularly when a shared network is being used, replication can increase resource contention and may impact the performance of other operations. This option allows administrators to specify the maximum bandwidth that replication can consume.

- **Objects Being replicated**: Select the objects you wish to be replicated from the source engine to the target engine. In the right-hand column, under Objects Being Replicated, you can select the objects you want to replicate. Some selected objects may have dependencies – other objects that will be pulled into replication because they share data.

  - This is not guaranteed to be the full set of dependent objects. The full set of objects and their dependents will be calculated at the time of replication.
  - You can not configure a Replication Profile without selecting an object and the last object from a Replication Profile can not be removed. If you need to remove the last object from the Replication Profile, you must delete that Replication Profile.
  - The object is removed from the replication target namespace only after a subsequent replication job is executed for the associated replication specification.
    a. Remove the object from the specification on the source
    b. Execute the replication specification
    c. The object is removed as a part of the replication job. The object is not removed while modifying the replication specification. To add back the object, the entire object needs to be sent again.

1. When replicating a group, all dSources and VDBs currently in the group, or added to the group at a later time, will be included.
2. If you select a Delphix Self-Service data template, all data containers created from that template will be included. Likewise, if you select a data container, its parent data template will be included.
3. Regardless of whether you select a VDB individually or as part of a group, the parent dSource or VDB (and any parents in its lineage) are automatically included.
   a. This is required because VDBs share data with their parent object.
   b. In addition, any environments containing database instances used as part of a replicated dSource or VDB are included as well.
4. When replicating individual VDBs, only those database instances and repositories required to represent the replicated VDBs are included. Other database instances that may be part of the environment, such as those for other VDBs, are not included.
5. Non-data objects (Delphix users, roles, permissions, authorizations, policies, database configuration templates) that are associated with selected objects will be automatically included during replication. They will not be shown on the selection interface.

Configuring the target Delphix engine

Additional configuration on the target engine is not needed. Replicated objects will appear in an alternate received replica (or namespace) that mirrors the original object layout.

To view replicated objects from the Delphix Engine:

1. Click System, then select Replication.
2. Look under Received Replicas. All replicated objects are read-only until the replica is failed over. For more information about managing replicas and how to activate a replica, see the topics Replicas and Failover and Controlled Failover.

You can create and manage objects on the target server without affecting subsequent updates, though this can cause conflicts on failover that require additional time to resolve. For disaster recovery use cases, it is recommended to keep the target passive and not create any local objects. This will avoid conflicts and guarantee a smooth failover operation.

Multiple sources can replicate to the same target, allowing for the flexible geographical distribution of data. This is not a recommended practice for disaster recovery, because it increases the probability of conflicts on failover and

may oversubscribe resources on the target if multiple replicas are failed over and there is insufficient infrastructure to support the combined workloads.

Controlled failover

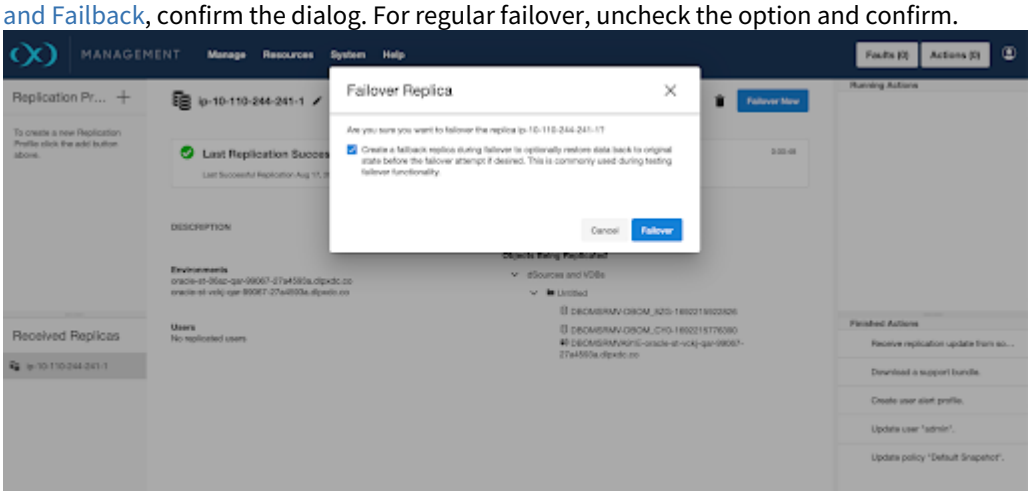
This topic describes the process of failing over a replica. Objects stored in a replica are read-only, and failing over a replica moves the replicated objects to the live system. After a failover, all of the objects will appear in the system as if they had been created locally.

Prerequisites

- A Delphix system that contains a replica is required.
- For an overview of what replicas are and what failover implies, see Replicas and Failover.
- For more information on configuring replication, refer to the Configuring Replication topics.

Procedure

1. Locate the replica to failover.
    a. Click **System**.
    b. Select **Replication**.
    In the **Received Replicas** section, you will see the list of replicas. Each replica has a default name which is the hostname of the source that sent the update. If you wish, you can customize these names. Each replica will list the databases and environments it contains.
    Note:
    If this replica is the result of a replication update, check to see whether or not the source Delphix appliance is still active. If it is still active, then disable any dSource or VDB that is part of the replica being failed over to ensure that only one instance is enabled. You can disable dSources and VDBs by selecting **Datasets**, finding the appropriate database, select **Disable**. After disabling the objects, navigate to **System > Replication** and click **Replicate Now** to get the most recent data to the target environment. If any scripts are accessing the engine, then you must repoint those scripts to the replication target environment to avoid VDBs from being started up in the old replication source environment.
2. Click **Failover** and a confirmation dialog will appear as below.To perform a test-failover see, Test-failover and Failback, confirm the dialog. For regular failover, uncheck the option and confirm.



    This will pause while the replica is failed over.
3. When failover completes, the replica page will update.
4. Apply any configuration changes that are required to customize the objects for the system. This might include updating object states such as IP addresses, mount paths, or credentials. For more details, see the Replicas and Failover topic.
5. Enable the environments that were failed over.

    a. Click **Manage**.
    b. Select **Environments**. The environments that were failed over will be disabled.
    c. From the Actions menu (...) select **Enable.**
6. Refresh any environments that were consolidated during failover.
    a. Click the **Refresh** icon for each affected environment.
7. Enable the dSources and VDBs that were failed over.
    a. Click **Manage**
    b. Select **Datasets**.
    c. Select the desired **database**.
8. From the Actions menu (...) select **Enable.**

> If the dSources and VDBs that were failed over belong to a plugin, depending on what version of the plugin existed on the target engine, the plugin may be in an inactive state. The plugin must be moved out of the inactive state before the object can be enabled. For more information see the Delphix Engine Plugin Management topic.

After replication failover

Navigate to **System > Replication** and select the delete button to delete the replication specification.

A failover is similar to a migration operation for TDE-enabled vPDBs. If after the failover the original target hosts are still available, then the vPDB can be enabled on the failed over Delphix Engine without requiring additional manual steps. If the vPDB is to be restored to a different target host, then the same manual steps are required for migration, as the vPDB will now be located on a new target and thus needs the artifact directory, parent Keystore, and merged Keystore available.

Uncontrolled failover

Before starting an uncontrolled failover, keep a note of the source and target engine's IP address and engine UUID. These details are available on the Delphix server setup application

Uncontrolled failover procedure

1. On the source engine, create a replication spec for the dSources and VDBs, by clicking **Replicate now** to replicate these dSources/VDBs to the target engine.
2. To simulate a crash of the source engine, Power of the source engine.
3. On the target engine, click on **Failover.**
   At this point, the target engine will become active and the replicated environments/dSources/VDBs will be seen in the default namespace of the target engine.

The following sections describe the procedures to handle an Uncontrolled Failover for each data platform:

- Oracle environment and data sources
- SAP ASE environment and data sources
- SQL server environment and data sources

Oracle environment and data sources

After an uncontrolled failover, for Oracle SI and RAC complete the following procedure:

1. Enable dSources and VDBs on the target engine. dSources will be enabled but the VDBs/vPDBs will fail.
2. To enable VDBs, shutdown the database instance using **SHUTDOWN ABORT**.
3. Remove all the mounts from the target environment with sudo privilege.
    a. List all the mounts on the target environment using the `mount` command.
    b. Run `umount -lf <Delphix-mount>` to remove the mounts. For example: `umount -lf /mnt/provision/VDBOMSR8A1718_FPX/datafile`
4. Remove all the stale oracle processes of the database instance using `kill -p <process id>.`
    a. Run `ps -ef | grep “DATABASE INSTANCE` to get all the running processes associated with the database instance. For example `ps -ef | grep VDBOMSR8A1718_FPX`. This command list all the running processes, e.g. **oracle 15236 9227 0 11:01 pts/0 00:00:00 grep --color=auto VDBOMSR8A1718_FPX**
    b. Run `kill -p <process id` to kill all the running processes. For example: `kill -9 15236`
5. Now enabling VDBs and VPDBs will be successful.

SAP ASE environment and data sources

After an uncontrolled failover has been triggered, on the target/staging host, the SAP ASE dSource/VDBs mounts, corresponding to the source engine, will still be present. This can be checked using the following command on the target host:

mount | grep <source-engine-ip>
These mounts will be mounted in the user-provided toolkit directory (Example */work*) and these would be of the form **/ork/<Engine-UUID>-[staging/vdb]/[datafile/archive/temp**]. The following is an example for a dSource staging database mount **/work/4201763f-2f8d-1c8f-381a-effb180b0328-staging-2/datafile**.

All the mounts will be using the source engine's UUID.

Attempting to '**Enable**' or '**Start**' the ASE dSources/VDBs on the target engine while the NFS mounts are in this stale state, will cause the jobs to hang or timeout or result in a job failure.

To clear off these stale mounts, one of the following options can be used:

Manually unmount

1. Shutdown the concerned SAP ASE instances which host the dSource's staging database and VDBs.
2. List all the dSource/VDB mounts with a **mount | grep <source-engine-ip>.**
3. Remove all the mounts with a **umount -lf <Delphix-mount>** for example: **umount -lf /work/ 4201763f-2f8d-1c8f-381a-effb180b0328-staging-1/archive**, this might require sudo privileges. NOTE: Use the appropriate syntax for **umount** for your operating system. The above example is for Linux.
4. **Restart** the SAP ASE instances.

OR

Reboot the target/staging host

Reboot the target/staging host corresponding to the concerned dSources/VDBs. This option is probably the cleanest way of clearing the source engine's leftover dSource/VDB mounts, but should only be used if the concerned target/staging host(s) do not host any other databases that are not part of the uncontrolled failover.

Enabling Environments, dSources, and VDBs

1. Enable the source environment and then the target environment.
2. Enable the dSources and VDBs on the target engine.
3. Mounts will be moved to a new Delphix(Target Engine) and can be checked by using the following command: **mount | grep <target-engine-ip>** These mounts still use the source engine's UUID and not the target engine's UUID, but validated/manual syncs on dSources and refreshes/rewinds on VDB operations will be successful.

SQL server environment and data sources

Enabling environments and dSources

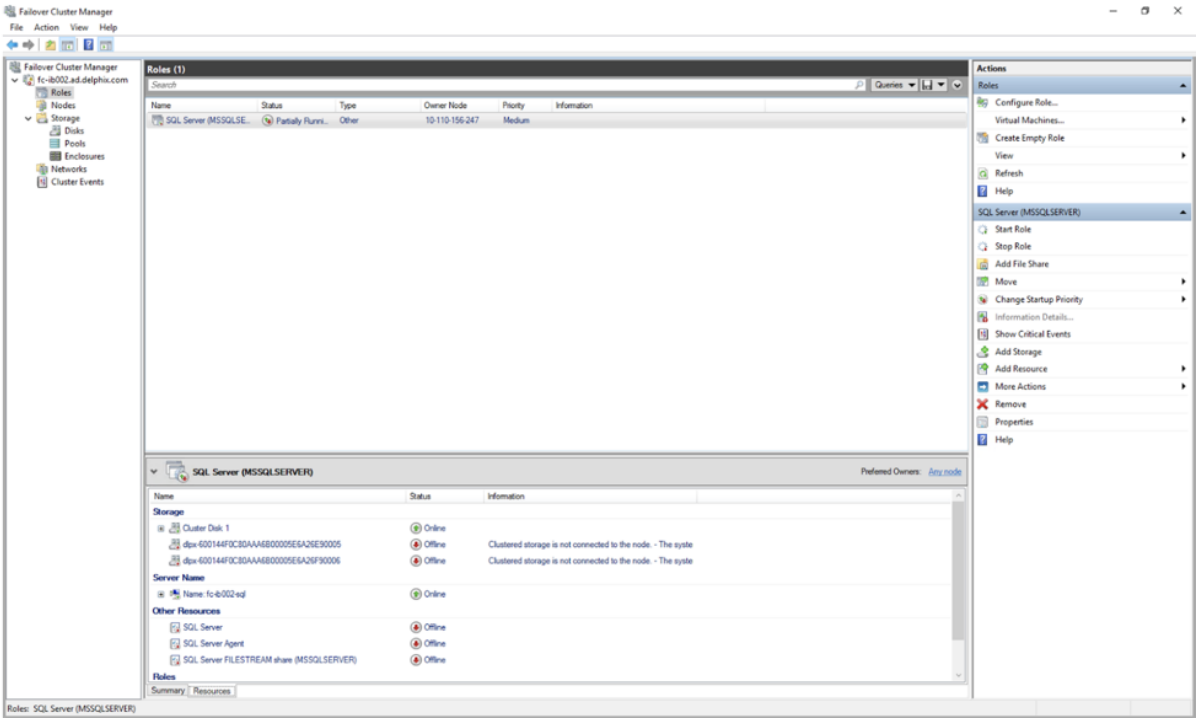
1. Enable the proxy(staging) environments followed by the source environments.
2. Enable dSources on the target engine.
3. Delete the staging database stale mounts from the connector directories. An example directory: **C:\ProgramFiles\Delphix\DelphixConnector\<Old Engine ID >-staging-<X>** where **X** is an arbitrary id representing the staging database.
4. Delete the staging database from the SQL instance on the proxy host using the following convention: <**Old Engine ID**>-**staging**-<**X**> where **X** is an arbitrary id representing the staging database.

Enabling VDBs on the standalone host

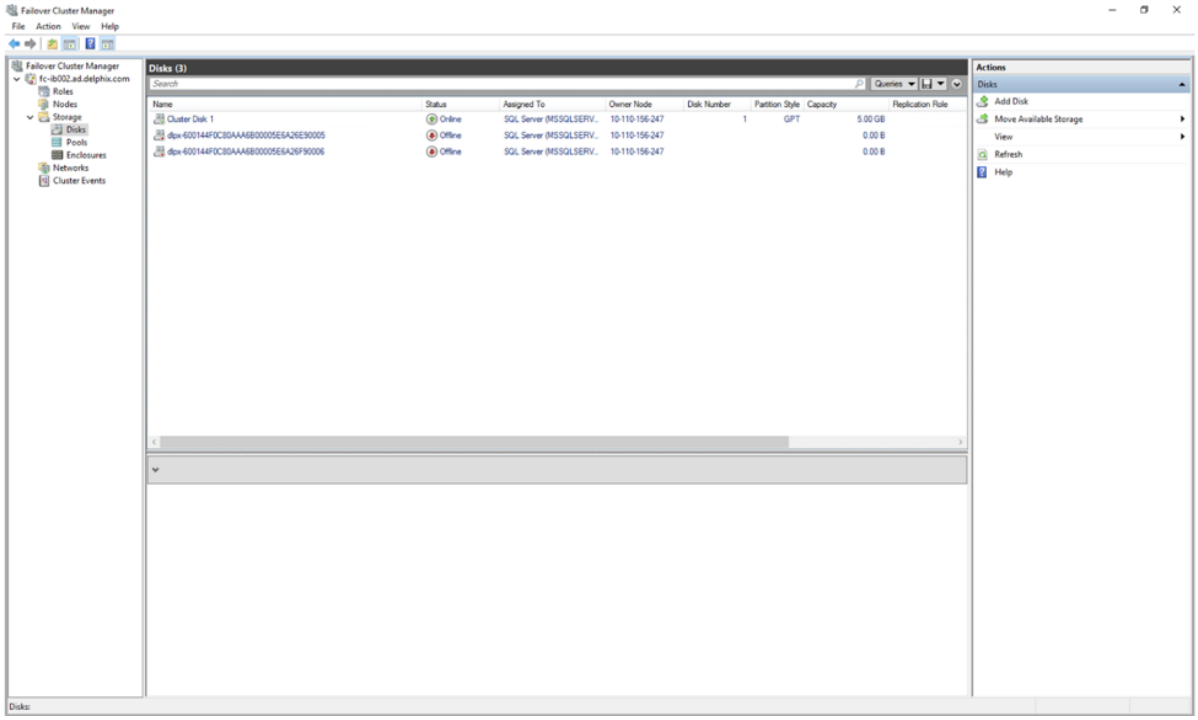
1. Delete the VDB directories stale mounts from: **C:\ProgramFiles\Delphix\DelphixConnector\\<Old Engine id>-vdb-<X>** where **X** is an arbitrary id representing the virtual database.
2. From the SQL instance delete VDB databases that are in recovery pending state.
3. Enable VDBs. New mounts will be created and a new engine ID will reflect on those mounts.

Enabling VDBs on failover cluster as target

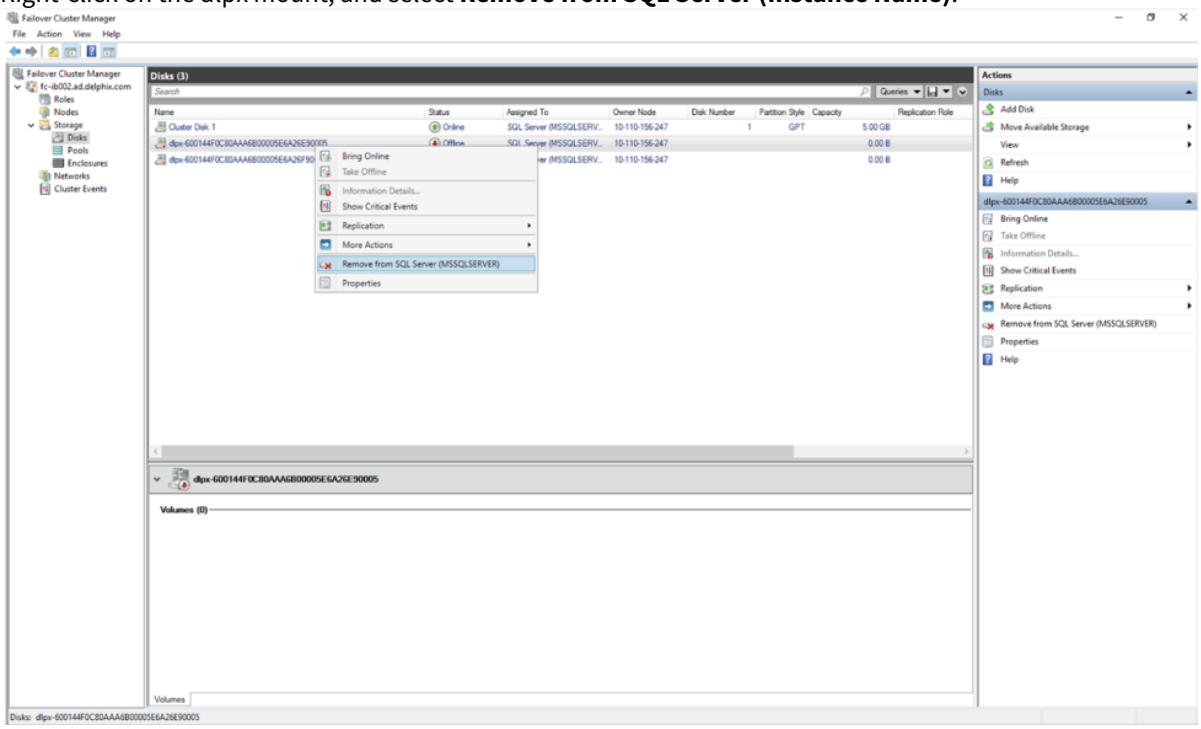
1. Once the source Engine is powered down login to **Failover cluster**.
2. Go to **Windows Administrative Tools** and select **Failover cluster manager**
3. On the left pane of the **Failover cluster manager** go to the **Roles**. As shown in the image below you will see the Failover cluster in the following state with the indicated status.



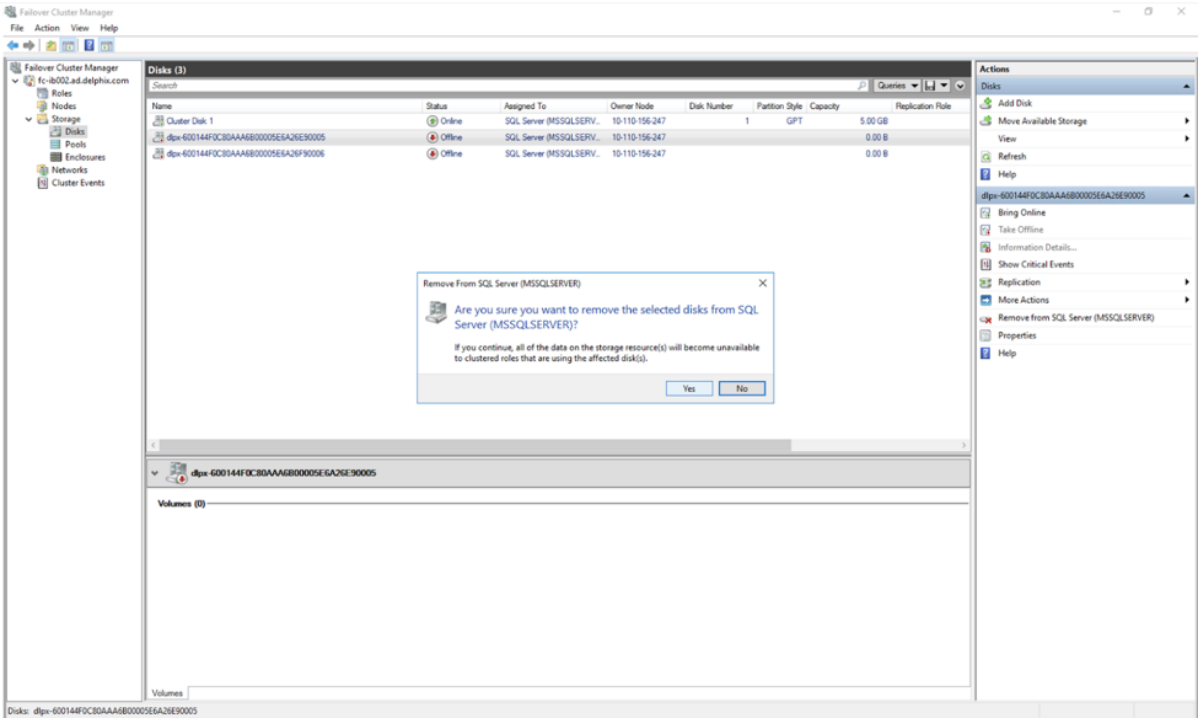
4. In order to fix the state go to the **Storage** section and select **Disks**. The **dlpx-XXXXXXXXX** mounts attached to the Cluster Disk will be in offline states. These mounts will be equal to the total number of VDBs provisioned on that cluster as the target.



5. Right-click on the dlpx mount, and select **Remove from SQL Server (Instance Name)**.



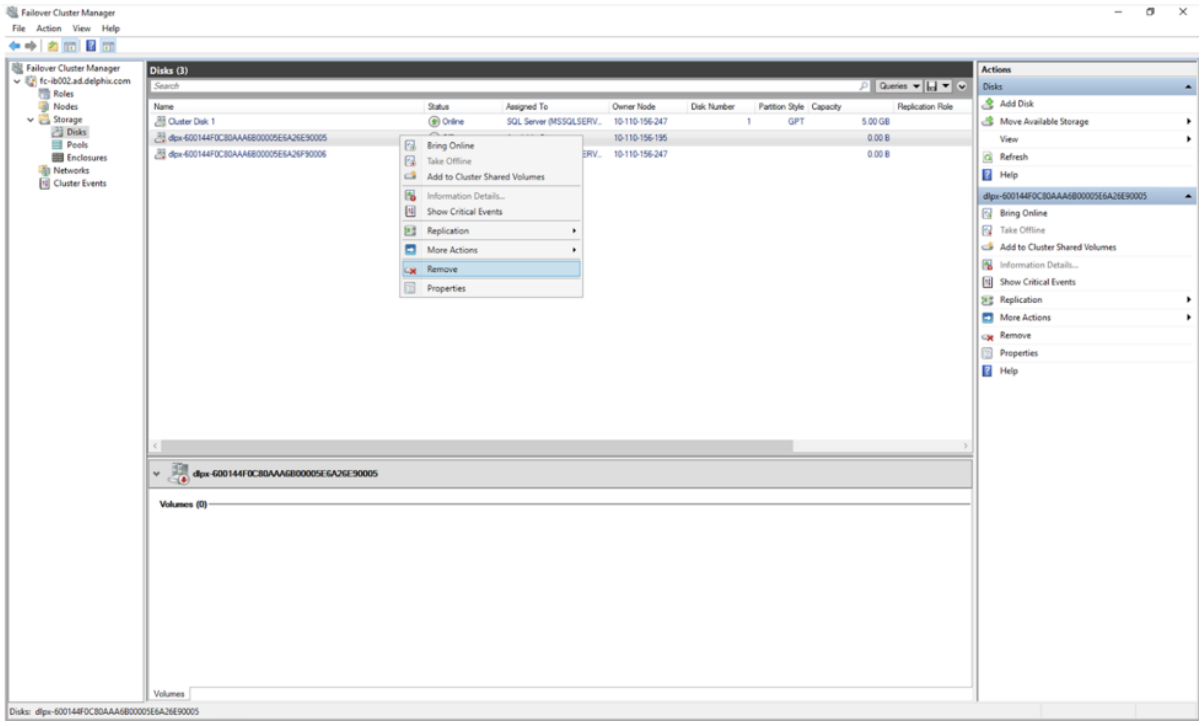
6. Click **Yes**.



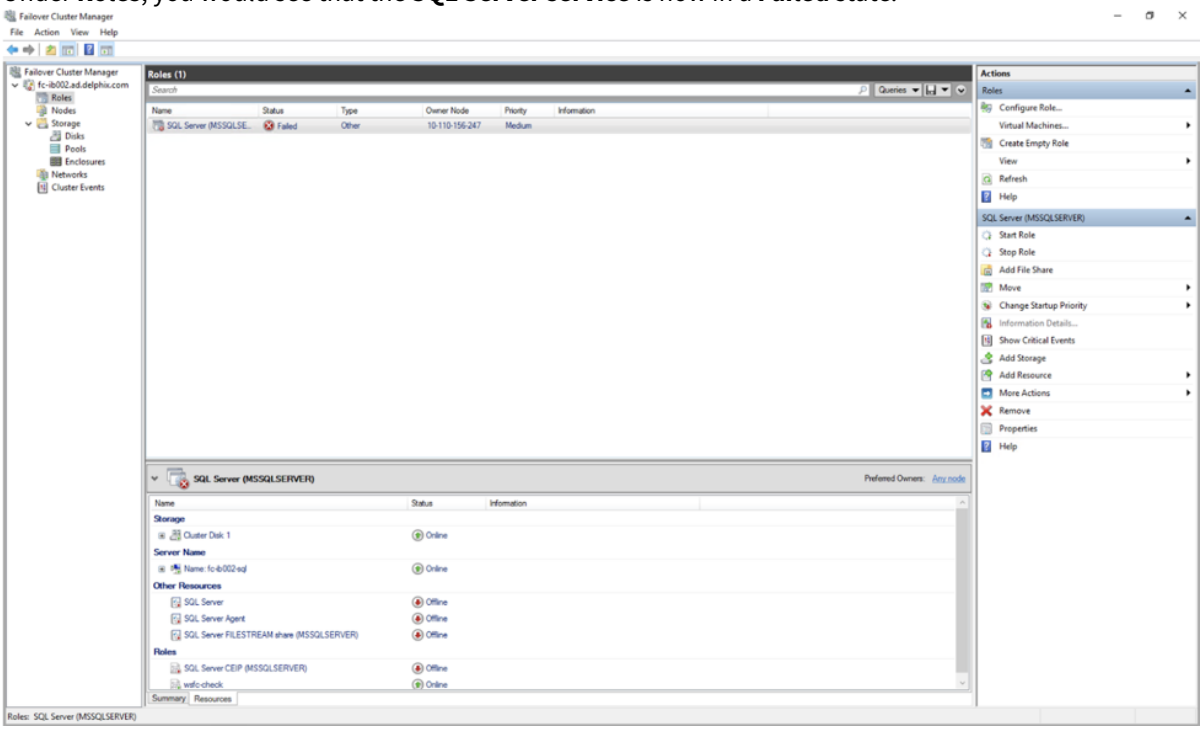
In the **Assigned To** column, you will see **Available Storage**.

Invalid file id - fbb88d…

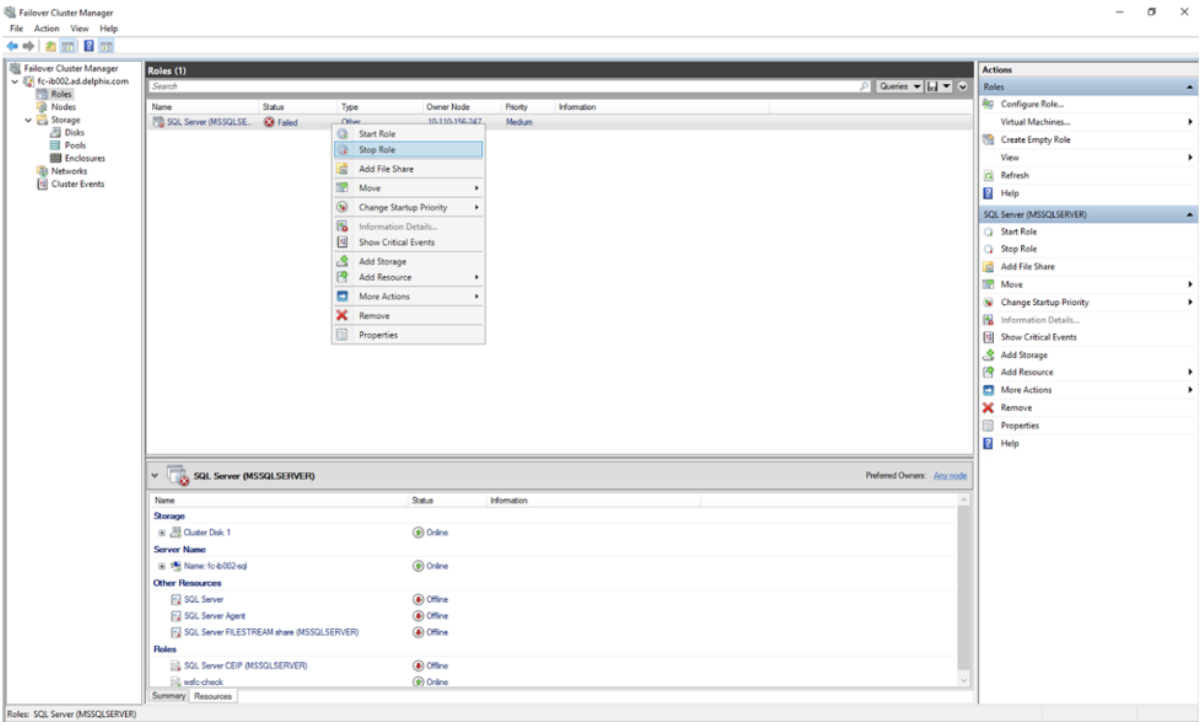
7. Right-click on the **dlpx mount** and click on **Remove**.



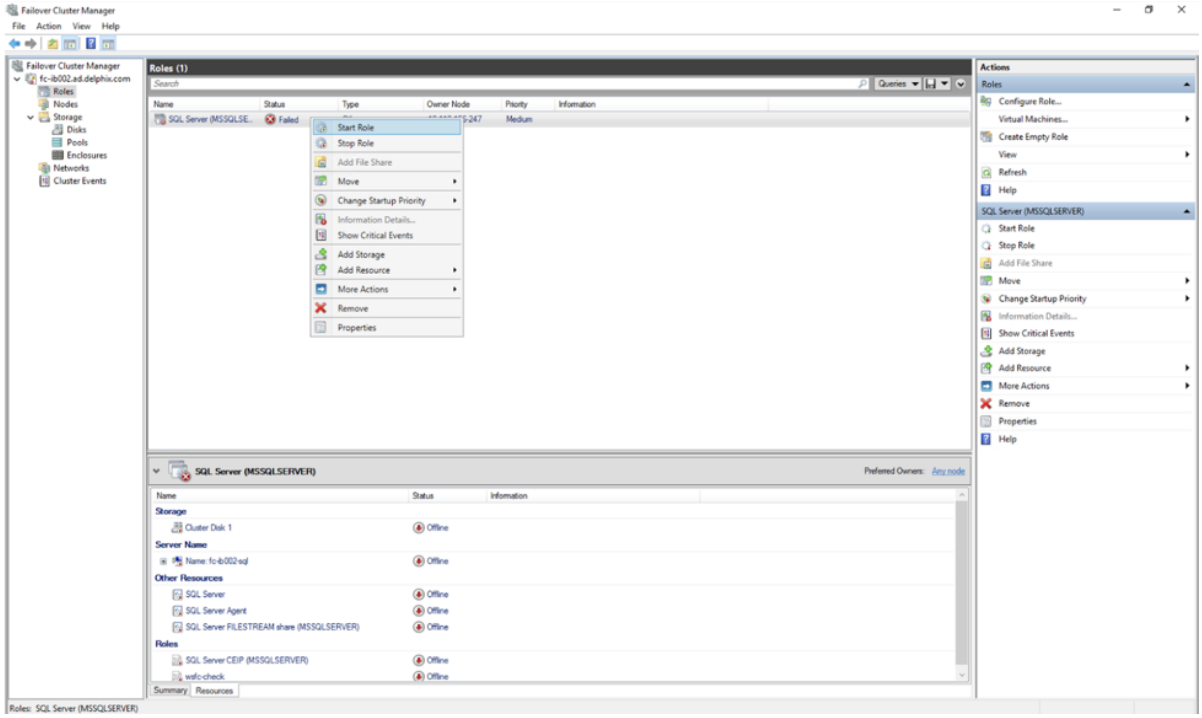
8. The dlpx mount will be deleted. Repeat the same process with other dlpx mounts assigned to **Available Storage**.
9. Under **Roles**, you would see that the **SQL Server service** is now in a **Failed** state.



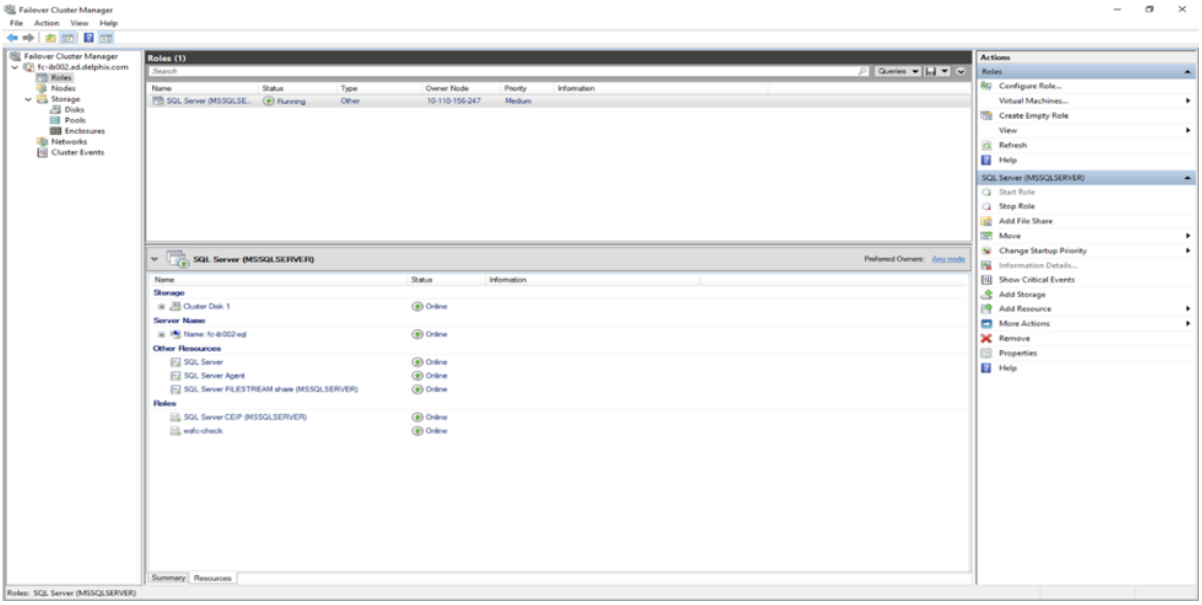
In order to fix it right-click on the service and click **Stop Role**.



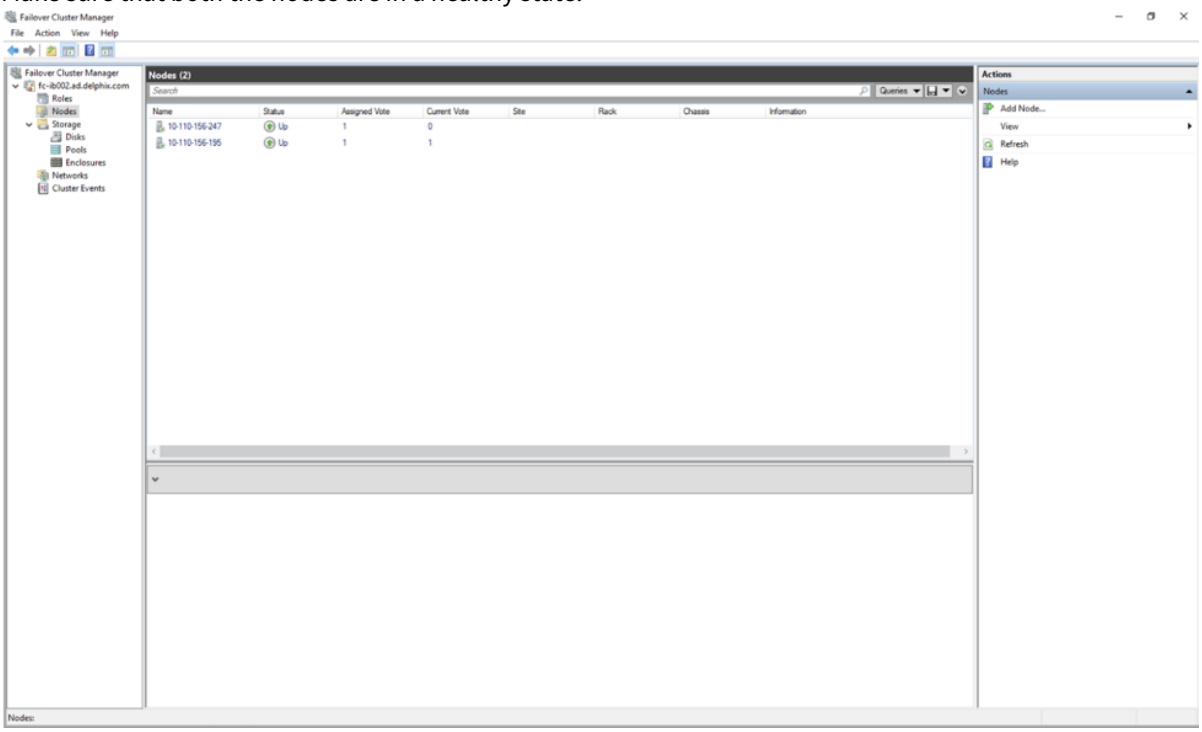
10. Once the Stop role is executed properly, right-click on the **SQL cluster service** and click on **Start Role**.



11. The SQL cluster service should now be online. Under the **Resources** tab the Storage, Server Name, Other Resources and Roles should have Online status. Ensure that the Disks and Nodes are also in an online state.



12. Make sure that both the nodes are in a healthy state.



Managing replicated objects

Once the failover process has completed the last step is to enable the objects on the Delphix Engine that received a replica. There are two major concepts:

1. **SSH Keys**: Public keys need to be re-published on the source, staging, and target environments for the replicated Delphix Engine, since the replication process does not automatically copy SSH keys.
2. **Enabling Replicated Objects**: Objects will need to be re-enabled once replication is complete. This ensures that replication has completed successfully and the configuration of each object is correct.

SSH keys

If you were previously using a password-less SSH key exchange on your source and/or target environments, you may need to publish the now-primary engine's public key for continued functionality.

1. Click **Manage**.
2. Select **Environments**.
3. Select **any environment**.
4. Under **Environment Users**, click the **edit** icon () next to the defined user.
5. Click **View Public Key** to display the public key for the Delphix Engine.
6. Highlight the public key string (starts with "ssh-rsa") and copy the key to your clipboard (Ctrl+C in Windows).
7. On each source and target host within your defined environments, paste the public key into the environment user's authorized_keys file, which is normally located in the user's ~/.ssh/ directory.

Once the public key has been copied to the hosts making up your environments, you are ready to enable the remaining objects.

Enabling environments

In order to begin using the environments and related dSources and VDBs, the environments must first be started.

1. Login to the Delphix Management application.
2. Click **Manage**.
3. Select **Environments**.
4. Select an **environment** that you want to enable.
5. From the Actions menu (...) select **Enable.** This will initiate jobs that will refresh and enable the environment.
6. Repeat these steps for each environment that you want to enable.

Enabling dSources

In order to again have actively syncing dSources, you must enable them.

1. Login to the Delphix Management application.
2. Click **Manage**.
3. Select **Datasets**.
4. Click the **dSource** you want to enable.
5. From the Actions menu (...) select **Enable.**
6. Click **Enable** to confirm.

Enabling VDBs

The final step in the failover would be to enable any needed VDBs.

1. Login to the Delphix Management application.

2. Click **Manage**.
3. Select **Datasets**.
4. Click the **VDB** you want to enable.
5. Click the **Configuration** tab.
6. From the Actions menu (...) select **Enable.**
7. Click **Enable** to confirm that you want to enable the VDB.

At this time the failover is complete. All objects previously running on the previous source Delphix Engine should now be running off of the target Delphix Engine on which the objects were failed over.

Replicas and failover

Replication recreates objects on the target system in a replica that preserves object relationships and naming on the target server without interfering with active objects on the system. Objects within a replica are read-only and disabled until a replica is failed over, at which point they can be activated. VDBs and dSources within a replica can be used as the source for provisioning new VDBs.

Replicas

A replica contains a set of replicated objects. These objects are read-only and disabled while replication is ongoing. To view replicated objects, look under **namespace** in the CLI. Or in the GUI:

1. Click **System**.
2. Select **Replication**.
3. Under **Received replicas**, select the replica.

On this screen, you can browse the contents of replicas, as well as failover or delete individual replicas. As described in the Replication overview topic, databases (dSouces and VDBs) and environments are included within the replica.

Deleting or failing over a replica
Deleting or failing over a replica will sever any link with the replication source. Subsequent incremental updates will fail, requiring the source to re-establish replication. Failover should only be triggered when no further updates from the source are possible, as in a disaster scenario.

Multiple replicas can exist on the system at the same time. Active objects can exist in the system alongside replicas without interfering with replication updates. You can also use VDBs and dSources within a replica as a source when provisioning. For more information, see Provisioning from replicated data sources or VDBs

Failover and conflict resolution

To activate the objects in a replica, you must first fail over the replica. This will sever replication and move the objects to the live system, where they can be manipulated in the same fashion as other objects on the system.

Objects that are failing over can conflict with objects in the live system. One reason for conflicts is identical names. For example, Groups will conflict if the failing over Group has the same name as a Group in the live system, as will most other objects including Environments, dSources, and VDBs.

Most of the object conflicts can be resolved automatically. Objects like Groups will be renamed to avoid conflict and objects like Environments will be merged and consolidated. After conflict resolution and successful failover, a report is presented with a list of objects that needed conflict resolution.

Conflict resolution is not supported for objects like dSources and VDBs.

Conflicting plugins

Due to the potential for plugin version incompatibilities, if any of your replicated objects are plugin-based, we highly recommend that you leave the target system completely passive with no active objects until the time that failover is required. For more information, see the Delphix engine plugin management page.

When environments are consolidated, the Delphix engine maintains the existing environment configuration and privilege elevation profile; and merges replicated objects with them. Therefore, you must check and update the configured privilege elevation profiles when Environments are consolidated.

Once a replica is failed over, the objects are active but will be automatically disabled.

Manual conflict resolution

To resolve conflicts in Groups, you may either rename the conflicting group on the target replication engine or if the source replication engine is still available, rename on the source replication engine and send a replication update, before failover. For all other object types, because the replica objects are read-only, you must rename the active objects on the replication source engine and send a replication update before the failover operation can complete successfully.

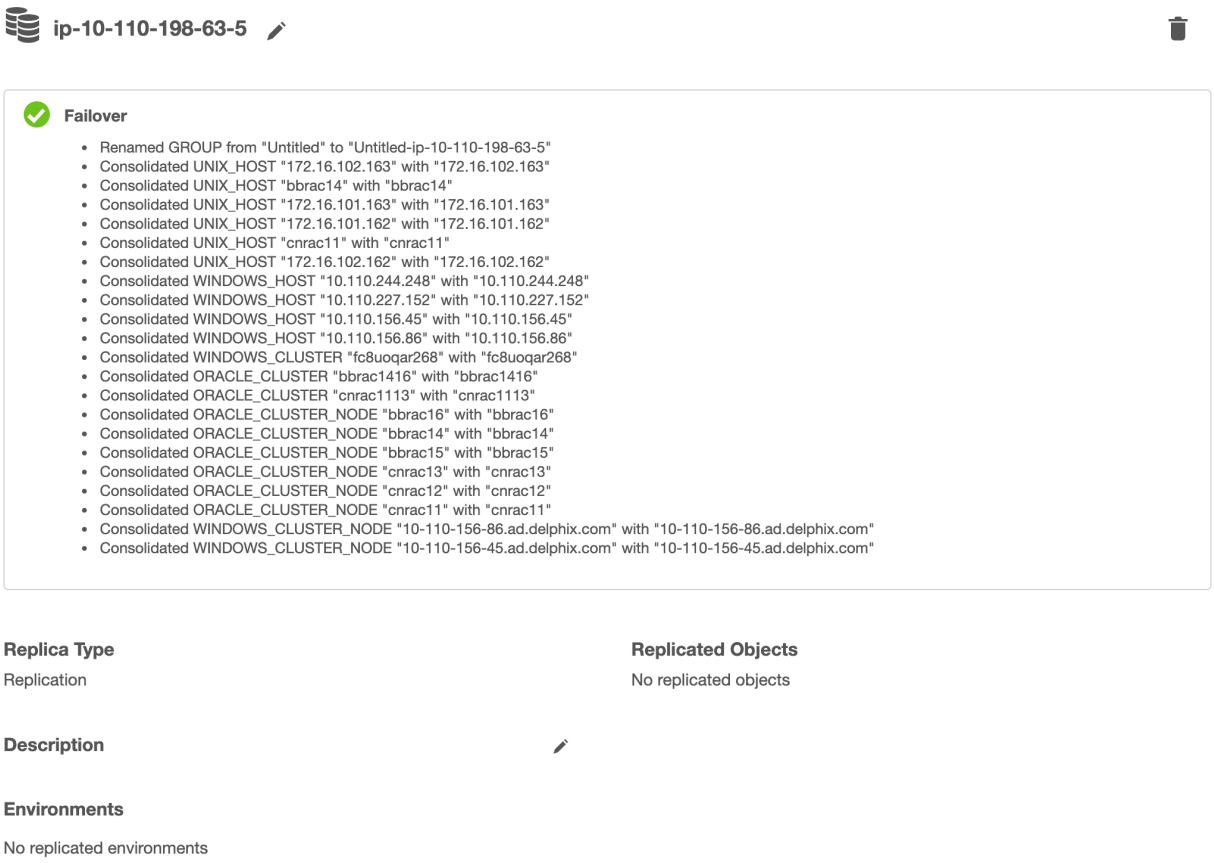
Automatic conflict resolution

> Starting with 9.0.0.0, smart-failover a.k.a automatic conflict resolution is no longer an option. It is always on and there a GUI option is no longer presented. Starting 14.0.0.0, the enableFailback option is presented instead.

Starting with 6.0.2.0, most of the object conflicts can be resolved automatically. When you select "Automatically resolve object conflicts", replica objects like Groups will be renamed whereas objects like Environments will be merged and consolidated.

As of 6.0.5.0, the Automatic Conflict Resolution option will be chosen by default.

After automatic conflict resolution and successful failover, a report is presented with a list of objects that needed conflict resolution.



Automatic conflict resolution is not supported for objects like dSources and VDBs.

Enabling databases and environments

Objects may refer to states (IP addresses, mount paths, etc) that differ between the source and target system. Because of this, all databases and objects within a replica automatically start in the disabled state after a failover. This allows the administrator to alter configuration prior to enabling databases and environments, without the system inadvertently connecting to invalid systems.

After failover is complete, you must explicitly enable all dSources, VDBs, and environments. If you need to change any configuration for the target environment, you can do so prior to enabling the objects. In the event that a failing-over environment is consolidated with a live system environment, it must be refreshed before all of it's databases can be used.

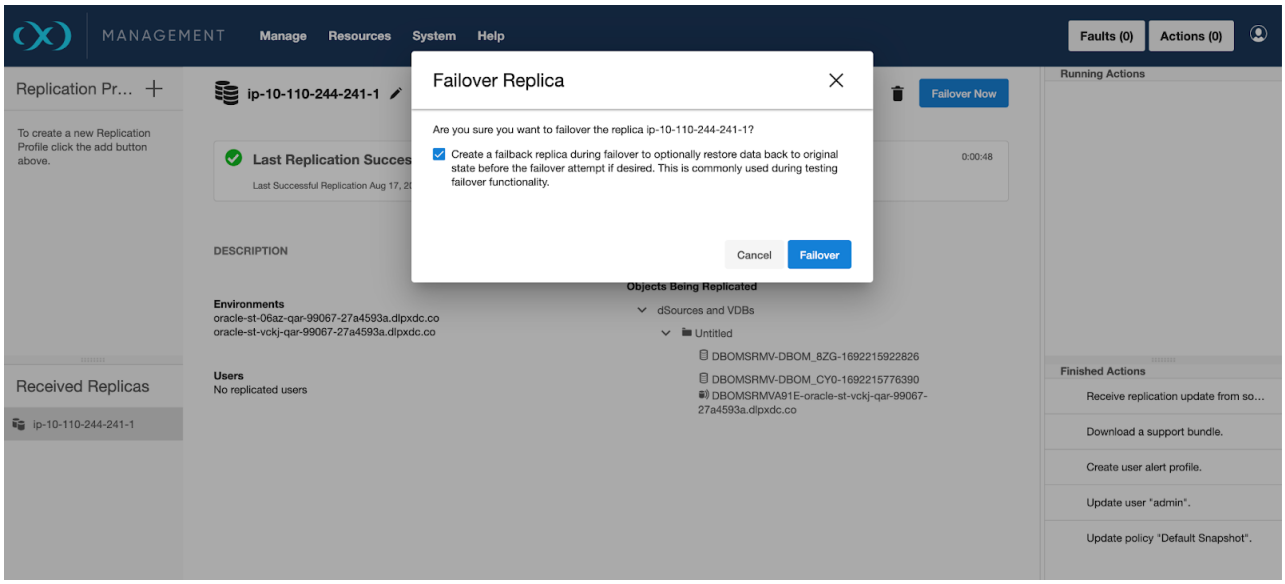
Test-failover and failback

Test-failover

Starting with 14.0.0.0, a test-failover can be performed to validate that the failover actually works when disaster strikes. The failover operation can be undone by doing a failback operation.

The failback operation takes the replication target back to the pre-failover state so that replication incremental can be received.

During failover, there is an option to save the state so that a failback can be performed. This is the `enableFailback` option and it is turned on by default.



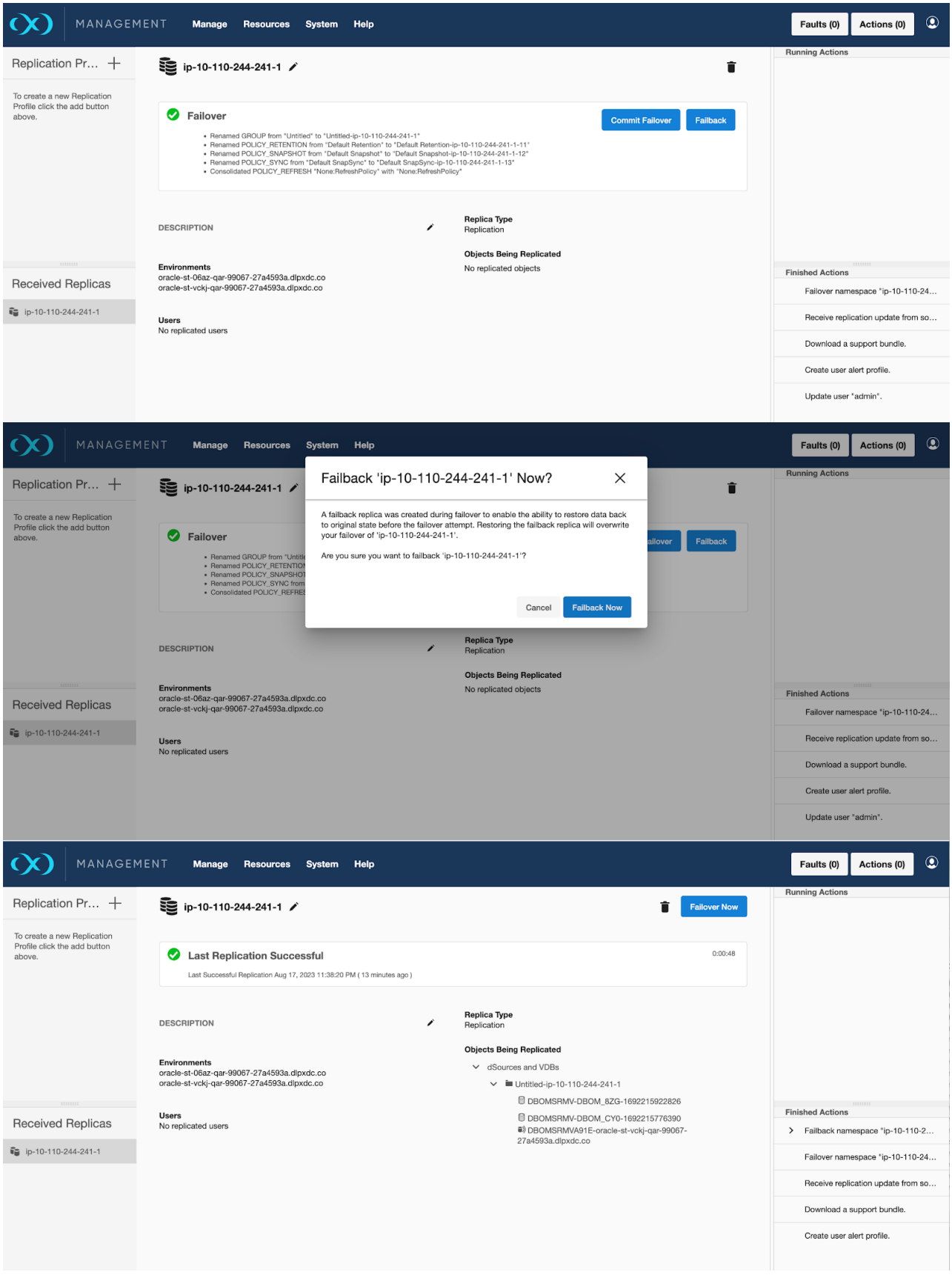
Once failover is complete, environments and datasets may be enabled and activated as mentioned in Controlled failover.

A test-failover does not place restrictions on the kinds of operations that may be performed on the failed over data. Environments and datasets may be enabled and used. New snapshots may be created. New VDBs may be provisioned from the failed-over datasets.

Failback

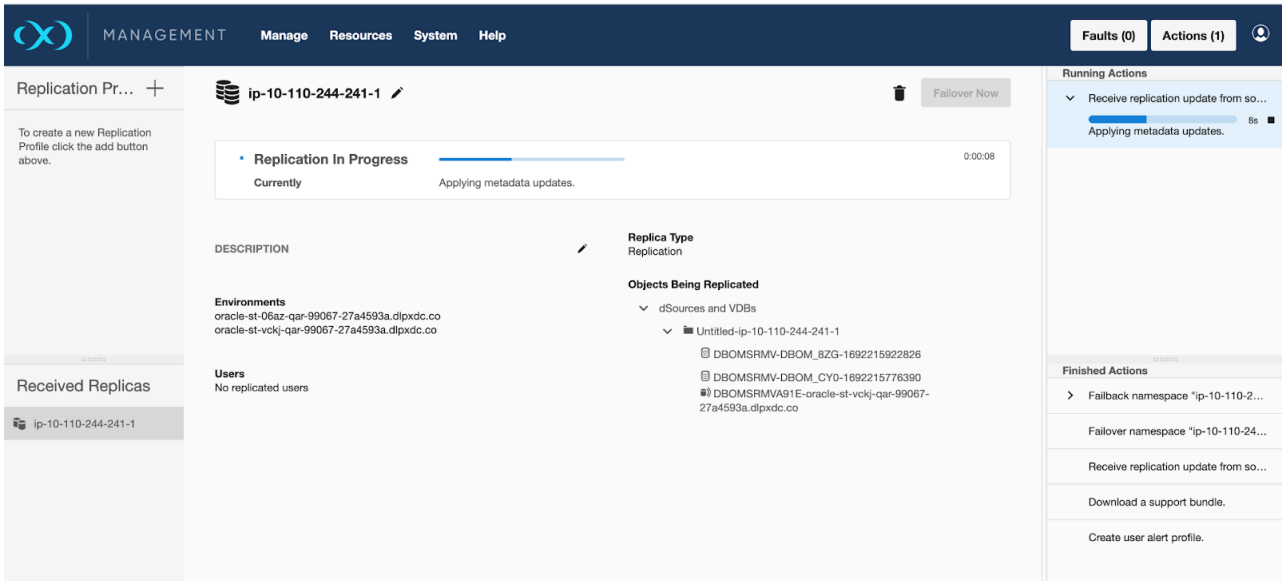
On failback, all newly provisioned child datasets will be destroyed. All new snapshots and refreshed time-flows will be deleted. The target-replication engine will thus be brought back to the state prior to failover by moving the objects into a read-only replica namespace so that the replication-source engine can continue replicating to the replica namespace.

No changes are made to the replication-source engine during failback. No changes from the replication target are propagated back to the replication source. Thus failback can be thought of as an “undo failover” operation.


MANAGEMENT
Manage
Resources
System
Help
Faults (0)
Actions (0)
Replication Pr...
To create a new Replication Profile click the add button above.
Received Replicas
ip-10-110-244-241-1
ip-10-110-244-241-1
Failover
Commit Failover
Failback
Renamed GROUP from "Untitled" to "Untitled-ip-10-110-244-241-1"
Renamed POLICY_RETENTION from "Default Retention" to "Default Retention-ip-10-110-244-241-1-11"
Renamed POLICY_SNAPSHOT from "Default Snapshot" to "Default Snapshot-ip-10-110-244-241-1-12"
Renamed POLICY_SYNC from "Default SnapSync" to "Default SnapSync-ip-10-110-244-241-1-13"
Consolidated POLICY_REFRESH "None:RefreshPolicy" with "None:RefreshPolicy"
DESCRIPTION
Replica Type
Replication
Objects Being Replicated
No replicated objects
Environments
oracle-st-06az-qar-99067-27a4593a.dlpxdc.co
oracle-st-vckj-qar-99067-27a4593a.dlpxdc.co
Users
No replicated users
Running Actions
Finished Actions
Failover namespace "ip-10-110-24...
Receive replication update from so...
Download a support bundle.
Create user alert profile.
Update user "admin".
Failback 'ip-10-110-244-241-1' Now?
A failback replica was created during failover to enable the ability to restore data back to original state before the failover attempt. Restoring the failback replica will overwrite your failover of 'ip-10-110-244-241-1'.
Are you sure you want to failback 'ip-10-110-244-241-1'?
Cancel
Failback Now
Failover Now
Last Replication Successful
0:00:48
Last Successful Replication Aug 17, 2023 11:38:20 PM (13 minutes ago)
dSources and VDBs
Untitled-ip-10-110-244-241-1
DBOMSRMV-DBOM_8ZG-1692215922826
DBOMSRMV-DBOM_CY0-1692215776390
DBOMSRMVA91E-oracle-st-vckj-qar-99067-27a4593a.dlpxdc.co
Failback namespace "ip-10-110-2...

Full restoration of the **Received Replica** namespace requires a successful replication-receive from the replication-source engine after performing the failback operation. It is a best practice to do this right after a failback.

The replication profile on the replication-source engine needs to be preserved for receiving updates to the *Received Replica* namespace.



Commit failover

If a failback is unnecessary, the test failover can be committed, making the failover permanent. This is similar to performing a failover with the enableFailback option off.

Provisioning from replicated data sources or VDBs

This topic describes how to provision from a replicated dSource or VDB. The process for provisioning from replicated objects is the same as the typical VDB provisioning process except for the need to first select the namespace containing the replicated object.

Prerequisites

You must have done the following:

- replicated a dSource or a VDB to the target host, as described in Replication overview
- added a compatible target environment on the target host as described in Provisioning Oracle VDBs: an overview
- installed on the target host any App Data plugin which the replicated objects depend on

Procedure

1. Login to the **Delphix Management** application for the target host.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **replica** that contains the dSource or VDB to be provisioned.
5. The provisioning process is now identical to the process for provisioning standard objects.If the dSource or VDB belongs to a plugin, the plugin must exist on the target engine to be successfully provisioned. The target's plugin's version must also be equal to or higher than the source's plugin's version. For more information see the Delphix engine plugin management topic.

Post-requisites

Once the provisioning job has started, the user interface will automatically display the new VDB in the live system.

Replication user interface

The replication user interface helps users manage replication on both the source and the target. Replication consists of a profile-replica pair. You can view and edit the replication profile on the source engine, and view the replica on the target engine.

Sources for replication

The **Replication Profiles** section provides you with several configuration options for your replication profiles. This makes it possible to replicate objects from a single source to multiple targets. Each profile defines the set of data objects and the associated configuration between a single source and target.

Targets for replication

The **Received Replicas** section shows the set of all objects in the replica and allows you to initiate failover.

Canceling a replication job

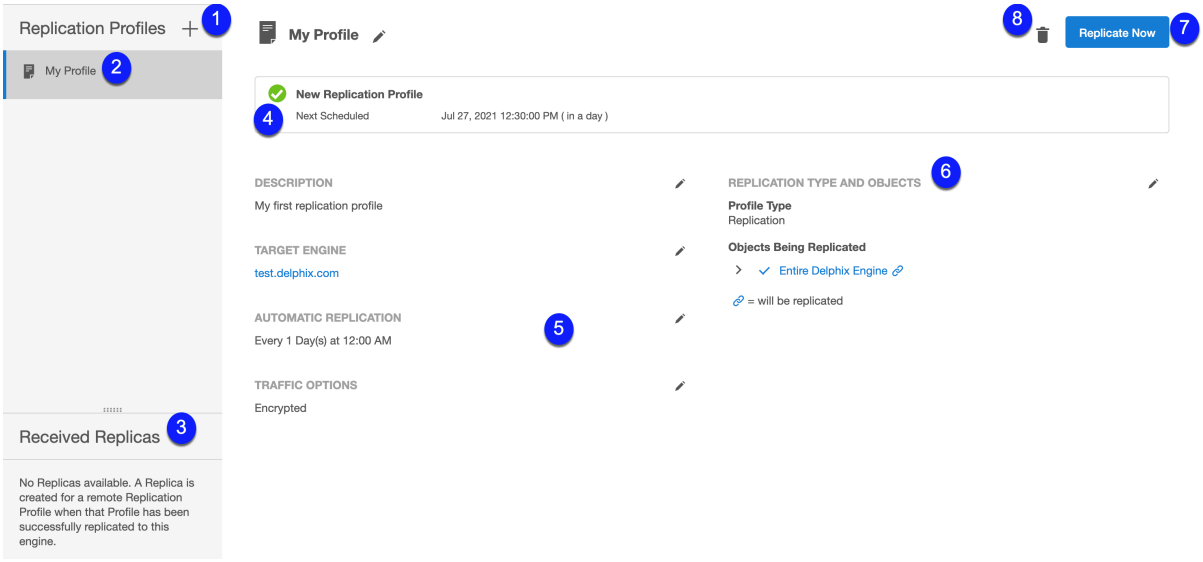
You must cancel the replication job at the source engine. Canceling the replication job at the target engine results in the re-starting of the receiving job at the target engine. If the source engine is down, then you can cancel the replication job at the target engine. If you re-start the source engine where the replication job was previously running, the source engine will attempt to restart the replication send job.

Replication UI

The Replication UI consists of the following sections.

Replication profiles

The screenshot and descriptions below illustrate the capabilities in the **Replication profiles** section.



1. **Create profile button**
   It allows you to configure a Replication Profile.
2. **Replication profiles list**
   Provides a list of existing replication profiles. Click a profile in this list to view its details.

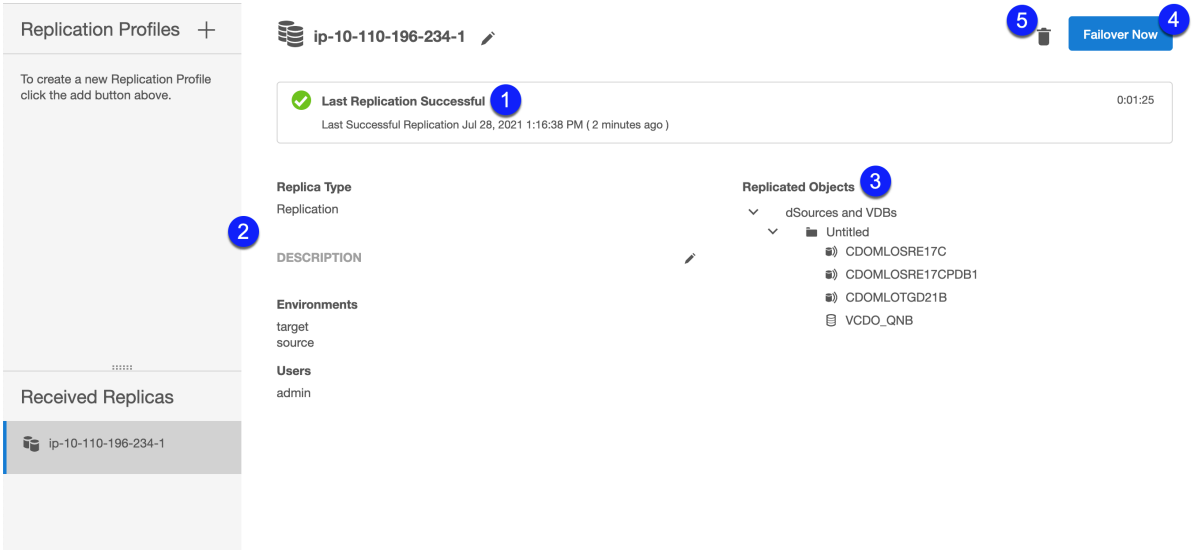
3. **Received replicas list**
   Provides a list of all existing replicas on this Delphix Engine. Click a replica in this list to view its details.
4. **Status box**
   Shows the replication status of the selected profile. This includes the result of the most recent or current replication event and statistics for the replication run (i.e. data transferred, duration, average throughput, etc.).

	This icon appears while a replication job is in progress.
	This icon appears after a successful replication job.
	This icon appears when a replication job has failed.
	This icon appears when a replication job was canceled.

5. **Configuration options**
   Additional configuration options for the selected replication profile.
   Description: Free text field for a profile description.
   Target Engine: The Delphix engine on the receiving end of this replication pair.
   Automatic Replication: If enabled, shows the frequency and time that regular replication will be run.
   Traffic Options: Summarizes the traffic options with which this profile has been configured.
6. **Objection selection tree**
   Shows all of the objects, such as groups, dSources, VDBs, and Self-service (JetStream) data layouts, that you have selected for replication in this replication profile. Selecting **Entire Delphix Engine** will cause all objects on the engine to be replicated, and thus the tree is collapsed.
7. **Replicate now button**
   Begins the replication process
8. **Delete button**
   Allows you to delete the current profile.

Received replicas

The screenshot and descriptions below provide more details of the functionality of this section.



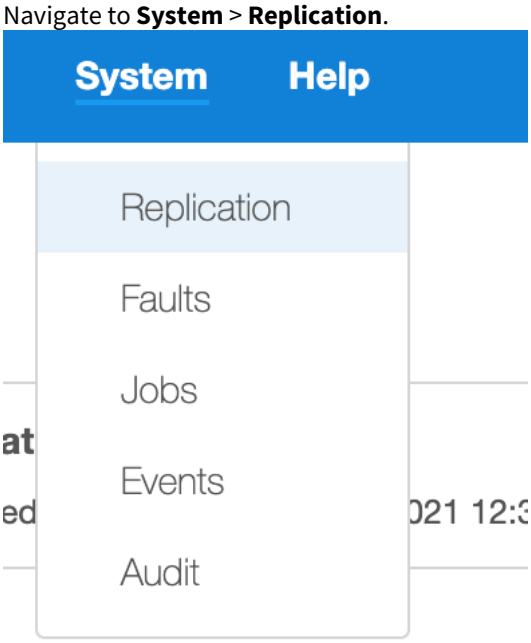
1. **Status box**
   Similar to the **Replication profile** status box, this shows the most up-to-date status information for the replica on the target

2. **Replicated environments**
   Replicating a dSource or VDB will automatically replicate any environments associated with those objects. For more information, see Replication overview
3. **Replicated objects tree**
   A read-only view of the objects in this replica.
4. **Failover button**
   Initiates a failover for this replica
5. **Delete button**
   Deletes this Replica on the target. This does not have an effect on the corresponding profile on the source engine.

Create new replication profile wizard

Perform the following steps to create a new replication profile.

1. Login to the **Delphix Management** application.
2. Navigate to **System** > **Replication**.



3. Next to **Replication Profiles**, Click the **plus** icon. The **Add Replication Profile** wizard screen appears.
4. Enter the **Name** of the replication profile and an optional **Description**, and click **Next**.
5. To specify the **Object**, select the **Profile Type** as **Replication Profile** and select from the **Objects Being Replicated** to a target engine. Click **Next**.
   Note:
   Selected Objects
   a. Some selected objects may have dependencies – other objects that will be pulled into replication because they share data. Objects that will be replicated are confirmed with a blue chain link icon.
      i. Note that this is not guaranteed to be the full set of dependent objects, but rather is the best guess. The full set of objects and their dependents will be calculated at the time of replication.
   b. When selecting objects, you can select the entire server (Entire Delphix Engine) or a set of groups, dSources, VDBs, and Jet Stream data layouts.
   c. When replicating a group, all dSources and VDBs currently in the group, or added to the group at a later time, will be included.

   d. If you select a Delphix Self-Service data template, all data containers created from that template will be included. Likewise, if you select a data container, its parent data template will be included.
   e. If you select the entire server, all groups and Delphix Self-Service objects will be included.
   f. Regardless of whether you select a VDB individually or as part of a group, the parent dSource or VDB (and any parents in its lineage) are automatically included. This is required because VDBs share data with their parent object. In addition, any environments containing database instances used as part of a replicated dSource or VDB are included as well.
   g. When replicating individual VDBs, only those database instances and repositories required to represent the replicated VDBs are included. Other database instances that may be part of the environment, such as those for other VDBs, are not included.
   h. Non-data objects (Delphix users, roles, permissions, authorizations, policies, database configuration templates) that are associated with selected objects will be automatically included during replication. They will not be shown on the selection interface.
6. For **Target Engine**, enter the **hostname** or **IP address** for the target Delphix Engine. Enter the **username** and **password** of a user who has Delphix Admin-level credentials on the target Delphix Engine. If the username or password changes on the target Delphix Engine, you must update these settings on the source Delphix Engine. Click **Next**.
7. By default, automatic replication is disabled and you must trigger replication updates manually. To **Schedule** automatic replication, click the **Enabled** checkbox.
   In the **Automatic Replication** field, enter the frequency and starting time for replication updates to the target Delphix Engine. Once you have entered and saved your replication settings, you will also see an option to trigger replication immediately with the **Replicate Now** button.
   Under **Traffic Options**, select whether you want to **Encrypt** traffic or **Limit bandwidth** during replication updates. Click **Next**.
8. The final summary tab will enable you to review your configurations. Click the **Back** button to go back and to change any of the configurations or click **Submit** to complete the wizard.

Viewing and editing an existing replication profile

Perform the following steps to view and edit existing replicas.

1. Login to the **Delphix Management** application.
2. Navigate to **System** > **Replication**.
3. Under the **Received Replicas** section, select a replica.
4. You can edit the **Name** and **Description** fields. All other fields are view-only.

Perform the following steps to edit the fields.

1. Click the **pencil** icon next to the corresponding field or group of fields to edit the fields.
2. To save the edits and/or selections, click the **Checkmark** icon.
3. To cancel the edits and/or selections, click the **cross** icon.

Selective data distribution

These topics describe concepts and procedures for using Selective Data Distribution (SDD) to replicate data from one Delphix Engine to another.

Support matrix for selective data distribution

SDD is supported across all environments supported by each data source. Refer to the support matrix for each data source below.

- Oracle Matrix
- SQL Server Matrix
- SAP ASE Matrix
- Db2 Matrix
- EBS Matrix
- HANA Matrix
- PostgreSQL Matrix

Selective data distribution overview

The selective data distribution technology permits the distribution of masked data between Delphix Engines without bringing over the unmasked parent source. These engines must be the identical versions of the Delphix Data as a Service Engine with the Delphix Masking Engine. Otherwise, they can be asymmetric in terms of engine configuration. You can provision VDBs from distributed masked objects, allowing for the geographical distribution of data and remote provisioning.

You can run selective data distribution ad hoc, but it is typically run according to a predefined schedule. After the initial update, each subsequent update sends only the changes incurred since the previous update. Selective data distribution does not provide synchronous semantics, meaning that the data distributed to the target is only as current as of the most recent update.

Selective data distribution features

As virtual appliances, you can backup, restore, replicate, and migrate data objects between Delphix Engines using features of VMWare and the underlying storage infrastructure. In addition to the replication capabilities provided by this infrastructure, selective data distribution permits the distribution of masked data between Delphix Engines. The sources received on a target Delphix Engine do not include the original parent source, thereby making the original source inaccessible from the target.

Selective data distribution is configured on the source Delphix Engine. It first copies a subset of masked VDBs to a target Delphix Engine, then sends incremental updates either manually or according to a schedule. As illustrated below, sensitive data from the Production Data Center is brought into Delphix as a dSource. The Masking Engine then masks the dSource data as a VDB. Synchronously, DxFS redacts the sensitive data within that dSource before sending it across into the Non-Production datacenter using Delphix Replication. Using SDD capability, sensitive data is never exposed because it is protected both at the dSource and VDB layers.

You can use replicated masked VDBs to provision new VDBs on the target Delphix Engine. The provisioned VDBs contain the data in their masked parent and are therefore also considered masked. You can refresh these VDBs to snapshots sent as part of an incremental replication update, as long as you do not destroy the parent object on the replication source. For more information, see Provisioning from a Replicated Data Sources or VDBs.

During replication, replicated masked VDBs are maintained in an alternate replica and are not active on the target side. The failover of a selective data distribution replica is not supported.

Selective data distribution details

When you select masked objects for selective data distribution, the engine will automatically include any dependencies, such as environments, associated with the VDB. The parent dSource and any parent VDBs are not included automatically. The data associated with parent objects are selectively included for disk space efficiency, but data in the parent dSource and VDBs that the masked VDB does not need are excluded.

During replication, the Delphix Engine will negotiate an SSL connection with its server peer to use SSL_RSA_WITH_RC4_128_MD5 as the cipher suite, and TLSv1 as the protocol.

Only database objects and their dependencies are copied as part of a selective data distribution operation, including:

- Masked VDBs
- Environments
- Environment configuration (users, database instances, and installations)

The following objects are not copied as part of a selective data distribution operation:

- Parent dSources of masked VDBs - The storage blocks for non-sensitive dSource data are sent from the source to the target replication host. This storage is displayed under Held Space, for more information see An Overview of Held Space.
- Groups of the parent dSources
- Users and roles
- Policies
- VDB (init.ora) configuration templates
- Events and faults
- Job history
- System services settings, such as SMTP

Resumable selective data distribution

A single selective data distribution instance can fail for a number of environmental and internal reasons. However, using the Resume feature, you can restart selective data distribution from an intermediate point; no data is retransmitted. Selective data distribution is resumable across machine reboot, stack restart, and network partitions. The resumable replication feature is fully automated and does not require or allow any user intervention.

For example, you can resume a large, time-consuming initial distribution or incremental update after it is interrupted. Suppose a selective data distribution profile has already been configured from a source to a target. A large, full send from the source begins that is expected to take weeks to complete. Halfway through, a power outage at the data center that houses the source causes the source machine to go down and only come back up after a few hours. On startup, the source will detect that a selective data distribution was ongoing, automatically re-contact the target, and resume the distribution where it left off. In the user interface (UI) on the source, the same selective data distribution send job will appear as active and continue to update its progress. However, in the UI of the target, a new distribution receives job will appear, although it will track its progress as a percentage of the entire replication.

Selective data distribution will not resume after failures that leave the source and target connected. For example, if a storage failure on the target, such as an out-of-space error, causes a distribution to fail, then the source and target remain connected. As a result, the Engine will discard state data associated with the failed Replication operation.

Selective data distribution restrictions

- Only masked VDBs can be added to a selective data distribution spec. You cannot add dSources, groups, or the entire domain.
- Only masked VDBs with a Snapshot Policy of None should be added to a selective data distribution spec.
- Unmasked VDBs cannot be added to a selective data distribution spec.
- VDBs that undergo selective data distribution and their children cannot be selectively redistributed to another target.
- You cannot go to the target engine and create a selective data distribution spec that includes VDBs that are present because of selective data distribution. However, you can replicate this data using a traditional replication spec.

Best practices for using Selective Data Distribution are described in the Selective Data Distribution Best Practices knowledge base article.

Selective data distribution supported platforms

Selective Data Distribution supports the following platforms:

- Oracle
- Microsoft SQL Server
- SAP ASE (Sybase)

- Db2

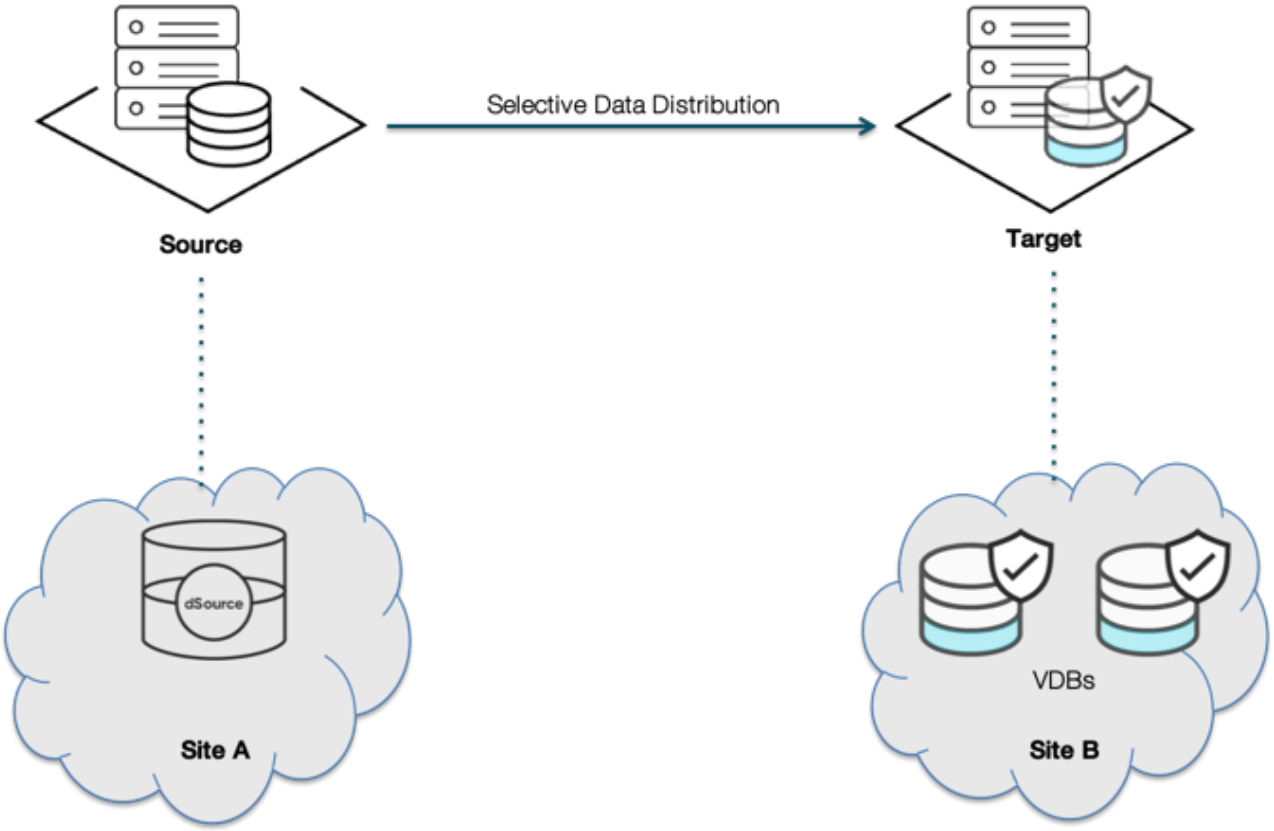
Selective data distribution use cases

Selective data distribution is a flexible tool that allows you to move dSources and virtual databases (VDBs) between Delphix Engines. Selective data distribution relies on the replication infrastructure to transmit masked data, as such selectively distributed data has much of the same capabilities as replicated data. These topics describe several different use cases for selective data distribution.

Geographically distributed development

The Delphix Engine allows you to provision VDBs from distributed dSources and VDBs, as described in Provisioning from Replicated Data Sources or VDBs. This allows dSources that are linked in a single central location to be geographically distributed so that developers can provision VDBs remotely without having to sync from the source database in multiple locations. Selective distribution ensures that masked VDBs are sent to the target without transmitting the original unmasked source. As such, the original unmasked source will not be accessible on the target.

Selective data distribution does not support failover. In this environment, the link between the source and the target is never broken, except when remote VDBs need to be preserved. You can refresh remote VDBs as long as the parent objects continue to exist on the source. If they are deleted, then remote VDBs will continue to function but cannot be refreshed.



Because there is no failover, this technology can support more complex topologies such as 1-to-many and many-to-1. Chained distribution (replicating from Site A -> Site B -> Site C) is not supported.

Best practices

For geographical distribution, follow these best practices:

- Because each replication stream induces load on the source system:
    - Minimize the number of simultaneous replication updates
    - If possible, avoid heavy VDB workloads on the source
- On the target, the provision only from sources that are effectively permanent. If a source is deleted remote VDBs can no longer be refreshed.
- Provision additional storage capacity on the target
    - Remotely provisioned VDBs can consume shared storage on the target even when the parent is deleted on the source

Migration

You cannot use selective data distribution for data migration. Full replication is needed for data migration.

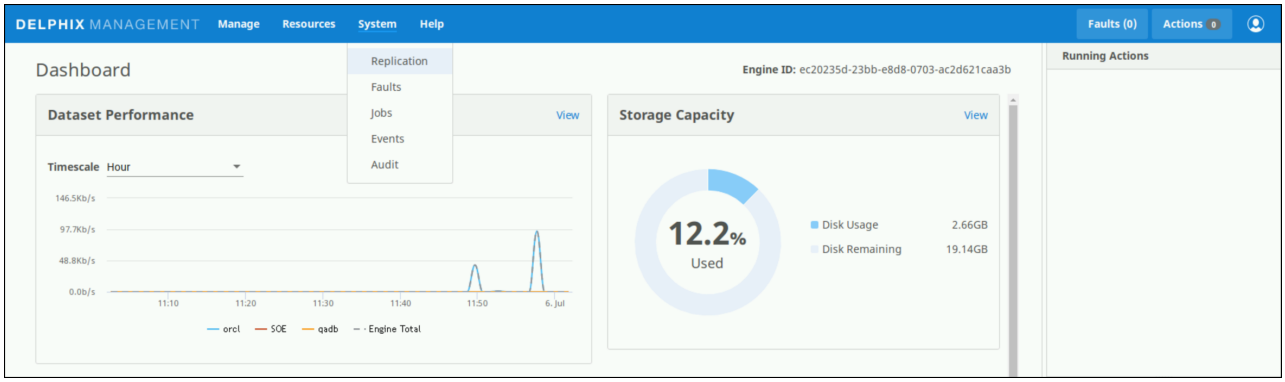
Selective data distribution user interface

Sources for selective data distribution

The **Replication Profile** section continues to handle Replication functions, but also allows for the configuration of **Selective Data Distribution (SDD)** profiles. This makes it possible to distribute objects from a single source to multiple targets. Each profile defines the set of data objects and the associated configuration between a single source and target.

Replication/selective data distribution section

This section is used to illustrate the capabilities in the **SDD Profiles** section. From the Delphix Management portal, navigate to **System** > **Replication**.



Select the plus icon to add a profile, this will open a new window for configurations. Below the **Replication Profiles** header and the plus icon is a list of existing Replication Profiles and SDD profiles. Select a profile in this list to view its details.

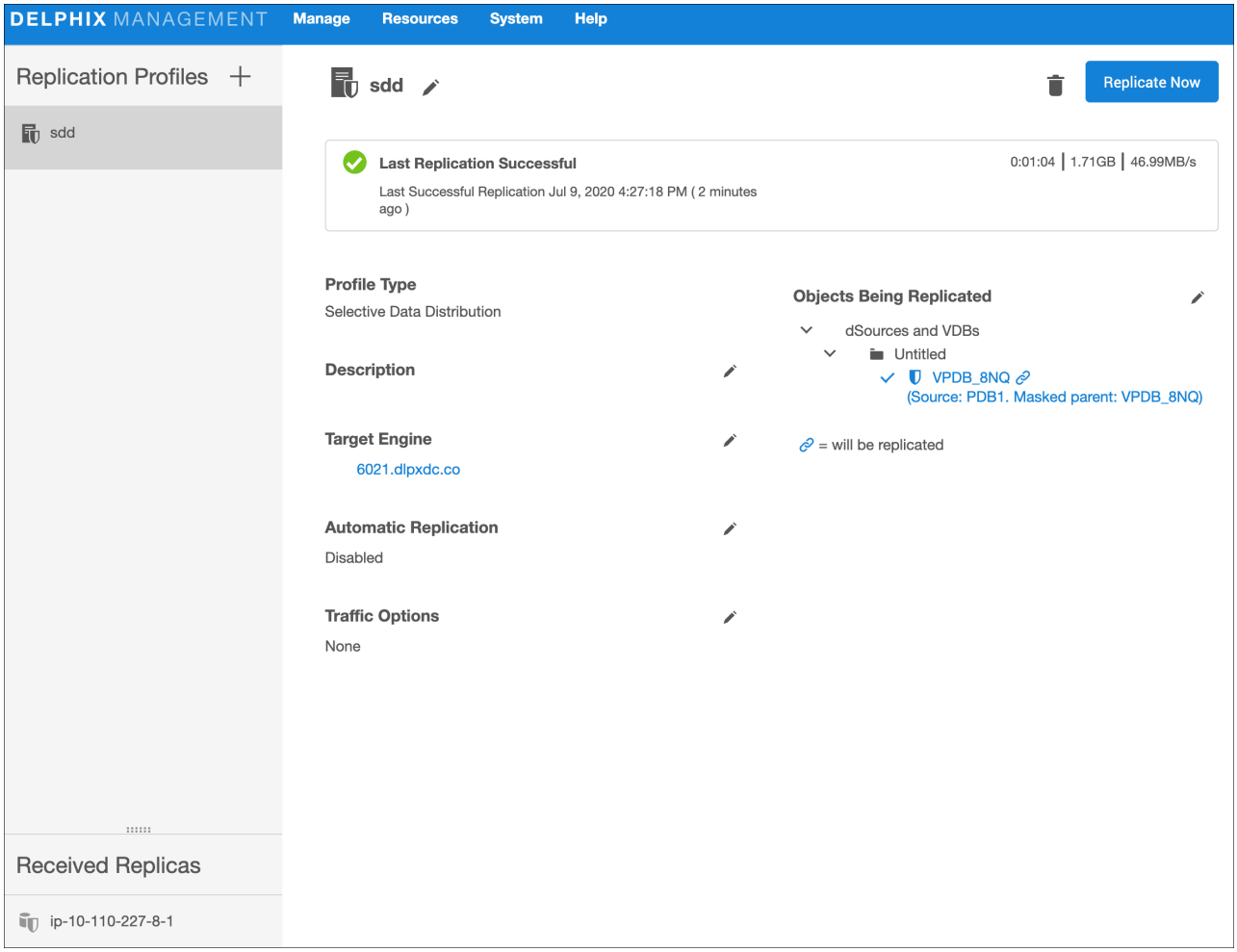
The **Received Replicas List** shows all replicas received to this Delphix Engine, including normal Replication and SDD replicas. Click a **replica** in this list to view its details.

A **Status Box** appears when a profile is selected from the list that shows the distribution status of the selected profile. This includes the result of the most recent or current distribution event and statistics for the distribution run, such as data transferred, duration, and average throughput. The **Profile Type** shows the type of the selected profile or replica.

There are several configuration options available for the selected profile or replica listed below the Profile Type. These sections, in addition to the profile name itself, can be modified by selecting the pencil icon.

- Description – Free text description of the profile
- Target Engine – The Delphix Engine on the receiving end of this data distribution pair
- Automatic Replication – If enabled, shows the frequency and time that regular distribution will be run
- Traffic Options – Summarizes the traffic options with which this profile has been configured

The **Objects Being Replicated** shows all of the masked objects selected for distribution in this SDD profile. The **Replicate Now** button initiates the distribution process and the **bin icon** to the left allows for current profile deletion.



Create new profile

1. In the left-hand navigation section, click the **+** icon.
2. Enter the **name** of the SDD profile and an optional **description**.
3. For **Type**, select **Selective Data Distribution**.
4. For **Target Engine**, enter the **hostname** or **IP address** for the target Delphix Engine.
5. Enter the **username** and **password** of a user who has Delphix Admin-level credentials on the target Delphix Engine. If the username and password change on the target Delphix Engine, these settings must be updated on the source Delphix Engine.
6. Automatic replication is disabled by default, which means the distribution updates must be triggered manually via the **Replicate Now** button. To enable automatic distribution, click the **Enabled** checkbox.
7. In the **Automatic Replication** field (if enabled), enter the **Frequency** and **Starting Time** for distribution updates to the target Delphix Engine.
   Note: Automatic replication uses Quartz for scheduling. Starting with Delphix version 4.2, the Quartz-formatted string is editable via the **Advanced** option. An example **Cron String** would be seen as: `0 0 0 * * ? *`
8. Under **Traffic Options**, select to **Encrypt** traffic or **Limit bandwidth** during distribution updates.
9. In the right-hand column under **Objects Being Replicated**, select the **boxes** next to the objects to distribute. Only masked VDBs can be used and multiple masked VDBs cannot be added to the same profile.
10. Click **Create Profile** to submit the new profile. This saves the SDD profile details. Leaving the **Create** page prior to submitting the profile will result in the draft being discarded.

Configuring selective data distribution

This topic describes how to configure Selective Data Distribution between Delphix Engines. Selective data distribution relies on the replication infrastructure to transmit masked data, as such configuring selective data distribution is similar to configuring replication.

Prerequisites

- The replication target can be on the same or newer version than the replication source. Starting 6.0.10.0, replication operation can not go beyond engine versions released more than 12 months apart. For more information, see Replication Overview.
- The target Delphix Engine must be reachable from the source engine.
- The target Delphix Engine must have sufficient free storage to receive the replicated data.
- The user must have administrative privileges on the source and the target engines.

Configuring the network

Replication and selective data distribution operate using a private network protocol between two Delphix Engines. Apart from standard network considerations for performance, no additional configuration is required for replication. Replication and selective data distribution can run over dedicated networks by configuring routing to direct traffic destined for the target IP address over a specific interface. The selective data distribution process can recover from transient network outages, but extended outages may cause the process to start from the previous update.

The selective data distribution network protocol uses TCP port 8415. If there is a firewall between the source and target that is blocking this port, then there are two possible solutions:

- Enable port 8415 on the firewall in order to allow connections to this port from the source to the target.
- Selective data distribution can connect through a SOCKS proxy if one exists. Configure the SOCKS proxy address and port by connecting to the command-line interface (CLI) as a system administrator and navigating to "service proxy" to update the socks configuration. Example:

```
dlpx-engine> service proxy
dlpx-engine service proxy> update
dlpx-engine service proxy update *> set socks.enabled=true
dlpx-engine service proxy update *> set socks.host=10.2.3.4
dlpx-engine service proxy update *> set socks.username=someuser
dlpx-engine service proxy update *> set socks.password=somepassword
dlpx-engine service proxy update *> commit
dlpx-engine service proxy> get
    type: ProxyService
    https:
        type: ProxyConfiguration
        enabled: false
        host: (unset)
        password: (unset)
        port: 8080
        username: (unset)
    socks:
        type: ProxyConfiguration
        enabled: true
```

```
        host: 10.2.3.4
        password: ********
        port: 1080
        username: someuser
```

Note that SOCKS port 1080 is used by default, but you can override it.

Configuring the source Delphix engine

1. On the source Delphix Engine, click **System**.
2. Select **Replication**.
3. In the left-hand navigation section, click **Create Profile**.
4. Enter the **name** of the replication profile and an optional **description**.
5. Select type **Selective Data Distribution**.
6. For **Target Engine**, enter the **hostname** or **IP address** for the target Delphix Engine.
7. Enter the **username** and **password** of a user who has Delphix Admin-level credentials on the target Delphix Engine. If the username and password change on the target Delphix Engine, you must update these settings on the source Delphix Engine.
8. By default, automatic replication is disabled, meaning that you must trigger replication updates manually. To enable automatic replication, click the **Enabled** checkbox.
9. In the **Automatic Replication** field, enter the **Frequency** and **Starting Time** for replication updates to the target Delphix Engine. Once you have entered and saved your replication settings, you will also see an option to trigger replication immediately with the **Replicate Now** button.
   Note:
   Quartz scheduling Automatic replication uses Quartz for scheduling. Starting with Delphix version 4.2, the Quartz-formatted string is editable via the **Advanced** option.
10. Under **Traffic Options**, select whether you want to **Encrypt** traffic or **Limit bandwidth** during replication updates. **Note**:
    Encrypting Traffic By default, replication streams are sent unencrypted. This provides maximum performance on a secure network. If the network is insecure, encryption can be enabled. Note that encrypting the replication stream will consume additional CPU resources and may limit the maximum bandwidth that can be achieved.
    Note:
    Limiting Bandwidth By default, replication will run at the maximum speed permitted by the underlying infrastructure. In some cases, particularly when a shared network is being used, replication can increase resource contention and may impact the performance of other operations. This option allows you to specify the maximum bandwidth that replication can consume.
11. In the right-hand column, under **Objects Being Replicated**, click the **checkboxes**next to the objects you want to replicate.
    Note:
    Selected Objects
    a. You can only select masked VDBs for selective data distribution.
    b. The parent dSource or VDB (and any parents in its lineage) are NOT automatically included. Some of the data from the parent may be included for disk space optimization. In addition, any environments containing database instances used as part of a replicated VDB are included as well.
    c. When replicating individual VDBs, only those database instances and repositories required to represent the replicated VDBs are included. Other database instances that may be part of the environment, such as those for other VDBs, are not included.
12. Click **Create Profile** to submit the new profile. This saves the replication profile details. If you leave the **Create** page prior to submitting the profile, the draft replication profile will be discarded.

Note:
Configuring Replication and Multiple Target Engines through the CLI

You can also configure replication on the Source Delphix Engine by using the replication spec in the command line interface. For more information, see the topics under CLI Cookbook: Replication.

Note:

Enabling Configuration of Multiple Replication Profile
Learn how to configure and use functionality for multiple replication profiles on the source using the replication profiles in the Replication User Interface.

Configuring the target Delphix engine

No additional configuration on the target is needed. Selectively distributed objects will appear in an alternate replica that mirrors the original object layout. To view these replicas:

1. Click **System**.
2. Select **Replication**.
3. Review items listed under **Received Replicas**.

Alternatively, you can view replicas under **namespace** in the command-line interface (CLI). All replicated objects are read-only. For more information about managing replicas, see Selective Data Distribution and Failover.

Multiple sources can replicate to the same target, allowing for the flexible geographical distribution of data. You can create and manage objects on the target server without affecting subsequent updates. However, you cannot use selective data distribution for disaster recovery.

Selective data distribution and failover

Selective data distribution recreates objects on the target system in a replica that preserves the object relationships and naming on the target server without interfering with active objects on the system. Objects within a replica are read-only and disabled. They cannot be failed over. However, you can use virtual databases (VDBs) within a replica as the source for provisioning new VDBs.

Replicas

A replica contains a set of replicated objects. These objects are read-only and disabled while replication is ongoing. To view replicated objects:

1. Click **System**.
2. Select **Replication**.
3. Under **Received Replicas**, select the replica. On this screen, you can browse the contents of replicas or delete individual replicas. As described in the Selective Data Distribution Overview, VDBs and environments are included within the replica.

You can also view replicated objects under **namespace** in the command-line interface (CLI).

Deleting a replica will sever the link with the replication source. Subsequent incremental updates will fail, requiring the source to re-establish replication.

Multiple replicas can exist on the system at the same time. Active objects can exist in the system alongside replicas without interfering with replication updates. VDBs within a replica can also be used as a source when provisioning. For more information, see Provisioning from Replicated Data Sources or VDBs.

Failover

Selective data distribution does not support the failover of replicas.

Delphix self-service

Delphix self-service

Delphix Self-Service eliminates data related bottlenecks by extending virtual data on demand to application development teams as a self-service.

Delphix self-service admin guide

Delphix self-service data user guide

Delphix self-service admin guide

Getting started with Delphix self-service

Welcome to Delphix Self-Service The Delphix Engine has greatly improved the speed at which end users can get the data that they need. While end-users reap the benefits, they do not typically interact with the Delphix Engine directly, nor are they n...

Updated on : 25 May 2023

Delphix self-service concepts

Data Sources A data source can be a database, an application, or a set of unstructured files. Engine Administrators configure the Delphix Engine to link to data sources, which pulls in the data of these sources. The Delphix Engine will periodical...

Updated on : 25 May 2023

Navigating the Delphix self-service admin interface

The following screenshots provide a roadmap for how to navigate the primary screens and places a user will go within the Delphix Self-Service application. The application includes screens such as the Management Overview , Data Platform Management ...

Updated on : 25 May 2023

Understanding data templates

Data Templates: An Overview A data template represents a collection of data sources that you can provision to a user. A data source can be a dSource, VDB, or vFiles. These sources can be used in multiple data templates. Once you have created a dat...

Updated on : 25 May 2023

Understanding how to manage data template details

The Data Template Details Page In the Overview page, under the Templates , click the data template's name . This will direct you to the templates details page. You can use this page to view and configure details of an individual data temp...

Updated on : 25 May 2023

Understanding data containers

Delphix Self-Service Data Container Overview Data containers are provisioned from data templates by administrators and assigned to a user. A data container represents a socket that is capable of making any data within the data template accessible. ...

Updated on : 25 May 2023

Delphix self-service data container activities

Configuring Data Containers in Delphix Self-Service A data container is comprised of a set of virtual databases (VDBs), where each VDB is a direct child of the dSource, VDB, or vFiles in the data template's data sources. Delphix Self-Service does...

Updated on : 25 May 2023

Using masked data sources with Delphix self-service

SDD Overview You can now replicate masked data in a VDB directly to a target Delphix Engine without transmitting the unmasked data in its parent source. This is called selective data distribution (SDD). Although you can run selecti...

Updated on : 25 May 2023

Understanding Delphix self-service user management

User Management Activities This document describes the process of creating a user and assigning that user to a data container. It also provides an overview of the User Details page. Creating a User Follow the same process when creating a new us...

Updated on : 25 May 2023

Working with multiple container owners

Delphix Self-Service administrators can designate multiple users as owners of a single data container. These users all share access to the same data container which means actions taken by one user will impact all users on the same data container. Fo...

Updated on : 25 May 2023

Understanding bookmarks

Bookmarks Overview Bookmarks are a way to mark and name a particular moment of data on a timeline. You can restore the active branch's timeline to the moment of data marked with a bookmark. You can also share bookmarks with other users, which all...

Updated on : 25 May 2023

Understanding Delphix self-service usage management

Usage Management Dashboard Overview Data templates are comprised of dSources, virtual databases (VDBs), and vFiles. These data sources are controlled by the standard policies configured in the Management application of the Delphix Engine. ...

Updated on : 25 May 2023

Getting started with delphix self-service

Welcome to Delphix self-service

The Delphix Engine has greatly improved the speed at which end users can get the data that they need. While end-users reap the benefits, they do not typically interact with the Delphix Engine directly, nor are they necessarily even aware that they are using it. End users most commonly file tickets for data management operations and wait for the tickets to be serviced by their IT organization. Delphix data management workflows allow database administrators (DBAs) to respond to these tickets much more quickly and reliably, but DBAs are often overloaded, and resolving high-priority issues takes precedence over requests from users. Requiring interactions between users and IT for every data operation is inefficient and can lead to unwanted delays.

The goal of Delphix Self-Service is to create a clear separation of IT infrastructure and data management. As with the current Delphix Engine, IT administrators and DBAs continue to control decisions about how resources such as virtual databases (VDBs) and vFiles are allocated. However, with Delphix Self-Service, administrators can also assign these resources directly to a user. A user has the ability to control what data their container presents, even though the details of the physical resources are hidden from them. This separation of roles empowers Delphix Self-Service users to get the data they need, when they need it while providing administrators with the controls to ensure resources are accounted for appropriately.

User roles and permissions

Delphix Self-Service has two types of users:

Admin user

Admin users have full access to all report data and can configure Delphix Self-Service, additionally, they can:

- use the Delphix Engine to add/delete users
- change tunable settings
- add/delete tags
- create and assign data templates and containers

Data user

Data users have access to production data provided in a data container. The data container provides these users with a playground in which to work with data using the **Self-Service Toolbar**.

Login

1. Access Delphix Self-Service by opening a web browser and using the **IP address** or **DNS qualified hostname**.
2. Login with the **Delphix Admin User ID** and **Password** provided for you.
3. Upon successful login, you should be able to click on your username at the top right corner and select a Self-service view from the dropdown menu. This will open the Delphix Self-Service interface.

Delphix self-service concepts

Data sources

A data source can be a database, an application, or a set of unstructured files. Engine Administrators configure the Delphix Engine to link to data sources, which pulls in the data of these sources. The Delphix Engine will periodically pull in new changes to the data, based on a specific policy. This, in turn, begins building a custom timeline for each data source. Additionally, the Delphix Engine can rapidly provision new data sources that are space-efficient copies, allowing users to work in parallel without impacting each other.

Data templates

Data templates are the backbone of data containers. They are created by the Engine Administrator and consist of the data sources that users need in order to manage their data playground and their testing and/or development environments. Data templates serve as the parent for a set of data containers that the administrator assigns to users. Additionally, data templates enforce the boundaries for how data is shared. Data can only be shared directly with other users whose containers were created from the same parent data template.

Data containers

A data container allows data users to access and manage their data in powerful ways. Their data can consist of application binaries, supporting information, and even the entire database(s) that underlie it.

A data container allows users to:

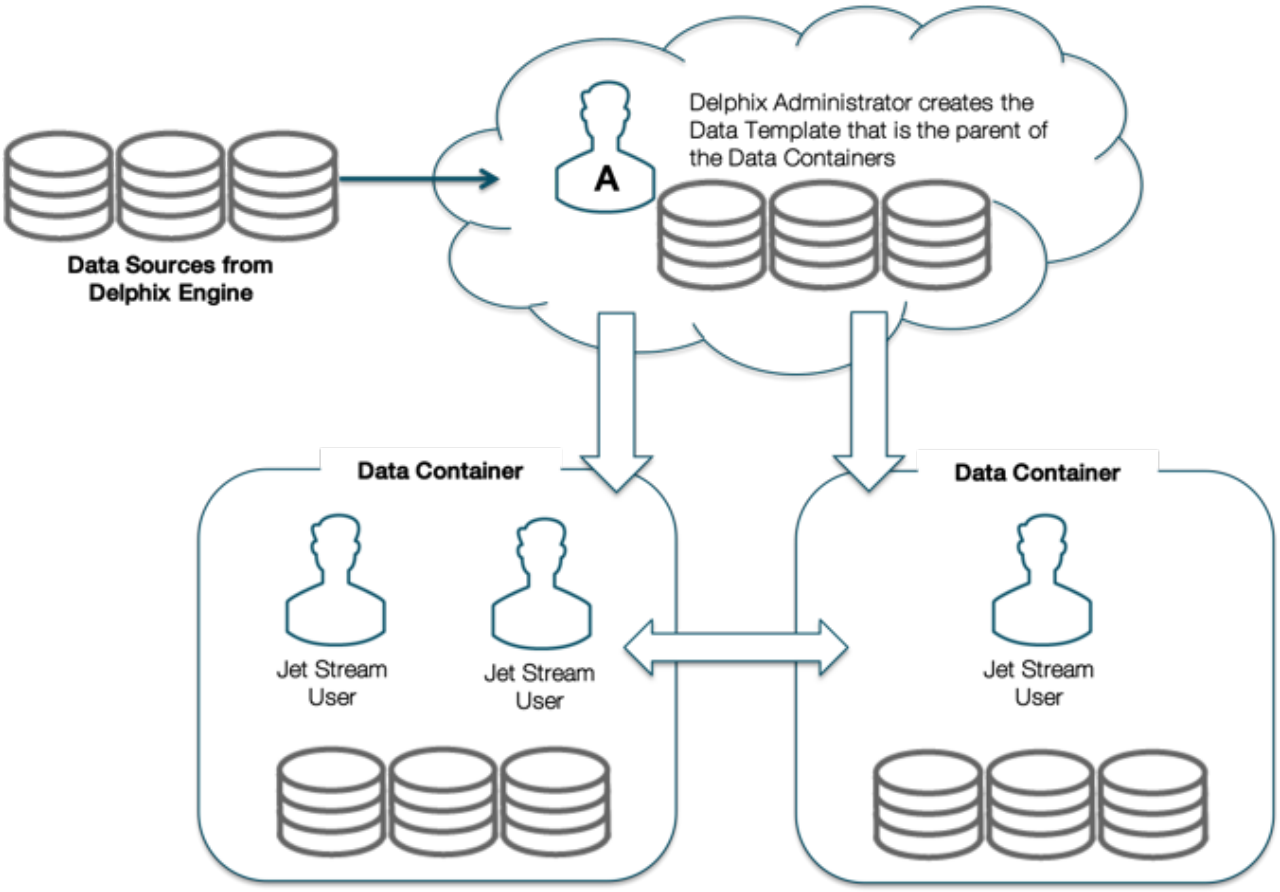
- Undo any changes to their application data in seconds or minutes
- Have immediate access to any version of their data over the course of their project
- Share their data with other people on their team, without needing to relinquish control of their own container
- Refresh their data from production data without waiting for an overworked DBA

A data container consists of one or more data sources, such as databases, application binaries, or other application data. The user controls the data made available by these data sources. Just like data sources in a template, changes that the user makes will be tracked, providing the user with their own data history.

The **Data Container Interface** lets users view the details and status of their data container and its associated data sources, as well as manipulating which data is in those sources. The **Data Container Interface** includes a section called the **Data Container Report Panel**, which displays details about each source, including the connection information needed to access it – for example, the java database connectivity (JDBC) string for a database. This connection of information is persistent and stable for the life of the data container, regardless of what data the resources are hosting.

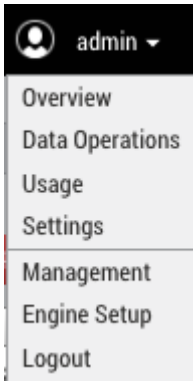
Delphix self-service data flow

The Delphix Self-Service data flow diagram below demonstrates how a Delphix Self-Service data user accesses data sources. Data sources are connected to a Delphix Engine, which is controlled by the Engine Administrator. The Engine Administrator will connect all data sources that developers and quality assurance (QA) teams need to a data template. This data template acts as a parent source to create the data containers that the administrator will assign to data users. Data sources flow from the Delphix Engine into a data template and downstream into a data container, where a data user or users will use the data sources to complete tasks. The data container acts as a self-contained testing environment and playground for the data user. Additionally, data users are able to set, bookmark, and share data points in their container with other data users of other data containers, as long as all the data containers were created from the same parent data template.


Data Sources from
Delphix Engine
A
Delphix Administrator creates the
Data Template that is the parent of
the Data Containers
Data Container
Jet Stream
User
Jet Stream
User
Data Container
Jet Stream
User

Navigating the Delphix self-service admin interface

The following screenshots provide a roadmap for how to navigate the primary screens and places a user will go within the Delphix Self-Service application. The application includes screens such as the **Management Overview**, **Data Platform Management**, **Users and Permissions**, and **Data User Management**.



Overview screen

The **Overview** screen is the homepage for Delphix Self-Service. To reach this page from the Management application as an admin user, select **Self-Service** in the user login drop-down.

Data management overview page

On this page, you can:

1. You can delete one or more templates at one time by selecting the checkboxes and clicking the **Delete** button.
    a. You cannot delete replicated templates or templates that have containers.
2. From this column, you can see which replica the template belongs to if it is a replicated template. You can sort by this column and see all templates from the same replica.
    a. This column lists replica names, not replication sources. You can view a list of replicas in the Management application at **System** > **Replication** > **Received Replicas**.
3. The number of containers associated with the template.
4. Shows the last time the template was updated.
5. Create a new data template.

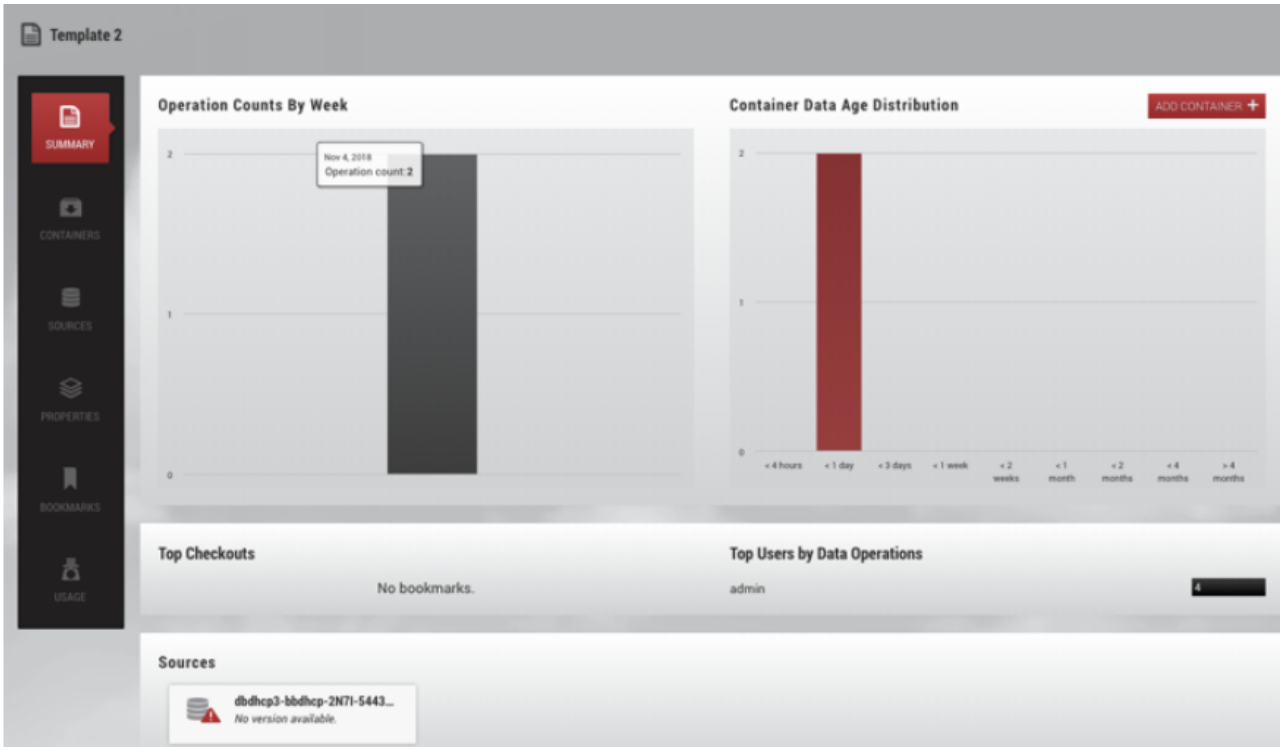
Users screen

By selecting the Users Button you can view users, whom you can then assign to data containers that you create from existing data templates.

1. List of all configured users. You can click on a user to see their details, such as what containers they are assigned to.
2. The number of containers assigned to the user.
3. The Role column indicates whether a user is a **Jet Stream Only** user or a **Delphix Admin**.
4. Lists the associated email address for the user.
5. Select this link to add a new user or manage users.

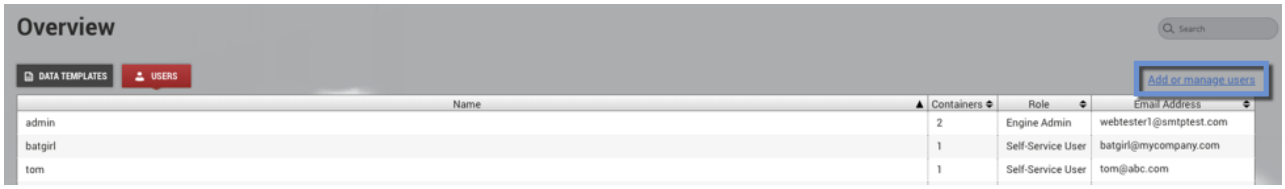
Data template management

The **Data Template Management** page contains a view panel of 6 tiles on the left-hand side of the screen. Each tile reports on a variety of useful information, such as user activity, data sources, data capacity, specific details about data containers, and data templates. They also help you navigate to areas where you can complete specific tasks, such as creating a new template or container, working with data timeflows, assigning users to containers, and bookmarking important points in time.

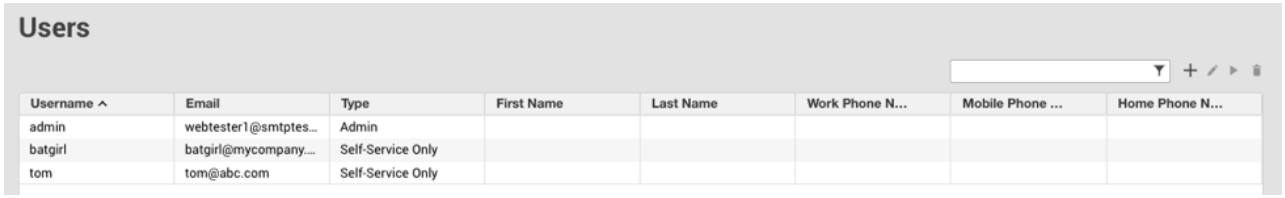


User roles in admin App

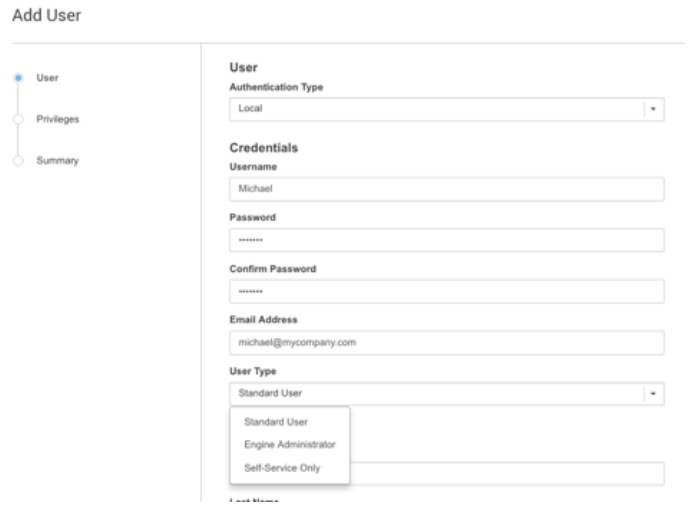
To add or manage user roles and permissions, select the **Add or manage users** link from the **Users Overview** screen.



The **Users** screen in the Management application will open. From this screen, you can add a new user, edit a user, suspend a user, or delete a user.



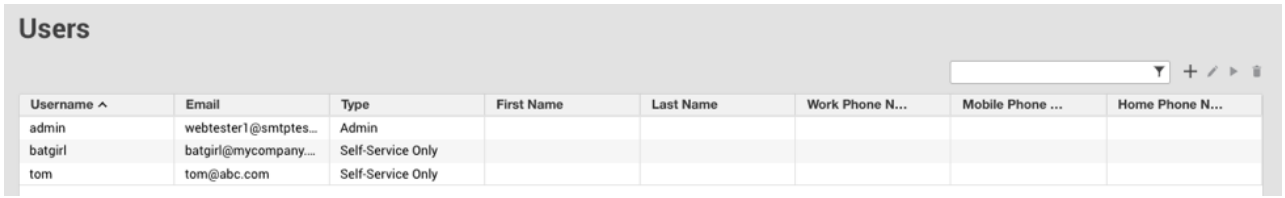
Selecting the **+** icon opens the **Add User** window. Use this window to add a user and select the user role and permissions.



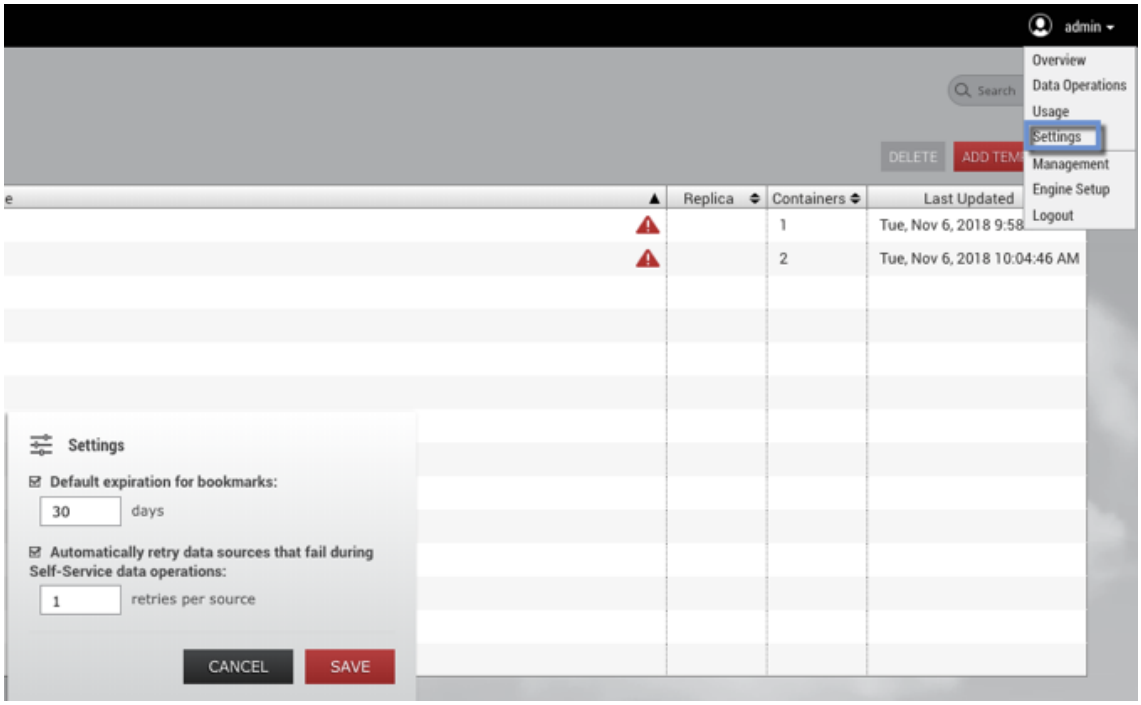
Data operations interface for Delphix administrators

The **Overview** page is the only interface to which Delphix Self-Service data users have access and with which they interact. The user interface is the environment in which a data user works with data in an assigned data container, using data sources from a data template.

As shown below, the **Data Operations** screen contains a list of available templates. Select a template to view details.



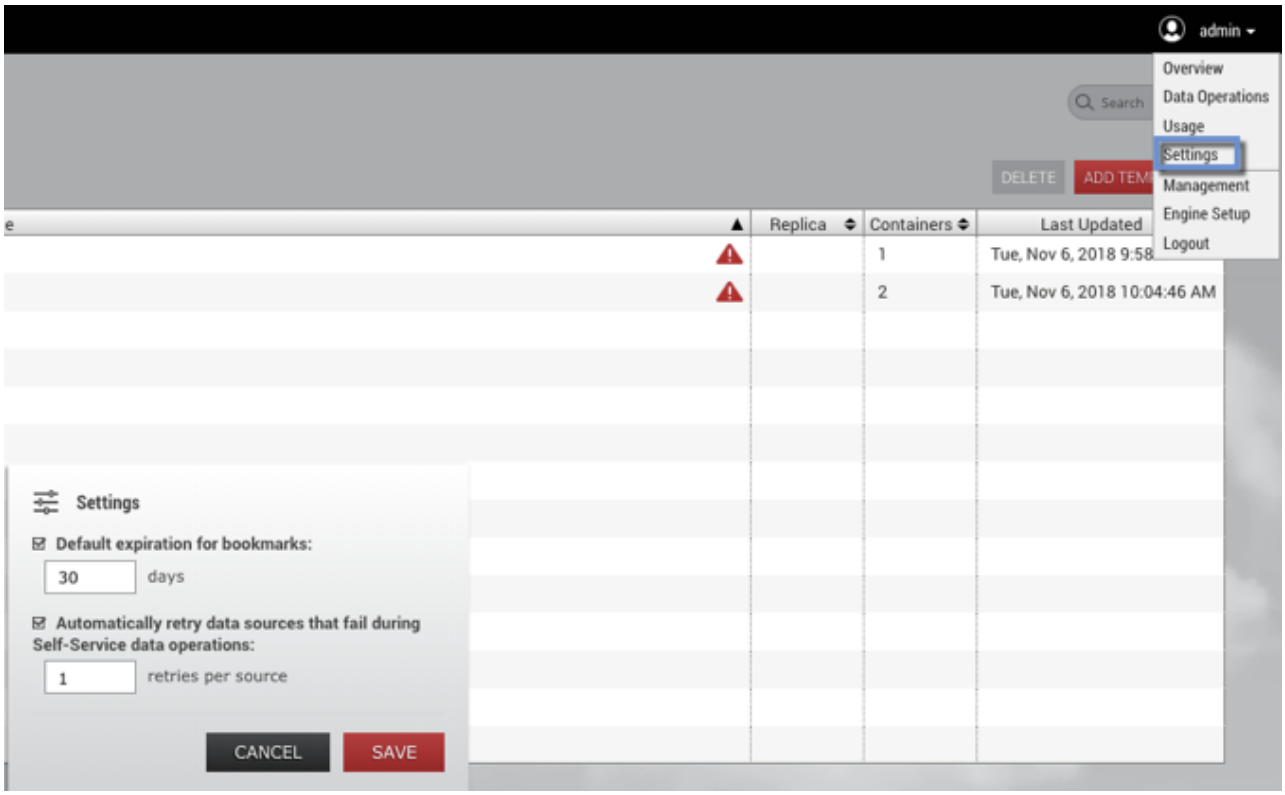
The following screen allows users to complete tasks using self-service operations.



For more details about how to use this interface, refer to the Delphix Self-Service Data User Guide.

Administrator settings

You can open the **Administrator Settings** dialog from the user drop-down menu. Here you can edit various settings that apply globally to all Delphix Self-Service users.



Default bookmark expiration

You can set a value that controls the default expiration time, in days, for Bookmarks. This setting only applies to new bookmarks that are created through the Delphix Self-Service application, not the CLI or API. Note that this only controls the default selection; users can still disable expiration or pick a different date for a specific bookmark if they wish. This setting is disabled by default.

Automatic retry for data operations

To make operations on data containers more robust, Delphix Self-Service supports automatically retrying failed sources during data operations. You can specify a maximum number of retry attempts so that if an operation fails on any individual data source within a data container, it will be automatically retried until it succeeds, or until the retry limit is reached.

Automatic retry applies to any Delphix Self-Service operation that changes the data in the data container, such as Refresh, Restore, Reset, or Create Branch. This setting can be especially useful in scenarios where there are a large number of sources in a data container, and some sources fail to update the first time. If the reason for the failures was intermittent, an automatic retry may allow the sources that failed to succeed, and the operation can still complete successfully. The default number of retry attempts is **1**.

To change the number of automatic retries:

1. Click the user icon in the upper right-hand corner of the screen.
2. From the drop-down menu, select **Settings**.
3. In the field **Automatically retry data sources...**, enter a new maximum; alternatively, use the arrows to increase or decrease the number.
4. Click **Save**.

Understanding data templates

Data templates: an overview

A data template represents a collection of data sources that you can provision to a user. A data source can be a dSource, VDB, or vFiles. These sources can be used in multiple data templates. Once you have created a data template, the set of data sources associated with it is fixed; you cannot add data sources to an existing template, nor can you remove data sources from it. In addition to data sources, you can define the set of metadata that is relevant for a given template – for example, notes, descriptions, names for sources that are relevant to an end-user, and other configuration details. Once you have created the template, it provides a stencil for provisioning data containers. This, in turn, enables users to have self-service access to a space-efficient copy of the data sources defined in the data template.

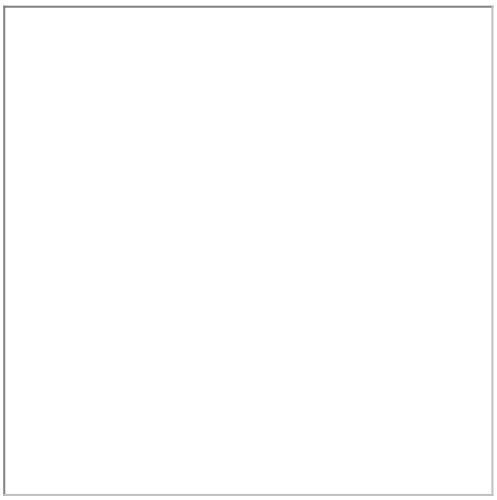
Data template activities

Data templates are managed by a Delphix admin. The admin can provision data containers from the data template and assign a data container to an end-user. The admin can also create bookmarks on the data template timeline in order to mark meaningful points in time.

When creating a data template, it is important to consider the set of users who will own data containers provisioned from it. In Delphix Self-Service, templates effectively define the boundaries of the data that users can share directly with each other. Only owners of data containers created from the same data template are able to share data using bookmarks.

Creating a Data template and adding data sources

A data template consists of an arbitrary set of dSources, virtual databases (VDBs), and vFiles. These are created and managed in the **Delphix Management** interface and can be used in Delphix Self-Service as data sources. You can use any data type supported by the Delphix Engine as a data source. For more information, refer to the Linking/ Provisioning documentation for the standard Delphix Engine. The following is an example of the many kinds of data sources you can use to create a data template.



When adding data sources to the data template, it is important to consider whether there are any dependencies between them. For example, do data operations need to begin with a VDB (database) source before the same operation occurs on vFiles (application binary)? Or can data operations be performed in parallel with each of the data sources? The data source dependencies are by default synced together in parallel during any data operation,

including starting the data container and its sources. When working with specific ordering constraints, such as with Oracle EBS, you can set up and configure the ordering sequence for each data source.

Procedure for creating a data template

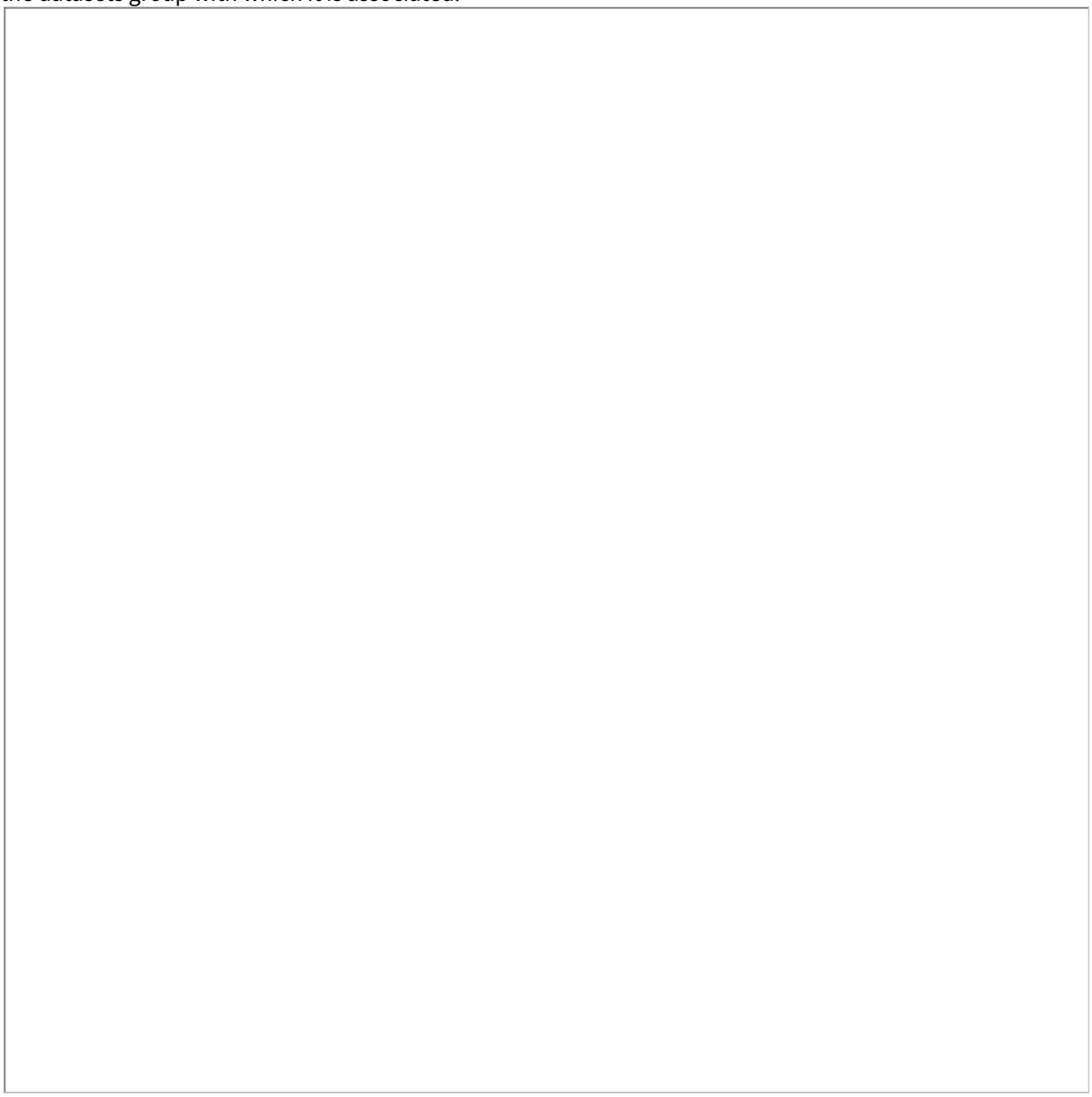
To create a data template:

1. From the drop-down menu in the upper right-hand corner of the Delphix Management application, select **Self-Service**.
2. On the **Overview** page, click **Add Template**.

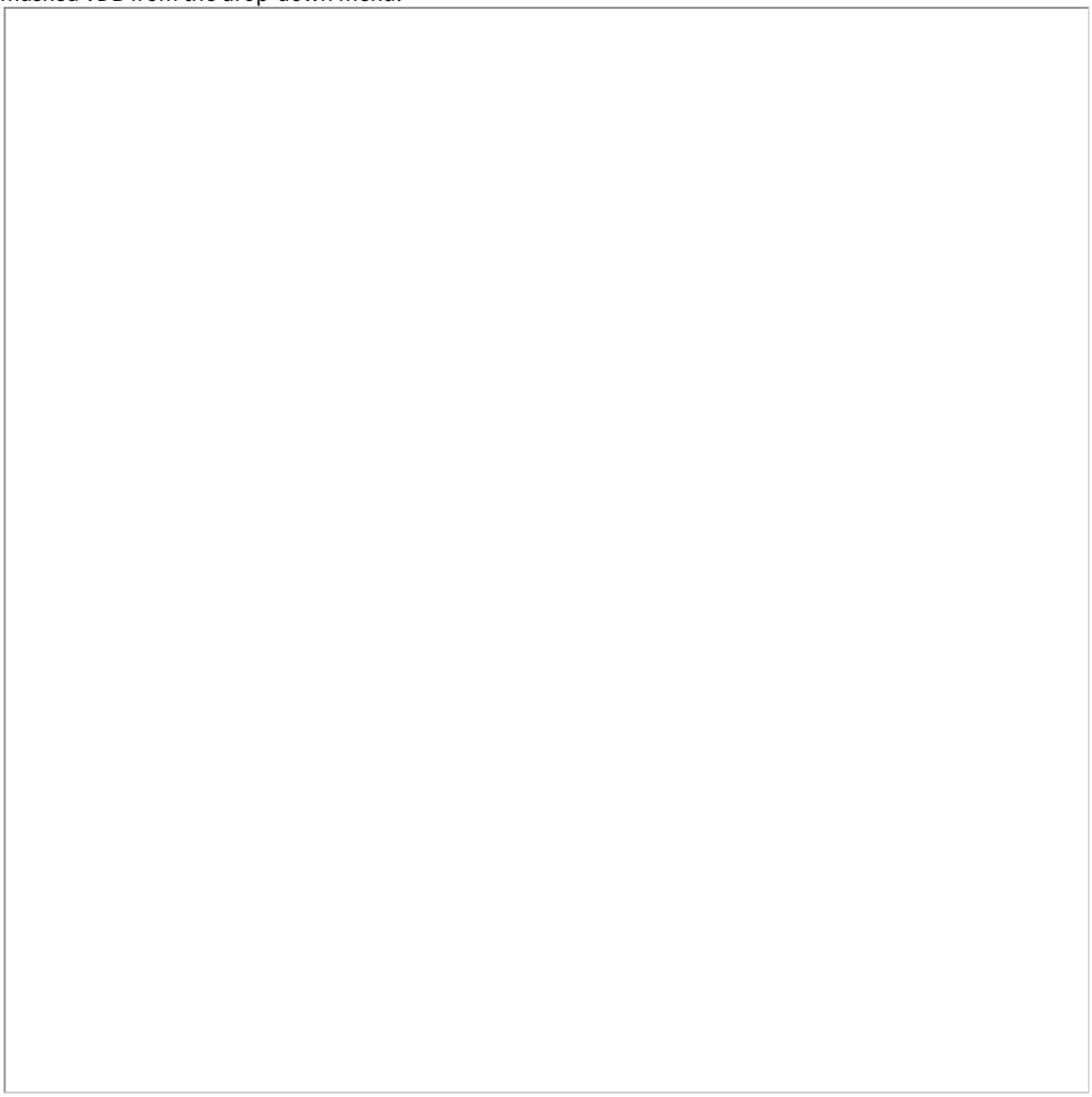
   This will send you to the **Create Data Template** page.
3. Enter a **Name** for the data template.
4. Optionally, enter a **description** for the data template.

5. Click **Add Data Source** to add data sources to the template. Each data source name will include the name of the datasets group with which it is associated.



Create Data Template window with data source drop-down menu

To select a replica data source, first, select the name of the replica it belongs to. Then pick the replica masked VDB from the drop-down menu.



6. To set a startup order, select the Set startup order of data source checkbox, then from the drop-down select the order.
7. Select **Create**.

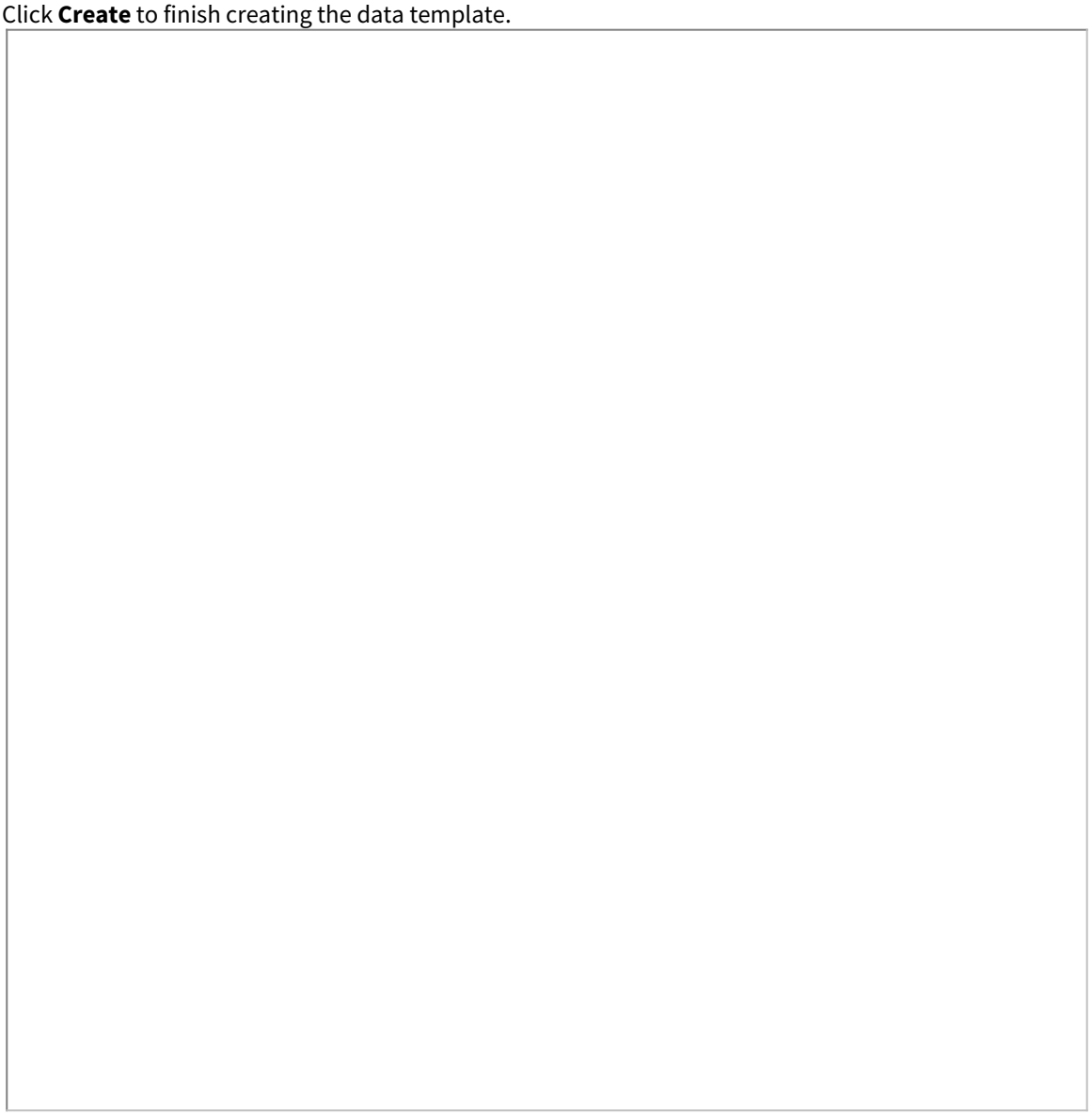
⚠ Default vs. Setting the Ordering of Data Sources to a Data Template
You have the option of setting the ordering of data sources to a data template. This option executes Delphix Self-Service operations sequentially on each data source (rather than in parallel), to ensure consistency among sources that need to be started/shut down in a particular order. You cannot change this setting after the data template has been created. **If you want the default behavior of running operations in parallel, do NOT select the box labeled "Set startup order of data sources".**

When your template has ordering constraints, as with Oracle EBS, you must set the startup order for each data source. Select the **Set startup order of data sources** box. The Delphix Engine will select the data source with order 1 as the first source started and the last one to be stopped. The data source with order 2 will be selected as the second source started, and this sequence will continue until the last data source is selected and ordered. Note that it is not possible to have operations performed in parallel on a subset of data sources and sequentially on a different subset of data sources.

Configure the synchronization and consistent ordering of data sources

1. Select the **Set startup order of data sources** box.
2. Use the drop-down menu to select the **source** you want to include. The drop-down menu will display all dSources in the system and all VDBs and vFiles that are not already assigned to a Delphix Self-Service data container.
3. Enter a specific **name** for the data source.
4. Optionally, enter a **description** in the **Notes** section. Delphix Self-Service users see a copy of these notes in the data containers they own.
5. Click **Add Data Source** to continue to add and configure more data sources to the data template. You can remove data sources using the **Delete** icon.
6. By default, the **Order** of the sources will correspond to the order they are added. You can also edit the **Order** using the dropdown for each source.

7. Click **Create** to finish creating the data template.



⚠ For Oracle EBS, the vFiles dbTechStack will have order 1, the Oracle database order 2, and the vFiles appsTier order 3. For more information about EBS, see the EBS documentation.
Once you have created a template, you cannot change the set of data sources in it. Any VDBs or dSources being used as data sources in Delphix Self-Service will appear with a special badge in the Management application.

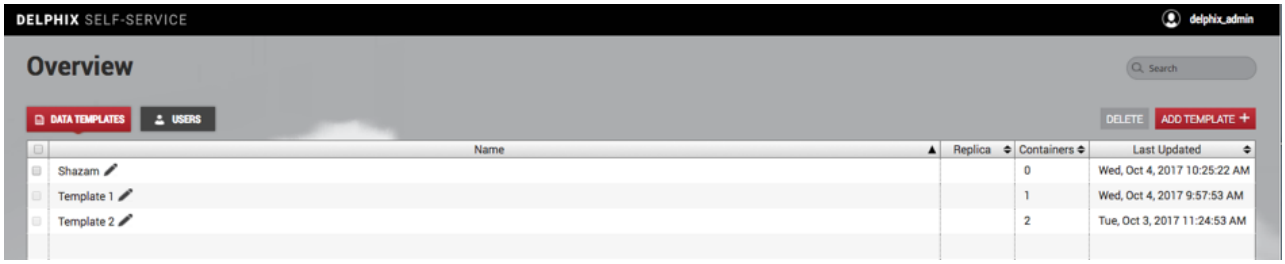
Viewing data templates

As the Delphix Admin user, you can view what sources have been included in a data template. You can distinguish the masked sources from the unmasked sources by referring to the corresponding data icons, as seen below.

Example of a template containing both masked and unmasked VDBs

Managing data templates

After you have created the data template, it will be visible from the **Overview** page under the **Data Templates** tab, which is the default tab.



Data Template Details in the Overview page

Notes

- Each line corresponds to a data template and contains high-level information about that data template. For example, the number of child data containers is visible in the **Containers** column.

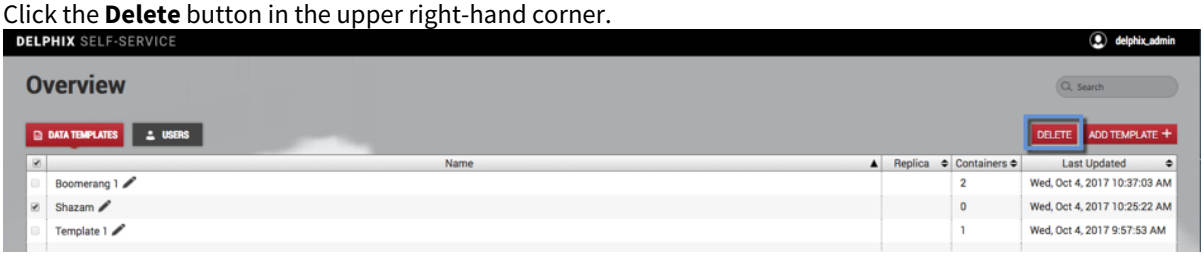
- You can search, sort, and filter the list of data templates, making it easy to manage a large number of data templates in Delphix Self-Service.

Editing a data template's name

1. Click the **Edit** icon next to the data template name.
2. Enter the new **name**.
3. Click the **checkmark** icon to confirm changes.

Deleting a data template

1. Select the **data template** you want to delete.
2. Click the **Delete** button in the upper right-hand corner.



Data Containers
If there are any data containers provisioned from the data template, you must remove them before you can delete the data template. See instructions in Data Container Activities.

Understanding how to manage data template details

The data template details page

In the **Overview** page, under the **Templates**, click the data template's **name**. This will direct you to the templates details page. You can use this page to view and configure details of an individual data template. It consists of a number of tiles, described below.

Summary

Use this tile to get an overview of the data template and its child data containers.

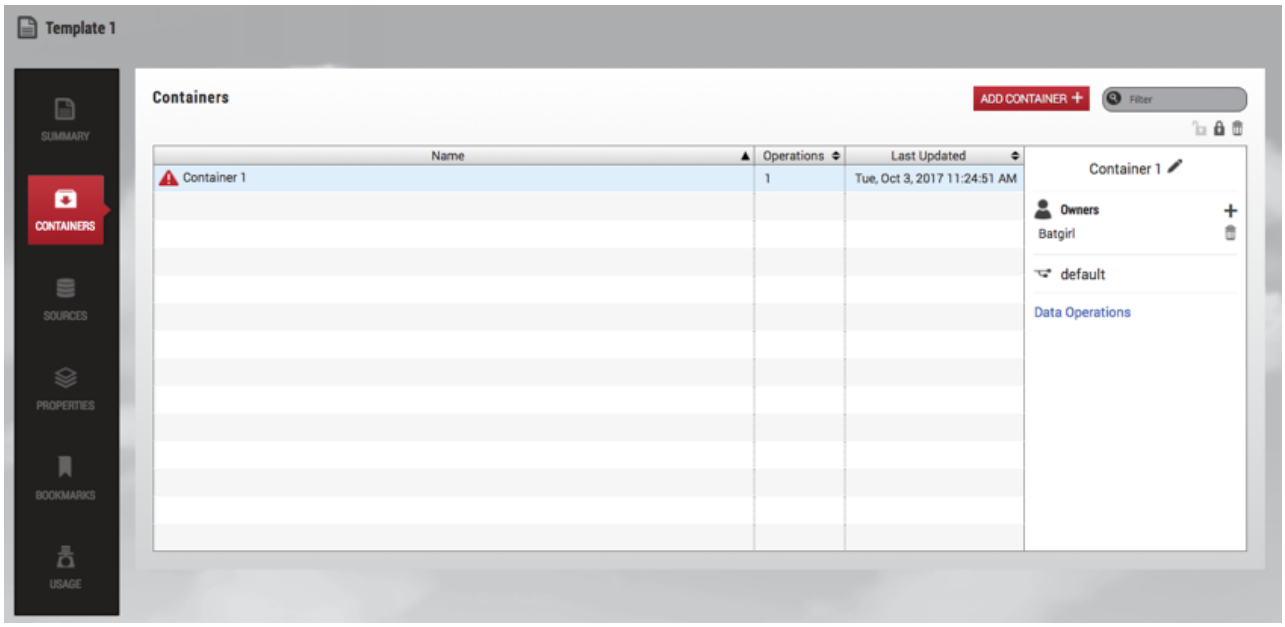
Summary Details for Templates

Notes

- The graphs labeled **Operation Counts By Week** and **Container Data Age Distribution** give a sense of the amount of activity in the data template over time
- **Top Checkouts** shows at a glance which bookmarks have been used most frequently as part of a **Restore** or **Branch** operation
- **Top Users by Data Operations** shows at a glance which users are the most active

Containers

Use this tile to create, view, and delete child data containers from this data template.



Container Details

Sources

In this tile, you can view the data sources that this data template uses. Each data source has a user-visible name, a description, and a set of properties that consist of arbitrary key/value pairs. This information will be included in the data containers provisioned from this template.

Sources Details

Properties

Use this tile to edit the data template's properties. Properties are arbitrary key/value pairs associated with the data template. These values will be propagated to all data containers provisioned from this template. This provides a way for you to annotate data templates and data containers with whatever information is relevant to their use case.

Properties Details

Bookmarks

Use this tile to create and manage bookmarks on the data template. A bookmark represents a given point in time that is protected against retention. Bookmarks created on a data template are visible to all of the data containers provisioned from it. For more details, refer to the Bookmarks section in the Delphix Self-Service Data User Guide.

Bookmarks Details

Capacity

Use this tile to get information about the storage associated with the data template and its child containers.

Understanding data containers

Delphix self-service data container overview

Data containers are provisioned from data templates by administrators and assigned to a user. A data container represents a socket that is capable of making any data within the data template accessible. The user controls what data they want to access. Delphix Self-Service users have effectively been provisioned a set of "physical" resources, such as a database on a host that consumes some set of resources. A data container is comprised of a virtual database (VDB) or vFiles provisioned from each source in the data template from which it is created. The data container manages these VDBs, and the data operations performed on a data container will only impact these VDBs. Data containers represent the separation between IT infrastructure and end-users. IT determines the set of VDBs or vFiles to allocate to a data container, and users determine the data that they want accessible in the containers allocated to them.

Data containers can be used to access any data within a single data template, but not across templates. Users have the ability to populate the data within their data container from any point in time on the data template, the data container's history, or shared bookmarks from other data containers. Although operations are all accomplished by performing Timeflow operations on the underlying VDBs, the data containers hide the VDBs and their underlying properties from users. None of the data container operations require provisioning additional VDBs; everything is accomplished using the resources assigned when the data container is created.

Operation	Description
Refresh	This is the same basic concept as **Refresh** in VDBs. In Delphix Self-Service, **Refresh** will update the data on the active branch of a user's data container. The user will then have the latest data in the sources of the data template from which the container was provisioned.
Restore	**Restore** allows a Delphix Self-Service user to update the data on the active branch of their data container to one of the following: • any point in time on the data container • the data template from which the container was provisioned • a bookmark This operation effectively means, "Take me to the data at this time."
Reset	**Reset** is a simplified version of **Restore** built to support the notion of "undo." It allows a user to reset the state of their application container to the latest operation. This can be useful for testing workflows where, after each test, users want to reset the state of their environment.
Branch	A branch represents a logical timeline, effectively a task on which a user is working. Only one branch can be active at a time, but a user can use multiple branches to track logically separate tasks. Delphix Self-Service branches do not require the allocation of a new VDB; instead, they are comprised of a collection of TimeFlows within a VDB.
Activate	This allows the user to select which branch they want to be active. Only a single branch within a data container can be active at a time.

Operation	Description
Bookmark	This creates a semantic name for a point in time and prevents this data from being removed by the retention policy. Bookmarks can be annotated with tags to make them easier to search for. In addition to tags, bookmarks allow a user to enter a description of what the bookmark represents.
Share	Bookmarks can be shared, which allows them to be seen by users who own data containers that have been provisioned from the same data template. This allows users to share data, providing a way for other users to either restore their existing timeline or create a new branch from these shared points.

Once a data operation has been selected its progress can be viewed from the Action sidebar in the Management application. From the Action sidebar, users cannot cancel the enable, disable or activate branch operations.

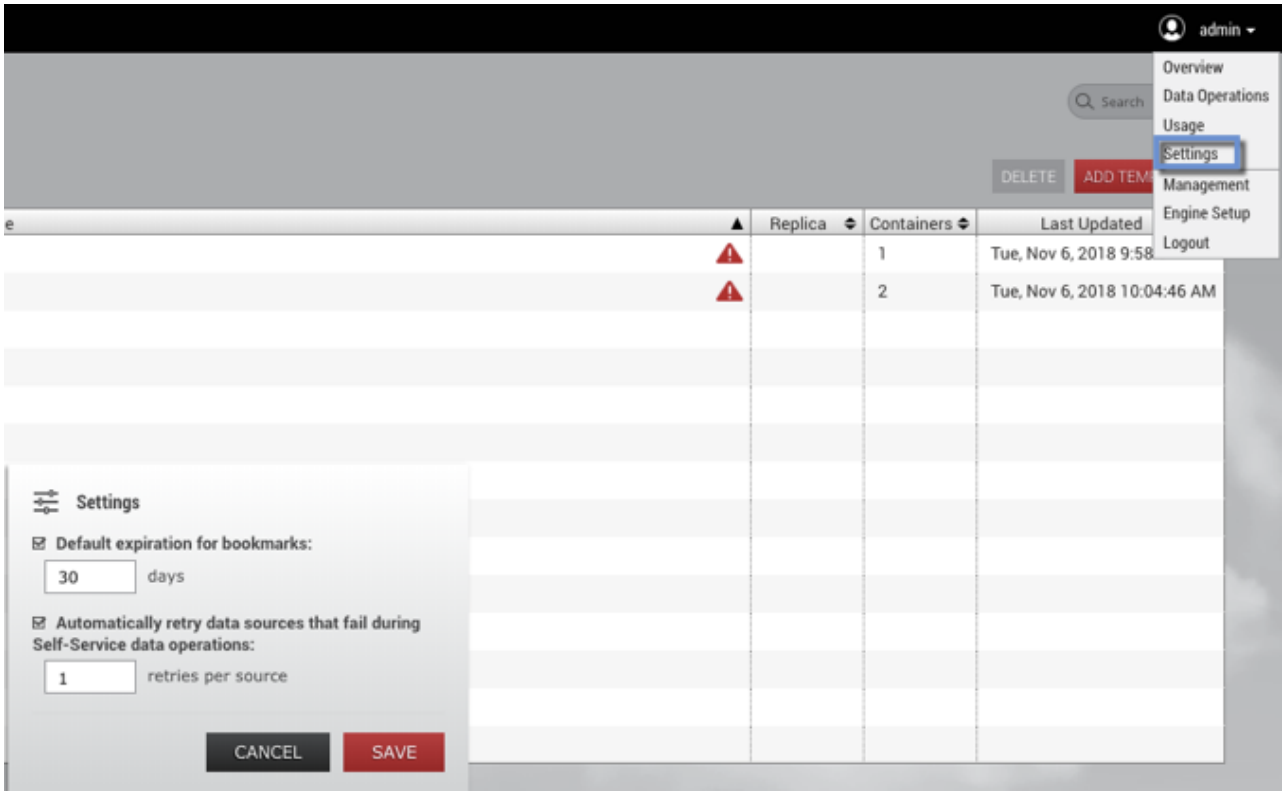
Automatic retry for data operations

To make operations on data containers more robust, Delphix Self-Service supports automatically retrying failed sources during data operations. You can specify a maximum number of retry attempts so that if an operation fails on any individual data source within a data container, it will be automatically retried until it succeeds, or until the retry limit is reached.

Automatic retry applies to any Delphix Self-Service operation that changes the data in the data container, such as Refresh, Restore, Reset, or Create Branch. This setting can be especially useful in scenarios where there are a large number of sources in a data container, and some sources fail to update the first time. If the reason for the failures was intermittent, an automatic retry may allow the sources that failed to succeed, and the operation can still complete successfully. The default number of retry attempts is **1**.

To change the number of automatic retries:

1. Click the user icon in the upper right-hand corner of the screen.
2. From the drop-down menu, select **Settings**.
3. In the field **Automatically retry data sources...**, enter a new maximum; alternatively, use the arrows to increase or decrease the number.
4. Click **Save**.



Settings in the user drop-down menu

Delphix self-service data container recovery

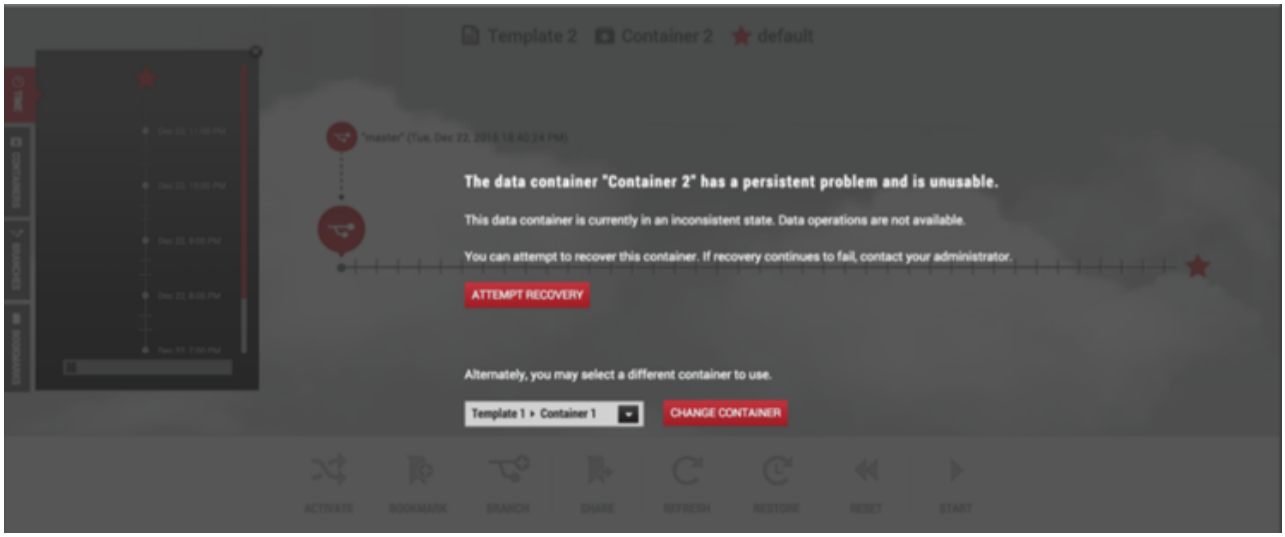
Data containers consistency

Delphix Self-Service allows you to group multiple datasets in the same data container. This makes it easy for you to access entire applications such as PeopleSoft, including binaries and code. If a data container represents an application, then there are likely to be dependencies between the application's datasets. For example, the vFiles data source containing the code will depend on a specific version of the database's schema. Therefore, it is important that all dataset sources are drawn from the same point in time. If they are, the data container is in a "consistent" state; if they are out of sync, or "inconsistent," errors will occur. For example, if the vFiles data source containing the code has been updated more recently than the database's schema, the dependency cannot work.

Delphix Self-Service currently has no way to determine whether the application is consistent. However, it attempts to minimize the chance that dataset sources are out of sync whenever it performs a data operation such as refresh, restore, or reset. When performing a data operation, Delphix Self-Service attempts to snapshot all dataset sources from a point in time as close as possible to the desired time. If at least one of the data sources fails to go to the desired point, then Delphix Self-Service considers the data container to be in an inconsistent state. The application as a whole may still be working, but Delphix Self-Service assumes that the failed dataset's data is not the correct version. To return to a consistent state, you must perform a recovery operation on the data container.

Data container recovery

Prior to performing any data operation, Delphix Self-Service takes snapshots of all datasets. Recovery is the process of rolling back a data container to a snapshot, thereby restoring it to a consistent state. When a failure occurs, you will see the following screen:



You can either perform recovery or use a different data container. Whether the recovery will fail or succeed depends on exactly why the data operation failed in the first place. If the problem was intermittent, such as a temporary network problem causing SSH failure, then performing recovery should work. If the problem is persistent – for example, the target host is out of space – then intervention is required; recovery will not succeed until you address the underlying root cause of the failure.

Admins can see the underlying failure in the **Actions** sidebar or the **Job History** dashboard. The **Actions** sidebar is the preferred place to view the failure; it has a hierarchical display that makes diagnosing the failure more straightforward.

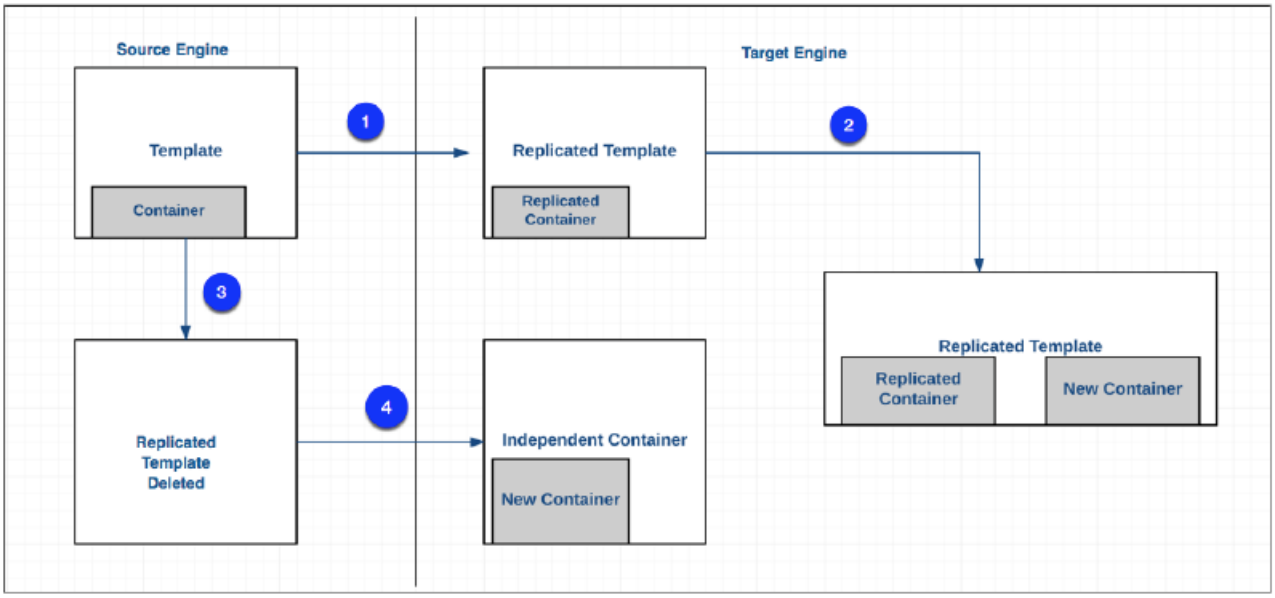
Preserving independent containers in delphix self-service during replication

Replication is used for data backup and recovery as well as for managing and sharing data across remote data centers. Delphix Self-Service users can preserve their data after replication jobs. In the past, if replication occurred on templates in containers, users would lose the data in their containers. Now admin users can preserve containers to be used independently of replication jobs.

Independent containers behave in the same way as other containers, with two exceptions:

- You cannot refresh them.
- The bookmarks created on them cannot be shared, because they do not have a template reference.

The functional overview of independent containers seen below represents the flow of steps between the source engine and the target engine. A description of what is occurring between each of the steps appears below the diagram.



Functional Overview of Independent Containers

In Delphix Self-Service, you can create a template on the source engine and then replicate the template to the target engine.

the target engine, an admin can use the replicated template to create new containers and assign them to users. You cannot change the replicated template's name or the names of the containers with which it was replicated over.

Due to an update, the replicated template is deleted from the source engine

The deleted replicated template will be removed from the target engine. Any new container created in step 2 loses the reference to the deleted template and becomes an independent container.

Creating independent containers

Prerequisites

- The replication source and the replication target must be running identical versions of the Delphix Engine – for example, Delphix Engine version 5.1.
- The target Delphix Engine must be reachable from the source engine.
- The target Delphix Engine must have sufficient free storage to receive the replicated data.
- The user must have administrative privileges on the source and the target engines.

For more information, see Configuring Replication and Understanding Data Templates.

Limitations of this functionality

You can find independent containers in Delphix Self-Service on the target engine under the **Independent Containers** tab. They have the following characteristics:

- They cannot be refreshed, because they are no longer bound to a template.

- You can create bookmarks on them, but you cannot share those bookmarks because there is no common template.
- You can use them for branching, restoring, resetting, starting, and stopping.

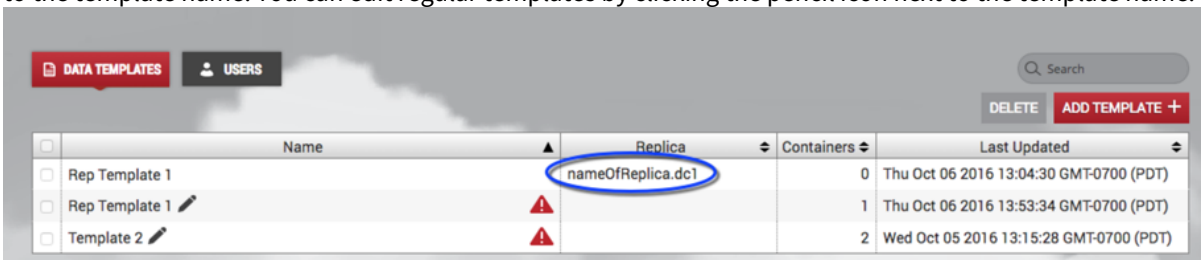
Procedure

To create an independent container, complete the following steps:

1. On the source engine, create a template with a container.
2. From the user drop-down menu, select **Management**.
3. From the **System** menu, select **Replication**.
4. Next to **Replicated Profiles**, select the **plus** icon to **Create New Profile**.
5. Under **Objects Being Replicated**, select your Self-Service **template** and its associated **container**.
6. Enter your profile information.
7. Click **Create Profile**.

When replicating templates, you can select all, some, or none of their associated containers in the replication profile. This is done by selecting the checkbox next to the container's name in the **Create New Profile** window. When replicating a container, you must also replicate its associated template. Replicated objects cannot be modified on the target engine unless they are failed over, so you cannot modify the names of replicated data containers and templates.

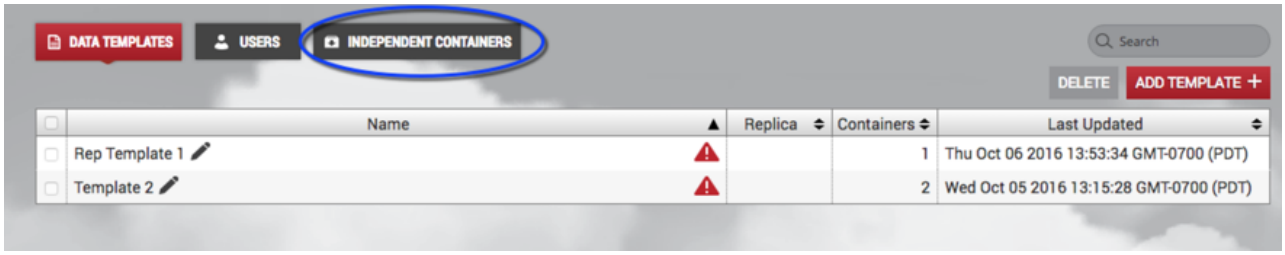
8. Once the profile has been created, click **Replicate Now**.
9. On the target engine, click the user menu.
10. Select **Self-Service**.
11. The replicated template will appear in Self-Service on the target engine. The replica name is displayed next to the template name. You can edit regular templates by clicking the pencil icon next to the template name.



12. Select the replicated template, then select **Containers**.
13. In the **Containers** window, click **Add Container**. In order to complete this action, you will need to ensure that there is data available from each data source in the template. This means that VDBs must have been provisioned from each replica dSource or VDB in the template. After the container is created, your replicated

template should have the new container you just created and the original container created in step 1.

14. On the source engine in Self-Service, delete your template.
15. From the user menu, select **Management**.
16. From the **System** menu, select **Replication**.
17. Replicate your profile to create a new independent container.
18. On your target engine, select **Self-Service**.
19. The new container is created. To find it:
    a. Login to the target engine.
    b. Click the user menu.
    c. Select **Self-Service**.
    d. In the Overview page, select the **Independent Containers** tab.



Delphix self-service data container activities

Configuring data containers in Delphix self-service

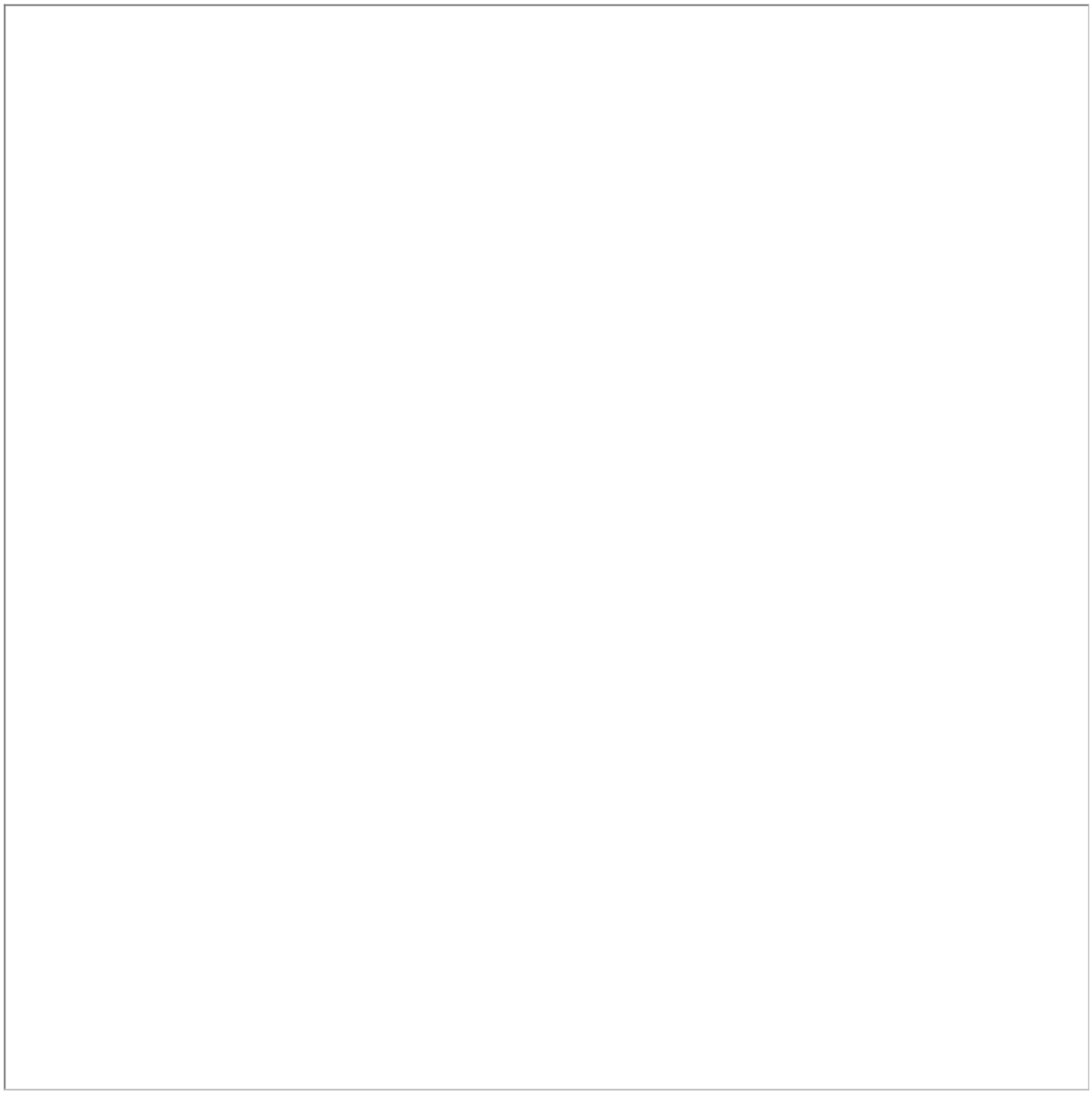
A data container is comprised of a set of virtual databases (VDBs), where each VDB is a direct child of the dSource, VDB, or vFiles in the data template's data sources. Delphix Self-Service does not automatically provision VDBs when creating a data container; a Delphix admin must create the required VDBs using the **Management** application.

> Once a VDB or vFile is part of a Self-Service container, you cannot use the Management Service to rewind, refresh, or delete it. You can still use the Management Service to disable or enable it, take a snapshot of it, or provision a new VDB or vFile from it.

1. Select the **Overview** page.
2. Select a **template** from which you want to create the data container. This will take you to the **Data Template** page.

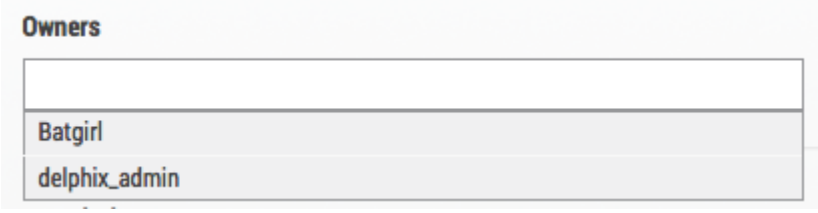
Add a data container

1. Click **Add Container** in the upper right-hand corner of the screen.  *Details Panel and Dashboard*
   This will take you the **Create Data Container** page.



Data Container Dialog

2. **Optional**: Enter information about the data container, such as the **Name** and **Description**.
3. Once the data container has been createdSelect the **Owners** for the data container from the search box. It is acceptable to have multiple owners for each data container. Any Delphix administrator is able to manage all containers, so the owners should be end-users. For details, see Understanding Delphix Self-Service User Management.



4. When a data container is created, you now have the option to:

   a. **Refresh data sources to most recent template state** – This option will refresh VDBs before adding them to the container. This is done to enforce that when multiple sources are used in a container, the sources are consistent.
   b. **Add data sources to container as-is** – This option will not refresh the data sources.
5. Select the **VDBs** to use for this container's data sources. The available VDBs have the following constraints:
   - They have been provisioned from the dSources/VDBs belonging to the parent data template
   - They are not already part of another data template or container

   If there are no VDBs that meet these constraints, you may see a message informing you that you do not have any compatible VDBs. *VDB Warning Alert*
6. Click **Create**.

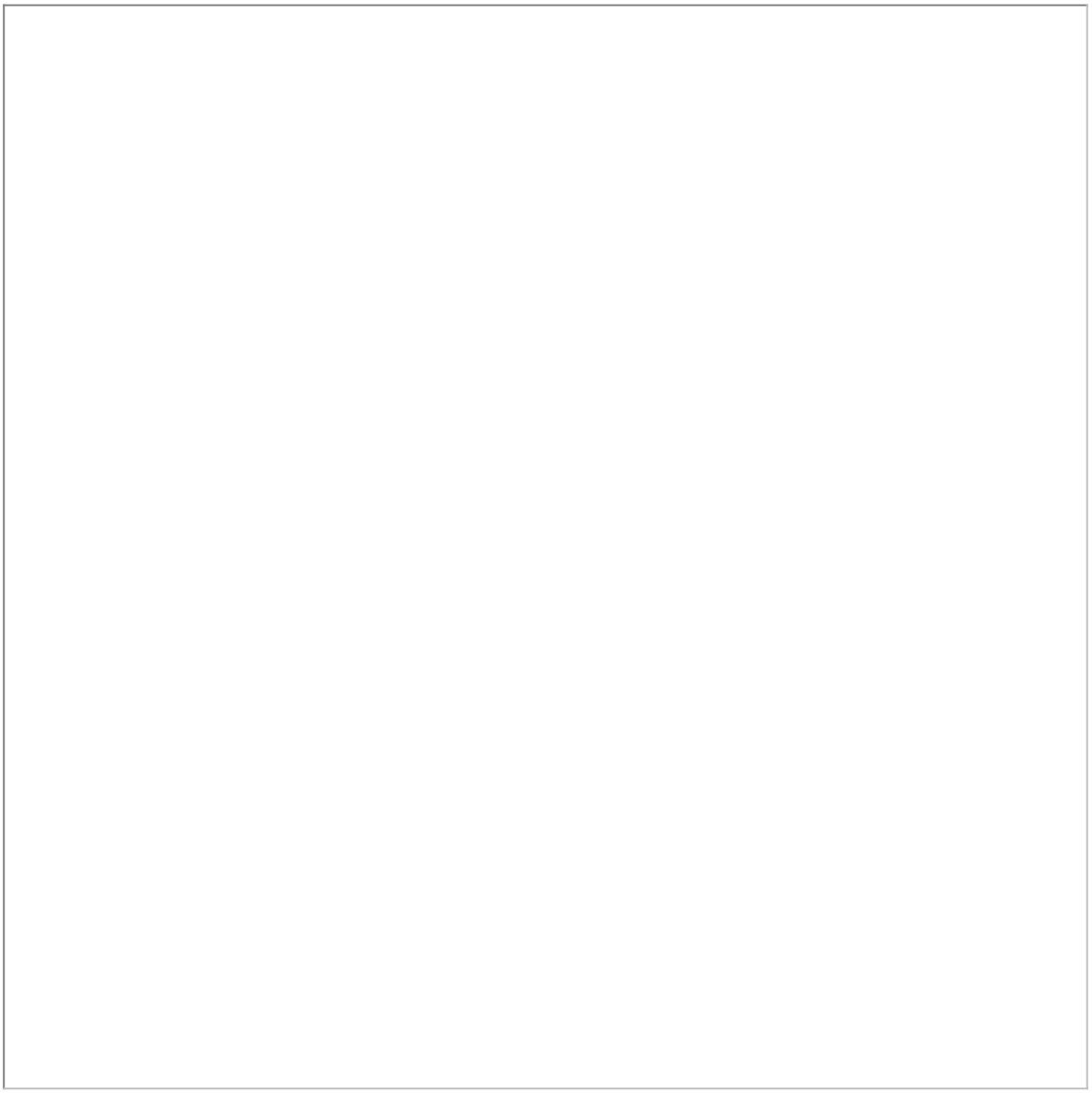
Selecting Masked Data Sources for Data Containers

Prerequisites

- Using Masked Data Sources with Delphix Self-Service
- Selecting Masked Data Sources in Data Templates

Procedure

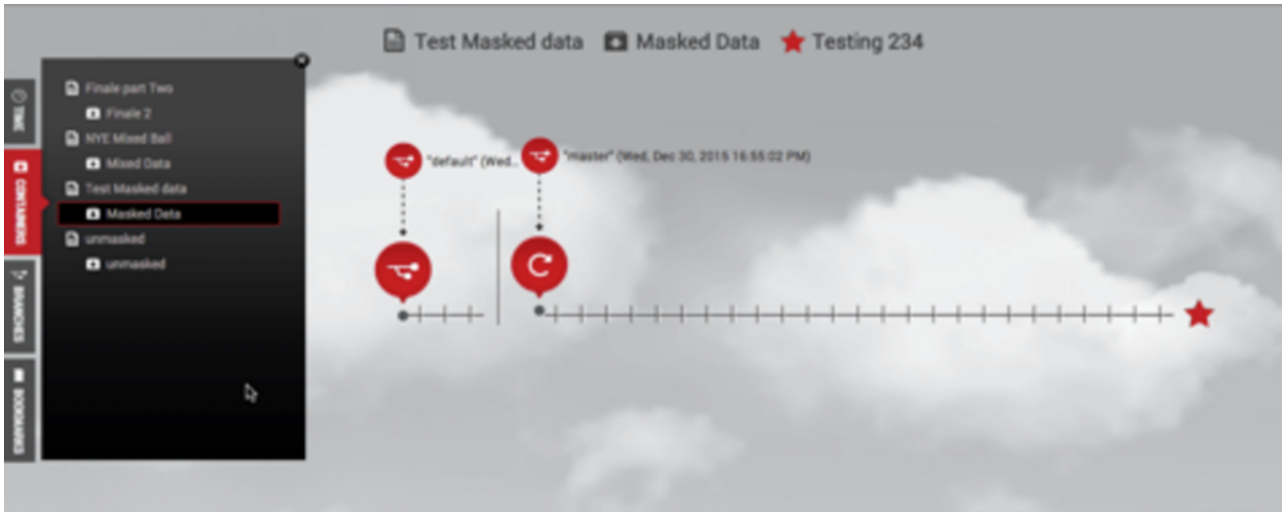
Once you have selected a child masked VDB for the data container, you can see the parent-child relationship as a masked source under data sources.



Masked Data Sources Parent/Child Relationship

As an admin user, you can select both masked and unmasked data sources in both Delphix Self-Service templates and data containers.

Delphix Self-Service users will not know whether the data in their containers and branches is masked or unmasked. All Delphix Self-Service functionality remains the same regardless of whether a data source is masked or unmasked.



The figure above is an example of a data container with masked data.

Delete a Data Container

All data sources (VDBs and vFiles) in a Data Container are not deleted as part of the Data Container deletion process.

When performing the **Delete Container** operation, you can check the **Delete associated VDBs and vFiles** box in the dialog window to delete the data sources associated with the container.

Data management operations

Start a data container

Starting a Data Container does the following:

- Starts the data sources, This means that each data source listed in the **Source Details** section of the **Data Container** page will start using CPU and network resources on the host system it is running.
- Puts a copy of the data from the active branch into those data sources.

On the **Self-Service Toolbar**, click **Start**.

Stop a data container

Stopping a data container does the following:

- If not already done, copies the current data in the data sources into the active branch of the data container
- Shuts down the data sources. This means each data source listed in the **Source Details** section of the **Data Container** page will stop using CPU and network resources on the host system.

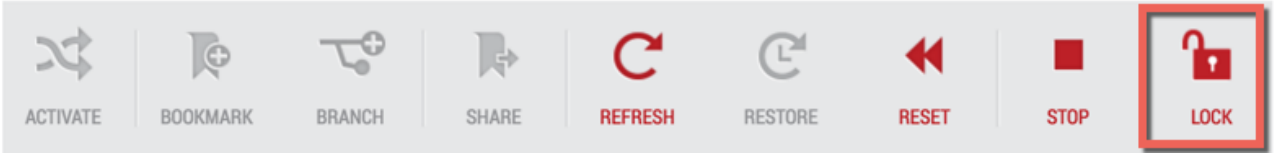
On the **Self-Service Toolbar**, click **Stop**.

Other operations on the data container, such as Stop, Reset, and Refresh, must be performed from the **Data Management** page:

Data Management Interface Shortcut in Delphix Self-Service Data Template

Locking a Container

On the Self-Service Toolbar, click **Lock**.



Locking a data container does the following:

- You become the only user who can perform operations on it.
- For all other users, the container appears disabled.

Unlocking a container

Unlocking a data container does the following allows other Self-Service users to perform operations on that container.

This operation is only enabled if a container is currently locked. Only the user who locked a container or a Delphix Self-Service administrator can unlock it.

In Delphix Engine version 8.0.0.0 (or later), if a Self-service data container contains an Oracle virtual PDB (vPDB) and the PDB's virtual container database (vCDB) contains more than one vPDB, the start of the container will fail with error 'failed to enable data container "<data container name>"' if the vCDB is not enabled or it is not running. Start the vCDB first (using Delphix Management UI) and then start the data container again.

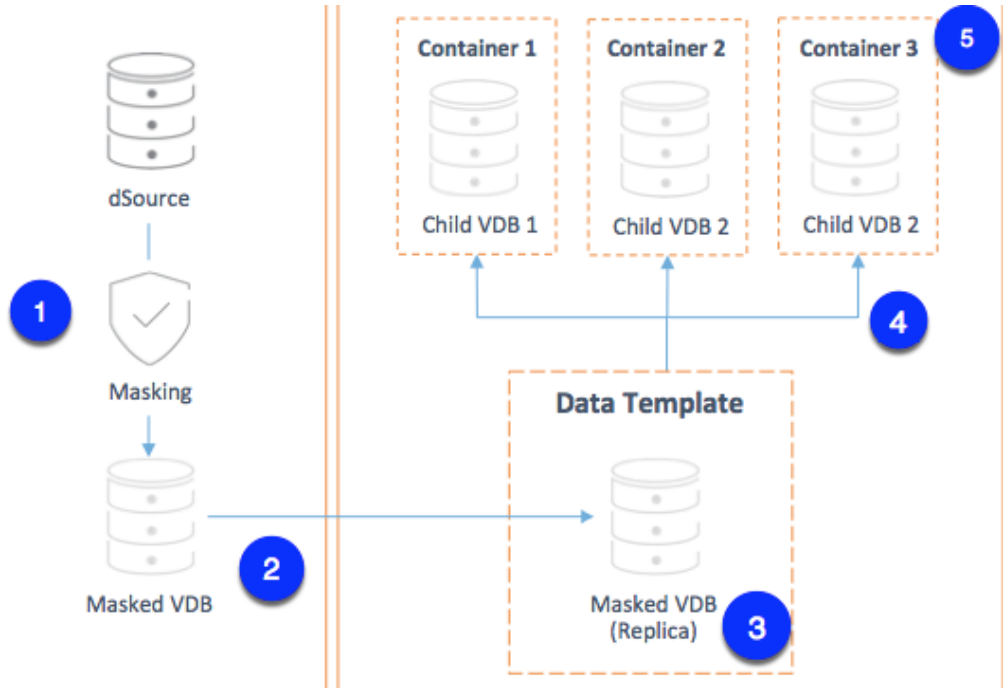
Using masked data sources with Delphix self-service

SDD Overview

You can now replicate masked data in a VDB directly to a target Delphix Engine without transmitting the unmasked data in its parent source. This is called selective data distribution (SDD). Although you can run selective data distribution ad hoc, it is typically run according to a predefined schedule. In the current release, there are some best practices and limitations you should know about before using data from SDD in Delphix Self-Service. This section covers the workflow for using SDD replicated data in templates and containers. It is aimed at administrators who are familiar with the process of creating a masked VDB, SDD replication and setting up objects.

Configuring Delphix Self-Service with Masked Data Sources

The diagram below illustrates the steps for using masked data sources in Delphix Self-Service.



Step 1: Provision a masked VDB on the source.

Step 2: Use SDD to replicate masked data to the target.

Use SDD to replicate your masked VDB to the target. The target VDB will be called the “replica masked VDB.” Replica Masked VDBs

To keep the replica masked VDB up to date, configure a refresh policy for how often it should refresh. The refresh policy should be related to the schedule for SDD updates from the source. Refreshing more frequently will result in the VDB being unavailable to Delphix Self-Service more often than needed.

Step 3: Create a data template on the target.

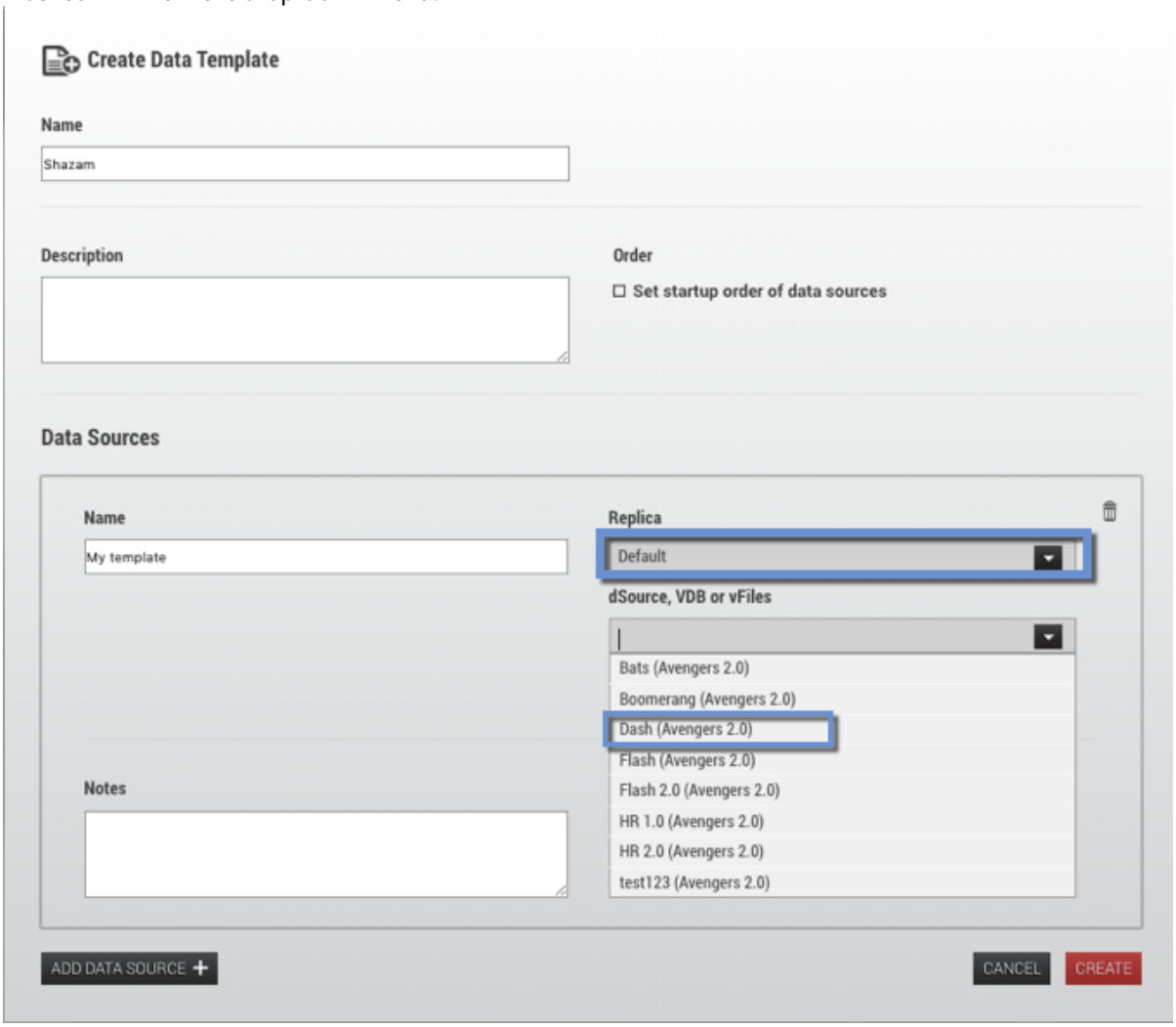
To create a data template:

1. From the drop-down menu in the upper right-hand corner of the Delphix Management application, select **Self-Service**.

2. On the **Overview** page, click **Add Template**. *Add Data Template* This will send you to the **Create Data Template** page.
3. Enter a **Name** for the data template.
4. Optionally, enter a **description** for the data template.
5. Click **Add Data Source** to add data sources to the template. Each data source name will include the name of the datasets group with which it is associated.

   Create Data Template window with data source drop-down menu

   To select a replica data source, first, select the name of the replica it belongs to. Then pick the replica masked VDB from the drop-down menu.



6. To set a startup order, select the Set startup order of data source checkbox, then from the drop-down select the order.
7. Select **Create**.

Step 4: Provision child VDBs from the replica masked VDB.

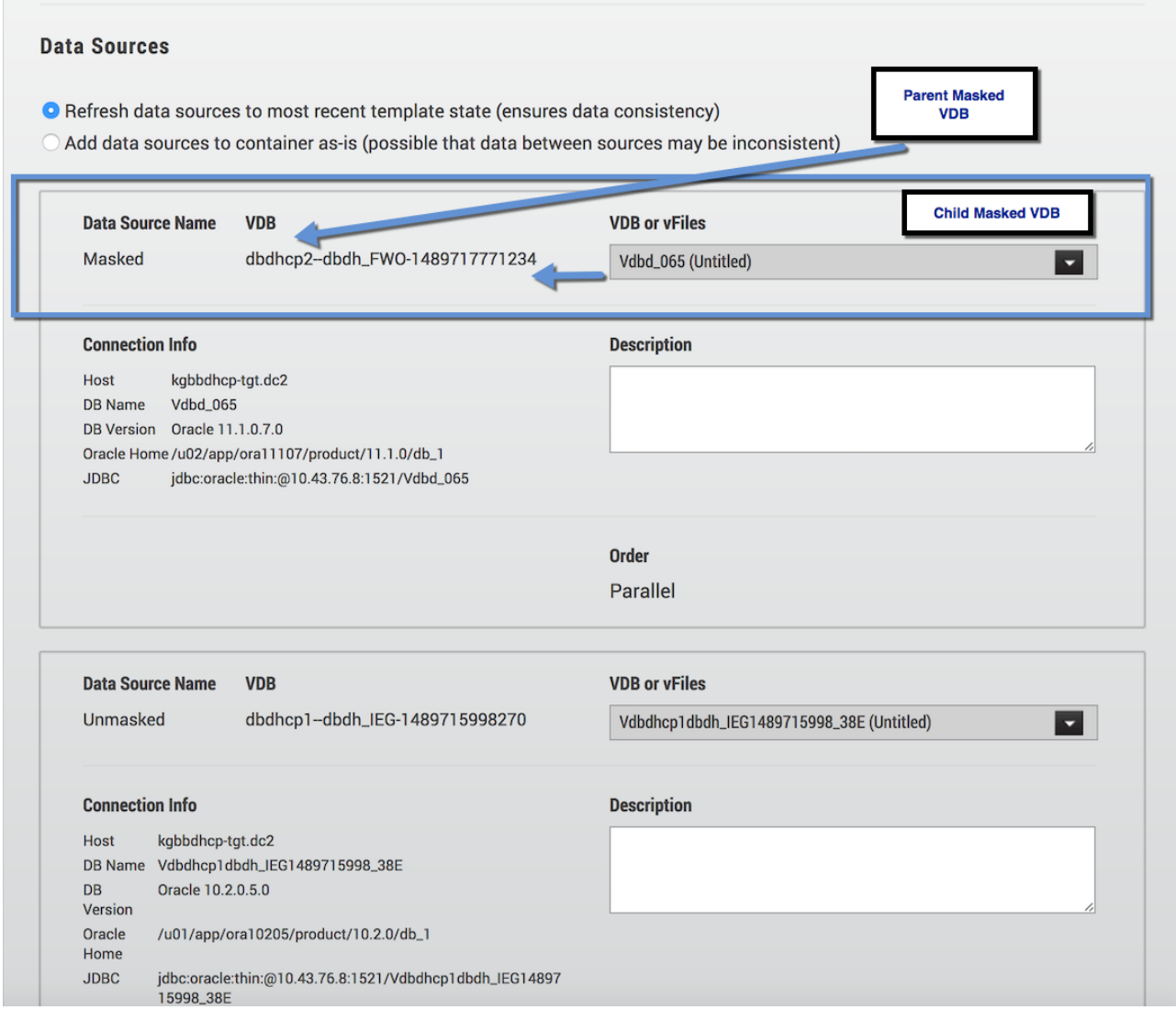
1. Login to the **Delphix Management** application for the target host.
2. Click **Manage**.
3. Select **Datasets**.
4. Select the **replica** that contains the dSource or VDB to be provisioned.
5. The provisioning process is now identical to the process for provisioning standard objects.

Step 5: Add data containers and select the child VDBs as data sources.

Follow the instructions to add a data container.

Select a masked child VDB as a source for the container. As an admin user, you can select both masked and unmasked data sources in templates and data containers.

Once you select a child masked VDB for the data container, you can see the parent-child relationship as a masked source under data sources.



Masked Data Sources Parent/Child Relationship

Refreshing Masked VDBs in Delphix Self-Service Data Templates

Make sure that in step 4 above, you select the replica masked VDB as the source for the data template.

In order for new data to be available in the template on the target, you must do the following:

1. Refresh the masked VDB on the source. This will re-run the masking job.
2. After the refresh completes, execute the SDD spec for the masked VDB.

Wait to refresh If you do not wait until the refresh is complete, unmasked data may be sent to the targ

SDD update
Although you can employ a policy to drive refreshes of the masked VDB, you cannot use that policy to drive the SDD update as well. You may need a combination of policies and scripts to automate the workflow.

Limitations

SDD Replication Profile
You cannot add data templates to an SDD replication profile. As a result, you must create the data template on the target. This is step 3 above.

Understanding Delphix self-service user management

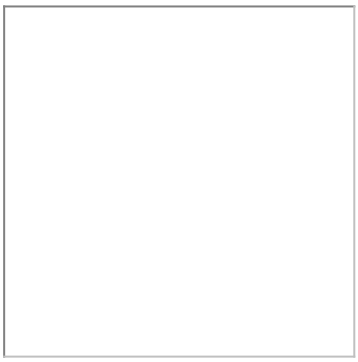
User management activities

This document describes the process of creating a user and assigning that user to a data container. It also provides an overview of the **User Details** page.

Creating a User

Follow the same process when creating a new user or modifying an existing Delphix user. Delphix Self-Service users do not have access to the existing admin user interface, and they can only access the **Data Container** page for containers they own.

1. From the **Management** application, select **Manage**.
2. Select **Users**.
3. Click **+** to add a new user or to make an existing Delphix Engine user a Delphix Self-Service user, select the user from the list.
4. Enter the appropriate information.



5. From the **User Type** drop-down menu, select **Self-Service Only**.
6. Press **Submit**.

The user is now a Delphix Self-Service user! They can now login to the Delphix Self-Service user interface, and you can make them the owner of a data container.

Notes

- Users will only be able to access the **Data Management** page. They will not be able to access the other portions of the Delphix Self-Service interface, nor the **Management** application.
- A Delphix admin user cannot be made a Self-Service Only user. However, admins can still use Delphix Self-Service and own a data container. Admins are also able to manage all data containers.
- A user who owns one or more data containers cannot be deleted.
    - For the list of data containers that a given user owns, see **User Details**.
- You cannot revoke a user's Self-Service Only role if they own any data containers.
    - For the list of data containers that a given user owns, see **User Details**.

Assigning a user to a data container

This section describes how to assign a user (created in the previous section) to a data container. Making a user the owner of a data container allows them to perform operations such as **Refresh** on that data container. Users cannot see or manipulate data containers that they do not own. You can either assign a user when creating a new data container or modify the owner of an existing data container.

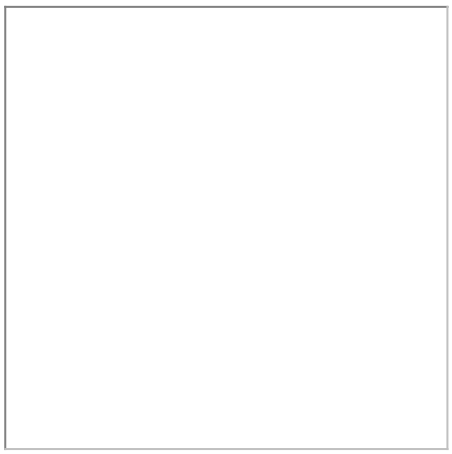
Case 1: Assigning a user to a new data container

1. Navigate to the **Overview** page.

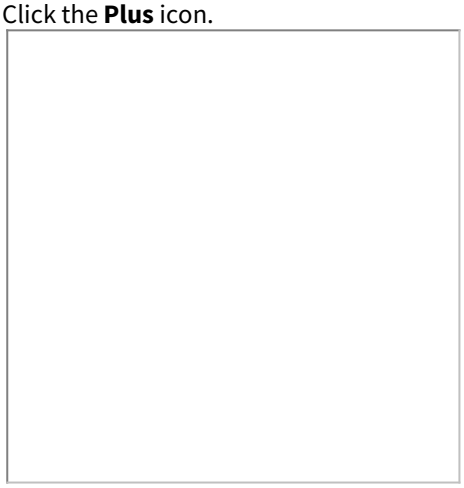
2. Select a **template** from which you want to create the data container. This will take you to the **Data Template** page.
3. In the upper right-hand corner of the screen, click **Add Container**.
4. Optional: Enter information about the data container, such as the **Name** and **Description**.
5. Select the **Owners** for the data container from the search box. It is acceptable to have multiple owners for each data container.
6. Select the **VDBs** to use for this container's data sources and Save.

Case 2: Changing the owner of an existing data container

1. On the **Management Overview** page, select the **data template** from which the data container was provisioned.
2. Click the **Containers** tile in the left-hand panel.
3. Select the **trash can** icon next to the username.



4. Click the **Plus** icon.



5. Select the desired **owner** from the drop-down list.

The user you selected is now the owner of the data container and can perform operations on that data container.

User details page

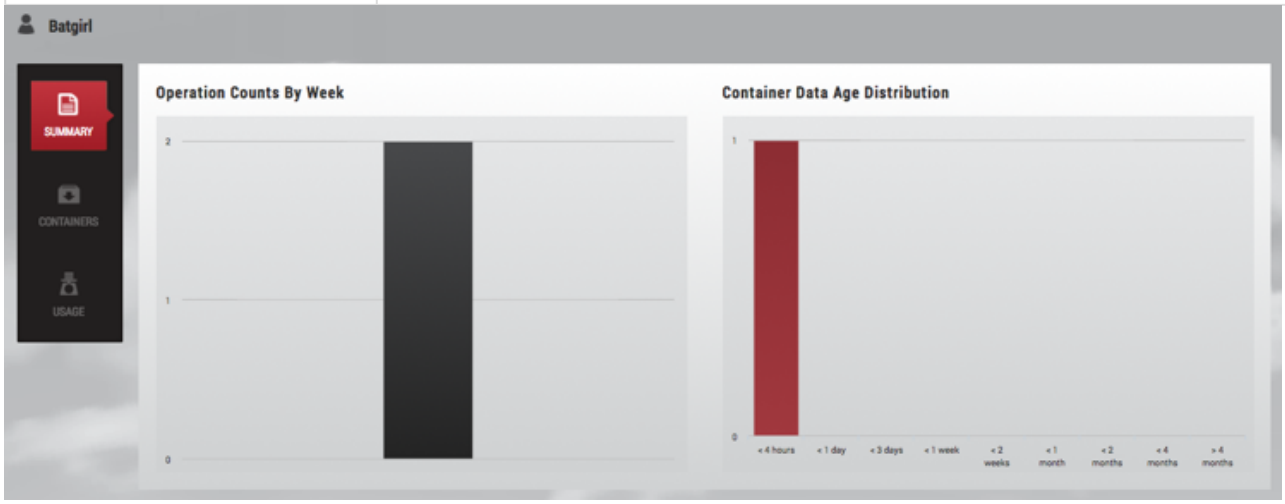
This section provides an overview of **Delphix Self-Service User Details**. This page displays graphs related to the user's Delphix Self-Service activity, as well as a list of all of the data containers that the user owns.

1. On the **Management Overview** page, click the **Users** tab.

2. Select the **name** of the desired user to go to their **User Details** page.

The following sections will display important information:

Section	Information displayed
Operation Counts By Week graph	Shows the aggregate of all Delphix Self-Service operations performed this user has performed on all of their containers.
Container Age Distribution graph	Shows the average time since a data operation was performed on all of the user's containers
Containers	Lists all containers that the user owns



User Details

Working with multiple container owners

Delphix Self-Service administrators can designate multiple users as owners of a single data container. These users all share access to the same data container which means actions taken by one user will impact all users on the same data container. For example, if User A activates Branch X, User B will also see Branch X as the active branch. This ability for one user's actions to impact another user on the same containers creates new concerns for users sharing the same container. As a result, more processes should be put into place in order to coordinate usage between users. Each team is different, but strategies include:

- designating a person to perform certain data operations
- saving your work with a bookmark or creating/working on a personal branch
- being aware of who is using your data container/data before performing operations
- locking a container to prevent others from performing any operations on it

How many owners should a container ideally be shared between?

There is no technical limit built into the software, but it is best if a team of 5-10 users shares a single data container. In most cases, having fewer owners minimizes overhead and conflicting usage. One owner per container provides maximum productivity and minimal overhead, so this feature should only be used if your infrastructure or processes require that multiple users share a container. Additionally, Jet Stream Only users currently cannot see other users with whom they share the container.

How should users handle potentially disruptive operations?

If one user performs an operation on a data container, it will affect the other owners of that container. Additionally, each user has permission to perform the same operations on the data container; currently there are no fine-granularity permissions that limit the operations a user can perform. All operations are potentially disruptive, but the level of disruption varies by operation. If any of the following operations are performed at the same time, the second operation will fail due to a conflict when processing the job.

Conflicting operations

- Refresh
- Restore
- Reset
- Enable/Disable
- Create Branch
- Activate Branch
- Delete Branch
- Create Bookmark
- Delete Bookmark

If User A performs a destructive operation while User B is "using" the data container, the operation will destroy User B’s current state. Currently, the interface does not provide insight into whether the data container is in use by another user.

Destructive operations

- Refresh
- Restore
- Reset
- Enable/Disable
- Create Branch
- Activate Branch

Deleting objects

All owners can delete any bookmarks or branches in the container, regardless of who created them.

Coordinating users

Opportunity for disruption increases as more owners are sharing a single container. Sharing a container works best when users can communicate with each other, such as when they are part of a team, or when they are working with the container at different times.

Additionally, these disruptions can be avoided when Delphix Self-Service users lock their containers to prevent other users from performing operations on them. Users cannot see the other users with whom they share the container. However, if a user locks a container, only that user will be able to use the container; it will appear disabled to others.

What operations could disrupt others using a container?

Potentially disruptive operations include:

- Refresh
- Switching active branches
- Deleting bookmarks
- Creating Branches
- Un-sharing bookmarks
- Restore
- Reset
- Staring/ stopping your container

What processes should I put in place?

The more owners you assign to a single container, the more processes you should put in place to coordinate usage between users. Each team is different, but strategies include:

- designating one person to perform certain data operations
- saving your work with a bookmark or a creating and working on a personal branch
- being aware of who is using your data container before performing operations
- locking a container to prevent others from performing any operations on it

Where can I see which user has performed what operation?

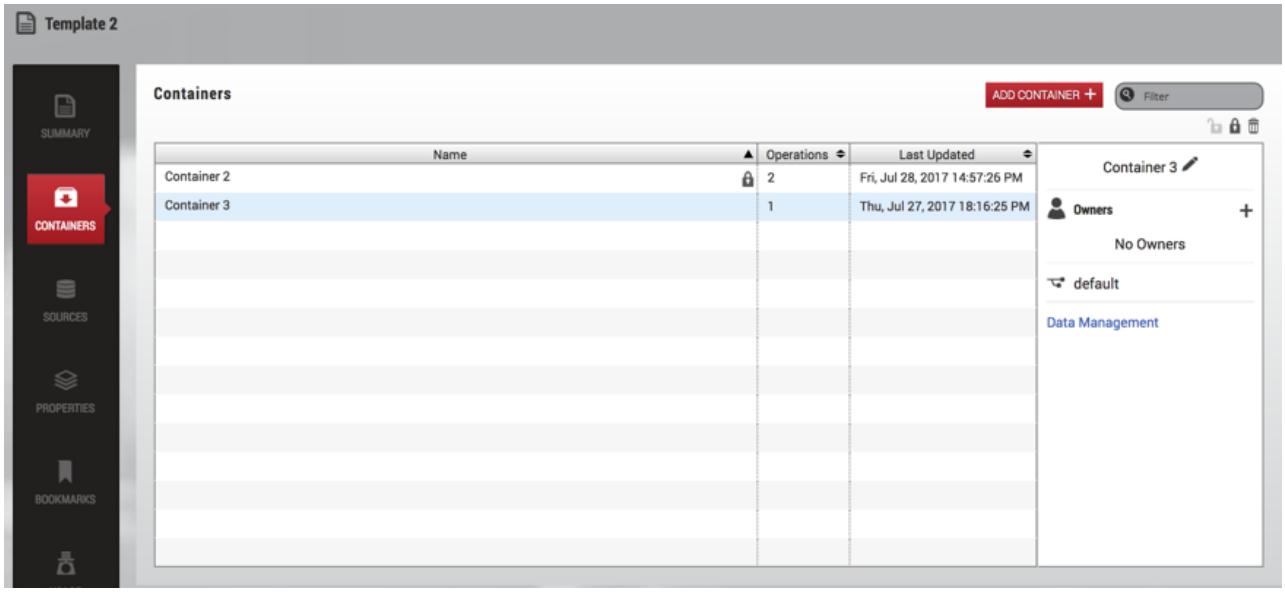
You can see which user has performed which action in the **History** tab of the **Data Operations** screen.

Be aware that operation counts in the template view are currently tabulated based on the container, not the user performing the operation.

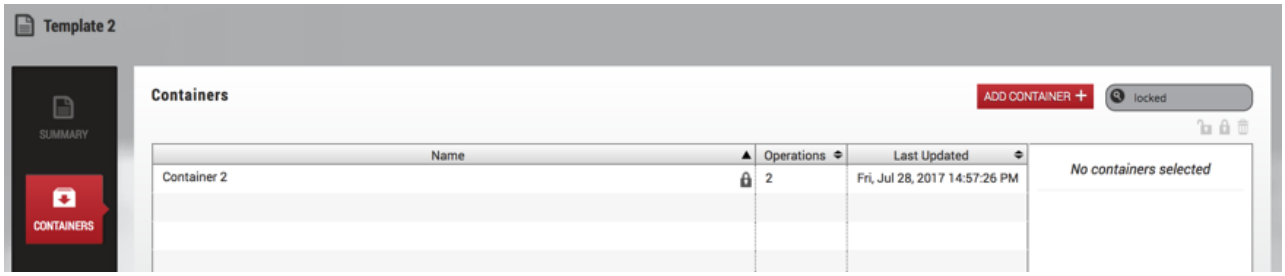
Data Operations history tab

Where can I see which containers are unlocked/locked?

Unlocked containers can be viewed by typing “unlocked” in the search filter.



You can view locked containers by typing “locked” in the search filter. To find containers locked by a specific user, type “locked by {username}” in the search filter.



Understanding bookmarks

Bookmarks overview

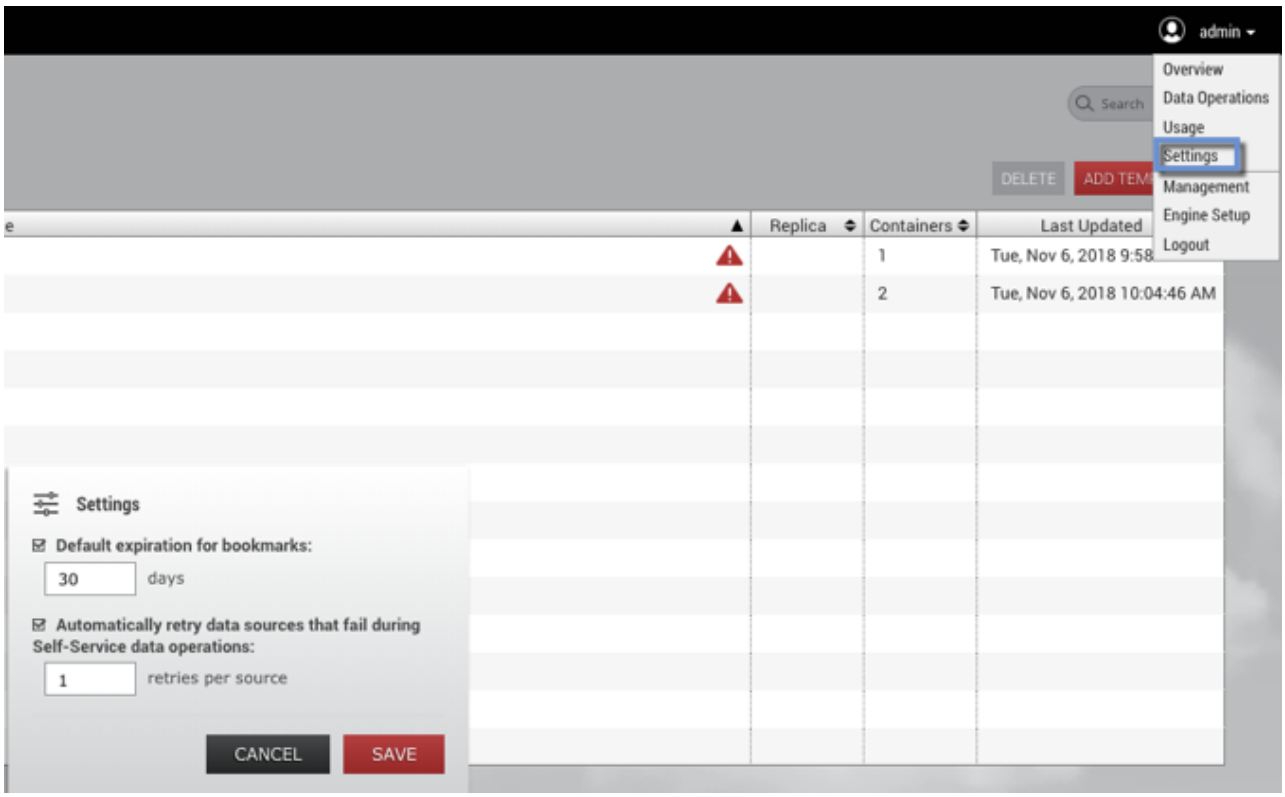
Bookmarks are a way to mark and name a particular moment of data on a timeline. You can restore the active branch's timeline to the moment of data marked with a bookmark. You can also share bookmarks with other users, which allows them to restore their own active branches to the moment of data in your container. The data represented by a bookmark is protected and will not be deleted until the bookmark is deleted. To help manage the space used by this data, users can set an optional expiration date for a bookmark. At the end of the set date, the bookmark will automatically be deleted. Once created, you can easily locate a bookmark through one of the bookmark viewers in the interface. To understand how to use bookmarks, please refer to the Delphix Self-Service Data User Guide.

Using bookmarks in data templates

An admin user can create a bookmark on a template that will then be automatically shared to all containers created from that template. Additionally, an admin user can create a bookmark on the master template timeline with the point of time you are interested in. The bookmark will always be saved from retention policies, and a new branch can be created from this bookmark.

Default bookmark expiration

You can set a value that controls the default expiration time, in days, for Bookmarks. This setting only applies to new bookmarks that are created through the Delphix Self-Service application, not the CLI or API. Note that this only controls the default selection; users can still disable expiration or pick a different date for a specific bookmark if they wish. This setting is disabled by default.



Understanding Delphix self-service usage management

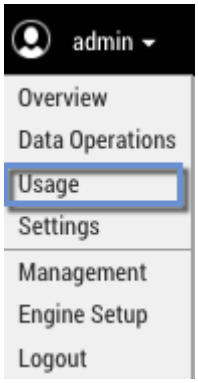
Usage management dashboard overview

Data templates are comprised of dSources, virtual databases (VDBs), and vFiles. These data sources are controlled by the standard policies configured in the **Management** application of the Delphix Engine. As with existing containers, space will be reclaimed by the retention policy over time. As retention cleans up historical data, users will no longer be able to use those points in time to restore or branch. In Delphix Self-Service, an admin can create a bookmark on the data template timeline, which will prevent retention from cleaning up the data that a bookmark references.

Data containers are comprised of VDBs provisioned from the sources defined in the data template. Similar to VDBs in the **Management** application, data containers' VDBs will share blocks with the source from which they are provisioned. This prevents the referenced data on the source from being cleaned up by retention. Retention for these VDBs is controlled by the standard Delphix Engine retention policies. As on templates, bookmarks in data containers will prevent storage from being reclaimed by retention. In addition, Self-Service will ensure that the latest data on each branch is never removed.

The **Usage** pages of the data templates and data containers provide information that can help you understand how storage is being used, how to reclaim space, and how much space you are able to reclaim.

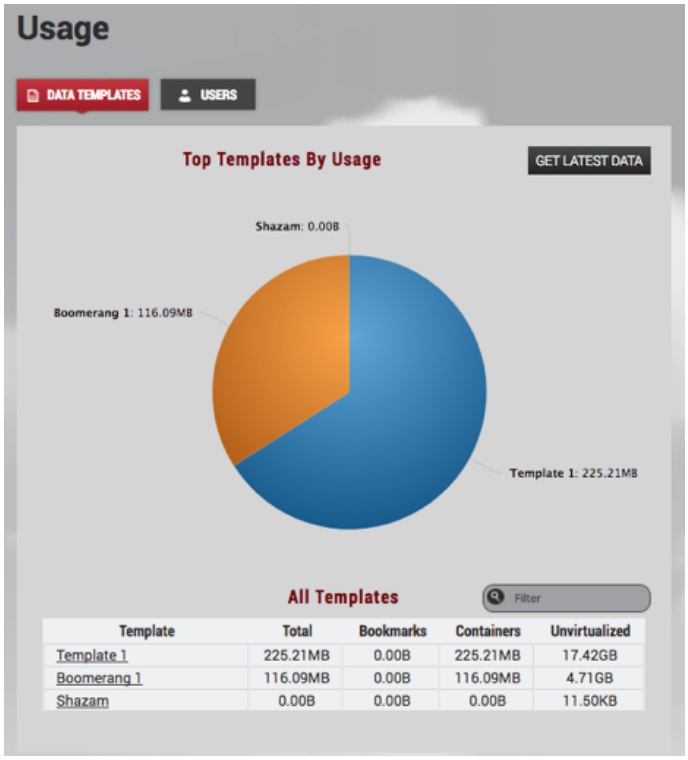
Usage Overview is a top-level page, along with the **Data Operations** and **Overview** pages. It contains the space usage breakdowns by data templates and users.



Template usage overview

The **Template Usage** page, seen in the image below, contains the usage breakdowns for data templates and users. The interface is interactive and allows you to visualize data by interacting with pie charts, bar graphs, and tables. The pie chart contains information about the top 10 space consumers; the table at the bottom contains information about all of the templates and/or users.

The table below the charts includes category fields. You can find corresponding descriptions by hovering over the names of the fields in the table:



The Template Usage page

Additionally, the table allows you to sort, navigate, and interact by clicking the field category of interest. For example, to sort the table, click a **column header** such as **Unvirtualized** and the table will sort by that category. To navigate to a particular data template or user, you can click either the **pie slice** or the **name** of the template/user in the table.

Table of templates/users

The field categories display the following information:

- **Total** – The sum of the space used by the data containers provisioned from this data template and by the bookmarks created on this template. This is the space that will be freed if you delete the template.
- **Containers** – The amount of space used by the data containers provisioned from this data template. This is the space that will be freed if you delete or purge all of the data containers.
- **Bookmarks** – The amount of space used by the bookmarks on this data template. This is the space that will be freed if you delete all bookmarks on the template.
- **Unvirtualized** – The amount of space that would be used by the data in this template and its child data containers without Continuous Data.

The pie chart and table graphs can help you analyze storage usage information.

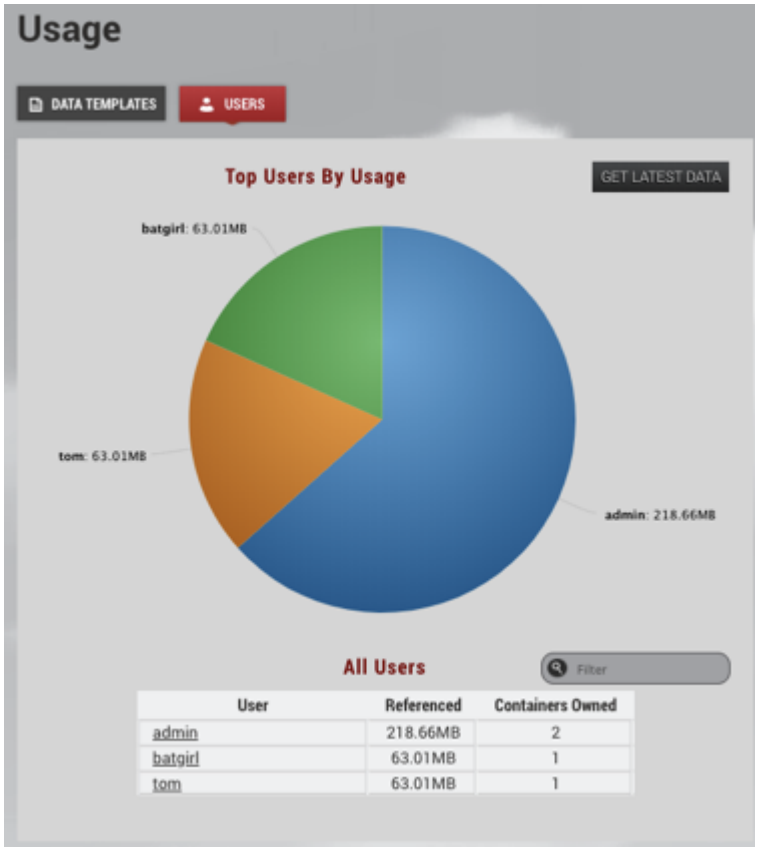
Template usage details

The **Usage** tile appears at the bottom of the Self-Service navigation sidebar, as seen in the image below. Usage summaries are available for templates, containers, and users. For example, when you click the **Usage** tile on the **Template Details** page, the usage details you interact with will be in the context of the selected data template.

The Usage tile

User usage overview

The **User Usage Overview** page provides graphical visualizations of space used by users assigned to data containers.



The field categories display the following information:

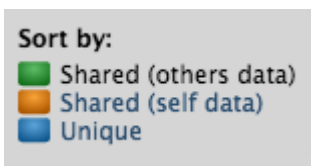
- **Referenced** – The amount of space used by data containers that are owned by this user. This excludes the space that this user is sharing with other users.
- **Containers Owned** – The number of data containers owned by this user

Template usage (Containers) overview

The **Template Usage Details** page, as seen below, shows the space used by data containers provisioned from the template and the bookmarks created on the template.

Container Usage

The stacked bar graph shows information about the top 10 space users. You can re-sort the graph based on the fields in the **Sort by** the legend on the top right-hand corner of the screen as seen in the image above. For example, if you want to know which data containers are sharing the most data with others, you can un-select **Shared (others data)** and **Unique** by clicking them in the legend.



The Sort by legend

> Legend Items When the legend items are not selected, their corresponding colored boxes turn gray and the data is removed from the chart. The data and name will reappear when you re-select by clicking the grayed-out category you want.

The field categories display the following information:

- **Unique** – The amount of space that will be freed if you delete this data container. This assumes that it also deletes underlying data sources.
- **Shared (others data)** – The amount of space that cannot be freed on the parent data template (or sibling data containers) because it is also being referenced by this data container due to Restore or Create Branch operations. The snapshots on the template or sibling container are what use up space.
- **Shared (self data)** – The amount of space that cannot be freed on this data container because it is also being referenced by sibling data containers due to Restore or Create Branch operations, via shared bookmarks
- **Unvirtualized** – The amount of space that would be used by the data in this container without Continuous Data

Template usage (Bookmarks) overview

As shown in the image below, the **Template Usage Details** page provides the usage information about bookmarks created on a template. The primary categories of information include **Unique**, **Shared (others data)** and **Shared (self data)**.

Template Usage (Bookmarks)

The field categories display the following information:

- **Unique** – The amount of space that will be freed if you delete this bookmark
- **Shared** – The amount of space referenced by this bookmark that cannot be freed by deleting this bookmark because it is also referenced by neighboring bookmarks or branches that have been created or restored from this bookmark
- **Externally Referenced** – The amount of space referenced by this bookmark that cannot be freed by deleting this bookmark because it is also being referenced outside of Self-Service – for example, by a retention policy.

Container usage (Branches) overview

The **Container Usage Details** page shows the usage information about the branches and bookmarks created on a container. The primary categories of information include **Unique**, **Shared (others data)**, and **Shared (self data)**.

The Container Usage Details page

The field categories display the following information:

- **Unique** – The amount of space that will be freed if you delete this branch
- **Shared (others data)** – The amount of space that cannot be freed on the parent data template or sibling branches because it is also being referenced by this branch due to Restore or Create Branch operations. The snapshots on the template or sibling container are what use up space.
- **Shared (self data)** – The amount of space that cannot be freed on this branch because it is also being referenced by sibling data containers due to Restore or Create Branch operations, via shared bookmarks.

Delphix self-service data user guide

Welcome to Delphix self-service

Delphix Self-Service grants access to the data that users need, whenever they need it. Once users have been assigned a data container, they can control the data available within it. This means they can refresh to the latest production data, roll bac...

Updated on : 25 May 2023

Delphix self-service data concepts

Understanding Data Sources A data source can be a database, an application, or a set of unstructured files. Engine Administrators configure the Delphix Engine to link to data sources, which pulls in the data of these sources. The Delphix Engine ...

Updated on : 25 May 2023

Delphix self-service user interface

The User Interface is organized within a single web browser page. The screen serves as a data container report and management panel. Data Container Report Panel Data Container Workspace Data Container Report Panel The Data Container Re...

Updated on : 25 May 2023

Understanding timelines and how to preserve data in a point in time

Understanding Timelines Branch Timeline A branch timeline acts as a dynamic point-in-time interface for user actions within the branch. You can interact with the source data in the active branch by using both the branch timeline and icons along ...

Updated on : 25 May 2023

Data container activities

Getting Started Data containers can be shared between multiple users. In this situation, users should coordinate with their co-owners when performing data operations that could disrupt other users' workflow such as stopping or refreshing the data c...

Updated on : 25 May 2023

Containers with multiple owners

Delphix Self-Service administrators can designate multiple users as owners of a single data container. These users all share access to the same data container which means actions taken by one user will impact all users on the same data container. F...

Updated on : 25 May 2023

Working with bookmarks in a data container

Working with bookmarks is an easy way to share data with other users of any container created from the same template. By sharing with others, you can integrate testing, development, and QA needs. For example, in the past, if you found a bug you woul...

Updated on : 25 May 2023

Understanding Delphix self-service usage

Usage Management Dashboard Overview Data templates are comprised of dSources, virtual databases (VDBs), and vFiles. These data sources are controlled by the standard policies configured in the Management application of the Delphix Engine. As w...

Updated on : 25 May 2023

Welcome to delphix self-service

Delphix Self-Service grants access to the data that users need, whenever they need it. Once users have been assigned a data container, they can control the data available within it. This means they can refresh to the latest production data, roll back to a previous point in the data container's timeline, and share data with another user without requiring any involvement from Information Technology or database administrators (DBAs). Delphix Self-service data management allows developers to be more productive while using fewer resources, dramatically improving operational efficiency.

User roles and permissions

Self-Service has two types of users:

Admin user

Admin users have full access to all report data and can configure Delphix Self-Service. Additionally, they can use the Delphix Engine to add/delete reports, add/delete users, change tunable settings, add/delete tags, and create and assign data templates and containers.

Data user

Data users have access to production data provided in a data container. The data container provides these users with a playground in which to work with data using the self-service toolbar.

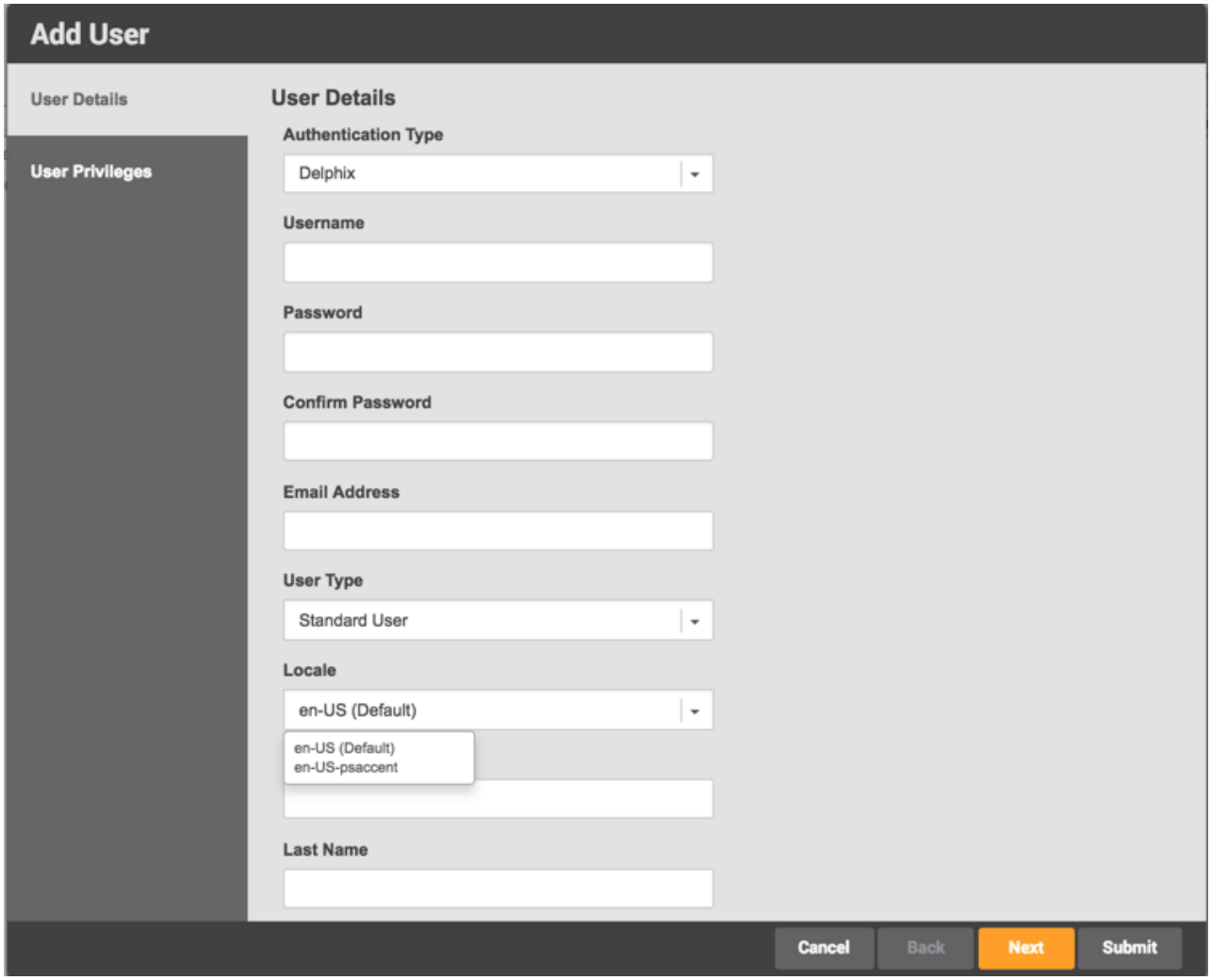
Login

1. Access Delphix Self-Service by opening a web browser using the **IP address** or **DNS qualified hostname**.
2. Login with the **User ID** and **Password** the Delphix Administrator has provided for you.

Changing your default locale

As a user, you can change your default locale by doing the following:

1. Click the **user login** icon in the upper right-hand corner of the screen.
2. Click the **Locale** drop-down menu.
3. Select the desired locale.



User Profile window

Delphix self-service data concepts

Understanding data sources

A data source can be a database, an application, or a set of unstructured files. Engine Administrators configure the Delphix Engine to link to data sources, which pulls in the data of these sources. The Delphix Engine will periodically pull in new changes to the data, based on a specific policy. This, in turn, begins building a custom timeline for each data source. Additionally, the Delphix Engine can rapidly provision new data sources that are space-efficient copies, allowing users to work in parallel without impacting each other.

Understanding data templates

Data templates are the backbone of data containers. They are created by the Engine Administrator and consist of the data sources that users need in order to manage their data playground and their testing and/or development environments. Data templates serve as the parent for a set of data containers that the administrator assigns to users. Additionally, data templates enforce the boundaries for how data is shared. Data can only be shared directly with other users whose containers were created from the same parent data template.

Understanding data containers

A data container allows data users to access and manage their data in powerful ways. Their data can consist of application binaries, supporting information, and even the entire database(s) that underlie it.

A data container allows users to:

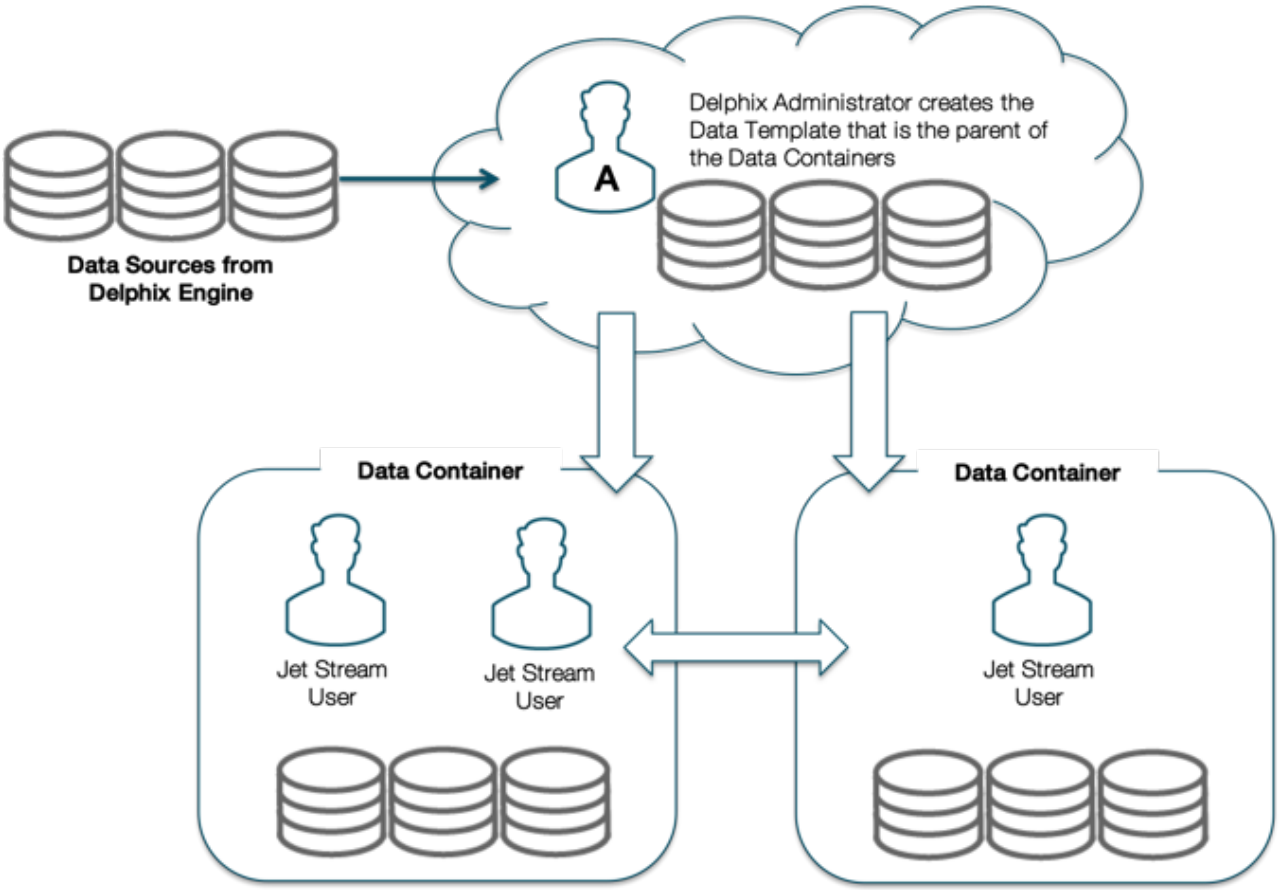
- Undo any changes to their application data in seconds or minutes
- Have immediate access to any version of their data over the course of their project
- Share their data with other people on their team, without needing to relinquish control of their own container
- Refresh their data from production data without waiting for an overworked DBA

A data container consists of one or more data sources, such as databases, application binaries, or other application data. The user controls the data made available by these data sources. Just like data sources in a template, changes that the user makes will be tracked, providing the user with their own data history.

The **Data Container Interface** lets users view the details and status of their data container and its associated data sources, as well as manipulating which data is in those sources. The **Data Container Interface** includes a section called the **Data Container Report Panel**, which displays details about each source, including the connection information needed to access it – for example, the java database connectivity (JDBC) string for a database. This connection of information is persistent and stable for the life of the data container, regardless of what data the resources are hosting.

Data flow

The Delphix Self-Service data flow diagram below demonstrates how a Delphix Self-Service data user accesses data sources. Data sources are connected to a Delphix Engine, which is controlled by the Engine Administrator. The Engine Administrator will connect all data sources that developers and quality assurance (QA) teams need to a data template. This data template acts as a parent source to create the data containers that the administrator will assign to data users. Data sources flow from the Delphix Engine into a data template and downstream into a data container, where a data user or users will use the data sources to complete tasks. The data container acts as a self-contained testing environment and playground for the data user. Additionally, data users are able to set, bookmark, and share data points in their container with other data users of other data containers, as long as all the data containers were created from the same parent data template.



Data Flow

Understanding branches

You can organize data in the data container into task-specific groupings, called "branches." For example, you can use a branch to group all the data you have used while addressing a particular bug, testing a new feature in an application, or exploring a business analytics scenario. By default, Delphix Self-Service automatically creates the first branch of source data for you when you login for the first time. You can view the default branch and any additional branches that you create over time by clicking the Branch tab. Additionally, to the right of the default branch, you will see an interconnected branch timeline unique to whichever branch is currently active. The illustration below displays both the default branch in the Branch tab of the Data Container View Panel and the default branch timeline.

Branch Tab

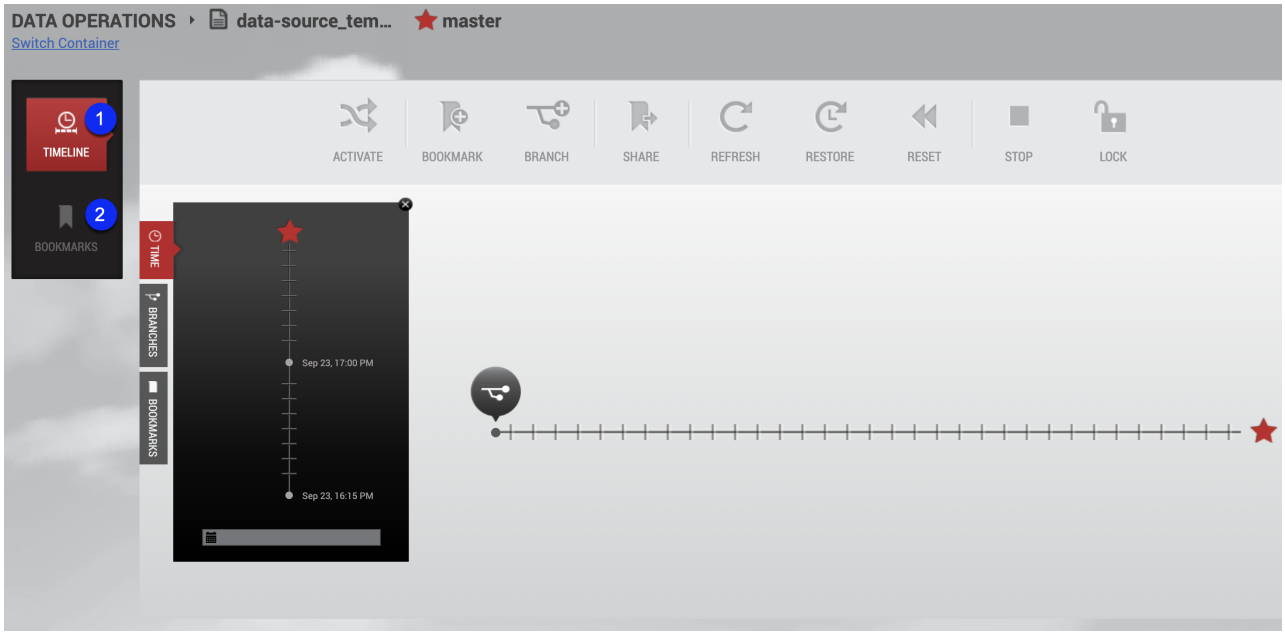
Branch View Panel and Branch Timeline

A branch is used to track a logical task and contains a timeline of the historical data for that task. One branch is the "active" branch, which means that it is the branch that is currently being updated with new data from the data sources. At any time, you can change which branch is active and thus change which data is in the associated data sources.

Delphix self-service user interface

The User Interface is organized within a single web browser page. The screen serves as a data container report and management panel.

Data container report panel



Data Container Workspace

Data container report panel

The Data Container Report Panel for users consists of two tile buttons. They are summarized below as **Timeline** and **Bookmarks**.

Timeline

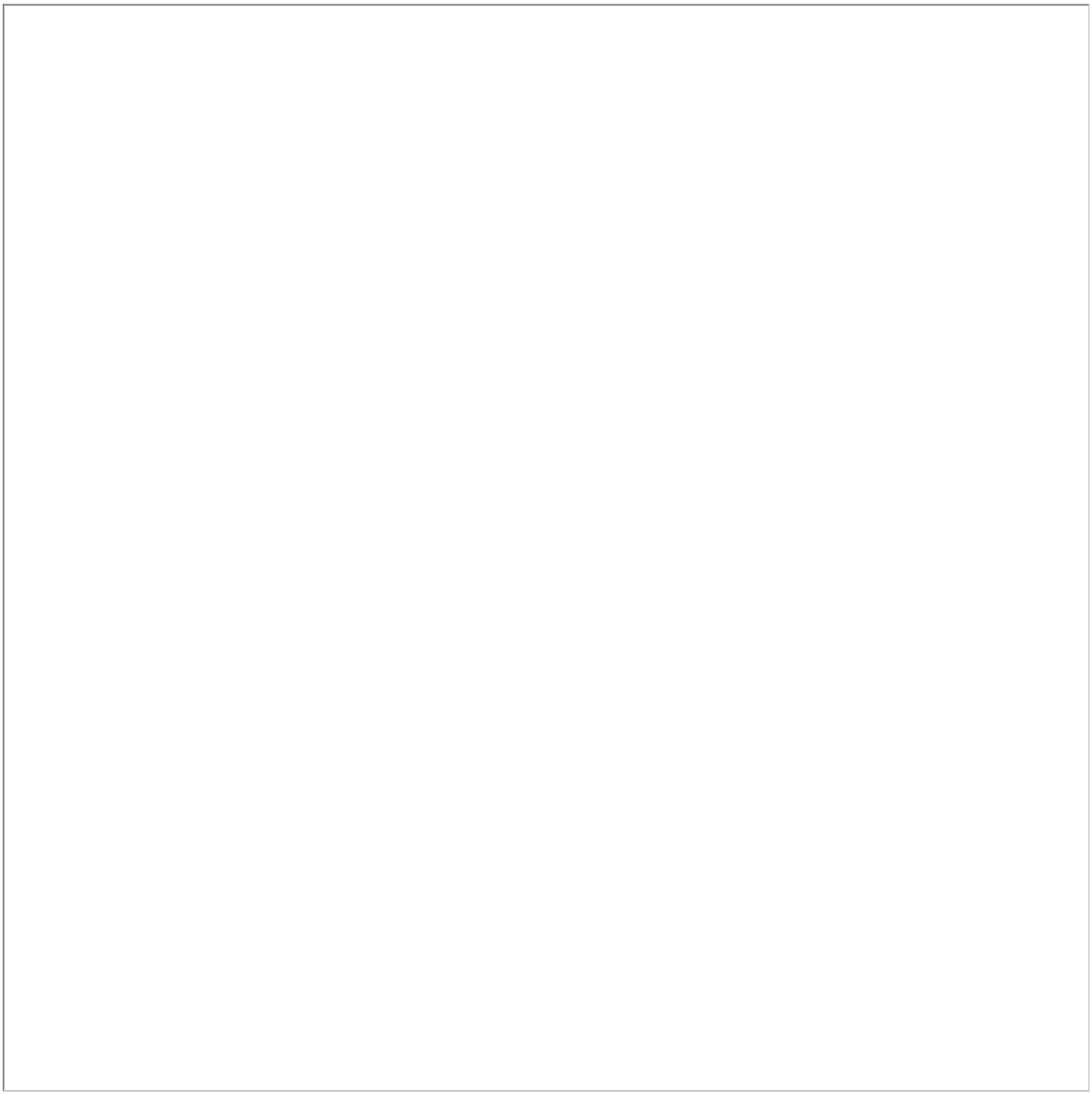
The **Timeline** tile allows you to view the timeline associated with a branch. Note that this only shows the timeline for a single branch. The branch timeline is how a user interacts with data in the container to mark, stamp, and perform tasks that occur at various points in time.

Bookmarks

The **Bookmarks** tile allows you to view and edit details about bookmarks within this data container and bookmarks accessible from it.

Data container workspace

The Data Container workspace is reached by selecting a template from the Management Overview page. Select a container and then select the Data Management link.



Data Container Workspace (Top Half)

Data container workspace

The Data Container Workspace contains all the tools, actions, and view panels needed to begin using Self-Service features. For example, the workspace allows a user to view the history of their data on a branch, and to refresh, reset, and restore that data.

User login and settings drop down menu

 The **user login** icon in the upper right-hand corner of the screen provides a drop-down menu with options to change your password and/or log out.

Data container view panel

 The **Data Container View** Panel, found on the left-hand side of the screen, is divided into three tabular sections: time, branches, and bookmarks. These tabs allow you to find and select data that you are interested in. Based on user selections made in the view panel, the corresponding branch timeline can change.

Switch container

 Allows you to switch between containers.

Data container self-service toolbar

 The Data Container Self-Service Toolbar allows you to perform tasks and activities with data in the current container, by clicking on the following user action icons:

- **Activate** will make a branch active
- **Bookmark** will mark an interesting point of data on a branch timeline
- **Branch** will create a branch that supports one task. A branch is a group of data time segments called a "timeline."
- **Share** will share a bookmark with users of other data containers from the same template
- **Refresh** will refresh each source in the data container on a branch timeline to the latest data in the corresponding source of the data template.
- **Restore** will restore the data to a point in time from the template, the container, or a shared bookmark.
- **Reset** will reset to the last interesting moment of data time on the current data timeline
- **Stop** will stop a data container
- **Start** will start a data container
- **Lock** will lock a data container for the current user
- **Unlock** will unlock a data container.

Branch timeline

 Use this to view the timeline associated with a branch. Note that this only shows the timeline for a single branch. The branch timeline is how a user interacts with data in the container to mark, stamp, and perform tasks that occur at various points in time.

Understanding timelines and how to preserve data in a point in time

Understanding timelines

Branch timeline

A branch timeline acts as a dynamic point-in-time interface for user actions within the branch. You can interact with the source data in the active branch by using both the branch timeline and icons along the **Self-Service Toolbar** at specific points in time. Common activities include re-setting data sources to run a test, refreshing the data container with the most current source data, and bookmarking data to share or track interesting moments of time along the branch timeline. Users work with one branch at a time to perform a series of actions related to a particular testing or debugging task such as data updates or starting and stopping data. The initial branch on your data container may include data from before the data sources were made into a container. As you work within your data container, you can create more branches overtime to run or complete separate tasks. Additionally, the data container tracks each branch and the corresponding actions you perform on the branches. To view the actions completed over the life of a branch, see the container timeline in the **Time** tab of the **Data Container View Panel**.

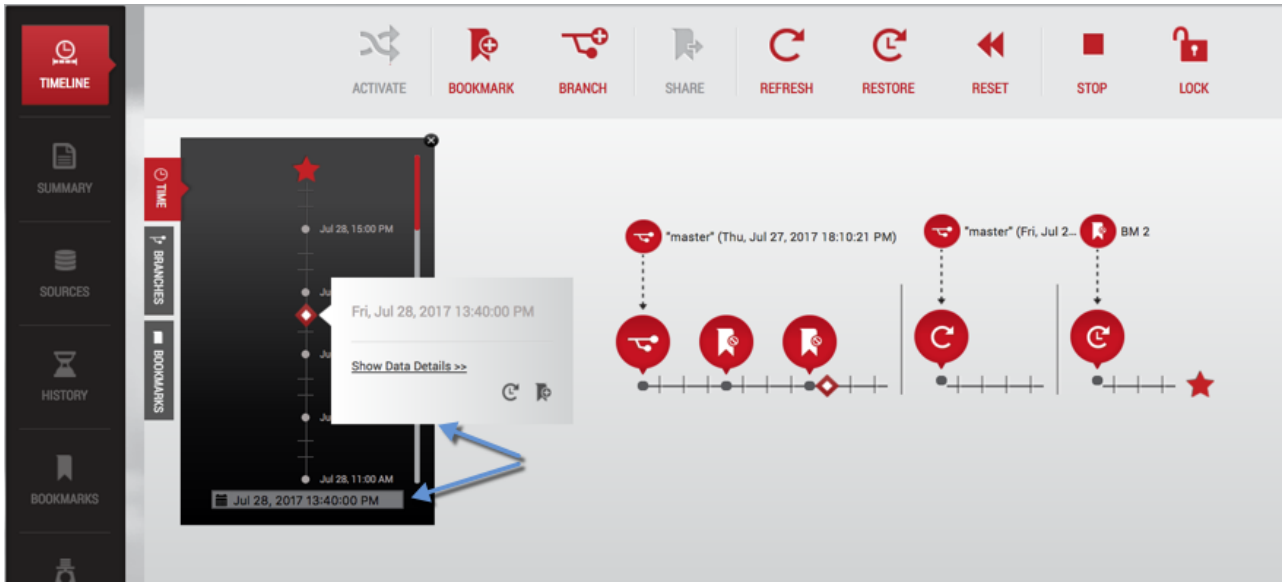
Branch with Timeline Segments Over the Life of the Branch

Container Timeline

The **Time** tab displays the data container's timeline, which acts as a wall clock of time. It shows continuous real-time across all branches and timeline segments. You can scroll up and down in the container timeline to find the point in time that interests you.

Time Tab Timeline

Clicking a point in time in the container timeline will display the corresponding branch timeline capturing any actions performed on the branch. Additionally, should you need to select a time between tick-marks, you can use the **time input field** in the time selector on the left side of the screen.



Container Timeline

Selecting a Point in Time with the Time Selector

1. In the **time selector**, type in a **date** and **time** with the following format: Month/Day/Year Hour:Minute:Second{am|pm}. For example: 1/26/2015 1:14:13pm.
2. Press **Enter**.

The time input field will show the selected time. Now that you have entered the specific time you want, you can use the toolbar to select the data operation that you want to be performed at this point in time. Data operations can include Create Bookmark, Create Branch, and Restore.

> ⚠ Invalid Time Value
> If you type in an invalid time value, or a time that is out of range, the value you typed in will revert to the previous default that existed before.

Selecting a Point in Time with the Time Selector Calendar

1. In the time selector, click the **calendar** icon to the left of the input field.
2. From the window that appears, click the **date** you want to use.
3. Select and **time** you want to use.
4. In the toolbar, click the button for the data operation that you want to perform at this point in time. Data operations can include Reset, Create Branch, and Create a Bookmark.

> ⓘ Picking a date
> The flyout will not let you pick a date that is before the first point of data time in the container, or after the present moment.

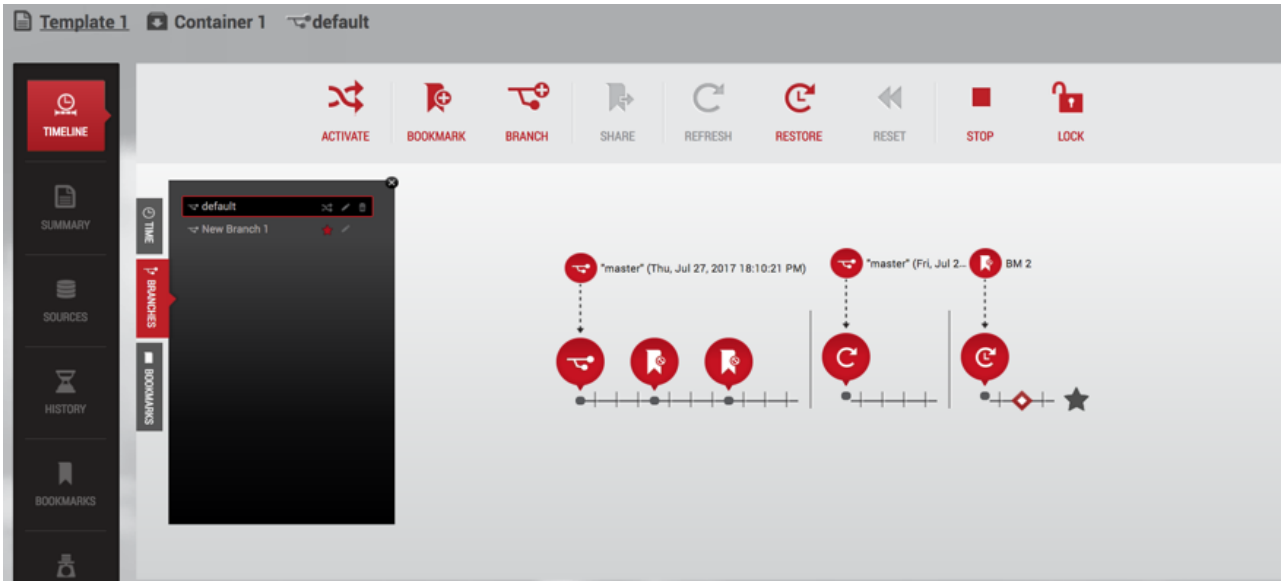
Understanding the self-service toolbar

The **Self-Service Toolbar** contains self-service action icons that represent available actions a data user can perform. You can distinguish between available and unavailable icon actions by the use of color on the toolbar. Actions available to you will be red, and actions that are unavailable will be grey. All actions are dynamic, and

availability will change based on how you use and work with data in both the branches and data container(s) that are assigned to you.

Self-Service Toolbar

For example, your options for actions on the **Self-Service Toolbar** can change if the branch of the branch timeline you are working with is activated. In the illustration below, the screen shows a user working in an active branch. Notice the bright red star at the end of the timeline. This indicates that the branch is active. Also, notice which actions are and are not available to the user on the **Self-Service Toolbar**.

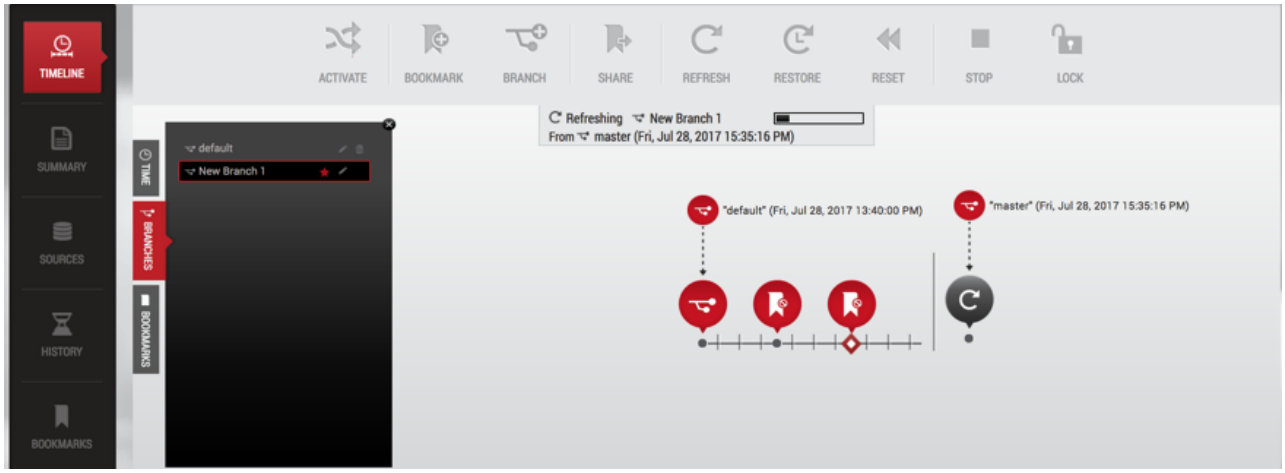


Self Service Toolbar with a Point in Time selected on an Active Branch Timeline

The **Self-Service Toolbar** is dynamic and will change based on tasks a user performs in Self-Service. These workflows will influence how and when self-service actions become available on the Self-Service Toolbar.

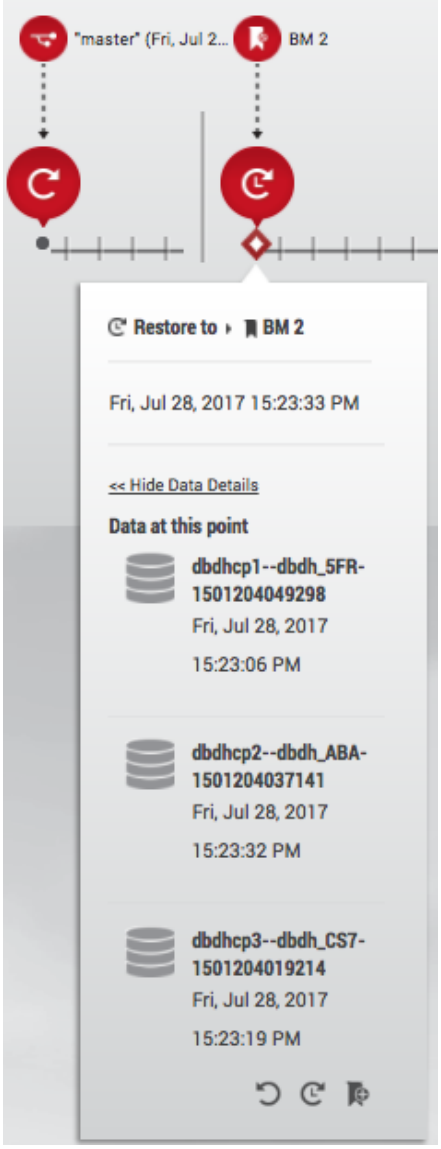
Branch timeline segments

A branch timeline with segments is a visual representation of actions taken on a branch timeline over a time span. The timeline segments represents data in time that is no longer contiguous once a user clicks **Create Branch**, **Refresh**, **Reset**, or **Restore** on the **Self-Service Toolbar**. A vertical bar between each of the segments appears to remind a Delphix Self-Service user that the data in one timeline segment is a completely new data start. In other words, while the data within one segment is logically contiguous, the data is never contiguous across segments. For example, the following image shows a timeline with multiple segments.



Segmented Branch Timeline

As mentioned above, the branch timeline becomes segmented after you have performed a specific action or task, such as **Refresh**. Based on the action, two red bubbles will appear in the time segment. The top bubble indicates where the data used for this action came from – for example, the data template, a different branch, or a shared bookmark. The second red bubble appears on the timeline as the actual data stream in a point in time from the parent data. It appears because of actions such as **Refresh**, **Reset**, **Restore**, **Create Branch** and **Bookmark**. Clicking the second bubble will show you specific details of the action, such as the specifics of the action including its name, the time the action occurred, and the data sources used at a point in time. This is illustrated below.



Parent Data Sources and Child Data Sources

Working with multiple branches and timelines

As you work in your data container, you can switch between branches to work on resolving a bug or to test a new application feature.

For example, consider what occurs on two different branches in a container:

Branch 1:

Branch 1 Timeline

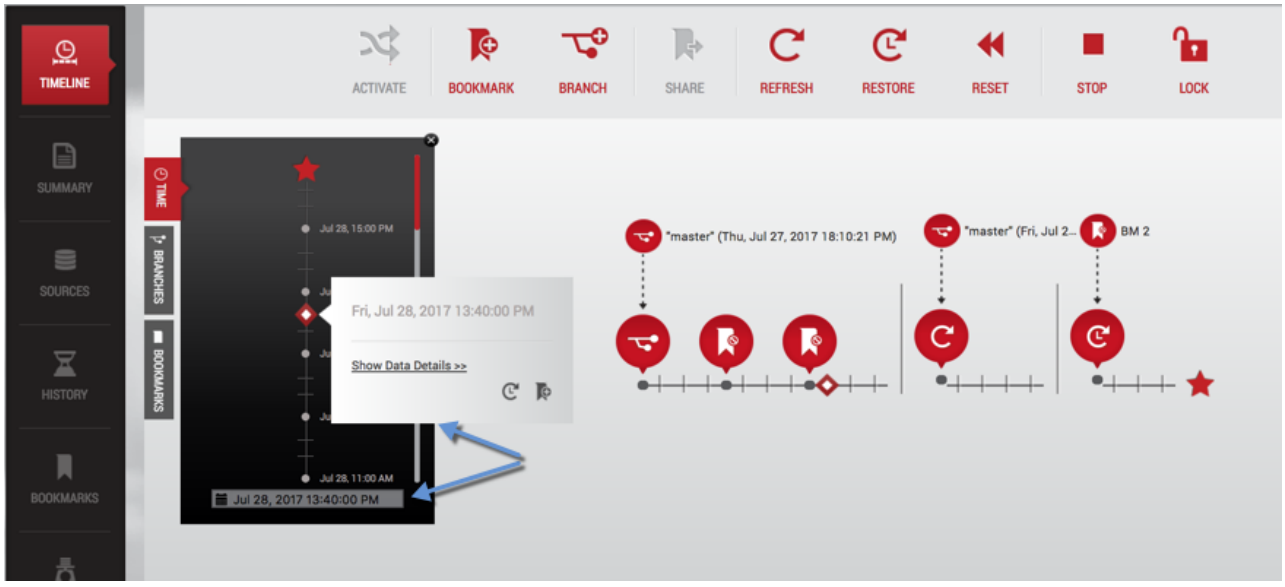
Branch 2:

Branch 2 Timeline

A user may have actually worked with these branches in the following order over time:

Branch 1: Create a branch and use ☐
☐ Branch 2: Create another branch and use
Branch 1: Activate branch, Restore the data source and use ☐
Branch 2: Activate branch and create bookmarks ☐
Branch 2: Refresh the data source from a particular point in time ☐
Branch 2: Reset a branch to the last action (e.g., refresh) on the timeline, and use ☐

In the above illustrations, an individual branch's timeline shows all actions performed on the branch while the branch was active. The active branch timeline can be interrupted and deactivated when you choose to perform actions such as switching to another branch, **Create Branch**, **Activate**, or **Stop** a data container. Additionally, you will only be able to view actions on a single branch at a time. A better way to manage multiple branches is to go to the **Time** tab in the **Data Container View Panel**. The **Time** tab allows you to access the **container timeline**, which becomes useful as you toggle back and forth between branches to complete different tasks. The **container timeline** allows you to view all the continuous data points of time, with all actions taken on all branches in a single data container.



Container Timeline

> ⚠ Merging keystores requires several manual steps. Refer to Migrating a TDE-enabled vPDB for more information.

Understanding how to preserve data in a point in time

The following illustration shows that on 8/27/14, at 9:33:09am, data was reset to the parent data branch (master) at 9:28:48am, capturing data points from 9:33:06am.

Preserve Data in a Point in Time

The black arrows above point to a tick, (representing a point in time) clicked on the branch timeline. This represents the time the **Reset** action was performed on the data container. The red arrows point to when time was captured in a data source using the **Reset** action on the branch timeline. When clicked, the reset bubble provides more details with a flyout, indicating where the data comes from and the time that the data represents. Additionally, the reset bubble detail flip card provides additional information about each data source. Specifically, the blue arrows point to the time used for each data source at this point in the data container.

> ⚠ This does not show the time that was used for each source that pulled the data.

The time represented on the branch timeline varies based on many factors. For example, after you select a specific point in time on the branch timeline, the Delphix Engine will map that point to the closest usable point in time for each data source. Based on the properties of the underlying data sources, these times may be different. Not all data sources track changes at the same granularity, as illustrated below.

Point In Time

While a branch timeline can follow a continuous-time flow, the data sources being selected for each time segment may not be continuous.

Understanding bookmarks

Bookmarks are a way to mark and name a particular moment of data on a timeline that makes it easy to search and find events later. Once a point in time is selected, you can create a bookmark. You can select a specific point in time on the branch timeline, the Delphix Engine will then map that point to the closest usable point in time for each data source. Based on the properties of the underlying data sources, time may not be the same for every data source in the container. Not all data sources track changes at the same granularity. For more information, see Working with Bookmarks in a Data Container.

> **Note**
> A bookmark captures the latest point-in-time (not exceeding the bookmark timestamp) that is available while creating the bookmark. When you access the bookmark later, it does not get a closer time even if new logs are available.

Once created, you can easily locate a bookmark through one of the bookmark viewers in the interface.

Bookmarks tab in the data container view panel

The **Bookmarks** tab is the third tab in the **Data Container View Panel** within the data container workspace of the Delphix Self-Service interface. It allows you to find a bookmark that is within your data container and view the branch where the bookmark has been placed.

Bookmark Tab

Bookmarks tile in the data container report Panel

The **Bookmarks** tile in the **Data Container Report Panel** allows you to see all bookmarks within your container and all bookmarks that other users have made available to you. Here you can also edit details about bookmarks, create new branches, and restore the active branch to the bookmark's point of data time.

Bookmark Tile

Bookmark sharing permissions

When you first create them, bookmarks are private to your data container, but you can share a bookmark with other data users. Bookmarks that other users have shared with you are called "available" bookmarks.
Your bookmarks will only be shared with data users in data containers created from the same data template. This is because all data containers created from the same data template have a compatible set of data sources.

Bookmark appearance

	A bookmark that is private
	A bookmark you have shared
	A bookmark that has been shared with you

Data container storage and retention for branches and timelines

Bookmarks mark a moment of data. Delphix Self-Service will never automatically delete the data marked by a bookmark. However, Delphix Self-Service will automatically delete a bookmark with an expiration date set after it has expired. For more information on setting or removing an expiration date, see Data Container Activities. Delphix Self-Service may delete data from any time in the past on your branches, depending on the retention policies configured by your administrator. If you select a moment of data that has been deleted, the flyout will indicate that retention has removed data for this point in time.

Data Container and Retention

Data container activities

Getting started

Data containers can be shared between multiple users. In this situation, users should coordinate with their co-owners when performing data operations that could disrupt other users' workflow such as stopping or refreshing the data container.

Activity One: how to start and stop a data container

Starting a data container does the following:

- Starts the data sources
    - This means that each data source listed in the **Source Details** section of the **Data Container** page will start using CPU and network resources on the target system it is running on
- Makes the data in the active branch available
    - Once the container has been started, the data represented by the active branch is available

Stopping a Data Container does the following:

- Shuts down the data sources
    - This means each data source listed in the **Source Details** section of the **Data Container** page will stop using CPU and network resources on the target system.

To start a data container, click **Start** on the **Self-Service Toolbar**.

To stop a data container, click **Stop** on the **Self-Service Toolbar**.

Working with a branch, a branch timeline, and the self-service toolbar

Activity Two: using reset from a bookmark to facilitate destructive testing

Reset is a data user workflow that is optimized to enable destructive testing. **Reset** automatically restores the data to the last operation conducted in the data container, which can include creating a bookmark, resetting, or restoring data. As an example, you can do a refresh and then get your data into a state required for testing. Once you are satisfied with the state of your data, you can create a bookmark, which will preserve the data at this point in time.

Afterward, you can then run destructive tests on the data. When you are done, you can click the **Reset** icon, which will automatically restore the state of the container to the last operation – in this case, the bookmark. This workflow ensures that each test has a clean copy of the data and is not impacted by the results of other tests. You only need to create a bookmark and click **Reset** on the **Self-Service Toolbar**.

Create a bookmark

1. Select a **Data Point** on a branch's timeline.
2. On the **Self-Service Toolbar**, click the **Bookmark** icon.
3. In the **Bookmark Window**, enter a new **name**.

   Name a Created Bookmark
4. Optionally, fill in a **description.**
5. Optionally, set an expiration date. The bookmark will be automatically deleted at the end of this day.
6. Optionally, add one or more **tags**.
   You can use these to help filter a set of bookmarks.

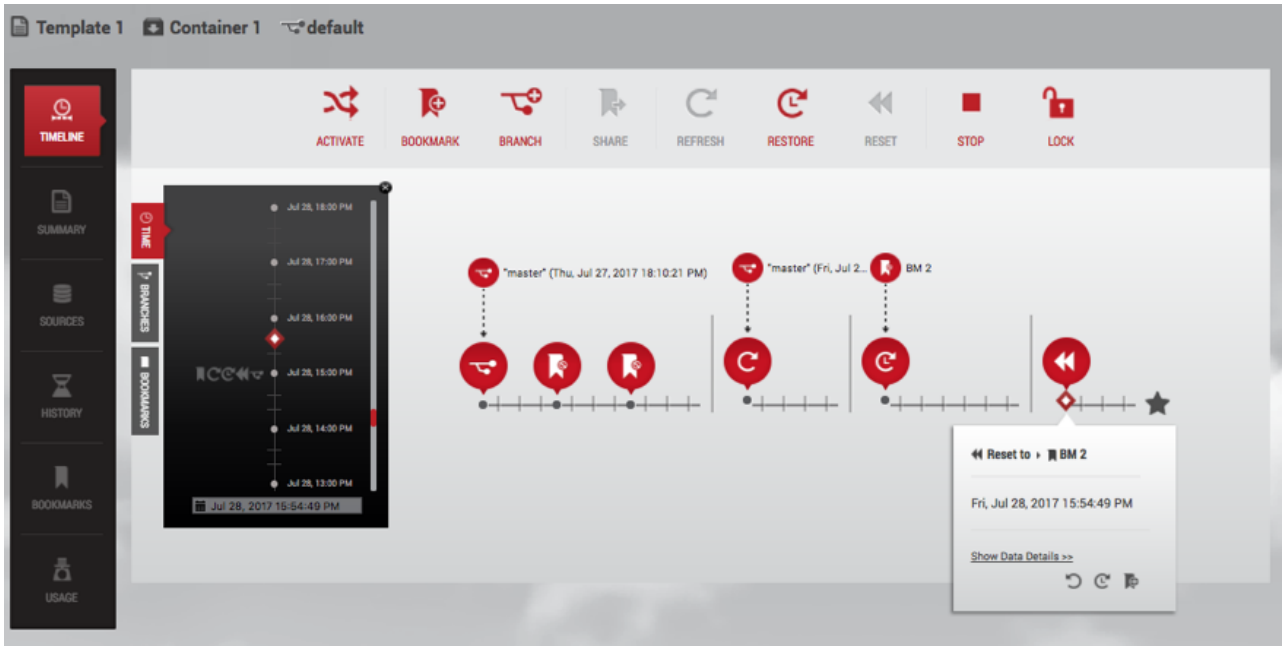
7. Click **Create**.

After the bookmark has been created, you will see the **bookmark** icon appear on the timeline. When you click the **Reset** button, all data will be reset to that point in time.

Reset to data from a bookmark

Click the **Reset** icon.

This action reflects the moment of data marked by the closest operation bubble (**Refresh**, **Restore**, **Reset**, or **Bookmark**) into a new timeline segment on the active branch. It also copies the moment of data into the data sources.



Update Data with Reset

Undo data

Click the **Undo Data** icon for an operation.

Undo the given operation. This is only valid for RESET, RESTORE, UNDO, and REFRESH operations.

Activity Three: using refresh to get the latest data from a data template

1. Start a new timeline segment with the most recent point of data from the data container's data template.
2. Click the **Refresh** icon.

Refresh creates a new timeline segment on the active branch. This refreshes each source in the data container to the latest data in the corresponding source of the data template.

Update Data with Refresh

Activity Four: using restore to return data back to a point in time

This starts a new timeline segment on the active branch with the selected point of data.

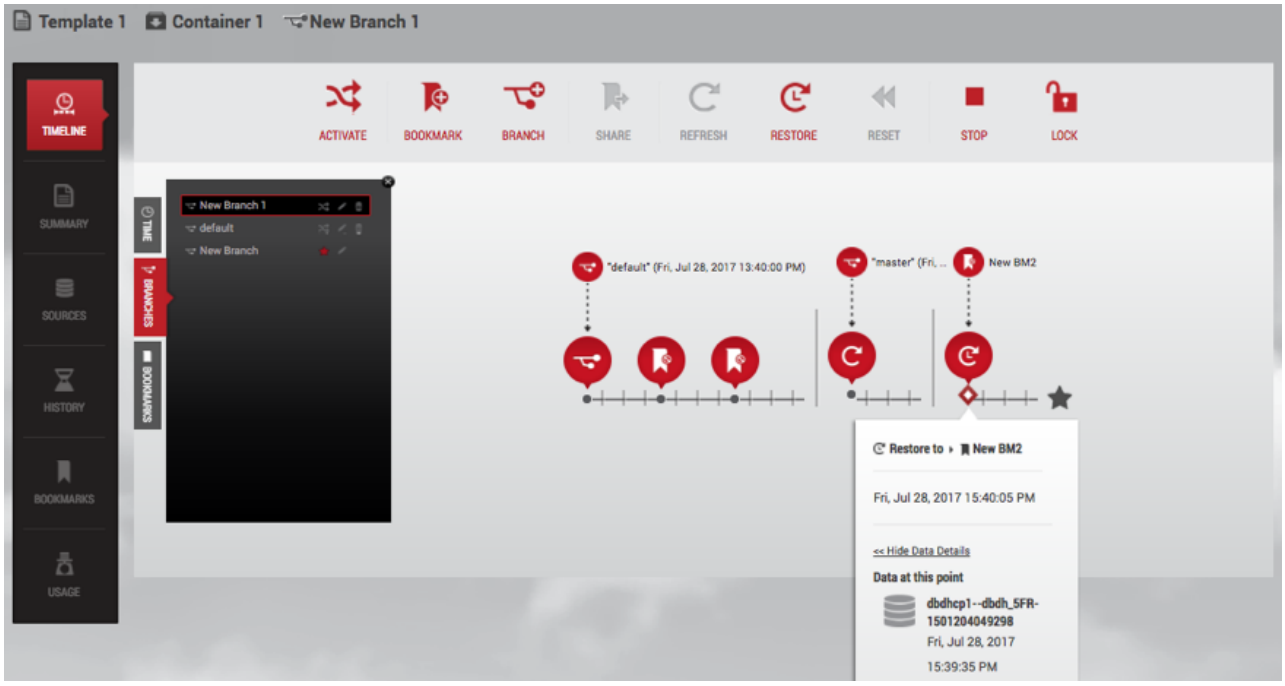
1. Select one of the following:
    a. A **point of data** on a timeline.
    b. A **bookmark** on a timeline.
    c. A **bookmark** under the **Bookmarks** tile in the **Data Container Report Panel**.
2. Click the **Restore** icon.

If you restore data back to a point in time on the data template master timeline, you will be asked which data container to restore into. It will then:

- Reflect the selected point of data into a new timeline segment on the active branch
- Copy the moment of data into the data sources

If the timeline segment on a branch timeline was created by a **Restore** operation, then the segment starts with the moment of data from the branch that was selected when the **Restore** operation was done. This is illustrated below.

> Parent Branch
> The parent branch for this segment can be the same branch of which this segment is a part. It is possible to restore the active branch from a point in time on the same branch.



Update Data with Restore

> Source Branch
> The source branch for this segment can be the same branch of which this segment is a part. It is possible to restore the active branch from a point in time on the same branch.

Activity Five: restoring to a point on the parent template

Data templates serve as the parent for a set of data containers, and as a data user, you have the flexibility to restore your container to any point on the template.

1. Choose the container tab and select the template by clicking on the template name located at the top of the screen.
2. Select one of the following:
    - A point of data on the timeline
    - A bookmark on the timeline
    - A bookmark under the Bookmarks tile in the Data Container Report Panel

3. Click the restore icon

4. A dialog will pop up. Use it to select the container you'd like to restore.

force option

forceOption is an API/CLI-only feature. Generally, if a source database is corrupted or otherwise prevents taking VDB snapshots, the Refresh, Reset, and Restore actions cannot be completed. With **forceOption**, you can bypass taking a pre-operation screenshot and proceed with the desired action.

Because **forceOption** does not take a snapshot of the VDB before refreshing/resetting/restoring, you cannot undo the operation afterward.

From the CLI:

1. Go to the Delphix Self-Service container endpoint.

2. Select the container.
3. Attempt the execute the operation.
4. Set **forceOption** parameter to **true**.
5. Commit the change.

The operation will now perform without taking a snapshot first.

Activity Six: create a new branch and switch between branches

Developers and QA teams can have multiple branches that can represent data from different points in time or different sources. You have many options for how you create a new branch. These include:

- A **point of data** time on a data timeline within the Delphix Self-Service data container, or
- A **bookmark** bubble on the timeline, or
- A **bookmark** in the **Bookmarks** tile in the **Data Container Report Panel**

1. Click the **Branch**  icon to create a new branch.
2. Enter a **name** for the new branch.
3. Click **OK**.
4. On the **Self-Service Toolbar**, click the **Activate** icon. 

If the inactive branch is not showing in the data container workspace:

1. Find the **branch** in the **Branch** tab.
   
   Selection of Branches in Branch Tab
2. Click the **Activate**  icon.
3. After a moment, the branch will become active.

Active branch

Within a single data container, only one branch is active at any given time. The data located at the red star of the active branch's timeline is the newest copy of the data from the data container's data sources.
The active branch is distinguished by a red star, which appears at the far right of the timeline, alongside its name in the **Branch Name** area, and in the **Branch** tab.

	
Active branch	Inactive branch

Activity Seven: rename and/or delete a branch

Rename the Default Branch

1. Select the **Default Branch** in the **Branch** tab.
2. Click the **Pencil** icon to the right of the name.
3. Enter the **new name**.
4. Click the **Checkmark** icon.

Delete a created branch

1. Select the **branch** in the **Branch** tab.

2. Click the **Delete** icon to the right of the name.
3. Click **Delete** in the confirmation window that appears.

Activity Eight: restoring a data container to a consistent state with the recovery operation

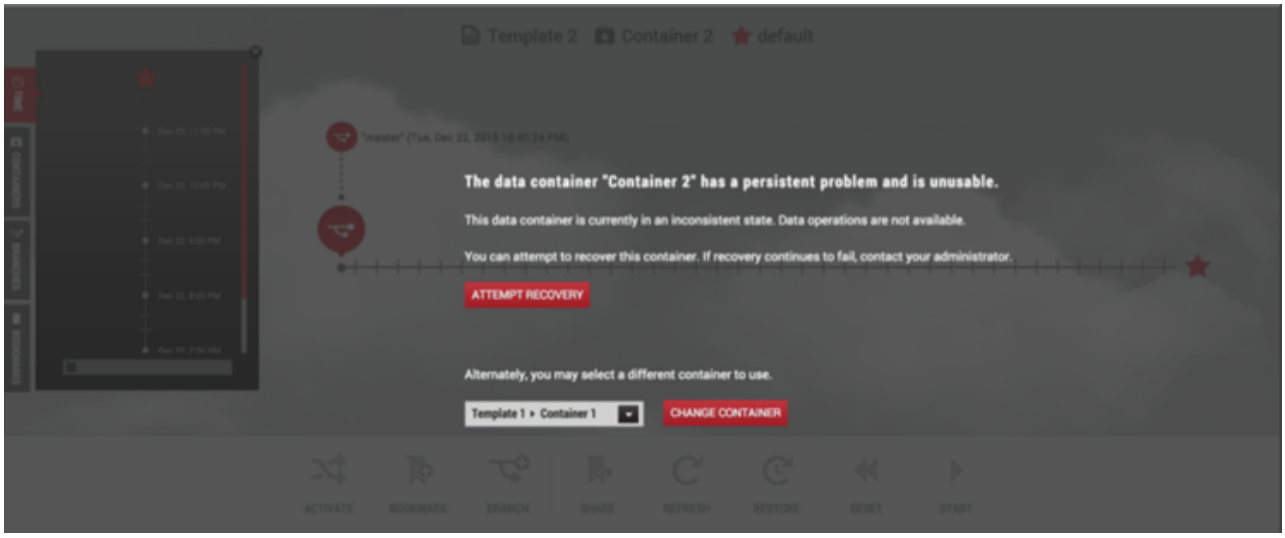
Data containers consistency

Delphix Self-Service allows you to group multiple datasets in the same data container. This makes it easy for you to access entire applications such as PeopleSoft, including binaries and code. If a data container represents an application, then there are likely to be dependencies between the application's datasets. For example, the vFiles data source containing the code will depend on a specific version of the database's schema. Therefore, it is important that all dataset sources are drawn from the same point in time. If they are, the data container is in a "consistent" state; if they are out of sync, or "inconsistent," errors will occur. For example, if the vFiles data source containing the code has been updated more recently than the database's schema, the dependency cannot work.

Delphix Self-Service currently has no way to determine whether the application is consistent. However, it attempts to minimize the chance that dataset sources are out of sync whenever it performs a data operation such as refresh, restore, or reset. When performing a data operation, Delphix Self-Service attempts to snapshot all dataset sources from a point in time as close as possible to the desired time. If at least one of the data sources fails to go to the desired point, then Delphix Self-Service considers the data container to be in an inconsistent state. The application as a whole may still be working, but Delphix Self-Service assumes that the failed dataset's data is not the correct version. To return to a consistent state, you must perform a recovery operation on the data container.

Data container recovery

Prior to performing any data operation, Delphix Self-Service takes snapshots of all datasets. Recovery is the process of rolling back a data container to a snapshot, thereby restoring it to a consistent state. When a failure occurs, you will see the following screen:

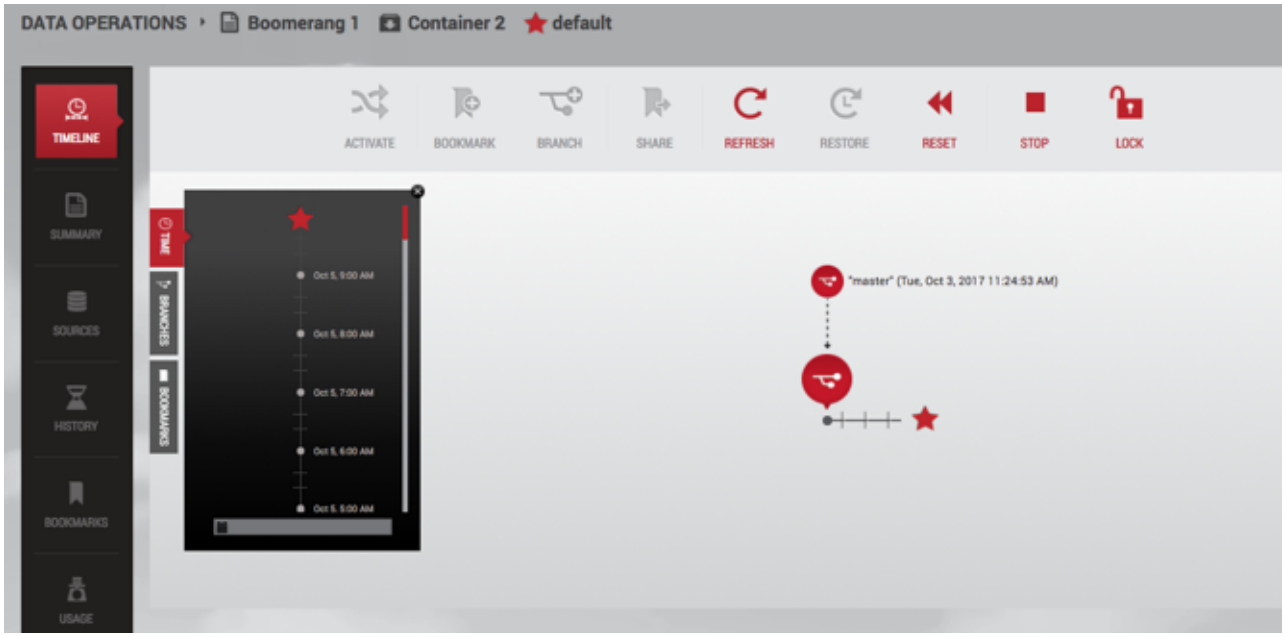


You can either perform recovery or use a different data container. Whether the recovery will fail or succeed depends on exactly why the data operation failed in the first place. If the problem was intermittent, such as a temporary network problem causing SSH failure, then performing recovery should work. If the problem is persistent – for example, the target host is out of space – then intervention is required; recovery will not succeed until you address the underlying root cause of the failure.

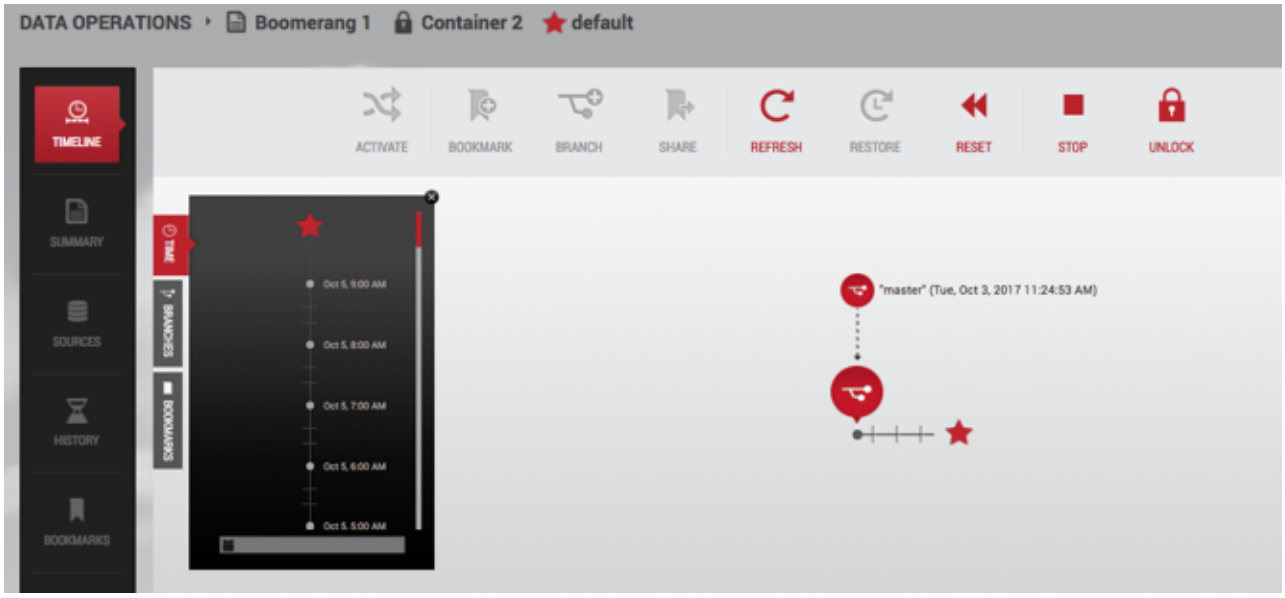
Admins can see the underlying failure in the **Actions** sidebar or the **Job History** dashboard. The **Actions** sidebar is the preferred place to view the failure; it has a hierarchical display that makes diagnosing the failure more straightforward.

Activity Nine: working with container locks

Container owners have the ability to lock/unlock containers. By default, a container is unlocked, which means that all the container's owners can perform operations on it:



You can lock a container by clicking the **Lock** button in order to prevent other users from operating on it.



The information on who locked the container is available in three places:

- In the history below the timeline
- In the tooltip on the locked container's name in the container selector
- In the tooltip on the locked container's name in the container label above the timeline

Errors

If an action such as refresh fails, a dialog box will give you further information. The **Error Details** field will

The **copy** button enables you to copy and paste error details, which you can then send to your Delphix administrator for further assistance.

Error dialog

Containers with multiple owners

Delphix Self-Service administrators can designate multiple users as owners of a single data container. These users all share access to the same data container which means actions taken by one user will impact all users on the same data container. For example, if User A activates Branch X, User B will also see Branch X as the active branch. This ability to for one user's actions to impact another user on the same containers creates new concerns for users sharing the same container. As a result, more processes should be put into place in order to coordinate usage between users. Each team is different, but strategies include:

- designating a person to perform certain data operations
- saving your work with a bookmark or creating / working on a personal branch
- being aware of who is using your data container / data before performing operations
- locking a container to prevent others from performing any operations on it

How many owners should a container ideally be shared between?

There is no technical limit built into the software, but it is best if a team of 5-10 users shares a single data container. In most cases, having fewer owners minimizes overhead and conflicting usage. One owner per container provides maximum productivity and minimal overhead, so this feature should only be used if your infrastructure or processes require that multiple users share a container. Additionally, Jet Stream Only users currently cannot see other users with whom they share the container.

How should users handle potentially disruptive operations?

If one user performs an operation on a data container, it will affect the other owners of that container. Additionally, each user has permission to perform the same operations on the data container; currently there are no fine-granularity permissions that limit the operations a user can perform. All operations are potentially disruptive, but the level of disruption varies by operation. If any of the following operations are performed at the same time, the second operation will fail due to a conflict when processing the job.

Conflicting operations

- Refresh
- Restore
- Reset
- Enable/Disable
- Create Branch
- Activate Branch
- Delete Branch
- Create Bookmark
- Delete Bookmark

If User A performs a destructive operation while User B is "using" the data container, the operation will destroy User B’s current state. Currently, the interface does not provide insight into whether the data container is in use by another user.

Destructive operations

- Refresh
- Restore
- Reset
- Enable/Disable
- Create Branch

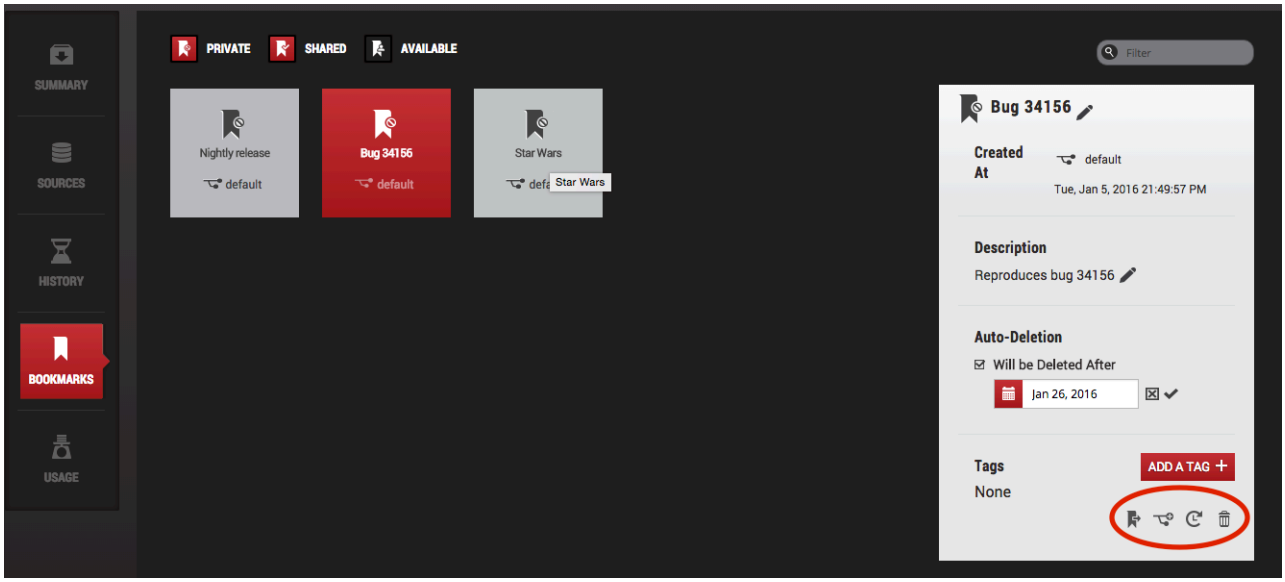
- Activate Branch

Deleting objects

All owners can delete any bookmarks or branches in the container, regardless of who created them.

Working with bookmarks in a data container

Working with bookmarks is an easy way to share data with other users of any container created from the same template. By sharing with others, you can integrate testing, development, and QA needs. For example, in the past, if you found a bug you would wait until it was fixed. But with bookmarks, you do not have to stop your work while someone tries to fix the problem. Sharing a bookmark allows users to work with data as they see fit. Bookmarks mark a moment of data. Delphix Self-Service will never automatically delete the data marked by that bookmark. However, Delphix Self-Service will automatically delete a bookmark with an expiration date set at the end of that day.



Bookmarks Management in the Data Report View Panel

Activity Nine: Share a bookmark with other Delphix self-service users

Share a bookmark

1. Select a **bookmark** by clicking one of the following:
   - The bookmark's **bubble** on the **branch timeline**.
   - The **Bookmarks** tab in the data container workspace.
   - The **Bookmarks** tile in the **Data Container Report Panel**.
2. Click the **Share** icon.

You cannot share a bookmark that you or another user have already shared.

Unshare a bookmark

1. Select a **bookmark** by clicking one of the following:
   - The bookmark's **bubble** on the **branch timeline**.
   - The **Bookmarks** tab in the data container workspace.
   - The **Bookmarks** tile in the **Data Container Report Panel**.
2. Click the **Unshare** icon.

You cannot unshare a bookmark that is already private or a bookmark which someone else has shared.

Delete a bookmark

1. Select a **bookmark** by clicking one of the following:
    - The bookmark's **bubble** on the **branch timeline**.
    - The **Bookmarks** tile in the **Data Container Report Panel**.
2. Click the **Delete** icon.

Activity Ten: editing bookmarks

Rename a Bookmark

1. In the **Data Container Report Panel**, click the **Bookmarks** til**e**. A selection of bookmarks will appear based on whether you have chosen to view private, shared, and/or available bookmarks.
2. In the **detail bookmarks** window, click the **Edit** icon to the right of its name.
3. Enter the **new name** in the edit field.
4. Click the **checkmark** to the right of the field to accept and save the new name.

Edit the description of a bookmark

1. Select a **bookmark** by clicking the **Bookmarks** tile in the **Data Container Report Panel**.
2. Click the **Edit** icon to the right of its name.

Remove the expiration date of a bookmark

1. Select a **bookmark** by clicking the **Bookmarks** tile in the **Data Container Report Panel**.
2. To the right of the bookmark's name, click the **Edit** icon.
3. Uncheck the **Will be deleted after** checkbox.
4. Click the **checkmark** to the right of the date selector.

Set or update the expiration date of a Bookmark

1. In the **Data Container Report Panel**, click the **Bookmarks** til**e**.
2. To the right of the bookmark's name, click the **Edit** icon.
3. Check the **Will be deleted after** checkbox.
4. Use the date selector to pick a new date.
5. Click the **checkmark** to the right of the date selector.

Activity Eleven: filter and view bookmarks

View Only Your Created Bookmarks

In the **Bookmarks** tile in the **Data Container Report Panel**, bookmarks that belong to you are shown. To see only your own bookmarks:

1. In the **Data Container Report Panel**, click the **Bookmarks** tile.
2. De-select **Available**.

View Bookmarks You Have Shared with Others

1. In the **Data Container Report Panel**, click the **Bookmarks** tile.
2. De-select **Private**.
3. De-select **Available**.

Only your shared bookmarks will be shown.

View Bookmarks That Others Have Shared with You

1. In the **Data Container Report Panel**, click the **Bookmarks** tile.
2. De-select **Private**.
3. De-select **Shared**.
4. Select **Available**.

These are the bookmarks that have been shared with you.

Adding Tags To Your Bookmark

1. In the **Data Container Report Panel**, click the **Bookmarks** tile.
2. Select the **bookmark** to which you want to add tags.
3. Click **Add a Tag**.
4. Enter the **tag name**.
5. Click the **Accept** icon.

Your tags will be shown at the bottom of the **Bookmarks** tile in the **Data Container Report Panel**.

You can only add tags to bookmarks that you have created.

Finding Bookmarks

In either the **Bookmarks** tab in the data container workspace or the **Bookmarks** tile in the **Data Container Report Panel**:

- Type into the **Filter** field.

This will only show bookmarks that have names or tags that match the text you have entered.

Understanding Delphix self-service usage

Usage management Dashboard overview

Data templates are comprised of dSources, virtual databases (VDBs), and vFiles. These data sources are controlled by the standard policies configured in the **Management** application of the Delphix Engine. As with existing containers, space will be reclaimed by the retention policy over time. As retention cleans up historical data, users will no longer be able to use those points in time to restore or branch. In Delphix Self-Service, an admin can create a bookmark on the data template timeline, which will prevent retention from cleaning up the data that bookmark references.

Data containers are comprised of VDBs provisioned from the sources defined in the data template. Similar to VDBs in the **Management** application, data containers' VDBs will share blocks with the source from which they are provisioned. This prevents the referenced data on the source from being cleaned up by retention. Retention for these VDBs is controlled by the standard Delphix Engine retention policies. As on templates, bookmarks in data containers will prevent storage from being reclaimed by retention. In addition, Self-Service will ensure that the latest data on each branch is never removed.

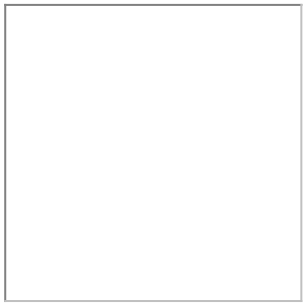
The **Usage** pages of the data templates and data containers provide information that can help you understand how storage is being used, how to reclaim space, and how much space you are able to reclaim.

Usage Overview is a top-level page, along with the **Data Operations** and **Overview** pages. It contains the space usage breakdowns by data templates and users.

Container usage overview

The **Usage Details** page, shows the space used by data containers provisioned from the template and the bookmarks created on the template.

The stacked bar graph shows information about the top 10 space users. You can re-sort the graph based on the fields in the **Sort by** legend on the top right-hand corner of the screen as seen in the image above. For example, if you want to know which data containers are sharing the most data with others, you can un-select **Shared (others data)** and **Unique** by clicking them in the legend.



When the legend items are not selected, their corresponding colored boxes turn gray and the data is removed from the chart. The data and name will reappear when you re-select by click on the preferred grayed-out category.

The field categories display the following information:

- **Unique** – The amount of space that will be freed if you delete this data container. This assumes that also delete underlying data sources.

- **Shared (others data)** – The amount of space that cannot be freed on the parent data template (or sibling data containers) because it is also being referenced by this data container due to Restore or Create Branch operations. The snapshots on the template or sibling container are what use up space.
- **Shared (self data)** – The amount of space that cannot be freed on this data container because it is also being referenced by sibling data containers due to Restore or Create Branch operations, via shared bookmarks
- **Unvirtualized** – The amount of space that would be used by the data in this container without Delphix virtualization

Bookmarks usage overview

As shown in the image above, the **Container Usage** page provides the usage information about bookmarks created on a template. The primary categories of information include **Unique**, **Shared (others data)**, and **Shared (self data).**

The field categories display the following information:

- **Unique** – The amount of space that will be freed if you delete this bookmark

- **Shared** – The amount of space referenced by this bookmark that cannot be freed by deleting this bookmark because it is also referenced by neighboring bookmarks or branches that have been created or restored from this bookmark
- **Externally Referenced** – The amount of space referenced by this bookmark cannot be freed by deleting it because it is also being referenced outside of Self-Service – for example, by a retention policy.

Branches usage overview

As detailed in the image above, the **Container Usage Details** page shows the usage information about the branches and bookmarks created on a container. The primary categories of information include **Unique**, **Shared (others data)**, and **Shared (self data)**.

The field categories display the following information:

- **Unique** – The amount of space that will be freed if you delete this branch
- **Shared (others data)** – The amount of space that cannot be freed on the parent data template or sibling branches because it is also being referenced by this branch due to Restore or Create Branch operations. The snapshots on the template or sibling container are what use up the space.
- **Shared (self data)** – The amount of space that cannot be freed on this branch because it is also being referenced by sibling data containers due to Restore or Create Branch operations, via shared bookmarks.

Developer's guide

Developer's guide

Command line interface guide

This section contains the following topics:

Command line interface overview

This topic provides an overview of the Delphix Engine command-line interface and links to additional topics.

The Delphix Engine provides a native command-line interface (CLI) accessible over SSH. This CLI provides an interactive layer on top of the public web service APIs, and is intended for users that wish to automate interactions with the Delphix Engine, or simply prefer a text-based interface. All of the functionality available in the CLI is also available through the public stable web service APIs should more full-featured automation be required. For more information on automation using the web service APIs, see the Web service API guide.

The CLI has an internal help system and supports tab-completion to help guide users. Running the `help` command will display a list of valid commands and properties, if applicable. Specifying the command or property as an argument to `help` will display more specific information about that command or property. This guide serves as an overview of CLI operation and examples of some basic tasks, and is not a reference for all CLI commands or properties. As the CLI content is identical to the public web services, complete information about particular commands, properties, or other operations can be found in the API documentation delivered with each server instance, found at:

```
http://<server>/api
```

The API documentation is guaranteed to be consistent with the set of APIs exported by that particular server. All of the APIs used by the GUI will be supported by the CLI. While all the database and environment APIs are available, most of the system-oriented APIs (such as those required to do initial setup) will be made available in a later release.

- Connecting to the CLI
- CLI contexts
- Managing objects
- Managing properties
- Array properties
- Untyped object properties
- CLI automation

Connecting to the CLI

This topic describes how to connect to the Delphix Engine command line interface.

If the same username exists in both the SYSTEM and DOMAIN namespaces (for example a sysadmin and a domain user), when logging in via ssh it is necessary for the user to explicitly provide the namespace SYSTEM or DOMAIN. For example, a user whose name exists in both namespaces must ssh in as either "someone@SYSTEM" or "someone@DOMAIN". If the user tries to log in as just "someone", the engine will give an error: Ambiguous username. Add a @DOMAIN or @SYSTEM suffix to the username.

The CLI is available over SSH or the terminal console on any Delphix Engine version 3.0 or later. To connect, use any SSH client appropriate for your workstation environment and connect to the Delphix Engine by IP or hostname on the standard SSH port (22). Enter a username for either a domain or system user followed by the namespace appropriate to that user (either DOMAIN or SYSTEM). For example:

```
ssh admin@DOMAIN@delphix-server.example.com
```

```
ssh sysadmin@SYSTEM@delphix-server.example.com
```

The default domain user created on Delphix Engines is now **admin** instead of delphix_admin. When engines created before 5.3.1 are upgraded to 5.3.1 or later they will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then Disabling delphix_admin.

At the prompt, enter your user password. Once connected, you will be placed at the CLI prompt:

```
delphix>
```

While both admin and sysadmin produce the same prompt once logged in, be aware that the two users have different menus and different functional areas.

Sysadmin menu

```
delphix> ls
Children
network
service
storage
system
user
Operations
version
Operations
version
delphix>
```

Delphix admin menu

```
delphix> ls
Children
alert
audit
authorization
connectivity
database
environment
fault
group
host
job
namespace
network
policy
replication
repository
service
session
snapshot
source
sourceconfig
system
timeflow
user
Operations
version
delphix>
```

Individual commands passed as arguments to the SSH client will be interpreted as if they had been read from the terminal. More complex scripts can be passed as input to the SSH command. When running SSH in non-interactive mode via these mechanisms, the command line prompt will be suppressed, as will terminal font decorations such as underline and bold.

The CLI is also available from the serial terminal console should the network be unavailable. Consult your VM platform documentation for information on how to connect to the terminal console. Once connected, log in using your Delphix user credentials just as you would over SSH.

If the management service is unavailable due to a software bug or other problem, the CLI can still be accessed as a system user provided that user is locally authenticated (not via LDAP) and has logged in at least once before. While in this state, only the `system` commands are available, including `restart`, which will attempt to restart the management service without rebooting the entire server. If this problem persists, please contact Delphix support.

The topic CLI cookbook: configuring Key-Based SSH Authentication for Automation shows an example of how to connect to the CLI using SSH key exchange instead of the standard password-based authentication.

CLI contexts

This topic explains the concept of contexts within the Delphix Engine command-line interface.

The CLI is built on the concept of modal “contexts” that represent an administrative point for interacting with the web service APIs. These contexts can be divided into the following types:

Context	Description
Static children	These contexts exist for the purpose of navigating between points in the hierarchy, but have no properties of their own and do not correspond to any server-side object. The root context is an example of this, as are most of the top-level contexts such as `database` or `group`.
Object	These contexts represent an object on the server, either a specific object (such as databases) or system-wide state (such as SMTP configuration). These contexts have properties that can be retrieved via the get command.
Operation	These contexts represent a request to the server. Commands may or may not require input and may or may not change state on the server, but in all cases require an explicit commit operation to execute the command. When in a command context, the prompt includes a trailing asterisk (*) to indicate that `commit` or `discard` is required before exiting the context.

User can move between contents by typing the name of the context. To move to a previous context, the `up` or `back` commands can be used. In addition, the CLI supports UNIX-like aliases for `cd` and `ls`, allowing navigation similar to a UNIX filesystem. For more information on these commands, see the Command Reference section.

Managing objects

This topic describes the use of objects in the Delphix Engine command-line interface and provides a list of the object management operations.

The Delphix Engine represents state through objects. These objects are typically managed through the following operations, covered in more detail in the Command Reference topics.

The topic CLI cookbook: Changing the Default Group Name illustrates the use of object management commands such as `list` and `get`.

Operation	Description
`list`	For a given object type (represented by a static context such as `database`), list the objects on the system, optionally constrained by some set of attributes. Some objects are global to the system and do not support this operation.
`select`	Select a particular object by name to get properties or perform an operation on the object. See the “Delphix Objects” section for more information on object naming.
`get`	Display all or some of the properties of an object after selecting it.
`update`	Enter a command context to change one or more properties of an object after selecting. Not all objects support this operation, and only properties that can be edited are shown when in the update command context.
`create`	Create a new instance of the object type from the root static context. Not all objects can be created in this simplified fashion. Databases, for example, are created through the `link` and `provision` commands.
`delete`	Deletes an object that has been selected. Not all objects can be deleted.

In contexts where there are multiple objects of a given type, the `list` command can be used to display available objects, and the select command can select an object for subsequent operation.

When listing objects, each context has its own set of default columns to display. The display option can be used to control what columns are displayed to the user. This is a comma-separated list of property names as they would be retrieved by the `get` command. It is possible to specify properties that do not exist in order to accommodate lists of objects of varying types and untyped objects.

The topic CLI cookbook: Listing Data Source Sizes provide an example of using the `list` command.

Managing properties

This topic describes the use of properties in relation to objects in the Delphix Engine command-line interface.

Object properties are represented as a hierarchy of typed name/value pairs. The `get` command by itself will display the complete hierarchy for a particular object. This hierarchy is displayed with each nested object indented by an additional level. The set of available properties depends on the command context and may change if the type of an object is changed.

Property state

Properties are typically set to a specific value, but they can also be `unset`. Unset properties indicate there is no known value, either because it hasn't been provided yet, or it has been explicitly removed. Properties in this state are displayed via the following means:

- `(unset)` – The property is not currently set. It may never have been given a value or it may have been explicitly unset through the `unset` command.
- `(required)` – This has the same underlying semantics as `(unset)`, but indicates that the property **must** be set before the current command can be committed. Failure to do so will result in a validation error at the time the commit operation is attempted. Required properties are displayed in bold.

In addition, all objects have a default state when in command context. A property that has been modified is noted with an asterisk (*****), and can be reverted to its default state through the `revert` command.

When updating properties, only those properties are sent to the server. The exception is arrays and untyped objects, covered in Array properties and Untyped object properties. These objects are always sent in their entirety, so changing any one element will send the entire object.

Basic properties

Most properties are displayed and input as a string, though the underlying type may be more specific. The following are some of the basic types:

- **String** – An arbitrary string. This may be subject to additional validation (such as an IP address) that is enforced at the time the property is set.
- **Number** – An integer number.
- **Boolean** – Either "true" or "false".
- **Enumeration** – A string that must be chosen from a known set of options.

Nested properties

Some properties are in fact other objects and are represented as a nested set of properties. These properties can be manipulated in one of two ways: by specifying a dot-delimited name or changing the context via the `edit` command.

A dot (.) in a property name indicates that the portion to the left of the dot is the parent object name, and the portion to the right is a child of that object. For example, `sourcingPolicy.logsyncDisabled` denotes the `logsyncDisabled` property within the `sourcingPolicy` property. These dots can be arbitrarily nested. An alternative syntax of using brackets to enclose property names `(sourcingPolicy[logsyncDisabled])` is also supported for familiarity with other programming languages.

The `edit` command, in contrast, will change the current context such that all properties are relative to the specified object. This can be useful when changing many nested properties at once, or when the complete set of properties can be confusing to manage all at once.

The topic CLI cookbook: Disabling LogSync for a dSource provides an example of manipulating nested properties.

Array properties

This topic describes the use of array properties in the Delphix Engine command-line interface.

Some Delphix objects represent properties as arrays. Arrays are effectively objects whose namespace is a contiguous set of integers. While they behave like objects and their properties can be referenced via the same object property notation, they differ in several key areas.

Arrays can be divided into two types: arrays of primitive types (strings, integers, etc.) and arrays of objects. Arrays of objects can be managed like other objects via nested property names and the `edit` command, but differ in the following respects:

- When an array element is `unset`, it removes the element from the array and shifts all other elements down to preserve the contiguous index space.
- New array elements can only be appended to the end of the array by specifying an index that is one more than the maximum index of the array.
- When displaying a property that is an array, if the length is greater than 3, then it is displayed only as "[…]". The complete contents of the array can be displayed by getting or editing that particular property.

Arrays of primitive types can be managed as arrays of objects, but also support an inline notation using comma-separated notation. This allows single-element arrays to be set as standard property, and for arrays of strings to be set on a single line instead of having to edit each element.

Regardless of the element type, arrays are sent as complete objects when updated. When any array element is changed and subsequently committed, the complete array is sent to the server. When a single array element is reverted, the entire contents of the array are reverted.

The topic CLI cookbook: Setting Multiple Addresses for a Target Host provides an example of working with a property that is an array of strings.

Untyped object properties

This topic describes the use of the `type` field in the Delphix Engine command-line interface object model, and the use of untyped objects.,

Most Delphix objects are typed, meaning they have a `type` field that controls what properties are available and their types. Object types and their associated hierarchy are described in more detail in the topic Object Type Hierarchy topic. In contrast, some properties are “untyped” objects, which means that there are no constraints on the property namespace, and all properties are plain strings. These objects are used for database configuration templates and other scenarios where the property namespace is unbounded or under the control of the user.

Untyped objects are always sent in their entirety when making updates. This means that when anyone value is changed and then committed, all values are sent. In addition, when reverting a single value within an untyped object, the entire parent object is reverted to its default state.

CLI automation

This topic describes using automation with both the Delphix Engine command-line interface (CLI) and the web service API.

All functionality is available in both because the CLI is built upon the web services API. The CLI enables you to create scripts for simple automation, and it is a useful aid in the development of more complex code that uses the web service API.

Using the CLI for simple scripts

For simple automation, you can build routines that make CLI calls through SSH.

This snippet lists all environment names. It leverages the SSH key exchange explained in CLI cookbook: Configuring Key-Based SSH Authentication for Automation so that no password is required for the user named "automation".

```
DELPHIX_ENGINE=172.16.180.33
SSH_CMD="ssh automation@${DELPHIX_ENGINE}"


${SSH_CMD}  "cd host; list display=name"
```

Backward compatibility

Both the CLI and web services API are versioned to support backward compatibility. Future Delphix versions are guaranteed to support clients that explicitly set a version provided the major version identifier is compatible. For more information, see the Web Service API Guide. The CLI will always connect with the latest version, but the `version` command can be used to both display the current version and explicitly bind to a supported version.

Users building a stable set of scripts can run `version` to get the current version. Scripts can then run the `version <id>` command to guarantee that their scripts will be supported on future versions. For more information on the different API versions and how they map to Delphix versions, see the API Version Information section.

Parsing CLI output

The default text output of the CLI is unstable. Any attempt to parse the output is certain to run into difficulties in repeatable results for unknown input, as well as instability as the text output is changed in subsequent releases. Column headings, column order, and the number of columns will change in subsequent releases.

You can specify a version in your scripts to counteract this, but you will not be able to take advantage of new features and fixes.

CLI as a development tool for complex automation

While the CLI is useful for simple automation tasks, it can be slow and overly complicated due to the many round trips needed to control the automation logic. For example, to disable all the environments for an engine, you could write a script which lists the environments and modifies each one:

```
DELPHIX_ENGINE=172.16.180.33
SSH_CMD="ssh automation@${DELPHIX_ENGINE}"
```

```
env_array=(`${SSH_CMD}  "version 1.5.0; cd environment; list display=name" | grep -v
NAME` )
for i in "${env_array[@]}"
do
  ${SSH_CMD} "version 1.5.0; cd environment; select $i; disable; commit"
done
```

This script works, but it will be slow on systems with many environments since each SSH command will start a new session.

The web service APIs are superior when performing many operations as a single logical unit. The web service APIs also provide substantially more data with a single call than what is shown in the CLI output, which can greatly simplify your code and avoid multiple round trips.

However, the input and output of web service API calls are JSON data, and it can be difficult to quickly determine what the input and output will look like.

For this reason, the CLI provides two options that can greatly assist you in the development of complex automation: **JSON Output** and **Tracing.**

`(setopt format=json)` changes the CLI to the output of all results to parseable JSON (javascript object notation). This is the fastest and easiest way to quickly see what the JSON output will look like when executed via the Web Service APIs. The JSON format has wide support in a variety of programming languages; see http://www.json.org for more information.

`(setopt trace=true)` will display the underlying HTTP calls being made with each operation and their JSON payload. This allows you to determine the GET and POST calls, and their JSON payloads, which perform the actions that you need to power your automation.

`(setopt format=text)` changes the CLI back into its regular output mode. `(setopt trace=false)` turns off the trace display.

> Using both options will show the JSON output twice

The fastest way to develop complex automation is to experiment with the CLI and copy the underlying API calls to a custom system for better control over behavior.

```
delphix421> setopt trace=true
delphix421> cd user
delphix421 user> create
delphix421 user create *> ls
Properties
    type: User
    name: (required)
    authenticationType: (unset)
    credential: (unset)
..... (Output Truncated) .....
    userType: DOMAIN
    workPhoneNumber: (unset)
delphix421 user create *> set name=Jose
delphix421 user create *> set authenticationType=NATIVE
delphix421 user create *> set credential.password=Password1
delphix421 user create *> commit;
```

```
=== POST /resources/json/delphix/user ===
{
    "type": "User",
    "name": "Jose",
    "authenticationType": "NATIVE",
    "credential": {
        "type": "PasswordCredential",
        "password": "Password1"
    }
}
=== RESPONSE ===
{
    "type": "OKResult",
    "status": "OK",
    "result": "USER-35",
    "job": null,
    "action": "ACTION-107"
}
```

Using the output above, you can see that to create a user you must use the URL "http://myengine/resources/json/delphix/user". You will use a POST command and pass a JSON payload which looks like the above. You will get a JSON response like the above, and can validate that the status is "OK".

Delphix objects

These topics describe the object model for the Delphix Engine command-line interface.

The Delphix object model is a flexible system for describing arbitrary hierarchies and relationships of objects. In order to enable the current and future functionality of the system, the relationship between objects is not always immediately obvious. The CLI is merely a veneer atop the web services layer to ensure that the full complement of functionality expressed by the API is always available, but this requires users to have some understanding of how objects are represented in the system.

This section covers the following topics:

Object type hierarchy

This topic describes the object type hierarchy for the Delphix Engine command-line interface.

All Delphix objects have an associated type. This type determines what properties are available for a particular object, the format of those properties, and controls how the system interprets objects and commands. The type hierarchy uses polymorphic inheritance to allow for common properties and behavior to be defined at a single point while permitting dramatically different types of objects to co-exist without requiring a completely separate API for each. For example, the `SourceConfig` object is the base type for all external database configurations, but it has children that include `OracleSIConfig` and `OracleRACConfig` types that refer to a single instance and RAC databases, respectively.

When specifying input types, the system will attempt to determine types appropriate for the current operation, but there are times when the type must be explicitly set, either because the operation supports multiple possible inputs, or the object can embed an abstract type. In these cases, it may be necessary to explicitly set the `type` property. Setting the type may change the set of visible properties and the resulting validation that is performed, but it will not affect any properties that are already set.

Object names and references

This topic describes the use of object names and references in the Delphix Engine command-line interface.

Most Delphix objects are persistent objects in that they have a well-known identity on the server and associated persistent state. The exceptions are objects used only as input to other operations, or global objects that have a persistent state but don't require any explicit identity since they always exist.

Persistent objects have both a name and a reference. The reference is the canonical identifier for the object and remains valid even if the object is renamed on the server. It is an opaque token that should never be interpreted by the client; the format may change in future releases though backward compatibility with current references will be maintained. All web service APIs operate using references. References can be used in the CLI when selecting objects, but given that they are a programmatically generated internal concept, they are difficult for most users to use.

The object name, on the other hand, is a much more convenient way to refer to objects but suffers from the fact that it is not guaranteed to be globally unique. When displaying or setting references, the CLI will convert to or from the 'canonical name' based on the type of the reference and the current set of objects on the system. The canonical name has the form:

```
<Type>:/<Parent>/<Object>@<Namespace>
```

The type, parent, and namespace are only included if the local object name conflicts with other objects on the system that would otherwise be valid for the given type specification. Not all objects have names relative to their parent; groups, environments, users, and many other objects are globally unique on the system. This "best fit" method is used both when displaying references as well as when setting properties that are references. If the given name potentially matches multiple objects when attempting to set a reference property, then an error is displayed that includes a list of possible names to clarify which object is being referred to. The conversion from reference to name on display only happens with text output format. When the output format is JSON, the raw content is displayed (including the local name) and it is up to the consumer to format names appropriately based on their semantics. The conversion from name to reference when setting properties always occurs. Consumers can use references, optionally prefixed with a backtick (`) character to signify they are references in the unlikely event that someone has created an object with the same name as a valid reference.

> ⓘ Providing unique names for objects without the use of forward slashes ('/') and at signs ('@') will provide the simplest CLI experience when referencing objects.

Here are some scenarios for databases and groups and their resulting behavior:

No conflicting database name

The local name will be used when displaying references to the object, and can be used when setting references:

```
set container=example
```

Databases with the same name in different groups

The parent group name must be used when displaying references to the object and when setting references to the object:

```
set container=group1/example
```

Databases with the same name in different namespaces

The namespace name must be used when displaying references to the object and when setting references to the object:

```
set container=example@namespace
```

Objects of different types but with the same name

This conflict is exceptionally rare, as the reference context typically constrains the set of possible objects to be a single type, but there are cases (such as alerts, or policy targets) that can be applied to any object. In these cases, the type name must be included to uniquely identify the object:

```
set target=Container:/group1/example
```

In the event that one of the named components contains a slash (‘/’) or an at-sign ('@'), single quotes must be used to disambiguate the name from its parent or namespace.

Databases and environments

This topic describes the relationship between database container objects and environments in the Delphix Engine object model.

The core Delphix objects revolve around the notion of environments and databases, known at the API layer as **containers**. Understanding how these objects relate to each other is crucial to operating effectively within the CLI. This section provides an overview of these objects; for more information about a particular representation such as Oracle RAC, see the Web Service API Guide.

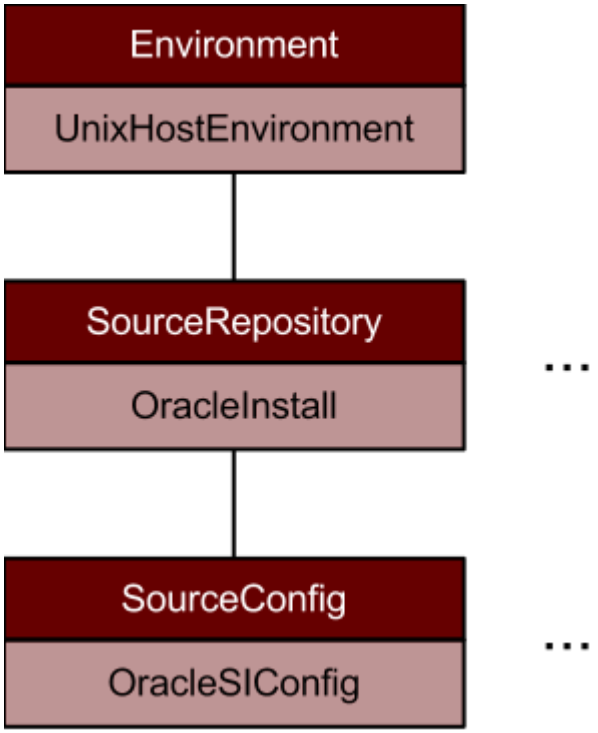
> In variations of a use case where a database propagates down to lower environment databases (for keeping consistent data), records created in the lower tables would likely be overwritten by the propagating database once they are synced. Delphix is keeping track of changes at the data block level on disk, which would render this result.
> As a solution, you could create a clone of the propagating database and then programmatically compare/merge the changes. You can also use a product like Oracle Goldengate, HVR, IBM Datastage, etc, to capture (extract) and propagate (replicate to lower) changes.

Environment components

- An **environment** is the root representation of the external state that manages database instances. An environment could be a single host (`UnixHostEnvironment`) or an Oracle cluster (`OracleClusterEnvironment`). Environments exist to contain repositories, and each environment may have any number of repositories associated with it.
- A **repository** is an entity that contains database instances. Repositories are typically installation directories (`OracleInstall`) within an environment. Within each repository is any number of `SourceConfig` objects, which represent known database instances.
- The **source config** exists independent of Delphix, and could represent a possible dSource (in which case there is no associated database object), or could be managed entirely by Delphix (for VDBs). The source config contains intrinsic properties of the database instance, while the source (described below) contains information specific to Delphix and only exists when the source config is linked to a dSource or VDB.

Most environment objects are created through the act of **discovery**. By specifying a host, Delphix will attempt to automatically discover all environments, repositories, and source configs. These objects can also be added manually after the fact in cases where discovery fails.

The environment hierarchy can be represented as depicted below. The generic type is listed in the top portion of each box, with an example of the Oracle single instance objects in the lower portion of each box. Each of these objects can contain multiple child objects with it.

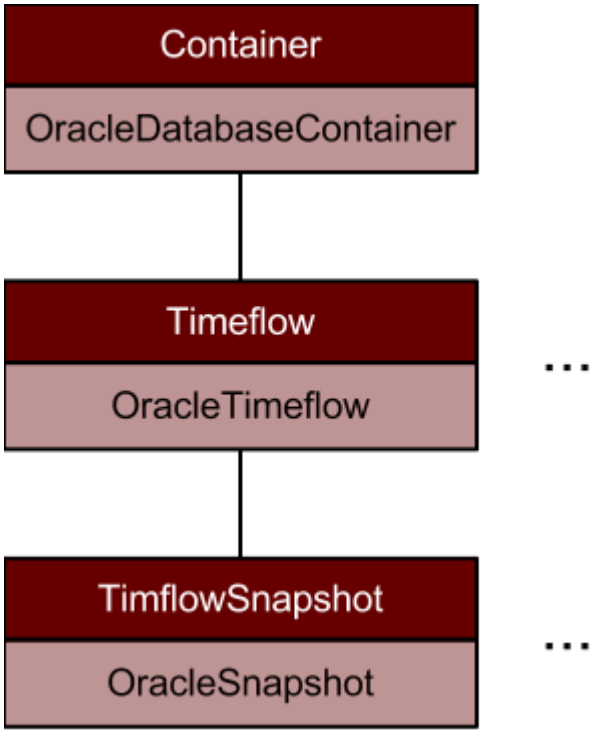


Database components

The core of all databases within Delphix is the container that holds all the physical data associated with the database, whether a dSource or VDB. Within each container is a **Timeflow**, which represents a single timeline of change within the database history. Currently, a container can only have one Timeflow, though this limitation may be relaxed in a future release.

Within a Timeflow are two important objects: `TimeflowSnapshot` objects and `TimeflowRange` objects. Timeflow **range** represents the provisionable ranges within the history of the Timeflow, while Timeflow **snapshot** represents a point at which a snapshot was taken and therefore is more likely to provision in a short amount of time.

The environment hierarchy can be represented as depicted below. Each container may be associated with a **source**, which is the Delphix representation of an external database when it is associated with a container, and contains information specific to managing that source. Not all source configs within an environment have a source associated with them (as is the case with linkable databases), but all sources must have a source config. Containers may have no sources associated with them if they are unlined; sources can be manually attached at a later point. Currently, each container can have at most once source associated with it, though this may change in a future release.


Container
OracleDatabaseContainer
Timeflow
OracleTimeflow
...
TimflowSnapshot
OracleSnapshot
...

Asynchronous jobs

This topic describes conditions under which command-line interface operations may spawn jobs that run in the background and using the `wait` option to wait for job completion.

Not all operations can be performed in the context of a single web service API call. For cases where there is a long-running operation that cannot be executed quickly and transactionally, a job may be dispatched to do the remaining work in the background. For more information on jobs and their semantics, see the topic Viewing action status. Within the CLI, any command can potentially result in an asynchronous operation. The default behavior is to wait for any such job to complete and display its progress in the CLI.

In the event that you do not want to wait for the operation to complete, the global wait option can be set `(setopt wait=false)`. If disabled, the CLI will display the reference to any job that was dispatched, but not wait for it to complete.

Command reference

These topics describe the core built-in commands within the CLI. It is not an exhaustive list of all commands in all contexts. For an object or type-specific commands, consult the API documentation.

This section covers the following topics:

CLI help and display commands

This topic describes help and display commands for the Delphix Engine command-line interface.

Command	Description
`children`	Display all statically defined children valid for the current context. These children can be targets of the `cd` command.
`commands`	Display all build in commands valid for this context.
`help`	Display all commands and properties valid for the current context. Specifying a command or property will provide more information about that command or object. When nested properties are present, only top-level properties are displayed by default, though specifying a particular property will display the entire hierarchy.
`ls`	Display children, commands, objects, and operations valid in the current context. Only those sections that are relevant in the current context are displayed.
`operations`	Display available context-specific operations. These operations require an explicit `commit` command to execute the operation, or `discard` to abort it.

CLI context commands

This topic describes context commands for the Delphix Engine command-line interface.

Command	Description
`back`	Return to the previous visited valid context. This history only tracks contexts that were actually visited, so running `database "example"` followed by back will return you to the root context, not the database (because the two were executed as part of one action and never actually visited). If a previous context was deleted or is no longer valid, this command will skip over it.
`cd`	Switch to the given child. This is identical to typing the name of the child itself, but also support UNIX-style directory structures, such as "/" and "..". This allows for contexts to be chained such as `cd ../database/template`.
`history`	Display the history of input to the shell. The shell supports the ability to move back and forth in the history using the up and down arrows.
`up`	This is an alias for `cd ..` for the benefit of those less familiar with UNIX filesystem navigation. Unlike back, which only returns to the previous context only if it was visited, and may return to a child context, this command will always return to the immediate parent context.

CLI object commands

This topic describes object commands for the Delphix Engine Command Line interface.

Command	Description
`list`	List all objects of a particular type when in the appropriate root context. Different contexts may support different options to the list command to constrain the output; run `help list` to see possibilities.
`select`	Select an object by name within a list.

CLI miscellaneous commands

This topic describes miscellaneous commands for the Delphix Engine command-line interface.

Command	Description
`echo`	Print the input arguments.
`exit`	Exit from the current CLI session. This is equivalent to sending the EOF control character (typically Ctrl-D) or closing your client SSH application.
`getopt`	Get the current value of a global configuration option. The list of global options can be retrieved by running `help getopt`, but include options for controlling JSON output (`format`), tracing HTTP calls (`trace`), and enabling synchronous job semantics (`wait`).
`setopt`	Set the value of a global configuration option.
`version`	Display the current API version or bind to a particular version. See the CLI automation section for more information.

CLI property commands

This topic describes property commands for the DelphixEngine command-line interface.

Command	Description
`commit`	When in operation context, commit the changes and execute the operation.
`discard`	When in operation context, discard any changes and abort the operation.
`edit`	Change the current context to be relative to a particular object property when in operation context.
`get`	Get all properties (with no arguments) or a particular property of the current object.
`revert`	Revert a particular property to its default value, either the value of the underlying object during an update or the default command input value.
`set`	Set the value of one or more properties. These properties can be specified as `name=value`, or as simply the property name. When only the property name is specified the CLI will prompt for the value to use, optionally obscuring the input if the property is a password.
`unset`	Clear the current value of a property. This is not the same as reverting the property, though this can have semantically identical behavior in the case that the default value is unset.

CLI cookbook: common workflows, tasks, and examples

This section contains the following topics:

Authentication and users

These topics describe command-line interface procedures for authentication and managing users.

- CLI cookbook: configuring SAP ASE manual discovery
- CLI cookbook: changing HTTP and HTTPS web connections
- CLI cookbook: configuring key-based SSH authentication for automation
- CLI cookbook: setting up SSH key authentication for UNIX environment users
- CLI cookbook: configuring SSH host verification for UNIX environments

CLI cookbook: configuring SAP ASE manual discovery

Overview

This topic describes how to use CLI commands to manually add ASE repositories to an SAP ASE environment. Discovery is the process by which the Delphix Engine identifies data sources and data dependencies on a remote environment. SAP ASE repository discovery is done automatically when an environment is added to the Delphix Engine or when an already added environment is refreshed. In some cases, automatic discovery does not discover all of the repositories in an SAP ASE environment. These repositories may be added using manual discovery.

> Unlike automatically discovered instances, manually discovered instances are not automatically deleted if the environment is refreshed when the instance is not running. The physical attributes such as ASE listener port, installation directory, and instance owner are not updated during an environment refresh either. If you change a physical attribute, you must manually update the repository.

To manually discover an SAP ASE repository you will need to:

- Add an SAP ASE environment
- Use CLI to manually discover a repository

Creating an SAP ASE environment

Please refer to Adding an SAP ASE Environment for detailed steps.

Manually discover a repository

This example uses `sc-dev3.dc1` as the example environment.

1. Log into CLI and cd to the **repository** menu:

```
$ ssh admin@sc-dev3.dc1
Password:
sc-dev3.dc1> cd repository
sc-dev3.dc1 repository>
```

2. Add (manually discover) an SAP ASE repository instance:

> **Note:** The values used in the following code block are specific to the example instance we are adding.

3.
```
sc-dev3.dc1 repository> create
sc-dev3.dc1 repository create *> ls
Properties
    type: ASEInstance
    credentials: (unset)
    dbUser: (unset)
    dumpHistoryFile: (unset)
    environment: (required)
    installationPath: (required)
    instanceName: (required)
```

```
    instanceOwner: (required)
    isqlPath: (unset)
    linkingEnabled: true
    ports: (required)
    provisioningEnabled: true
    servicePrincipalName: (unset)
    staging: false
    version: (unset)
sc-dev3.dc1 repository create *> set credentials.type=PasswordCredential
sc-dev3.dc1 repository create *> set credentials.password=sybase
sc-dev3.dc1 repository create *> set dbUser=sa
sc-dev3.dc1 repository create *> set environment=rh610-ebf-ase-tgt-
dev3.dc3.delphix.com
sc-dev3.dc1 repository create *> set installationPath=/opt/sybase/15-7/sp139/
install
sc-dev3.dc1 repository create *> set instanceName=ASE157_TGT
sc-dev3.dc1 repository create *> set instanceOwner=sybase
sc-dev3.dc1 repository create *> set ports=5400
sc-dev3.dc1 repository create *> ls
Properties
    type: ASEInstance
    credentials:
        type: PasswordCredential (*)
        password: ******** (*)
    dbUser: sa (*)
    dumpHistoryFile: (unset)
    environment: rh610-ebf-ase-tgt-dev3.dc3.delphix.com (*)
    installationPath: /opt/sybase/15-7/sp139/install (*)
    instanceName: ASE157_TGT (*)
    instanceOwner: sybase (*)
    isqlPath: (unset)
    linkingEnabled: true
    ports: 5400 (*)
    provisioningEnabled: true
    servicePrincipalName: (unset)
    staging: false
    version: (unset)
sc-dev3.dc1 repository create *> commit
    `ASE_INSTANCE-3
sc-dev3.dc1 repository>
```

Updating a repository

Adding onto the above, the following example illustrates updating an SAP ASE instance's version after upgrading SAP ASE:

Take caution when setting the version string. Make sure it matches the output as displayed by the "select @@version" query all the way out to the patch level (PL). For example, "15.7 SP138" or "16.0 SP02 PL01".

```
sc-dev3.dc1> repository
sc-dev3.dc1 repository> select ASE157_TGT
```

```
sc-dev3.dc1 repository 'ASE157_TGT'> update
sc-dev3.dc1 repository 'ASE157_TGT' update *> set version="15.7 SP139"
sc-dev3.dc1 repository 'ASE157_TGT' update *> ls
Properties
    type: ASEInstance
    credentials:
        type: PasswordCredential
        password: ********
    dbUser: sa
    dumpHistoryFile: (unset)
    installationPath: /opt/sybase/15-7/sp139/install
    instanceOwner: sybase
    isqlPath: /opt/sybase/15-7/sp139/install/OCS-15_0/bin/isql_r64
    linkingEnabled: true
    ports: 5400
    provisioningEnabled: true
    servicePrincipalName: (unset)
    staging: false
    version: 15.7 SP139
sc-dev3.dc1 repository 'ASE157_TGT' update *> commit
sc-dev3.dc1 repository 'ASE157_TGT'>
```

CLI cookbook: changing HTTP and HTTPS web connections

By default, the Delphix Engine allows both HTTP and HTTPS web connections. The following steps provide instructions on how to change their configuration:

1. Via CLI, login to the Delphix Engine as a system administrator (sysadmin).
2. Go to "service > httpConnector".
3. Initiate an update.
4. Set "httpMode" to the desired value among:
    a. "BOTH": accepts HTTP and HTTPS connections (this is the default)
    b. "HTTPS_ONLY": accepts only HTTPS connections
    c. "HTTP_ONLY": accepts only HTTP connections
    d. "HTTP_REDIRECT": accepts HTTPS connections and redirects HTTP connections to HTTPS.
    e. "HTTP_REDIRECT_WITH_HSTS": redirect all requests made over HTTP to HTTPS and add Strict-Transport-Security Header to all responses.
5. Commit your Change. The Delphix web application will restart.

The following is an example of how to set the Delphix Engine to accept HTTPS connections and redirect HTTP connections to HTTPS.

```
cd /service/httpConnector
update
set httpMode=HTTP_REDIRECT
commit
```

Replacing the HTTPS (HTTP secure) certificate

This topic explains how to replace the HTTPS (HTTP Secure) certificate used by the Delphix Virtualization Engine. There are two methods of replacing the certificate. The key difference between the two is whether Delphix or the user is providing the key pair (public and private key).

Delphix provided key pair

Use the following instructions to provide an HTTPS certificate chain for a key pair created by the Delphix Engine. Once the key pair is created users can download a Certificate Signing Request (CSR) to generate a signed certificate from the CA of their choice. This is done using the "create" operation in the "/service/tls/csr" API as seen below when using the CLI.

```
hostname.domainname> service tls csr
hostname.domainname service tls csr> create
hostname.domainname service tls csr create *> ls
Properties
   type: CertificateSigningRequestCreateParameters
   dname:
       type: X500DistinguishedNameComposite
       dname: (required)
  :
       type: EndEntityHttps
   forceReplace: false
   keyPair:
       type: RsaKeyPair
       keySize: 2048
       signatureAlgorithm: SHA256withRSA
```

The first key property is the dname. This will be used as the subject name of the CSR and resulting X.509 certificate unless it is changed when the certificate is signed. Delphix supports two different formats for dname:

- a composite string
- a list of fields

Use the composite string as follows:

```
hostname.domainname service tls csr create *> set dname.dname="CN=Delphix CA,
O=Delphix, C=US"
hostname.domainname service tls csr create *> ls
Properties
   type: CertificateSigningRequestCreateParameters
   dname:
       type: X500DistinguishedNameComposite (*)
       dname: CN=Delphix CA, O=Delphix, C=US (*)
```

Use the list of field formats as follows:

```
hostname.domainname service tls csr create *> set
dname.type=X500DistinguishedNameFields
hostname.domainname service tls csr create *> ls
Properties
```

```
    type: CertificateSigningRequestCreateParameters
    dname:
        type: X500DistinguishedNameFields (*)
        city: (unset)
        commonName: Delphix CA (*)
        country: US (*)
        organization: Delphix (*)
        organizationUnit: (unset)
        stateRegion: (unset)
```

The only required field is the commonName (CN).

The only currently supported type for endEntity is EndEntityHttps.

The next property is forceReplace. By default, this is false and means Delphix will not replace the active key pair and certificate with the newly generated keypair and self-signed certificate. If the user wants to replace the active key pair right away before the signed certificate has been created this can be set to true.

The final property keyPair impacts the generated key pair. When creating a new key pair the engine supports two algorithms:

- **RSA -** The supported signature algorithms are SHA256withRSA, SHA384withRSA, and SHA512withRSA . The valid key sizes range from 2048 to 4096.
- **ECDSA -** The supported signature algorithms are SHA256withECDSA, SHA384withECDSA, and SHA512withECDSA. The valid key sizes range from 256 to 571

Once the create operation has completed you can get the CSR in PEM format by selecting the CSR object and looking at the requestInPem property:

```
requestInPem: -----BEGIN CERTIFICATE REQUEST-----
MIIBezCB0gIBADAhMR8wHQYDVQQDExZiYmFrZXIuZGMyLmRlbHBoaXguY29tMIGn
MBAGByqGSM49AgEGBSuBBAAmA4GSAAQBU5WY9+GkCTFvbGHTNJDb/QM3t4YI/9S6
fhCJELx7SbJNti2n0l3mCePenyUuBY9m6BWvUQzlhawZG5YAJ9WdcM+IIPciiNsD
Xmw0eFH05z6yTLMnfBYYZKFbpu/dcK5V8WoltrIC7jTxg/k6jf/WeD+dmyIMQ0Z7
VmwnD6RsaAs7T5lajXkurwPfqQ5MnsmgADAKBggqhkjOPQQDAwOBlwAwgZMCSAGM
quqcnIAxIRDxQ+BzzSywNtozn5ihtfFxtTF/EW/ARBib2l9hq0pwHrIinnLvjW9u
avpAH1pkWHx1w0/O6W6DCZAPIIL3ugJHKsScJqsvaeZzVqJVfQt8g42cL9hKc7ic
HLhuAyMGQOXrEdLbOxtOH6SiExnyEv2Y9LHHYYgRafgGzOoA5tx+mrkr9J+zm8Y=
-----END CERTIFICATE REQUEST-----
```

Once the CSR has been signed and turned into an X.509 Certificate you can replace the certificate using the "service/tls/endEntityCertificate" API. To replace using the CSR method begin by setting the correct type of replace parameters as seen below:

```
hostname.domainname service tls endEntityCertificate> replace
hostname.domainname service tls endEntityCertificate replace *> set
type=EndEntityCertificateReplaceChainParameters
hostname.domainname service tls endEntityCertificate replace *> ls
Properties
    type: EndEntityCertificateReplaceChainParameters
    chain:
        type: PemCertificateChain
        chain: (required)
    endEntity:
```

```
        type: EndEntityHttps
```

The "chain" property must contain a list of the entire trust chain from the newly generated end-entity certificate to the root CA.

The CLI might not always interpret newline characters in PEM certificates correctly. Therefore, it is highly recommended to find and replace all newlines ('\n') with an empty string ('') prior to pasting the PEM certificate into the CLI.

To do this in the CLI first run:

```
hostname.domainname service tls endEntityCertificate replace *> edit chain.chain
Then `add` and `set contents` to the PEM certificate for each certificate in the
chain.
hostname.domainname service tls endEntityCertificate replace chain.chain *> add
```

When adding multiple certificates, use the command back after each add. After the final add, enter back and then commit.

The order in which the PEM certificates are added to the list does not matter.

Customer provided key pair

This section describes the steps to take if you are replacing the HTTPS with your own key pair and certificate.

1. To start, you need to add the key pair and full certificate chain as an entry in a file in JKS or PKCS #12 format.
2. Then, send a file upload request to the following endpoint:

```
hostname.domainname service tls endEntityCertificate
requestKeyPairAndCertChainUpload *> ls
Properties
   type: CertificateUploadParameters
   alias: alias_in_keystore (*)
   keypass: (unset)
   keystoreType: JKS
   storepass: ******** (*)
hostname.domainname service tls endEntityCertificate
requestKeyPairAndCertChainUpload *> commit
   type: FileUploadResult
   token: 8f4361c5-019c-4fee-9306-b7c85e977cf4
   url: /resources/json/delphix/data/upload
```

The **alias** field is where the key pair and certificate is saved in your JKS or PKCS #12 store.
The **keypass** field is the password for the given alias' key. If not set, it uses the keystore's password.
The **storepass** field is the keystore's password.

3. Then, establish a session from the host with the keystore to the Delphix Engine.
   Choose the location of the cookies, and determine the API version (command example uses 1.9.2):

```
curl -c <path/to/cookies> -X POST --data '{ "type": "APISession", "version":
{ "type": "APIVersion", "major": 1, "minor": 9, "micro": 2 } }' -H "Content-
Type: application/json" http://<delphix_engine_url>/resources/json/delphix/
session
```

4. Login to the Delphix Engine using the established session as a domain or system admin:

```
curl -c -b <path/to/cookies> -X POST --data '{ "type": "LoginRequest",
"username": "sysadmin", "password": "sysadmin" }' -H "Content-Type:
application/json" http://<delphix_engine_url>/resources/json/delphix/login
```

5. Send the file upload request with the location of your keystore and token from above:

```
curl -b <path/to/cookies> -X POST -F "file=@<path/to/keystore>" -F
"token=8f4361c5-019c-4fee-9306-b7c85e977cf4" http://<delphix_engine_url>/
resources/json/delphix/data/upload
```

6. You can now replace the HTTPS end-entity certificate with the keystore you have uploaded, identified by the token:

```
hostname.domainname service tls endEntityCertificate replace *> set
type=EndEntityCertificateReplaceKeystoreParameters
hostname.domainname service tls endEntityCertificate replace *> set
token=8f4361c5-019c-4fee-9306-b7c85e977cf4
hostname.domainname service tls endEntityCertificate replace *> ls
Properties
   type: EndEntityCertificateReplaceKeystoreParameters (*)
   endEntity:
       type: EndEntityHttps
   token: b0e889ff-847a-4d7d-bd17-c1292ddbb63e (*)
hostname.domainname service tls endEntityCertificate replace *> commit
```

CLI cookbook: configuring key-based SSH authentication for automation

This topic describes how to use CLI commands to configure individual users with SSH keys to allow for password-less authentication from a remote host to the CLI in an automated environment.

What is SSH Key-based authentication?

Secure Shell (SSH) is a connection method used to log into UNIX or Linux servers remotely. With Delphix, is it used to connect to the Delphix Command Line Interface (CLI) from a remote computer. This normally requires a password on each connection; however, it is possible to use Key-based Authentication to avoid the password requirement and allow the automation of Delphix commands.

Key-based Authentication relies on a public/private key pair generated on the client system. The private key allows access to any server acknowledging the matching public key as being authorized to login. In order to configure this, a public/private key pair must be created, and the resulting public key should be added to the Delphix server using the CLI.

The default domain user created on Delphix Engines is now admin instead of delphix_admin. When engines created before 5.3.1 are upgraded to 5.3.1 or later they will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then Disabling delphix_admin.

Procedure

1. Consult your client documentation for information on generating a public/private key pair. The `ssh-keygen` program is typical on UNIX platforms. If you need details on ssh-keygen usage or have unique requirements (such as named RSA keys), see Third-Party SSH Key Generation Example. If you already have a public/private key pair generated on your system, you can skip to step 2.
2. Connect as the user you wish to configure or as a Engine Administrator.

Warning: When you connect to the Delphix Engine with the CLI, you should specify the appropriate namespace (either DOMAIN or SYSTEM). See Connecting to the CLI for more information.

3. Select the current user, or select a specific user if configuring another user as an administrator.

```
delphix> user current
```

4. Update the user and set the SSH key.

```
delphix user "admin"> update
delphix user "admin" update *> set publicKey="[PASTE KEY]"
delphix user "admin" update *> commit
delphix>
```

Note:
Avoid Newline Characters with Public Key Entry The public key value, which can be quite long, must be entered as a single string with no newlines. When copying and pasting the public key, be sure to avoid introducing any newline characters. For more information on how to manage multiple public keys for password-less user authentication on Delphix, please visit this Knowledge Base article.

5. Verify you can authenticate through the Delphix CLI without a passphrase.
   Example Using Default SSH Key

```
ssh admin@DOMAIN@delphix-server.example.com
Last login: Thu Dec 13 22:16:28 2012 from 192.168.0.2
delphix>
```

   Example Using a Non-default SSH Key File Located at path/to/delphix_key

```
ssh -i path/to/delphix_key admin@DOMAIN@delphix-server.example.com
Last login: Thu Dec 13 22:16:28 2012 from 192.168.0.2
delphix>
```

CLI cookbook: setting up SSH key authentication for UNIX environment users

This topic describes adding public-key authentication for a UNIX environment user, thus allowing the Delphix server to connect to your UNIX Environments without an explicit password. This method uses the Delphix CLI in order to set up the environment user and gather SSH keys.

UNIX host environments (and Oracle cluster environments) can have users configured to use SSH-key-based authentication instead of the traditional password authentication method.

Prerequisites

- You must be able to log into the remote host (or all hosts of an Oracle cluster) and have write access to the `~/.ssh/authorized_keys` file within the desired user's home directory.

Option 1: system key

Within Delphix, there is a per-system SSH public key that can be placed into the `~/.ssh/authorized_keys` file of the remote user. Once this has been done, the Delphix environment user can be configured to use the private key instead of an explicit password. Note that it is also possible to configure an environment to use this system key in the Delphix Management application by navigating to **Manage** > **Environments** and selecting **Public Key** as the **Login Type** for the environment. For details, see Managing Environments.

1. Get the current system public key:

```
delphix> system get sshPublicKey
    ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAQEAse1M7uJX44lVPBljhnxB6MZUTx8VF6cupaVATg120lQonIqx29l
P+Mwp0AWh7C983IDoYDo+AY7RXpcFP9nKksiJnGSGiK6wo9RIiqSnF1x/
VXNkTt2/67RVofoiui4W5fuxD4hOIvoTr47Bg1hh9L6nhP0tnUvS/
rusHFJ+ogxGHm46mwNlgUJUGmLTNao+W0YU693HRLukEch01t4k6olVGaC0eLjYlgBf0Z5XiIcBX6ZW
qVHAhwMinVjAvmfQhirAgCI7gYrd5/PwNl/DC8xyhWuxd2jgA7sSPeRqWY0JHt/
xcmdpIaPxTwtxQLKTnPxrFrQd+l4uf6LKxr5g7w== root@delphix
```

2. Add this key (starting with `ssh-rsa`) to the remote user's `~/.ssh/authorized_keys` file. You will need to get access to this file using an alternate authentication mechanism (such as logging in as the user with a password or logging in as an administrator). Depending on the target OS, you may need to do the following:
    a. If the directory does not exist:

```
$ mkdir ~/.ssh
```

    b. If creating the file or directory as an administrator:

```
# chown -R <username> <home>/.ssh
```

    c. If required by the host SSH configuration, ensure the directory is not world-readable:

```
$ chmod 600 ~/.ssh/authorized_keys
$ chmod 755 ~
```

3. Create or edit an environment user:

```
delphix> environment user create
```

4. Set the user environment and name:

```
delphix environment user create *> set environment=environment1
delphix environment user create *> set name=username
```

5. Set the user credential type to `SystemKeyCredential`:

```
delphix environment user create *> set credential.type=SystemKeyCredential
```

6. Commit the results:

```
delphix environment user create *> commit
```

Option 2: Per-environment key pair

Each environment user can also be configured to use an SSH key pair provided via the CLI or API.

1. Add the public key to the remote user's `~/.ssh/authorized_keys` file. You will need to get access to this file using an alternate authentication mechanism (such as logging in as the user with a password or logging in as an administrator). Depending on the target OS, you may need to do the following:
   a. If the directory does not exist:

```
$ mkdir ~/.ssh
```

   b. If creating the file or directory as an administrator:

```
# chown -R <username> <home>/.ssh
```

   c. If required by the host SSH configuration, ensure the directory is not world-readable:

```
$ chmod 600 ~/.ssh/authorized_keys
$ chmod 755 ~
```

2. Create or edit an environment user:

```
delphix> environment user create
```

3. Set the user environment and name:

```
delphix environment user create *> set environment=environment1
delphix environment user create *> set name=username
```

4. Set the user credential type to `KeyPairCredential`:

```
delphix environment user create *> set credential.type=KeyPairCredential
```

5. Set the private and public keys:

```
delphix environment user create *> set credential.privateKey="----BEGIN ..."
delphix environment user create *> set credential.publicKey="ssh-rsa AA..."
```

(these example values were trimmed for brevity)

6. Commit the results:

```
delphix environment user create *> commit
```

CLI cookbook: configuring SSH host verification for UNIX environments

This topic describes how to configure SSH host verification when authenticating to UNIX environments, which lets the Delphix server ensure it connects to the intended environment hosts. This method uses the Delphix CLI to set the SSH key or fingerprint of each host. Currently, it is only possible to configure SSH host verification via the CLI or the Web Service API.

When an SSH key or fingerprint is specified for an environment host, the Delphix server will use it when connecting to that host to verify that host. If the key or fingerprint does not match the information presented by that host, the Delphix server will close that connection and report the problem to the user.

The key types supported by the Delphix server are `RSA` (`ssh-rsa`), `DSA` (`ssh-dsa`), `ECDSA` (`ecdsa-sha2-nistp256`) and `ED25519` (`ssh-ed25519`). The fingerprint types supported are `SHA256` and `SHA512`; the `MD5` type is considered insecure and, therefore, is not supported.

Prerequisites

- To obtain the SSH public key or fingerprint of a host remotely from another machine, you must have the `ssh-keyscan` and `ssh-keygen` utilities.
- To obtain the SSH public key or fingerprint directly from a host, you must be able to log into that host.

Obtaining an SSH key or fingerprint

1. Remotely: List the SSH public keys of the host using the standard utility `ssh-keyscan` and choose one of them. For example:

   ```
   $ ssh-keyscan example.environment.host # example.environment.host:22 SSH-2.0-
   OpenSSH_7.4 example.environment.host ssh-rsa
   AAAAB3NzaC1yc2EAAAADAQABAAABAQDBsOAAokSRO67jI28syRmX0wY/fKIboLLu/
   ofk6BzYLKtkMaK1QC78/6QlelIJUP5HdK8E7Um/iM1JMxry4h9Rl13onYOuJVZkDB9wnJiztSu/
   Wl9Eqbt59TU1vGmp/4ulWS3PISl7bxs+l43HzsrjM4dTs2efQ7sLWoW86CDlL7Je4va65/
   aopvifxKZeZkT0srB3L8VzHKw9+NJOumy1CI3DIBiICURJd4WZ10IH5TFUDRaUFAc/trzW1gvJY/
   Whp892tPHekyP32hOZNIc7oDPx2boZauJVR6/
   BHmKpmLlhkPpEqfZP8JW+JNsNnLr9BEmwJXaEpwnua1BUii8F ...
   ```

   where the key is the Base64-code string to the right of the key type. In this example, the RSA SSH public key is the string starting in "`AAAAB3Nza`" and ending in "`a1BUii8F`".
2. Alternatively, from the host: Log into the host and print the file contents of your public key of choice. For example:

   ```
   $ cat /etc/ssh/ssh_host_rsa_key.pub ssh-rsa
   AAAAB3NzaC1yc2EAAAADAQABAAABAQDBsOAAokSRO67jI28syRmX0wY/fKIboLLu/
   ofk6BzYLKtkMaK1QC78/6QlelIJUP5HdK8E7Um/iM1JMxry4h9Rl13onYOuJVZkDB9wnJiztSu/
   Wl9Eqbt59TU1vGmp/4ulWS3PISl7bxs+l43HzsrjM4dTs2efQ7sLWoW86CDlL7Je4va65/
   aopvifxKZeZkT0srB3L8VzHKw9+NJOumy1CI3DIBiICURJd4WZ10IH5TFUDRaUFAc/trzW1gvJY/
   Whp892tPHekyP32hOZNIc7oDPx2boZauJVR6/
   BHmKpmLlhkPpEqfZP8JW+JNsNnLr9BEmwJXaEpwnua1BUii8F
   ```
3. If a fingerprint is preferred, use `ssh-keygen` in conjunction with the above commands. For example, remotely:

```
$ ssh-keyscan example.environment.host | ssh-keygen -E sha256 -lf - ... 2048
 SHA256:8Cx8cBg/pSbkId3uu2vATeugkAXcm+Ruu9hu66OXEGI example.environment.host
(RSA) ...
```

where the fingerprint is " `SHA256:8Cx8cBg/pSbkId3uu2vATeugkAXcm+Ruu9hu66OXEGI` " (the string between the key size and the hostname).
Alternatively, from the host:

```
$ cat /etc/ssh/ssh_host_rsa_key.pub | ssh-keygen -E sha256 -lf - 2048
 SHA256:8Cx8cBg/pSbkId3uu2vATeugkAXcm+Ruu9hu66OXEGI user@environment.host (RSA)
```

Configuring SSH host verification during environment creation

The default SSH verification strategy is `SshAcceptAlways`, which always trusts the key or fingerprint presented by a remote host. The procedure to change this strategy to perform fingerprint-based host verification for single-host Unix environments is:

1. Set the new strategy to `SshVerifyFingerprint`:

```
delphix environment create *> edit hostParameters.host.sshVerificationStrategy
delphix environment create hostParameters.host.sshVerificationStrategy *> set
type=SshVerifyFingerprint
```

2. Set the key type and fingerprint type. For example:

```
delphix environment create hostParameters.host.sshVerificationStrategy *> set
keyType=RSA delphix environment create
hostParameters.host.sshVerificationStrategy *> set fingerprintType=SHA256
```

3. Set the fingerprint. For example:

```
delphix environment create hostParameters.host.sshVerificationStrategy *> set
fingerprint=SHA256:8Cx8cBg/pSbkId3uu2vATeugkAXcm+Ruu9hu66OXEGI
```

4. Alternatively, you can specify the key itself using the `SshVerifyRawKey` strategy. For example:

```
delphix environment create hostParameters.host.sshVerificationStrategy *> set
type=SshVerifyRawKey delphix environment create
hostParameters.host.sshVerificationStrategy *> set keyType=RSA delphix
environment create hostParameters.host.sshVerificationStrategy *> set
rawKey=AAAAB3NzaC1yc2EAAAADAQABAAABAQDBsOAAokSRO67jI28syRmX0wY/fKIboLLu/
ofk6BzYLKtkMaK1QC78/6QlelIJUP5HdK8E7Um/iM1JMxry4h9Rl13onYOuJVZkDB9wnJiztSu/
Wl9Eqbt59TU1vGmp/4ulWS3PISl7bxs+l43HzsrjM4dTs2efQ7sLWoW86CDlL7Je4va65/
aopvifxKZeZkT0srB3L8VzHKw9+NJOumy1CI3DIBiICURJd4WZ10IH5TFUDRaUFAc/trzW1gvJY/
Whp892tPHekyP32hOZNIc7oDPx2boZauJVR6/
BHmKpmLlhkPpEqfZP8JW+JNsNnLr9BEmwJXaEpwnua1BUii8F
```

5. When you are done specifying all other environment creation parameters, create the environment:

```
delphix environment create *> commit
```

6. If you are creating a Unix cluster, the procedure to start editing the SSH verification settings for the first node in that cluster is similar to the single-host case. For example:

```
delphix> environment create delphix environment create *> set
type=OracleClusterCreateParameters delphix environment create *> edit nodes
delphix environment create nodes *> add delphix environment create nodes 0 *>
edit delphix environment create nodes 0 *> edit
hostParameters.host.sshVerificationStrategy # configure SSH verification
settings
```

Note that only one node (host) can be specified and configured when creating a Unix cluster environment. The SSH verification settings for the remaining hosts can only be specified afterward by editing them via "`host select <hostname> update`", once the corresponding nodes have been discovered or added. See the next section.

Configuring SSH host verification for existing hosts

For any Unix environment host, whether it is single or part of a cluster, you can set up or change its configuration for SSH verification after the environment has been added by editing the host. For example:

```
delphix> host select example.environment.host update sshVerificationStrategy delphix
host 'example.environment.host' update sshVerificationStrategy *> edit
sshVerificationStrategy # configure SSH verification settings
```

Testing SSH host verification

It is possible to configure an SSH key or fingerprint when performing a connectivity test to a Unix host. This can be done without even creating an environment for that host. For example:

```
delphix> connectivity ssh # configure address and credentials ... delphix
connectivity ssh *> edit sshVerificationStrategy # configure SSH verification
settings
```

SSH host verification errors

When the Delphix server initiates an SSH connection to a host, if SSH host verification is configured (i.e. the verification strategy is not the default `SshAcceptAlways`), the server will first check the key presented by the host. Only if this check passes, the server will attempt to authenticate. Therefore, a host key verification failure will be reported as "Unrecognized key or fingerprint" to the user before any authentication failure. For example:

```
delphix connectivity ssh *> set credentials.password=<BAD PASSWORD> delphix
connectivity
```

CLI cookbook: system administration

These topics describe various system administration tasks that can be performed with the command-line interface, such as changing the name of the <default> group and setting up network connectivity.

This section covers the following topics:

CLI cookbook: setting NFS version

This topic describes how to configure the NFS version used for mounting VDBs.

Procedure

1. Log in to the Delphix Engine as the sysadmin user and switch to the Service NFS context. Then use the `ls` command to view the current configuration.

```
delphix service nfs> ls
Properties
    type: NfsConfig
    mountVersion: Automatic
```

2. Run the `update` command and configure the version (Options are Automatic, NFSv3, and NFSv4).

```
delphix service nfs> update
delphix service nfs update *> set mountVersion=NFSv4
```

3. Commit the operation.

```
delphix service nfs update *> commit
ip-10-110-212-129 service nfs> list
Properties
    type: NfsConfig
    mountVersion: NFSv4
```

CLI cookbook: configuring a second network interface

This topic describes how to configure a static IP address on a second network interface.

Procedure

1. Add a NIC to the Delphix virtual machine. The specific procedure will depend on the platform. For example, on VMware, a VMXNET3 virtual network adapter can be added dynamically from vSphere or other administrative interfaces, and the Delphix Engine will recognize the new NIC without a reboot. On other platforms, a reboot may be required for the Delphix Engine to recognize the new virtual hardware.
2. Log in to the Delphix Engine as the sysadmin user and switch to the network interface context. Then use the `list` command to view the available network interfaces, and select the new interface to be configured.

```
delphix network interface> list
NAME
vmxnet3s0
vmxnet3s1
delphix network interface> select vmxnet3s1
delphix network interface "vmxnet3s1"> get
    type: NetworkInterface
    name: vmxnet3s1
    addresses: (empty)
    device: vmxnet3s1
    macAddress: 0:c:29:e5:4c:c1
    mtu: 1500
    mtuRange: 60-9000
    reference: NETWORK_INTERFACE-vmxnet3s1
    state: DOWN
```

3. Run the `update` command and configure a static address.

```
delphix network interface "vmxnet3s1"> update
delphix network interface "vmxnet3s1" update *> edit addresses.0
delphix network interface "vmxnet3s1" update addresses.0 *> set address=10.1.2.
3/24
delphix network interface "vmxnet3s1" update addresses.0 *> get
    type: InterfaceAddress (*)
    address: 10.1.2.3/24 (*)
    addressType: STATIC (*)
```

4. `Commit` the operation.

```
delphix network interface "vmxnet3s1" update addresses.0 *> commit
delphix network interface "vmxnet3s1"> get
    type: NetworkInterface
    name: vmxnet3s1
    addresses:
        0:
```

```
        type: InterfaceAddress
        address: 10.1.2.3/24
        addressType: STATIC
        state: OK
device: vmxnet3s1
macAddress: 0:c:29:e5:4c:c1
mtu: 1500
mtuRange: 60-9000
reference: NETWORK_INTERFACE-vmxnet3s1
state: OK
```

Removing IP addresses from a network interface

This topic describes how to remove one or more IP addresses from a network interface.

Removing all IP addresses

Procedure

1. Log in to the Delphix Engine as the sysadmin user and switch to the network interface context. Then use the `list` command to view the available network interfaces, and select the interface.

```
delphix> network interface
delphix network interface> list
NAME    MACADDRESS         MTU
ens160  02:dc:02:00:7f:06  1500
delphix network interface> select ens160
delphix network interface 'ens160'> get
    type: NetworkInterface
    name: ens160
    addresses:
        0:
            type: InterfaceAddress
            address: 10.43.70.135/16
            addressType: DHCP
            enableSSH: true
            sessionInUse: false
            state: OK
    device: ens160
    macAddress: 02:dc:02:00:7f:06
    mtu: 1500
    mtuRange: 68-9000
    reference: NETWORK_INTERFACE-ens160
    state: OK
```

2. Run the `update` command and unset the addresses property to remove all addresses. Note that in this example, we only had one address to remove, but this procedure removes all addresses even if more than one were configured.

```
delphix network interface 'ens160'> update
delphix network interface 'ens160' update *> unset addresses
delphix network interface 'ens160' update *> get
    type: NetworkInterface
    name: ens160
    addresses: (unset) (*)
    mtu: 1500
```

3. `Commit` the operation.

```
delphix network interface 'ens160' update *> commit
```

```
delphix network interface 'ens160'> get
    type: NetworkInterface
    name: ens160
    addresses: (empty)
    device: ens160
    macAddress: 02:dc:02:00:7f:06
    mtu: 1500
    mtuRange: 68-9000
    reference: NETWORK_INTERFACE-ens160
    state: OK
```

Removing a specific IP address

1. Log in to the Delphix Engine as the sysadmin user and switch to the network interface context. Then use the `list` command to view the available network interfaces, and select the interface. Note that in this example, there are two addresses, a static address, and a DHCP address.

```
delphix> network interface
delphix network interface> list
NAME    MACADDRESS         MTU
ens160  02:dc:02:00:7f:06  1500
delphix network interface> select ens160
delphix network interface 'ens160'> get
    type: NetworkInterface
    name: ens160
    addresses:
        0:
            type: InterfaceAddress
            address: 10.11.12.13/24
            addressType: STATIC
            enableSSH: true
            sessionInUse: false
            state: OK
        1:
            type: InterfaceAddress
            address: 10.43.70.135/16
            addressType: DHCP
            enableSSH: true
            sessionInUse: false
            state: OK
    device: ens160
    macAddress: 02:dc:02:00:7f:06
    mtu: 1500
    mtuRange: 68-9000
    reference: NETWORK_INTERFACE-ens160
    state: OK
```

2. Run the `update` command and unset one of the addresses. In this example, only the static address is removed, retaining the DHCP address.

```
delphix network interface 'ens160'> update
delphix network interface 'ens160' update *> get
    type: NetworkInterface
    name: ens160
    addresses:
        0:
            type: InterfaceAddress
            address: 10.11.12.13/24
            addressType: STATIC
            enableSSH: true
        1:
            type: InterfaceAddress
            address: 10.43.70.135/16
            addressType: DHCP
            enableSSH: true
    mtu: 1500
delphix network interface 'ens160' update *> unset addresses.0
delphix network interface 'ens160' update *> get
    type: NetworkInterface
    name: ens160
    addresses:
        0:
            type: InterfaceAddress (*)
            address: 10.43.70.135/16 (*)
            addressType: DHCP (*)
            enableSSH: true (*)
    mtu: 1500
```

3. `Commit` the operation.

```
delphix network interface 'ens160' update *> commit
delphix network interface 'ens160'> get
    type: NetworkInterface
    name: ens160
    addresses:
        0:
            type: InterfaceAddress
            address: 10.43.70.135/16
            addressType: DHCP
            enableSSH: true
            sessionInUse: false
            state: OK
    device: ens160
    macAddress: 02:dc:02:00:7f:06
    mtu: 1500
    mtuRange: 68-9000
    reference: NETWORK_INTERFACE-ens160
    state: OK
```

CLI cookbook: adding a static route

This topic describes how to add a static route.

Procedure

1. Log in to the Delphix Engine as the sysadmin user and switch to the network route context.

```
delphix network route> list
DESTINATION      GATEWAY       OUTINTERFACE  PROTOCOL
default          172.16.0.1    ens192        DHCP
10.1.2.0/24      -             ens224        KERNEL
172.16.0.0/24    -             ens192        KERNEL
```

2. Run the `create` command to add a new route.

```
delphix network route> create
delphix network route create *> set destination=192.168.11.0/24
delphix network route create *> set gateway=10.1.2.1
delphix network route create *> get
    type: NetworkRoute
    destination: 192.168.11.0/24 (*)
    gateway: 10.1.2.1 (*)
    outInterface: (unset)
```

3. Optional outInterface Property Setting the `outInterface` property is optional, as the system will automatically determine the output interface based on the gateway address provided, as shown below.
4. `Commit` the operation.

```
delphix network route create *> commit
delphix network route> list
DESTINATION      GATEWAY       OUTINTERFACE  PROTOCOL
default          172.16.0.1    ens192        DHCP
10.1.2.0/24      -             ens224        KERNEL
172.16.0.0/24    -             ens192        KERNEL
192.168.11.0/24  10.1.2.1      ens224        STATIC
```

CLI cookbook: changing the default group name

This topic describes how to change the name of the default group <New Group> on the Delphix Engine as a simple example of CLI interactions. You must have delphix_admin credentials to perform this procedure.

Procedure

1. Switch to the group context and list groups on the system.

```
delphix> group
delphix group> list
NAME         DESCRIPTION
<New Group>  -
```

2. Select the default group and show current properties.

```
delphix group> select ""
delphix group ""> get
    name: <New Group>
    type: Group
    description: (unset)
    reference: GROUP-1
```

3. Run the `update` command and set the name.

```
delphix group ""> update
delphix group "" update *> set name=default
delphix group "" update *> get
    name: default (*)
    type: Group
    description: (unset)
    reference: GROUP-1
```

4. `Commit` the operation.

```
delphix group "" update *> commit
delphix group "default">
```

CLI cookbook: how to change a Delphix user password

> The default domain user created on Delphix Engines is now admin instead of delphix_admin. When engines created before 5.3.1 are upgraded to 5.3.1 or later they will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then Disabling delphix_admin.

1. ssh into your engine with a user that has Admin privileges.

```
ssh admin@delphix
```

2. Go to Users and select the User you would like to change the password of.

```
delphix > user
delphix user > ls
delphix user > select example_user
delphix user "example_user" > ls
```

3. Select `updateCredential` to allow you to change the password and set a new password.

```
delphix user "example user" > updateCredential
delphix user "example_user" updateCredential *> set newCredential.password=<new
 password>
```

4. Commit the operation.

```
delphix user "example_user" updateCredential *> commit
```

Example:

```
ssh admin@delphixengine
delphixengine > user
delphixengine user > ls

Objects

NAME           EMAILADDRESS

sysadmin       -

admin  no@delphix.com

test_user      no@delphix.com


Operations
```

```
create

current

delphixengine user > select test_user
delphixengine user "test_user" > ls

Properties

    type: User

    name: test_user

    authenticationType: NATIVE

    credential:

        type: PasswordCredential

        password: ********

    emailAddress: no@delphix.com

    enabled: true

    firstName: (unset)

    homePhoneNumber: (unset)

    isDefault: true

    lastName: (unset)

    locale: en_US

    mobilePhoneNumber: (unset)

    passwordUpdateRequested: false

    principal: test_user

    publicKey: (unset)

    reference: USER-2

    sessionTimeout: 30min

    userType: DOMAIN

    workPhoneNumber: (unset)
```

```
Operations

delete

update

disable

enable

updateCredential

delphixengine user "test_user" > updateCredential
delphixengine user "test_user" updateCredential *> set newCredential.password=<n
ew password>
delphixengine user "test_user" update *> commit
```

CLI cookbook: retrieve capacity information

This topic describes how to gather capacity information from your Delphix Engine. This information includes:

- dSource Space Breakdown
- Virtual Object Space Breakdown
- Total Space

Procedure

1. Switch to the capacity system context.

```
delphix> capacity system
```

2. List the properties of this content.

```
delphix capacity system> ls
Properties
type: CurrentSystemCapacityData
source:
type: CapacityBreakdown
activeSpace: 940582400B
actualSpace: 1075381760B
descendantSpace: 134583808B
logSpace: 145920B
manualSpace: 0B
policySpace: 0B
syncSpace: 134583808B
timeflowUnvirtualizedSpace: 7725215744B
unvirtualizedSpace: 2624235520B
timestamp: 2015-12-11T11:49:18.998Z
totalSpace: 25568477184B
virtual:
type: CapacityBreakdown
activeSpace: 176684032B
actualSpace: 313768448B
descendantSpace: 0B
logSpace: 47820288B
manualSpace: 85958144B
policySpace: 0B
syncSpace: 85958144B
timeflowUnvirtualizedSpace: 5475587584B
unvirtualizedSpace: 2667149312B
```

For more information about capacity management in Delphix, visit An Overview of Capacity and Performance Information

CLI cookbook: view storage test results

1. Log into the CLI as the sysadmin.

```
ssh sysadmin@yourengine
```

2. Navigate to storage test.

```
delphix > storage
delphix storage > test
```

3. List your tests and select the one that you would like to view and get the results.

```
delphix storage test > ls
delphix storage test > select STORAGE_TEST-X
delphix storage test STORAGE_TEST-X > result
delphix storage test STORAGE_TEST-X result *> commit
```

CLI cookbook: how to change IP address of Delphix Engine

This topic describes how to change an IP address on the Delphix Engine.

Procedure

1. Stop all running VDBs by clicking the **Stop** button on the VDB card.
2. Disable all dSources.
3. Log into the Delphix Engine as **sysadmin** user and switch to the network interface context. Then use the list command to view the available network interfaces, and select the public interface to be changed.

```
delphix> network
delphix network> setup
delphix network interface> list
NAME
vmxnet3s0
delphix network interface> select vmxnet3s0
delphix network interface 'vmxnet3s0'> get
    type: NetworkInterface
    name: vmxnet3s0
    addresses:
        0:
            type: InterfaceAddress
            address: 10.1.2.3/24
            addressType: STATIC
            enableSSH: true
            state: OK
    dataNode: DATA_NODE-34
    device: vmxnet3s0
    macAddress: 0:c:29:32:96:a3
    mtu: 1500
    mtuRange: 60-9000
    reference: NETWORK_INTERFACE-vmxnet3s0-DATA_NODE-34
    state: OK
```

4. Run the update command and update the address to the new IP address.

```
delphix network interface 'vmxnet3s0'> update
delphix network interface 'vmxnet3s0' update *> edit addresses.0
delphix network interface 'vmxnet3s0' update addresses.0 *> get
Properties
    type: InterfaceAddress
    address: 172.16.151.154/24
    addressType: STATIC
    enableSSH: true

delphix network interface 'vmxnet3s0' update addresses.0 *> set address=10.1.2.
4/24
delphix network interface 'vmxnet3s0' update addresses.0 *> get
    type: InterfaceAddress (*)
    address: 10.1.2.4/24 (*)
```

```
    addressType: STATIC (*)
    enableSSH: true (*)
```

5. Commit the operation.

```
delphix network interface 'vmxnet3s0' update addresses.0 *> commit
delphix network interface 'vmxnet3s0'> get
    type: NetworkInterface
    name: vmxnet3s0
    addresses:
        0:
            type: InterfaceAddress
            address: 10.1.2.4/24
            addressType: STATIC
            enableSSH: true
            state: OK
    dataNode: DATA_NODE-34
    device: vmxnet3s0
    macAddress: 0:c:29:32:96:a3
    mtu: 1500
    mtuRange: 60-9000
    reference: NETWORK_INTERFACE-vmxnet3s0-DATA_NODE-34
    state: OK
```

6. Re-enable the VDBs and dSources running from the engine.

CLI cookbook: about alert notifications

The Delphix Engine can send out email notifications when alerts happen. Alert profiles control this functionality.

An alert profile is composed of two things:

- **Filter specification:** A filter, or combination of filters, that specifies which alerts are of interest.
- **Alert action:** This specifies the email addresses to which the Delphix Engine will send an email when an alert matches the filter specification.

By default, the Delphix Engine has a single alert profile configured with the following parameters:

- Filter Specification: Match any alert with a severity level of **CRITICAL** or **WARNING**.
- Alert Actions: Send an email to the address defined for user **admin**.

> The default domain user created on Delphix Engines is now admin instead of delphix_admin. When engines created before 5.3.1 are upgraded to 5.3.1 or later they will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then Disabling delphix_admin.

Using the CLI, it is possible to:

- Modify the system default alert profile
- Create additional profiles in addition to the default one
- Set multiple actions for a single profile, such as "email dephix_admin" *and* "email user1@mycompany.com"

Simple filters

- Filtered by Owner of alerts target – for example, objects owned by user 1

Complex filters

Complex filters combine/modify other sub-filters:

- “And” filter – Used when all conditions defined must be met for the filter to notify the user with an email
- “Or” filter – Used when either one or the other of the conditions defined in the filters must be met for the filter to notify the user with an email
- “Not” filter – Used to exclude items

Limitations

- This is a CLI feature.
- Alert Profiles do not override permission settings. If you do not have Read permission on an object then your alert profile will never get triggered for that object's alerts, regardless of your filter settings.

The following CLI examples will run through how to create these three filters. Each example provides three different methods of setting up a profile. These include the following:

- A simple profile
- A profile with two filters
- A complicated profile

For more information, see CLI Cookbook: creating alert profiles

A simple profile

A simple profile approach matches the Delphix out-of-the-box default alert profiles. To create a simple alert profile using the CLI as seen in the figure below, go into the alert profile section of the command-line interface (CLI) and create a new profile. Line four prompts the engine to send an email when the filters are triggered. The following

three command lines refer to the filter specifications. Follow two severity levels: warning and critical. This will trigger an email alert when any warning or critical events occur.

```
twalsh-trunk.dcenter> cd alert
twalsh-trunk.dcenter alert> cd profile
twalsh-trunk.dcenter alert profile> create
twalsh-trunk.dcenter alert profile create *> set actions.0.type=AlertActionEmailUser
twalsh-trunk.dcenter alert profile create *> set filterSpec.type=SeverityFilter
twalsh-trunk.dcenter alert profile create *> set filterSpec.severityLevels.0=CRITICAL
twalsh-trunk.dcenter alert profile create *> set filterSpec.severityLevels.1=WARNING
twalsh-trunk.dcenter alert profile create *> commit
```

A compound alert profile

Creating a compound alert profile in the CLI will combine two filters together. This profile triggers an email about any alert on objects owned by the delphix_admin plus any other alert that is critical. The compound alert profile creates two filters. The first one will be the target owner filter, which in this case is **admin**. The second filter is the severity filter, allowing users to match anything that is critical. Combine these two filters using the OR logic so that if any of the sub-filters match, the whole filter matches. An example of this can be seen in the figure below.

Alert profile using OR logic

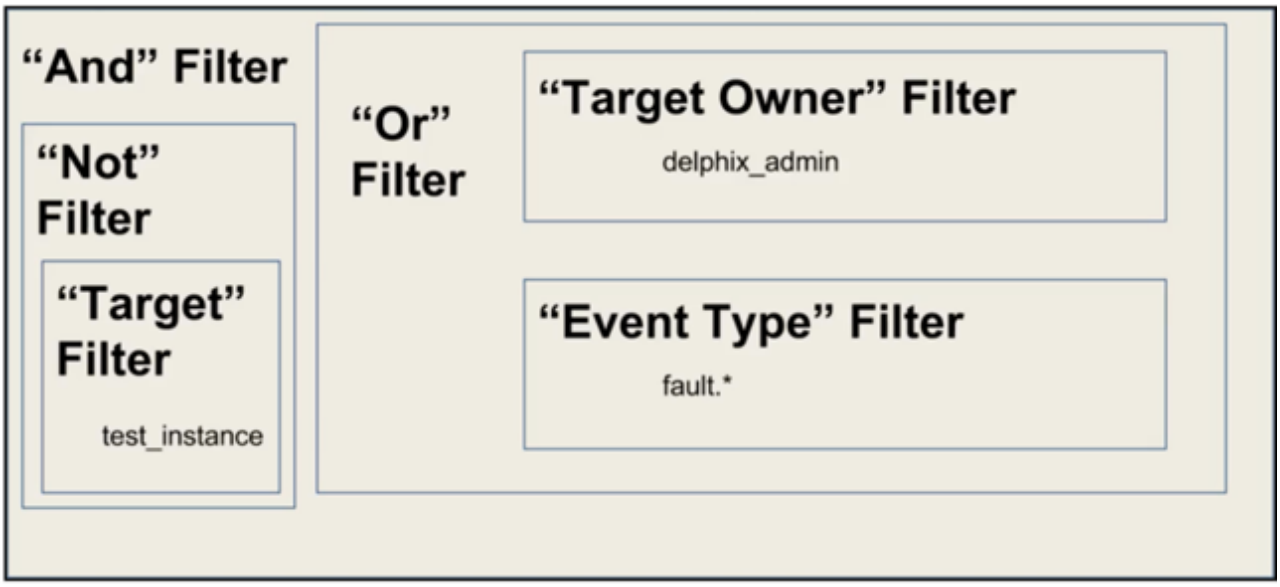
While working in the CLI, the first four lines describe a simple alert profile. The distinction between simple and compound alert profiles is that in a compound profile, the top-level filter uses an OR filter with sub-filters for target owner and severity level, as seen in line five of the figure below.

A compound alert profile

Complex alert profile

A complex alert profile uses the profile filter created in the compound alert profile and modifies it. For the example shown in the figure below, you have a VDB named test_instance that you do not need any emails about. The following commands will create an effective filter.

1. Create an OR filter with two sub filters: target owner and event type.
2. Create a NOT filter which will exclude the VDB (test_instance) from which you do not want to receive notifications.
3. Use the AND logic to combine all these filters together, as seen below.



Complex alert profile

Below is an example of the command lines used to set up this complex profile.

```
twalsh-trunk.dcenter> cd alert
twalsh-trunk.dcenter alert> cd profile
twalsh-trunk.dcenter alert profile> create
twalsh-trunk.dcenter alert profile create *> set actions.0.type=AlertActionEmailUser
twalsh-trunk.dcenter alert profile create *> set filterSpec.type=AndFilter
twalsh-trunk.dcenter alert profile create *> set filterSpec.subFilters.0.type=NotFilter
twalsh-trunk.dcenter alert profile create *> edit filterSpec.subFilters.0.subFilter
twalsh-trunk.dcenter alert profile create filterSpec.subFilters.0.subFilter *> set type=TargetFilter
twalsh-trunk.dcenter alert profile create filterSpec.subFilters.0.subFilter *> set targets.0=test_instance
twalsh-trunk.dcenter alert profile create filterSpec.subFilters.0.subFilter *> back
twalsh-trunk.dcenter alert profile create *> set filterSpec.subFilters.1.type=OrFilter
twalsh-trunk.dcenter alert profile create *> set filterSpec.subFilters.1.subFilters.0.type=TargetOwnerFilter
twalsh-trunk.dcenter alert profile create *> set filterSpec.subFilters.1.subFilters.0.owners.0=delphix_admin
twalsh-trunk.dcenter alert profile create *> set filterSpec.subFilters.1.subFilters.1.type=SeverityFilter
twalsh-trunk.dcenter alert profile create *> set filterSpec.subFilters.1.subFilters.1.severityLevels=CRITICAL
twalsh-trunk.dcenter alert profile create *> commit
```

Complex alert profile CLI

CLI cookbook: creating alert profiles

This article describes how to create alert profiles.

Delphix generates alerts for different events. Users may want to be notified of events based on certain criteria such as the type of event or severity. An alert profile allows a user or group of users to be notified of the desired event.

The default domain user created on Delphix Engines is now admin instead of delphix_admin. When engines created before 5.3.1 are upgraded to 5.3.1 or later they will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then Disabling delphix_admin.

Procedure

1. ssh into your engine using your delphix_admin username and password.

```
ssh admin@yourdelphixengine
```

2. Go into your alerts and list the alerts you already have.

```
delphix > alert
delphix alert > ls
```

3. Create your profile.

```
delphix alert > profile
delphix alert profile > create
delphix alert profile create * > ls
```

4. Set Actions, filterSpec, and User.

1. **Warning:**
   Valid Values for Parameters
   actions.0.type:
   a. AlertActionEmailList: This type of alert is used to create an alert for any number of users. When this type is selected, an email address may be defined in each element of the "actions.0.addresses" array as illustrated above.
   b. AlertActionEmailUser: This type of alert is created for the email address of the user currently logged into the command-line interface. The "actions.0.addresses" array is not available for this type.

5. filterSpec.type:
   a. AndFilter
   b. EventFilter
   c. NotFilter
   d. OrFilter
   e. SeverityFilter
   f. TargetFilter
   g. TargetOwnerFilter

```
delphix alert profile create *> set actions.0.type=<AlertActionEmailList or
AlertActionEmailUser>
delphix alert profile create *> set actions.0.addresses.0=<email address to send to>
delphix alert profile create *> set actions.0.addresses.1=<additional email address>
delphix alert profile create *> set actions.0.addresses.2=<additional email address>
delphix alert profile create *> ls

delphix alert profile create *> set filterSpec.type=SeverityFilter
delphix alert profile create *> set filterSpec.severityLevels=<AUDIT|WARNING|
CRITICAL|INFORMATIONAL>
```

5. Commit your changes.

```
delphix alert profile create *> commit
```

Example

```
ssh admin@yourengine
delphix > alert
delphix alert> ls
Objects
REFERENCE  TIMESTAMP                 TARGETNAME
EVENTTITLE
ALERT-102  2015-01-14T21:00:04.380Z  ASE/pubs2
Job complete
ALERT-101  2015-01-14T20:55:57.880Z  ASE/pubs2VDB
Job complete
ALERT-100  2015-01-14T19:35:32.958Z  ASE/pubs2VDB
Job complete
ALERT-99   2015-01-14T19:35:32.850Z  ASE/pubs2VDB
Job complete
ALERT-98   2015-01-14T19:34:58.744Z  ASE/pubs2
Error during job execution
ALERT-97   2015-01-14T18:12:01.928Z  ASE/pubs2
Job complete
ALERT-96   2015-01-14T18:03:10.664Z  ASE/pubs2
Job complete
ALERT-95   2015-01-14T17:16:07.464Z  ASE/pubs2
Job complete
ALERT-94   2015-01-14T17:15:55.298Z  ASE/market
Job complete
ALERT-93   2015-01-14T17:15:45.995Z  ASE/pubs2VDB
Job complete
ALERT-92   2015-01-14T16:39:33.133Z  nstacksolase2.acme.com-2015-01-14T16:39:13.821Z
Job complete
ALERT-91   2015-01-14T16:38:33.719Z  nstacksolase2.acme.com
Job complete
ALERT-90   2015-01-14T15:47:35.005Z  market
Validated sync failed for dSource
ALERT-89   2015-01-14T15:45:40.895Z  pubs2
Validated sync failed for dSource
```

```
ALERT-88   2015-01-14T15:02:14.874Z  ASE/market
Job complete
ALERT-87   2015-01-14T11:33:28.766Z  ASE/pubs2VDB
Job complete
ALERT-86   2015-01-13T23:11:46.838Z  ASE/market
Job complete
ALERT-85   2015-01-13T11:30:01.154Z  ASE/pubs2VDB
Job complete
ALERT-84   2015-01-13T11:07:04.385Z  pubs2
Backup detection failed
ALERT-83   2015-01-12T22:35:18.774Z  pubs2
Backup detection failed
ALERT-82   2015-01-12T11:30:00.063Z  ASE/pubs2VDB
Unable to connect to remote database during virtual database policy enforcement
ALERT-81   2015-01-12T11:30:00.054Z  ASE/pubs2
Unable to connect to remote database during dSource policy enforcement
ALERT-80   2015-01-12T08:38:26.983Z  pubs2
Backup detection failed
ALERT-79   2015-01-12T06:04:34.666Z  pubs2
Validated sync failed for dSource
ALERT-78   2015-01-11T11:30:03.393Z  ASE/pubs2VDB
Job complete
Children
profile
delphix alert> select ALERT-98
delphix alert "ALERT-98"> ls
Properties
    type: Alert
    event: alert.jobs.failed.object
    eventAction: Create the database on the target host.
    eventDescription: DB_EXPORT job for "ASE/pubs2" failed due to an error during
execution: Could not find database "pubs2VDB" on target instance "SRC_157_4K",
environment "ASE".
    eventSeverity: CRITICAL
    eventTitle: Error during job execution
    reference: ALERT-98
    target: ASE/pubs2
    targetName: ASE/pubs2
    targetObjectType: ASEDBContainer
    timestamp: 2015-01-14T19:34:58.744Z
delphix alert> profile
delphix alert profile> select ALERT_PROFILE-1
delphix alert profile "ALERT_PROFILE-1"> ls
Properties
    type: AlertProfile
    actions:
        0:
            type: AlertActionEmailList
            addresses: sys_admin@acme.com
            format: HTML
    filterSpec:
        type: SeverityFilter
        severityLevels: CRITICAL,WARNING
```

```
    reference: ALERT_PROFILE-1
    user: admin
Operations
delete
update
delphix alert profile> create
delphix alert profile create *> set actions.0.type=AlertActionEmailList
delphix alert profile create *> set actions.0.addresses.0=johndoe@acme.com
delphix alert profile create *> set actions.0.addresses.1=samsmith@acme.com
delphix alert profile create *> set filterSpec.type=SeverityFillter
delphix alert profile create *> set filterSpec.severityLevels=INFORMATIONAL
```

*The last piece of the alert profile that needs to be configured is the "targetFilter". This is an array so you can define multiple targets. In the following example, there is a dSource named "pubs2" the user wants to define an alert on. If they try to set the filter to just the name of the dSource itself ("pubs2"), it will warn them that this is ambiguous and gives a hint on how to fully qualify it:

```
delphix > alert profile create
delphix alert profile create *> ls
Properties
 type: AlertProfile
 actions:
 0:
 type: AlertActionEmailList (*)
 addresses: foo@bar.com (*)
 format: HTML (*)
 filterSpec: (unset)
delphix alert profile create *> edit actions
delphix alert profile create actions *> add
delphix alert profile create actions 0 *> set addresses=foo@bar.com
delphix alert profile create actions 0 *> back
delphix alert profile create actions *> back
delphix alert profile create *> set filterSpec.type=TargetFilter
delphix alert profile create *> set filterSpec.targets="REPLICATION_SPEC_EXECUTE"
delphix alert profile create *> commit
```

Use the tab button freely to autocomplete and also see available options, for instance, while changing the severityLevels property, you can use the tab key like so:

```
DELPHIX-4221.dcenter alert profile 'ALERT_PROFILE-1' update *> set
filterSpec.severityLevels= <I HIT TAB HERE TO SEE OPTIONS BELOW>

AUDIT          CRITICAL       INFORMATIONAL  WARNING
```

Please note that the above tab autocomplete feature will only work if the filterSpec.type is already set to SeverityFilter.

Note on names used in the example

SRC_157_4K: Repository (entity containing the database instances)

ASE: Group name

pubs2: Name of an individual database instance

The user sets the target filter to be equal to "pubs2/pubs2" and "ASE/pubs2" because if you review the "TARGETNAME" column from step 1 above, you will see alerts generated for both of these targets.

CLI cookbook: obtaining CPU performance information using CLI

There are times when it may be desirable to obtain analytics information about the Delphix Engine CPU, which is formatted differently than what is available from the Delphix User Interface (UI).

Troubleshooting

You can obtain running information by looking at the **GUI > Resources > Performance Analytics**. You will see the graph for CPU if the CPU is selected.

When logged into the Delphix Engine command-line interface (CLI) as delphix_admin, you can see the default analytics gathered:

```
dlpx5120.dcenter> analytics
dlpx5120.dcenter analytics> list
NAME STATISTICTYPE STATE COLLECTIONINTERVAL COLLECTIONAXES
default.cpu CPU_UTIL RUNNING 1sec idle,kernel,user
default.disk DISK_OPS RUNNING 1sec op,avgLatency,latency,count,throughput
default.iscsi iSCSI_OPS RUNNING 1sec op,latency,count,throughput
default.network NETWORK_INTERFACE_UTIL RUNNING 1sec
outBytes,networkInterface,inPackets,inBytes,outPackets
default.nfs NFS_OPS RUNNING 1sec op,latency,count,throughput
default.tcp TCP_STATS RUNNING 1sec
congestionWindowSize,localPort,remotePort,receiveWindowSize,inUnorderedBytes,s
endWindowSize,retransmittedBytes,outBytes,localAddress,roundTripTime,inBytes,u
nacknowledgedBytes,remoteAddress
```

Resolution

You can also obtain some information from the CLI, using the Performance Analytics Tool API Reference.

In order to view this information, you must log in with a user that had Delphix Admin privileges.

Option 1: Running an Existing Analytic

1.Login as admin and run `analytics.`

```
dlpx5120.dcenter> analytics
dlpx5120.dcenter analytics> list
NAME STATISTICTYPE STATE COLLECTIONINTERVAL COLLECTIONAXES
default.cpu CPU_UTIL RUNNING 1sec idle,kernel,user
default.disk DISK_OPS RUNNING 1sec op,avgLatency,latency,count,throughput
default.iscsi iSCSI_OPS RUNNING 1sec op,latency,count,throughput
default.network NETWORK_INTERFACE_UTIL RUNNING 1sec
outBytes,networkInterface,inPackets,inBytes,outPackets
default.nfs NFS_OPS RUNNING 1sec op,latency,count,throughput
default.tcp TCP_STATS RUNNING 1sec
congestionWindowSize,localPort,remotePort,receiveWindowSize,inUnorderedBytes,s
endWindowSize,retransmittedBytes,outBytes,localAddress,roundTripTime,inBytes,u
nacknowledgedBytes,remoteAddress

Children
statistic

Operations
create
```

2. To see the available properties for CPU analytics, `select` and list the `default.cpu.`

```
dlpx5120.dcenter analytics> select default.cpu
dlpx5120.dcenter analytics 'default.cpu'> ls
Properties
 type: StatisticSlice
 name: default.cpu
 axisConstraints: (empty)
 collectionAxes: idle,kernel,user
 collectionInterval: 1sec
 dataNode: DATA_NODE-1
 reference: ANALYTICS_STATISTIC_SLICE-1
 state: RUNNING
 statisticType: CPU_UTIL

Operations
delete
getData
pause
rememberRange
resume
stopRememberingRange
```

3. Specify `setopt trace=true` in order to see the `CpuUtilDatapointStream` datapoints.

```
dlpx5120.dcenter analytics 'default.cpu'> setopt trace=true

=== GET /resources/json/delphix/analytics/ANALYTICS_STATISTIC_SLICE-1 ===
=== RESPONSE ===
{
 "type": "OKResult",
 "status": "OK",
 "result": {
 "type": "StatisticSlice",
 "reference": "ANALYTICS_STATISTIC_SLICE-1",
 "namespace": null,
 "name": "default.cpu",
 "statisticType": "CPU_UTIL",
 "collectionInterval": 1,
 "state": "RUNNING",
 "collectionAxes": [
 "idle",
 "kernel",
 "user"
 ],
 "axisConstraints": [],
 "dataNode": "DATA_NODE-1"
 },
 "job": null,
 "action": null
}
=== END ===
```

4. Enter `getData` and `commit` to obtain the data gathered.

```
dlpx5120.dcenter analytics 'default.cpu'> getData
=== GET /resources/json/delphix/analytics/ANALYTICS_STATISTIC_SLICE-1 ===
=== RESPONSE ===
{
 "type": "OKResult",
 "status": "OK",
 "result": {
```

```
 "type": "StatisticSlice",
 "reference": "ANALYTICS_STATISTIC_SLICE-1",
 "namespace": null,
 "name": "default.cpu",
 "statisticType": "CPU_UTIL",
 "collectionInterval": 1,
 "state": "RUNNING",
 "collectionAxes": [
 "idle",
 "kernel",
 "user"
 ],
 "axisConstraints": [],
 "dataNode": "DATA_NODE-1"
 },
 "job": null,
 "action": null
}
=== END ===

dlpx5120.dcenter analytics 'default.cpu' getData *> commit
=== GET /resources/json/delphix/analytics/ANALYTICS_STATISTIC_SLICE-1/getData
===
=== RESPONSE ===
{
 "type": "OKResult",
 "status": "OK",
 "result": {
 "type": "DatapointSet",
 "resolution": 1,
 "datapointStreams": [
 {
 "type": "CpuUtilDatapointStream",
 "datapoints": [
 {
 "type": "CpuUtilDatapoint",
 "timestamp": "2016-12-06T13:53:30.000Z",
```

```
"idle": 1946,
"kernel": 33,
"user": 19,
"dtrace": null
},
{
"type": "CpuUtilDatapoint",
"timestamp": "2016-12-06T13:53:31.000Z",
"idle": 1966,
"kernel": 17,
"user": 14,
"dtrace": null
},
{
"type": "CpuUtilDatapoint",
"timestamp": "2016-12-06T13:53:32.000Z",
"idle": 1968,
"kernel": 17,
"user": 13,
"dtrace": null
},
{
"type": "CpuUtilDatapoint",
"timestamp": "2016-12-06T13:53:33.000Z",
"idle": 1963,
"kernel": 19,
"user": 17,
"dtrace": null
},
{
"type": "CpuUtilDatapoint",
"timestamp": "2016-12-06T13:53:34.000Z",
"idle": 1968,
"kernel": 16,
"user": 15,
"dtrace": null
},
```

```
...
...
 {
 "type": "CpuUtilDatapoint",
 "timestamp": "2016-12-06T19:54:16.000Z",
 "idle": 1922,
 "kernel": 36,
 "user": 41,
 "dtrace": null
 },
 {
 "type": "CpuUtilDatapoint",
 "timestamp": "2016-12-06T19:54:17.000Z",
 "idle": 1953,
 "kernel": 26,
 "user": 20,
 "dtrace": null
 }
 ],
 "cpu": null
 }
 ],
 "overflow": false
 },
 "job": null,
 "action": null
}
=== END ===
 type: DatapointSet
 datapointStreams:
 0:
 type: CpuUtilDatapointStream
 datapoints: [ ... ]
 overflow: false
 resolution: 1sec
```

Option 2: Creating a New Analytic

1. Log in as admin and then run `analytics`
2. Run `create` and `list` the properties.

```
dlpx5120.dcenter analytics> create
dlpx5120.dcenter analytics create *> ls
Properties
 type: StatisticSlice
 name: (required)
 axisConstraints: (unset)
 collectionAxes: (required)
 collectionInterval: (unset)
 dataNode: (unset)
 statisticType: (required)
```

3. Set the required Properties `name`, `collectionAxes` and statisticType (if you are unsure you can run help for options) for=""> for options)>

```
set name=test.cpu
set collectionAxes=kernel
set statisticType=CPU_UTIL

dlpx5120.dcenter analytics create *> ls
Properties
 type: StatisticSlice
 name: test.cpu
 axisConstraints: (unset)
 collectionAxes: kernel
 collectionInterval: (unset)
 dataNode: (unset)
 statisticType: CPU_UTIL
```

4. `Commit` to being data collection.

```
dlpx5120.dcenter analytics create *> commit


=== POST /resources/json/delphix/analytics ===
{
 "type": "StatisticSlice",
 "name": "test.cpu",
 "collectionAxes": [
 "kernel"
 ],
 "statisticType": "CPU_UTIL"
}
=== RESPONSE ===
{
 "type": "OKResult",
 "status": "OK",
 "result": "ANALYTICS_STATISTIC_SLICE-7",
 "job": null,
 "action": "ACTION-656"
}
=== END ===
 `ANALYTICS_STATISTIC_SLICE-7
```

5. Use `ls` to see the new `test.cpu` analytic

```
dlpx5120.dcenter analytics> ls

NAME STATISTICTYPE STATE COLLECTIONINTERVAL COLLECTIONAXES
default.cpu CPU_UTIL RUNNING 1sec idle,kernel,user
test.cpu CPU_UTIL RUNNING 1sec kernel
default.disk DISK_OPS RUNNING 1sec op,avgLatency,latency,count,throughput
default.iscsi iSCSI_OPS RUNNING 1sec op,latency,count,throughput
default.network NETWORK_INTERFACE_UTIL RUNNING 1sec
outBytes,networkInterface,inPackets,inBytes,outPackets
default.nfs NFS_OPS RUNNING 1sec op,latency,count,throughput
default.tcp TCP_STATS RUNNING 1sec
congestionWindowSize,localPort,remotePort,receiveWindowSize,inUnorderedBytes,s
endWindowSize,retransmittedBytes,outBytes,localAddress,roundTripTime,inBytes,u
nacknowledgedBytes,remoteAddress

Children
statistic

Operations
create
```

6. Select `test.cpu` and use `getData` and `commit` to see the data gathered.

```
dlpx5120.dcenter analytics> select test.cpu
dlpx5120.dcenter analytics  'test.cpu'> ls
dlpx5120.dcenter analytics  'test.cpu'> getData
dlpx5120.dcenter analytics  'test.cpu' getData *> commit
```

7. Use `delete` to stop the collection and delete `test.cpu`

```
dlpx5120.dcenter analytics 'test.cpu'> delete
dlpx5120.dcenter analytics 'test.cpu' delete *> commit
=== POST /resources/json/delphix/analytics/ANALYTICS_STATISTIC_SLICE-7/
delete ===
{}
=== RESPONSE ===
{
 "type": "OKResult",
 "status": "OK",
 "result": "",
 "job": null,
 "action": "ACTION-657"
}
=== END ===

dlpx5120.dcenter analytics> ls

NAME STATISTICTYPE STATE COLLECTIONINTERVAL COLLECTIONAXES
default.cpu CPU_UTIL RUNNING 1sec idle,kernel,user
default.disk DISK_OPS RUNNING 1sec op,avgLatency,latency,count,throughput
default.iscsi iSCSI_OPS RUNNING 1sec op,latency,count,throughput
default.network NETWORK_INTERFACE_UTIL RUNNING 1sec
outBytes,networkInterface,inPackets,inBytes,outPackets
default.nfs NFS_OPS RUNNING 1sec op,latency,count,throughput
default.tcp TCP_STATS RUNNING 1sec
congestionWindowSize,localPort,remotePort,receiveWindowSize,inUnorderedBytes,s
endWindowSize,retransmittedBytes,outBytes,localAddress,roundTripTime,inBytes,u
nacknowledgedBytes,remoteAddress

Children
statistic

Operations
create
dlpx5120.dcenter analytics>
```

CLI cookbook: rebooting the Delphix engine via CLI

Occasionally, the management stack will hang on the GUI, or routine maintenance will require a reboot of the Delphix Engine. You can do this safely through either the Delphix Setup or CLI. Before performing a reboot, it is important that all your VDBs are shut down and dSources disabled in order to maintain data integrity.

Prerequisites

- Shut down all VDBS
- Shut down dSources

Procedure

Complete the following steps to reboot the Delphix Engine via CLI:

1. Login to the CLI using the sysadmin **username** and **password**.

```
ssh sysadmin@yourdelphixengine
```

2. Go to **system > reboot**.

```
delphix > system
delphix system > reboot
```

3. Commit the action.

```
delphix system reboot *> commit
```

CLI cookbook: disabling user-click analytics

The Delphix User-click Analytics feature is a lightweight method to capture how users interact with Delphix product user interfaces. The goal of capturing this data is to get a better understanding of product usage, engagement, and user behavior, and to use this data to improve Delphix products and customer experience. This feature is enabled by default for customers deploying on or upgrading to this version. User-click Analytics may also be disabled via the UI.

This procedure will disable user-click analytics on both the Delphix Engine and Delphix Self-Service.

Procedure to disable user-click analytics

1. ssh into your engine with a user that has Admin privileges.

```
ssh sysadmin@delphix
```

2. Go to Services and select the userInterface.

```
delphix > cd service
delphix service > cd userInterface
```

3. Update the analyticsEnabled option to false.

```
delphix service userInterface > ls
Properties
 Type userInterfaceConfig
 analyticsenabled: true
Operations
delphix service userInterface > update
delphix service userInterface update *> set analyticsEnabled=false
delphix service userInterface update *> commit
```

CLI cookbook: changing the API version

This topic describes how to change the API version.

Procedure

1. Log in to the Delphix Engine as the admin user.

```
delphix> su - admin@DOMAIN
```

2. Run the version command.

```
delphix> version
1.10.3 # This lists the current API verison.
delphix> version <version you want to set to>
delphix> version
1.8.2 # Displays the new version that you had set above.
```

CLI cookbook: hosts and environments

This section contains the following topics:

CLI cookbook: adding a UNIX host

This topic describes the process of adding a UNIX host using the 3.0 command-line interface.

Within Delphix, there are both hosts and host environments. A host represents a remote system, but may or may not be a source or target for linking or provisioning. For example, in an Oracle RAC cluster, the cluster environment represents the location of the Oracle installation(s), and while there are hosts within that cluster they are not individually manageable as environments.

Procedure

1. Create a new environment and set the parameter type to be a UNIX host.

```
delphix> environment create
delphix environment create *> set type=HostEnvironmentCreateParameters
delphix environment create *> set hostEnvironment.type=UnixHostEnvironment
delphix environment create *> set hostParameters.type=UnixHostCreateParameters
delphix environment create *> set
primaryUser.credential.type=PasswordCredential
delphix environment create *> get
    type: HostEnvironmentCreateParameters
    hostEnvironment:
        type: UnixHostEnvironment
        name: (unset)
        aseHostEnvironmentParameters: (unset)
        description: (unset)
        logCollectionEnabled: false
    hostParameters:
        type: UnixHostCreateParameters
        host:
            type: UnixHost
            address: (required)
            dspKeystoreAlias: (unset)
            dspKeystorePassword: (unset)
            dspKeystorePath: (unset)
            dspTruststorePassword: (unset)
            dspTruststorePath: (unset)
            javaHome: (unset)
            nfsAddressList: (unset)
            oracleJdbcKeystorePassword: (unset)
            privilegeElevationProfile: (unset)
            sshPort: 22
            sshVerificationStrategy: (unset)
            toolkitPath: (required)
    logCollectionEnabled: false
    primaryUser:
        type: EnvironmentUser
        name: (unset)
        credential:
            type: PasswordCredential
            password: (required)
        environment: (unset)
```

```
        groupId: (unset)
        userId: (unset)
```

2. Set the host address. The name of the environment is derived from the address used, though you can provide a more descriptive name if desired. The address can be DNS names, IP addresses, or a comma-separated list of the above.

```
delphix environment create *> set hostParameters.host.address=192.168.1.2
```

3. Set the toolkit path. This is where Delphix will store temporary binaries used while the host is configured as part of Delphix.

```
delphix environment create *> set hostParameters.host.toolkitPath=/work
```

4. Set the username and password to use when connecting over SSH. This user must have the privileges described in the Delphix Administration Guide. To configure an SSH user, change the credential type to `SystemKeyCredential`.

```
delphix environment create *> set primaryUser.name=oracle
delphix environment create *> set primaryUser.credential.password
Enter primaryUser.credential.password: *******
```

5. Commit the result. The environment discovery process will execute as an asynchronous job. The default behavior is to wait for the result, so progress will be updated until the discovery process is complete or fails.

```
delphix environment create *> commit
    UNIX_HOST_ENVIRONMENT-4
    Dispatched job JOB-39
    ENVIRONMENT_CREATE_AND_DISCOVER job started for "192.168.1.2".
    ENVIRONMENT_CREATE_AND_DISCOVER job for "192.168.1.2" completed
successfully.
delphix>
```

CLI cookbook: adding a SQL Server source environment

This topic describes how to add a SQL Server source environment using the command line interface.

Since SQL Server source environments do not have the Delphix Connector running on them, you must use a target environment as a proxy when adding source environments. Delphix uses the connector running on the proxy environment to run commands against the source environment. See Overview of Requirements for SQL Server Environments for more information.

Procedure

Enter these commands through the command-line interface:

```
/environment; create;set type=HostEnvironmentCreateParameters; set
hostEnvironment.type=WindowsHostEnvironment;set hostEnvironment.name=<Source
environment name>;set hostEnvironment.proxy=<target host name>; set
hostParameters.type=WindowsHostCreateParameters;set
hostParameters.host.type=WindowsHost;set hostParameters.host.address="<Source host IP
address or hostname>"; set primaryUser.name="<domain\username>";set
primaryUser.credential.type=PasswordCredential;set
primaryUser.credential.password=<password>; commit;
```

Example

The CLI commands for adding source host `"mssql_source_1"` using target host `“mssql_target_1”` as proxy and environment user `“ad\delphix_user”` would be:

```
/environment; create;set type=HostEnvironmentCreateParameters; set
hostEnvironment.type=WindowsHostEnvironment;set hostEnvironment.name="mssql_source_1";
set hostEnvironment.proxy="mssql_target_1"; set
hostParameters.type=WindowsHostCreateParameters;set
hostParameters.host.type=WindowsHost;set hostParameters.host.address="mssql_source_1";
 set primaryUser.name="ad\delphix_user";set
primaryUser.credential.type=PasswordCredential;set
primaryUser.credential.password="i_am_the_password"; commit;
```

CLI cookbook: setting multiple addresses for a target host

This topic is an example of using arrays to configure a target host to support multiple IP addresses. The `nfsAddressList` property is an array of strings.

Procedure

1. Select the host to update

```
delphix> hostdelphix host> select exampledelphix host "example"> update
```

2. Set the address:

```
Delphix host '192.168.121.141' update *> set nfsAddressList="192.168.1.23,192.1
68.2.44"
```

3. Get the current addresses, both as a string and as an array object.

```
delphix host "example" update *> get nfsAddressList    192.168.1.23,192.168.2.4
4 (*)delphix host "example" update *> get nfsAddressList[0]    192.168.1.23
 (*)delphix host "example" update *> edit nfsAddressListdelphix host "example"
 update addresses *> get    0:        192.168.1.23 (*)    1:        192.168.2.4
4 (*)
```

4. Commit the result:

```
delphix host "example" update addresses *> commitdelphix host "example">
```

CLI cookbook: how to change environment user

1. ssh into your engine using Admin privileges

```
ssh admin@delphix
```

2. Go to Environment and find the Environment you would like to update

```
delphix > environmentdelphix environment > lsdelphix environment > select
test_env
```

3. Select Environment updating and Update

```
delphix environment "test_env" > updatedelphix environment "test_env" update *>
ls
```

4. Set `primaryUser` to new user you would like to use for the Environment (NOTE: new user should be added as an environment user before making as a primaryUser)

```
delphix environment "test_env update" *> set primaryUser=<new user>
```

5. Commit the operation.

```
delphix environment "test_env" update *> commit
```

Example:

```
ssh admin@delphixdelphix > environmentdelphix environment > ls ObjectsNAME
DESCRIPTIONDemo  ChildrenoracleuserOperationscreate delphix environment > select
Demodelphix environment "Demo" > updatedelphix environment "Demo" update *> ls
Properties    type: UnixHostEnvironment    name: Demo    description:
primaryUser: delphix delphix environment "Demo" update *> set primaryUser=<new
 user>delphix environment "Demo" update *> commit
```

CLI cookbook: how to create or edit a privilege elevation profile and profile scripts

Background:

If you are running a version prior to 6.0.0 please contact your Professional Services representative.

Procedure for creating an elevation profile:

1. Log into the CLI using delphix_admin or a user with Admin privileges and got to 'Host'.

```
ssh admin@yourengine
yourengine > host
```

2. Select `privilegeElevation` then `profile.`

```
yourengine host > privilegeElevation
yourengine host privilegeElevation > profile
```

3. Set the name of the profile and the version of the profile.

```
yourengine host privilegeElevation profile *> set name=<profilename>
yourengine host privilegeElevation profile *> set version=<profileversion>
```

4. Commit the profile to save it.

```
yourengine host privilegeElevation profile *> commit
```

Procedure for creating a profileScript

Please note that you will need to create the script yourself or with the help of the Professional Services team.

1. Log into the CLI using delphix_admin or a user with Admin privileges and got to 'Host'.

```
ssh delphix_admin@yourengine
yourengine > host
```

2. Select `privilegeElevation` then `profileScript.`

```
yourengine host > privilegeElevation
yourengine host privilegeElevation > profileScript
```

3. Create your script by setting, name, contents, and profile (this can be your previously created profile or the default sudo).

```
yourengine host privilegeElevation profileScript > create
yourengine host privilegeElevation profileScript *> set name=<scriptname>
yourengine host privilegeElevation profileScript *> set contents=<your script>
```

```
yourengine host privilegeElevation profileScript *> set profile=<yourprofile>
```

4. Commit to save.

```
yourengine host privilegeElevation profileScript *> commit
```

Privilege elevation profiles and Delphix replication

Because all Delphix Engines have a default Profile called sudo, which would normally exist on both source and target Delphix Engines, replication collisions which would normally prevent a successful failover are automatically resolved.

- Only Profiles which are actually assigned to a host are replicated. All currently unassigned profiles are ignored.
- Profile name collisions are resolved by the display names of duplicate Profiles being prefixed with a unique object identifier. This is described in more detail in the next section.
- A default Profile will not retain default status after replication failover. The Profile assigned as default on the replication target Delphix Engine will remain the default. Therefore, if the source Delphix Engine has a non-standard default Profile, it will need to be manually set as the new default on the replication target Delphix Engine after failover.

Caveats

By design, the Delphix Engine allows the creation of Profiles with duplicate names. This is not a bug. It exists for several reasons:

- This allows replication failover to complete without duplicate Profile names triggering a collision.
- Makes versioning possible so that a profile with the same name can have multiple versions as iterations are made (some of which may not be production-ready).

However, this behavior has the consequence of changing the display name of Profiles. Once a duplicate name exists, a unique object identifier is prefixed to the name. Any references to such a profile (such as assigning to a host) must use the long format with the unique identifier. For example, standard Delphix Engine has the following Profile:

```
Delphix5031HWv8> host privilegeElevation profile ls
Objects
NAME      ISDEFAULT
sudo      true
Operations
create
```

If a new profile that is also called "sudo" is created, the display names automatically change as follows:

```
Objects
NAME                                          ISDEFAULT
`HOST_PRIVILEGE_ELEVATION_PROFILE-1/sudo      true
`HOST_PRIVILEGE_ELEVATION_PROFILE-5/sudo      false

Operations
create
```

Known issues

- It is not possible to delete a Profile. Attempting to do so results in an API error. However, Profiles can be renamed to something meaningful like "unused_1".
- Profiles created that contain single quotation marks can no longer be selected. They become orphaned Profiles.

- When pasting in script contents, the cursor does not correctly move to the end of the last line being pasted. Unless the cursor is moved to the end of the line before pressing ENTER, the script will not be complete.

Neither of the above issues has any operational impact.

CLI cookbooks: enabling and configuring environment permissions

This topic describes how to enable the environment permissions feature to restrict what users can do with environments.

By default, all engine users can list all environments and hosts and see their details. Moreover, all users are able to link dSources from and provision VDBs to any environment without requiring any permissions on environments, as long as they have appropriate permissions on the target group where the dsource or VDB will be located.

Enabling environment and permissions

To restrict non-administrator users from seeing, linking from, and provisioning to any environment, Engine Administrators can enable environment authorizations.

```
delphix> authorization configurationdelphix authorization configuration >
lsProperties    type: AuthorizationConfig    environmentAndHostAuth: false
 Operationsupdatedelphix authorization configuration> updatedelphix authorization
configuration update *> set environmentAndHostAuth=truedelphix authorization
configuration update *> commit
```

Similarly, to go back to the default state in which all users have permission to perform those operations, the Engine Administrator must set the `environmentAndHostAuth` property back to `false`.

Granting and revoking permissions on environments and hosts

When environment permissions are enabled, only Engine Administrators can list environments and hosts, see their details, or link dSources from or provision VDBs to environments.

To authorize any other user to perform such an operation on an environment or host, a Engine Administrator must create an appropriate authorization.

```
delphix> authorization createdelphix authorization create *> set user=someuserdelphix
authorization create *> set role=PROVISIONERdelphix authorization create *> set
target=SourceEnvironment:/somehost.example.com
```

To revoke an authorization, a Engine Administrator must delete the corresponding authorization object.

```
delphix> authorizationdelphix> lsREFERENCE        USER      ROLE   TARGET
AUTHORIZATION-1  sysadmin  OWNER  sysadminAUTHORIZATION-2  admin     OWNER
adminAUTHORIZATION-3  admin     OWNER  domain0AUTHORIZATION-4  someuser  Data
SourceEnvironment:/somehost.example.com delphix authorization> select
`AUTHORIZATION-4delphix authorization '(USER-2, ROLE-2, UNIX_HOST_ENVIRONMENT-1)'>
deletedelphix authorization '(USER-2, ROLE-2, UNIX_HOST_ENVIRONMENT-1)' delete *>
commit
```

Permissions on Environments and Hosts

Role	Environment privileges	Host privileges
Owner	• Can provision VDBs from the environment • Can link dSources from the environment • Can access the same information as a Reader	• Can access the same information as a Reader
Provisioner	• Can access statistics on the dSource, VDB, or snapshot such as usage, history, and space consumption • Can provision VDBs from owned dSources and VDBs	• Can access the same information as a Reader
Data Operator	• Can access statistics on the dSource, VDB, or snapshot such as usage, history, and space consumption • Can refresh or rollback VDBs • Can snapshot dSources and VDBs	• Can access the same information as a Reader
Reader	• Can see the configuration of the environment	• Can see the configuration of the host
Self-Service Only	• Can access the same information as a Reader	• Can see the configuration of the host

CLI cookbook: source databases and dSources

These topics describe command-line interface procedures for working with dSources.

This section covers the following topics:

CLI cookbook: linking an Oracle staging push database

This topic describes how to link an Oracle Staging Push database using the Delphix Engine command-line interface.

Prerequisites

You will need the following information:

- The name of the dSource you want to create.
- The group in which you want to create the dSource.
- The Database Name of the Source Database whose data will be populated on this staging database.
- The Repository and the Mount Base.

Linking a CDB

Procedure

- Execute the *database link* command

```
delphix> database link
delphix database link>
```

- The default *linkData.type* in the *LinkParameters* is set to *ASELinkData*, but you can confirm that by getting the input type. Set *linkData.type* to *OracleLinkFromStaging* for Oracle Staging Push database.

```
delphix database link *> get linkData.type
    ASELinkData
delphix database link *> set linkData.type=OracleLinkFromStaging
delphix database link *> get linkData.type
    OracleLinkFromStaging (*)
delphix database link *> ls
Properties
    type: LinkParameters
    name: (required)
    description: (unset)
    group: (required)
    linkData:
        type: OracleLinkFromStaging (*)
        allowAutoStagingRestartOnHostReboot: (required)
        containerType: (required)
        customEnvVars: (unset)
        databaseName: (required)
        externalFilePath: (unset)
        operations: (unset)
        sourcingPolicy: (unset)
        stagingSourceParameters: (required)
        syncParameters:
            type: OracleStagingPushSyncParameters
        syncStrategy: (required)
        uniqueName: (required)

Operations
defaults
```

- Set the name for the CDB dSource and the group in which you want to create it.

```
delphix database link *> set name=exampleCDB1
delphix database link *> set group="<Group Name>"
```

- Set the container type to *ROOT_CDB* for the container database.

```
delphix database link *> set linkData.containerType=ROOT_CDB
```

- Set the *databaseName*, *uniqueName*, and other *linkData* properties. The database name here should be the same as the database name of the Source Database whose data will be populated on this staging database.

```
delphix database link *> set linkData.databaseName="<database_name>"
delphix database link *> set linkData.uniqueName=CDB_1_UNQ_NAME
delphix database link *> set linkData.allowAutoStagingRestartOnHostReboot=true
delphix database link *> set linkData.syncStrategy.type=OracleStagingPushSyncStrategy
```

- Set staging source properties - *mountBase, instanceName,* and *repository* among others.

```
delphix database link *> edit linkData.stagingSourceParameters
delphix database link linkData.stagingSourceParameters *> ls
```

```
Properties
    type: OracleStagingSourceParameters (*)
    configParams: (unset)
    configTemplate: (unset)
    environmentUser: (unset)
    instanceName: (required)
    mountBase: (required)
    physicalStandby: (unset)
    repository: (required)
 delphix database link linkData.stagingSourceParameters *> set mountBase=/mnt/staging
delphix database link linkData.stagingSourceParameters *> set
instanceName=CDB_1_INSTANCE_NAME
```

If you are unsure of the available repositories, you can list available repositories:

```
elphix database link linkData.stagingSourceParameters *> /repository list
NAME                                              VERSION     ENVIRONMENT
'/u01/app/oracle/product/18.0.0.0/dbhome_1'       18.0.0.0.0  env1
'/u01/app/oracle/product/19.8.0.0/dbhome_1'       19.8.0.0.0  env2
delphix database link linkData.stagingSourceParameters *> set repository='/u01/app/
oracle/product/19.8.0.0/dbhome_1'
```

- Commit the result.

```
delphix database link *> commit
    `ORACLE_DB_CONTAINER-45
    Dispatched job JOB-129
    DB_LINK job started for "<Group Name>/exampleCDB1".
    Creating new TimeFlow for dSource "exampleCDB1".
    Generating required scripts needed for the staging database.
    Mounting datasets on the staging host "<staging host>".
    Starting staging database "<database_name>" for dSource "exampleCDB1" in NOMOUNT
mode.
    DB_LINK job for "<Group Name>/exampleCDB1" completed successfully.
delphix>
```

Linking a PDB

Procedure

- Execute the *database link* command.

```
delphix> database link
delphix database link>
```

- Set *linkData.type* to *OraclePDBLinkFromStaging* for Oracle Staging Push pluggable database.

```
delphix database link *> set linkData.type=OraclePDBLinkFromStaging
```

- Set the name for the PDB dSource and the group in which you want to create it.

```
delphix database link *> set name=examplePDB1
delphix database link *> set group="<Group Name>"
```

- Set *linkData.cdbConfig* to the config of the linked staging push CDB dSource, where this PDB should be plugged into. For Oracle databases, configs are identified by the database's unique name. You can list available source configurations:

```
delphix database link *> /sourceconfig list
NAME                 REPOSITORY            LINKINGENABLED
CDB_1_UNQ_NAME    '/opt/ora/dexample1'  true
CDB_2_UNQ_NAME    '/opt/ora/dexample2'  true
delphix database link *> set linkData.cdbConfig=CDB_1_UNQ_NAME
```

- Set *databaseName, syncStrategy.type* and other *linkData* properties. The database name here should be the same as the database name of the source PDB whose data will be populated on this staging PDB.

```
delphix database link *> set linkData.databaseName=CDOMLOSR381EPDB1
delphix database link *> set linkData.syncStrategy.type=OracleStagingPushSyncStrategy
delphix database link *> set linkData.allowAutoStagingRestartOnHostReboot=true
```

- Commit the result.

```
delphix database link *> commit
examplePDB1
    Dispatched job JOB-136
    DB_LINK job started for "<Group Name>/examplePDB1".
    Creating new TimeFlow for dSource "examplePDB1".
    Generating required scripts needed for the staging database.
    Mounting datasets on the staging host "<staging_host>".
    DB_LINK job for "<Group Name>/examplePDB1" completed successfully.
```

CLI cookbook: detaching and attaching a SQL server dSource

This topic describes how to detach and attach a SQL server dSource. When attaching a SQL Server dSource to a new data source, the new data source must be in the same database satisfying the following constraints:

- pptRepository needs to be set to the name of the SQL instance on the staging server. The unlink operation removes the database from the SQL instance on the staging server and unmounts the iscsi luns, reattaching the dSource via the CLI will remount the iscsi luns and puts the database back.

Prerequisites

A dSource can only be attached to a new data source once it has been unlinked.

Procedure

1. Select dSource.

```
delphix> database "dexample"
```

2. Run the detachSourcecommand, specifying the current active source. This step can be skipped if the dSource has already been detached. through the GUI.

```
delphix database "dexample"> detachSourcedelphix database "dexample"
 detachSource *> set source=dexampledelphix database "dexample" detachSource *>
commit
```

3. Run the `attachSource` command.

```
delphix database "dexample"> attachSource
```

4. Set the following for SQL Server: You can also type help pptRepository to see what is wanted You can also set pptRepository=<then press tab> to list all values.

```
delphix database "dexample" attachSource *> edit attachDatadelphix database
"dexample" attachSource attachData *> set type=MSSqlAttachDatadelphix database
"dexample" attachSource attachData *> set config=SQLSERVER/dexampledelphix
database "dexample" attachSource attachData *> set sharedBackupLocations="<\
\SERVER1\Backups>"delphix database "dexample" attachSource attachData *> set
pptRepository=SQL2008R2delphix database "dexample" attachSource attachData *>
set mssqlUser.type=MSSqlDomainUserdelphix database "dexample" attachSource
attachData *> set mssqlUser.user=ad\dbuserdelphix database "dexample"
 attachSource attachData *> set mssqlUser.password.password=dbuserpwddelphix
database "dexample" attachSource attachData *> edit ingestionStrategydelphix
database "dexample" attachSource attachData ingestionStrategy *>set
type=ExternalBackupIngestionStrategydelphix database "dexample" attachSource
attachData ingestionStrategy *>set validatedSyncMode=TRANSACTION_LOGdelphix
database "dexample" attachSource attachData ingestionStrategy *> backdelphix
database "dexample" attachSource attachData *> edit operations.preSyncdelphix
database "dexample" attachSource attachData  operations.preSync *> backdelphix
```

```
database "dexample" attachSource attachData*> edit operations.postSyncdelphix
database "dexample" attachSource attachData operations.postSync *> back
```

5. Commit the operation.

```
delphix database "dexample" attachSource *> commit
```

CLI cookbook: disabling LogSync for a dSource

This topic provides a simple example of how the nested state is represented and manipulated. The LogSync state is maintained in the sourcingPolicy property of dSources, itself an object with several different fields.

Procedure

1. Select the dSource to be changed and run the `update` command.

```
delphix> database "example"
delphix "example"> update
```

2. Get the current property using dot-delimited notation.

```
delphix "example" update *> get sourcingPolicy.logsyncEnabled
    true
```

3. The property could also be set using dot-delimited notation, but for illustrative purposes, we can also use the `edit` command and set it directly.

```
delphix "example" update *> edit sourcingPolicy
delphix "example" update sourcingPolicy *> set logsyncEnabled=false
```

4. Commit the state, either from within the editing context or after running `back` to return to the parent context.

```
delphix "example" update sourcingPolicy *> commit
delphix "example">
```

CLI cookbook: enabling Oracle validated sync

Prerequisite - designating a staging host

In order to validate an Oracle dSource snapshot for syncing, the Delphix Engine requires a host with an Oracle installation that is compatible with the dSource. This machine is known as the **staging** host. You must explicitly designate which machines you want the Delphix Engine to use as staging hosts. All machines that have been marked as staging sites are added to a pool. During validated sync, the Delphix Engine will select a compatible host from the pool, export the requisite archived redo logs and datafiles, and execute Oracle media recovery on the host. Follow these steps to designate a staging host.

1. Select the repository you want to designate as staging.

```
delphix>/repository/select '/u01/app/ora10205/product/10.2.0/db_1'
```

2. Execute the `update` command.

```
delphix repository "'/u01/app/ora10205/product/10.2.0/db_1'">update
```

3. Set staging to `true`.

```
delphix repository "'/u01/app/ora10205/product/10.2.0/db_1'" update *>set
staging=true
```

4. `Commit` the operation to designate the repository as staging.

```
delphix repository "'/u01/app/ora10205/product/10.2.0/db_1'" update *> commit
```

To configure validated sync for multiple dSources with different Oracle versions, you must designate a compatible staging source for each. If multiple compatible staging sites exist, the Delphix Engine will select one at random.

The validated sync process will consume some resources on the staging host when snapshots are taken. Designating a performance-critical host as a staging host is not recommended.

Procedure - Enabling validated sync

1. Select the dSource for which you want to enable validated sync.

```
delphix>/database/select redsox1
```

2. Execute the `update` command.

```
delphix database "redsox1">update
```

3. Set `preProvisioningEnabled` to `true`.

```
delphix database "redsox1" update *>set preProvisioningEnabled=true
```

4. `Commit` the operation to enable validated sync.

```
delphix database "redsox1" update *>commit
```

CLI cookbook: linking a SQL Server database loading from a specific full backup of the source database

This topic describes how to use the command-line interface to link a SQL Server database by loading from a specific full backup of the source database as indicated by the backup UUID.

Prerequisites

- You can get the backup UUID for the all backup files of a chosen database using the following query on the source database under the column ***backup_set_uuid***

```
Use masterselect backupset.database_name,      backupset.type,
backupset.backup_set_id,      backupset.backup_set_uuid,
backupset.family_guid,      backupset.position,      backupset.first_lsn,
backupset.last_lsn,      backupset.database_backup_lsn,      backupset.name,
backupset.has_bulk_logged_data,      backupset.is_damaged,
backupset.begins_log_chain,      backupset.is_copy_only,
backupset.backup_finish_date,      backupset.database_version,
backupset.database_guid,mediafamily.logical_device_name,mediafamily.physical_de
vice_namefrom msdb.dbo.backupmediafamily mediafamily join msdb.dbo.backupset
backupseton mediafamily.media_set_id = backupset.media_set_id where
backupset.database_name = N'<Database Name>'order by
backupset.backup_finish_date desc
```

Procedure

Enter these commands through the Delphix Engine command-line interface:

```
/database; link;set type=LinkParameters;set name=<dSource name>;set group=<group
name>;set linkData.type=MSSqlLinkData;set
linkData.syncParameters.type=MSSqlExistingSpecificBackupSyncParameters;set
linkData.config=<source database>;set linkData.sharedBackupLocations="<source
database backup locations>";set linkData.pptRepository=<SQL instance on the staging
server>;set linkData.sourcingPolicy.type=SourcingPolicy;set
linkData.mssqlUser.type=MSSqlDomainUserset linkData.mssqlUser.user=ad\dbuserset
linkData.mssqlUser.password.password=dbuserpwdset
linkData.syncParameters.backupUUID=<backup UUID>;set
linkData.ingestionStrategy.type=<ingestion strategy type>;set
linkData.ingestionStrategy.validatedSyncMode=<validated sync mode type>; commit;
```

CLI cookbook: linking a SQL Server database loading from the last full backup of the source database

This topic describes how to use the command-line interface to link a SQL Server database by loading from the last full backup of the source database.

Procedure

Enter the following commands in the Delphix Engine command-line interface:

```
/database; link;set type=LinkParameters;set name=<dSource name>;set group=<group
name>; set linkData.type=MSSqlLinkData;set
linkData.syncParameters.type=MSSqlExistingMostRecentBackupSyncParametersset
linkData.config=<source database>;set linkData.sharedBackupLocations="<source
database backup locations>";set linkData.pptRepository=<SQL instance on the staging
server>;set linkData.sourcingPolicy.type=SourcingPolicy;set
linkData.mssqlUser.type=MSSqlDomainUserset linkData.mssqlUser.user=ad\dbuserset
linkData.mssqlUser.password.password=dbuserpwdset
linkData.ingestionStrategy.type=<ingestion strategy type>; commit;
```

CLI cookbook: linking to a single instance Oracle database

This topic describes how to link to a single instance Oracle database using the Delphix Engine command-line interface.

Prerequisites

You will need the following information:

- The name of the dSource you want to create.
- The group in which you want to create the dSource.
- The database unique name of the Oracle database you want to link to.
- The database username/password with sufficient privileges as described in the Delphix User Guide.
- The host environment user with sufficient privileges as described in the Delphix User Guide.

Procedure

1. Execute the `database link` command.

```
delphix> database linkdelphix database link>
```

2. The default `linkData.type` in the `LinkParameters` is set to `ASELinkData`, but you can confirm that by getting the input type. Set `linkData.type` to `OracleLinkFromExternal` for Oracle database.

```
delphix database link *> get linkData.type    ASELinkDatadelphix database link
*> set linkData.type=OracleLinkFromExternaldelphix database link *> get
linkData.type    OracleLinkFromExternal (*)delphix database link *>
lsProperties    type: LinkParameters    name: (required)    description:
(unset)    group: (required)    linkData:        type: OracleLinkFromExternal
(*)        backupLevelEnabled: (unset)        bandwidthLimit: (unset)
checkLogical: (unset)        compressedLinkingEnabled: (unset)        config:
(required)        dbCredentials:            type: PasswordCredential
password: (required)        dbUser: (required)        diagnoseNoLoggingFaults:
(unset)        encryptedLinkingEnabled: (unset)        environmentUser:
(required)        externalFilePath: (unset)        filesPerSet: (unset)
linkNow: (unset)        nonSysCredentials: (unset)        nonSysUser: (unset)
numberOfConnections: (unset)        operations: (unset)
preProvisioningEnabled: (unset)        rmanChannels: (unset)
sourcingPolicy: (unset)        syncParameters:            type:
OracleSyncFromExternalParameters            doNotResume: (unset)
doubleSync: (unset)            forceFullBackup: (unset)
skipSpaceCheck: (unset)Operationsdefaults
```

3. Set the name for the dSource and the group in which you want to create it.

```
delphix database link *> set name=example1delphix database link *> set group=""
```

4. Set the source configuration. For Oracle databases, these are identified by the database unique name. If you are unsure of the set of available databases, you can list available source configurations.

```
delphix database link *> /sourceconfig listNAME      REPOSITORY
LINKINGENABLEDexample1  '/opt/ora/dexample1'  trueexample2  '/opt/ora/
dexample1'  truedelphix database link *> set linkData.config=example1
```

5. Set the privileged database username/password.
   The password can be set like other properties, or the value can be omitted so that it can be manually inputted without exposing the password.

```
delphix database link *> set linkData.dbUser=delphixdelphix database link *> set
linkData.dbCredentials.passwordEnter dbCredentials.password: ********
```

6. Set the privileged environment user.

   This user must be from the same environment as the associated source config set in step 4. You can list the set of available users through the `environment user list` command.

```
delphix database link *> /environment/user listNAME         oracledelphix database
link *> set linkData.environmentUser=oracle
```

7. Adjust any other properties you may want, such as RMAN tunables, description, and whether to link now.
   The full set of options is described in the API documentation for the `OracleLinkFromExternal` type. If you set the `linkNow` property, then this operation will wait for the sync to complete, otherwise, you can perform the initial link by running the sync command at a later point

```
delphix database link *> set linkData.linkNow=true
```

8. Commit the result.

```
delphix database link *> commit    `ORACLE_DB_CONTAINER-1    Dispatched job
JOB-8    DB_LINK job started for "/example1".    Obtaining information from
source database "/example1".    Creating new TimeFlow for dSource "/example1".
The dSource "example1" was successfully linked from source database "/example1".
    DB_LINK job for "/example1" completed successfully.delphix>
```

CLI cookbook: listing data source sizes

This topic describes a basic use of the CLI `list` command.

1. Switch to the source view and view the default list.

```
delphix> sourcedelphix source> listNAME      CONTAINER  VIRTUAL  CONFIG example
example    false    examplevexample  vexample   true     vexample
```

2. List sources with their database size.

```
delphix> sourcedelphix source> select exampledelphix source 'example'> get
runtime.databaseSize    2.80GB
```

CLI cookbook: detaching and attaching an Oracle dSource

This topic describes how to attach a dSource to a different data source.

Prerequisites

Before detaching an Oracle dSource, you must capture the following RMAN configuration:

- Level or SCN based backups
- Data load channel settings: number of channels and files per channel

The above RMAN configuration will be removed once the dSource is unlinked.

A dSource can only be attached to a new data source once it has been unlinked.

When attaching an Oracle dSource to a new data source, the new data source must be the same logical database satisfying the following constraints:

- Same dbid
- Same dbname
- Same creation time
- Same resetlogs SCN
- Same resetlogs time
- Same redo stream, where a log must exist with
    - Same sequence
    - Same thread
    - Same end SCN

For Oracle dSources, this procedure can be used to initially link from a standby server that is faster or less disruptive, unlink the dSource, and then attach it to the production server for subsequent incremental SnapSync operations. When you perform the attach operation, you will need the source config name of an unlinked database.

Procedure

1. Select **dSource**.

```
delphix> database "dexample"
```

2. Run the `detachSource` **command**, specifying the currently active source. This step can be skipped if the dSource has already been detached through the GUI.

```
delphix database "dexample"> detachSourcedelphix database "dexample"
 detachSource *> set source=name-of-old-src-DB-serverdelphix database
"dexample" detachSource *> commit
```

3. Run the `attachSource` **command**.

```
delphix database "dexample"> attachSource
```

4. Set the config to point to an unlinked source. The following is an example to attach to an Oracle data source: (Hint: use <TAB> to complete variable names and known values)

```
delphix database "dexample" attachSource *> set
attachData.type=OracleAttachDatadelphix database "dexample" attachSource *> set
attachData.config=name-of-dSource-as-shown-in-environmentdelphix database
"dexample" attachSource *> set attachData.environmentUser=myuserdelphix
database "dexample" attachSource *> set
attachData.oracleFallbackUser=orauserdelphix database "dexample" attachSource
*> edit attachData.oracleFallbackCredentialsdelphix database "dexample"
 attachSource *> set set type=PasswordCredentialdelphix database "dexample"
 attachSource *> set password=orauserpwd
```

The attachData.config listing can be observed in the administration GUI, under Manage (pull-down at the top) --> Environments --> choose your dSource --> select Databases (upper right).
The dSources shown on the new source DB server may acquire odd names eg. "P02:UNKNOWN:vwxyz". These are a side-effect of having multiple instances of the same container name (possible in DR); Delphix needs to disambiguate. If these do show, they indicate containers in the new environment. Successfully attaching to these and deleting the old environment will re-establish the original names.

5.
```
delphix database "dexample" attachSource *> commit
```

CLI cookbook: how to change database user password

1. ssh into your engine using Admin privileges.

```
ssh admin@delphixengine
```

2. Go to sourceconfig and find the Database that you need to update the password on.

```
delphix > sourceconfigdelphix sourceconfig > lsdelphix sourceconfig > select
<yourdatabase>
```

3. Update the password.

```
delphix sourceconfig "yourdatabase" > updatedelphix sourceconfig "yourdatabase"
 update *> lsdelphix sourceconfig "yourdatabase" update *> set
credentials.password=<new password>
```

4. Commit the change.

```
delphix sourceconfig "database" update *> commit
```

Example:

```
ssh admin@exampledelphix > sourceconfigdelphix sourceconfig > lsObjects NAME
REPOSITORY                  LINKINGENABLEDmeta1  '/u01/oracle/10.2.0.4/ee1'  true
 Operationscreate delphix sourceconfig > select meta1delphix sourceconfig "meta1" >
ls  Properties    type: OracleSIConfig    name: meta1    credentials:        type:
PasswordCredential        password: ********    databaseName: meta1    discovered:
true    environmentUser: delphix    instance:        type: OracleInstance
instanceName: meta1        instanceNumber: 1    linkingEnabled: true
nonSysCredentials: (unset)    nonSysUser: (unset)    reference:
ORACLE_SINGLE_CONFIG-1    repository: '/u01/oracle/10.2.0.4/ee1'    services:
0:            type: OracleService            discovered: true
jdbcConnectionString: jdbc:oracle:thin:@172.16.100.69:1525:meta1        1:
type: OracleService            discovered: true            jdbcConnectionString:
jdbc:oracle:thin:@172.16.100.69:1521:meta1    uniqueName: meta1    user: delphix
Operationsdeleteupdatevalidatecredentials delphix sourceconfig "meta1" >
updatedelphix sourceconfig "meta1" update *> set credentials.password=<new
 password>delphix sourceconfig "meta1" update *> commit
```

CLI cookbook: changing an SAP ASE dSource's staging database

Prerequisites

In order to change an SAP ASE dSource's staging database, you need to know the name of the staging database and you need to disable the dSource.

Procedure

1. To find the name of the staging database, hover the mouse over the "Staging Database" name on the dSource card in the GUI or issue the following command via the CLI (replacing "pubs2" with the name of the dSource and "delphix.acme.com" with the hostname of your Delphix Engine):

```
$ echo "/source; select pubs2; ls" | ssh admin@delphix.acme.com | grep
stagingSourcePassword:     stagingSource: dxpb91a7LJlwj933xvlPvd_pubs2
```

2. Disable the dSource by either toggling the "Enable/Disable" toggle in the Delphix Management application or by using the CLI:

```
$ ssh admin@delphix.acme.comdelphix> sourcedelphix source> select pubs2delphix
source 'pubs2'> disabledelphix source 'pubs2' disable *> set
type=SourceDisableParameters delphix source 'pubs2' disable *> commit
Dispatched job JOB-365    SOURCE_DISABLE job started for "pubs2".
SOURCE_DISABLE job for "pubs2" completed successfully.
```

3. Change the repository of the staging database to reside in the desired SAP ASE instance:

```
$ ssh admin@delphix.acme.comdelphix> sourceconfigdelphix sourceconfig> select
dxpb91a7LJlwj933xvlPvd_pubs2delphix sourceconfig 'dxpb91a7LJlwj933xvlPvd_pubs2'>
 updatedelphix sourceconfig 'dxpb91a7LJlwj933xvlPvd_pubs2' update *>
lsProperties    type: ASESIConfig    credentials: (unset)    databaseName:
dxpb91a7LJlwj933xvlPvd_pubs2    environmentUser: nstacksolasetest/sybase
linkingEnabled: false    repository: nstacksolasetest/SRC_157_4K    user:
(unset)delphix sourceconfig 'dxpb91a7LJlwj933xvlPvd_pubs2' update *> set
environmentUser=nstacksolasestg/sybasedelphix sourceconfig
'dxpb91a7LJlwj933xvlPvd_pubs2' update *> set repository=nstacksolaseprod/
SRC_157_4K                              NSTACK_16K
RH68_ASE157_S1SQL2008R2                                                SQL2012
nstacksolasestg                                          nstacksolasetest/
SRC_157_4Kdelphix sourceconfig 'dxpb91a7LJlwj933xvlPvd_pubs2' update *> set
repository=nstacksolasestgdelphix sourceconfig 'dxpb91a7LJlwj933xvlPvd_pubs2'
 update *> commit
```

Hint: Rapidly hit the "tab" key twice after typing "set repository=" to make the CLI present a list of available instances.

4. Enable the dSource:

```
delphix sourceconfig 'dxpb91a7LJlwj933xvlPvd_pubs2'> /sourcedelphix source>
select pubs2delphix source 'pubs2'> enabledelphix source 'pubs2' enable *> set
type=SourceEnableParameters delphix source 'pubs2' enable *> commit
```

```
Dispatched job JOB-367    SOURCE_ENABLE job started for "pubs2".
SOURCE_ENABLE job for "pubs2" completed successfully.
```

CLI cookbook: detaching and attaching a SAP ASE dSource

This CLI cookbook recipe describes how to Detach and Attach an SAP ASE dSource using the CLI.

Prerequisites

A dSource can only be attached to a new data source once it has been unlinked.
When attaching an SAP ASE dSource to a new data source, the new data source must be the same logical database satisfying the following constraints:

- Same dbid
- Same dbname
- Same creation time

You must also make sure that you follow the normal prerequisites for an SAP ASE data source found in SAP ASE Support and Requirements.

Procedure

Detach a dSource

1. Login to the CLI as admin or a user with Admin privileges.
2. Select dSource.

```
delphix> database "dexample"
```

3. Run the `detachSourcecommand`, specifying the currently active source. Note: This step can be skipped if the dSource has already been detached through the GUI.

```
delphix database "dexample"> detachSourcedelphix database "dexample"
 detachSource *> set source=dexampledelphix database "dexample" detachSource *>
commit
```

Attach a dSource

1. Login to the CLI as admin or a user with Admin privileges.
2. Run the `attachSource` command.

```
delphix database "dexample"> attachSourcedelphix database "dexample"
 attachSource *> set attachData.config=dexampledelphix database "dexample"
 attachSource *> set attachData.dbCredentials.password=sybasedelphix database
"dexample" attachSource *> set attachData.dbUser=sadelphix database "dexample"
 attachSource *> set attachData.loadBackupPath=/tmp/backupsdelphix database
"dexample" attachSource *> set
attachData.sourceHostUser="source_host_environment/sybase"delphix database
"dexample" attachSource *> set
attachData.stagingHostUser="staging_host_environment/sybase"delphix database
"dexample" attachSource *> set
attachData.stagingRepository="staging_ASE_servername_example"delphix database
"dexample" attachSource *> lsProperties      type: ASEAttachSourceParameters
attachData:          type: ASEAttachData          config: dexample (*)
```

```
dbCredentials:              type: PasswordCredential              password:
******** (*)         dbUser: sa (*)                 dumpCredentials: (unset)
externalFilePath: (unset)          loadBackupPath: /tmp/backups (*)
loadLocation: (unset)         mountBase: (unset)          operations: (unset)
sourceHostUser: source_ASE_servername_example/sybase (*)
stagingHostUser: staging_ASE_servername_example/sybase (*)
stagingPostScript: (unset)          stagingPreScript: (unset)
stagingRepository: staging_ASE_servername_example (*)
validatedSyncMode: ENABLED  delphix database "dexample" attachSource *> commit
ASE_DB_CONTAINER-3      Dispatched job JOB-25      DB_ATTACH_SOURCE job started
for "Untitled/dexample".      DB_ATTACH_SOURCE job for "Untitled/dexample"
 completed successfully.
```

This command is only necessary if you are using a Remote Backup Server configuration from staging to the source, instead of an NFS mounted shared directory for backups and transaction log dumps:

```
delphix database "dexample" attachSource *> set
attachData.loadLocation.backupServerName=source_backupserver_name_example
```

CLI cookbook: linking an SAP ASE database loading from the last full backup of the source database

This topic describes how to use the command-line interface to link an SAP ASE database by loading from the most recent full backup of the source database.

Procedure

Enter the following commands in the Delphix Engine command-line interface:

```
ssh delphix_admin@delphixPassword: delphix> /database linkdelphix database link *>
set linkData.type=ASELinkDatadelphix database link *> set name=db2delphix database
link *> set group="ASE dSource"delphix database link *> set
linkData.config=db2delphix database link *> set linkData.dbUser=sadelphix database
link *> set linkData.dbCredentials.password=sybasedelphix database link *> set
linkData.loadBackupPath=/mnt/dumpdelphix database link *> set linkData.mountBase=/
mnt/provision/vdb_or_staging_database_namedelphix database link *> set
linkData.sourceHostUser=nstackrh69/sybasedelphix database link *> set
linkData.stagingHostUser=nealorarh75/sybasedelphix database link *> set
linkData.stagingRepository=ASE157SP138delphix database link *> set
linkData.sourcingPolicy.logsyncEnabled=truedelphix database link *> set
linkData.sourcingPolicy.type=SourcingPolicydelphix database link *> unset
linkData.syncParametersdelphix database link *> edit linkData.syncParametersdelphix
database link linkData.syncParameters *> backdelphix database link *> lsProperties
type: LinkParameters    name: db2 (*)    description: (unset)    group: ASE dSource
(*)    linkData:        type: ASELinkData        config: db2 (*)
dbCredentials:            type: PasswordCredential            password: ******** (*)
dbUser: sa (*)        dumpCredentials: (unset)        dumpHistoryFileEnabled: false
externalFilePath: (unset)        loadBackupPath: /mnt/dump (*)        loadLocation:
(unset)        mountBase: /mnt/provision/vdb_or_staging_database_name (*)
operations: (unset)        sourceHostUser: nstackrh69/sybase (*)
sourcingPolicy:            type: SourcingPolicy (*)            logsyncEnabled: true
 (*)        stagingHostUser: nealorarh75/sybase (*)        stagingOperations: (unset)
stagingPostScript: (unset)        stagingPreScript: (unset)        stagingRepository:
ASE157SP138 (*)        syncParameters:            type: ASELatestBackupSyncParameters
(*)        validatedSyncMode: ENABLEDOperationsdefaultsdelphix database link *>
commit    `ASE_DB_CONTAINER-209    Dispatched job JOB-3704    DB_LINK job started for
 "ASE dSource/db2".    DB_LINK job for "ASE dSource/db2" completed successfully.
```

mountBase option

The **mountBase** parameter is an option that was added in Delphix 5.2 and higher. By default, Delphix mounts the staging database under the Delphix toolkit directory. If a directory is specified for this parameter, the staging database's NFS devices will be mounted under this directory rather than under the toolkit directory. This can be especially helpful when the dSource is linked with LogSync enabled. SAP ASE has a limit of 127 characters for the fully qualified path to the transaction logs specified in the "LOAD TRANSACTION" statement. When LogSync is enabled, Delphix keeps a copy of the transaction logs on the engine under the dSources "archive" folder. The default naming convention for the folders under the toolkit can easily cause the 127 character limit to be exceeded so it is highly recommended to use this parameter when enabling LogSync. The mountBase is limited to 87 characters (the device names Delphix generates are 32 characters and the subdirectory containing the archive files is about 8 characters long). This leaves approximately 40 characters for the name of the transaction logs themselves. The mountBase

parameter must be unique for each VDB or staging database. The path can reside under a common parent directory for example, /mnt/provision but you must specify a unique child directory under the parent for each VDB or staging database using this optional parameter.xs

CLI cookbook: VDBs

This section covers the following topics:

CLI cookbook: attaching or detaching a PDB

PDB CLI attach

1. Ensure the new source environment has been fully discovered by the Delphix Engine, including CDB discovery.
2. Login to the engine via SSH with a Delphix admin account.
3. Select the PDB dSource database to be attached.

```
dvde.dcenter> database
dvde.dcenter database> select "R268PDB1"
dvde.dcenter database 'R268PDB1'>
```

4. Attach the PDB dSource to the correct source PDB in the new environment using the attachSource command; you will need to specify the environment user and login credentials for the PDBdSource.

```
dvde.dcenter database 'R268PDB1'> attachSource
dvde.dcenter database 'R268PDB1' attachSource *> set
attachData.type=OraclePDBAttachData
dvde.dcenter database 'R268PDB1' attachSource *> set attachData.config=R268PDB1
dvde.dcenter database 'R268PDB1' attachSource *> set
attachData.environmentUser=ora12201
dvde.dcenter database 'R268PDB1' attachSource *> set attachData.dbUser=delphix
dvde.dcenter database 'R268PDB1' attachSource *> set
attachData.dbCredentials.password
Enter attachData.dbCredentials.password: *******
dvde.dcenter database 'R268PDB1' attachSource *> commit
    R268PDB1
    Dispatched job JOB-46
    DB_ATTACH_SOURCE job started for "Untitled/R268PDB1".
    Obtaining information from source database "Untitled/R268PDB1".
    The dSource "R268PDB1" was successfully linked from source database
"Untitled/R268PDB1".
```

PDB CLI detach

1. Login to the engine via SSH with a Delphix admin account.
2. Select the PDB dSource database to be detached.

```
dvde.dcenter> database
dvde.dcenter database> select "R268PDB1"
dvde.dcenter database 'R268PDB1'>
```

3. Detach the PDB dSource using the detachSource command:

```
dvde.dcenter database 'R268PDB1'> detachSource
dvde.dcenter database 'R268PDB1' detachSource *> set source=R268PDB1
dvde.dcenter database 'R268PDB1' detachSource *> commit
```

```
Dispatched job JOB-45
DB_DETACH_SOURCE job started for "Untitled/R268PDB1".
DB_DETACH_SOURCE job for "Untitled/R268PDB1" completed successfully.
```

CLI cookbook: attaching, detaching, or linking a CDB

This topic describes how to attach, detach, or link a CDB using the command-line interface.

A CDB is not automatically detached when you detach the last PDB of the CDB. The CDB still remains in an inactive state which in turn prevents you from removing the environment. Therefore, you must detach the CDB in order to remove the environment.

> You can perform detach or attach operations only via Command-Line Interface (CLI).

CDB CLI detach

Perform the following steps to detach a CDB.

> You can detach a CDB only when there are no PDBs linked and the CDB does not have any vPDBs.

1. Login to the engine via SSH with a Delphix admin account.

```
$ ssh admin@YOUR_ENGINE
```

2. Select the CDB dSource database to be detached.

```
dvde.dcenter> cd database
dvde.dcenter database> select CDOMLOSR421F
dvde.dcenter database 'CDOMLOSR421F'>
```

3. Detach the CDB dSource using the detachSource command.

```
dvde.dcenter database 'CDOMLOSR421F'> detachSource
dvde.dcenter database 'CDOMLOSR421F' detachSource *> set source=CDOMLOSR421F
dvde.dcenter database 'CDOMLOSR421F' detachSource *> commit
    Dispatched job JOB-45
    DB_DETACH_SOURCE job started for "Untitled/CDOMLOSR421F".
    DB_DETACH_SOURCE job for "Untitled/CDOMLOSR421F" completed successfully.
```

You must attach the CDB back before performing the following operations: linking a new PDB, attaching a PDB, provisioning a vPDB, or attaching a converted PDB. Failure to do this will either result in an error due to duplicated object or will create a second CDB that will duplicate the space used for the CDB.

CDB CLI link

You can not perform an attach operation for PDBs that have already been detached from the primary CDB and are required to be attached to a standby CDB if the standby CDB is not already linked or attached. You must link the CDB manually before performing the PDB attach operation.

Perform the following steps to link a CDB.

1. Login to the engine via SSH with a Delphix admin account.

```
$ ssh admin@YOUR_ENGINE
```

2. Select the CDB dSource database to be linked.

```
dvde.dcenter> cd database
dvde.dcenter database> link
dvde.dcenter database link *>
```

3. Link the CDB dSource using the link command.

```
dvde.dcenter database link *> set name=mycdb
dvde.dcenter database link *> set group=Untitled
dvde.dcenter database link *> edit linkData
dvde.dcenter database link linkData *> set type=OracleLinkFromExternal
dvde.dcenter database link linkData *> set syncStrategy.config=CDOMSHSR6706
dvde.dcenter database link linkData *> set environmentUser=oracle
dvde.dcenter database link linkData *> set linkNow=true
dvde.dcenter database link linkData *> commit
    `ORACLE_DB_CONTAINER-23
    Dispatched job JOB-171
    \DB_LINK job started for "Untitled/mycdb".
    Obtaining information from source database "Untitled/mycdb".
    Creating new TimeFlow for dSource "Untitled/mycdb".
    Request to SnapSync container database "Untitled/mycdb" after attaching it
will be ignored.
    Action: Direct SnapSync of container database is not allowed. Container
              database SnapSync will be taken when SnapSync of corresponding
              pluggable database is taken.
    The dSource"mycdb" as successfully linked from source database "Untitled/
mycdb".
    DB_LINK job for "Untitled/mycdb" completed successfully.
```

CDB CLI Attach

Perform the following steps to attach a CDB.

1. Ensure the new source environment has been fully discovered by the Delphix Engine.
2. Login to the engine via SSH with a Delphix admin account.

```
$ ssh admin@YOUR_ENGINE
```

3. Select the CDB dSource database to be attached.

```
dvde.dcenter> cd database
dvde.dcenter database> select CDOMLOSR421F
dvde.dcenter database 'CDOMLOSR421F'>
```

4. Attach the CDB dSource to the correct source CDB in the new environment using the attachSource command.

When attaching a CDB linkNow=true command is ignored, then the CDBs can not be snapshotted directly. CDB snapshots will be taken when PDB requires a snapshot of the CDB.

```
dvde.dcenter database 'CDOMLOSR421F'> attachSource
dvde.dcenter database 'CDOMLOSR421F' attachSource *> edit attachData
dvde.dcenter database 'CDOMLOSR421F' attachSource attachData *> set
type=OracleAttachData
dvde.dcenter database 'CDOMLOSR421F' attachSource attachData *> set
config=CDOMLOSR421F
dvde.dcenter database 'CDOMLOSR421F' attachSource attachData *> set
environmentUser=oracle
dvde.dcenter database 'CDOMLOSR421F' attachSource *> commit
    CDOMLOSR421F
    Dispatched job JOB-46
    DB_ATTACH_SOURCE job started for "Untitled/CDOMLOSR421F".
        Starting validation of attach parameters for database "CDOMLOSR421F".
    Obtaining information from source database "Untitled/CDOMLOSR421F".
    The dSource "CDOMLOSR421F" was successfully linked from source database
"Untitled/CDOMLOSR421F".
```

CLI cookbook: changing SGA parameter

Below outlines the procedure to change SGA parameter setting on a provisioned VDB.

Procedure

1. Log into the Delphix Management application as `admin` or a user with Admin privileges.
2. Go to `source` and then `select` the name of the VDB that you would like to change the parameters of.
3. You are then going to `update` and edit the `configParams.`
4. Next, you are going to set `sga_target=` correct value.
5. `Commit` the operation so that it saves.

Example

```
ssh admin@enginedelphix > source
delphix source > select "vdb_example"
delphix source "vdb_example" > update
delphix source "vdb_example" *> edit configParams
delphix source "vdb_example" *> set sga_target=new value
delphix source "vdb_example" *> commit
```

ⓘ Modifying configuration parameters for a vPDB is not supported, so this workflow will not succeed for a vPDB.

CLI cookbook: changing the SID of Oracle RAC VDBs

eThis topic describes how to change the SID of instances in an Oracle RAC VDB.

This example demonstrates how to switch the instance name and number between two different hosts, from

```
SQL> select * FROM V$ACTIVE_INSTANCES;
INST_NUMBER INST_NAME
----------- ------------------------------------------------------------
      1 cnrac3:VchiBEB1
      2 cnrac4:VchiBEB2
```

to

```
SQL> select * FROM V$ACTIVE_INSTANCES;
INST_NUMBER INST_NAME
----------- ------------------------------------------------------------
      1 cnrac4:VchiBEB1
      2 cnrac3:VchiBEB2
```

Procedure

1. Stop the VDB through the GUI and login to the Delphix CLI.
2. Select the sourceconfig of the RAC VDB whose instances you would like to rename.

   ```
   kfc-manual.dcenter> sourceconfig
   kfc-manual.dcenter sourceconfig> select Vchicago_BEB
   ```

3. Use the update command to change the properties of the sourceconfig.

   ```
   kfc-manual.dcenter sourceconfig "Vchicago_BEB"> update
   ```

4. Type 'ls' to view the complete list of properties associated with the VDB's sourceconfig. For configurations with larger numbers of RAC instances, the listing may not show the individual instances but will instead display [...]. In order to see the instance configuration, type 'edit instances'.

   ```
   kfc-manual.dcenter sourceconfig "Vchicago_BEB" update *> ls
   Properties
       type: OracleRACConfig
       credentials:
           type: PasswordCredential
           password: ********
       environmentUser: ora1024
       instances:
           0:
               type: OracleRACInstance
               instanceName: VchiBEB1
               instanceNumber: 1
   ```

```
            node: cnrac4
        1:
            type: OracleRACInstance
            instanceName: VchiBEB2
            instanceNumber: 2
            node: cnrac3
    linkingEnabled: true
    nonSysCredentials: (unset)
    nonSysUser: (unset)
    repository: '/u01/app/ora1024/product/10.2.0/db_1'
    services: [ ... ]
    user: delphix
```

5. Use the Set command to change the values for instanceName and instanceNumber for each instance.

```
kfc-manual.dcenter sourceconfig "Vchicago_BEB" update *> set instances.0.instan
ceName=VchiBEB2
kfc-manual.dcenter sourceconfig "Vchicago_BEB" update *> set instances.0.instan
ceNumber=2
kfc-manual.dcenter sourceconfig "Vchicago_BEB" update *> set instances.1.instan
ceName=VchiBEB1
kfc-manual.dcenter sourceconfig "Vchicago_BEB" update *> set instances.1.instan
ceNumber=1
```

6. Finally, commit the changes.

```
kfc-manual.dcenter sourceconfig "Vchicago_BEB" update *> commit;
```

7. Restart the VDB through the GUI for the changes to take effect on the VDB.

CLI cookbook: toggle new DBID generation upon refresh options for Oracle VDBs

This topic describes how to toggle **Generate new DBID upon refresh** option for Oracle VDBs.

Procedure

1. Login to the Delphix CLI.
2. Go to source.

```
delphix> source
```

3. Select the VDB that you need to update

```
delphix> source select 'VDBOMSRE71EE4_QGE'
```

4. Toggle new DBID value. The new DBID value can be true or false. If the `newDBID` parameter is set to false then the new DBID generation will stop.

```
delphix source VDBOMSRE71EE4_QGE> update
delphix source VDBOMSRE71EE4_QGE> set newDBID=<new value>
```

5. Finally, commit the changes to save the changes.

```
delphix source VDBOMSRE71EE4_QGE> commit
```

6. Refresh the VDB through CLI to generate a new DBID if the `newDBID` parameter was set to true. If the `newDBID` parameter was set to false, the Delphix engine will use the same DBID as the parent.

```
delphix> /database
delphix database>  select 'VDBO_QGE'
delphix database 'VDBO_QGE'> refresh
delphix database 'VDBO_QGE' refresh> commit
```

CLI cookbook: creating a policy

This will outline how to create a policy in the CLI, please note that you can also do this in the GUI.

Procedure

1. ssh into your Delphix Engine using admin credentials.

```
ssh admin@delphixengine
delphix > ls
```

2. Go to `policies` and `createAndApply` (please note that you cannot just create a policy, you must `createAndApply`, in the GUI you have the option to just create) and set your policy parameter.

```
delphix > policy
delphix policy > createAndApply
delphix policy createAndApply *> set policy.type=< choose from QuotaPolicy,
RefreshPolicy, RetentionPolicy, SnapshotPolicy or SyncPolicy)
delphix policy createAndApply *> set policy.name=< name your policy>
delphix policy createAndApply *> set policy.customized=true
delphix policy createAndApply *> set policy.
delphix policy createAndApply *> set policy.provisionSource=(LATEST_SNAPSHOT or
LATEST_TIME_FLOW_LOG)
```

Info:
If doing a `RefreshPolicy, SyncPolicy` or `SnapshotPolicy` you are also going to need to add the following:

1.
```
delphix policy createAndApply *> edit policy.scheduleList
delphix policy createAndApply policy.scheduleList * > add
delphix policy createAndApply policy.scheduleList * > set cronString=
delphix policy createAndApply policy.scheduleList * > set cutoffTime=
delphix policy createAndApply policy.scheduleList * > back
```

3. Set your target parameters which are going to be a container, group etc.

```
delphix policy createAndApply *> set target=
```

4. Verify and `commit.`

```
delphix policy createAndApply *> ls
delphix policy createAndApply *> commit
```

CLI cookbook: creating a VDB config template

This topic will address how to create a VDB Config Template in the CLI; this functionality is also available in the GUI.

Procedure

1. ssh into your Delphix Engine using admin credentials.

```
ssh admin@<yourdelphixengine>
```

2. Go to `database` and then `template` and then `create.`

```
delphix > database template
delphix database template > create
delphix database template create *> set name=<set the name of database
template>
delphix database template create *> set parameters.<set parameters you want>
delphix database template create *> set sourceType=<set the source>
```

3. Verify information and commit.

```
delphix database template create *> ls
delphix database template create *> commit
```

Example for Oracle:

```
ssh admin@testenginetestengine > database templatetestengine
database template > createtestengine
database template create *> set name=oracletestengine
database template create *> set parameters.PGA_TARGET=100Mtestengine
database template create *> set parameters.MEMORY_TARGET=100Mtestengine
database template create *> set sourceType=OracleVirtualSourcetestengine
database template create *> lsProperties
  type: DatabaseTemplate
  name: oracle (*)
  description: (unset)
  parameters:
    MEMORY_TARGET: 100M (*)
    PGA_TARGET: 100M (*)
  sourceType: OracleVirtualSource (*)testengine
database template create *> commit
```

Example for SQL Server:

```
ssh admin@testengine testengine > database template
 testengine database template > create
testengine database template create *> set name=mssqlTemplate
testengine database template create *> set parameters.READ_COMMITTED_SNAPSHOT=ON
testengine database template create *> set sourceType=MSSqlVirtualSource
```

```
testengine database template create *> ls
Properties  type: DatabaseTemplate  name: mssqlTemplate (*)     description: (unset)

    parameters:
         READ_COMMITTED_SNAPSHOT: ON (*)
    sourceType: MSSqlVirtualSource (*)
testengine database template create *> commit
```

CLI cookbook: determining the snapshot used to provision a VDB

Procedure:

The parent snapshot can be determined using the CLI as follows:

1. Log into the server as a Engine Administrator:

```
ssh admin@<server_ip>
```

2. Select the VDB.

```
delphix> databased
elphix database> ls
Objects
NAME          PARENTCONTAINER DESCRIPTION
dSource1      -
dSource2      -
VDB1          dSource1        -
VDB2          dSource2        -
VDB3          dSource1        -
delphix database> select VDB1
```

3. List the VDB parameters, and make a note of the currentTimeflow value.

```
delphix database "VDB1"> ls
Properties
    type: OracleDatabaseContainer
    name: VDB1
    currentTimeflow: VDB1/default
    description: (unset)
    diagnoseNoLoggingFaults: true
    endianness: BIG_ENDIAN
    group: <New Group>
    masked: false
    os: HP-UX
    parentContainer: dSource1
    performanceMode: false
    processor: ia64
    reference: ORACLE_DB_CONTAINER-10
    runtime:
        type: OracleDBContainerRuntime
        logSyncActive: true
    sourcingPolicy:
        type: OracleSourcingPolicy
        encryptedLinkingEnabled: false
        logsyncEnabled: true
        logsyncInterval: 300
        logsyncMode: ARCHIVE_ONLY_MODE
version:
```

4. Select the Timeflow listed for the VDB.

```
delphix database "VDB1"> /timeflow
delphix timeflow> select VDB1/default
List the timeflow parameters.  The Snapshot used to provision the VDB is listed
as parentSnapshot

delphix timeflow "VDB1/default"> ls
Properties
```

5. List the Timeflow parameters. The Snapshot used to provision the VDB is listed as parentSnapshot.

```
delphix timeflow "VDB1/default"> ls
Properties
type: OracleTimeflow
    name: VDB1/default
    container: VDB1
    parentPoint:
        type: OracleTimeflowPoint
        location: 141285148
        timeflow: dSource1/default
    parentSnapshot: @2013-02-14T15:07:28.491Z
    reference: ORACLE_TIMEFLOW-92572
```

CLI cookbook: how to refresh a VDB from a specific snapshot

These steps will allow you to refresh a VDB from any specific snapshot, not just the most recent one.

1. Identify the VDB and snapshot that you want to use.

```
ssh admin@<yourengine>
delphix > database ls
delphix database > /snapshot
delphix snapshot > list database=<SOURCEOFSNAPSHOT>
```

2. Go to the `database` and `refresh.`

```
delphix > /database
delphix database > refresh
```

3. Now set what type of refresh you are going to do.

```
delphix database 'VDB' refresh *> set
timeflowPointParameters.type=TimeflowPointSnapshot
```

4. Set the snapshot

```
delphix database 'VDB' refresh *> set timeflowPointParameters.snapshot=@XXXX-
XX-XXTXX:XX:XX.XXXZ
```

5. Commit the action.

```
delphix database 'VDB' refresh *> commit
```

You can use tab to complete most actions in the CLI in addition to listing the possibilities that are available when setting parameters.

CLI cookbook: Oracle VDB migration

This topic describes moving a VDB from one environment or installation to another.

VDBs can be moved (or migrated) between hosts by changing the source repository associated with the VDB source config.

Restrictions

The following restrictions apply when migrating a VDB between repositories:

- When migrating a RAC VDB, the host of each `OracleRACInstance` must be updated as well.
- The mount point of the VDB source cannot be changed.
- The `database_unique_name` and `db_name` cannot be changed.
- The new environment and repository must be a compatible target environment.

Procedure

1. Select the source associated with the VDB. By default, sources are named the same as the VDB.

```
delphix> source "vexample"
```

2. Disable the source by running the `disable` command and committing the operation.

```
delphix source "vexample"> disable
delphix source "vexample" disable *> commit
    Dispatched job JOB-171
    SOURCE_DISABLE job started for "vexample".
    Starting disable of virtual database.
    Unexporting storage.
    Virtual database disable successful.
    SOURCE_DISABLE job for "vexample" completed successfully.
delphix source "vexample">
```

3. Select the source config associated with the source. By default this is also the same name as the VDB. Update the repository and repository user associated with the source config.

```
delphix source "vexample"> get config
    vexample
delphix source "vexample"> /sourceconfig "vexample"
delphix sourceconfig "vexample">
```

4. Update the repository and repository user associated with the source config.

Warning: You must use the Environment name of the Environment because the repository requires the environment name and Oracle home location. Enable the source.

```
delphix sourceconfig "vexample"> update
delphix sourceconfig "vexample" update *> set repository=192.168.100.247/'/opt/
oracle/product/10.2.0.4/db_1'
```

```
delphix sourceconfig "vexample" update *> set environmentUser=192.168.100.247/ora1024
delphix sourceconfig "vexample" update *> commit
delphix sourceconfig "vexample">
```

1. Enable the source.

```
delphix sourceconfig "vexample"> /source "vexample" enable
delphix source "vexample" enable *> commit
    Dispatched job JOB-18
    SOURCE_ENABLE job started for "vexample".
    Enabling dataset "vexample".
    Exporting storage containers from the Delphix Engine.
    Mounting datasets.
    Mounting filesystems for the virtual database instance "1".
    Starting virtual database.
    Starting instance 1 on virtual database "vexample".
    Virtual database "vexample" was successfully started.
    Dataset "vexample" enabled.
    SOURCE_ENABLE job for "vexample" completed successfully.
delphix sorceconfig "vexample">
```

CLI cookbook: provisioning a SAP ASE VDB

This topic describes how to provision an SAP ASE VDB using the command line interface.

Prerequisites

You will need the following information:

- The name of the VDB you want to create
- The group in which to create the VDB
- The SAP ASE database name for the VDB
- The source dSource or VDB from which you wish to provision
- The semanticLocation, LSN, or timestamp of the point you want to provision from (if not using the most recent). You can run these commands to get the list of snapshots or timeflow ranges:

```
snapshot list database=dexample
snapshot list timeflow=dexample
snapshot list fromDate="2020-03-01T00:00:00.000Z" toDate="2020-03-04T11:31:27.8
83Z"
```

- The target host on which you want to create the VDB. You can list the hosts with the /host list command.
- The source repository (SAP ASE instance on the target host) in which to create the VDB. These can be listed with the /repository list command.

Procedure

1. Execute the database provision command.

```
delphix> database provision
```

2. Set the type for the new VDB

```
delphix database provision *> set type=ASEProvisionParameters
```

3. Use defaults to fill in most of the information and then customize any additional information that you do not want defaulted, for what information has been filled in after defaults you can do an ls for all fields:

```
delphix database provision *> defaults
delphix database provision *> ls
Properties:
    type: TimeflowPointSemantic
    container: (unset)
    location: LATEST_POINT
delphix database provision *> set container=<dexample>
delphix database provision *> commit
```

4. Set the name and group for the new VDB.

```
delphix database provision *> set container.name=<vexample>
delphix database provision *> set container.group="<New Group>"
```

5. Set the name of the new VDB.

```
delphix database provision *> set sourceConfig.databaseName=<vexample>
```

6. Set the target Dataset Home.

```
delphix database provision *> set sourceConfig.repository=<Dataset Home>
delphix database provision *> set sourceConfig.environmentUser=<Host
environment name/sybase>
```

7. Set the source container from which to provision.

```
delphix database provision *> set timeflowPointParameters.container=<dexample>
```

8. Set the desired value for truncateLogOnCheckpoint

```
delphix database provision *> set truncateLogOnCheckpoint=false
```

9. Commit the configuration and start the DB_PROVISION job

```
delphix database provision *> commit
```

CLI cookbook: provisioning a single instance non-multitenant Oracle VDB

This topic describes how to provision a single instance non-multitenant Oracle VDB using the Delphix Engine command-line interface.

Prerequisites

You will need the following information:

- The name of the VDB you want to create
- The group in which to create the VDB
- The Oracle database name
- The Oracle database unique name
- The Oracle database instance number
- The Oracle database instance name
- The source dSource or VDB from which you wish to provision. This will be referenced as the "container" in the "defaults" command below.
- The semanticLocation, SCN, or timestamp of the point you want to provision from. You can run these commands to get the list of snapshots or Timeflow ranges:

```
snapshot list database=dexample
timeflow "dexample" timeflowRanges; commit
```

- The base mount point on the target server where VDB data should be mounted
- The source repository (oracle install) in which to create the VDB. These can be listed with the `/repository list` command.
- If you are using a VDB template, the name of the template to use.

Procedure

1. Execute the `database provision` command.

```
delphix> database provision
```

2. Execute the `defaults` command. Once you commit this command, it will return a partially constructed provision parameters object.

```
delphix database provision> defaults
```

3. Set the Timeflow point source Timeflow and location.

```
delphix database provision defaults *> set type=TimeflowPointSemantic
delphix database provision defaults *> set container=dexample
delphix database provision defaults *> set location=LATEST_SNAPSHOT
```

4. Commit the operation to populate the defaults, as provided by the browser interface. At this point, the operation can be committed, though you will likely need to change the defaults to match the information.

```
delphix database provision defaults *> commit
```

5. Set the name and group for the new VDB

```
delphix database provision *> set container.name=vexample
delphix database provision *> set container.group=""
```

6. Set the base mount point.

```
delphix database provision *> set source.mountBase=/mnt
```

7. Set the source config type to be a single instance non-multitenant Oracle, and set the database name and database unique name. When provisioning from a RAC or single instance non-multitenant oracle source, the default type will match that of the repository selected by the defaults operation.

```
delphix database provision *> set sourceConfig.type=OracleSIConfig
delphix database provision *> set sourceConfig.databaseName=vexample
delphix database provision *> set sourceConfig.uniqueName=vexample123
```

8. Set the instance name and number.

```
delphix database provision *> edit sourceConfig.instance
delphix database provision sourceConfig.instance *> set instanceNumber=1
delphix database provision sourceConfig.instance *> set instanceName=vexample
delphix database provision sourceConfig.instance *> back
```

9. Set the target repository.

```
delphix database provision *> set sourceConfig.repository='/env/opt/oracle'
```

10. Configure the Oracle database parameters. If you are using manually specified parameters, you can set the contents of `source.configParams`. If you want to use a template, you can set `source.configTemplate`.
11. (Optional) Configure customer environment variables.
Setting Environment Pair Values

```
delphix database provision *> edit customEnvVars
delphix database provision source.customEnvVars *> add
delphix database provision source.customEnvVars 0 *> set
type=OracleCustomEnvVarSIPair
delphix database provision source.customEnvVars 0 *> set varName=MYVAR1
delphix database provision source.customEnvVars 0 *> set varValue=Value1
delphix database provision source.customEnvVars 0 *> back
delphix database provision source.customEnvVars 1 *> set
type=OracleCustomEnvVarSIPair
delphix database provision source.customEnvVars 1 *> set varName=MYVAR2
```

```
delphix database provision source.customEnvVars 1 *> set varValue=Value2
delphix database provision source.customEnvVars 1 *> back
```

Note:

1.

OracleCustomEnvVarSIFile file should be in the following format.

- ```
  export VARNAME1=VARVALUE1export VARNAME2=VARVALUE2
  ```

2. When provisioning to a RAC target, use the type as **OracleCustomEnvVarRACPair or OracleCustomEnvVarRACFile.**

- ```
  delphix database provision source.customEnvVars 0 *> set
  type=OracleCustomEnvVarRACPairdelphix database provision
  source.customEnvVars 0 *> set clusterNode=targetnode1delphix database
  provision source.customEnvVars 0 *> set varName=MYVAR1delphix database
  provision source.customEnvVars 0 *> set varValue=Value1delphix database
  provision source.customEnvVars 0 *> back
  ```

12. Commit the result.

```
delphix database provision *> commit
```

CLI cookbook: provisioning a SQL server VDB

This topic describes how to provision a SQL Server VDB using the command line interface.

Prerequisites

You will need the following information:

- The name of the VDB you want to create
- The group in which to create the VDB
- The SQL Server database name for the VDB
- The source dSource or VDB from which you wish to provision
- The semanticLocation, LSN, or timestamp of the point you want to provision from. You can run these commands to get the list of snapshots or timeflow ranges:

```
snapshot list database=dexample
snapshot list timeflow=dexample
snapshot list fromDate="2020-03-01T00:00:00.000Z" toDate="2020-03-04T11:31:27.8
83Z"
```

- The target host on which you want to create the VDB. You can list the hosts with the `/host list` command.
- The source repository (SQL Server instance on the target host) in which to create the VDB. These can be listed with the `/repository list` command.

Procedure

1. Execute the `database provision` command.

```
delphix> database provision
```

2. Execute the `defaults` command.

```
delphix database provision> defaults
```

3. Set the timeflow point source timeflow and location.

```
delphix database provision defaults *> set type=TimeflowPointSemantic
delphix database provision defaults *> set container=dexample
delphix database provision defaults *> set location=LATEST_SNAPSHOT
```

4. Commit the operation to populate the defaults, as provided by the browser interface. At this point, the operation can be committed, though you will likely need to change the defaults to match the information.

```
delphix database provision defaults *> commit
```

5. Set the name and group for the new VDB.

```
delphix database provision *> set container.name=vexample
delphix database provision *> set container.group=""
```

6. Set the database name for the VDB on the target SQL Server instance.

```
delphix database provision *> set sourceConfig.databaseName=vexample
```

7. Set the target repository

```
delphix database provision *> set sourceConfig.repository=targetEnv/
SQLServer2008
```

8. Commit the result.

```
delphix database provision *> commit
```

CLI cookbook: provisioning a VDB from a timeflow bookmark

This topic describes how to create a Timeflow bookmark and use it to provision a single instance Oracle VDB using the Delphix Engine command-line interface.

You can create Timeflow bookmarks to give a semantically meaningful name to a TimeFlow point (scn, location or timestamp within a Timeflow). You can then use the bookmarks you created to execute the following database operations:

- Provision
- Refresh
- Export
- Test file mappings
- VDB Rewind

Prerequisites

You will need the following information:

- The name of the Timeflow bookmark you want to create
- The name of the VDB you want to create
- The group in which to create the VDB
- The Oracle database name
- The Oracle database unique name
- The Oracle database instance number
- The Oracle database instance name
- The source dSource or VDB from which you wish to provision
- The SCN, or timestamp of the point you want to provision from. You can run these commands to get the list of snapshots or Timeflow ranges:

```
snapshot list database=dexample
timeflow "dexample" timeflowRanges; commit
```

- The base mountpoint on the target server where VDB data should be mounted
- The source repository (oracle install) in which to create the VDB. These can be listed with the `/repository list` command.

Creating the timeflow bookmark

1. Execute the `timeflow bookmark create` command.

```
delphix> timeflow bookmark create
```

2. Set the timeflow point to be Oracle timeflow point.

```
delphix timeflow bookmark create *> set timeflowPoint.type=OracleTimeflowPoint
```

3. Set the timeflow point timeflow and location

```
delphix timeflow bookmark create *> set timeflowPoint.timeflow=dexample/default
```

```
delphix timeflow bookmark create *> set timeflowPoint.location=1945519455791
```

4. Set the name of the timeflow bookmark

```
delphix timeflow bookmark create *> set name=myTimeFlowBookmark
```

5. Commit the result

```
delphix timeflow bookmark create *> commit
    TIMEFLOW_BOOKMARK-1
```

6. Display the list of timeflow bookmarks

```
delphix> timeflow bookmark ls
Objects
NAME                TAG   TIMEFLOW
myTimeFlowBookmark  -     dexample/default
Operations
create
```

Provisioning from a TimeFlow bookmark

1. Execute the `database provision` command.

```
delphix> database provision
```

2. Set `defaults` and provide container (VDB or dSource) that you will be provisioning from

```
delphix database provision > defaults
delphix database provision defaults > set container=<VDB or dSource>
delphix database provision defaults > commit
```

3. Set the timeflowPointParameters type to be TimeflowBookmark.

```
delphix database provision *> set
timeflowPointParameters.type=TimeflowPointBookmark
```

4. Set the timeflow bookmark.

```
database provision *> set timeflowPointParameters.bookmark=myTimeFlowBookmark
```

5. Set the name and group for the new VDB.

```
delphix database provision *> set container.name=vexample
delphix database provision *> set container.group="Untitled"
```

6. Set the base mountpoint

```
delphix database provision *> set source.mountBase=/mnt
```

7. Set the source config type to be single instance Oracle, and set the database name and database unique name.

```
delphix database provision *> set sourceConfig.type=OracleSIConfig
delphix database provision *> set sourceConfig.databaseName=vexample
delphix database provision *> set sourceConfig.uniqueName=vexample123
```

8. Set the instance name and number.

```
delphix database provision *> edit sourceConfig.instance
delphix database provision sourceConfig.instance *> set instanceNumber=1
delphix database provision sourceConfig.instance *> set instanceName=vexample
delphix database provision sourceConfig.instance *> back
```

9. Set the target repository.

```
delphix database provision *> set sourceConfig.repository=env/'/opt/oracle'
```

10. Commit the result.

```
delphix database provision *> commit
```

CLI cookbook: provisioning a virtual PDB to a new virtual CDB

This topic describes how to provision a virtual pluggable database (vPDB) to a virtual container database (vCDB) using the command-line interface.

> ⚠ This process applies to Oracle container databases (CDBs) and the pluggable databases (PDBs) found within them that have been linked to Delphix as dSources. As container databases can only be found in Oracle 12.1.0.1 releases and above of Oracle this topic is relevant to only these releases.

Prerequisites

The provisioning of virtual PDBs in Delphix is dependent on the following entities being in place prior to attempting the creation of this type of Delphix dataset.

- A source container database must be linked to the engine.
- The source container database must have pluggable databases within it linked as dSources to Delphix.
- A destination host with an Oracle home of the same release as the source database must be available and already discovered as an Environment in Delphix.

In the example CLI provision detailed below the following databases are in place:

- The source container database is called "CDOMLOSR1TB"
- The pluggable database found in "CDOMLOSR1TB" that has been linked as a dSource is called "CDOMLOSR1TBPDB1"
- The destination virtual container database will be called "cdbvirt"
- Within the Delphix Engine CLI the name of the repository (target Oracle home) on the target system is '/u01/app/oracle/product/12.2.0.1/dbhome_1'
- The destination virtual pluggable database will be called "PDBS3" and run from virtual container database "cdbvirt"
- Delphix creates a temporary container database on the target host which will be used to establish the virtualized copy of the dSource PDB "pdborcl". This temporary CDB will be created and running during the provisioning process, it will be destroyed at the end of the provision of the virtual PDB.

Procedure

1. Log into the Delphix command-line interface using the admin user or a user with admin privileges.

```
$ ssh admin@YOUR_ENGINE
```

2. Move to the database provisioning command line object.

```
delphix> database provision
```

3. Give the dataset a name.

```
delphix database provision *> set container.name=PDBS3
```

4. Place the new dataset in a Group that appears in the Delphix GUI, in this case, the Targets group.

```
delphix database provision *> set container.group=Targets
```

5. Set the type of provision to perform, for Oracle virtual database (VDB) - VDB/vPDBs, the type will be OracleVirtualPdbSource.

```
delphix database provision *> setsource.type=OracleVirtualPdbSource
```

6. Set the destination mount point which Delphix NFS mounts are to be linked to under the virtual PDB. This folder must exist at a file system level on the target host. Do not use single quotes around the mount path.

```
delphix database provision *> set source.mountBase="/mnt/provision"
```

7. Name the vPDB. This is what it will appear as in the destination container database.

```
delphix database provision *> set sourceConfig.databaseName=PDBS3
```

8. Supply the dSource PDBs details. In this example, the provision will use the latest point in time available to the dSource PDB as the point in time from which to provision the vPDB. Setting a different timeflowPointParameters.type would allow you to use points in time other than the latest snapshot or latest point in time if this is what you desire. Using other types is not covered in this example

```
delphix database provision *> set
timeflowPointParameters.container=CDOMLOSR1TBPDB1
```

9. Give the virtual CDB a name:

```
delphix database provision *> set virtualCdb.container.name=cdbvirt
```

10. Place the virtual CDB in a Group that appears in the Delphix GUI, in this case, the target group.

```
delphix database provision *> set virtualCdb.container.group=Targets
```

11. If automatically restarting the vPDB and vCDB is not required after a reboot of the target host, set this to option to false.

```
delphix database provision *> set virtualCdb.source.
allowAutoVDBRestartOnHostReboot=false
```

12. Set the destination mount point which Delphix NFS mounts are to be linked to under the virtual CDB. This folder must exist at a file system level on the target host. Do not use single quotes around the mount path.

```
delphix database provision *> set virtualCdb.source.mountBase="/mnt/provision"
```

13. Set the vCDB configuration type to one of OracleRACConfig or OracleSIConfig. RAC configurations are not covered in this article:

```
delphix database provision *> set virtualCdb.sourceConfig.type=OracleSIConfig
```

14. Set the database name for the virtual CDB:

```
delphix database provision *> set virtualCdb.sourceConfig.databaseName=cdbvirt
```

15. Set the target system environment user that while be used to run the virtual CDB:

```
delphix database provision *> set
virtualCdb.sourceConfig.environmentUser=oracle
```

16. Set the instance name and number for the virtual CDB

```
delphix database provision *> set
virtualCdb.sourceConfig.instance.instanceName=cdbvirt
delphix database provision *> set
virtualCdb.sourceConfig.instance.instanceNumber=1
```

17. Set the repository (Oracle home) on the target system that will be used to run the virtual CDB. In this example the Oracle home is */u01/app/oracle/product/12.2.0.1/dbhome_1*:

```
delphix database provision *> set virtualCdb.sourceConfig.repository='/u01/app/
oracle/product/12.2.0.1/dbhome_1'
```

18. Set the database unique name for the virtual CDB:

```
delphix database provision *> set virtualCdb.sourceConfig.uniqueName=cdbvirt
```

19. Set the following two parameters to true or false, depending on whether the vCDB and vPDB should be restarted automatically following a target host reboot:

```
delphix database provision *> set source.allowAutoVDBRestartOnHostReboot=true
delphix database provision *> set virtualCdb.source.
allowAutoVDBRestartOnHostReboot=true
```

20. Check that all the settings you require are in place using the "ls" command

```
delphix database provision *> ls
Properties
    type: OracleMultitenantProvisionParameters
    container:
        type: OracleDatabaseContainer
        name: PDBS3 (*)
        description: (unset)
        diagnoseNoLoggingFaults: true
        group: Untitled (*)
        performanceMode: DISABLED
        preProvisioningEnabled: false
        sourcingPolicy: (unset)
    credential: (unset)
    masked: (unset)
```

```
maskingJob: (unset)
source:
    type: OracleVirtualPdbSource
    name: (unset)
    allowAutoVDBRestartOnHostReboot: true (*)
    config: (unset)
    customEnvVars: (unset)
    fileMappingRules: (unset)
    logCollectionEnabled: false
    mountBase: /mnt/provision (*)
    operations: (unset)
sourceConfig:
    type: OraclePDBConfig
    cdbConfig: (unset)
    databaseName: cdbvirt (*)
    environmentUser: (unset)
    linkingEnabled: true
    nonSysCredentials: (unset)
    nonSysUser: (unset)
    repository: (unset)
    services: (unset)
timeflowPointParameters:
    type: TimeflowPointSemantic
    container: CDOMLOSR1TBPDB1 (*)
    location: LATEST_POINT
username: (unset)
virtualCdb:
    type: OracleVirtualCdbProvisionParameters (*)
    container:
        type: OracleDatabaseContainer (*)
        name: cdbvirt (*)
        description: (unset)
        diagnoseNoLoggingFaults: true (*)
        group: Untitled (*)
        performanceMode: DISABLED (*)
        preProvisioningEnabled: false (*)
        sourcingPolicy: (unset)
    source:
        type: OracleVirtualCdbSource (*)
        name: (unset)
        allowAutoVDBRestartOnHostReboot: true (*)
        config: (unset)
        configParams: (unset)
        configTemplate: vcdb (*)
        logCollectionEnabled: false (*)
        mountBase: /mnt/provision (*)
    sourceConfig:
        type: OracleSIConfig (*)
        databaseName: cdbvirt (*)
        environmentUser: (unset)
        instance:
            type: OracleInstance (*)
```

```
                instanceName: cdbvirt (*)
                instanceNumber: 1 (*)
            linkingEnabled: true (*)
            nonSysCredentials: (unset)
            nonSysUser: (unset)
            repository: '/u01/app/oracle/product/12.2.0.1/dbhome_1' (*)
            services: (unset)
            tdeKeystorePassword: (unset)
            uniqueName: cdbvirt (*)

Operations
defaults
```

21. Initiate the provision by committing the operation in the CLI.

```
delphix database provision *> commit
    PDBS3
    Dispatched job JOB-24
    DB_PROVISION job started for "Targets/PDBS3".
    Starting provision of the virtual database "PDBS3".
    Preparing multitenant container database "cdbvirt".
    Creating new TimeFlow.
    Generating recovery scripts.
    Exporting storage.
    Mounting filesystems to recover pluggable database on instance "1".
    Mounting read-only archive log filesystem for the virtual database instance
"1".
    Backing up Oracle spfile.
    Mounting virtual database instance.
    Disabling flashback on Oracle database.
    Renaming Oracle datafiles.
    Mounting virtual database instance.
    Recovering Oracle pluggable database.
    Creating control file.
    Creating Oracle online logs.
    Processing startup init file.
    Configuring initialization and server parameter files.
    Mounting virtual database instance.
    Performing Oracle resetlogs.
    Creating tempfiles.
    Renaming readonly datafiles.
    Finalizing Oracle pluggable database.
    Registering listeners.
    Unmounting read-only archive log filesystem for the virtual database
instance "1".
    Unmounting filesystems after recovering pluggable database on instance "1".
    Plugging in Oracle pluggable database.
    Opening Oracle pluggable database.
    Setting OMF destination for Oracle pluggable database.
    Creating tempfiles.
    Online readonly tablespaces.
    Enabling Oracle instances.
```

```
    Checking Oracle pluggable database plugin violations.
    DB_PROVISION job for "Targets/PDBS3" completed successfully.
```

CLI cookbook: provisioning a virtual PDB in a target CDB

This topic describes how to provision a virtual pluggable database (vPDB) using the command-line interface.

> ⚠ This process applies to Oracle container databases (CDBs) and the pluggable databases (PDBs) found within them that have been linked to Delphix as dSources. As container databases can only be found in Oracle **12.1.0.1** releases and above of Oracle this topic is relevant to only these releases.

Prerequisites

Provisioning of virtual PDB's in Delphix is dependent on the following entities being in place prior to attempting the creation of this type of Delphix dataset.

- A source container database must be linked to the engine.
- The source container database must have pluggable databases within it linked as dSources to Delphix.
- A destination container database of the same Oracle release as the source database must exist in the target host. If this container database is a physical database, then it must be linked to Delphix. If it's a virtual container database, then it must be of Oracle version 12.1.0.2 or later.

In the example CLI provision detailed below the following databases are in place:

- The source container database is called "cdb12"
- The destination container database is called "cdbstage"
- The pluggable database found in "cdb12" that has been linked as a dSource is called "pdborcl"
- The destination virtual pluggable database will be called "pdbs2" and run from container database "cdbstage"

Delphix creates a temporary container database on the target host which will be used to establish the virtualized copy of the dSource PDB "pdborcl". This temporary CDB will be created and running during the provisioning process, it will be destroyed at the end of the provision of the virtual PDB.

Procedure

1. Log into the Delphix command-line interface using the admin user or a user with admin privileges.

```
$ ssh admin@YOUR_ENGINE
```

2. Set the type of provision to perform, for Oracle virtual database (VDB) - VDB/vPDBs, the type will be OracleVirtualSource.

```
set source.type=OracleVirtualPdbSource
```

3. Set the destination target environment/host through setting the sourceConfig environment user to perform the provision. A destination container database must already be running on this target host.

```
delphix database provision *> set sourceConfig.environmentUser=OEL6SIN1/delphix
```

4. Set the destination mount point which Delphix NFS mounts are to be linked to under the virtual PDB. This folder must exist at a file system level on the target host. Do not use single quotes around the mount path.

```
delphix database provision *> set source.mountBase="/mnt/provision"
```

5. Set the login details for the provision and Delphix OS user who is to perform the provision.

```
delphix database provision *> set username=delphix
delphix database provision *> set credential.type=PasswordCredential
delphix database provision *> set credential.password=delphix
```

6. Give the dataset a name.

```
delphix database provision *> set container.name=PDBS2
```

7. Place the new dataset in a Group that appears in the Delphix GUI, in this case, the target group.

```
delphix database provision *> set container.group=Targets
```

8. If automatically restarting the VDB is not required after a reboot of the VDB target host, set this to option to false. False is possibly a better option given the container database would need to be running prior to any attempt to pull up a vPDB.

```
delphix database provision *> set source.allowAutoVDBRestartOnHostReboot=false
```

9. Supply the destination container database name. This will be where the vPDB will ultimately be placed and run from on the target host.

```
delphix database provision *> set sourceConfig.cdbConfig=cdbstage
```

10. Name the vPDB. This is what it will appear as in the destination container database.

```
delphix database provision *> set sourceConfig.databaseName=pdbs2
```

11. Supply the dSource PDBs details. In this example, the provision will use the latest point in time available to the dSource PDB as the point in time from which to provision the vPDB. Setting a different timeflowPointParameters.type would allow you to use points in time other than the latest snapshot or latest point in time if this is what you desire. Using other types is not covered in this example.

```
delphix database provision *> set timeflowPointParameters.container=PDBORCL
```

12. Check that all the settings you require are in place using the "ls" command.

```
delphix database provision *> ls
Properties
  type: OracleProvisionParameters
  container:
    type: OracleDatabaseContainer
    name: PDBS2 (*)
    description: (unset)
    diagnoseNoLoggingFaults: true
    group: Targets (*)
```

```
    performanceMode: DISABLED
    preProvisioningEnabled: false
    sourcingPolicy: (unset)
  credential:
    type: PasswordCredential (*)
    password: ******** (*)
  maskingJob: (unset)
  newDBID: true (*)
  openResetlogs: true
  physicalStandby: false
  source:
    type: OracleVirtualSource (*)
    name: (unset)
    allowAutoVDBRestartOnHostReboot: false (*)
    archivelogMode: true
    config: (unset)
    configParams: (unset)
    configTemplate: (unset)
    customEnvVars: (unset)
    fileMappingRules: (unset)
    manualProvisioning: false
    mountBase: /mnt/provision (*)
    nodeListenerList: (unset)
    operations: (unset)
    redoLogGroups: 3
    redoLogSizeInMB: 0
  sourceConfig:
    type: OraclePDBConfig
    cdbConfig: cdbstage (*)
    databaseName: pdbs2 (*)
    environmentUser: (unset)
    linkingEnabled: true
    repository: (unset)
    services: (unset)
  timeflowPointParameters:
    type: TimeflowPointSemantic
    container: PDBORCL (*)
    location: LATEST_POINT
  username: delphix (*)
  Operationsdefaults
```

13. Initiate the provision by committing the operation in the CLI.

```
delphix database provision *> commit
  PDBS2
  Dispatched job JOB-333
  DB_PROVISION job started for "Targets/PDBS2".
  Starting provision of the virtual database "pdbs2".
  Preparing multitenant container database "cdbstage".
  Creating new TimeFlow.  Generating recovery scripts.
  Exporting storage.
  Mounting filesystems to recover pluggable database on instance "1".
```

```
  Mounting read-only archive log filesystem for the virtual database instance
"1".
  Recovering Oracle pluggable database.
  Mounting filesystems for the virtual database instance "1".
  Unmounting filesystems after recovering pluggable database on instance "1".
  Cleaning up objects created by pluggable database provisioning.
  Opening Oracle pluggable database.
  DB_PROVISION job for "Targets/PDBS2" completed successfully.
```

CLI cookbook: Provisioning a TDE-enabled virtual PDB to a new virtual CDB

This topic describes how to provision a TDE-enabled virtual pluggable database (vPDB) to a virtual container database (vCDB) using the command-line interface.

> This process applies to Oracle version 12.2.0.1 or later versions.

Prerequisites

The prerequisites are the same as described in CLI Cookbook: Provisioning a Virtual PDB to a new virtual CDB, additionally the following are the extra prerequisites:

- TDE must be configured for the source container database before it's linked to the engine.
- The source PDB must have TDE configured before it's linked as dSources to Delphix.
- The keystore file of the source container database must be accessible from the target host. If the target database is running in a RAC environment, the keystore file of the source container database must be accessible from all target nodes.
- If the target database is running in a RAC environment, **TDE Keystores Root** must be set for each node.

In the example CLI provision detailed below, assuming:

- The source container database **TDE Keystore Password** is `mypwd`.
- The source container database keystore file can be accessed from the target host(s) with path `/u01/app/oracle/keystores/CDOMLOSR1TB/wallet`.
- The new vCDB's keystore file will be created under the folder `/u01/app/oracle/keystores/cdbvirt/wallet`.
- The new vCDB's **TDE Keystore Password** is `mypwd`.

For more information about TDE parameters, please refer to Provisioning a TDE-enabled vPDB.

Procedure

After following all steps in the **Procedure** section of CLI Cookbook: Provisioning a Virtual PDB to a new virtual CDB to set provision parameters, set TDE-related parameters as follows before commit:

1. Set `parentTdeKeystorePath`, which is the path used to access the source CDB's TDE keystore file from target host(s).

```
delphix database provision *> set source.parentTdeKeystorePath=/u01/app/oracle/
keystores/cdb12/wallet
```

2. Set `parentTdeKeystorePassword`, which is the password of the source CDB's TDE keystore.

```
delphix database provision *> set source.parentTdeKeystorePassword=mypwd
```

3. Supply `tdeExportedKeyFileSecret`, which is the password used for exporting the vPDB's keys to keyfile.

```
delphix database provision *> set source.tdeExportedKeyFileSecret=mypwd
```

4. Set `targetVcdbTdeKeystorePath`, which is the folder where the new vCDB's TDE keystore file will be created.

```
delphix database provision *> set source.targetVcdbTdeKeystorePath=/u01/app/
oracle/keystores/cdbvirt/wallet
```

5. Set `targetVcdbTdeKeystorePassword`, which is the password of the new vCDB's TDE keystore.

```
delphix database provision *> set source.targetVcdbTdeKeystorePassword=mypwd
```

After all parameters are set, initiate the provision by committing the operation in the CLI:

```
delphix database provision *> commit
```

CLI cookbook: Provisioning a TDE-enabled vPDB in a target CDB

This topic describes how to provision a TDE-enabled virtual pluggable database (vPDB) in a target CDB (a linked CDB or existing vCDB) using the command-line interface.

> This process applies to Oracle version 12.2.0.1 or later versions.

Prerequisites

The prerequisites are the same as described in CLI Cookbook: Provisioning a Virtual PDB in a Target CDB, plus the following extra prerequisites:

- TDE must be configured for the source container database before it's linked to the engine.
- The source PDB must have TDE configured before it's linked as dSources to Delphix.
- The keystore file of the source container database must be accessible from the target host. If the target database is running in a RAC environment, the keystore file of the source container database must be accessible from all target nodes.
- **TDE Keystore Password** must be set for the target CDB.
- If the target database is running in a RAC environment, **TDE Keystores Root** must be set for each node.

In the example CLI provision detailed below, assuming:

- The source container database **TDE Keystore Password** is `mypwd`.
- The source container database keystore file can be accessed from the target host(s) with path `/u01/app/oracle/keystores/cdb12/wallet`.

For more information about TDE parameters, please refer to Provisioning a TDE-enabled vPDB.

Procedure

After following all steps in the **Procedure** section of CLI Cookbook: Provisioning a Virtual PDB in a Target CDB to set provision parameters, set TDE-related parameters as follows before the commit:

1. Set `parentTdeKeystorePath`, which is the path used to access the source CDB's TDE keystore file from the target host(s).

```
delphix database provision *> set source.parentTdeKeystorePath=/u01/app/oracle/
keystores/cdb12/wallet
```

2. Set `parentTdeKeystorePassword`, which is the password of the source CDB's TDE keystore.

```
delphix database provision *> set source.parentTdeKeystorePassword=mypwd
```

3. Supply `tdeExportedKeyFileSecret`, which is the password used for exporting the vPDB's keys to the keyfile.

```
delphix database provision *> set source.tdeExportedKeyFileSecret=mypwd
```

After all the parameters are set, initiate the provision by committing the operation in the CLI:

```
delphix database provision *> commit
```

CLI cookbook: refresh a VDB from a specific timepoint or latest

This topic describes the steps to Refresh a VDB from a specific Timepoint or from Latest.

You can refresh from any point on Timeflow using SCN, location, or timestamp.

Prerequisites

You will need the following information:

- The VDB name
- The TimeFlow location, SCN, or timestamp of the point you want to provision from.

The default domain user created on Delphix Engines is now **admin** instead of delphix_admin. When engines created before 5.3.1 are upgraded to 5.3.1 or later they will retain their old username 'delphix_admin'. To avoid complications Delphix recommends creating users with an admin role and then Disabling delphix_admin.

Log in to CLI:

```
$ ssh admin@<delphixengine>
```

Determine the timeflow

Run:

```
> timeflow "<dSource>" timeflowRanges
> commit
> cd
```

Perform the refresh from specific timepoint

```
> database
 > select <VDB name>
  > refresh
  > set timeflowPointParameters.type= (one of TimeflowPointBookmark,
TimeflowPointBookmarkTag, TimeflowPointLocation, TimeflowPointSemantic,
TimeflowPointTimestamp as appropriate)
  > set timeflowPointParameters.location= (the location, timestamp, or bookmark you
wish to refresh to)
  > set timeflowPointParameters.timeflow= (the timeflow associated with location)>
commit
```

Perform the refresh from latest

```
> database
> select <yourdatabase>
> refresh
> set timeflowPointParameters.container= (Parent of VDB)
> commit
```

CLI cookbook: repairing a timeflow

Prerequisites

- Know the dSource and Group you need to repair from
- Make sure that your retention policy is set correctly so that the ingested logs are within the retention

Procedure

1. Log into the Delphix Engine as an Admin user. Go to Timeflow and then list. Find the name of the Timeflow that needs to be repaired.

```
ssh admin@<yourengine>
delphix > timeflow
delphix timeflow > ls
```

2. List the missing logs for the Timeflow. The maximum number of logs reported is controlled by the value of the pageSize argument; if there are a very large number of missing logs, you may need to increase this value. Note the start and end scn of the missing log.

```
delphix timeflow > cd oracle/log
delphix timeflow oracle log> list timeflow=example missing=true pageSize=1000
```

Note: If the output from the above command does not list any logs as missing, but there are one or more sequences reported as missing by the **Delphix Admin** browser-based app, please contact Delphix Technical Support for assistance.

3. Stage the missing logs.
    a. Verify that there is sufficient free space.
    b. Copy or restore the missing archive logs into an alternative directory on a server the Delphix Engine can access via the network. All files and subdirectories in the configured directory are searched recursively, so staging the archive logs to a location with no other files or folders will ensure the process completes in a timely manner.
    c. Verify that the user being specified in the next step has permission to read these archive log files in the directory. The user credentials are optional if the host and user credentials have already been added to the Delphix Engine.

```
delphix timeflow oracle log  > fetch
delphix timeflow oracle log fetch *> set type=TimeflowLogFetchParameters
delphix timeflow oracle log fetch *> set timeflow=example
delphix timeflow oracle log fetch *> set directory=[directory where you
restored the log file]
delphix timeflow oracle log fetch *> set endLocation=[end SCN of the
sequence]
delphix timeflow oracle log fetch *> set startLocation=[start SCN of the
sequence]
```

```
delphix timeflow oracle log fetch *> set host=[hostname or IP of the host
you restored the file to]
delphix timeflow oracle log fetch *> set username=[a user that can read
the file]
delphix timeflow oracle log fetch *> edit credentials
delphix timeflow oracle log fetch *> set type=PasswordCredential
delphix timeflow oracle log fetch *> set password=[password for this user]
```

4. Commit the changes.

```
delphix timeflow oracle log  > fetch
delphix timeflow oracle log fetch *> set type=TimeflowLogFetchParameters
delphix timeflow oracle log fetch *> set timeflow=example
delphix timeflow oracle log fetch *> set directory=[directory where you
restored the log file]
delphix timeflow oracle log fetch *> set endLocation=[end SCN of the sequence]
delphix timeflow oracle log fetch *> set startLocation=[start SCN of the
sequence]
delphix timeflow oracle log fetch *> set host=[hostname or IP of the host you
restored the file to]
delphix timeflow oracle log fetch *> set username=[a user that can read the
file]
delphix timeflow oracle log fetch *> edit credentials
delphix timeflow oracle log fetch *> set type=PasswordCredential
delphix timeflow oracle log fetch *> set password=[password for this user]
```

Only do ONE repair job at a time.

It is possible for there to be more than one Timeflow visible for a given container/source. If that is the case, you can verify the current TimeFlow being used with:

```
delphix > database
delphix database > select 'example'
delphix database "example"> ls
```

Look for the `currentTimeflow` value.

CLI cookbook: rolling back a VDB

The following sections provide examples of how to rollback or rewind your VDB.

- Rolling back or rewinding to a snapshot from a VDB
- Rolling back or rewinding to a snapshot using timeflow
- Rolling back or rewinding to a timeflow bookmark

Rolling back or rewinding to a snapshot from a VDB

1. Log into the Delphix Engine.

```
ssh admin@delphix_engine
```

2. List Timeflows for the database that you want to rollback to.

```
de > ls
de > timeflow
de timeflow > ls
```

3. Switch to the VDB you want to rollback.

```
de timeflow > cd /database
de database > ls
de database > select "vdb_example"
```

4. Rollback VDB using the VDB rollback function (note this can also be done in the GUI).

```
de database 'vdb_example' > rollback
de database 'vdb_example' rollback *> set timeflowPointParameters.container=
de database 'vdb_example' rollback *> set timeflowPointParameters.location=
de database 'vdb_example' rollback *> commit
```

Rolling back or rewinding to a snapshot using timeflow

1. Log into the Delphix Engine.

```
ssh admin@delphix_engine
```

2. List Timeflows for the database that you want to rollback to.

```
de > ls
de > timeflow
de timeflow > ls
```

3. Switch to the VDB you want to rollback.

```
de timeflow > cd /database
de database > ls
de database > select "vdb_example"
```

4. Use the switchTimeflow operation.

```
de database 'vdb_example' > switchTimeflow
de database 'vdb_example' switchTimeflow *> set timeflow=<different timeflow>
de database 'vdb_example' switchTimeflow *> commit
```

Rolling back or rewinding to a timeflow bookmark

Requirements: Know the Timeflow bookmark that you want to use.

1. Log into the Delphix Engine.

```
ssh admin@<yourengine>
```

2. Retrieve database and Timeflow information that you would like to rewind/rollback to.

```
delphix > ls
delphix database > select "dexample"
delphix database "dexample" > get currentTimeflow
```

3. Rollback/Rewind VDB.

```
delphix database "dexample" > rollback
delphix database "dexample" rollback *> ls
delphix database "dexample" rollback *> set
timeflowPointParameters.type=TimeflowPointBookmark
delphix database "dexample" rollback *> set timeflowPointParameters.bookmark="b
ookmark example"
delphix database "dexample" rollback *> commit
```

CLI cookbook: rolling forward a VDB

This topic describes how to roll forward a virtual database after it has been rewound, as described in Provisioning and Managing Virtual Databases.

Once a VDB has rewound to a specific TimeFlow point, the snapshots of its previous states are still available in Delphix Engine storage and are accessed via the command-line interface to restore those previous states. This is referred to as "rolling forward" a VDB.

Procedure

1. Use the `ls` command to find the VDB you want to roll forward. In this example, the TimeFlows and their associated containers are listed. The VDB called `PVDB` will be the one to roll forward.

```
delphix timeflow> ls
ObjectsNAME                                CONTAINER  PARENTPOINT.
TIMEFLOW               PARENTPOINT.LOCATION  PARENTPOINT.TIMESTAMP
hrprod/default                        hrprod     -
-                     -
erpprod/default
                     erpprod    -                                    -
                    -
'DB_PROVISION@2013-11-25T17:37:06'  PVDB       erpprod/default
657925                -'DB_ROLLBACK@2013-11-25T18:24:16'   PVDB
'DB_PROVISION@2013-11-25T17:37:06'  678552
```

2. Use the `Select` command to select the database.

```
delphix database> select PVDB
```

3. Use the `rollback` command to roll forward the VDB.

```
delphix database "PVDB"> rollback
```

4. Use the `ls` command to display options for selecting TimeFlow parameters.

```
delphix database "PVDB" rollback *> ls
Properties
    type: OracleRollbackParameters
    credential: (unset)
    timeflowPointParameters:
        type: TimeflowPointSemantic
        container: (required)
        location: LATEST_POINT
    username: (unset)
```

5. Because this VDB was rolled back, two TimeFlows now exist for it. To rollback the VDB and roll it forward, select the original TimeFlow, because the original snapshots are associated with that TimeFlow.

```
delphix database "PVDB" rollback *> set
timeflowPointParameters.type=TimeflowPointLocation
delphix database "PVDB" rollback *> set timeflowPointParameters.timeflow='DB_PR
OVISION@2013-11-25T17:37:06'
```

6. Use the `ls` command to view the parameter options for the TimeFlow you selected.

```
delphix database "PVDB" rollback *> ls
Properties
    type: OracleRollbackParameters
    credential: (unset)
    timeflowPointParameters:
        type: TimeflowPointLocation (*)
        location: LATEST_POINT
        timeflow: 'DB_PROVISION@2013-11-25T17:37:06' (*)
    username: (unset)
```

7. Set the TimeFlow location to rollback the VDB to a particular Oracle SCN.

```
delphix database "PVDB" rollback *> set timeflowPointParameters.location=678994
```

8. Use the `ls` command to review all the options you selected before executing the commit.

```
delphix database "PVDB" rollback *> ls
Properties
    type: OracleRollbackParameters
    credential: (unset)
    timeflowPointParameters:
        type: TimeflowPointLocation (*)
        location: 678994 (*)
        timeflow: 'DB_PROVISION@2013-11-25T17:37:06' (*)
    username: (unset)
```

9. `Commit` the changes.

```
delphix database "PVDB" rollback *> commit
    Dispatched job JOB-369
    DB_ROLLBACK job started for "ERP/PVDB".
    Starting provision of the virtual database "PVDB".
    Creating new TimeFlow.
    Generating recovery scripts.
    Exporting storage.
    Validating user environment settings on target host.
    Mounting filesystems for the virtual database instance "1".
    Mounting read-only archive log filesystem for the virtual database instance
"1".
    Running user-specified pre-provisioning script.
```

```
    Recovering Oracle database.
    Opening the virtual database "PVDB".
    Opening Oracle database.
    Oracle recovery was successful.
    Unmounting read-only archive log filesystem for the virtual database
instance "1".
    Running user-specified post-provisioning script.
    The virtual database "PVDB" was successfully provisioned.
    Starting snapshot of virtual database.
    Processing database files of virtual database.
    Creating snapshot of virtual database.
    Finalizing snapshot of virtual database.
    Virtual database "PVDB" snapshot successful.
    DB_ROLLBACK job for "ERP/PVDB" completed successfully.
```

CLI Cookbook: taking a snapshot

This article is to document how to take a Snapshot outside of the normal snapshot policy time using the CLI, you can also do this in the GUI using the camera icon. A Snapshot of a VDB is similar to bookmarking a point in time in the life of the VDB.

Procedure:

1. ssh into the Delphix Engine using delphix_admin credentials.
2. Go into databases and select the VDB or dSource you would like to take a Snapshot of .

```
ssh admin@dengine
delphix > database
delphix database > select vdb_test
```

3. You are now going to sync and commit the operation .

```
delphix database "vdb_test" > sync
delphix database "vdb_test" sync *> commit
```

4. You can verify the snapshot by going to snapshots and listing them

```
delphix database "vdb_test" > /snapshot
delphix snapshot > ls
```

CLI cookbook: V2P: virtual to physical of a single instance non-multitenant Oracle database

This topic describes how to provision a physical single instance non-multitenant Oracle database using the Delphix Engine command-line interface.

Prerequisites

You will need the following information:

- The instance name, instance number, and unique name of the Oracle database you wish to create
- The source dSource or VDB from which you wish to provision.
- The semanticLocation, SCN, or timestamp of the point you want to provision from. You can run these commands to get the list of snapshots or Timeflow ranges:

```
snapshot list database=dexample
timeflow "dexample" timeflowRanges; commit
```

- The layout of the filesystems on the target server where data should be exported.
- The source repository (oracle install) in which to create the VDB. These can be listed with the `/repository list` command.

Procedure

1. Execute the `database export` command.

```
delphix> database export
```

2. Set the Timeflow point type, source container, and location.

```
delphix database export *> set
timeflowPointParameters.type=TimeflowPointSemantic
delphix database export *> set timeflowPointParameters.container=dexample
delphix database export *> set timeflowPointParameters.location=LATEST_SNAPSHOT
```

3. Edit the sourceConfig configuration, specifying the parameters for the database created via V2P.

```
delphix database export *> edit sourceConfig
delphix database export sourceConfig *> set type=OracleSIConfig
delphix database export sourceConfig *> edit instance
delphix database export sourceConfig instance *> ls
delphix database export sourceConfig instance *> set instanceName=v2p_db
delphix database export sourceConfig instance *> set instanceNumber=1
delphix database export sourceConfig instance *> back
delphix database export sourceConfig *> set repository=tserver/'/u01/app/
ora11204/product/11.2.0/dbhome_1'
 delphix database export sourceConfig *> set uniqueName=v2p_db
delphix database export sourceConfig *> set databaseName=v2p_db
delphix database export sourceConfig *> back
```

4. Set the destination locations.

```
delphix database export *> set filesystemLayout.targetDirectory=//u01/app/
oracle/oradata/v2p_db
delphix database export *> set filesystemLayout.archiveDirectory=archive
delphix database export *> set filesystemLayout.dataDirectory=datafiles
delphix database export *> set filesystemLayout.externalDirectory=external
delphix database export *> set filesystemLayout.scriptDirectory=script
delphix database export *> set filesystemLayout.tempDirectory=temp
```

5. Commit the configuration to execute the job.

```
delphix database export *> commit
```

CLI cookbook: V2P (virtual to physical) of a single instance Oracle database with datafiles on separate file systems

This topic describes how to provision a physical single-instance Oracle database, with datafiles on separate file systems, using the Delphix Engine command-line interface.

By default, all of the customizable directories - `data`, `archive`, `temp`, `external`, and `script` will be placed under the `target` directory, with additional config files placed in the `target` directory itself. This provides no mechanism to place datafiles on separate file systems (i.e. `/data1` and `/data2`, which have the root directory as a common point). In order to allow this, the `useAbsolutePathForDataFiles` parameter must be specified.

Prerequisites

You will need the following information:

- The instance name, instance number, and unique name of the Oracle database to create.
- The source dSource or VDB from which to provision.
- The semanticLocation, SCN, or timestamp of the point of which to provision from. Run these commands to get the list of snapshots or Timeflow ranges:

```
snapshot list database=dexample
timeflow "dexample" timeflowRanges; commit
```

- The layout of the filesystems on the target server where data should be exported.
- The source repository (oracle install) in which to create the VDB. These can be listed with the `/repository list` command.

Procedure

1. Execute the `database export` command.

```
delphix> database export
```

2. Set the Timeflow point type, source container, and location.

```
delphix database export *> set
timeflowPointParameters.type=TimeflowPointSemanticdelphix database export *>
set timeflowPointParameters.container=dexample
delphix database export *> set timeflowPointParameters.location=LATEST_SNAPSHOT
```

3. Edit the sourceConfig configuration, specifying the parameters for the database created via V2P.

```
delphix database export *> edit sourceConfig
delphix database export sourceConfig *> set type=OracleSIConfig
delphix database export sourceConfig *> edit instance
delphix database export sourceConfig instance *> ls
delphix database export sourceConfig instance *> set instanceName=v2p_db
delphix database export sourceConfig instance *> set instanceNumber=1
```

```
delphix database export sourceConfig instance *> back
delphix database export sourceConfig *> set repository=tserver/'/u01/app/
ora11204/product/11.2.0/dbhome_1'
delphix database export sourceConfig *> set uniqueName=v2p_db
delphix database export sourceConfig *> set databaseName=v2p_db
delphix database export sourceConfig *> back
```

4. Set the destination locations, including `setAbsolutePathForDataFiles` to be true.

```
delphix database export *> set filesystemLayout.targetDirectory=/v2p_db
delphix database export *> set filesystemLayout.archiveDirectory=archive
delphix database export *> set filesystemLayout.dataDirectory=datafiles
delphix database export *> set filesystemLayout.externalDirectory=external
delphix database export *> set filesystemLayout.scriptDirectory=script
delphix database export *> set filesystemLayout.tempDirectory=tempdelphix
database export *> set filesystemLayout.useAbsolutePathForDataFiles=true
```

5. Specify the file mapping rules for each datafile and the location to which it should be exported. Each entry should be separated by a newline, and the entire mapping within double quotes.

```
set fileMappingRules="/u01/app/oracle/oradata/datafile1.dbf:/data1/
datafile1.dbf
/u01/app/oracle/oradata/datafile2.dbf:/data1/datafile2.dbf
/u01/app/oracle/oradata/datafile3.dbf:/data2/datafile3.dbf
/u01/app/oracle/oradata/datafile4.dbf:/data2/datafile4.dbf"
```

6. Commit the configuration to execute the job.

```
delphix database export *> commit
```

CLI cookbook: export a non-multitenant virtual Oracle database to ASM

This topic describes how to perform an export or an in-place conversion of a non-multitenant virtual database to a physical database using the Delphix Engine command-line interface. This procedure will work irrespective of the VDB being a single instance configuration or RAC.

Prerequisites

You must have the following configuration before you start the export:

- Target ASM data diskgroup, or the diskgroup that will contain all the database files
- The VDB that needs to be exported to ASM.
- Optionally, the target ASM disk group for redo log files, database unique name for the resulting physical database and the number of RMAN channels.

⚠ Ensure that a database instance with the same unique name as the VDB is not running on the target host before starting the below procedure, otherwise the export operation will fail.

Procedure

1. Execute the `database export` command.

```
delphix> database export
```

2. Set the database export parameters type, transfer strategy type, storage strategy type, default target data diskgroup, and virtual source.

```
delphix database export *> set type=OracleDBExportParameters
delphix database export *> set storageStrategy.type=OracleExportASMStorageStrat
egy
delphix database export *> set
 storageStrategy.asmLayout.defaultDataDiskgroup=+DATA
delphix database export *> set transferStrategy.type=OracleExportDBInPlaceTrans
ferStrategy
delphix database export *> set transferStrategy.virtualSource=v2p_db
```

ⓘ **Info:** The above values are just examples. Replace the default data diskgroup and virtual source name to match your needs.

3. Optionally set the following parameters:
    a. Target ASM disk group for redo log files.
    b. Database unique name for the resulting physical database.
        i. If no database unique name is specified, the default is the unique name of the VDB.
        ii. If a database unique name is specified, as noted in step 2 of this procedure, you need to ensure that a database instance with the same database unique name as the VDB is not running on the target host.
    c. Number of RMAN channels. Default value for the RMAN channels is 8.
    d. RMAN file section size. Default value is 0 (i.e. RMAN file section size is not set).

```
delphix database export *> set
storageStrategy.asmLayout.redoDiskgroup=+REDO
delphix database export *> set transferStrategy.dbUniqueName=v2asm_db
delphix database export *> set transferStrategy.rmanChannels=10
delphix database export *> set transferStrategy.rmanFileSectionSizeInGb=64
```

Info: The above values are just examples. Replace the redo diskgroup, database unique name, number of RMAN channels and RMAN file section size to match your needs. For more details on RMAN channels and RMAN file section size, refer to the section titled "*Performance tuning considerations for Oracle databases with bigfile tablespaces*" on the Export an Oracle VDB or a vPDB to a Physical ASM or Exadata Database page.

4. Commit the configuration to execute the job.

```
delphix database export *> commit
```

The export operation takes a snapshot of the VDB before proceeding with the export. After a successful export operation, the original VDB will be left in a DISABLED state. If the VDB needs to be re-enabled, the VDB needs to be migrated to a different host. Ensure there is no instance running with the same name as the VDB on the target host. Subsequently, you must rewind the VDB to the latest snapshot, after which the VDB will be back up and running. Since the export operation takes a snapshot before the actual export, the rewind can be performed to the point just before the export started.

Refer to the Export an Oracle VDB or a vPDB to a Physical ASM or Exadata Database page for additional considerations after a successful export.

CLI cookbook: export a multitenant virtual pluggable Oracle database to ASM

This topic describes how to perform an export or an in-place conversion of a multitenant virtual pluggable database (vPDB) to a multitenant physical Oracle pluggable database (PDB) using the Delphix Engine Command-line Interface (CLI). This procedure will work irrespective of the vPDB being a single instance configuration or RAC.

Prerequisites

You must have the following configuration before you start the export:

- Target ASM data diskgroup, or the diskgroup that will contain all the database files
- The vPDB that needs to be exported to ASM.
- Optionally, the pluggable database name for the resulting physical pluggable database and the number of RMAN channels.

⚠ If the vPDB is being renamed during the export, confirm there is no other PDB with the same name in the target CDB before starting the below procedure. If a PDB with the same name exists in the target CDB, the export operation will fail before starting the actual movement of datafiles.

Procedure

1. Execute the `database export` command.

```
delphix> database export
```

2. Set the database export parameters type, transfer strategy type, storage strategy type, default target data diskgroup, and virtual source.

```
delphix database export *> set type=OraclePDBExportParameters
delphix database export *> set
storageStrategy.type=OracleExportASMStorageStrategy
delphix database export *> set
storageStrategy.asmLayout.defaultDataDiskgroup=+DATA
delphix database export *> set
transferStrategy.type=OracleExportPDBInPlaceTransferStrategy
delphix database export *> set transferStrategy.virtualSource=v2p_pdb
```

ⓘ **Info:** The above values are just examples. Replace the default data diskgroup and virtual source name to match your needs.

3. Optionally set the following parameters:
    a. pluggable database name for the resulting physical pluggable database.
        i. If a pluggable database name is specified, the target CDB should not have a PDB with the same name, otherwise the export will fail.
        ii. If no pluggable database name is specified, the default is the current name of the pluggable database.
    b. Number of RMAN channels. Default value for the RMAN channels is 8.
    c. RMAN file section size. Default value is 0 (i.e. RMAN file section size is not set).

```
delphix database export *> set transferStrategy.pdbName=v2asm_pdb
delphix database export *> set transferStrategy.rmanChannels=10
delphix database export *> set transferStrategy.rmanFileSectionSizeInGb=64
```

The above values are just examples. Please note that redoDiskgroup parameter is not required for PDB export. Replace the PDB name, number of RMAN channels and RMAN file section size to match your needs. For more details on RMAN channels and RMAN file section size, refer to the section titled "*Performance tuning considerations for Oracle databases with bigfile tablespaces*" on the Export an Oracle VDB or a vPDB to a Physical ASM or Exadata Database page.

4. Commit the configuration to execute the job.

```
delphix database export *> commit
```

The export operation takes a snapshot of the vPDB before proceeding with the export. After a successful export operation, the original vPDB will be left in a DISABLED state. If the vPDB needs to be re-enabled, the vPDB needs to be migrated to a different host and/or CDB. Ensure there is no PDB with the same name as the vPDB on the target CDB. Subsequently, you must rewind the vPDB to the latest snapshot, after which the vPDB will be back up and running. Since the export operation takes a snapshot before the actual export, the rewind can be performed to the point just before export started.

Refer to the Export an Oracle VDB or a vPDB to a Physical ASM or Exadata Database page for additional considerations after a successful export.

CLI Cookbook: V2P virtual to physical on SQL server

This topic describes how to export a physical single instance SQL Server database from a VDB using the Delphix Engine command-line interface.

Prerequisites

You will need the following information:

- The VDB name which you wish to export.
- The database name of the SQL Server database you wish to create
- The layout of the filesystems on the target server where data should be exported.
- The target repository in which to create the physical database. Repositories can be listed with the `/repository list` command.

Procedure

1. Execute the command `database export`.

```
delphix> database export
```

2. Specify the set type=MSSqlExportParameters

```
delphix database export *> set type=MSSqlExportParameters
```

3. Set the VDB you wish to export

```
delphix database export *> edit timeflowPointParameters
delphix database export timeflowPointParameters *> set container=vdb
delphix database export timeflowPointParameters *> back
```

4. Edit the sourceConfig configuration, specifying the parameters for the database created via V2P.

```
delphix database export *> edit sourceConfig
delphix database export sourceConfig *> set type=MSSqlSIConfig
delphix database export sourceConfig *> set databaseName=v2p_db
delphix database export sourceConfig *> set repository=win2012/SQL2016
delphix database export sourceConfig *> back
```

5. Set the destination locations.

```
delphix database export *> edit filesystemLayout
delphix database export filesystemLayout *> set targetDirectory=C:\temp
delphix database export filesystemLayout *> set archiveDirectory=archive
delphix database export filesystemLayout *> set dataDirectory=datafiles
delphix database export filesystemLayout *> set externalDirectory=external
delphix database export filesystemLayout *> set scriptDirectory=script
delphix database export filesystemLayout *> set tempDirectory=temp
```

6. Commit the configuration to execute the job.

```
delphix database export *> commit
```

CLI Cookbook: VDB status

It is possible to get a virtual database (VDB) status from the CLI.

1. Log into the CLI as any user that has privileges on the VDB.

```
ssh admin@yourengine
```

2. From source go to the VDB you want to get a status on.

```
delphix > sourcedelphix source > lsdelphix source > select <yourvdb>
```

3. Run the get runtime command to see all information or just get runtime.status for if the VDB is running.

```
delphix source 'vdb' > get runtime
```

or

```
delphix source 'vdb' > get runtime.status
```

4. If you would like to see more than one VDBs status you can also do the following:

```
delphix > sourcedelphix source > list display=name,runtime.status
```

CLI cookbook: provisioning a virtual PDB from a non-multitenant source database

Delphix supports provisioning a vPDB from a non-multitenant source database. This feature is only available through the API or command-line interface.

This topic describes how to provision a virtual pluggable database (vPDB) from a non-multitenant source database using the command-line interface.

> ⚠ **This feature has the following restrictions:**
> Transparent Data Encryption (TDE) is not supported.
> The provision point must correspond to a snapshot. Provisioning from a point in time between snapshots is not supported.
> The target CDB must be a physical CDB. Virtual CDB targets are not supported.

Prerequisites

Provisioning a vPDB from a non-multitenant source has the following environment requirements:

- Source host with a non-multitenant Oracle 11g or newer source database.
- VDB target host for provisioning a virtual non-multitenant VDB from the source database.
- CDB target host with a running Oracle target version CDB. The target CDB will be automatically linked if it is not already linked.

The target CDB can be a newer Oracle version than the source database (for example, the source is 12.2 and target is 19c). When an upgrade is also required, there are two options for upgrading the database:

- Upgrade Option 1: After provisioning the VDB. This option requires the ability to upgrade to the Oracle target version on the VDB target host.
- Upgrade Option 2: After plugging into the Linked CDB target database.

There are three scripts used during this procedure:

1. Pre-snapshot Hook on VDB: This hook will open the database in "read only" mode and issue a call to the `dbms_pdb.describe` procedure to generate an XML file describing the VDB.
2. **Post-snapshot Hook on VDB**: This hook will return the VDB to "read write" mode.
3. **Non-CDB to PDB Script**: This script will run as a hook present on the Linked CDB target host. This should call into the Oracle script `$ORACLE_HOME/rdbms/admin/noncdb_to_pdb.sql` to convert the VDB into a PDB. This script should also upgrade the vPDB if doing Upgrade Option 2.

Workflow

1. Link the non-multitenant Oracle 11g or newer source database as a dSource within Delphix.
2. Provision a non-multitenant Oracle VDB from the dSource onto the VDB target host. This will be referred to as the **Golden VDB**.
3. (If using Upgrade Option 1) Upgrade the **Golden VDB** to the Oracle target version: manually upgrade the database and point it to the new Oracle home. This step is only necessary if the source and target Oracle versions are not the same and the data files will not be upgraded when they are converted below.
4. Create a Pre-snapshot hook on the **Golden VDB** to open the database in read only mode and issue the `dbms_pdb.describe` procedure call to create an XML file called `delphix_plugin.xml.` The XML file will be used to plug the **Golden VDB** data files into the target CDB. The **Golden VDB** must be open read only during the subsequent snapshot so that the VDB data files do not require recovery when plugging them into the target CDB.
5. (Optional) Create a Post-snapshot hook on the **Golden VDB** to remove the database from read only mode. The **Golden VDB** can also remain read only.

6. Take a snapshot of the **Golden VDB**.
7. Create a PDB conversion script named `dx-post-plug-hook.sh` in the root of the Delphix toolkit directory of the Linked CDB target host. The name of the vPDB being provisioned/converted will be supplied by Delphix as the first parameter to the script when it invokes the script. The VDB data files will already be plugged into the Linked CDB target at the time the script is invoked. The script should do the following:
    a. (If using Upgrade Option 2) Upgrade the vPDB data files prior to the conversion.
    b. Call into `$ORACLE_HOME/rdbms/admin/noncdb_to_pdb.sql` and perform any customizations for the multitenant conversion.
8. Select a **snapshot** (point-in-time not supported) on the **Golden VDB** that has the `delphix_plugin.xml` file and provision a virtual PDB to the Linked CDB target. Virtual CDB targets are not supported.Note: This step can be executed via the **API / CLI only**, and will not be allowed via the Delphix UI.

CLI procedure to provision a vPDB from a VDB

1. Log into the Delphix command-line interface using the admin user or a user with admin privileges.

```
$ ssh admin@YOUR_ENGINE
```

2. Move to the database provisioning command line object.

```
delphix> database provision
```

3. Set the parameter type to `OracleMultitenantProvisionParameters`.

```
set type=OracleMultitenantProvisionParameters
```

4. Set the login details for the provision and Delphix OS user who is to perform the provision.

```
delphix database provision *> set username=delphix
delphix database provision *> set credential.type=PasswordCredential
delphix database provision *> set credential.password=delphix
```

5. Give the dataset a name.

```
delphix database provision *> set container.name=vpdb
```

6. Place the new dataset in a Group that appears in the Delphix GUI, in this case, the Targets group.

```
delphix database provision *> set container.group=Targets
```

7. Set the destination mount point which Delphix NFS mounts are to be linked to under the virtual PDB. This folder must exist at a file system level on the Linked CDB target host. Do not use single quotes around the mount path.

```
delphix database provision *> set source.mountBase="/mnt/provision"
```

8. If automatically restarting the vPDB is not required after a reboot of the Linked CDB target host, set this to option to false. False is possibly a better option given the container database would need to be running prior to any attempt to pull up a vPDB.

```
delphix database provision *> set source.allowAutoVDBRestartOnHostReboot=false
```

9. Supply the destination container database name. The container database should already be discovered. This will be where the vPDB will ultimately be placed.

```
delphix database provision *> set sourceConfig.cdbConfig=CDBSTAGE
```

10. Name the vPDB. This is what it will appear as in the destination container database.

```
delphix database provision *> set sourceConfig.databaseName=vpdb
```

11. Supply the source **Golden VDB** details. In this example, the provision will use the latest snapshot available from the **Golden VDB** as the point in time from which to provision the vPDB. A specific snapshot can also be picked, but an arbitrary point in time is not supported.

```
delphix database provision *> set
timeflowPointParameters.type=TimeflowPointSemantic
delphix database provision *> set timeflowPointParameters.container=gold_vdb
delphix database provision *> set
timeflowPointParameters.location=LATEST_SNAPSHOT
```

12. Check that all the settings you require are in place using the "ls" command.

```
delphix database provision *> ls
Properties
  type: OracleMultitenantProvisionParameters
  container:
    type: OracleDatabaseContainer
    name: vpdb (*)
    description: (unset)
    diagnoseNoLoggingFaults: true
    group: Targets (*)
    performanceMode: DISABLED
    preProvisioningEnabled: false
    sourcingPolicy: (unset)
  credential:
    type: Password Credential (*)
    password: ******** (*)
  masked: (unset)
  maskingJob: (unset)
  source:
    type: OracleVirtualPdbSource (*)
    name: (unset)
    allowAutoVDBRestartOnHostReboot: false (*)
    config: (unset)
```

```
    customEnvVars: (unset)
    fileMappingRules: (unset)
    LogCollectionEnabled: false
    mountBase: /mnt/provision (*)
    operations: (unset)
    parentTdeKeystorePassword: (unset)
    parentTdeKeystorePath: (unset)
    tdeExportedKeyFileSecret: (unset)
  sourceConfig:
    type: OraclePDBConfig
    cdbConfig: CDBSTAGE (*)
    databaseName: vpdb (*)
    environmentUser: (unset)
    linkingEnabled: true
    nonSysCredentials: (unset)
    nonSysUser: (unset)
    repository: (unset)
    services: (unset)
  timeflowPointParameters:
    type: TimeflowPointSemantic
    container: gold_vdb (*)
    location: LATEST_SNAPSHOT (*)
  username: delphix (*)
  VirtualCdb: (unset)  Operationsdefaults
```

13. Initiate the provision by committing the operation in the CLI.

```
delphix database provision *> commit
  vpdb
  Dispatched job JOB-333
  DB_PROVISION job started for "Targets/vpdb".
  Starting provision of virtual PDB database "vpdb" converted from a single
tenant database.
  Preparing multitenant container database "CDBSTAGE".
  Creating new TimeFlow.
  Generating recovery scripts.
  Exporting storage.
  Preparing XML manifest file prior to plugin.
  Plugging in Oracle pluggable database.
  Running user-defined post plug hook.
  Opening Oracle pluggable database.
  Setting OMF destination for Oracle pluggable database.
  Creating PDB tempfiles.
  Checking Oracle pluggable database plugin violations.
  DB_PROVISION job for "Targets/vpdb" completed successfully.
```

ⓘ To refresh the data in the vPDB from production, first, refresh the Golden VDB from the dSource, then refresh the vPDB from the new snapshot in the Golden VDB.

ⓘ There are some workflow customizations required for RAC databases:

1. The PDB conversion script must be in the root of the Delphix toolkit directory for all the target CDB RAC instances.
2. The **Golden VDB** Pre-Snapshot hook, as provided below, will not work in a clustered (RAC) environment with more than one active instance because it only shuts down the local instance. `dbms_pdb.describe` will not execute while an instance is open read-write. The workarounds are:
    a. Provision the **Golden VDB** as single-instance, either by provisioning to a non-RAC target or by provisioning to a RAC target with only one active instance. The sample hook will work in this case.
    b. Write a customized pre-snapshot hook that shuts down all instances, restarts only one instance in read-only mode, and runs `dbms_pdb.describe` .
    c. Manually perform the actions of the hook: shutdown the **Golden VDB**, restart one of the instances in read-only mode and then run `dbms_pdb.describe` .

Sample scripts

Golden VDB pre-snapshot hook

Restarts the source VDB in read only mode and runs `dbms_pdb.describe` to generate an XML file describing the VDB. The XML file will be used to plug the VDB into the Linked CDB target. The target for the XML file must be `$DELPHIX_MOUNT_PATH/$DELPHIX_DATABASE_UNIQUE_NAME/datafile/delphix_plugin.xml.`

```
#!/bin/sh

sqlplus "/ AS SYSDBA" <<-EOF
  whenever sqlerror exit 2;
  spool $DELPHIX_MOUNT_PATH/$DELPHIX_DATABASE_UNIQUE_NAME/datafile/presnapshot.
out replace
  shutdown immediate
  startup mount
  alter database open read only;
  exec dbms_pdb.describe(pdb_descr_file=>'$DELPHIX_MOUNT_PATH/
$DELPHIX_DATABASE_UNIQUE_NAME/datafile/delphix_plugin.xml');
  exit;
EOF
```

Golden VDB post-snapshot hook

This is only necessary if the VDB should not be left in read-only mode after the snapshot.

```
#!/bin/sh

sqlplus "/ AS SYSDBA" <<-EOF
  whenever sqlerror exit 2;
  spool $DELPHIX_MOUNT_PATH/$DELPHIX_DATABASE_UNIQUE_NAME/datafile/postsnapshot.out
replace
  shutdown immediate
```

```
  startup
  exit;
EOF
```

PDB conversion script

The script should be named `dx-post-plug-hook.sh` and reside in the root of the Delphix toolkit directory of the Linked CDB target host. Delphix will supply the name of the PDB being provisioned/converted as the first parameter.

The VDB datafiles will have already been plugged into the target CDB at the time the script is invoked and the virtual PDB will be in the mounted (not open) state. The PDB conversion script should return with the virtual PDB in either the mounted or open (not restricted) state. Delphix does not enforce a time-out for the script.

```
#!/bin/sh
DELPHIX_PDB_NAME=$1SCRIPT_DIR="$( cd "$( dirname "$0" )" && pwd )"
CONVERT_LOGFILE=$SCRIPT_DIR/$DELPHIX_PDB_NAME-pdbconvert.log

sqlplus "/ AS SYSDBA" <<-EOF
  whenever sqlerror exit 2;
  spool $CONVERT_LOGFILE replace
  alter session set container=$DELPHIX_PDB_NAME;
  @?/rdbms/admin/noncdb_to_pdb.sql
  exit;
EOF
```

PDB upgrade script

The following script will upgrade the vPDB. Use a wrapper that runs both this script and the prior conversion script (or combine the two in a single script) if doing both an upgrade and a conversion.

```
#!/bin/sh

DELPHIX_PDB_NAME=$1
SCRIPT_DIR="$( cd "$( dirname "$0" )" && pwd )"
UPGRADE_LOGFILE=$SCRIPT_DIR/$DELPHIX_PDB_NAME-dx-post-plug-upgrade.log
UPGRADE_LOGDIR=$SCRIPT_DIR/$DELPHIX_PDB_NAME-upgrade

mkdir $UPGRADE_LOGDIR
cd $ORACLE_HOME/rdbms/admin
switches="-c '$DELPHIX_PDB_NAME' -l $UPGRADE_LOGDIR"
$ORACLE_HOME/perl/bin/perl catctl.pl $switches catupgrd.sql &>> $UPGRADE_LOGFILE
```

CLI cookbook: migrating a virtual PDB in a virtual CDB

Delphix supports migrating from a vPDB in a virtual CDB, to another linked or virtual CDB.

This topic describes how to migrate a virtual pluggable database (vPDB) in a virtual CDB to another virtual CDB using the command-line interface.

Pre-requisites

You should already set up and have Delphix discover a container database in the same environment as the vPDB currently is or from an environment to which the vPDB will be migrated to.

Procedure

1. Log into the Delphix command-line interface using the admin user or a user with admin privileges.

```
ssh admin@YOUR_ENGINE
```

2. Disable the vPDB.

```
delphix> /sourced
elphix source> select myPDB
delphix source 'myPDB'> disable
delphix source 'myPDB' disable *> commit
    Dispatched job JOB-35
    SOURCE_DISABLE job started for "myPDB".
    Disabling virtual database "myPDB".
    Unexporting storage containers.    Virtual database "myPDB" disabled.
    SOURCE_DISABLE job for "myPDB" completed successfully.
delphix source 'myPDB'>
```

3. Change the environment user and repository (ORACLE_HOME) for the vPDB to the appropriate user from the new host.

```
delphix> /sourceconfig
delphix sourceconfig> select myPDB
delphix sourceconfig 'myPDB'> update
delphix sourceconfig 'myPDB' update *> set repository='/u01/app/oracle/product/
19.0.0.0/dbhome_1'
delphix sourceconfig 'myPDB' update *> set environmentUser='/oracle'
delphix sourceconfig 'myPDB' update *> commit
    Dispatched job JOB-38
    SOURCE_CONFIG_UPDATE job started for "myPDB".
    SOURCE_CONFIG_UPDATE job for "myPDB" completed successfully.
```

4. Change the environment user and repository (ORACLE_HOME) for the vCDB to the appropriate user from the new host. This must match the settings in step 3.

```
delphix sourceconfig 'myPDB'> /sourceconfig/
delphix sourceconfig> select myCDB
```

```
delphix sourceconfig 'myCDB'> update
delphix sourceconfig 'myCDB' update *> set repository='/u01/app/oracle/product/
19.0.0.0/dbhome_1'
delphix sourceconfig 'myCDB' update *> set environmentUser='/oracle'
delphix sourceconfig 'myCDB' update *> commit
    Dispatched job JOB-39
    SOURCE_CONFIG_UPDATE job started for "myCDB".
    SOURCE_CONFIG_UPDATE job for "myCDB" completed successfully.
```

5. Enable the vPDB.

```
delphix sourceconfig 'myCDB'> /source
delphix source> select myPDB
delphix source 'myPDB'> enable
delphix source 'myPDB' enable *> commit
    Dispatched job JOB-40
    SOURCE_ENABLE job started for "myPDB".
    Enabling dataset "myPDB".
    Exporting storage containers from the Delphix Engine.
    Mounting datasets.
    Mounting filesystems for the virtual database instance "1".
    Starting virtual database.
    Starting instance 1 on virtual database "myCDB".
    Virtual database "myCDB" was successfully started.
    Starting virtual database.
    Starting instance 1 on virtual database "myPDB".
    Plugging in Oracle pluggable database.
    Opening Oracle pluggable database.
    Setting OMF destination for Oracle pluggable database.
    Creating PDB tempfiles.
    Checking Oracle pluggable database plugin violations.
    Virtual database "myPDB" was successfully started.
    Dataset "myPDB" enabled.
    SOURCE_ENABLE job for "myPDB" completed successfully.
```

CLI cookbook: Migrating an Oracle RAC virtual PDB and a virtual CDB

Delphix supports migrating from a RAC vPDB in a virtual CDB, to another RAC linked CDB or virtual CDB. This feature is only available through the command-line interface.

> ⚠ Attempts to perform this migration via GUI will fail with the following error:
> **Error**
> The operation could not be completed because the virtual database “vPDB1" is disabled.
> **Error Code**
> exception.db.genericvdb.disabled

This topic describes how to migrate a RAC virtual pluggable database (vPDB) in a virtual CDB to another RAC virtual CDB using the command-line interface.

Pre-requisites

You should already be set up and have Delphix discover a container database in the same environment as the vPDB currently is or from an environment to which the vPDB will be migrated to.

Procedure

1. Log into the Delphix command-line interface using the admin user or a user with admin privileges.

```
ssh admin@YOUR_ENGINE
```

2. Disable the vPDB.

```
delphix> /source
delphix source> select racPDB
delphix source 'racPDB'> disable
delphix source 'racPDB' disable *> commit
    Dispatched job JOB-74
    SOURCE_DISABLE job started for "racPDB".
    Disabling virtual database "racPDB".
    Unexporting storage containers.
    Virtual database "racPDB" disabled.
    SOURCE_DISABLE job for "racPDB" completed successfully.
```

3. Change the environment user and repository (ORACLE_HOME) for the vPDB to the appropriate user from the new host.

```
delphix source 'racPDB'> /sourceconfig/
delphix sourceconfig> select racPDB
delphix sourceconfig 'racPDB'> update
delphix sourceconfig 'racPDB' update *> set repository='/u01/app/oracle/
product/12.1.0.2/dbhome_1'
delphix sourceconfig 'racPDB' update *> set environmentUser=' /oracle'
delphix sourceconfig 'racPDB' update *> commit
    Dispatched job JOB-76
    SOURCE_CONFIG_UPDATE job started for "racPDB".
```

```
    SOURCE_CONFIG_UPDATE job for "racPDB" completed successfully.
```

4. Change the environment user and repository (ORACLE_HOME) for the vCDB to the appropriate user from the new host. This must match the settings in step 3. The instance configuration must also be configured for the new hosts.

```
delphix sourceconfig 'racPDB'> /sourceconfig/
delphix sourceconfig> select racCDB
delphix sourceconfig 'racCDB'> update
delphix sourceconfig 'racCDB' update *> set repository='/u01/app/oracle/
product/12.1.0.2/dbhome_1'
delphix sourceconfig 'racCDB' update *> set environmentUser=' /oracle'
delphix sourceconfig 'racCDB' update *> edit instances
delphix sourceconfig 'racCDB' update instances *> ls
Properties
    0:
        type: OracleRACInstance
        instanceName: racCDB1
        instanceNumber: 1
        node: mwtestx1    1:
        type: OracleRACInstance
        instanceName: racCDB2
        instanceNumber: 2
        node: mwtestx2Use the "add" command to add an element to this array.
delphix sourceconfig 'racCDB' update instances *> set 0.node=mwtestz1
delphix sourceconfig 'racCDB' update instances *> set 1.node=mwtestz2
delphix sourceconfig 'racCDB' update instances *> commit
    Dispatched job JOB-77
    SOURCE_CONFIG_UPDATE job started for "racCDB".
    SOURCE_CONFIG_UPDATE job for "racCDB" completed successfully.
```

5. Enable the vPDB.

```
delphix sourceconfig 'racCDB'> /sourced
elphix source> select racPDB
delphix source 'racPDB'> enable
delphix source 'racPDB' enable *> commit
    Dispatched job JOB-78
    SOURCE_ENABLE job started for "racPDB".
    Enabling dataset "racPDB".
    Exporting storage containers from the Delphix Engine.
    Mounting datasets.
    Mounting filesystems for the virtual database instance "2".
    Starting virtual database.
    Starting instance 2 on virtual database "racCDB".
    Virtual database "racCDB" was successfully started.
    Starting virtual database.
    Starting instance 2 on virtual database "racPDB".
    Plugging in Oracle pluggable database.
    Opening Oracle pluggable database.
    Setting OMF destination for Oracle pluggable database.
```

```
Creating PDB tempfiles.
Checking Oracle pluggable database plugin violations.
Virtual database "racPDB" was successfully started.
Mounting datasets.
Mounting filesystems for the virtual database instance "1".
Starting virtual database.
Starting instance 1 on virtual database "racCDB".
Virtual database "racCDB" was successfully started.
Starting virtual database.
Starting instance 1 on virtual database "racPDB".
Opening Oracle database.
Checking Oracle pluggable database plugin violations.
Virtual database "racPDB" was successfully started.
Dataset "racPDB" enabled.
SOURCE_ENABLE job for "racPDB" completed successfully.
```

CLI cookbook: Locating and updating the value of tdeKeyIdentifier

This topic describes how to manage the tdeKeyIdentifier field that is associated with the vPDB source object using the command-line interface.

> This process is currently supported only via CLI.

Procedure

1. Log into the Delphix command-line interface using the admin user or a user with admin privileges.

```
$ ssh admin@YOUR_ENGINE
```

2. Move to the database.

```
delphix> source
delphix source> "VCDO_1JL"
```

3. Viewing all the settings using the "ls" command.

```
delphix source "VCDO_1JL" *> ls
Properties
    type: OracleVirtualPdbSource
    name: VCDO_1JL
    allowAutoVDBRestartOnHostReboot: false
    archivelogMode: true
    config: VCDO_1JL
    configParams:
        _cdb_disable_pdb_limit: TRUE
        audit_file_dest: '/u01/app/oracle/admin/CDOMLOSR197/adump'
        audit_trail: 'DB'
        compatible: '19.0.0'
        core_dump_dest: '/u01/app/oracle/diag/rdbms/cdomlosr197/CDOMLOSR197/
cdump'
        diagnostic_dest: '/u01/app/oracle'
        dispatchers: '(PROTOCOL=TCP) (SERVICE=CDOMLOSRCA1DXDB)'
        enable_pluggable_database: TRUE
        log_archive_format: '%t_%s_%r.dbf'
        max_pdbs: 4098
        memory_max_target: 1342177280
        memory_target: 1342177280
        nls_language: 'AMERICAN'
        nls_territory: 'AMERICA'
        open_cursors: 300
        processes: 300
        remote_login_passwordfile: 'EXCLUSIVE'
    configTemplate: (unset)
    container: VCDO_1JL
    customEnvVars: (empty)
```

```
    linked: false
    logCollectionEnabled: false
    mountBase: /mnt/provision
    newDBID: false
    nodeListeners: (empty)
    operations:
        type: VirtualSourceOperations
        configureClone: (empty)
        postRefresh: (empty)
        postRollback: (empty)
        postSnapshot: (empty)
        postStart: (empty)
        postStop: (empty)
        preRefresh: (empty)
        preRollback: (empty)
        preSnapshot: (empty)
        preStart: (empty)
        preStop: (empty)
    parentTdeKeystorePassword: ********
    parentTdeKeystorePath: /u01/app/oracle/keystores/CDOMLOSR197/wallet
    redoLogGroups: 3
    redoLogSizeInMB: 50
    reference: ORACLE_VIRTUAL_PDB_SOURCE-2
    runtime:
        type: OraclePDBSourceRuntime
        accessible: true
        accessibleTimestamp: 2021-10-06T22:02:15.718Z
        activeInstances:
            0:
                type: OracleActiveInstance
                hostName: ip-10-110-234-67.delphix.com
                instanceName: CDOMLOSR197
                instanceNumber: 1
        databaseMode: READ_WRITE
        databaseRole: PRIMARY
        databaseSize: 913.4MB
        databaseStats: [ ... ]
        enabled: ENABLED
        lastNonLoggedLocation: 0
        status: RUNNING
    runtimeMountInformation:
        type: UnixRuntimeMountInformation
        name: (unset)
        nfsVersion: 4
        nfsVersionReason: DEFAULT
    staging: false
    status: DEFAULT
    tdeExportedKeyFileSecret: ********
    tdeKeyIdentifier: AbSP7gninU+Gv1YQ/iEcJAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
    tdeUUID: a3f26971-1df6-4c81-994f-4b2c582ded87
    virtual: true
Operations
```

```
update
enable
disable
start
stop
upgrade
```

4. Note that tdeKeyIdentifier is one of the last fields listed above. If we query the vPDB via sqlplus, we can see the matching key_id.
   Note that any key generated by Delphix will include a tag with the format `dlpx_key_<tdeUUID>`.

```
SQL> alter session set container=VCDO_1JL;
Session altered.
SQL> select key_id, tag from v$encryption_keys;
KEY_ID----------------------------------------------------------------------------
-----
TAG---------------------------------------------------------------------------
----
AbSP7gninU+Gv1YQ/iEcJAAAAAAAAAAAAAAAAAAAAAAAAAAAA
dlpx_key_a3f26971-1df6-4c81-994f-4b2c582ded87
```

5. To generate a new unique encryption key, unset the value of tdeKeyIdentifier before a refresh or rewind operation.

```
delphix source 'VCDO_1JL'> update
delphix source 'VCDO_1JL' update *> unset tdeKeyIdentifier
delphix source 'VCDO_1JL' update *> ls
Properties
    type: OracleVirtualPdbSource
    name: VCDO_1JL
    allowAutoVDBRestartOnHostReboot: false
    customEnvVars: (empty)
    logCollectionEnabled: false
    newDBID: false
    operations:
        type: VirtualSourceOperations
        configureClone: (empty)
        postRefresh: (empty)
        postRollback: (empty)
        postSnapshot: (empty)
        postStart: (empty)
        postStop: (empty)
        preRefresh: (empty)
        preRollback: (empty)
        preSnapshot: (empty)
        preStart: (empty)
        preStop: (empty)
    parentTdeKeystorePassword: ********
    parentTdeKeystorePath: /u01/app/oracle/keystores/CDOMLOSR197/wallet
    tdeKeyIdentifier: (unset) (*)
```

```
delphix source 'VCDO_1JL' update *> commit
    Dispatched job JOB-18
    SOURCE_UPDATE job started for "VCDO_1JL".
    SOURCE_UPDATE job for "VCDO_1JL" completed successfully.
```

6. After the refresh or rewind, the new key identifier is now associated with vPDB that can be used for all future Delphix operations. View all the settings using the "ls" command.

```
delphix source 'VCDO_1JL'> ls
Properties
    type: OracleVirtualPdbSource
    name: VCDO_1JL
    allowAutoVDBRestartOnHostReboot: false
    archivelogMode: true
    config: VCDO_1JL
    configParams:
        _cdb_disable_pdb_limit: TRUE
        audit_file_dest: '/u01/app/oracle/admin/CDOMLOSR197/adump'
        audit_trail: 'DB'
        compatible: '19.0.0'
        core_dump_dest: '/u01/app/oracle/diag/rdbms/cdomlosr197/CDOMLOSR197/
cdump'
        diagnostic_dest: '/u01/app/oracle'
        dispatchers: '(PROTOCOL=TCP) (SERVICE=CDOMLOSRCA1DXDB)'
        enable_pluggable_database: TRUE
        log_archive_format: '%t_%s_%r.dbf'
        max_pdbs: 4098
        memory_max_target: 1342177280
        memory_target: 1342177280
        nls_language: 'AMERICAN'
        nls_territory: 'AMERICA'
        open_cursors: 300
        processes: 300
        remote_login_passwordfile: 'EXCLUSIVE'
    configTemplate: (unset)
    container: VCDO_1JL
    customEnvVars: (empty)
    linked: false
    logCollectionEnabled: false
    mountBase: /mnt/provision
    newDBID: false
    nodeListeners: (empty)
    operations:
       type: VirtualSourceOperations
        configureClone: (empty)
        postRefresh: (empty)
        postRollback: (empty)
        postSnapshot: (empty)
        postStart: (empty)
        postStop: (empty)
        preRefresh: (empty)
        preRollback: (empty)
```

```
        preSnapshot: (empty)
        preStart: (empty)
        preStop: (empty)
    parentTdeKeystorePassword: ********
    parentTdeKeystorePath: /u01/app/oracle/keystores/CDOMLOSR197/wallet
    redoLogGroups: 3
    redoLogSizeInMB: 50
    reference: ORACLE_VIRTUAL_PDB_SOURCE-2
    runtime:
        type: OraclePDBSourceRuntime
        accessible: true
        accessibleTimestamp: 2021-10-06T22:17:15.907Z
        activeInstances:
            0:
                type: OracleActiveInstance
                hostName: ip-10-110-234-67.delphix.com
                instanceName: CDOMLOSR197
                instanceNumber: 1
        databaseMode: READ_WRITE
        databaseRole: PRIMARY
        databaseSize: 913.4MB
        databaseStats: [ ... ]
        enabled: ENABLED
        lastNonLoggedLocation: 0
        status: RUNNING
    runtimeMountInformation:
        type: UnixRuntimeMountInformation
        name: (unset)
        nfsVersion: 4
        nfsVersionReason: DEFAULT
    staging: false
    status: DEFAULT
    tdeExportedKeyFileSecret: ********
    tdeKeyIdentifier: AVEhXrBvmU+Cv+lK6ghT6oMAAAAAAAAAAAAAAAAAAAAAAAAAAAA
    tdeUUID: a3f26971-1df6-4c81-994f-4b2c582ded87
    virtual: true
```

7. To specify a user-defined encryption key to be used for future Delphix operations, set tdeKeyIdentifier to the value of a valid key_id in the CDB's keystore. Note that if an invalid key_id is provided, refresh or rewind will fail and it will be necessary to unset or update the tdeKeyIdentifier parameter with a valid key_id. Note that this key_id will not have a corresponding dlpx tag unless it is a key previously generated by Delphix.

```
delphix source 'VCDO_1JL'> update
delphix source 'VCDO_1JL' update *> set tdeKeyIdentifier="AbSP7gninU+Gv1YQ/
iEcJAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"
delphix source 'VCDO_1JL' update *> ls
Properties
    type: OracleVirtualPdbSource
    name: VCDO_1JL
    allowAutoVDBRestartOnHostReboot: false
    customEnvVars: (empty)
    logCollectionEnabled: false
```

```
    newDBID: false
    operations:
        type: VirtualSourceOperations
        configureClone: (empty)
        postRefresh: (empty)
        postRollback: (empty)
        postSnapshot: (empty)
        postStart: (empty)
        postStop: (empty)
        preRefresh: (empty)
        preRollback: (empty)
        preSnapshot: (empty)
        preStart: (empty)
        preStop: (empty)
    parentTdeKeystorePassword: ********
    parentTdeKeystorePath: /u01/app/oracle/keystores/CDOMLOSR197/wallet
    tdeKeyIdentifier: AbSP7gninU+Gv1YQ/iEcJAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
delphix source 'VCDO_1JL' update *> commit
    Dispatched job JOB-22
    SOURCE_UPDATE job started for "VCDO_1JL".
    SOURCE_UPDATE job for "VCDO_1JL" completed successfully.
```

8. After a refresh or rewind, this key identifier will be associated with vPDB and will be used for all future Delphix operations. View all the settings using the "ls" command.

```
delphix source 'VCDO_1JL'> ls
Properties
    type: OracleVirtualPdbSource
    name: VCDO_1JL
    allowAutoVDBRestartOnHostReboot: false
    archivelogMode: true
    config: VCDO_1JL
    configParams:
        _cdb_disable_pdb_limit: TRUE
        audit_file_dest: '/u01/app/oracle/admin/CDOMLOSR197/adump'
        audit_trail: 'DB'
        compatible: '19.0.0'
        core_dump_dest: '/u01/app/oracle/diag/rdbms/cdomlosr197/CDOMLOSR197/
cdump'
        diagnostic_dest: '/u01/app/oracle'
        dispatchers: '(PROTOCOL=TCP) (SERVICE=CDOMLOSRCA1DXDB)'
        enable_pluggable_database: TRUE
        log_archive_format: '%t_%s_%r.dbf'
        max_pdbs: 4098
        memory_max_target: 1342177280
        memory_target: 1342177280
        nls_language: 'AMERICAN'
        nls_territory: 'AMERICA'
        open_cursors: 300
        processes: 300
        remote_login_passwordfile: 'EXCLUSIVE'
    configTemplate: (unset)
```

```
container: VCDO_1JL
customEnvVars: (empty)
linked: false
logCollectionEnabled: false
mountBase: /mnt/provision
newDBID: false
nodeListeners: (empty)
operations:
    type: VirtualSourceOperations
    configureClone: (empty)
    postRefresh: (empty)
    postRollback: (empty)
    postSnapshot: (empty)
    postStart: (empty)
    postStop: (empty)
    preRefresh: (empty)
    preRollback: (empty)
    preSnapshot: (empty)
    preStart: (empty)
    preStop: (empty)
parentTdeKeystorePassword: ********
parentTdeKeystorePath: /u01/app/oracle/keystores/CDOMLOSR197/wallet
redoLogGroups: 3
redoLogSizeInMB: 50
reference: ORACLE_VIRTUAL_PDB_SOURCE-2
runtime:
    type: OraclePDBSourceRuntime
    accessible: true
    accessibleTimestamp: 2021-10-06T22:17:15.907Z
    activeInstances:
        0:
            type: OracleActiveInstance
            hostName: ip-10-110-234-67.delphix.com
            instanceName: CDOMLOSR197
            instanceNumber: 1
    databaseMode: READ_WRITE
    databaseRole: PRIMARY
    databaseSize: 913.4MB
    databaseStats: [ ... ]
    enabled: ENABLED
    lastNonLoggedLocation: 0
    status: RUNNING
runtimeMountInformation:
    type: UnixRuntimeMountInformation
    name: (unset)
    nfsVersion: 4
    nfsVersionReason: DEFAULT
staging: false
status: DEFAULT
tdeExportedKeyFileSecret: ********
tdeKeyIdentifier: AbSP7gninU+Gv1YQ/iEcJAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
tdeUUID: a3f26971-1df6-4c81-994f-4b2c582ded87
```

```
    virtual: true
```

CLI cookbook: Force refresh/rewind a virtual PDB

This topic describes how to force refresh/rewind a virtual pluggable database. To know more about the `force` option and when to use it refer to Refreshing or rewinding a broken/unusable virtual PDB.

Procedure

Refresh

1. Follow the steps in CLI Cookbook: Refresh a VDB from a specific timepoint or latest to select a Specific or Latest Timepoint for refreshing the virtual pluggable database but do not commit.
2. Set the `force` parameter

```
delphix database '<virtual_pdb_name>' refresh *> set force=true
```

3. After all the parameters are set (including `timeflowPointParameters`), initiate the refresh by committing the operation in the CLI:

```
delphix database '<virtual_pdb_name>' refresh *> commit
```

Rewind (Rollback)

1. Follow the steps in Rolling back or rewinding to a snapshot from a VDB to initiate rewind and set the `timeflowPointParameters` but do not commit.
2. Set the `force` parameter:

```
delphix database '<virtual_pdb_name>' rollback *> set force=true
```

3. After all the parameters are set (including `timeflowPointParameters`), initiate the rewind (rollback) by committing the operation in the CLI:

```
delphix database '<virtual_pdb_name>' rollback *> commit
```

CLI cookbook: replication

These topics describe how to use the command-line interface for replication tasks.

This section covers the following topics:

CLI cookbook: adding a replication spec

This topic describes how to use the command-line interface to add a replication specification to the Delphix Engine.

Unlike the GUI, the CLI supports the ability to manage multiple replication specifications within a single system. This allows updates to be sent to multiple systems from a single point.

Prerequisites

You should review the topic Replication Overview to understand which objects are copied as part of a backup or restore operation, as well as the dependencies between objects.

Procedure

1. Switch to the replication spec context.

```
delphix> cd replication/spec
delphix replication spec> ls
Operations
create
```

2. Create a new replication spec.

```
delphix replication spec> create
delphix replication spec create *> ls
Properties
    type: ReplicationSpec
    name: (unset)
    bandwidthLimit: (unset)
    enabled: (unset)
    encrypted: (unset)
    objects: (required)
    schedule: (unset)
    targetCredential:
        type: PasswordCredential
        password: (required)
    targetHost: (required)
    targetPrincipal: (required)
```

3. Specify the target hostname, user, and credentials.

```
delphix replication spec create *> set targetHost=exampleHost.mycompany.com
delphix replication spec create *> set targetPrincipal=delphix_admin
delphix replication spec create *> set targetCredential.password=password
```

Info:
SDD Secure replication
To create a Selective Data Distribution (SDD) type spec set the objectSpecification.type

1.
```
delphix replication spec create *> set
objectSpecification.type=ReplicationSecureList
```

This parameter defaults to ReplicationList which implies a regular replication spec.

Target Principal
The target principal must be a Delphix user on the target host who has domain privileges.

4. Specify the set of objects to replicate.
   a. To replicate all dSources and VDBs on the system, specify `` `DOMAIN `` as the list of objects.

```
delphix replication spec create *> set objectSpecification.objects=`DOMAIN
```

   b. To replicate a subset of Groups, VDBs, dSources, and Timeflows specify their names as a comma-separated list.

```
delphix replication spec create *> set objectSpecification.objects=Group1/
vdb1,Group2/vdb2
```

1. **Name Completion**
   The CLI will provide possible completions for all objects in the system, but only groups, dSources and VDBs can be specified. Attempts to replicate other types of objects will generate an error when the operation is committed.
2. **SSD Secure replication**
   If objectSpecification.type=ReplicationSecureList was selected, then objectSpecification.containers needs to be used instead of objectSpecification.objects.
   The containers have to be Masked VDBs

5. Commit the operation.

```
delphix replication spec create *> commit
    `REPLICATION_SPEC-1
```

CLI cookbook: deleting a replication spec

This topic describes how to use the command-line interface to delete a replication spec.

Procedure

1. Switch to the replication spec context and list the specs on the system.

```
delphix> cd replication/spec
delphix replication spec> ls
Objects
REFERENCE           TARGETHOST
REPLICATION_SPEC-1  exampleHost.mycompany.com

Operations
create
```

2. Select the replication spec to remove.

```
delphix replication spec> select REPLICATION_SPEC-1
delphix replication spec "exampleHost.mycompany.com">
```

3. Remove the spec.

```
delphix replication spec "exampleHost.mycompany.com"> delete
delphix replication spec "exampleHost.mycompany.com" delete *> commit
```

CLI cookbook: failing over a namespace

This topic describes how to use the command line interface to fail over a namespace.

Procedure

1. Switch to the namespace context and list the namespaces on the system.

```
delphix> cd namespace
delphix namespace> ls
Objects
NAME
[172.16.203.93]

Operations
lookup
```

2. Select the namespace to failover.

```
delphix namespace> select [172.16.203.93]
delphix namespace "[172.16.203.93]">
```

3. Failover the namespace.

```
delphix namespace "[172.16.203.93]"> failover
delphix namespace "[172.16.203.93]" failover *> commit
```

Failover
Failover will sever the replication connection and make objects in the namespace part of the live system. This will prevent the target from receiving subsequent incremental updates.

CLI cookbook: mapping replication specs to objects

After creating replication specs often you will want to see what is mapping to which target, this document will show you how to find that information in the CLI. It will also show you how to navigate the replication directory in the CLI.

1. ssh into the CLI as a user with delphix_admin privileges.

```
ssh admin@yourengine
```

2. Next go to the replication directory in the CLI and list all of the specs.

```
delphix > replication
delphix replication > spec
delphix replication spec > ls
Objects
REFERENCE           TARGETHOST
REPLICATION_SPEC-1  test1
Operations
create
```

3. Then select the replication spec to find where is maps to and what objects are selected with it.

```
delphix replication spec> select REPLICATION_SPEC-1
delphix replication spec 'test'> ls
Properties
    type: ReplicationSpec
    name: test
    bandwidthLimit: 0
    description: (unset)
    enabled: false
    encrypted: false
    numberOfConnections: 1
    objectSpecification:
        type: ReplicationList
        name: (unset)
        objects: Untitled/dbdhcp3
    reference: REPLICATION_SPEC-1
    runtime:
        type: ReplicationSpecRuntime
    schedule: (unset)
    tag: 0ddae174-9486-4363-9704-bfc3398e547e
    targetCredential:
        type: PasswordCredential
        password: ********
    targetHost: test1
    targetPort: 8415
    targetPrincipal: delphix
    useSystemSocksSetting: false
Operations
delete
```

```
update
execute
```

CLI cookbook: triggering immediate execution of a replication spec

This topic describes how to use the command-line interface to trigger an immediate execution of a replication spec in the Delphix Engine.

Procedure

1. Switch to the replication spec context and list the specs on the system.

```
delphix> cd replication/spec
delphix replication spec> ls
Objects
REFERENCE           TARGETHOST
REPLICATION_SPEC-1  exampleHost.mycompany.com

Operations
create
```

2. Select the replication spec to execute.

```
delphix replication spec> select REPLICATION_SPEC-1
delphix replication spec "exampleHost.mycompany.com">
```

3. Execute the spec.

```
delphix replication spec "exampleHost.mycompany.com"> execute
delphix replication spec "exampleHost.mycompany.com" execute *> commit
    Dispatched job JOB-7
    REPLICATION_SEND job started.
    Connecting to target "exampleHost.mycompany.com".
    Preparing replication update.
    Starting incremental replication update.
    Sending metadata.
    Sending data for "Untitled".
    Sending data for "Untitled/redsox1".
    Transfer completed in 0:00:01, sent 1.39MB (1.39MB/s).
    Committing serialization state.
    REPLICATION_SEND job completed successfully.
```

CLI cookbook: Delphix self-service actions

These entries will help with some of the Delphix Self-Service Actions that can be performed in the CLI and therefore with the Delphix APIs:

- CLI cookbook: how to create a Delphix self-service bookmark
- CLI cookbook: how to create a Delphix self-service branch
- CLI cookbook: how to create a Delphix self-service container
- CLI cookbook: how to create a Delphix self-service database template
- CLI cookbook: how to create a Delphix self-service user
- CLI cookbook: how to delete a Delphix self-service bookmark
- CLI cookbook: how to delete a Delphix self-service container
- CLI cookbook: how to delete a Delphix self-service template
- CLI cookbook: how to refresh a Delphix self-service container
- CLI cookbook: how to share a Delphix self-service bookmark
- CLI cookbook: how to update a Delphix self-service bookmark

CLI cookbook: how to create a Delphix self-service bookmark

Prerequisites:

- Have Delphix Self-Service user privileges.
- Know the branch you would like to create the bookmark on.
- Know the container or template that the branch belongs to.
- (Optional) Know when you would like the bookmark to expire.

Delphix Self-Service administrators can use this CLI cookbook recipe to create a bookmark on Delphix Self-Service using the Delphix Engine CLI.

> The following script is for educational and demonstration purposes only and is not supported by Delphix.

This script can be downloaded by selecting createBookmark.sh**.**

Creating a bookmark in Delphix Self-Service

```
#!/bin/bash
# A sample script for calls to the CLI. This one creates a Jet Stream bookmark.
#
# VERY IMPORTANT: In order for this to work, you need to go through the steps here:
# https://docs.delphix.com/display/DOCS43/CLI+Cookbook%3A+Configuring+Key-
Based+SSH+Authentication+for+Automation
# After this you will not need to use a username and password to log into the delphix
engine. If you do not
# setup the SSH authentication you will have to manually enter the password.
#


##example##
#./createBookmark.sh -e "2016-07-27T23:38:56.453Z" -t "2016-07-27T01:45:56.453Z" -T
[tag1,tag2,tag3,tag4,tag5] bkmrk3 JS_BRANCH-41


##### Constants

# Describes a Delphix software revision.
VERSION="1.11.10"


##### Default Values. These can be overwriten with optional arguments.
engine="ars-6010.dlpxdc.co"
username="admin"



##### Functions


# Help Menu
function usage {
```

```
    echo "Usage: createBookmark.sh [[-h] | options...] <bookmark name> <branch name
format JS_BRANCH-n>"
    echo "Create a Jet Stream Bookmark on the given branch."
    echo ""
    echo "Positional arguments"
    echo "  <bookmark name> "
    echo "  <branch name>"
    echo ""
    echo "Optional Arguments:"
    echo "  -h                Show this message and exit"
    echo "  -d                Delphix engine IP address or host name, otherwise
revert to default"
    echo "  -u                Server user. Password needs to manually provide at run
time, otherwise revert to default"
    echo "  -D                Description of this bookmark. Type: string"
    echo "  -s                Pass [-s true]  if need to make bookmark in shared
mode"
    echo "  -e                The time at which the bookmark should be expired"
    echo "  -t                The time at which the bookmark should be created. If no
time is included, the bookmark will be created at the latest point in time. Type:
date, must be in ISO 8601 extended format [yyyy]-[MM]-[dd]T[HH]:[mm]:[ss].[SSS]Z"
    echo "  -T                A set of user-defined labels for this bookmark. No
spaces allowed. Array of Type: string. In format, [tag1,tag2,..] "
}

# Create Our Session, including establishing the API version.
function create_session
{
    echo "creating session..."
    SSH_CMD="ssh ${username}@${engine}"
    ${SSH_CMD}  "version $VERSION"
    check_result
}


# Check the result of the curl. If there are problems, inform the user then exit.
function check_result
{
    exitStatus=$?
    if [ $exitStatus -ne 0 ]
    then
        echo "command failed with exit status $exitStatus"
        echo $result
        exit 1
    fi
}


function get_branch
{
    echo "retrieveing branch $branchRef to find Source Data Layout..."
    result=$(${SSH_CMD}  "selfservice branch; ls; select ${branchRef}; ls")
```

```
    # Get everything in the result that comes after dataLayout.
    temp=${result#*dataLayout: }

    # Get rid of everything after
    dataLayout=${temp%% *}

}



function create_bookmark
{
    get_branch

    echo "creating bookmark..."

    # If there is not creation time, we need to use JSTimelinePointLatestTimeInput.
    if [ -z $creationTime ]
    then
        pointParams="set timelinePointParameters.sourceDataLayout=$dataLayout;"
        pointParams="$pointParams set
timelinePointParameters.type=JSTimelinePointLatestTimeInput;"
    else
        pointParams="set timelinePointParameters.sourceDataLayout=$dataLayout;"
        pointParams="$pointParams set
timelinePointParameters.type=JSTimelinePointTimeInput;"
        pointParams="$pointParams set
timelinePointParameters.time=\"$creationTime\";"
        pointParams="$pointParams set timelinePointParameters.branch=\"$branchRef\";"

    fi

    if [[ -n $expirationTime ]]
    then
        pointParams="$pointParams set bookmark.expiration=\"$expirationTime\";"
    fi

    # These are the required parameters.
    paramString="
        \"bookmark\": {
            \"branch\": \"${branchRef}\",
            \"name\": \"${bookmarkName}\","
    paramString="selfservice bookmark; create; set bookmark.name=$bookmarkName; set
bookmark.branch=$branchRef;"
    paramString="$paramString $pointParams"

    # Fill in optional parameters if there are any.
    if [[ -n $description ]]
    then
        paramString="$paramString set bookmark.description=\"$description\";"
    fi

    if [[ -n $shared ]]
```

```
    then
        paramString="$paramString set bookmark.shared=$shared;"
    fi

    if [[ -n $tags ]]
    then
        # Add quotes back to the passed in tags so they are processed correctly.
        tags=${tags//[/[\"}
        tags=${tags//,/\",\"}
        tags=${tags//]/\"]}

        paramString="$paramString set bookmark.tags=$tags;"
    fi
    paramString="$paramString commit;"
    result=$(${SSH_CMD} $paramString)
    check_result
    echo "Verifying job status..."
    # Get everything in the result that comes after job.
        temp=${result#*job}
    # Get rid of everything after
    resultArray=($temp)
    jobRef=($resultArray)
        jobString="job;select $jobRef;ls"
        result=$(${SSH_CMD} $jobString)
    check_result
    # Get everything in the result that comes after job.
    temp=${result#*jobState:}
    # Get rid of everything after
    resultArray=($temp)
    jobState=($resultArray)

    if [ $jobState = "COMPLETED" ]
    then
        echo "successfully created bookmark $bookmarkName"
    else
        echo "unable to create bookmark"
        echo result
    fi

}


##### Main

while getopts "u:d:D:s:e:t:T:h" flag; do
    case "$flag" in
            u )             username=${OPTARG%:*}
                        ;;
        d )             engine=$OPTARG
                        ;;
        D )             description=$OPTARG
                        ;;
        s )             shared=true
```

```
                        ;;
        t )             creationTime=$OPTARG
                        ;;
        e )             expirationTime=$OPTARG
                        ;;
        T )             tags=$OPTARG
                        ;;
        h )             usage
                        exit
                        ;;
        * )             usage
                        exit 1
    esac
done


# Shift the parameters so we only have the positional arguments left
shift $((OPTIND-1))

# Check that there are 2 positional arguments
if [ $# != 2 ]
then
    usage
    exit 1
fi

# Get the two positional arguments
bookmarkName=$1
shift
branchRef=$1
create_session
create_bookmark
```

CLI cookbook: how to create a Delphix self-service branch

Delphix Self-Service administrators can use this CLI cookbook recipe to create a branch on Delphix Self-Service using the Delphix Engine CLI.

> The following script is for educational and demonstration purposes only and is not supported by Delphix.

This script can be downloaded by selecting **createBranch.sh**.

Creating a branch in Delphix Self-Service

```
#!/bin/bash

# A sample script for calls to the CLI. This one creates a Jet Stream branch.
#
# VERY IMPORTANT: In order for this to work, you need to go through the steps here:
# https://docs.delphix.com/display/DOCS43/CLI+Cookbook%3A+Configuring+Key-
Based+SSH+Authentication+for+Automation
# After this you will not need to use a username and password to log into the delphix
engine. If you do not
# setup the SSH authentication you will have to manually enter the password.
#
# Note that the CLI only allows branches to be created from below
# 1) From latest point in time
# 2) From specific bookmark
# 3) From specific point in time



##examples##
# Create branch from latest point in time
#./createBranch.sh NewBranchName JS_DATA_CONTAINER-20
# Create branch from specific bookmark
#./createBranch.sh -b ExistingBookmarkName NewBranchName JS_DATA_CONTAINER
# Create branch from specific point in time
#./createBranch.sh -t "2016-07-27T01:45:56.453Z" -B JS_BRANCH-2 NewBranchName
JS_DATA_CONTAINER-20


##### Constants

# Describes a Delphix software revision.
VERSION="1.11.10"


##### Default Values. These can be overwriten with optional arguments.
engine="ars3-6010.dlpxdc.co"
username="admin"



##### Functions
```

```
# Help Menu
function usage {
    echo "Usage: createBranch.sh [[-h] | options...] <name> <container>"
    echo "Create a Jet Stream Bookmark on the given branch."
    echo ""
    echo "Positional arguments"
    echo "  <NewBranchName>"
    echo "  <container> format JS_DATA_CONTAINER-<n>"
    echo ""
    echo "Optional Arguments:"
    echo "  -h                Show this message and exit"
    echo "  -d                Delphix engine IP address or host name, otherwise
revert to default"
    echo "  -u                Server user. Password needs to manually provide at run
time, otherwise revert to default"
    echo "  -b                Bookmark name from which need to create branch. If no
bookmark is included, the branch will be created at the latest point in time. Type:
string"
    echo "  -t                The time at which the branch should be created. If no
time is included, the branch will be created at the latest point in time. Type: date,
must be in ISO 8601 extended format [yyyy]-[MM]-[dd]T[HH]:[mm]:[ss].[SSS]Z"
}

# Create Our Session, including establishing the API version.
function create_session
{
    echo "creating session..."
    SSH_CMD="ssh ${username}@${engine}"
    ${SSH_CMD}  "version $VERSION"
    check_result
}


# Check the result of the curl. If there are problems, inform the user then exit.
function check_result
{
    exitStatus=$?
    if [ $exitStatus -ne 0 ]
    then
        echo "command failed with exit status $exitStatus"
        echo $result
        exit 1
    fi
}



function create_branch
{
```

```
    # If there is not timeInput and no bookmark name, we need to use
JSTimelinePointLatestTimeInput.
    if [ -z $inputTime ] && [ -z $bookmark ]
    then
        #code to use JSTimelinePointLatestTimeInput
        pointParams="edit timelinePointParameters; set
type=JSTimelinePointLatestTimeInput;"
        pointParams="$pointParams set sourceDataLayout=$container; commit;"
    # If there is a timeInput and no bookmark name, we need to use Input Time.
    elif [ -n $inputTime ] && [ -z $bookmark ]
     then
        #code to use JSTimelinePointTimeInput
        pointParams="edit timelinePointParameters; set
type=JSTimelinePointTimeInput;"
        pointParams="$pointParams set time=$inputTime; set branch=$branch; commit;"
    # If there is a bookmark name and no time input, we need to use bookmark
    elif [ -z $inputTime ] && [ -n $bookmark ]
    then
        #code to use JSTimelinePointBookmarkInput
        pointParams="set timelinePointParameters.bookmark=$bookmark;"
        pointParams="$pointParams commit;"
    fi

    # These are the required parameters.
    paramString="selfservice branch create;set name=$branchName; set
dataContainer=$container;"

    #Add additional optional parameter
    paramString="$paramString $pointParams"
    #echo $paramString
    echo "Creating Branch..."
    result=$(${SSH_CMD}  $paramString)
    check_result

    echo "Verifying job status..."
    # Get everything in the result that comes after job.
        temp=${result#*job}
    # Get rid of everything after
    resultArray=($temp)
    jobRef=($resultArray)
        jobString="job;select $jobRef;ls"
        result=$(${SSH_CMD} $jobString)
    check_result
    # Get everything in the result that comes after job.
    temp=${result#*jobState:}
    # Get rid of everything after
    resultArray=($temp)
    jobState=($resultArray)
        if [ $jobState = "COMPLETED" ]
    then
        echo "Successfully created branch $branchName"
    else
        echo "Unable to create branch"
```

```
        echo $result
    fi

}


##### Main

while getopts "u:d:b:t:B:h" flag; do
    case "$flag" in
        u )             username=${OPTARG%:*}
                        ;;
        d )             engine=$OPTARG
                        ;;
        b )             bookmark=$OPTARG
                        ;;
        B )             branch=$OPTARG
                        ;;
        t )             inputTime=$OPTARG
                        ;;
        h )             usage
                        exit
                        ;;
        * )             usage
                        exit 1
    esac
done


# Shift the parameters so we only have the positional arguments left
shift $((OPTIND-1))

# Check that there are 2 positional arguments
if [ $# != 2 ]
then
    usage
    exit 1
fi

# Get the two positional arguments
branchName=$1
shift
container=$1

create_session
create_branch
```

CLI cookbook: how to create a Delphix self-service container

Delphix Self-Service administrators can use this CLI cookbook recipe to create a container on Delphix Self-Service using the Delphix Engine CLI.

> The following script is for educational and demonstration purposes only and is not supported by Delphix.

This script can be downloaded by selecting createContainer.sh**.**

Creating a container in Delphix self-service container

```
#!/bin/bash

# A sample script for calls to the CLI. This one creates a Jet Stream container.
#
# VERY IMPORTANT: In order for this to work, you need to go through the steps here:
# https://docs.delphix.com/display/DOCS43/CLI+Cookbook%3A+Configuring+Key-
Based+SSH+Authentication+for+Automation
# After this you will not need to use a username and password to log into the delphix
engine. If you do not
# setup the SSH authentication you will have to manually enter the password.
#


##### Constants

# Describes a Delphix software revision.
VERSION="1.11.10"


##### Default Values. These can be overwriten with optional arguments.
engine="name-of-engine.dlpxdc.co"
username="admin"



##examples##
# Create container from latest point in time
#./createContainer.sh -n "testsource" testcont ORACLE_DB_CONTAINER-269
 JS_DATA_TEMPLATE-13
# Create container from specific bookmark
#./createContainer.sh -n "testsource" -b JS_BOOKMARK-77 testcont ORACLE_DB_CONTAINER-2
69 JS_DATA_TEMPLATE-13
# Create container from specific point in time
#./createContainer.sh -n "testsource" -t "2016-08-08T10:00:00.000Z" -B JS_BRANCH-50
 testcont ORACLE_DB_CONTAINER-269 JS_DATA_TEMPLATE-13

#NOTE: this script will add one container and assign one owner for the container.


##### Functions
```

```
# Help Menu
function usage {
    echo "Usage: createContainer.sh [[-h] | options...] <containername> <vdb>
<template>"
    echo "Create a Jet Stream Container."
    echo ""
    echo "Positional arguments"
    echo "  <name>"
    echo "  <container> format JS_DATA_CONTAINER-<n>"
    echo ""
    echo "Optional Arguments:"
    echo "  -h                Show this message and exit"
    echo "  -d                Delphix engine IP address or host name, otherwise
revert to default"
    echo "  -u                Server user. Password needs to manually provide at run
time, otherwise revert to default"
        echo "  -n                SourceName need to display for container.
(Mandatory)"
    echo "  -b                Bookmark name from which need to create container. If
no bookmark is included, the branch will be created at the latest point in time.
Type: string. Format JS_BOOKMARK-<n> (Optional)"
    echo "  -B                Branch reference from which we need to pick up time
from where the container should be created. Type: string. Format JS_BRANCH-<n>"
    echo "  -t                The time at which the branch should be created. This
must be accompanied with branch name from which need to pick up time. Type: date,
must be in ISO 8601 extended format [yyyy]-[MM]-[dd]T[HH]:[mm]:[ss].[SSS]Z"
    echo "  -N                Optional container notes, if need to add any. Type:
String"
    echo "  -o                Optional owner, to whom we need to assign this
container. Type: String. Format USER-<n>"
}

# Create Our Session, including establishing the API version.
function create_session
{
    echo "creating session..."
    SSH_CMD="ssh ${username}@${engine}"
    ${SSH_CMD}  "version $VERSION"
    check_result
}

# Check the result of the curl. If there are problems, inform the user then exit.
function check_result
{
    exitStatus=$?
    if [ $exitStatus -ne 0 ]
    then
        echo "command failed with exit status $exitStatus"
        echo $result
        exit 1
    fi
}
```

```
function create_container
{
    # If there is not timeInput and no bookmark name, we need to use
JSTimelinePointLatestTimeInput.
    if [[ -z $inputTime  &&  -z $bookmark ]]
    then
        pointParams="set
timelinePointParameters.type=JSTimelinePointLatestTimeInput;"
        pointParams="$pointParams set
timelinePointParameters.sourceDataLayout=$template;"

   # If there is a timeInput, no bookmark name and a branch name, we need to use
Input Time.
    elif [[ -n $inputTime  && -z $bookmark && -n $branch ]]
     then
        pointParams="set timelinePointParameters.type=JSTimelinePointTimeInput;"
        pointParams="$pointParams set timelinePointParameters.time=\"${inputTime}\";"
        pointParams="$pointParams set timelinePointParameters.branch=$branch;"
# If there is a bookmark name and no time input, we need to use bookmark
    elif [[ -z $inputTime  &&  -n $bookmark ]]
     then
        pointParams="set timelinePointParameters.type=JSTimelinePointBookmarkInput;"
        pointParams="$pointParams set timelinePointParameters.bookmark=\"${bookmark}
\";"
    else
        usage
        exit 1
    fi

    # These are the required parameters.

    paramString="selfservice container create;set name=\"${containerName}\";"
    paramString="$paramString set template=\"${template}\";"

    if [[ -n $containerNotes ]]
    then
       paramString="$paramString set notes=\"${containerNotes}\";"
    fi

    if [[ -n $owners ]]
    then
      paramString="$paramString set owner=\"${owners}\";"
    fi
    paramString="$paramString $pointParams;"
    paramString="$paramString edit dataSources; add; set container=$VDB;"
    paramString="$paramString edit source; set type=JSDataSource; set priority=1;set
name=\"${sourceName}\";"

    if [[ -n $sourcedesc ]]
    then
       paramString="$paramString set description=\"${sourcedesc}\";"
```

```
    fi

    paramString="$paramString commit;"
    #echo $paramString
    echo "Creating Container..."
    result=$(${SSH_CMD}  $paramString)
    check_result

    echo "Verifying job status..."
    # Get everything in the result that comes after job.
        temp=${result#*job}
    # Get rid of everything after
    resultArray=($temp)
    jobRef=($resultArray)
        jobString="job;select $jobRef;ls"
        result=$(${SSH_CMD} $jobString)
    check_result
    # Get everything in the result that comes after job.
    temp=${result#*jobState:}
    # Get rid of everything after
    resultArray=($temp)
    jobState=($resultArray)
        if [ $jobState = "COMPLETED" ]
    then
        echo "Successfully created Container $containerName"
    else
        echo "Unable to create Container"
        echo $result
    fi
}

##### Main

while getopts "u:d:b:t:B:D:n:N:o:h" flag; do
    case "$flag" in
        u )             username=${OPTARG%:*}
                        ;;
        d )             engine=$OPTARG
                        ;;
        b )             bookmark=$OPTARG
                        ;;
        t )             inputTime=$OPTARG
                        ;;
        D )             sourcedesc=$OPTARG
                        ;;
        n )             sourceName=$OPTARG
                        ;;
    B )             branch=$OPTARG
            ;;
        N )             containerNotes=$OPTARG
                        ;;
        o )             owners=$OPTARG
                        ;;
```

```
        h )             usage
                        exit
                        ;;
        * )             usage
                        exit 1
    esac
done


# Shift the parameters so we only have the positional arguments left
shift $((OPTIND-1))

# Check that there are 3 positional arguments
if [ $# != 3 ]
then
    usage
    exit 1
fi

# Get the three positional arguments
containerName=$1
shift
VDB=$1
shift
template=$1

create_session
create_container
```

CLI cookbook: how to create a Delphix self-service database template

Delphix Self-Service administrators can use this CLI cookbook recipe to create a database template on Delphix Self-Service using the Delphix Engine CLI.

> The following script is for educational and demonstration purposes only and is not supported by Delphix.

This script can be downloaded by selecting createDBTemplate.sh.

Creating a database template in Delphix Self-Service

```
#!/bin/bash

# A sample script for calls to the CLI. This one creates a Jet Stream Template.
#
# VERY IMPORTANT: In order for this to work, you need to go through the steps here:
# https://docs.delphix.com/display/DOCS43/CLI+Cookbook%3A+Configuring+Key-
Based+SSH+Authentication+for+Automation
# After this you will not need to use a username and password to log into the delphix
engine. If you do not
# setup the SSH authentication you will have to manually enter the password.
#
#

##### Constants

# Describes a Delphix software revision.
VERSION="1.11.10"


##### Default Values. These can be overwriten with optional arguments.
engine="ars-6010.dlpxdc.co"
username="admin"


##examples##
# Create template with mandatory params
# Create template with adding optional params, Notes and Description
#./createDBTemplate.sh -n <sourceName> -N "<templateNotes>" -D "<AnyDescription>"
 <templateName> <containerName>

## NOTE: This script is to add one source per template and it will not add any
properties for template, container or source.


##### Functions

# Help Menu
function usage {
    echo "Usage: createDBTemplate.sh [[-h] | options...] <template_name>
<source_container>"
    echo "Create a Jet Stream Dat Template."
```

```
    echo ""
    echo "Positional arguments"
    echo "  <template_name>"
    echo "  <source_container>"
    echo ""
    echo "Optional Arguments:"
    echo "  -h                Show this message and exit"
    echo "  -d                Delphix engine IP address or host name, otherwise
revert to default"
    echo "  -u        Server user. Password needs to manually provide at run time,
otherwise revert to default"
    echo "  -n                source name to display on JS template screen"
    echo "  -N                template notes, if any. Type: String"
    echo "  -D                source description, if any. Type: String"
}

# Create Our Session, including establishing the API version.
function create_session
{
    echo "creating session..."
    SSH_CMD="ssh ${username}@${engine}"
    ${SSH_CMD}  "version $VERSION"
    check_result
}

# Check the result of the curl. If there are problems, inform the user then exit.
function check_result
{
    exitStatus=$?
    if [ $exitStatus -ne 0 ]
    then
        echo "command failed with exit status $exitStatus"
        echo $result
        exit 1
    fi
}



function create_template
{
    paramString="selfservice template create;set name=$templateName;"
    if [[ -n $templatenotes ]]
        then
           paramString="$paramString set notes=\"$templatenotes\";"
    fi
    paramString="$paramString edit dataSources;add;"
    paramString="$paramString set container=$sourceContainer;set
source.name=$sourcename;set source.priority=1;"
    if [[ -n $sourcedesc ]]
    then
       paramString="$paramString set source.description=\"$sourcedesc\";"
    fi
```

```
    paramString="$paramString commit;"
    #echo $paramString
    echo "Creating Data Template..."
    result=$(${SSH_CMD}  $paramString)
    check_result
    echo "New JetStream template $templateName successfully created"
}


##### Main

while getopts "u:d:n:N:D:h" flag; do
    case "$flag" in
        u )             username=${OPTARG%:*}
                        ;;
        d )             engine=$OPTARG
                        ;;
        n )             sourcename=$OPTARG
                        ;;
        N )             templatenotes=$OPTARG
                        ;;
        D )             sourcedesc=$OPTARG
                        ;;
        h )             usage
                        exit
                        ;;
        * )             usage
                        exit 1

    esac
done

# Shift the parameters so we only have the positional arguments left
shift $((OPTIND-1))

# Check that there are 2 positional arguments
if [ $# != 2 ]
then
    usage
    exit 1
fi

# Get the two positional arguments
templateName=$1
shift
sourceContainer=$1


create_session
create_template
```

CLI cookbook: how to create a Delphix self-service user

Delphix Self-Service administrators can use this CLI cookbook recipe to create a user on Delphix Self-Service using the Delphix Engine CLI.

> The following script is for educational and demonstration purposes only and is not supported by Delphix.

This script can be downloaded by selecting createJSUser.sh.

Creating a Delphix self-service user

```
#!/bin/bash
# A sample script for calls to the CLI. This one creates a Self-Service user.
#
# VERY IMPORTANT: In order for this to work, you need to go through the steps here:
# https://docs.delphix.com/display/DOCS533/CLI+Cookbook%3A+Configuring+Key-Based+SSH+Authentication+for+Automation
# After this you will not need to use a username and password to log into the delphix engine. If you do not
# setup the SSH authentication you will have to manually enter the password.
#
# Note that the CLI only allows branches to be created from existing bookmarks.
##### Constants
# Describes a Delphix software revision.
VERSION="1.6.2"

##### Default Values. These can be overwriten with optional arguments.
engine="10.0.1.10"
username="dev"

##examples##
# Create user with NATIVE authentication
#./createJSUser.sh -P <password> NATIVE <username>
# Create user with LDAP authentication
#./createJSUser.sh -r <principal> <LDAP username>

##### Functions
# Help Menu
function usage {
echo "Usage: createJSUser.sh [[-h] | options...] <auth> <newjsuser>"
echo "Create a Self-Service Only user."
echo ""
echo "Positional arguments"
echo " <auth type NATIVE/LDAP>"
echo " <newjsuser>"
echo ""
echo "Optional Arguments:"
echo " -h Show this message and exit"
echo " -d Delphix engine IP address or host name, otherwise revert to default"
echo " -u Server user. Password needs to manually provide at run time, otherwise revert to default"
echo " -P password for NATIVE authentication, MUST incase auth=NATIVE"
```

```
echo " -f firstName of user"
echo " -l lastName of user"
echo " -e emailAddress of user"
echo " -o homePhoneNumber of user"
echo " -m mobilePhoneNumber of user"
echo " -w workPhoneNumber of user"
echo " -r principal for LDAP authentication, MUST incase of auth=LDAP"
}

# Create Our Session, including establishing the API version.
function create_session
{
echo "creating session..."
SSH_CMD="ssh ${username}@${engine}"
${SSH_CMD} "version $VERSION"
check_result
}
# Check the result of the curl. If there are problems, inform the user then exit.
function check_result
{
exitStatus=$?
if [ $exitStatus -ne 0 ]
then
echo "command failed with exit status $exitStatus"
echo $result
exit 1
fi
}

function create_user
{
# Check on authorization type
paramString="user create;"
if [[ $authtype = "NATIVE" && -n $userpwd ]]
then
pointParams="set authenticationType=$authtype;"
pointParams="$pointParams set credential.type=PasswordCredential; set
credential.password=$userpwd;"
elif [[ $authtype = "LDAP" && -n $principal ]]
then
pointParams="set authenticationType=$authtype; set principal=$principal;"
fi
# These are the required parameters.
paramString="$paramString set type=User; set name=$newjsuser;"

# Fill in optional parameters if there are any.
if [[ -n $firstname ]]
then
paramString="$paramString set firstName=\"$firstname\";"
fi

if [[ -n $lastname ]]
then
```

```
paramString="$paramString set lastName=\"$lastname\";"
fi

if [[ -n $emailaddress ]]
then
paramString="$paramString set emailAddress=\"$emailaddress\";"
fi

if [[ -n $homephone ]]
then
paramString="$paramString set homePhoneNumber=\"$homephone\";"
fi

if [[ -n $mobilephone ]]
then
paramString="$paramString set mobilePhoneNumber=\"$mobilephone\";"
fi

if [[ -n $workphone ]]
then
paramString="$paramString set workPhoneNumber=\"$workphone\";"
fi
paramString="$paramString ${pointParams} commit;"
#echo $paramString
echo "Creating user..."
result=$(${SSH_CMD} $paramString)
check_result
#echo $result
echo "New user $newjsuser successfully created"

##### ROLE-3 is Self-Service Role
paramString="authorization create;"
paramString="$paramString set type=Authorization; set role=ROLE-3; set
target=$newjsuser; set user=$newjsuser;commit;"
#echo $paramString
result=$(${SSH_CMD} $paramString)
check_result
echo "Assigned Self-Service Role to user $newjsuser"
}

##### Main
##### Main
while getopts "u:d:P:r:f:l:e:o:m:w:h" flag; do
case "$flag" in
u ) username=${OPTARG%:*}
;;
d ) engine=$OPTARG
;;
P ) userpwd=$OPTARG
;;
r ) principal=$OPTARG
;;
f ) firstname=$OPTARG
```

```
;;
l ) lastname=$OPTARG
;;
e ) emailaddress=$OPTARG
;;
o ) homephone=$OPTARG
;;
m ) mobilephone=$OPTARG
;;
w ) workphone=$OPTARG
;;
h ) usage
exit
;;
* ) usage
exit 1

esac
done

# Shift the parameters so we only have the positional arguments left
shift $((OPTIND-1))
# Check that there are 2 positional arguments
if [ $# != 2 ]
then
echo "usage1"
usage
exit 1
fi
# Get the two positional arguments
authtype=$1
shift
newjsuser=$1
create_session
create_user
```

CLI cookbook: how to delete a Delphix self-service bookmark

Prerequisites:

- Have Delphix Self-Service user privileges
- Know the bookmark you would like to delete

Procedure:

1. Log into Delphix Engine as a Delphix Self-Serviceuser or admin.

```
ssh jsUser@<yourengine>
```

2. Navigate to the Delphix Self-Servicebookmarks, and choose the one you would like to delete.

```
jsUser > selfservice
bookmark
jsUser selfservice bookmark > ls
jsUser selfservice bookmark > select <jsBookmark>
```

3. Delete the bookmark.

```
jsUser selfservice bookmark <jsBookmark> *> delete
jsUser selfservice bookmark <jsBookmark> delete *> commit
```

CLI cookbook: how to delete a Delphix self-service container

Prerequisites:

- Know the container you want to delete
- Have delphix_admin privileges

Procedure:

1. Log into Delphix Engine as admin

```
ssh delphix@<yourengine>
```

2. Navigate to the Delphix Self-Servicecontainer that you want to delete

```
delphix > selfservice container
delphix selfservice container > ls
delphix selfservice container > select CONTAINER_X
delphix selfservice container 'CONTAINER_X' >
```

3. Delete container, **note you need to set if you want to delete the VDBs in the container (false will preserve the VDBs and true the VDBs will be deleted along with the container)**

```
delphix selfservice container 'CONTAINER_X' > delete
delphix selfservice container 'CONTAINER_X'delete *> set deleteDataSources=<tru
e/false>
delphix selfservice container 'CONTAINER_X'delete *> commit
```

CLI cookbook: how to delete a Delphix self-service template

Prerequisites:

- Template has no dependant containers
- Know the name of the template you are going to delete
- Have delphix_admin privileges (note JetStream Only users and Delphix GUI Owners cannot delete templates)

Procedure:

1. Log into your Delphix Engine using delphix_admin (or admin privileged account)

```
ssh delphix_admin@<yourengine>
```

2. Find the template you want to delete

```
delphix > selfservice
delphix selfservice template > ls
delphix selfservice template > select 'TEMPLATE_X'
delphix selfservice template 'TEMPLATE_X'>
```

3. Delete template and commit

```
delphix selfservice template 'TEMPLATE_X'> delete
delphix selfservice template 'TEMPLATE_X' delete *> commit
```

CLI cookbook: how to refresh a Delphix self-service container

> The following script is for educational and demonstration purposes only and is not supported by Delphix.

This script can be downloaded by selecting **refreshContainer.sh**.

Refreshing a container in Delphix self-service

```
#!/bin/bash

# A sample script for calls to the CLI. This one refresh Jet Stream container.
#
# VERY IMPORTANT: In order for this to work, you need to go through the steps here:
# https://docs.delphix.com/display/DOCS43/CLI+Cookbook%3A+Configuring+Key-
Based+SSH+Authentication+for+Automation
# After this you will not need to use a username and password to log into the delphix
engine. If you do not
# setup the SSH authentication you will have to manually enter the password.
#


##### Constants

# Describes a Delphix software revision.
VERSION="1.11.4"


##### Default Values. These can be overwriten with optional arguments.
engine="de-6041-doc-938-vn.dlpxdc.co"
username="admin"


##examples##
# Refresh container from latest point in time of Template
#./refreshContainer.sh -T JS_DATA_TEMPLATE-13 JS_DATA_CONTAINER-20
# Refresh container from specific bookmark
#./refreshContainer.sh -b JS_BOOKMARK-76 JS_DATA_CONTAINER-20
# Refresh container from specific point in time of branch
#./refreshContainer.sh -t "2016-08-08T10:00:00.000Z" -B JS_BRANCH-4
 JS_DATA_CONTAINER-20


##### Functions

# Help Menu
function usage {
echo "Usage: refreshContainer.sh [[-h] | options...] <containername>"
echo "Create a Jet Stream Bookmark on the given branch."
echo ""
echo " You need to specify either -T, -B, -b or -t to refresh container. With -t
option you also need to specify -B also"
echo "Positional arguments"
```

```
echo " <containerName>"
echo ""
echo "Optional Arguments:"
echo " -h Show this message and exit"
echo " -d Delphix engine IP address or host name, otherwise revert to default"
echo " -u Server user. Password needs to manually provide at run time, otherwise
revert to default"
echo " -T template reference from which need to refresh from latest point in time"
echo " -B Branch reference from which we need to pick up time from where the
container should be refreshed. Type: string. Format JS_BRANCH-<n> (Optional)"
echo " -b Bookmark name from which need to refresh container. If no bookmark is
included, the branch will be created at the latest point in time. Type: string.
Format JS_BOOKMARK-<n> (Optional)"
echo " -t The time from where the container should be refreshed. This must be
accompanied with branch reference from which need to pick up time. Type: date, must
be in ISO 8601 extended format [yyyy]-[MM]-[dd]T[HH]:[mm]:[ss].[SSS]Z"
}

# Create Our Session, including establishing the API version.
function create_session
{
echo "Creating session ..."
SSH_CMD="ssh ${username}@${engine}"
${SSH_CMD} "version $VERSION"
check_result
}

# Check the result of the curl. If there are problems, inform the user then exit.
function check_result
{
exitStatus=$?
if [ $exitStatus -ne 0 ]
then
echo "command failed with exit status $exitStatus"
echo $result
exit 1
fi
}


function restore_container
{

paramString="selfservice/container/select ${containerRef}; restore;"

# If there is no time input and no bookmark name, we need to use
JSTimelinePointLatestTimeInput from template.
if [[ -n $template && -z $inputTime && -z $bookmark ]]
then
pointParams="edit timelinePointParameters; set type=JSTimelinePointLatestTimeInput;
set sourceDataLayout=\"${template}\";"
# If there is a timeInput and no bookmark name, we need to use Input Time.
elif [[ -n $inputTime && -n $branch && -z $bookmark && -z $template ]]
```

```
then
pointParams="edit timelinePointParameters; set type=JSTimelinePointTimeInput; set
time=\"${inputTime}\"; set branch=\"${branch}\";"
# If there is a bookmark name and no time input, we need to use bookmark
elif [[ -n $bookmark && -z $template && -z $inputTime ]]
then
pointParams="edit timelinePointParameters; set type=JSTimelinePointBookmarkInput; set
bookmark=\"${bookmark}\";"
else
usage
exit 1
fi

paramString="$paramString $pointParams commit;"
echo "Executing the command on CLI ..."
echo $paramString
result=$(${SSH_CMD} $paramString)
check_result

echo "Verifying job status ..."
# Get everything in the result that comes after job.
temp=${result#*job}
# Get rid of everything after
resultArray=($temp)
jobRef=($resultArray)
jobString="job;select $jobRef;ls"
result=$(${SSH_CMD} $jobString)
check_result
# Get everything in the result that comes after job.
temp=${result#*jobState:}
# Get rid of everything after
resultArray=($temp)
jobState=($resultArray)

if [ $jobState = "COMPLETED" ]
then
echo "Successfully refreshed container- [$containerRef]."
else
echo "Unable to refresh container- [$containerRef]."
echo result
fi

}


##### Main

while getopts "u:d:T:b:t:B:h" flag; do
case "$flag" in
u ) username=${OPTARG%:*}
;;
d ) engine=$OPTARG
;;
```

```
T ) template=$OPTARG
;;
b ) bookmark=$OPTARG
;;
t ) inputTime=$OPTARG
;;
B ) branch=$OPTARG
;;
h ) usage
exit
;;
* ) usage
exit 1
esac

done


# Shift the parameters so we only have the positional arguments left
shift $((OPTIND-1))

# Check that there are 1 positional arguments
if [ $# != 1 ]
then
usage
exit 1
fi

# Get the one positional arguments
containerRef=$1

create_session
restore_container
```

CLI cookbook: how to share a Delphix self-service bookmark

Prerequisites:

- Have Delphix Self-Service user privileges.
- Know the bookmark you would like to share.

Delphix Self-Service administrators can use this CLI cookbook recipe to share a bookmark on Delphix Self-Service using the Delphix Engine CLI.

> ⓘ The following script is for educational and demonstration purposes only and is not supported by Delphix.

This script can be downloaded by selecting **shareBookmark.sh**.

Sharing a bookmark in Delphix self-service

```
#!/bin/bash

# A sample script for calls to the CLI. This one shares Bookmark across containers in
same template.
#
# VERY IMPORTANT: In order for this to work, you need to go through the steps here:
# https://docs.delphix.com/display/DOCS43/CLI+Cookbook%3A+Configuring+Key-
Based+SSH+Authentication+for+Automation
# After this you will not need to use a username and password to log into the delphix
engine. If you do not
# setup the SSH authentication you will have to manually enter the password.
#




##examples##
# Share Bookmark
#./shareBookmark.sh -a share JS_BOOKMARK-75
# Unshare Bookmark
#./shareBookmark.sh -a unshare JS_BOOKMARK-75



##### Constants

# Describes a Delphix software revision.
VERSION="1.11.10"


##### Default Values. These can be overwriten with optional arguments.
engine="10.110.213.109"
username="admin"



##### Functions
```

```
# Help Menu
function usage {
    echo "Usage: shareBookmark.sh [[-h] | options...] -a share/unshare
<bookmarkName>"
    echo "Share/Unshare JetStream bookmark"
    echo ""
    echo "Positional arguments"
    echo "bookmarkName. Format: JS_BOOKMARK-<n>"
    echo ""
    echo "Optional Arguments:"
    echo "  -h                Show this message and exit"
    echo "  -d                Delphix engine IP address or host name, otherwise
revert to default"
    echo "  -u                Server user. Password needs to manually provide at run
time, otherwise revert to default"
    echo "  -a                action to perform on bookmark. Type:String.
Values:share/unshare"
}

# Create Our Session, including establishing the API version.
function create_session
{
    echo "creating session..."
    SSH_CMD="ssh ${username}@${engine}"
    ${SSH_CMD}  "version $VERSION"
    check_result
}


# Check the result of the curl. If there are problems, inform the user then exit.
function check_result
{
    exitStatus=$?
    if [ $exitStatus -ne 0 ]
    then
        echo "command failed with exit status $exitStatus"
        echo $result
        exit 1
    fi
}


function bookmark_action
{
    # Change share mode of bookmark
    echo "Changing share mode.."
    if [[ $action = "share" ]]
    then
        paramString="selfservice bookmark select $bookmarkName share; commit;"
    elif [[ $action = "unshare" ]]
    then
```

```
        paramString="selfservice bookmark select $bookmarkName unshare; commit;"
    else
        usage
        exit 1
    fi
    result=$(${SSH_CMD}  $paramString)
    check_result
    if [[ $action = "share" ]]
    then
        echo "Bookmark ${bookmarkName} is now in shared mode"
    elif [[ $action = "unshare" ]]
    then
        echo "Bookmark ${bookmarkName} is now in not-share mode"
    fi
}


##### Main

while getopts "u:d:a:h" flag; do
    case "$flag" in
        u )             username=${OPTARG%:*}
                        ;;
        d )             engine=$OPTARG
                        ;;
        a )             action=$OPTARG
                        ;;
        h )             usage
                        exit
                        ;;
        * )             usage
                        exit 1
    esac

done


# Shift the parameters so we only have the positional arguments left
shift $((OPTIND-1))

# Check that there is 1 positional arguments
if [ $# != 1 ]
then
    usage
    exit 1
fi

# Get the one positional arguments
bookmarkName=$1

create_session
bookmark_action
```

CLI cookbook: how to update a Delphix self-service bookmark

Prerequisites:

- Have Delphix Self-Service user privileges
- Know the bookmark you would like to update

Procedure:

1. Log into Delphix Engine as a Delphix Self-Serviceuser or admin.

```
ssh jsUser@<yourengine>
```

2. Navigate to the Delphix Self-Servicebookmarks, and choose the one you would like to update.

```
jsUser > selfservice bookmark
jsUser selfservice bookmark > ls
jsUser selfservice bookmark > select <jsBookmark>
```

3. Update the bookmark.

```
jsUser selfservice bookmark <jsBookmark> *> update
jsUser selfservice bookmark <jsBookmark> update *> set tags="tag text"
jsUser selfservice bookmark <jsBookmark> update *> commit
```

CLI cookbook: hooks and hook templates

These topics describe the command-line interface procedures for working with hooks and hook templates.

This section covers the following topics:

- CLI cookbook: changing the PowerShell version associated with a hook template

CLI cookbook: changing the PowerShell version associated with a hook template

This article is to document how to change the PowerShell version associated with a hook template using the CLI.

> This operation is not allowed through GUI.

Procedure:

1. Login into Delphix Engine CLI using "admin" credentials.

```
ssh admin@<delphix_engine>
```

2. Go to **source** and then to **operationTemplate**.

```
delphix> source
delphix source> operationTemplate
```

3. Execute the **ls**command to see the existing hooks template and select one of the existing hook templates which you want to update.

```
delphix source operationTemplate> ls
Objects
NAME                DESCRIPTION                 LASTUPDATED
OPERATION.TYPE
Test_Template       Test_Template               2020-06-12T04:15:01.759Z
RunDefaultPowerShellOnSourceOperation

delphix source operationTemplate> select Test_Template

delphix source operationTemplate 'Test_Template'> ls
Properties
    type: OperationTemplate
    name: Test_Template
    description: Test_Template
    lastUpdated: 2020-06-12T04:15:01.759Z
    operation:
        type: RunDefaultPowerShellOnSourceOperation
        name: Test_Template
        command: echo_command
    reference: OPERATION_TEMPLATE-1
```

4. Run the **update** command and change the hook template type from **RunDefaultPowerShellOnSourceOperation** to **RunPowerShellOnSourceOperation**.

> Users can also change the template type from RunPowerShellOnSourceOperation to RunDefaultPowerShellOnSourceOperation.

```
delphix source operationTemplate 'Test_Template'> update
```

```
delphix source operationTemplate 'Test_Template' update *> set
operation.type=RunPowerShellOnSourceOperation
delphix source operationTemplate 'Test_Template' update *> ls
Properties
    type: OperationTemplate
    name: Test_Template
    description: Test_Template
    operation:
        type: RunPowerShellOnSourceOperation (*)
        name: Test_Template
        command: echo_command
```

5. Run the **commit** command to perform the update on the hook template. Execute the **ls** command again to see the change in the hook template type.

```
delphix source operationTemplate 'Test_Template' update *> commit

delphix source operationTemplate 'Test_Template'> ls
Properties
    type: OperationTemplate
    name: Test_Template
    description: Test_Template
    lastUpdated: 2020-06-12T04:30:44.203Z
    operation:
        type: RunPowerShellOnSourceOperation
        name: Test_Template
        command: echo_command
    reference: OPERATION_TEMPLATE-1
```

1. Template type can only be changed from "RunDefaultPowerShellOnSourceOperation" to "RunPowerShellOnSourceOperation" or vice versa. You **can not** change the type to other available types. If you try to do that, you will get this error: "Error: The type of this operation (RUN_POWERSHELL_ON_SOURCE_OPERATION) cannot be modified after creation.".
2. Template type "RunPowerShellOnSourceOperation" denotes "PowerShell version 2" and "RunDefaultPowerShellOnSourceOperation" denotes "Default PowerShell version". Here, default version is the version of PowerShell installed on the target host.

CLI cookbook: network performance

These topics describe command-line interface procedures using the iperf-based Network Performance Tool:

- CLI cookbook: Delphix session protocol test from primary engine to replication engine
- CLI cookbook: running the network test via CLI - latency
- CLI cookbook: running the network test via CLI - throughput

CLI cookbook: Delphix session protocol test from primary engine to replication engine

Prerequisites

The network performance tool measures network performance between a Delphix Engine and an environment host. You must have added an environment in order to use this tool.

This transmission control protocol (TCP) throughput test uses TCP port 50001 by default. The port can also be configured on a per-test-run basis. For the duration of a given throughput test, a server on the receiver will be listening on this port. For a transmit test, the receiver is the remote host; for a receive test, the receiver is the Delphix Engine

Procedure

Delphix uses the Delphix Session Protocol (DSP) protocol to communicate between primary and replication engines.

1. Login to Delphix Engine using a Delphix administrator account such as **delphix_admin.** As soon as you login, you will get prompt with engine name.

```
-bash-4.3$ ssh delphix_admin@delphix
Password:
delphix>
```

2. Create a DSP test.

```
delphix> network test dsp create
```

3. Set the destinationType to DELPHIX_ENGINE to do a DSP test between two Delphix Engines. The default is REMOTE_HOST which executes a DSP test between a source or target host and the Delphix Engine.

```
delphix network test dsp create *> set destinationType=DELPHIX_ENGINE
```

4. Use **get** to see other optional arguments. Modify the test parameters as needed and commit to start the test.

```
delphix network test dsp create *> get
   type: NetworkDSPTestParameters
   blockSize: 64KB
   compression: false
   destinationType: REMOTE_HOST
   direction: TRANSMIT
   duration: 30
   encryption: false
   numConnections: 0
   queueDepth: 32
   receiveSocketBuffer: 256KB
   remoteDelphixEngineInfo: (unset)
   remoteHost: (unset)
   sendSocketBuffer: 256KB
delphix network test dsp create *> set remoteDelphixEngineInfo.address=delphix2
```

```
delphix network test dsp create *> set
remoteDelphixEngineInfo.principal=delphix_admin
delphix network test dsp create *> edit remoteDelphixEngineInfo
delphix network test dsp create remoteDelphixEngineInfo *> get
   type: RemoteDelphixEngineInfo (*)
   address: delphix2 (*)
   credential: (required)
   principal: delphix_admin (*)
delphix network test dsp create remoteDelphixEngineInfo *> set
credential.password=delphix
delphix network test dsp create remoteDelphixEngineInfo *> commit
   `NETWORK_DSP_TEST-2
   Dispatched job JOB-8
   NETWORK_DSP_TEST_EXECUTE job started.
   Measuring throughput with variable number of connections: 716 Mbps measured
for 1 connections.
   Measuring throughput with variable number of connections: 711 Mbps measured
for 2 connections.
   Measuring throughput with variable number of connections: 611 Mbps measured
for 4 connections.
   Measuring throughput with variable number of connections: 646 Mbps measured
for 6 connections.
   Measuring throughput with variable number of connections: 567 Mbps measured
for 8 connections.
   Measuring average throughput for 30 seconds with 1 connections.
   Measured throughput of 408 Mbps.
   NETWORK_DSP_TEST_EXECUTE job completed successfully.
```

5. Retrieve the test results.

```
delphix> network test dsp list
NAME                              PARAMETERS.DIRECTION  STATE      THROUGHPUT
delphix2-2018-01-23T21:25:31.172Z  TRANSMIT              COMPLETED  880.5Mbps
delphix2-2018-02-02T17:54:58.322Z  TRANSMIT              COMPLETED  408.5Mbps
```

CLI cookbook: running the network test via CLI - latency

Prerequisites

The network performance tool measures network performance between a Delphix Engine and an environment host. You must have added an environment in order to use this tool.

This transmission control protocol (TCP) throughput test uses TCP port 50001 by default. The port can also be configured on a per-test-run basis. For the duration of a given throughput test, a server on the receiver will be listening on this port. For a transmit test, the receiver is the remote host; for a receive test, the receiver is the Delphix Engine.

Procedure

The network latency test measures network round-trip latency by transmitting ICMP echo requests (like the ping utility) and measuring the time to receive replies from the remote host. To execute a test:

1. Login as a domain user to the Delphix Engine CLI using ssh.
2. Create a network latency test.

```
delphix> network test latency
delphix network test latency> create
delphix network test latency create *>
```

3. You must set **remoteHost** to the name of an environment host already configured in the Delphix Engine. (You should press tab after the "=" (equal sign) to auto-populate and confirm registered destinations). Use 'get' to see other optional arguments. Modify the test parameters as needed and **commit** to start the test.

```
delphix network test latency create *> set remoteHost=oracletarget
delphix network test latency create *> get
    type: NetworkLatencyTestExecuteParameters
    remoteHost: oracletarget
    requestCount: 20
    requestSize: 8B
delphix network test latency create *> commit
    Dispatched job JOB-20
    NETWORK_LATENCY_TEST_EXECUTE job started for
"oracletarget-2014-06-20T18:57:28.659Z".
    Executing network latency test.
    NETWORK_LATENCY_TEST_EXECUTE job for
"oracletarget-2014-06-20T18:57:28.659Z" completed successfully.
```

4. The job will be submitted and visible in the Delphix Management application.
5. Retrieve the test results. All times are in microseconds.

```
delphix network test latency> list
NAME                                  AVERAGE
oraclesource-2014-06-20T18:57:28.659Z  872
delphix network test latency> select oraclesource-2014-06-20T18:57:28.659Z
delphix network test latency "oraclesource-2014-06-20T18:57:28.659Z"> get
    type: NetworkLatencyTest
```

```
    name: oraclesource-2014-06-20T18:57:28.659Z
    average: 872
    endTime: 2014-06-20T18:57:48.558Z
    loss: 0
    maximum: 2755
    minimum: 294
    reference: NETWORK_LATENCY_TEST-2
    remoteAddress: 172.16.203.184
    remoteHost: oraclesource
    requestCount: 20
    requestSize: 8B
    startTime: 2014-06-20T18:57:28.659Z
    stddev: 527
```

CLI cookbook: running the network test via CLI - throughput

Prerequisites

The network performance tool measures network performance between a Delphix Engine and an environment host. You must have added an environment in order to use this tool.

This transmission control protocol (TCP) throughput test uses TCP port 50001 by default. The port can also be configured on a per-test-run basis. For the duration of a given throughput test, a server on the receiver will be listening on this port. For a transmit test, the receiver is the remote host; for a receive test, the receiver is the Delphix Engine.

Procedure

The network throughput test measures sustained throughput using a synthetic workload to or from a remote host. To execute a test:

1. Login as a domain user to the **Delphix Engine CLI** using ssh.
2. Create a network throughput test.

```
delphix> network test throughput
delphix network test throughput> create
delphix network test throughput create *>
```

3. You must set **remoteHost** to the name of an environment host already configured in the Delphix Engine. Use 'get' to see other optional arguments. Modify the test parameters as needed and **commit** to start the test.

```
delphix network test throughput create *> set remoteHost=oraclesource
delphix network test throughput create *> ls
Properties
    type: NetworkThroughputTestParameters
    blockSize: 128KB
    direction: TRANSMIT
    duration: 30
    numConnections: 0
    port: 50001
    receiveSocketBuffer: 4MB
    remoteHost: oraclesource
    sendSocketBuffer: 4MB
delphix network test throughput create *> commit
    Dispatched job JOB-21
    NETWORK_THROUGHPUT_TEST_EXECUTE job started for
"oraclesource-2014-06-20T19:30:12.566Z".
    Executing network throughput transmit test.
    Measuring throughput with variable number of connections: 1.
    Measuring throughput with variable number of connections: 2.
    Measuring throughput with variable number of connections: 4.
    Measuring throughput with variable number of connections: 6.
    Measuring throughput with variable number of connections: 8.
    Measuring maximum sustained throughput for 30 seconds with 8 connections.
    NETWORK_THROUGHPUT_TEST_EXECUTE job for
"oraclesource-2014-06-20T19:30:12.566Z" completed successfully.
```

4. The job will be submitted and visible in the Delphix Management application.

```
delphix network test throughput> list
NAME                                  DIRECTION    STATE     THROUGHPUT
oraclesource-2014-06-20T19:30:12.566Z  TRANSMIT     COMPLETED 695.6Mbps
delphix network test throughput> select oraclesource-2014-06-20T19:30:12.566Z
delphix network test throughput "oraclesource-2014-06-20T19:30:12.566Z"> get
    type: NetworkThroughputTest
    name: oraclesource-2014-06-20T19:30:12.566Z
    endTime: 2014-06-20T19:31:15.041Z
    numConnections: 8
    parameters:
        type: NetworkThroughputTestParameters
        blockSize: 128KB
        direction: TRANSMIT
        duration: 30
        numConnections: 0
        port: 50001
        receiveSocketBuffer: 4MB
        remoteHost: oraclesource
        sendSocketBuffer: 4MB
    reference: NETWORK_THROUGHPUT_TEST-2
    remoteAddress: 172.16.203.184
    startTime: 2014-06-20T19:30:12.566Z
    state: COMPLETED
    throughput: 695.6Mbps
```

Kerberos CLIs

The following topics provide information related to the Kerberos CLIs.

- CLI cookbook: admin application configuration
- CLI cookbook system app configuration

CLI cookbook: admin application configuration

The following are done after logging into the Delphix admin app as `admin@<delphix engine hostname>.`

Add Kerberos environment

```
jkb-5160.dcenter> cd /environment/
jkb-5160.dcenter environment> ls
Children
oracle
user
windows

Operations
create
jkb-5160.dcenter environment> create
jkb-5160.dcenter environment create *> ls
Properties
   type: HostEnvironmentCreateParameters
   hostEnvironment:
       type: UnixHostEnvironment
       name: (unset)
       aseHostEnvironmentParameters: (unset)
       description: (unset)
   hostParameters:
       type: UnixHostCreateParameters
       host:
           type: UnixHost
           address: (required)
           privilegeElevationProfile: (unset)
           sshPort: 22
           toolkitPath: (required)
           truststorePassword: (unset)
   primaryUser:
       type: EnvironmentUser
       name: (unset)
       credential:
           type: PasswordCredential
           password: (required)
       environment: (unset)
       groupId: (unset)
       userId: (unset)
jkb-5160.dcenter environment create *> set hostParameters.host.address=ln-rh64-
tgt.dc2.delphix.com
jkb-5160.dcenter environment create *> set hostParameters.host.toolkitPath=/tmp
jkb-5160.dcenter environment create *> set
primaryUser.credential.type=KerberosCredential
jkb-5160.dcenter environment create *> commit
   `UNIX_HOST_ENVIRONMENT-2
   Dispatched job JOB-4
   ENVIRONMENT_CREATE_AND_DISCOVER job started for "ln-rh64-tgt.dc2.delphix.com".
```

```
   ENVIRONMENT_CREATE_AND_DISCOVER job for "ln-rh64-tgt.dc2.delphix.com" completed
successfully.
jkb-5160.dcenter environment>
jkb-5160-2.dcenter environment> ls
Objects
NAME                          DESCRIPTION
ln-rh64-tgt.dc2.delphix.com   -

Children
oracle
user
windows

Operations
Create
jkb-5160-2.dcenter environment> select ln-rh64-tgt.dc2.delphix.com
jkb-5160-2.dcenter environment 'ln-rh64-tgt.dc2.delphix.com'> ls
Properties
   type: UnixHostEnvironment
   name: ln-rh64-tgt.dc2.delphix.com
   aseHostEnvironmentParameters: (unset)
   description: (unset)
   enabled: true
   host: ln-rh64-tgt.dc2.delphix.com
   primaryUser: sybase
   reference: UNIX_HOST_ENVIRONMENT-1

Operations
delete
update
disable
enable
refresh
jkb-5160-2.dcenter environment 'ln-rh64-tgt.dc2.delphix.com'>
```

Add Kerberos ASE instance

```
jkb-5160-2.dcenter environment> cd /repository/
jkb-5160-2.dcenter repository> ls
Children
template

Operations
create
compatibleRepositories
jkb-5160-2.dcenter repository> create
jkb-5160-2.dcenter repository create *> ls
Properties
   type: ASEInstance
   credentials: (unset)
   dbUser: (unset)
```

```
   environment: (required)
   installationPath: (required)
   instanceName: (required)
   instanceOwner: (required)
   ports: (required)
   servicePrincipalName: (unset)
   version: (unset)
jkb-5160-2.dcenter repository create *> set credentials.type=KerberosCredential
jkb-5160-2.dcenter repository create *> set environment=ln-rh64-tgt.dc2.delphix.com
jkb-5160-2.dcenter repository create *> set installationPath=/opt/sybase/15-0
jkb-5160-2.dcenter repository create *> set ports=5100
jkb-5160-2.dcenter repository create *> set instanceName=ASE1570_S1
jkb-5160-2.dcenter repository create *> set instanceOwner=sybase
jkb-5160-2.dcenter repository create *> set servicePrincipalName=ASE1570_S1
jkb-5160-2.dcenter repository create *> commit
   `ASE_INSTANCE-1
jkb-5160-2.dcenter repository> ls
Objects
NAME        VERSION ENVIRONMENT
ASE1570_S1  15.7 SP138 ln-rh64-tgt.dc2.delphix.com

Children
template

Operations
create
compatibleRepositories
jkb-5160-2.dcenter repository> select ASE1570_S1
jkb-5160-2.dcenter repository 'ASE1570_S1'> ls
Properties
   type: ASEInstance
   name: ASE1570_S1
   credentials:
       type: KerberosCredential
   dbUser: sybase
   discovered: false
   environment: ln-rh64-tgt.dc2.delphix.com
   installationPath: /opt/sybase/15-0
   instanceName: ASE1570_S1
   instanceOwner: sybase
   instanceOwnerGid: 500
   instanceOwnerUid: 500
   internalVersion: 15.7 SP138
   linkingEnabled: true
   pageSize: 4096
   ports: 5100
   provisioningEnabled: true
   reference: ASE_INSTANCE-1
   servicePrincipalName: ASE1570_S1
   staging: false
   version: 15.7 SP138
```

```
Operations
delete
update
jkb-5160-2.dcenter repository 'ASE1570_S1'>
```

Link a dSource

```
jkb-5160-2.dcenter> cd /database
jkb-5160-2.dcenter database> ls
Children
template

Operations
createEmpty
createRestorationDataset
export
fileMapping
link
oracleSupportedCharacterSets
provision
validateXpp
xpp
jkb-5160-2.dcenter database> link
jkb-5160-2.dcenter database link *> ls
Properties
   type: LinkParameters
   name: (required)
   description: (unset)
   group: (required)
   linkData:
       type: ASELinkData
       config: (required)
       dbCredentials:
           type: PasswordCredential
           password: (required)
       dbUser: (unset)
       dumpCredentials: (unset)
       externalFilePath: (unset)
       loadBackupPath: (required)
       loadLocation: (unset)
       operations: (unset)
       sourceHostUser: (required)
       sourcingPolicy: (unset)
       stagingHostUser: (required)
       stagingPostScript: (unset)
       stagingPreScript: (unset)
       stagingRepository: (required)
       syncParameters:
           type: ASELatestBackupSyncParameters
       validatedSyncMode: ENABLED
```

```
Operations
defaults
jkb-5160-2.dcenter database link *> set name=test1
jkb-5160-2.dcenter database link *> set group=Untitled
jkb-5160-2.dcenter database link *> set linkData.config=test1
jkb-5160-2.dcenter database link *> set
linkData.dbCredentials.type=KerberosCredential
jkb-5160-2.dcenter database link *> set linkData.loadBackupPath=/home/sybase/db
jkb-5160-2.dcenter database link *> set linkData.stagingRepository=ASE1570_S1
jkb-5160-2.dcenter database link *> set linkData.sourceHostUser=sybase
jkb-5160-2.dcenter database link *> set linkData.stagingHostUser=sybase
jkb-5160-2.dcenter database link *> set
linkData.syncParameters.type=ASENewBackupSyncParameters
jkb-5160-3.dcenter database link linkData syncParameters *> commit
   `ASE_DB_CONTAINER-1
   Dispatched job JOB-4
   DB_LINK job started for "Untitled/test1".
   DB_LINK job for "Untitled/test1" completed successfully.
jkb-5160-2.dcenter database>
jkb-5160-2.dcenter database> ls
Objects
NAME   PROVISIONCONTAINER  DESCRIPTION
test1  -             -

Children
template

Operations
createEmpty
createRestorationDataset
export
fileMapping
link
oracleSupportedCharacterSets
provision
validateXpp
xpp
jkb-5160-2.dcenter database>
```

CLI cookbook system app configuration

The following are done after logging into the Delphix System configuration application as `sysadmin@<delphix_engine_hostname;.`

Define the system Kerberos configuration

```
jkb-5160.dcenter> cd /service/kerberos/
jkb-5160.dcenter service kerberos> ls
Properties
   type: KerberosConfig
   enabled: false
   kdcs:
       0:
           type: KerberosKDC
           hostname:
           port: 88
   keytab: (unset)
   principal:
   realm:

Operations
update
reset
jkb-5160.dcenter service kerberos> update
jkb-5160.dcenter service kerberos update *> set realm=DELPHIX.COM
jkb-5160.dcenter service kerberos update *> set principal=sybase
jkb-5160.dcenter service kerberos update *> set
keytab=BQIAAABEAAEAC0RFTFBISVguQ09NAAZzeWJhc2UAAAABWNVqQgMAEgAgjF/
zgNuw27Uy9vEgvPvpeevAaAw6c5HaVAWVSLZnmngAAAA0AAEAC0RFTFBISVguQ09NAAZzeWJhc2UAAAABWNVq
QgMAEQAQTsNZOXmHhaeuXPeyQRXelAAAADwAAQALREVMUEhJWC5DT00ABnN5YmFzZQAAAAFY1WpCAwAQABjHS
VHfC+p8yIYfgFTIv1T7m99n2Spn7wIAAAA0AAEAC0RFTFBISVguQ09NAAZzeWJhc2UAAAABWNVqQgMAFwAQhJ
PmRokSfz310fe3eVUD3wAAACwAAQALREVMUEhJWC5DT00ABnN5YmFzZQAAAAFY1WpCAwAIAAhiFUmzFjttmwA
AACwAAQALREVMUEhJWC5DT00ABnN5YmFzZQAAAAFY1WpCAwADAAj4rgff+Au/ng==
jkb-5160.dcenter service kerberos update *> edit kdcs
jkb-5160.dcenter service kerberos update kdcs *> edit 0
jkb-5160.dcenter service kerberos update kdcs 0 *> set hostname=kerberos-01.delphix.c
om
jkb-5160.dcenter service kerberos update kdcs 0 *> back
jkb-5160.dcenter service kerberos update kdcs *> back
jkb-5160.dcenter service kerberos update *> ls
Properties
   type: KerberosConfig
   kdcs:
       0:
           type: KerberosKDC (*)
           hostname: kerberos-01.delphix.com (*)
           port: 88 (*)
   keytab: ******** (*)
   principal: sybase (*)
   realm: DELPHIX.COM (*)
```

```
jkb-5160.dcenter service kerberos update *> commit
jkb-5160.dcenter service kerberos> ls
Properties
   type: KerberosConfig
   enabled: true
   kdcs:
       0:
           type: KerberosKDC
           hostname: kerberos-01.delphix.com
           port: 88
   keytab: (unset)
   principal: sybase
   realm: DELPHIX.COM

Operations
update
reset
jkb-5160.dcenter service kerberos>
```

Clear Kerberos configuration

```
jkb-5160.dcenter service kerberos> reset
jkb-5160.dcenter service kerberos reset *> commit
jkb-5160.dcenter service kerberos> ls
Properties
   type: KerberosConfig
   enabled: false
   kdcs:
       0:
           type: KerberosKDC
           hostname:
           port: 88
   keytab: (unset)
   principal:
   realm:

Operations
update
reset
jkb-5160.dcenter service kerberos>
```

Web services API guide

These topics describe interfacing with the public web service APIs, building automation facilities and integrating with third-party orchestration tools.

This section covers the following topics:

- API version information
- Web service object model
- Web service protocol
- CLI to web services translation
- GUI API mapping
- CLI to Python transition
- Python Cookbook: adding a UNIX host
- So you want to work with Delphix APIs?
- API Cookbook: common tasks, workflows, and examples
- Kerberos APIs

API version information

This topic describes API version information for each release of the Delphix Engine, including schema changes and links to the relevant version of the schema.

Delphix engine version	API version	Link to schema within the appliance
5.0.0.x - 5.0.3.x	1.7.0	http://<engine-address>/api/json/versions/1.7.0/delphix.json
5.0.4.x	1.7.1	http://<engine-address>/api/json/versions/1.7.1/delphix.json
5.1.0.x - 5.1.2.x	1.8.0	http://<engine-address>/api/json/versions/1.8.0/delphix.json
5.1.3.x - 5.1.5.x	1.8.1	http://<engine-address>/api/json/versions/1.8.1/delphix.json
5.1.6.0	1.8.2	http://<engine-address>/api/json/versions/1.8.2/delphix.json
5.2.0.0 - 5.2.2.1	1.9.0	http://<engine-address>/api/json/versions/1.9.0/delphix.json
5.2.3.0 - 5.2.3.1	1.9.1	http://<engine-address>/api/json/versions/1.9.1/delphix.json
5.2.4.0	1.9.2	http://<engine-address>/api/json/versions/1.9.2/delphix.json
5.2.5.0 - 5.2.6.2	1.9.3	http://<engine-address>/api/json/versions/1.9.3/delphix.json
5.3.0.0 - 5.3.0.3	1.10.0	http://<engine-address>/api/json/versions/1.10.0/delphix.json
5.3.1.0	1.10.1	http://<engine-address>/api/json/versions/1.10.1/delphix.json
5.3.2.0	1.10.2	http://<engine-address>/api/json/versions/1.10.2/delphix.json
5.3.3.0 - 5.3.3.1	1.10.3	http://<engine-address>/api/json/versions/1.10.3/delphix.json
5.3.4.0	1.10.4	http://<engine-address>/api/json/versions/1.10.4/delphix.json
5.3.5.0	1.10.5	http://<engine-address>/api/json/versions/1.10.5/delphix.json
5.3.6.0-5.3.9.0	1.10.6	http://<engine-address>/api/json/versions/1.10.6/delphix.json
6.0.0.0	1.11.0	http://<engine-address>/api/json/versions/1.11.0/delphix.json

Delphix engine version	API version	Link to schema within the appliance
6.0.1.0	1.11.1	http://<engine-address>/api/json/versions/1.11.1/delphix.json
6.0.2.0	1.11.2	http://<engine-address>/api/json/versions/1.11.2/delphix.json
6.0.3.0	1.11.3	http://<engine-address>/api/json/versions/1.11.3/delphix.json
6.0.4.0	1.11.4	http://<engine-address>/api/json/versions/1.11.4/delphix.json
6.0.5.0	1.11.5	http://<engine-address>/api/json/versions/1.11.5/delphix.json
6.0.6.0	1.11.6	http://<engine-address>/api/json/versions/1.11.6/delphix.json
6.0.7.0	1.11.7	http://<engine-address>/api/json/versions/1.11.7/delphix.json
6.0.8.0	1.11.8	http://<engine-address>/api/json/versions/1.11.8/delphix.json
6.0.9.0	1.11.9	http://<engine-address>/api/json/versions/1.11.9/delphix.json
6.0.10.0	1.11.10	http://<engine-address>/api/json/versions/1.11.10/delphix.json
6.0.11.0	1.11.11	http://<engine-address>/api/json/versions/1.11.11/delphix.json
6.0.12.0	1.11.12	http://<engine-address>/api/json/versions/1.11.12/delphix.json
6.0.13.0	1.11.13	http://<engine-address>/api/json/versions/1.11.13/delphix.json
6.0.14.0	1.11.14	http://<engine-address>/api/json/versions/1.11.14/delphix.json
6.0.15.0	1.11.15	http://<engine-address>/api/json/versions/1.11.15/delphix.json
6.0.16.0	1.11.16	http://<engine-address>/api/json/versions/1.11.16/delphix.json

Delphix engine version	API version	Link to schema within the appliance
6.0.17.0	1.11.17	http://<engine-address>/api/json/versions/1.11.17/delphix.json
7.0.0.0	1.11.18	http://<engine-address>/api/json/versions/1.11.18/delphix.json
8.0.0.0	1.11.19	http://<engine-address>/api/json/versions/1.11.19/delphix.json
9.0.0.0	1.11.20	http://<engine-address>/api/json/versions/1.11.20/delphix.json
10.0.0.0	1.11.21	http://<engine-address>/api/json/delphix.json

For more details see API Changes

Web service object model

This topic describes the Delphix object model as exported over the web services.

Object types

All objects in the Delphix API are "typed objects." All such objects have a `type` field that indicates the type of the object and its associated semantics. This allows for object inheritance and polymorphism without requiring separate APIs for each type and allows generic client-specific semantic encoding and decoding without having to be aware of the context. This means that even APIs that operate only a specific type (such as the `Group` API) still require a type field to be specified as part of the input, and will continue to report the type of objects when listing or retrieving objects. This allows the APIs to evolve in a backward-compatible fashion through the introduction of new types.

Certain "root" object types (groups, containers, sources, etc) have an associated API. This API is rooted at a particular point under `/resources/json/delphix`, but all APIs follow a standard format beneath this namespace. The CLI namespace is a direct reflection of this URL, and the API reference has an index both by object type as well as by object (CLI) path. These APIs may operate on different extended types (such as `OracleSIConfig` and `OracleRACConfig`), but the base operations remain the same regardless of input type.

Object references

Some objects returned by the APIs are pure typed objects, in that they don't represent the persistent state on the Delphix Engine but are rather calculated and returned upon request. Most of the objects in the system, however, are "persistent objects." Persistent objects (of type `PersistentObject`) have a stable reference that uniquely identifies the object on the system. This reference is separate from its name so that objects can be renamed without affecting the programmatic API. More information about object names and how they can be represented generically can be found in the CLI documentation. Object references are opaque tokens; while they can be stored and reused for later use, and interpretation of their contents is unstable and may break in a future release.

The Delphix object hierarchy is stitched together by these object references. When fetching an object that refers to another object, the member will be returned as a reference, rather than being inserted directly within the parent object. This allows consumers to control when and how links are resolved and make it clear when an object may change independently from its parent. The per-object APIs outlined below all operate on object references.

Note that some Delphix objects are singleton objects, and there is only one such object on the system. These objects do not have references because the API URL uniquely identifies the object on the system.

API operations

All APIs optionally support the following operations:

- `CREATE` - Create a new instance of the given object type. This is used for most objects, but more complicated objects, such as dSources and VDBs, must be created through a dedicated `link` or `provision` operation. The input to this operation is typically the object itself, though some objects may have specialized parameters used during the creation of objects. An example of this is `HostEnvironmentCreateParameters`.
- `UPDATE` - Update properties of the given object, specified as an object reference in the URL.

- `DELETE` - Delete a particular object, specified as an object reference in the URL. These operations are typically done as HTTP `DELETE` operations, but it is also possible to do a `POST` operation to the `/delete` API to do the same thing. The `POST` form allows for delete-specific parameters, such as `OracleDeleteParameters`.
- `GET` - Get the properties for a particular object, specified as an object reference in the URL.
- `LIST` - List all objects of the given type within the system. These APIs typically take optional query parameters that allow the set of results to be constrained, filtered, paginated, or sorted.

In addition, the following non-CRUD operations may be supported:

- **Root Operation** - A `POST or GET` operation to the root of an API namespace, not associated with a particular object. This can be used for singleton objects, such as `NDMPConfig`, operations that create objects, such as `link`, and operations that operate on multiple objects at once.
- **Per-object Operation** - A `POST` operation to a particular object reference. These operations are typically read-write but are not required to be so. These would include read-only operations that cannot be modeled as CRUD operations or require complex input use per-object operations.

Database object mModel

In order to support a wide variety of databases and database configurations, the database object model is more complex than it may initially appear after having used the Delphix Management application. For example, there is no such thing as a "dSource" or "VDB" object, only data "containers" with attached "sources". More information about how Database objects are modeled within Delphix can be found in the CLI documentation

Asynchronous execution

All APIs are designed to be transactionally safe and "quick." However, there are operations that may take a long period of time or may need to reach out to external hosts or databases such that they cannot be done safely within the context of a single API call. Such operations will dispatch a `Job` to handle asynchronous execution of the operation. **Any API** can potentially spawn a job, and which APIs spawn jobs and which do not may differ between object types or releases. If you are developing a full-featured automation system, it is recommended that you build a generic infrastructure to handle job monitoring, rather than encoding the behavior of particular APIs that may change over time.

Every operation, except for `LIST` and `GET`, which are guaranteed to be read-only, can potentially spawn a job. This is represented by the `job` field of the `APIResult` object. If this field is `null`, then the action can be completed within the bounds of the API call. Otherwise, a reference to a dispatched job is returned.

Jobs can spawn other jobs for especially complex operations, such as provisioning to an Oracle cluster environment. The job returned in the API invocation is the root job, and overall success or failure of the operation is determined by the state of this job. Some operations may succeed even if a child job fails. An example would be provisioning to an Oracle cluster where one node failed, but others were successful.

Progress can be monitored by examining the `JobEvent` objects in the `Job` object returned through the job API.

Web service protocol

This topic describes an overview of the web service API and available facilities.

Introduction

The Delphix Engine provides a set of public stable web service APIs (application programming interfaces). The web services form the basis upon with the GUI and CLI are built, and are designed to be **public** and **stable**. This guide covers the basic operation of the protocol, concepts, and considerations when building layered infrastructure. It is not a reference for all available APIs. For more information on available APIs, go to the '/api' URL of a Delphix appliance, which will provide a complete reference of all available APIs for the version of Delphix running on that system.

```
http://<server>/api
```

The CLI is a thin veneer over the web services. If you are new to the web services, it is recommended you first test out operations with the CLI, and use the `setopt trace=true` option to dump the raw data being sent and received to see the API in action.

Protocol operation

The Delphix web services are a RESTful API with loose CRUD semantics using JSON encoding.

The following HTTP methods are supported:

- `GET` - Retrieve data from the server where complex input is not needed. All `GET` requests are guaranteed to be read-only, but not all read-only requests are required to use `GET`. Simple input (strings,number, boolean values) can be passed as query parameters.
- `POST` - Issue a read/write operation, or make a read-only call that requires complex input. The optional body of the call is expressed as JSON.
- `DELETE` - Delete an object on the system. For languages that don't provide a native wrapper for `DELETE`, or for delete operations with optional input, all delete operations can also be invoked as `POST` to the same URL with `/delete` appended to it.

Regardless of the operation, the result is returned as a JSON encoded result, the contents of which are covered below. For example, the following invocation of curl demonstrates establishing a new Session (pretty-printing the result):

```
$ curl -s -X POST -k --data @- http://delphix-server/resources/json/delphix/session \
-c ~/cookies.txt -H "Content-Type: application/json" <<EOF{   "type": "APISession",
"version": {       "type": "APIVersion",       "major": 1,       "minor": 4,
"micro": 3   }}EOF{   "status":"OK",   "result": {       "type":"APISession",
"version": {           "type": "APIVersion",           "major": 1,           "minor":
4,           "micro": 3       },       "locale": "en_US",       "client": null   },
"job": null}EOF
```

NOTE: It is generally recommended to set the API session version to the highest level supported by your Delphix Engine.

Session establishment

Login involves establishing a session and then authenticating to the Delphix Engine. Only authenticated users can access web APIs. Each user must establish a session prior to making any other API calls. This is done by sending a `Session` object to the URL `/resources/json/delphix/session`. This session object will specify the `APIVersion` to use for communication between the client and server. If the server doesn't support the version requested due to an incompatible change reflected in the API major version number, an error will be returned.

The resulting session object encodes the native server version, which can be different than the version requested by the client. If the server is running a more recent but compatible version, any translation of input and output to the native version is handled automatically. More information on versioning can be found in the documentation for the `APIVersion` object within the API reference on a Delphix system. If the client supports multiple versions, the appropriate type can be negotiated by trying to establish a session with each major version supported and then inspecting the version returned.

The session will also return an identifier through browser cookies that can be reused in subsequent calls to use the same session credentials and state without having to re-authenticate. The format of this cookie is private to the server and may change at any point. Sessions do not persist across a server restart or reboot. The mechanism by which this cookie is preserved and sent with subsequent requests is client-specific. The following demonstrates invoking the session login API call using curl and storing the cookies in the file `~/cookies.txt` for later use:

```
$ curl -s -X POST -k --data @- http://delphix-server/resources/json/delphix/session \
-c ~/cookies.txt -H "Content-Type: application/json" <<EOF{   "type": "APISession",
"version": {       "type": "APIVersion",       "major": 1,       "minor": 4,
"micro": 3   }}EOF{   "status":"OK",   "result": {       "type":"APISession",
"version": {           "type": "APIVersion",           "major": 1,           "minor":
4,           "micro": 3       },       "locale": "en_US",       "client": null   },
"job": null}EOF
```

Authentication

Once the session has been established, the next step is to authenticate to the server by executing the `LoginRequest` API. Unauthenticated sessions are prohibited from making any API calls other than this login request. The username can be either a system user or domain user, and the backend will authenticate using the appropriate method. This example illustrates logging in via curl using cookies created when the session was established:

```
$ curl -s -X POST -k --data @- http://delphix-server/resources/json/delphix/login \-b
cookies.txt -c cookies2.txt -H "Content-Type: application/json" <<EOF{"type":
"LoginRequest","username": "delphix_user","password": "delphix_pass","target":
"DOMAIN"}EOF
```

The new cookie (cookie2.txt) will need to be used in subsequent API requests. The login API currently only supports authentication by a password. There is no way to authenticate using any shared key or alternate authentication strategy.

CLI to web services translation

This topic describes using the CLI to understand public web service APIs.

The command-line interface is a direct translation of the web services API to an interactive environment. This allows you to use the CLI to explore functionality with tab completion, integrated help, stronger type checking, and an indication of expected types and required fields. When trying to determine how to invoke an operation through the web services or interpret the results, it is recommended that you first learn how to do the same through the CLI, and then use the provided tools to translate that into web services call.

CLI translation to HTTP

The CLI namespace is identical to the web service URLs for each base object and operation type. The root of all web services is `/resources/json/delphix`. Any additional CLI context is appended to this URL, joined by slashes. For example:

```
delphix> database provision
```

Is equivalent to:

```
POST /resources/json/delphix/database/provision
```

All operations in the CLI (those that require an explicit `commit` command) are modeled as `POST` HTTP calls. This is an example of a "root operation", as they are invoked not on any particular object, but across the system as a whole. Operations that are invoked on a particular object use a URL specific to that object:

```
delphix> database "dexample" refresh
```

Is equivalent to:

```
POST /resources/json/delphix/database/ORACLE_DB_CONTAINER-3/refresh
```

While the CLI uses names to refer to objects, at the API level we use references. Persistent objects (those with a permanent unique identity) have a `reference` field that is used in all cases to refer to the object. This allows references to remain constant even if objects are renamed.

For the standard CRUD (create, read, update, delete) operations, the mapping is slightly different:

CLI operation	HTTP API
`database list`	`GET /resources/json/delphix/database`
`database create`	`POST /resources/json/delphix/database`
`database "dexample" get`	`GET /resources/json/delphix/database/<>`

CLI operation	HTTP API
`database "dexample" update`	`POST /resources/json/delphix/database/<>`
`database "dexample" delete`	`DELETE /resources/json/delphix/database/<>` `POST /resources/json/delphix/database/>>`

The `DELETE` operation has an optional `POST` form that can take complex input for clients that don't support sending a payload as part of a `DELETE` operation.

Tracing HTTP calls

The CLI also provides facilities to see the raw HTTP calls being made as part of any operation. To start with, viewing data in JSON format (`setopt format=json`) will provide an example of what the raw output looks like from the server. In its raw form, the CLI does not make any attempt to interpret the results, so references are displayed as references (and not names), and sizes are displayed as their raw numeric value.

This is helpful for scripting, but the CLI also has a mode to display the requests being sent to the server, the responses received, and the URLs used. To enable this mode, run `setopt trace=true` . Once you have determined how to do something through the CLI, you can use this mode as the basis for building direct integration with the raw HTTP APIs.

```
delphix group> setopt trace=true
delphix group> create
delphix group create *> set name=example
delphix group create *> set description="this is an example"
delphix group create *> commit
=== POST /resources/json/delphix/group ===
{
    "type": "Group",
    "name": "example",
    "description": "this is an example"
}
=== RESPONSE ===
{
    "status": "OK",
    "result": "GROUP-3",
    "action": "ACTION-37",
    "job": null
}
=== END ===
    GROUP-3
delphix group> "example"
delphix group "example"> delete
=== GET /resources/json/delphix/group/GROUP-3 ===
=== RESPONSE ===
{
```

```
    "status": "OK",
    "result": {
        "type": "Group",
        "reference": "GROUP-3",
        "namespace": null,
        "name": "example",
        "description": "this is an example"
    },
    "action": null,
    "job": null
}
=== END ===
delphix group "example" delete *> commit
=== POST /resources/json/delphix/group/GROUP-3/delete ===
{}
=== RESPONSE ===
{
    "status": "OK",
    "result": "",
    "action": "ACTION-38",
    "job": null
}
=== END ===
delphix group>
```

When using trace mode within the context of a specific object, we refresh the contents of the object before executing each command. This results in the `GET` request before the `delete` command in the above example.

GUI API mapping

This topic describes how to map from GUI operations to the corresponding CLI operation.

It is not always straightforward to convert from the visual layout of the GUI to the corresponding CLI operations. This topic outlines some common operations and indicates how they are represented in the CLI. web services, and the API documentation.

dSource operations

GUI	CLI	API topic	Input object	Web services
Link	`database link`	`Container`	`LinkParameters`	`POST /resources/json/delphix/database/link`
Show configuration	`database "name" get` `source "name" get`	`Container` `Source`	`-`	`GET /resources/json/delphix/database/{ref}` `GET /resources/json/delphix/source/{ref}`
Sync	`database "name" sync`	`Container`	`SyncParameters`	`POST /resources/json/delphix/database/{ref}/sync`
Update	`database "name" update`	`Container`	`Container`	`POST /resources/json/delphix/database/{ref}`
Delete	`database "name" delete`	`Container`	`DeleteParameters`	`POST /resources/json/delphix/database/{ref}/delete`
Detach	`database "name" detachSource`	`Container`	`DetachSourceParameters`	`POST /resources/json/delphix/database/{ref}/detachSource`

GUI	CLI	API topic	Input object	Web services
Attach	`database "name" attachSource`	`Container`	`AttachSourceParameters`	`POST /resources/json/delphix/database/{ref}/attachSource`
Disable	`source "name" disable`	`Source`	`SourceDisableParameters`	`POST /resources/json/delphix/source/{ref}/disable`
Enable	`source "name" enable`	`Source`	`SourceEnableParameters`	`POST /resources/json/delphix/source/{ref}/enable`

VDB operations

GUI	CLI	API topic	Input object	Web services
Provision	`database provision`	`Container`	`ProvisionParameters`	`POST /resources/json/delphix/database/provision`
V2P	`database export`	`Container`	`ExportParameters`	`POST /resources/json/delphix/database/export`
Refresh	`database "name" refresh`	`Container`	`RefreshParameters`	`POST /resources/json/delphix/database/{ref}/refresh`
Snapshot	`database "name" sync`	`Container`	`SyncParameters`	`POST /resources/json/delphix/database/{ref}/sync`
Update	`database "name" update`	`Container`	`Container`	`POST /resources/json/delphix/database/{ref}`

GUI	CLI	API topic	Input object	Web services
Delete	`database "name" delete`	`Container`	`DeleteParameters`	`POST /resources/json/delphix/database/{ref}/delete`
Start	`source "name" start`	`Source`	`StartParameters`	`POST /resources/json/delphix/source/{ref}/start`
Stop	`source "name" stop`	`Source`	`StopParameters`	`POST /resources/json/delphix/source/{ref}/stop`
Enable	`source "name" enable`	`Source`	`SourceEnableParameters`	`POST /resources/json/delphix/source/{ref}/enable`
Disable	`source "name" disable`	`Source`	`SourceDisableParameters`	`POST /resources/json/delphix/source/{ref}/disable`

Environment operations

GUI	CLI	API topic	Input object	Web services
Add environment	`environment create`	`SourceEnvironment`	`SourceEnvironmentCreateParameters`	`POST /resources/json/delphix/environment`
Update	`environment "name" update`	`SourceEnvironment`	`Environment`	`POST /resources/json/delphix/environment/{ref}`
Delete	`environment "name" delete`	`SourceEnvironment`	`-`	`DELETE /resources/json/delphix/environment/{ref}`

GUI	CLI	API topic	Input object	Web services
Refresh	`environment "name" refresh`	`SourceEnvironment`	`-`	`POST /resources/json/delphix/environment/{ref}/refresh`
Enable	`environment "name" enable`	`SourceEnvironment`	`-`	`POST /resources/json/delphix/environment/{ref}/enable`
Disable	`environment "name" disable`	`SourceEnvironment`	`-`	`POST /resources/json/delphix/environment/{ref}/disable`
Add manual repository	`repository create`	`SourceRepository`	`SourceRepository`	`POST /resources/json/delphix/repository`
Update repository	`repository "name" update`	`SourceRepository`	`SourceRepository`	`POST /resources/json/delphix/repository/{ref}`
Remove manual repository	`repository "name" delete`	`SourceRepository`	`-`	`DELETE /resources/json/delphix/repository/{ref}`
Show host details	`host "name" get`	`Host`	`-`	`GET /resources/json/delphix/host/{ref}`
Add cluster node	`environment oracle clusternode create`	`OracleClusterNode`	`OracleClusterNode`	`POST /resources/json/delphix/environment/oracle/clusternode`

GUI	CLI	API topic	Input object	Web services
Update cluster node	`environment oracle clusternode "name" update`	`OracleClusterNode`	`OracleClusterNode`	`POST /resources/json/delphix/environment/oracle/clusternode/{ref}`
Remove cluster node	`environment oracle clusternode "name" delete`	`OracleClusterNode`	`-`	`DELETE /resources/json/delphix/environment/oracle/clusternode/{ref}`
Add listener	`environment oracle listener create`	`OracleListener`	`OracleListener`	`POST /resources/json/delphix/environment/oracle/listener`
Update listener	`environment oracle listener "name" update`	`OracleListener`	`OracleListener`	`POST /resources/json/delphix/environment/oracle/listener/{ref}`
Remove listener	`environment oracle listener "name" delete`	`OracleListener`	`-`	`DELETE /resources/json/delphix/environment/oracle/listener/{ref}`

CLI to Python transition

This topic describes using the CLI to understand the Python APIs.

The command-line interface is a direct translation of the web services API to an interactive environment. This allows you to use the CLI to explore functionality with tab completion, integrated help, stronger type checking, and indication of expected types and required fields. When trying to determine how to invoke an operation through the Python API or interpret the results, it is recommended that you first learn how to do the same through the CLI, and then use the provided tools to translate that into Python calls.

Installation

The Delphix Python API is available through PyPI and you may install it with pip.

```
pip install delphixpy
```

Minimum python version
Requiring Version 2.7 and above

Connecting to the Delphix engine

In the Delphix Python API, all operations take an engine object which represents your connection to a Delphix Engine. Here is how you connect to the Delphix Engine using the Python API and acquire the engine object.

```
from delphixpy.delphix_engine import DelphixEngineengine = DelphixEngine("delphix-address", "delphix-user", "delphix-password", "DOMAIN")  # Instead of DOMAIN, use SYSTEM if you are using the sysadmin user.
```

CLI translation to Python

For backward compatibility purposes, delphixpy provides the ability to write integrations against a specific API version. The latest version is always in the root of the package. Writing against the latest version requires you to update your integrations if the API changes in future versions of the API.

Specific API versions can be used by importing the corresponding sub-package. The sub-packages are named after the API versions in the format `v<major>_<minor>_<micro>` . For example, if you want to look into API 1.5.0, you should be using modules from `delphixpy.v1_5_0` . Modules from different sub-package versions cannot interact with each other so be careful if you wish to mix API versions in the same code base.

All CLI namespaces have a corresponding Python package in which operations can be accessed. The main Python package is called `web` . All value objects which can be manipulated or read through the CLI can be found in `web.vo` .

```
delphix> database provision
```

Is equivalent to:

```
from delphixpy.web import databasedatabase.provision(engine, provision_parameters)
```

The provision_parameters object in this example is an instance of ProvisionParameters which can be found in delphixpy.web.vo. The properties of the object map to the parameters you would need to specify before doing a commit in the CLI provision context.

This is an example of an "operation", as they are invoked on an object. Operations that are invoked on a particular object take a reference to that object.

```
delphix> database "dexample" refresh
```

Is equivalent to (connection code omitted):

```
from delphixpy.web import databasedatabase.refresh(engine, "ORACLE_DB_CONTAINER-3",
RefreshParameters)
```

While the CLI uses names to refer to objects, the Python API, just like the web services, use references (ORACLE_DB_CONTAINER-3). Persistent objects (those with a permanent unique identity) have a `reference` field that is used in all cases to refer to the object. This allows references to remain constant even if objects are renamed.

For the standard CRUD (create, read, update, delete) operations, the mapping is slightly different:

CLI operation	Python API
`group list`	`group.get_all(engine)`
`group create`	`group.create(engine, group=`
`group "dexample" get`	`group.get(engine,`
`group "dexample" update`	`group.update(engine, , group=)`
`group "dexample" delete`	group.delete(engine,

Example: creating a group

This is how you can create a group as a fully working example.

```
from delphixpy.web import groupfrom delphixpy.web.vo import Groupfrom
delphixpy.delphix_engine import DelphixEngineengine = DelphixEngine("delphix-address",
 "delphix-user", "delphix-password", "DOMAIN")my_group = Group()my_group.name = "My
Group"my_group.description = "This is my new group!"group.create(engine, my_group)
```

Asynchronous mode

The Python API runs by default in synchronous mode. If you would wish to perform operations asynchronously there is a context manager that allows you to do that. If you need to track job progress in asynchronous mode, you can get the reference of the last job started from engine.last_job. When exiting the async context manager, it will wait for all jobs started within the context to finish. You can also clear the job from the context so that you do not wait for its completion or status when exiting the context manager. If a job fails, exceptions.JobError will be thrown.

Here is how you would perform a sync operation on all databases asynchronously.

```
from delphixpy.delphix_engine import DelphixEnginefrom delphixpy import
 job_contextfrom delphixpy.web import databaseengine = DelphixEngine("delphix-
address", "delphix-user", "delphix-password", "DOMAIN")all_databases =
database.get_all(engine)with job_context.asyncly(engine):    for db in all_databases:
database.sync(engine, db.reference)
```

Python cookbook: adding a UNIX host

This topic describes the process of adding a UNIX host using the delphixpy Python API.

Within Delphix, there are both hosts and host environments. A host represents a remote system, but may or may not be a source or target for linking or provisioning. For example, in an Oracle RAC cluster, the cluster environment represents the location of the Oracle installation(s), and while there are hosts within that cluster they are not individually manageable as environments.

Procedure

1. Create new environment creation parameters and initialize the structure for a UNIX host.
   ActionScript

   ```
   from delphixpy.web.vo import HostEnvironmentCreateParameters,
   UnixHostEnvironment, UnixHostCreateParameters, UnixHost, EnvironmentUser,
   PasswordCredentialhost_environment_create_parameters_vo =
   HostEnvironmentCreateParameters()host_environment_create_parameters_vo.host_env
   ironment =
   UnixHostEnvironment()host_environment_create_parameters_vo.host_parameters =
   UnixHostCreateParameters()host_environment_create_parameters_vo.host_parameters
   .host = UnixHost()host_environment_create_parameters_vo.primary_user =
   EnvironmentUser()host_environment_create_parameters_vo.primary_user.credential
   = PasswordCredential()
   ```

2. Set the host address and port. The name of the environment is derived from the address used, though you can provide a more descriptive name if desired. The address can be a DNS names, IP addresses, or a comma separated list of the above.
   ActionScript

   ```
   host_environment_create_parameters_vo.host_parameters.host.addresses =
   ["192.168.1.2"]host_environment_create_parameters_vo.host_parameters.host.port
   = 22
   ```

3. Set the toolkit path. This is where Delphix will store temporary binaries used while the host is configured as part of Delphix.
   ActionScript

   ```
   host_environment_create_parameters_vo.host_parameters.host.toolkit_path = "/
   var/delphix"
   ```

4. Set the username and password to use when connecting over SSH. This user must have the privileges described in the Delphix Administration Guide. To configure a SSH user, change the credential object to `SystemKeyCredential`.
   ActionScript

```
host_environment_create_parameters_vo.primary_user.name = "oracle"host_environm
ent_create_parameters_vo.primary_user.credential.password = "my secret
password"
```

5. Commit the result. A reference to your new environment will be returned from the create call. The environment discovery process will execute as an asynchronous job. The default behavior is to wait for the result, so progress will be updated until the discovery process is complete or fails.
ActionScript

```
from delphixpy.delphix_server import DelphixServerfrom delphixpy.web import
 environmentserver = DelphixServer("delphix-address", "delphix-user", "delphix-
password", "DOMAIN")new_environment_reference = environment.create(server,
host_environment_create_parameters_vo)
```

6. Full example
ActionScript

```
from delphixpy.delphix_server import DelphixServerfrom delphixpy.web import
 environmentfrom delphixpy.web.vo import HostEnvironmentCreateParameters,
UnixHostEnvironment, UnixHostCreateParameters, UnixHost, EnvironmentUser,
PasswordCredentialhost_environment_create_parameters_vo =
HostEnvironmentCreateParameters()host_environment_create_parameters_vo.host_env
ironment =
UnixHostEnvironment()host_environment_create_parameters_vo.host_parameters =
UnixHostCreateParameters()host_environment_create_parameters_vo.host_parameters
.host = UnixHost()host_environment_create_parameters_vo.primary_user =
EnvironmentUser()host_environment_create_parameters_vo.primary_user.credential
=
PasswordCredential()host_environment_create_parameters_vo.host_parameters.host.
addresses = ["192.168.1.2"]host_environment_create_parameters_vo.host_parameter
s.host.port =
22host_environment_create_parameters_vo.host_parameters.host.toolkit_path = "/
var/delphix"host_environment_create_parameters_vo.primary_user.name = "oracle"h
ost_environment_create_parameters_vo.primary_user.credential.password = "my
secret password"server = DelphixServer("delphix-address", "delphix-user",
"delphix-password", "DOMAIN")new_environment_reference =
environment.create(server, host_environment_create_parameters_vo)
```

So you want to work with Delphix APIs?

What is RESTFul? API? JSON? CLI? Object Reference? GET/POST? Cookies? HTTP/HTTPS? cURL? Where do I begin? Who can help me? Documentation? Tutorials? That's great for Linux, but I am on Windows? Parse? Don't have a clue about sed, awk, grep, cut, and other acronyms. What about Regular Expressions, like "Hello"?

Just a small sampling of questions that you may or may not know the answers to, let alone learning Delphix and a programming language. Delphix is a technical product, and being new to the Delphix family can be a bit overwhelming. The goal for this document is simple: to enable users to get up to speed quickly on how to use Delphix APIs.

This section covers the following topics:

Background information

This document assumes that you have some basic Delphix product experience and entry-level programming knowledge. The first two sections of this document, Delphix API Reference URLs and API Prerequisite Knowledge, are focused on providing the required information and reference material/URLs.

This document is for informational and demonstration purposes only. The examples are for demonstration purposes only and must be used at your own risk. As always, test and verify on development systems prior to migrating code to production environments.

What is RESTFul programming?

http://stackoverflow.com/questions/671118/what-exactly-is-restful-programming

A great way to learn how to generate the Delphix RESTFul API calls and the required JSON content is to use the Delphix CLI (Command Line Interface) and turn on the trace option.

```
Delphix5110HWv8>
```
setopt trace=true

All subsequent CLI commands will display the GET or POST API URL with the respective input or output JSON data string. This guide will walk you through an example later.

Delphix API reference URLs

There are a number of sources available to provide details, examples, and techniques for working with Delphix APIs. This section contains a small list of URLs that are worth reviewing/reading as required.

CLI (command line interface)

The Delphix Engine provides a native command-line interface (CLI) accessible over SSH. This CLI provides an interactive layer on top of the public web service APIs, and is intended for users that wish to automate interactions with the Delphix Engine, or simply prefer a text-based interface. All of the functionality available in the CLI is also available through the public stable web service APIs should more full-featured automation be required. For more information on automation using CLI commands see Command Line Interface Guide

RESTFul APIs

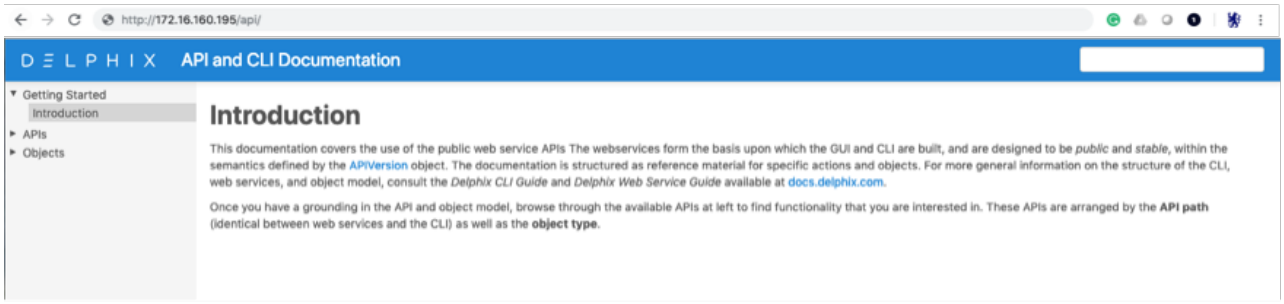
> **Must be logged in**
> Users must be successfully logged in before /API pages can be accessed.

For more information on automation using the web service APIs, see the Web Services API Guide.

API Documentation is also included within the Delphix Engine using the following formula:

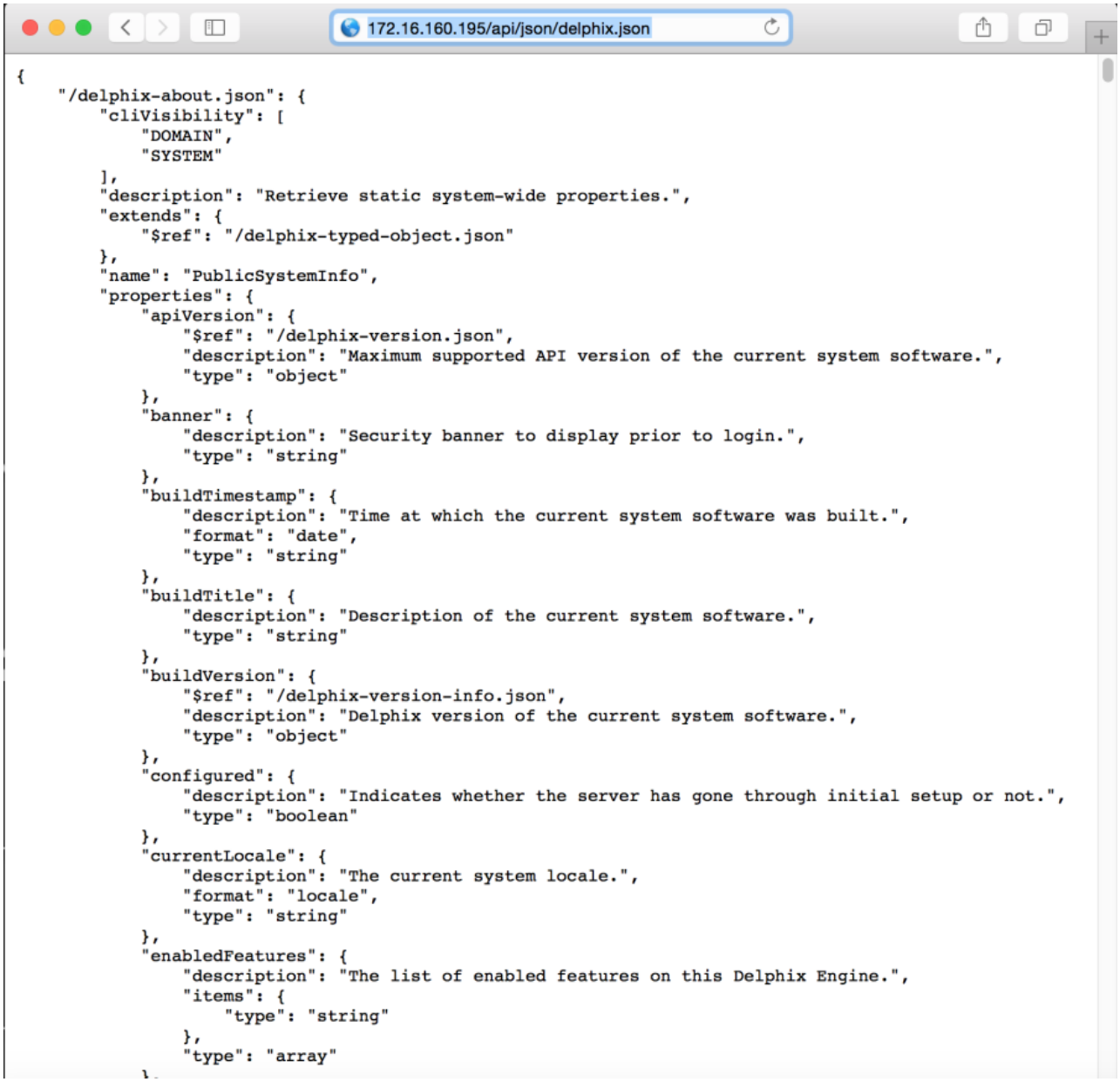
http://<delphix_engine>/api/

For Example http://172.16.160.195/api/



For a complete list of Delphix APIs - JSON schema format, use the following URL:

http://<delphix_engine>/api/json/delphix.json



So, looking at the first JSON key/name:

```
"/delphix-about.json": {    "cliVisibility": [        "DOMAIN",        "SYSTEM"    ]
,     "description": "Retrieve static system-wide properties.",     . . .
```

And after logging into the Delphix Engine, translating this into the URL API for *about*:

http://172.16.160.195/resources/json/delphix/about will respond with the returned JSON data string.


```
{"type":"OKResult","status":"OK","result":{"type":"PublicSystemInfo","productType":"s
tandard","productName":"Delphix Engine","buildTitle":"Delphix Engine 5.1.2.0","buildT
imestamp":"2016-09-02T22:28:43.000Z","buildVersion":{"type":"VersionInfo","major":5,"
minor":1,"micro":2,"patch":0},"configured":true,"enabledFeatures":["XPP","JETSTREAM"]
```

```
,"apiVersion":{"type":"APIVersion","major":1,"minor":8,"micro":0},"banner":null,"loca
les":["en-US"],"currentLocale":"en-US"},"job":null,"action":null}
```


For now, just remember that the Delphix Engine contains the API Documentation and Delphix JSON schema.

Masking APIs

Please refer to the Masking documentation at https://maskingdocs.delphix.com for information on the Masking APIs.

Cookbook examples

Delphix documentation includes a number of cookbook examples that will not be duplicated in this section but may be referenced.

API Cookbook: Common Tasks, Workflows, and Examples

There are also working examples provided within this document and are available for download.

API prerequisite knowledge

JSON

JSON (JavaScript Object Notation) is a minimal, readable format for structuring data. It is a simple format for transmitting data between applications, as an alternative to XML. The Delphix API uses JSON data structure in the format of strings to send and receive data from the API calls, as you will see later in the examples. First, let's look at the JSON fundamentals.

Keys and values

The two primary parts that makeup JSON are keys and values. Together they make key/value pairs, also called name/value pairs.

- **Key** – Always a string enclosed in quotation marks.
- **Value** – Can be a string, number, boolean expression, array, or object.
- **Key/Value Pair** – Follows a specific syntax, with the key followed by a colon followed by the value. Key/value pairs are comma separated.

Let's take a JSON sample string and identify each part of the code.

```
{    "foo" : "bar",    "rows" : 100}
```

The curly brackets start and end the string. The key is "foo" and the value is "bar". A colon (:) is the delimiter between them. A comma (,) is the delimiter for multiple key/value pairs. The second pair is "rows" and the value is a number of 100.

Types of values

Number	**An integer or a decimal number**
Boolean	True or false
String	Plain text alphanumeric readable characters
Null	Empty
Array	An associative array of values
Object	An associative array of key/value pairs

Numbers, booleans, and strings.

It is very important to understand the APIs JSON object definitions. Quoted values are treated as strings!

"x" : "1" is treated as a string, while

"x" : 1 is treated as a number

"y" : "true" is treated as a string, while

"y" : true is treated as a boolean true (false)

Null values

```
{    "z" :    , "b" : "World"}
```

Nulls are empty values, but sometimes programmers code "" as a null value.

```
{    "z" : ""   , "b" : "World"  }
```

So always verify how the null values are defined and handled by the application.

Arrays

An array is indicated with the square brackets: [value1, value2, etc.]. In this example, we have added a categories key with an array of values.

```
..."foo" : {  "bar" : "Hello",  "category" : [ "greetings", "morals" ]}...
```

Objects

An object is indicated by curly brackets: {"key", "value"}. Everything inside of the curly brackets is part of the object. We already learned that a value could be an object. Therefore, "foo" and the corresponding object are a key/value pair.

```
..."foo" : { "bar" : "Hello" }...
```

The key/value pair "bar" : "Hello" is nested inside the key/value pair "foo" : { ... }. That is an example of a hierarchy (or nested data) within JSON data.

Arrays and Objects can be nested or contained within the same level.

Summary

JSON arrays are `[ , , ]`

JSON nested objects are `, , "x":{ "a":"1", "b":"2" }, ,`

JSON data can be passed within the HTTP URL (file or argument), the header, or other handlers.

From within Shell Scripts or Programming Languages, JSON data is typically processed through a "JSON parser." This topic is covered later.

Delphix CLI

Connecting to the Delphix engine CLI

Reference: Connecting to the CLI

There are two user roles accessible, the **sysadmin** and the **delphix_admin**.

From a shell environment, you can connect using the ssh command. The IP Address (or Hostname) represents the Delphix Engine (case sensitive):

```
ssh sysadmin@127.16.160.195
```

`ssh delphix_admin@127.16.160.195`

From a putty session, open an ssh connection to the Delphix Engine IP Address or Hostname (case sensitive):

`open 127.16.160.195`

`Login User: sysadmin@SYSTEM`

`#...` or `...`

`Login User: delphix_admin@DOMAIN`

After entering the correct password for the respective user, the menus for that user's role will be different. For example, the **sysadmin@SYSTEM** user has engine storage, whereas the **delphix_admin@DOMAIN** user has database provisioning.

You can use the CLI for scripting and configure the connection for ssh passwordless connections.

CLI Cookbook: Configuring Key-Based SSH Authentication for Automation

How to use the CLI to learn the APIs

As stated earlier, a great way to learn how to generate the Delphix RESTFul API calls and the required JSON content is to use the Delphix CLI (Command Line Interface) and turn on the `CLI> setopt trace=true` option.

Below is an example of how to get the JSON required parameters for a database refresh per the type of refresh performed.

Other types or options may require other JSON parameters, so after changing any parameter, we recommend performing an "ls" command to see if there are any new parameters and/or required values.

```
The refresh database example below shows how to use the CLI to identify reference
objects for other CLI commands and the respective RESTFul API structure when
the  setopt trace=true  option is set.

$ ssh delphix_admin@172.16.160.195 Password: Delphix5030HWv8> ls
Childrenaboutaction...connectivitydatabaseenvironment... toolkituser
OperationsversionDelphix5030HWv8> database Delphix5030HWv8 database> ls
ObjectsNAME            PROVISIONCONTAINER    DESCRIPTIONDPXDEV01        -
Vdelphix_demo    delphix_demo        -delphix_demo    - Scripts         -
V_2C1            Scripts               -Vvfiles           -                    -
Childrentemplate
OperationscreateEmptycreateRestorationDatasetexportfileMappinglinkoracleSupportedChar
acterSetsprovisionvalidateXppxpp
```

First, we need to identify the target Delphix virtualized database object to refresh ...

Each Delphix object has a reference that is typically used for parameter values.

```
Delphix5030HWv8 database> select Vdelphix_demo Delphix5030HWv8 database
'Vdelphix_demo'> ls Properties    type: MSSqlDatabaseContainer    name:
Vdelphix_demo    creationTime: 2016-06-16T14:30:03.033Z    currentTimeflow:
```

```
'DB_PROVISION@2016-06-16T10:30:08'    delphixManaged: true    description:
(unset)    group: Windows    masked: false    os: Windows    performanceMode:
DISABLED    processor: x86    provisionContainer: delphix_demo    reference:
MSSQL_DB_CONTAINER-39    restoration: false    runtime:        type:
MSSqlDBContainerRuntime        logSyncActive: false    sourcingPolicy:        type:
SourcingPolicy        loadFromBackup: false        logsyncEnabled: false    transform
ation: false Operationsdelete...purgeLogsrefreshremoveLiveSource...Delphix5030HWv8
database 'Vdelphix_demo'> refresh Delphix5030HWv8 database 'Vdelphix_demo' refresh *>
ls Properties    type: RefreshParameters    timeflowPointParameters:        type:
TimeflowPointSemantic        container: (required)        location:
LATEST_POINTDelphix5030HWv8 database 'Vdelphix_demo' refresh *> set
timeflowPointParameters.container=delphix_demo Delphix5030HWv8 database
'Vdelphix_demo' refresh *> ls Properties    type:
RefreshParameters    timeflowPointParameters:        type:
TimeflowPointSemantic        container: delphix_demo (*)        location:
LATEST_POINTDelphix5030HWv8 database 'Vdelphix_demo' refresh > *commit    Dispatched
job JOB-100    DB_REFRESH job started for "Windows/Vdelphix_demo".    Validating that
this dataset is managed by Delphix.    Stopping virtual database.    Unmounting
datasets.    Unexporting storage containers.    Metadata for dSource "Vdelphix_demo"
 successfully deleted.    Starting provisioning of virtual database "Vdelphix_demo".
   Creating new TimeFlow.    Generating recovery scripts.    Mounting
datasets.    Mounting read-only source logs dataset.    Running user-specified pre-
provisioning script.    Recovering virtual database.    The virtual database recovery
was successful.    Unmounting read-only source logs dataset.    Running user-
specified post-provisioning script.    The virtual database "Vdelphix_demo" was
successfully provisioned.    DB_REFRESH job for "Windows/Vdelphix_demo" completed
successfully.
```

Refresh again but this time turn on the `setopt trace=true` option.

```
Delphix5030HWv8 database 'Vdelphix_demo'> refreshDelphix5030HWv8 database
'Vdelphix_demo' refresh *> lsProperties    type:
RefreshParameters    timeflowPointParameters:        type:
TimeflowPointSemantic        container: (required)        location:
LATEST_POINTDelphix5030HWv8 database 'Vdelphix_demo' refresh *> set
timeflowPointParameters.container=delphix_demoDelphix5030HWv8 database
'Vdelphix_demo' refresh *> lsProperties    type:
RefreshParameters    timeflowPointParameters:        type:
TimeflowPointSemantic        container: delphix_demo        location:
LATEST_POINTDelphix5030HWv8 database 'Vdelphix_demo' refresh *> setopt
trace=trueDelphix5030HWv8 database 'Vdelphix_demo' refresh *> commit=== POST /
resources/json/delphix/database/MSSQL_DB_CONTAINER-39/refresh ===={    "type":
"RefreshParameters",    "timeflowPointParameters": {        "type":
"TimeflowPointSemantic",        "container": "MSSQL_DB_CONTAINER-38"    }}...
```

⚠ The "container" value in the JSON output above is different from the target VDB reference because we are refreshing from the source database container! In this example, the **set timeflowPointParameters.container=delphix_demo** is represented in JSON output as **"container": "MSSQL_DB_CONTAINER-38"**

Using the CLI, you can identify the RESTFul API POST and GET commands along with the JSON input data requirements.

```
=== POST /resources/json/delphix/database/MSSQL_DB_CONTAINER-39/refresh ==={     "typ
e": "RefreshParameters",     "timeflowPointParameters": {         "type":
"TimeflowPointSemantic",         "container": "MSSQL_DB_CONTAINER-38"     } }
```

So framing the RESTFul URL for a virtual database refresh, the URL will look like

http://<delphix_engine>/resources/json/delphix/database/ MSSQL_DB_CONTAINER-39 /refresh

where the **MSSQL_DB_CONTAINER-39** represents the target virtualized database to refresh. We need to POST the JSON data to the URL for processing.

```
{    "type": "RefreshParameters",     "timeflowPointParameters": {          "type":
"TimeflowPointSemantic",         "container": "MSSQL_DB_CONTAINER-38"     }}
```

The **"timeflowPointParameters"** key has 6 **"type": "..."** options, each of which has its own set of parameters. The type **"TimeflowPointSemantic"** uses the default LATEST_POINT within the source container, so for simplicity, we will use this type. For more information on timeflowPointParameters 6 types, see the Advanced Section.

If this is a little confusing at this point, do not worry, that's typical. Complete examples will be shown later. The important items to remember are:

- Delphix often uses object reference names within the JSON data.
- Using the `setopt trace=true` the option provides the construct for the RESTFul API URLs and the JSON data for POST / GET operations.

HTTP

We use the HTTP protocol every day for web browsing and commercial business. From finding a new restaurant to buying a 1986 Ford Thunderbird Turbo Coupe!

Most people see the HTTP within the URL Address field within the Web Browser window – for example, http://www.google.com

But behind the scenes, HTTP is performing a wide range of functionality. For RESTFul APIs, they use HTTP's GET and POST form functionality to process data. In Delphix's case, the data is also represented as JSON structures.

HTTP GET operation is used to return data only, while HTTP POST operation is used to provide data input in the form of a structured JSON data string or file.

cURL

What is cURL?

The cURL client command is based on a library supporting a number of web protocols, including HTTP. The "curl" command can be called from the command line, while the cURL library is commonly integrated with your favorite programming languages, such as Java, JSP, Python, Perl, PHP, .NET, and PowerShell.

Due to its widespread adoption, we will use cURL for making the Delphix RESTFul API calls within this document. Some operating systems or languages support their own HTTP commands / related libraries, and you can use these instead of cURL. One alternative is the "wget" command described later.

Is cURL installed?

```
Operating System Prompt>    curl --versioncurl 7.19.7 (x86_64-redhat-linux-gnu)
libcurl/7.19.7 NSS/3.19.1 Basic ECC zlib/1.2.3 libidn/1.18 libssh2/1.4.2Protocols:
tftp ftp telnet dict ldap ldaps http file https ftps scp sftp Features: GSS-Negotiate
IDN IPv6 Largefile NTLM SSL libz
```

Get the HTTP output from google.com

```
Operating System Prompt> curl www.google.com
```

Wget

An alternative to cURL is Wget, which is typically a native command on all Linux environments. See the Appendix for a complete comparison between Wget and cURL.

dxtoolkit2

Delphix has developed a very robust toolkit, dxtoolkit2, which utilizes the Delphix RESTFul APIs. This toolkit is cross-platform. Its commands are built with the Perl programming language.

We recommend that you review the dxtoolkit2 documentation; you may find a utility that already performs your desired function. For example, the utility ***dx_get_analytics*** is absolutely great for dumping analytic data from the Delphix Engine into a .csv (comma-separated value) format, which you can then easily integrate into your enterprise monitoring tools. See the sample "Analytics" use case.

Contact Delphix personnel for the latest download.

Delphix RESTFul APIs command line basics

Authentication

RESTFul APIs require authentication. Just plugging the URL into a web browser or running an operating system cURL command will return an authentication/login required error message.

Command line:

curl http://172.16.160.195/resources/json/delphix/environment

Response:

```
<!DOCTYPE html><html><head><title>Apache Tomcat/8.0.29 - Error report</title><style
type="text/css">H1 {font-family:Tahoma,Arial,sans-serif;color:white;background-
color:#525D76;font-size:22px;} H2 {font-family:Tahoma,Arial,sans-
serif;color:white;background-color:#525D76;font-size:16px;} H3 {font-
family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-size:14px;}
BODY {font-family:Tahoma,Arial,sans-serif;color:black;background-color:white;} B
{font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;} P {font-
family:Tahoma,Arial,sans-serif;background:white;color:black;font-size:12px;}A
{color : black;}A.name {color : black;}.line {height: 1px; background-color: #525D76;
border: none;}</style> </head><body><h1>HTTP Status 403 - Use /resources/json/
delphix/login to log in first</h1><div class="line"></div><p><b>type</b> Status
report</p><p><b>message</b> <u>Use /resources/json/delphix/login to log in first</
u></p><p><b>description</b> <u>Access to the specified resource has been forbidden.</
u></p><hr class="line"><h3>Apache Tomcat/8.0.29</h3></body></html>
```

The authentication process requires you to establish a session first.

The example within this section illustrates the session/login and subsequent API calls with cURL using cookies created when the session was established.

Session

API Version Information

When programming for compatibility, the API version number is very important. Please be aware of the differences between versions for enterprise applications. For example, if you specify a 1.11.6 version, ONLY the available calls and functionality for that version will be used, and it will only be operational on Delphix Engine versions that support that version.

The session uses the –c for the cookie creation, the login uses the -b for the exiting cookie created by the session, and -c to create a new cookie, the other commands use the –b for using the existing cookie created by the login.

From the Unix/Linux command line

```
$ curl -s -X POST -k --data @- http://delphix-server/resources/json/delphix/session \
-c ~/cookies.txt -H "Content-Type: application/json" <<EOF{   "type": "APISession",
"version": {       "type": "APIVersion",       "major": 1,       "minor": 11,
"micro": 6   }}EOF
```

Returned to the command line are the results (added linefeeds for readability)

```
{   "status":"OK",   "result": {       "type":"APISession",       "version": {
"type": "APIVersion",           "major": 1,           "minor": 11,           "micro":
6       },       "locale": "en_US",       "client": null   },   "job": null}
```

Login

Once you have established the session, the next step is to authenticate to the server by executing the Login Request API. Unauthenticated sessions are prohibited from making any API calls other than this login request. The username can be either a system user or domain user; the backend will authenticate using the appropriate method.

The session uses the –c for the cookie creation, the login uses the -b for the exiting cookie created by the session, and -c to create a new cookie, the other commands use the –b for using the existing cookie created by the login.

From the Unix/Linux command line

```
$ curl -s -X POST -k --data @- http://delphix-server/resources/json/delphix/login \
-b ~/cookies.txt -c ~/cookies.txt -H "Content-Type: application/json"
<<EOF{   "type": "LoginRequest",   "username": "delphix_username",   "password":
"delphix_password"}EOF
```

Returned to the command line are the results (added linefeeds for readability)

```
{"status":"OK","result":"USER-2","job":null,"action":null}
```

Sample Delphix API call

With a successful authentication (session, login, and saved cookie), calls to the Delphix Engine can now be made to perform the desired functionality.

The session uses the –c for the cookie creation, the login uses the -b for the exiting cookie created by the session, and -c to create a new cookie, the other commands use the –b for using the existing cookie created by the login.

For starters, let's create a session, login, and get the existing environments defined within the Delphix Engine.

```
curl -s -X POST -k --data @- http://172.16.160.195/resources/json/delphix/session \-c
~/cookies.txt -H "Content-Type: application/json" <<EOF{    "type":
"APISession",    "version": {        "type": "APIVersion",        "major":
1,        "minor": 7,        "micro": 0    }}EOF
```

```
curl -s -X POST -k --data @- http://172.16.160.195/resources/json/delphix/login
\    -b ~/cookies.txt -c ~/cookies.txt -H "Content-Type: application/json"
<<EOF{    "type": "LoginRequest",    "username": "admin",    "password": "delphix"}
EOF
```

```
curl -X GET -k http://172.16.160.195/resources/json/delphix/environment \    -b ~/
cookies.txt -H "Content-Type: application/json"
```

Returned to the command line are the results (added linefeeds for readability)

```
{"type":"ListResult","status":"OK","result":[ {"type":"WindowsHostEnvironment",
"reference":"WINDOWS_HOST_ENVIRONMENT1",  "namespace":null,  "name":"Window Target",
"description":"",  "primaryUser":"HOST_USER-1",  "enabled":false,  "host":"WINDOWS_HO
ST1",  "proxy":null }, {  "type":"UnixHostEnvironment",  "reference":"UNIX_HOST_ENVIR
ONMENT-3",  "namespace":null,  "name":"Oracle Target",  "description":"",
"primaryUser":"HOST_USER-3",  "enabled":true,  "host":"UNIX_HOST-3","aseHostEnvironme
ntParameters":null }],"job":null,"action":null,"total":2,"overflow":false}
```

Windows PowerShell authentication example

See the PowerShell section below if cURL is not yet available on your operating system.

These commands work on Windows Command Prompt with the respective JSON files: session.json and login.json
Filename: session.json

```
{    "type": "APISession",    "version": {        "type": "APIVersion",        "major
": 1,        "minor": 7,        "micro": 0    }}
```

Filename: login.json

```
{"type": "LoginRequest","username": "admin","password": "delphix"}
```

(In Powershell you use curl.exe or modify the default alias)...

```
curl.exe --insecure -c cookies.txt -sX POST -H "Content-Type: application/json" -d
"@session.json" http://172.16.160.195/resources/json/delphix/sessioncurl.exe --
insecure -b cookies.txt -c cookies.txt -sX POST -H "Content-Type: application/json"
-d "@login.json" http://172.16.160.195/resources/json/delphix/logincurl.exe --
insecure -b cookies.txt -sX GET -H "Content-Type: application/json" -k http://
172.16.160.195/resources/json/delphix/system
```

Putting the above commands within a Powershell script:

Filename: auth1.ps1

```
<#Filename: auth.ps1Description: Delphix Powershell Sample Authentication
Script ...Date: 2016-08-02Author: Bitt...#>
```

Variables ...

```
$nl = [Environment]::NewLine$BaseURL = "http://172.16.160.195/resources/json/delphix"$
cookie = "cookies.txt"
```

Session JSON Data ...

```
write-output "${nl}Creating session.json file ..."$json = @"{    "type": "APISession"
,    "version": {        "type": "APIVersion",        "major": 1,        "minor":
11,        "micro": 6    }}"@
```

Output File using UTF8 encoding ...

```
write-output $json | Out-File "session.json" -encoding utf8
```

Delphix cURL Session API ...

```
write-output "${nl}Calling Session API ...${nl}"$results = (curl.exe --insecure -c
"$cookie" -sX POST -H "Content-Type: application/json" -d "@session.json" -k
$BaseURL/session)write-output "Session API Results: ${results}"
```

Login JSON Data ...

```
write-output "${nl}Creating login.json file ..."$user = "admin"$pass = "delphix"$json
= @"{    "type": "LoginRequest",    "username": "${user}",    "password": "${pass}"}"@
```

Output File using UTF8 encoding ...

```
write-output $json | Out-File "login.json" -encoding utf8
```

Delphix cURL Login API ...

```
write-output "${nl}Calling Login API ...${nl}"$results = (curl.exe --insecure -b
"$cookie" -c "$cookie" -sX POST -H "Content-Type: application/json" -d "@login.json"
 -k $BaseURL/login)write-output "Login API Results: ${results}"
```

Delphix cURL system API ...

```
write-output "${nl}Calling System API ...${nl}"$results = (curl.exe --insecure -b
"$cookie" -sX GET -H "Content-Type: application/json" -k $BaseURL/system)write-output
"System API Results: ${results}"
```

The end is near ...

```
echo "${nl}Done ...${nl}"exit;
```

Sample Powershell Script Output:

```
PS> . .\auth.ps1Creating session.json file ...Calling Session API ...Session API
Results:{"type":"OKResult","status":"OK","result":{"type":"APISession","version":{"ty
pe":"APIVersion","major":1,"minor":7,"micro":0},"locale":null,"client":null},"job":
null,"action":null}Creating login.json file ...Calling Login API ...Login API
Results: {"type":"OKResult","status":"OK","result":"USER2","job":null,"action":null}
Calling System API ... System API Results: {"type":"OKResult","status":"OK","result":
{"type":"SystemInfo","productType":"standard","productName":"Delphix Engine","buildTi
tle":"Delphix Engine 5.1.1.0","buildTimestamp":"20160721T07:23:41.000Z","buildVersion
":{"type":"VersionInfo","major":5,"minor":1,"micro":1,"patch":0},"configured":true,"e
nabedFeatures":["XPP","MSSQLHOOKS"],"apiVersion":{"type":"APIVersion","major":1,"mino
r":8,"micro":0},"banner":null,"locals":["enUS"],"currentLocale":"enUS","hostname":"De
lphix5110HWv8","sshPublicKey":"ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQDOsrp7Aj6hFQh9yBq7273B+qtPKmCu1B18nPvr08yjt/
IZeM4qKk7caxExQS9rpfU8AWoT7e8ESV7NkBmUzOHrHnLsuJtxPqeYoqeMubVxYjJuxlH368sZuYsnB04KM0m
i39e15lxVGvxQk9tyMpl7gs7cXRz1k6puncyiczU/
axGq7ALHU2uyQoVmlPasuHJbq23d21VAYLuscbtgpZLAFlR8eQH5Xqaa0RT+aQJ6B1ihZ7S0ZN914M2gZHHNY
cSGDWZHwUnBGttnxx1ofRcyN4/qwT5iHq5kjApjSaNgSAU0ExqDHiqgTq0wttf5nltCqGMTFR7XY38HiNq+
+atDroot@Delphix5110HWv8\n","memorySize":8.58107904E9,"platform":"VMware with BIOS
date 05/20/2014","uuid":"564d7e1df4cb-f91098fd348d74817683","processors":[{"type":"CP
UInfo","speed":2.5E9,"cores":1}],"storageUsed":2.158171648E9,"storageTotal":2
.0673724416E10,"installationTime":"2016-07-27T13:28:46.000Z"},"job":null,"action":
null} Done ... PS>
```

API shell scripts programming language examples

Why use shell scripts?

Shell scripts are great tools for rapid development and validation for simple (smaller) requirements. Most development is done iteratively, and shell scripts provide immediate feedback on logic and code.

Company-supported programming languages are the preferred tools for enterprise applications.

Most likely, your company employs more Java and/or PHP programmers than Linux Shell script programmers.

The "use cases" will be programmed using either Unix/Linux Shell and/or Windows PowerShell scripts. You can easily port the logic from these scripts into your favorite programming language. Basic examples of connection with API will be provided for a number of major programming languages in a later section of this document.

Linux/Unix/(and Mac too) shell scripts

There are numerous shell environments, including sh, bash, csh, ksh, and tsh. Identify the current shell environment using any of the commands below:

```
ps -p $$
ps -p $$ -ocomm=
echo $0
ps -ef | grep $$ | grep -v grep
ps -ef | egrep "^\s*\d+\s+$$\s+"
```

The examples provided have all been run from the bash shell environment and may or may not run the same as the other shells. We recommend that you start a bash shell by typing **bash** at the operating system prompt, like this:

```
Operating_System_Prompt> bash
```

The scripts included within this document have all been run on Linux and Mac environments within a bash shell and are NOT certified by any means. As always, test and verify in development for your environment.

For non-Linux platforms and non-bash shell environments, please re-validate for your configuration and search the web for any alternative methods/tools/utilities that may perform the same actions.

Windows PowerShell

Powershell Open Source is now available for Linux and Mac OS. for more information refer to Microsoft Open Source

Requirements

Windows has a number of versions of Powershell. The minimum version for Delphix is 2.0 for SQL Server 2008 environments. There are numerous enhancements and features with subsequent Powershell versions. Additionally, you must be aware of the architecture of 32bit or 64bit Powershell versions you are running from within.

```
PS> $PSVersionTable.PSVersion

Major Minor Build Revision
----- ----- ----- --------
2 0 -1 -1
```

32bit or 64 bit

If executing Powershell scripts from within Delphix Pre/Post Scripts commands or Delphix hooks, the default Powershell used is 32 bit, whereas the typical default Windows Powershell is 64 bit. However, Powershell allows you to execute 64 bit Powershell command from within the 32 bit environment. Shown below is a simple alias, ps64, to execute 64bit Powershell scripts.

```
PS> set-alias
ps64 "$env:windir\sysnative\WindowsPowerShell\v1.0\powershell.exe"
```

Sample call to execute 64bit Powershell script

```
PS> ps64 [path\to\any_64bit_powershell_script].ps1
```

Courtesy of this article: http://www.gregorystrike.com/2011/01/27/how-to-tell-if-powershell-is-32-bit-or-64-bit/

```
PS> if ($env:Processor_Architecture -eq "x86") { write "running on 32bit" }
else {write "running on 64bit"}
running on 32bit
```

. . . or . . .

```
PS> if ([System.IntPtr]::Size -eq 4) { "32-bit" } else { "64-bit" }
32-bit
```

It is worth noting that the locations of the 32-bit and 64-bit versions of Powershell are somewhat misleading. The 32-bit PowerShell is found at C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe

and the 64-bit PowerShell is at C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe

Execution of scripts security disabled

It is possible to disable Powershell environments on the system. If they are disabled, you will see the following error for any Powershell script that you try to execute.

```
PS> . .
[any_powershell_script].ps1
File [any_powershell_script].ps1 cannot be loaded because the execution of
scripts is disabled on this system. Please see "get-help about signing" for
more details.
At line:1 char:2
+ . <<<< .\ [any_powershell_script].ps1
 + CategoryInfo : NotSpecified: ( [], PSSecurityException
 + FullyQualifiedErrorId : RuntimeException
```

To enable Powershell scripts to be executed, set the execution policy to Yes.

```
PS> set-executionpolicy remotesigned
Execution Policy Change
The execution policy helps protect you from scripts that you do not trust.
Changing the execution policy might expose you to the security risks
described in the about_Execution_Policies help topic. Do you want to change
the execution policy?
[Y] Yes [N] No [S] Suspend [?] Help (default is "Y"): Y
PS>
```

Now your Powershell scripts will be executed.

curl.exe

Not all Windows platforms have the cURL executable installed, the easiest way to install and use cURL on Windows is as follows:

1. Download the package from https://curl.se/windows/ and unzip.
2. In the "bin" folder find "curl.exe" and move it to "C:\Windows\System32".

Then you will be able to use the curl command from the Windows Command Prompt or PowerShell console.

```
PS> curl.exe --version
curl 7.49.1 (x86_64-w64-mingw32) libcurl/7.49.1 OpenSSL/1.0.2h zlib/1.2.8
libidn/1.32 libssh2/1.7.0 nghttp2/1.12.0 librtmp/2.3
Protocols: dict file ftp ftps gopher http https imap imaps ldap ldaps pop3
pop3s rtmp rtsp scp sftp smtp smtps telnet tftp
Features: IDN IPv6 Largefile SSPI Kerberos SPNEGO NTLM SSL libz TLS-SRP HTTP2
Metalink
```

Invoking the curl or curl.exe from Powershell command line.

```
PS> Get-Command curl

CommandType Name ModuleName
----------- ---- ----------
Alias curl -> Invoke-WebRequest


PS> Get-Command curl.exe


CommandType Name ModuleName
----------- ---- ----------
Application curl.exe
```

If the alias curl name is to the Invoke-WebRequest, you will need to use the curl.exe command explicitly or remove the alias.

```
PS> Remove-item alias:curl
```

Verify that curl and/or curl.exe work from the respective Powershell environment:

```
PS> curl.exe --version
curl 7.49.1 (x86_64-w64-mingw32) ...

PS> curl --version
curl 7.49.1 (x86_64-w64-mingw32) ...
```

JSON parsing

Unix/Linux/Mac shell

Unix/Linux tools come natively with a host of shell utilities that one can use for parsing out the desired name/value pairs. Tools include sed, awk, cut, tr, and grep, to name a few. System administrators use these utilities frequently and may be able to assist with the methods for parsing JSON strings. For more information please refer to Parsing JSON with UNIX tools and Extract a JSON value from a BASH script

Basic awk and sed parsing

```
json='{"type":"OKResult","status":"OK","result":
{"type":"Job","reference":"JOB-53","namespace":null,"name":null,"actionType":"DB_SYNC
","target":"ORACLE_DB_CONTAINER-9","targetObjectType":"OracleDatabaseContainer","jobS
tate":"RUNNING","startTime":"2016-08-12T19:58:59.811Z","updateTime":"2016-08-12T19:58
:59.828Z","suspendable":true,"cancelable":true,"queued":false,"user":"USER-2","emailA
ddresses":null,"title":"Run SnapSync for database
\"VDPXDEV1\".","percentComplete":0.0,"targetName":"Oracle_Source/VDPXDEV1","events":
[{"type":"JobEvent","timestamp":"2016-08-12T19:58:59.840Z","state":null,"percentCompl
ete":0.0,"messageCode":"event.job.started","messageDetails":"DB_SYNC job started for
\"Oracle_Source/
VDPXDEV1\".","messageAction":null,"messageCommandOutput":null,"diagnoses":
[],"eventType":"INFO"}],"parentActionState":"WAITING","parentAction":"ACTION-238"},"j
ob":null,"action":null}' echo $json | sed -e 's/[{}]/''/g' | awk -v RS=',' -F:
'{print $1 $2}'"type""OKResult""status""OK""result""type""reference""JOB-53""namespac
e"null"name"null"actionType""DB_SYNC""target""ORACLE_DB_CONTAINER-9"
"targetObjectType""OracleDatabaseContainer""jobState""RUNNING""startTime"
"2016-08-12T19"updateTime""2016-08-12T19"suspendable"true"cancelable"true"queued"
false"user""USER-2""emailAddresses"null"title""Run SnapSync for database
\"VDPXDEV1\".""percentComplete"0.0"targetName""Oracle_Source/VDPXDEV1""events"["type""
timestamp""2016-08-12T19"state"null"percentComplete"0.0"messageCode""
event.job.started""messageDetails""DB_SYNC job started for \"Oracle_Source/
VDPXDEV1\".""messageAction"null"messageCommandOutput"null"diagnoses"[]"eventType"
"INFO"]"parentActionState""WAITING""parentAction""ACTION-238""job"null"action"null
```

Find jobState. Print the second argument, and remove the double-quotes.

```
echo $json | sed -e 's/[{}]/''/g' | sed s/\"//g | awk -v RS=',' -F: '$1=="jobState"{p
rint $2}' RUNNING
```

The first sed removed the brackets and braces. The second sed removes the double-quotes. The awk command parses the line by comma delimiters and then parses each line by the semi-colon delimiter and if the first variable $1 is equal to the **jobState** value then print the second $2 variable.

If the results contain an array of values, then you need to loop through each set and parse out the desired value. For example,

```
json='{"type":"ListResult","status":"OK","result":
[{"type":"WindowsHostEnvironment","reference":"WINDOWS_HOST_ENVIRONMENT-1","namespace
":null,"name":"Window
Target","description":"","primaryUser":"HOST_USER-1","enabled":false,"host":"WINDOWS_
```

```
HOST-1","proxy":null},
{"type":"UnixHostEnvironment","reference":"UNIX_HOST_ENVIRONMENT-3","namespace":null,
"name":"Oracle
Target","description":"","primaryUser":"HOST_USER-3","enabled":true,"host":"UNIX_HOST
-3","aseHostEnvironmentParameters":null}],"job":null,"action":null,"total":2,"overflo
w":false}'
```

Parse out array object into separate lines

```
SOURCE_ENV="Oracle Target"lines=`echo ${json} | cut -d "[" -f2 | cut -d "]" -f1 | awk
-v RS='},{}' -F: '{print $0}' `while read -r linedo    #echo "Processing $line"    #e
cho $line | sed -e 's/[{}]/''/g' | sed s/\"//g | awk -v RS=',' -F: '$1=="name"{print
$2}'    TMPNAME=`echo $line | sed -e 's/[{}]/''/g' | sed s/\"//g | awk -v RS=',' -F:
'$1=="name"{print $2}' `    #echo "Name: |${TMPNAME}| |${SOURCE_ENV}|"    if [[ "$
{TMPNAME}" == "${SOURCE_ENV}" ]]    then        echo $line | sed -e 's/[{}]/''/g' |
sed s/\"//g | awk -v RS=',' -F: '$1=="primaryUser"{print $2}'        PRI_USER=`echo
$line | sed -e 's/[{}]/''/g' | sed s/\"//g | awk -v RS=',' -F: '$1=="primaryUser"{pri
nt $2}' `        break    fidone <<< "$(echo -e "$lines")" echo "primaryUser
reference: ${PRI_USER}"
```

Output:

```
primaryUser reference: HOST_USER-3
```

The above methods will be used within the sample scripts since they use the native Linux tools. They typically do not require you to load extra packages or libraries onto the system.

There are a number of open-source utilities designed to simplify the parsing of JSON, such as *jsawk* and *jq*.

jsawk

Linux:

- jswak

Mac:

- jswak on MAC

jq

Reference- https://stedolan.github.io/jq/

64-bit system:

- wget https://github.com/stedolan/jq/releases/download/jq-1.6/jq-linux64
- chmod +x ./jq
- sudo cp jq /usr/local/bin

Older versions:

- Reference- https://stedolan.github.io/jq/download/

Another method is to use an existing programming language typically available with your native operating systems, such as Perl or Python.

```
$ which perl/usr/bin/perl$ which python/usr/bin/python
```

Example: Use Python to pretty format the JSON data string.

Pretty JSON using Python ...

```
json='{"type":"OKResult","status":"OK","result":
{"type":"SystemInfo","productType":"standard","productName":"Delphix
Engine","buildTitle":"Delphix Engine
5.1.1.0","buildTimestamp":"20160721T07:23:41.000Z","buildVersion":
{"type":"VersionInfo","major":5,"minor":1,"micro":1,"patch":0},"configured":true,"ena
bedFeatures":["XPP","MSSQLHOOKS"],"apiVersion":
{"type":"APIVersion","major":1,"minor":8,"micro":0},"banner":null,"locals":
["enUS"],"currentLocale":"enUS","hostname":"Delphix5110HWv8","sshPublicKey":"ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQDOsrp7Aj6hFQh9yBq7273B+qtPKmCu1B18nPvr08yjt/
IZeM4qKk7caxExQS9rpfU8AWoT7e8ESV7NkBmUzOHrHnLsuJtxPqeYoqeMubVxYjJuxlH368sZuYsnB04KM0m
i39e15lxVGvxQk9tyMpl7gs7cXRz1k6puncyiczU/
axGq7ALHU2uyQoVmlPasuHJbq23d21VAYLuscbtgpZLAFlR8eQH5Xqaa0RT+aQJ6B1ihZ7S0ZN914M2gZHHNY
cSGDWZHwUnBGttnxx1ofRcyN4/qwT5iHq5kjApjSaNgSAU0ExqDHiqgTq0wttf5nltCqGMTFR7XY38HiNq+
+atDroot@Delphix5110HWv8\n","memorySize":8.58107904E9,"platform":"VMware with BIOS
date 05/20/2014","uuid":"564d7e1df4cb-f91098fd348d74817683","processors":
[{"type":"CPUInfo","speed":2.5E9,"cores":1}],"storageUsed":2.158171648E9,"storageTota
l":2.0673724416E10,"installationTime":"2016-07-27T13:28:46.000Z"},"job":null,"action"
:null}'
```

Pipe the JSON data to Python programming language to pretty up the format the output for the $json string/data.

```
$ echo $json | python -mjson.tool{    "action": null,     "job": null,     "result":
{        "apiVersion": {            "major": 1,             "micro": 0,             "
minor": 8,             "type": "APIVersion"        },         "banner": null,
        "buildTimestamp": "20160721T07:23:41.000Z",         "buildTitle": "Delphix
Engine 5.1.1.0",         "buildVersion": {            "major": 5,             "micro":
 1,             "minor": 1,             "patch": 0,             "type": "VersionInfo"
       },         "configured": true,         "currentLocale": "enUS",         "enabe
dFeatures": [            "XPP",             "MSSQLHOOKS"    ],     "hostname":
"Delphix5110HWv8",     "installationTime": "2016-07-27T13:28:46.000Z",     "locals":
[        "enUS"    ],     "memorySize": 8581079040.0,     "platform": "VMware with
BIOS date 05/20/2014",     "processors": [        {            "cores": 1,
           "speed": 2500000000.0,             "type": "CPUInfo"        }    ],     "
productName": "Delphix Engine",     "productType": "standard",     "sshPublicKey":
"ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDOsrp7Aj6hFQh9yBq7273B+qtPKmCu1B18nPvr08yjt/
IZeM4qKk7caxExQS9rpfU8AWoT7e8ESV7NkBmUzOHrHnLsuJtxPqeYoqeMubVxYjJuxlH368sZuYsnB04KM0m
i39e15lxVGvxQk9tyMpl7gs7cXRz1k6puncyiczU/
axGq7ALHU2uyQoVmlPasuHJbq23d21VAYLuscbtgpZLAFlR8eQH5Xqaa0RT+aQJ6B1ihZ7S0ZN914M2gZHHNY
cSGDWZHwUnBGttnxx1ofRcyN4/qwT5iHq5kjApjSaNgSAU0ExqDHiqgTq0wttf5nltCqGMTFR7XY38HiNq+
+atDroot@Delphix5110HWv8\n",    "storageTotal": 20673724416.0,     "storageUsed":
2158171648.0,     "type": "SystemInfo",     "uuid": "564d7e1df4cb-
f91098fd348d74817683"    },     "status": "OK",     "type": "OKResult"}
```

jq parser

The "jq" command line parser is available on Unix, Linux, Mac, and Windows platforms. Typically, for Windows, the built-in ConvertFrom/To-Json object parser will be used. "jq" is being included in most native Linux distributions and is easy to install on the Mac OS.

References:

Parsing JSON with jq

Installation FAQ

Mac Installation

Example:

```
json='{"type":"ListResult","status":"OK","result":
[{"type":"OracleLinkedSource","reference":"ORACLE_LINKED_SOURCE-52","namespace":null,"name":"DPXDEV0
1","description":null,"virtual":false,"restoration":false,"staging":false,"container":"ORACLE_DB_CON
TAINER-120","config":"ORACLE_SINGLE_CONFIG-40","status":"DEFAULT","runtime":
{"type":"OracleSourceRuntime","status":"RUNNING","accessible":true,"databaseSize":2.409529344E9,"not
AccessibleReason":null,"databaseMode":"READ_WRITE","lastNonLoggedLocation":"0","activeInstances":
[{"type":"OracleActiveInstance","instanceNumber":1,"instanceName":"DPXDEV01","hostName":"linuxtarget
.delphix.local"}],"databaseStats":null,"bctEnabled":true,"racEnabled":null,"dnfsEnabled":false,"arch
ivelogEnabled":null},"backupLevelEnabled":false,"rmanChannels":2,"filesPerSet":5,"checkLogical":fals
e,"externalFilePath":null,"encryptedLinkingEnabled":false,"compressedLinkingEnabled":true,"bandwidth
Limit":0,"numberOfConnections":1,"enabled":true,"preScript":"","postScript":"","role":"PRIMARY"},
{"type":"OracleVirtualSource","reference":"ORACLE_VIRTUAL_SOURCE-25","namespace":null,"name":"VBITT"
,"description":null,"virtual":true,"restoration":false,"staging":false,"container":"ORACLE_DB_CONTAI
NER-121","config":"ORACLE_SINGLE_CONFIG-47","status":"DEFAULT","runtime":
{"type":"OracleSourceRuntime","status":"RUNNING","accessible":true,"databaseSize":2.410053632E9,"not
AccessibleReason":null,"databaseMode":"READ_WRITE","lastNonLoggedLocation":"0","activeInstances":
[{"type":"OracleActiveInstance","instanceNumber":1,"instanceName":"VBITT","hostName":"linuxtarget.de
lphix.local"}],"databaseStats":[{"type":"OracleDatabaseStatsSection","sectionName":"Open
Transactions","columnHeaders":["Transaction Count"],"rowValues":
[{"type":"OracleDatabaseStatistic","statisticValues":["0"]}]},
{"type":"OracleDatabaseStatsSection","sectionName":"Session Statistics","columnHeaders":["Current
Session","Total Session","High Watermark"],"rowValues":
[{"type":"OracleDatabaseStatistic","statisticValues":["2","46","5"]}]},
{"type":"OracleDatabaseStatsSection","sectionName":"Top Wait Events","columnHeaders":["Event","Wait
Count","Total Wait Time (s)"],"rowValues":[{"type":"OracleDatabaseStatistic","statisticValues":
["Disk file operations I/O","13","13"]},{"type":"OracleDatabaseStatistic","statisticValues":["log
file sequential read","11","12"]},{"type":"OracleDatabaseStatistic","statisticValues":["control file
parallel write","8","8"]},{"type":"OracleDatabaseStatistic","statisticValues":["control file
sequential read","6","3"]},{"type":"OracleDatabaseStatistic","statisticValues":["ARCH wait for
process start 3","2","2"]},{"type":"OracleDatabaseStatistic","statisticValues":["db file sequential
read","9","1"]},{"type":"OracleDatabaseStatistic","statisticValues":["rdbms ipc reply","1","1"]},
{"type":"OracleDatabaseStatistic","statisticValues":["JS coord start wait","1","1"]},
{"type":"OracleDatabaseStatistic","statisticValues":["os thread startup","2","0"]},
{"type":"OracleDatabaseStatistic","statisticValues":["Parameter File I/O","1","0"]}]},
{"type":"OracleDatabaseStatsSection","sectionName":"Top SQL by CPU","columnHeaders":["Percentage of
Load","SQL Statement"],"rowValues":
[]}],"bctEnabled":false,"racEnabled":null,"dnfsEnabled":false,"archivelogEnabled":null},"operations"
:{"type":"VirtualSourceOperations","configureClone":[],"preRefresh":[],"postRefresh":
[]},"mountBase":"/mnt/provision","fileMappingRules":null,"manualProvisioning":null,"configParams":
{"memory_target":"1191182336","processes":"150","log_archive_dest_1":"location=/mnt/provision/VBITT/
archive/
MANDATORY","_omf":"ENABLED","filesystemio_options":"setall","compatible":"11.2.0.4.0","audit_trail":
"NONE","remote_login_passwordfile":"EXCLUSIVE","open_cursors":"300","audit_sys_operations":"FALSE"},
"configTemplate":null,"nodeListenerList":
[],"enabled":true,"role":"PRIMARY"}],"job":null,"action":null,"total":2,"overflow":false}'
```

We have a very big JSON string above. Let's perform some basic jq parsing.

1. Pipe JSON string into jq command line parser.

```
ActionScript

echo $json | jq '.'
```

2. The output is a pretty human-readable JSON formatted string.

3. Get the first-level status value (. . . ,"status":"OK", . . .)

```
ActionScript

echo $json | jq '.status'"OK"
```

4. Get raw values (not quoted).

```
ActionScript

echo $json | jq --raw-output '.status'OK
```

5. Get a number of rows returned for the type equal to "ListResult" API returned request.

```
ActionScript

echo $json | jq --raw-output '.total'2
```

6. Get the first result set.

```
ActionScript

echo $json | jq '.result[0] '{    "type": "OracleLinkedSource",    "reference":
"ORACLE_LINKED_SOURCE-52",    "namespace": null,    "name": "DPXDEV01",    "des
cription": null,    "virtual": false,    "restoration": false,    "staging":
false,    "container": "ORACLE_DB_CONTAINER-120",    "config":
"ORACLE_SINGLE_CONFIG-40",    "status": "DEFAULT",    "runtime": {        "type
": "OracleSourceRuntime",        "status": "RUNNING",        "accessible": true,
        "databaseSize": 2409529344,        "notAccessibleReason": null,        "
databaseMode": "READ_WRITE",        "lastNonLoggedLocation": "0",        "activ
eInstances": [            {                "type": "OracleActiveInstance",
           "instanceNumber": 1,                "instanceName": "DPXDEV01",
           "hostName": "linuxtarget.delphix.local"            }        ],
  "databaseStats": null,        "bctEnabled": true,        "racEnabled": null,
      "dnfsEnabled": false,        "archivelogEnabled": null    },    "backupL
evelEnabled": false,    "rmanChannels": 2,    "filesPerSet": 5,    "checkLogica
l": false,    "externalFilePath": null,    "encryptedLinkingEnabled": false,
 "compressedLinkingEnabled": true,    "bandwidthLimit": 0,    "numberOfConnecti
ons": 1,    "enabled": true,    "preScript": "",    "postScript": "",    "role":
 "PRIMARY"}
```

7. Get the first result set name value.

```
ActionScript

echo $json | jq --raw-output '.result[0].name'DPXDEV01
```

8. Get first result set reference value.

```
ActionScript

echo $json | jq --raw-output '.result[0].reference'
```

9. Get first result set name=value pairs.

```
ActionScript

echo $json | jq '.result[0]' | jq -r "to_entries|map(\"(.key)=(.value|tostring)
\")|.[]" | grep container container=ORACLE_DB_CONTAINER-120
```

10. Get ALL result sets name values.

```
ActionScript

echo $json | jq '.result[].name'"DPXDEV01""VBITT"
```

11. Get ALL result sets "reference" and "container" values.

```
ActionScript

echo $json | jq '.result[].reference,.result[].container'"ORACLE_LINKED_SOURCE-
52""ORACLE_VIRTUAL_SOURCE-25""ORACLE_DB_CONTAINER-120""ORACLE_DB_CONTAINER-121"
```

12. Now, let's scan ALL result sets for a conditional match and return a related value.

```
echo $json | jq --raw-output '.result[] | select(.name=="VBITT") | .container'
 ORACLE_DB_CONTAINER-121echo $json | jq --raw-output '.result[] |
select(.name=="VBITT") | .reference' ORACLE_VIRTUAL_SOURCE-25echo $json | jq --raw-
output '.result[] | select(.name=="VBITT")
| .container, .reference'ORACLE_DB_CONTAINER-121ORACLE_VIRTUAL_SOURCE-25
```

This is the typical usage for Delphix, where the human-readable name is provided and we need to look up the object reference, container, status, etc. for the respective name. Some object references are based on expressions such as "and" or "or" conditions.

```
echo $json | jq --raw-output '.result[] |
select(.environment=="UNIX_HOST_ENVIRONMENT-9" and .name=="/u02/ora/app/product/
11.2.0/dbhome_1" ) | .reference '
```

In this case, the jq select command has an "and" condition in order to correctly identify the target result object index. This is important for getting the correct and single return value for `| .reference`, since there might be more than one instance within the environment.

For a working example of using the jq JSON parser, see the VDB Init using jq command-line JSON Parser use case, **Filename: vdb_init.sh**. A version of all the Unix/Linux/Mac shell scripts exists within the code provided. It contains the ***_jq.sh** within the filename.

PowerShell

Starting with Powershell 3.0, there are ConvertFrom-Json and ConvertTo-Json modules/commands to parse the JSON string data to/from objects. If you are stuck with Powershell 2.x., the next section provides similar functions as a method of working with JSON strings.

> ⚠ These 2.x functions are not 100% the same as the Powershell 3.0 ConvertFrom-Json/ConvertTo-Json modules.

PowerShell 2 example

Filename: parse_2.0.ps1

For Powershell 2.0, there are no JSON-provided functions or commands, so the following will serialize the JSON data to a serialized array.

```
function ConvertTo-Json20([object] $item){    add-type -assembly
system.web.extensions    $ps_js=new-object
system.web.script.serialization.javascriptSerializer    return
 $ps_js.Serialize($item)}function ConvertFrom-Json20([object] $item){     add-type
-assembly system.web.extensions    $ps_js=new-object
system.web.script.serialization.javascriptSerializer    # The comma operator is the
array construction operator in PowerShell    return ,$ps_js.DeserializeObject($item)}
```

Use the JSON from the system API Call.

```
$json='{"type":"OKResult","status":"OK","result":
{"type":"SystemInfo","productType":"standard","productName":"Delphix
Engine","buildTitle":"Delphix Engine
5.1.1.0","buildTimestamp":"20160721T07:23:41.000Z","buildVersion":
{"type":"VersionInfo","major":5,"minor":1,"micro":1,"patch":0},"configured":true,"ena
bedFeatures":["XPP","MSSQLHOOKS"],"apiVersion":
{"type":"APIVersion","major":1,"minor":8,"micro":0},"banner":null,"locals":
["enUS"],"currentLocale":"enUS","hostname":"Delphix5110HWv8","sshPublicKey":"ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQDOsrp7Aj6hFQh9yBq7273B+qtPKmCu1B18nPvr08yjt/
IZeM4qKk7caxExQS9rpfU8AWoT7e8ESV7NkBmUzOHrHnLsuJtxPqeYoqeMubVxYjJuxlH368sZuYsnB04KM0m
i39e15lxVGvxQk9tyMpl7gs7cXRz1k6puncyiczU/
axGq7ALHU2uyQoVmlPasuHJbq23d21VAYLuscbtgpZLAFlR8eQH5Xqaa0RT+aQJ6B1ihZ7S0ZN914M2gZHHNY
cSGDWZHwUnBGttnxx1ofRcyN4/qwT5iHq5kjApjSaNgSAU0ExqDHiqgTq0wttf5nltCqGMTFR7XY38HiNq+
+atDroot@Delphix5110HWv8\n","memorySize":8.58107904E9,"platform":"VMware with BIOS
date 05/20/2014","uuid":"564d7e1df4cb-f91098fd348d74817683","processors":
[{"type":"CPUInfo","speed":2.5E9,"cores":1}],"storageUsed":2.158171648E9,"storageTota
l":2.0673724416E10,"installationTime":"2016-07-27T13:28:46.000Z"},"job":null,"action"
:null}'
```

Convert the JSON string.

> ⚠ The job and action are null values.

```
PS> $o = ConvertFrom-Json20 $jsonPS> $o Key                Value---
                -----
type            OKResultstatus            OKresult            {[type, SystemInfo],
[productType, standard], [productNa...jobaction
```

Extract the result JSON string array.

```
PS> $a = $o.resultPS> $a Key                    Value---                      -----
type                SystemInfoproductType            standardproductName           D
elphix EnginebuildTitle            Delphix Engine 5.1.1.0buildTimestamp        201607
21T07:23:41.000ZbuildVersion        {[type, VersionInfo], [major, 5], [minor, 1],
[micro, 1]...configured            TrueenabedFeatures        {XPP, MSSQLHOOKS}
apiVersion            {[type, APIVersion], [major, 1], [minor, 8], [micro, 0]}
bannerlocals                {enUS}
currentLocale        enUShostname            Delphix5110HWv8sshPublicKey        ssh-
rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQDOsrp7Aj6hFQh9yBq7...memorySize            8581079040pl
atform            VMware with BIOS date 05/20/2014uuid               564d7e1df4cb-
f91098fd348d74817683processors            {System.Collections.Generic.Dictionary`2
[System.String,S...storageUsed            2158171648storageTotal        20673724416in
stallationTime    2016-07-27T13:28:46.000Z
```

Same output as above.

```
PS> foreach ($element in $a) {$element}PS> $a.typeSystemInfoPS> $a.buildTitleDelphix
Engine 5.1.1.0PS> $a.hostnameDelphix5110HWv8
```

Extract the result.buildVersion object.

```
PS> $a1 = $o.result.buildVersionPS> $a1Key                    Value---
                    -----
type                VersionInfomajor                5minor                1micro
          1patch                0PS> $a1.major 5
```

Extract the result.processors array collection.

```
PS> $b = $o.result.processors PS> $b Key                        Value---
                      -----
    type                    CPUInfospeed                    2500000000cores
          1PS> $a -is [Array] FalsePS> $a -is [Object] TruePS> $b -is [Array] True
```

Convert Array Collection to Object.

```
PS> $b1 = $b | Select-Object PS> $b1 Key                    Value---
                    ------
type                CPUInfospeed                2500000000cores                1PS>
  $b1.type CPUInfoPS> $b1.speed2500000000
```

PowerShell 3 or greater example

Starting with Powershell 3.0, there is are ConvertFrom-Json and ConvertTo-Json commands to parse the JSON data to/from objects.

Reference:

- ConvertFrom-Json
- ConvertTo-Json

```
$o = $json | ConvertFrom-Json
```

There are a number of tutorials and functional examples on the web. Below is an excerpt from the Powershell introduction video for Linux / Mac Open Source announcement.

Powershell JSON ConvertTo-Json and Python Example 15:55 through 21:16

The concept is straightforward:

- The `ConvertFrom-Json` JSON string is converted into a Powershell object that you can reference directly.
- The `ConvertTo-Json` takes the JSON object and converts it to a string.

JSON parsing from within programming languages

Most programming languages provide their own libraries, functions, and methods for parsing JSON data strings into objects/hashes/arrays/xml that the native programming language can easily process.

API use case commands and scripts

Sample script parsers

The Delphix Use Cases scripts provided use the native operating system "curl" command and then a JSON parser, depending on the engine on which you are runnning the scripts.

For Unix/Linux/Mac, the scripts provided use both native shell commands and/or the jq parser program commands. Subroutines have been provided for both methods:

```
Native Shell commands:        parseJSON_subroutines.sh
jq Parser commands:            jqJSON_subroutines.sh
```

For Windows, the scripts are for Powershell 2.0 and utilitze the custom `ConvertFrom-Json20` and `ConvertTo-Json20` functions provided. As noted, with Powershell 3.0, there are `ConvertFrom-Json` and `ConvertTo-Json` command-lets provided by Powershell.

Using the jq parser

These are some of the jq commands used within the scripts. The first is a shell script subroutine which is used for finding and returning a single item value.

Filename: jqJSON_subroutines.sh

Subroutines

This code requires the jq Linux/Mac JSON parser program

```
jqParse() {
    STR=$1                    # json string
    FND=$2                    # name to find
    RESULTS=""                # returned name value
    RESULTS=`echo $STR | jq --raw-output '."$FND"'`
    #echo "Results: ${RESULTS}"
    if [ "${FND}" == "status" ] && [ "${RESULTS}" != "OK" ]
    then
        echo "Error: Invalid Satus, please check code ... ${STR}"
        exit 1;
    elif [ "${RESULTS}" == "" ]
    then
        echo "Error: No Results ${FND}, please check code ... ${STR}"
        exit 1;
    fi
    echo "${RESULTS}"
}
```

The subroutine is called from the shell script to return values based on the key (or name) value provided.

Usage – After every curl command, check that the returned status value is "OK".

```
RESULTS=$( jqParse "${STATUS}" "status" )
```

Call the jqParse subroutine, where

- ${STATUS} is the returned JSON string from the curl command, and
- the value we want returned is where the name/key is equal to "status"

Usage – Get a single value within the returned nested result object:

```
JOBSTATE=$( jqParse "${JOB_STATUS}" "result.jobState" )
```

Call the jqParse subroutine, where

- ${JOB_STATUS} is the returned JSON string from the cURL command, and
- the value we want returned is where the name/key is equal to "jobState" with the nested ".result" object.

Usage – Find name/value result object and return another value within the select result object:

Use jq to parse out container reference for name of $SOURCE_SID ...

```
CONTAINER_REFERENCE=`echo ${STATUS} | jq --raw-output '.result[] | select(.name=="'"$
{SOURCE_SID}"'") | .reference '`
```

where

- ${STATUS} is the returned JSON string from the cURL command, and
- the value we want returned is based on the selected nested result object where the .result[].name is equal to "${SOURCE_SID}" and return the .reference value for the selected result object.

Delphix engine use cases

Delphix user session timeout

Some activities can take longer than the default 30 minute session timeout value. Therefore, the following script allows you to change the timeout value using the RESTFul API. As always, you can change it easily through the CLI.

> ⚠ This code is the first example showing how object references are used for input (either JSON or URL) into API calls. The name will be the DE_USER variable value delphix_admin. The object reference that the code identifies is USER-2, which in this case is passed into the API URL to update the user parameters passed via the JSON string.

Filename: user_timeout.sh

Edit the file to update the parameters as required for your environment.

```
#####################
#    DELPHIX CORP    #
#####################
#Parameter Initialization
DMIP=172.16.160.195
#DMPORT=8282
DMUSER=delphix_admin
DMPASS=delphix
COOKIE="~/cookies.txt"
COOKIE=`eval echo $COOKIE`
CONTENT_TYPE="Content-Type: application/json"
BaseURL="http://${DMIP}/resources/json/delphix"
```

```
#
Required for user timeout ...
#
DE_USER="delphix_admin" # Delphix Engine User
DE_TIMEOUT=120 # Timeout integer in minutes
##########################################
#    NO CHANGES REQUIRED BELOW THIS POINT    #
##########################################
```

Sample Output

```
$ ./user_timeout.sh # or ./user_timeout_jq.sh
Authenticating on http://172.16.160.195/resources/json/delphix
Session and Login Successful ...
user reference: USER-2
Update delphix_admin session timeout value to 120 minutes ...
Returned JSON: {"type":"OKResult","status":"OK","result":"","job":null,"action":"ACTI
ON-423"}
Results: OK
Done ...
$
```

VDB Init (start | stop | enable | disable | status | delete)

This script is used to start, stop, enable, disable, and delete a Delphix platform source object. Typically, this is done on a virtual databases (VDBs), but you can use it for dSources as well.

> ⚠ The `vdb_init.sh` and `vdb_operations.sh` require the "jq" command line json parser.

Filename: vdb_init.sh

The `vdb_init.sh` supports the start, stop, enable, disable, status, and delete command line options.

```
$ ./vdb_init.sh something VBITT
. . .
Unknown option (start | stop | enable | disable | status | delete): something
Exiting ...
$ ./vdb_init.sh status VBITT
database container reference: ORACLE_DB_CONTAINER-121
source reference: ORACLE_VIRTUAL_SOURCE-25
Runtime Status: "INACTIVE"
Enabled: true
Done ...
```

```
$ ./vdb_init.sh start VBITT
database container reference: ORACLE_DB_CONTAINER-121
source reference: ORACLE_VIRTUAL_SOURCE-25
Job: JOB-894
Current status as of Wed Sep 7 16:04:04 EDT 2016 : RUNNING 0% Completed
```

```
Current status as of Wed Sep 7 16:04:14 EDT 2016 : RUNNING 25% Completed
Current status as of Wed Sep 7 16:04:24 EDT 2016 : RUNNING 45% Completed
Job: JOB-894 COMPLETED 100% Completed ...
Done ...
```

```
$ ./vdb_init.sh delete VBITT
database container reference: ORACLE_DB_CONTAINER-123
source reference: ORACLE_VIRTUAL_SOURCE-27
vendor source: OracleVirtualSource
delete parameters type: OracleDeleteParameters
Job: JOB-927
Current status as of Sat Sep 10 12:55:32 EDT 2016 : RUNNING 0% Completed
Current status as of Sat Sep 10 12:55:32 EDT 2016 : RUNNING 0% Completed
Job: JOB-927 COMPLETED 100% Completed ...
Done ...
```

VDB operations (sync, refresh, rollback)

This script is used to perform a sync (snapshot), refresh, or rollback (reset) on the Delphix Engine source object. All these work on a virtual databases (VDBs), but only a sync operation can be used on dSources.

⚠ The `vdb_init.sh` and `vdb_operations.sh` require the "jq" command line json parser.

Filename: vdb_operations.sh

```
$ ./vdb_operations.sh sync VBITT
Session and Login Successful ...
database container reference: ORACLE_DB_CONTAINER-131
provision source container: ORACLE_DB_CONTAINER-129
json> {
    "type": "OracleSyncParameters"
}
Job: JOB-998
Current status as of Wed Sep 14 17:05:00 EDT 2016 : RUNNING 0% Completed
```

```
Current status as of Wed Sep 14 17:05:24 EDT 2016 : RUNNING 97% Completed
Job: JOB-998 COMPLETED 100% Completed ...
Done ...
```

Rollback rewinds the virtual database back to the last point in time within the source TimeFlow.

```
$ ./vdb_operations.sh rollback VBITT
Session and Login Successful ...
database container reference: ORACLE_DB_CONTAINER-131
provision source container: ORACLE_DB_CONTAINER-129
json> {
    "type": "OracleRollbackParameters",
```

```
    "timeflowPointParameters": {
        "type": "TimeflowPointSemantic",
        "container": "ORACLE_DB_CONTAINER-131"
    }
}
Job: JOB-1000
Current status as of Wed Sep 14 17:06:33 EDT 2016 : RUNNING 0% Completed
Current status as of Wed Sep 14 17:06:43 EDT 2016 : RUNNING 0% Completed
Current status as of Wed Sep 14 17:06:53 EDT 2016 : RUNNING 34% Completed
Current status as of Wed Sep 14 17:07:03 EDT 2016 : RUNNING 34% Completed
Current status as of Wed Sep 14 17:07:13 EDT 2016 : RUNNING 58% Completed
Current status as of Wed Sep 14 17:07:23 EDT 2016 : RUNNING 67% Completed
Current status as of Wed Sep 14 17:07:33 EDT 2016 : RUNNING 70% Completed
Current status as of Wed Sep 14 17:07:43 EDT 2016 : RUNNING 70% Completed
Current status as of Wed Sep 14 17:07:53 EDT 2016 : RUNNING 71% Completed
Current status as of Wed Sep 14 17:08:03 EDT 2016 : RUNNING 72% Completed
Current status as of Wed Sep 14 17:08:13 EDT 2016 : RUNNING 73% Completed
Current status as of Wed Sep 14 17:08:23 EDT 2016 : RUNNING 94% Completed
Current status as of Wed Sep 14 17:08:33 EDT 2016 : RUNNING 96% Completed
Current status as of Wed Sep 14 17:08:43 EDT 2016 : RUNNING 96% Completed
Current status as of Wed Sep 14 17:08:53 EDT 2016 : RUNNING 96% Completed
Job: JOB-1000 COMPLETED 100% Completed ...
Done ...
```

Refresh recreates the virtual database to the last point in time in the respective parent, provision source TimeFlow.

```
$ ./vdb_operations.sh refresh VBITT
Session and Login Successful ...
database container reference: ORACLE_DB_CONTAINER-131
provision source container: ORACLE_DB_CONTAINER-129
json> {
    "type": "OracleRefreshParameters",
    "timeflowPointParameters": {
        "type": "TimeflowPointSemantic",
        "container": "ORACLE_DB_CONTAINER-129"
    }
}
Job: JOB-1005
Current status as of Wed Sep 14 17:10:11 EDT 2016 : RUNNING 0% Completed
Current status as of Wed Sep 14 17:10:21 EDT 2016 : RUNNING 0% Completed
Current status as of Wed Sep 14 17:10:31 EDT 2016 : RUNNING 34% Completed
Current status as of Wed Sep 14 17:10:41 EDT 2016 : RUNNING 35% Completed
Current status as of Wed Sep 14 17:10:51 EDT 2016 : RUNNING 70% Completed
Current status as of Wed Sep 14 17:11:01 EDT 2016 : RUNNING 70% Completed
Current status as of Wed Sep 14 17:11:20 EDT 2016 : RUNNING 72% Completed
Current status as of Wed Sep 14 17:11:31 EDT 2016 : RUNNING 73% Completed
Current status as of Wed Sep 14 17:11:41 EDT 2016 : RUNNING 73% Completed
Job: JOB-1005 COMPLETED 100% Completed ...
Done ...
```

Delphix self-service use cases for APIs

Create Delphix self-service template

> ⚠ Jet Stream is now known as Delphix Self-Service.

Filename: jetstream_template.sh or *jetstream_template_jq.sh*

Edit the file to update the parameters as required for your environment.

Required for Delphix Self-Service Template ...

```
TPL_NAME="jstpl"            # JetStream Template Name
DATASOURCE_NAME="jsds"        # JetStream Data Source Name
DATASOURCE_VDB="VBITT"        # JetStream Data Source VDB or dSource
```

Sample Output

```
$ ./jetstream_template.sh# or ./Jetstream_template_jq.sh
Authenticating on http://172.16.160.195/resources/json/delphix
Session and Login Successful ...
Getting Database Container Reference Value ...
container reference: ORACLE_DB_CONTAINER-45
Create JetStream Template jstpl with Data Source DB VBITT ...
Database: {"type":"OKResult","status":"OK","result":"JS_DATA_TEMPLATE-3","job":null,"
action":"ACTION-547"}


Done ... (no job required for this action)
```

Create Delphix self-service data container

Filename: jetstream_container.sh# or *jetstream_container_jq.sh*

Edit the file to update the parameters as required for your environment.

Required for Delphix Self-Service Container ...

```
TPL_NAME="jstpl"        # JetStream Template Name
DS_NAME="jsds"            # JetStream Data Source Name


DC_NAME="jsdc"            # JetStream Data Container Name
DC_VDB="VBITT2"            # JetStream Data Container VDB
```

Sample Output

```
$ ./jetstream_container.sh# or ./jetstream_container_jq.sh
Authenticating on http://172.16.160.195/resources/json/delphix
Session and Login Successful ...
```

```
Getting Database Container Reference Value ...
container reference: ORACLE_DB_CONTAINER-46
JetStream Data Template: JS_DATA_TEMPLATE-4
JetStream sourceDataLayout: JS_DATA_TEMPLATE-4
Create JetStream Container jsdc with Data Source DB VBITT2 ...
JetStream Data Container Creation Results: {"type":"OKResult","status":"OK","result":"
JS_DATA_CONTAINER-4","job":"JOB-240","action":"ACTION-569"}
Job: JOB-240
Current status as of Wed Aug 17 04:11:54 EDT 2016 : RUNNING 0.0% Completed
Current status as of Wed Aug 17 04:11:54 EDT 2016 : RUNNING 0.0% Completed
Current status as of Wed Aug 17 04:12:04 EDT 2016 : RUNNING 5.0% Completed
Current status as of Wed Aug 17 04:12:14 EDT 2016 : RUNNING 5.0% Completed
Current status as of Wed Aug 17 04:12:24 EDT 2016 : RUNNING 30.0% Completed
Current status as of Wed Aug 17 04:12:34 EDT 2016 : RUNNING 31.0% Completed
Current status as of Wed Aug 17 04:12:44 EDT 2016 : RUNNING 53.0% Completed
Current status as of Wed Aug 17 04:12:54 EDT 2016 : RUNNING 57.0% Completed
Current status as of Wed Aug 17 04:13:04 EDT 2016 : RUNNING 57.0% Completed
Current status as of Wed Aug 17 04:13:14 EDT 2016 : RUNNING 57.0% Completed
Current status as of Wed Aug 17 04:13:24 EDT 2016 : RUNNING 59.0% Completed
Current status as of Wed Aug 17 04:13:34 EDT 2016 : RUNNING 60.0% Completed
Current status as of Wed Aug 17 04:13:44 EDT 2016 : RUNNING 77.0% Completed
Current status as of Wed Aug 17 04:13:54 EDT 2016 : RUNNING 77.0% Completed
Current status as of Wed Aug 17 04:14:04 EDT 2016 : RUNNING 77.0% Completed
Current status as of Wed Aug 17 04:14:14 EDT 2016 : RUNNING 77.0% Completed
Job: JOB-240 COMPLETED 100.0% Completed ...


Done ...
```

Create Delphix self-service bookmark

Filename: jetstream_api_examples.txt (part 1)

Create Bookmark ...

Change parameters as required and desired.

```
curl -X POST -k --data @-
http://172.16.160.177/resources/json/delphix/jetstream/bookmark \
    -b cookies.txt -H "Content-Type: application/json" <<EOF
{
    "type": "JSBookmarkCreateParameters",
    "bookmark": {
        "type": "JSBookmark",
        "name": "aalen",
        "branch": "JS_BRANCH-5",
        "shared": false,
        "tags": [
            "A",
            "B",
            "C"
        ]
    },
```

```
    "timelinePointParameters": {
        "type": "JSTimelinePointLatestTimeInput",
        "sourceDataLayout": "JS_DATA_CONTAINER-2"
    }
}
EOF
{"type":"OKResult","status":"OK","result":"JS_BOOKMARK-5","job":"JOB-512","action":
"ACTION-921"}
```

⚠ The timelinePointParameters type "JSTimelinePointLatestTimeInput" is the last point / latest time in the branch!

Filename: jetstream_bookmark.sh or *jetstream_bookmark_jq.sh*

Edit the file to update the parameters as required for your environment.

```
DT=`date '+%Y%m%d%H%M%S'`
```

Required for Delphix Self-Service Bookmark ...

```
JS_BRANCH="default"             # JetStream Branch
BM_NAME="aalen_${DT}"          # JetStream Bookmark Name appended timestamp
SHARED="false"                  # Share Bookmark true/false
TAGS='"API","Created"'          # Tags Array Values
```

Sample Output

```
$ ./jetstream_bookmark.sh # or ./jetstream_bookmark_jq.sh
Authenticating on http://172.16.160.195/resources/json/delphix
Session and Login Successful ...
Getting Jetstream Branch Reference Value ...
branch reference: JS_BRANCH-7
dataLayout container reference: JS_DATA_CONTAINER-4
JetStream Bookmark Creation Results: {"type":"OKResult","status":"OK","result":"JS_BO
OKMARK-4","job":"JOB-251","action":"ACTION-591"}
Job: JOB-251
Current status as of Wed Aug 17 04:59:53 EDT 2016 : COMPLETED 100.0% Completed
Job: JOB-251 COMPLETED 100.0% Completed ...


Done ...
```

Delphix self-service refresh

Filename: jetstream_api_examples.txt (part 2)

Use CLI command to get Delphix Self-Service Container Reference

```
/jetstream/container/list
```

```
            ...
            "reference": " JS_DATA_CONTAINER-4 ",
            "namespace": null,
            "name": " jsdc ",
            ...
```

Refresh Container Information ...

```
=== POST /resources/json/delphix/jetstream/container/ JS_DATA_CONTAINER-4 /refresh


curl -X POST -k --data @http://172.16.160.177/resources/json/delphix/jetstream/
container/ JS_DATA_CONTAINER-4 /refresh \ -b cookies.txt -H "Content-Type:
application/json" <<EOF
{}
EOF


=== RESPONSE ===
{
    "type": "OKResult",
    "status": "OK",
    "result": "",
    "job": "JOB-514",
    "action": "ACTION-924"
}
=== END ===
```

Filename: jetstream_refresh.sh or *jetstream_refresh_jq.sh*

Edit the file to update the parameters as required for your environment.

Required for Delphix Self-Service Refresh ...

```
CONTAINER_NAME="jsdc"        # Jetstream Container Name
```

Sample Output

```
$ ./jetstream_refresh.sh # or ./jetstream_refresh_jq.sh
Authenticating on http://172.16.160.195/resources/json/delphix
Session and Login Successful ...
Getting Jetstream Container Reference Value ...
container reference: JS_DATA_CONTAINER-4
abitterman-mbpro:JetStream abitterman$ vi jetstream_refresh.sh
abitterman-mbpro:JetStream abitterman$ ./jetstream_refresh.sh
Authenticating on http://172.16.160.195/resources/json/delphix
Session and Login Successful ...
Getting Jetstream Container Reference Value ...
container reference: JS_DATA_CONTAINER-4
JetStream Refresh API Results: {"type":"OKResult","status":"OK","result":"","job":"JO
B-257","action":"ACTION-602"}
Job: JOB-257
```

```
Current status as of Wed Aug 17 05:13:15 EDT 2016 : RUNNING 2.0% Completed
Current status as of Wed Aug 17 05:13:15 EDT 2016 : RUNNING 2.0% Completed
Current status as of Wed Aug 17 05:13:25 EDT 2016 : RUNNING 5.0% Completed
Current status as of Wed Aug 17 05:13:35 EDT 2016 : RUNNING 5.0% Completed
Current status as of Wed Aug 17 05:13:45 EDT 2016 : RUNNING 30.0% Completed
Current status as of Wed Aug 17 05:13:55 EDT 2016 : RUNNING 42.0% Completed
Current status as of Wed Aug 17 05:14:05 EDT 2016 : RUNNING 55.0% Completed
Current status as of Wed Aug 17 05:14:15 EDT 2016 : RUNNING 58.0% Completed
Current status as of Wed Aug 17 05:14:25 EDT 2016 : RUNNING 58.0% Completed
Current status as of Wed Aug 17 05:14:35 EDT 2016 : RUNNING 58.0% Completed
Current status as of Wed Aug 17 05:14:45 EDT 2016 : RUNNING 60.0% Completed
Current status as of Wed Aug 17 05:14:55 EDT 2016 : RUNNING 62.0% Completed
Current status as of Wed Aug 17 05:15:05 EDT 2016 : RUNNING 77.0% Completed
Current status as of Wed Aug 17 05:15:15 EDT 2016 : RUNNING 77.0% Completed
Job: JOB-257 COMPLETED 100.0% Completed ...


Done ...
```

Masking use cases

Masking API client

The Continuous Compliance Engine now features an interactive API client that can generate commands specific to your masking engine. With those commands, you can:

- make changes to your engine
- copy and paste the commands to write code that can automate your masking activities

The API client will make real changes to your virtual machine. Any operations you run using the API Client will persist on the machine!

To access the Masking API client, use the following URL: http://myMaskingEngine.com/masking/api-client/, replacing "myMaskingEngine.com" with the hostname or IP address of your virtual machine.

For detailed examples of using API calls to automate masking, see the Masking APIs

Masking in Parallel

Continuous Compliance supports launching masking jobs in parallel. When jobs have no dependencies, you can initiate parallel masking API jobs (with wrapper code as required) to allow the jobs to be run as a pre and/or post hook.

Oracle use cases for APIs

Oracle link + snapshot (sync)

The following script ingests links an environment database dSource (Oracle SID / Instance) and then takes a snapshot. See parameters for required values that you must provide.

This script demonstrates how to use name values inputs and get the respective Delphix object and/or object reference for use in the json input in downstream API calls.

Filename: link_oracle.sh # or link_oracle_jq.sh

Edit the file to update the parameters as required for your environment.

```
############################################################Parameter
InitializationDMIP=172.16.160.195#DMPORT=8282DMUSER=delphix_adminDMPASS=delphix . . .
Required for Database Link and Sync ...SOURCE_SID="DPXDEV01"            # Source
Environment Database SIDSOURCE_NAME="DPXDEV01"            # Delphix dSource
NameSOURCE_ENV="Oracle Target"        # Source Environment NameSOURCE_GRP="Oracle_Sou
rce"        # Delphix Group NameDB_USER="delphixdb"                # Source Database
SID user accountDB_PASS="delphixdb"                # Source Database SID user
password ##############################################    NO CHANGES REQUIRED BELOW
THIS POINT    ##############################################$ ./
link_oracle.sh        # or    ./link_oracle_jq.shAuthenticating on http://
172.16.160.195/resources/json/delphixSession and Login Successful ...group reference:
GROUP-35sourceconfig reference: ORACLE_SINGLE_CONFIG-1primaryUser reference:
HOST_USER-3Linking Source Database ...Job: JOB-92Job: JOB-92 100.0%
Completed ...Container: ORACLE_DB_CONTAINER-19Running SnapSync ...Job: JOB-93Current
status as of Mon Aug 15 13:07:53 EDT 2016 : RUNNING : 0.0% CompletedCurrent status as
of Mon Aug 15 13:08:03 EDT 2016 : RUNNING : 15.0% CompletedCurrent status as of Mon
Aug 15 13:08:13 EDT 2016 : RUNNING : 35.0% CompletedCurrent status as of Mon Aug 15
13:08:24 EDT 2016 : RUNNING : 59.0% Completed
```

```
Current status as of Mon Aug 15 13:08:34 EDT 2016 : RUNNING : 66.0% CompletedCurrent
status as of Mon Aug 15 13:08:44 EDT 2016 : RUNNING : 74.0% CompletedJob: JOB-93
100.0% Completed ... Done ... $
```

Oracle provision

Filename: provision_oracle.txt

Shown below is how to use the CLI to provision an Oracle 11g database that is already ingested into the Delphix Engine.

The key is to get the object reference names first. For example, to get the source database container name:

```
ssh delphix_admin[delphix_engine_ip_address_or_hostname]> database> ls...> select
"[database_name]"> lsDelphix5002HWv7 database> select 'DPXDEV01'Delphix5002HWv7
database 'DPXDEV01'> lsProperties    type: OracleDatabaseContainer    name:
DPXDEV01    . . .        reference: ORACLE_DB_CONTAINER-18     . . .
```

Minimum parameters required to provision:

```
Delphix5002HWv7 database provision > *commit=== POST /resources/json/delphix/
database/provision ==={    "type": "OracleProvisionParameters",    "container":
{        "type": "OracleDatabaseContainer",        "name": "VBITT" , # Delphix Object
Name, Typically matches VDB name         "group": "GROUP-36" # group ls select
"[group_name]" ls     },    "source": {        "type": "OracleVirtualSource",
        "mountBase": "/mnt/provision" # Delphix Filesystem Mount path     },    "sour
ceConfig": {        "type": "OracleSIConfig",         "repository":
"ORACLE_INSTALL-3" , # repository, select "[repository_name]"         "databaseName":
"VBITT" , # New VDB Name         "uniqueName": "VBITT",         "instance":
{            "type": "OracleInstance",            "instanceName": "VBITT",
 "instanceNumber": 1        }    },    "timeflowPointParameters": {        "type":
"TimeflowPointSemantic",        "container": "ORACLE_DB_CONTAINER-18" # select
"[datbase_name]" ls     }}=== RESPONSE ===
```

Sample CLI session

```
Delphix5002HWv7 database> setopt trace=false Delphix5002HWv7 database> provision
Delphix5002HWv7 database provision > *ls Properties    type:
OracleProvisionParameters    container:        type:
OracleDatabaseContainer        name: (required)        description:
(unset)        diagnoseNoLoggingFaults: true        group:
(required)        performanceMode: DISABLED        preProvisioningEnabled: false
   sourcingPolicy: (unset)    credential: (unset)    maskingJob: (unset)    newDBID:
false    openResetlogs: true    physicalStandby: false    source:        type:
OracleLiveSource        name: (unset)        archivelogMode: true        config:
(unset)        configParams: (unset)        configTemplate:
(unset)        customEnvVars: (unset)        dataAgeWarningThreshold:
900sec        fileMappingRules: (unset)        manualProvisioning: false        mount
Base: (required)        nodeListenerList: (unset)        operations:
(unset)        redoLogGroups: 3        redoLogSizeInMB: 0    sourceConfig:        typ
e: OraclePDBConfig        cdbConfig: (required)        databaseName:
(required)        environmentUser: (unset)        linkingEnabled: true        reposit
ory: (unset)        services: (unset)    timeflowPointParameters:        type:
TimeflowPointSemantic        container: (required)        location:
LATEST_POINT    username: (unset) OperationsdefaultsDelphix5002HWv7 database
provision > *edit container Delphix5002HWv7 database provision container> *ls
Properties    type: OracleDatabaseContainer    name: (required)    description:
(unset)    diagnoseNoLoggingFaults: true    group: (required)    performanceMode:
DISABLED    preProvisioningEnabled: false    sourcingPolicy: (unset)Delphix5002HWv7
database provision container> *set name=VBITT Delphix5002HWv7 database provision
container> *set group=GROUP-36 Delphix5002HWv7 database provision container> *back
Delphix5002HWv7 database provision > *edit source Delphix5002HWv7 database provision
source > *ls Properties    type: OracleLiveSource    name: (unset)    archivelogMode:
true    config: (unset)    configParams: (unset)    configTemplate:
(unset)    customEnvVars: (unset)    dataAgeWarningThreshold:
900sec    fileMappingRules: (unset)    manualProvisioning: false    mountBase:
(required)    nodeListenerList: (unset)    operations: (unset)    redoLogGroups: 3
 redoLogSizeInMB: 0Delphix5002HWv7 database provision source > *set
type=OracleVirtualSource Delphix5002HWv7 database provision source > *set mountBase=/
mnt/provision Delphix5002HWv7 database provision source > *back Delphix5002HWv7
database provision > *edit sourceConfig Delphix5002HWv7 database provision
```

```
sourceConfig > *ls Properties    type: OraclePDBConfig    cdbConfig:
(required)    databaseName: (required)    environmentUser: (unset)    linkingEnabled:
true    repository: (unset)    services: (unset)Delphix5002HWv7 database provision
sourceConfig > *set type=OracleSIConfig Delphix5002HWv7 database provision
sourceConfig > *ls Properties    type: OracleSIConfig      databaseName:
(required)    environmentUser: (unset)    instance: (required)    linkingEnabled:
true    nonSysCredentials: (unset)    nonSysUser: (unset)    repository:
(required)    services: (unset)    uniqueName: (required)Delphix5002HWv7 database
provision sourceConfig > *set databaseName=VBITT Delphix5002HWv7 database provision
sourceConfig > *set repository=ORACLE_INSTALL-3 Delphix5002HWv7 database provision
sourceConfig > *set uniqueName=VBITT Delphix5002HWv7 database provision sourceConfig
> *set instance.instanceName=VBITT Delphix5002HWv7 database provision sourceConfig >
*set instance.instanceNumber=1 Delphix5002HWv7 database provision sourceConfig > *ls
Properties    type: OracleSIConfig      databaseName: VBITT      environmentUser:
(unset)    instance:        type: OracleInstance          instanceName: VBITT
        instanceNumber: 1      linkingEnabled: true    nonSysCredentials:
(unset)    nonSysUser: (unset)    repository: '/u02/ora/app/product/11.2.0/dbhome_1'
    services: (unset)    uniqueName: VBITT  Delphix5002HWv7 database provision
sourceConfig > *back Delphix5002HWv7 database provision > *edit
timeflowPointParameters Delphix5002HWv7 database provision timeflowPointParameters>
*ls Properties    type: TimeflowPointSemantic    container: (required)    location:
LATEST_POINTDelphix5002HWv7 database provision timeflowPointParameters> *set
container=ORACLE_DB_CONTAINER-18 Delphix5002HWv7 database provision
timeflowPointParameters> *back Delphix5002HWv7 database provision > *commit
    VBITT    Dispatched job JOB-348    DB_PROVISION job started for "Oracle Target
Virtual Databases/VBITT".    Starting provision of the virtual database "VBITT".    C
reating new TimeFlow.    Generating recovery scripts.    Exporting
storage.    Mounting filesystems for the virtual database instance "1".    Mounting
read-only archive log filesystem for the virtual database instance "1".    Recovering
Oracle database.    \|/-    Opening the virtual database "VBITT".    Opening Oracle
database.    Oracle recovery was successful.    Unmounting read-only archive log
filesystem for the virtual database instance "1".    The virtual database "VBITT" was
successfully provisioned.    DB_PROVISION job for "Oracle Target Virtual Databases/
VBITT" completed successfully.Delphix5002HWv7 database>
```

With the **setopt trace=true** option set, you can convert the JSON output from the above CLI provision command to the RESTful API cURL commands. If VBITT exists, be sure to delete it first.

Request:

```
curl X POST -k --data @http://172.16.160.177/resources/json/delphix/database/
provision \    -b cookies.txt -H "Content-Type: application/json" <<EOF{    "type":
"OracleProvisionParameters",    "container": {        "type":
"OracleDatabaseContainer",        "name": "VBITT",        "group": "GROUP-36"
    },    "source": {        "type": "OracleVirtualSource",        "mountBase": "/
mnt/provision"    },    "sourceConfig": {        "type": "OracleSIConfig",
        "repository": "ORACLE_INSTALL-3",        "databaseName": "VBITT",
        "uniqueName": "VBITT",        "instance": {            "type":
"OracleInstance",            "instanceName": "VBITT",            "instanceNumber":
1        }    },    "timeflowPointParameters": {        "type":
"TimeflowPointSemantic",        "container": "ORACLE_DB_CONTAINER-18"    }}EOF
```

Response:

```
{"type":"OKResult","status":"OK","result":"ORACLE_DB_CONTAINER-22","job":"JOB-353",
"action":"ACTION-649"}
```

Put all the commands above within a shell script to automate the complete process of provisioning an Oracle 11.2.0.4 database.

Notice that the script below looks up 4 object references for use within the JSON input into the API.

Filename: provision_oracle.sh# or provision_oracle_jq.sh

Edit the file to update the parameters as required for your environment.

```
######################    DELPHIX CORP    ######################Parameter
Initialization DMIP=172.16.160.195DMUSER=delphix_adminDMPASS=delphixCOOKIE="~/
cookies.txt"COOKIE=`eval echo $COOKIE`CONTENT_TYPE="Content-Type: application/json"DE
LAYTIMESEC=10BaseURL="http://${DMIP}/resources/json/delphix"#Required for Database
Link and Sync ...#VDB_NAME="VBITT"                # Delphix VDB NameMOUNT_BASE="/mnt/
provision"        # Delphix Engine Mount PathSOURCE_GRP="Oracle_Target"        #
Delphix Engine Group NameTARGET_ENV="Oracle Target"        # Target Environment used
to get repository reference valueSOURCE_SID="DPXDEV01"            # dSource name used
to get db container reference value
#############################################    NO CHANGES REQUIRED BELOW THIS
POINT    ###############################################
```

Sample Output

```
$ ./provision_oracle.sh# or ./provision_oracle_jq.shAuthenticating on http://
172.16.160.195/resources/json/delphixSession and Login Successful ...group reference:
GROUP-36 container reference: ORACLE_DB_CONTAINER-36 env reference:
UNIX_HOST_ENVIRONMENT-3 repository reference: ORACLE_INSTALL-1 Provisioning VDB from
Source Database ...Job: JOB-155Current status as of Mon Aug 15 23:40:51 EDT 2016 :
RUNNING 0.0% CompletedCurrent status as of Mon Aug 15 23:40:51 EDT 2016 : RUNNING
0.0% CompletedCurrent status as of Mon Aug 15 23:41:01 EDT 2016 : RUNNING 9.0%
CompletedCurrent status as of Mon Aug 15 23:41:11 EDT 2016 : RUNNING 45.0%
CompletedCurrent status as of Mon Aug 15 23:41:21 EDT 2016 : RUNNING 45.0%
CompletedCurrent status as of Mon Aug 15 23:41:31 EDT 2016 : RUNNING 46.0%
CompletedCurrent status as of Mon Aug 15 23:41:41 EDT 2016 : RUNNING 48.0%
CompletedCurrent status as of Mon Aug 15 23:41:51 EDT 2016 : RUNNING 60.0%
CompletedJob: JOB-155 COMPLETED 100.0% Completed ... Done ... $
```

Filename: provision_oracle_child.sh# or provision_oracle_child_jq.sh

```
$ ./provision_oracle_child.sh# or ./provision_oracle_child_jq.shAuthenticating on
http://172.16.160.195/resources/json/delphixSession and Login Successful ...group
reference: GROUP-36 container reference: ORACLE_DB_CONTAINER-118 env reference:
UNIX_HOST_ENVIRONMENT-9 repository reference: ORACLE_INSTALL-6 Provisioning VDB from
Source Database ...Job: JOB-857Current status as of Mon Sep 5 22:48:28 EDT 2016 :
RUNNING 0.0% CompletedCurrent status as of Mon Sep 5 22:48:28 EDT 2016 : RUNNING 0.0%
CompletedCurrent status as of Mon Sep 5 22:48:38 EDT 2016 : RUNNING 9.0%
CompletedCurrent status as of Mon Sep 5 22:48:48 EDT 2016 : RUNNING 27.0%
CompletedCurrent status as of Mon Sep 5 22:48:58 EDT 2016 : RUNNING 42.0%
```

```
CompletedCurrent status as of Mon Sep 5 22:49:08 EDT 2016 : RUNNING 45.0%
CompletedCurrent status as of Mon Sep 5 22:49:28 EDT 2016 : RUNNING 46.0%
CompletedCurrent status as of Mon Sep 5 22:49:38 EDT 2016 : RUNNING 48.0%
CompletedCurrent status as of Mon Sep 5 22:49:48 EDT 2016 : RUNNING 51.0%
CompletedCurrent status as of Mon Sep 5 22:50:08 EDT 2016 : RUNNING 71.0%
CompletedJob: JOB-857 COMPLETED 100.0% Completed ... Done ...
```

SQL server API use cases

SQL server link/ingest environment dSource

For the **Window Target** environment, the dSource **delphixdb** in **MSSQLSERVER** instance will be linked/ingested into the Delphix Engine. It will appear in the **Windows_Source** group below.

Filename: link_sqlserver.ps1

```
PS> . .\link_sqlserver.ps1Authenticating on http://172.16.160.195/resources/json/
delphixLogin Successful ...group reference: GROUP-34 sourceconfig reference:
MSSQL_SINGLE_CONFIG-26 env reference: WINDOWS_ HOST_ENVIRONMENT-7 repository
reference: MSSQL_INSTANCE-4 database link API Results:
{"type":"OKResult","status":"OK","result":"MSSQL_DB_CONTAINER-114","job":"JOB-819","a
ction":"ACTION-1659"}DB Container: MSSQL_DB_CONTAINER-114
```

```
Job # JOB-819*****    waiting for status    *****Current status as of 09/05/2016 11:4
1:13 : COMPLETED : 100.0% CompletedJob COMPLETED Succesfully.
```

```
JOB JOB-820waiting for status *****Current status as of 09/05/2016 11:41:23 :
RUNNING : 5.0% CompletedCurrent status as of 09/05/2016 11:41:44 : RUNNING : 9.0%
Completed
```

```
Current status as of 09/05/2016 11:41:54 : RUNNING : 56.0% CompletedJob COMPLETED
Succesfully. Done ...
```

Successful dSource linked/ingested into the Delphix Engine.

SQL server provision

The example below is done from the command line once you know the parameters and reference object names.

Filename: windows_sqlserver_provision.txt

Create these 3 JSON text files:

```
session.json{    "type": "APISession",    "version": {        "type": "APIVersion",
      "major": 1,        "minor": 5,        "micro": 3    }} login.json{    "type":
"LoginRequest",    "username": "delphix_admin",    "password": "delphix"}
provision.json{    "type": "MSSqlProvisionParameters",    "container": {        "type
": "MSSqlDatabaseContainer",        "name": "Vbitt00",        "group": "GROUP-36",
     "sourcingPolicy": {            "type": "SourcingPolicy",            "loadFromBac
kup": false,            "logsyncEnabled": false        },        "validatedSyncMode":
"TRANSACTION_LOG"    },    "source": {        "type": "MSSqlVirtualSource",        "o
perations": {            "type": "VirtualSourceOperations",            "configureClon
e": [],            "postRefresh": [],            "postRollback": [],            "post
Snapshot": [],            "preRefresh": [],            "preSnapshot":
[]        }    },    "sourceConfig": {        "type": "MSSqlSIConfig",        "linkin
```

```
gEnabled": false,        "repository": "MSSQL_INSTANCE-1",        "databaseName":
"Vbitt00",        "recoveryModel": "SIMPLE",        "instance": {            "type":
"MSSqlInstanceConfig",            "host": "WINDOWS_HOST-1"        }    },    "timeflo
wPointParameters": {        "type": "TimeflowPointSemantic",        "container":
"MSSQL_DB_CONTAINER-23",        "location": "LATEST_SNAPSHOT"    }}
```

This works on Windows Powershell Command Prompt

⚠ Use curl, curl.exe or modify the default alias.

```
curl --insecure -c cookies.txt -i -X POST -H "Content-Type: application/json" -d
"@session.json" http://172.16.160.153/resources/json/delphix/sessioncurl --insecure
-b cookies.txt -i -X POST -H "Content-Type: application/json" -d "@login.json"
http://172.16.160.153/resources/json/delphix/logincurl --insecure -b cookies.txt -i
-X POST -H "Content-Type: application/json" -d "@provision.json" http://
172.16.160.153/resources/json/delphix/database/provision
```

Plug in the returned JOB #

```
curl --insecure -b cookies.txt -i -X GET -H "Content-Type: application/json" -k http:/
/172.16.160.153/resources/json/delphix/notification?channel=JOB-428
```

Get Example

```
curl --insecure -b cookies.txt -i -X GET -H "Content-Type: application/json" -k http:/
/172.16.160.153/resources/json/delphix/system
```

Complete example.

Provision the newly created **delphixdb** dSource in the **Windows_Source** group to a virtual database VBITT in the **Windows_Target** group.

Filename: provision_sqlserver.ps1

Variables ...

```
$nl = [Environment]::NewLine$BaseURL = " http://172.16.160.195/resources/json/delphix
"$cookie = "cookies.txt"$delphix_user = "delphix_admin"$delphix_pass = "delphix". . .
```

Required for Provisioning Virtual Database ...

```
$SOURCE_SID="delphixdb"            # dSource name used to get db container reference
value $VDB_NAME="VBITT"                # Delphix VDB Name$TARGET_GRP="Windows_Target"
   # Delphix Engine Group Name$TARGET_ENV="Window Target"        # Target Environment
used to get repository reference value $TARGET_REP="MSSQLSERVER"    # Target
Environment Repository / Instance name
##############################################    NO CHANGES REQUIRED BELOW THIS
POINT    ##############################################
```

Sample Run Output

```
PS> . .\provision_sqlserver.ps1Authenticating on http://172.16.160.195/resources/
json/delphixLogin Successful ...group reference:  GROUP-37 container reference:
MSSQL_DB_CONTAINER-114 env reference:  WINDOWS_HOST_ENVIRONMENT-7 repository
reference:  MSSQL_INSTANCE-4 database provision API Results:
{"type":"OKResult","status":"OK","result":"MSSQL_DB_CONTAINER-115","job":"JOB-822","a
ction":"ACTION-1664"}DB Container:  MSSQL_DB_CONTAINER-115 Job # JOB-822 jobState
RUNNINGpercentComplete 0.0***** waiting for status *****Current status as of
09/05/2016 11:43:51 : RUNNING : 0.0% CompletedCurrent status as of 09/05/2016
11:44:01 : RUNNING : 3.0% CompletedCurrent status as of 09/05/2016 11:44:12 :
RUNNING : 11.0% CompletedCurrent status as of 09/05/2016 11:44:22 : RUNNING : 18.0%
CompletedCurrent status as of 09/05/2016 11:44:32 : RUNNING : 18.0% Completed
```

```
Current status as of 09/05/2016 11:44:52 : RUNNING : 75.0% CompletedJob COMPLETED
Succesfully. Done ...
```

SQL server refresh

The following are curl commands that can be issued from the Powershell command line. For inclusion within a Powershell script, see the masking example, *masking.ps1*.

Filename: windows_sqlserver_refresh.txt

MS SQL Server Refresh Example ...

Session ...

```
curl --insecure -c cookies.txt -i -X POST -H "Content-Type: application/json" -d
"@session.json" http://172.16.160.179/resources/json/delphix/session
```

Filename: session.json

```
{    "type": "APISession",    "version": {        "type": "APIVersion",        "major
": 1,        "minor": 5,        "micro": 3    }} PS> *curl --insecure -c cookies.txt
-i -X POST -H "Content-Type: application/json" -d "@session.json" http://
172.16.160.179/resources/json/delphix/session*HTTP/1.1 200 OKServer: Apache-Coyote/
1.1Set-Cookie: JSESSIONID=8DE0362F5BBD73E6BFA9E13FF111E78C; Path=/resources/;
HttpOnlyContent-Type: application/jsonContent-Length: 179Date: Thu, 16 Jun 2016
07:24:34 GMT{"type":"OKResult","status":"OK","result":{"type":"APISession","version":
{"type":"APIVersion","major":1,"minor":5,"micro":3},"locale":null,"client":null},"job
":null,"action":null} PS>

Login ...

curl --insecure -b cookies.txt -i -X POST -H "Content-Type: application/json" -d
"@login.json" http://172.16.160.179/resources/json/delphix/login

Filename:login.json
```

```
{    "type": "LoginRequest",    "username": "delphix_admin",    "password": "delphix"}
  PS> *curl --insecure -b cookies.txt -i -X POST -H "Content-Type: application/json"
 -d "@login.json" http://172.16.160.179/resources/json/delphix/login*HTTP/1.1 200
OKServer: Apache-Coyote/1.1Content-Type: application/jsonContent-Length: 76Date: Thu,
16 Jun 2016 07:25:39 GMT
{"type":"OKResult","status":"OK","result":"USER-2","job":null,"action":null}PS C:
\Users\Administrator>
```

List Databases ...

```
curl --insecure -b cookies.txt -i -X GET -H "Content-Type: application/json" -k http:/
/172.16.160.179/resources/json/delphix/databasePS> *curl --insecure -b cookies.txt -i
-X GET -H "Content-Type: application/json" -k http://172.16.160.179/resources/json/
delphix/database*HTTP/1.1 200 OKServer: Apache-Coyote/1.1Content-Type: application/
jsonContent-Length: 4062Date: Thu, 16 Jun 2016 07:27:14 GMT
{"type":"ListResult","status":"OK","result":[......
{"type":"MSSqlDatabaseContainer","reference":"MSSQL_DB_CONTAINER-37","namespace":null
,"name":"Vdelphix_demo","group":"GROUP-35","provisionContainer":"MSSQL_DB_CONTAINER-3
6","creationTime":"2016-06-16T07:09:06.222Z","currentTimeflow":"MSSQL_TIMEFLOW-38","p
reviousTimeflow":"MSSQL_TIMEFLOW-37","description":null,"runtime":...
{"type":"So{"type":"MSSqlDatabaseContainer","reference":"MSSQL_DB_CONTAINER-36","name
space":null,"name":"delphix_demo","group":"GROUP-35","provisionContainer":null,"creat
ionTime":"2016-06-16T07:07:49.939Z","currentTimeflow":"MSSQL_TIMEFLOW-36","previousTi
meflow":null,"description":"","runtime":......}],"job":null,"action":null,"total":6,"
overflow":false} PS>
```

Need Reference Object from Database Information ...

For Parent Source Database delphix_demo, reference object is MSSQL_DB_CONTAINER-36

For Virtual Database Vdelphix_demo, reference object is MSSQL_DB_CONTAINER-37

[Optional: Get Database Info ...]

```
curl --insecure -b cookies.txt -i -X GET -H "Content-Type: application/json" -k http:/
/172.16.160.179/resources/json/delphix/database/MSSQL_DB_CONTAINER-37 PS> *curl --
insecure -b cookies.txt -i -X GET -H "Content-Type: application/json" -k http://
172.16.160.179/resources/json/delphix/database/MSSQL_DB_CONTAINER-37*HTTP/1.1 200
OKServer: Apache-Coyote/1.1Content-Type: application/jsonContent-Length: 696Date:
Thu, 16 Jun 2016 07:35:42 GMT {"type":"OKResult","status":"OK","result":
{"type":"MSSqlDatabaseContainer","reference":"MSSQL_DB_CONTAINER-37","namespace":null
,"name":"Vdelphix_demo","group":"GROUP-35","provisionContainer":"MSSQL_DB_CONTAINER-3
6","creationTime":"2016-06-16T07:09:06.222Z","currentTimeflow":"MSSQL_TIMEFLOW-38","p
reviousTimeflow":"MSSQL_TIMEFLOW-37","description":null,"runtime":
{"type":"MSSqlDBContainerRuntime","logSyncActive":false,"preProvisioningStatus":null,
"lastRestoredBackupSetUUID":null},"os":"Windows","processor":"x86","sourcingPolicy":
{"type":"SourcingPolicy","logsyncEnabled":false,"loadFromBackup":false},"performanceM
ode":"DISABLED","delphixManaged":true,"masked":false},"job":null,"action":null} PS>
```

Refresh Vdelphix_demo using parent delphix_demo (MSSQL_DB_CONTAINER-36) with the latest timecard ...

```
curl --insecure -b cookies.txt -i -X POST -H "Content-Type: application/json" -d
"@refresh.json" http://172.16.160.179/resources/json/delphix/database/
MSSQL_DB_CONTAINER-37/refresh=== POST /resources/json/delphix/database/
MSSQL_DB_CONTAINER-37/refresh ===refresh.json{    "type":
"RefreshParameters",    "timeflowPointParameters": {        "type":
"TimeflowPointSemantic",        "container": "MSSQL_DB_CONTAINER-36"    }} PS> *curl
--insecure -b cookies.txt -i -X POST -H "Content-Type: application/json" -d
"@refresh.json" http://172.16.160.179/resources/json/delphix/database/
MSSQL_DB_CONTAINER-37/refresh*HTTP/1.1 200 OKServer: Apache-Coyote/1.1Content-Type:
application/jsonContent-Length: 82Date: Thu, 16 Jun 2016 07:40:44 GMT
{"type":"OKResult","status":"OK","result":"","job":"JOB-60","action":"ACTION-167"}
PS>
```

[Observer Delphix GUI Action]

Done with SQL Server VDB Refresh ...

API programming language examples

The following programming language examples are just to show the bare minimum authentication and a sample functional API call. There are numerous modules, libraries, methods, functions, and code examples to further demonstrate how the languages work with the Delphix APIs and JSON data strings/objects.

You can execute PHP, Perl, and Python languages from the command line and/or from within a Web Server such as Apache or IIS. The following examples are formatted for command line / terminal output.

PHP

PHP provides cURL and JSON modules.

```
$ php -i | grep -iE "cURL|json"
curl
cURL support => enabled
cURL Information => 7.43.0
json
json support => enabled
json version => 1.2.1
```

Filename: delphix_curl.php

Sample Output:

```
$ php -f delphix_curl.php
Session json> {"type":"APISession","version":{"type":"APIVersion","major":1,"minor":7,"
micro":0}}
Session Results> {"type":"OKResult","status":"OK","result":{"type":"APISession","vers
ion":{"type":"APIVersion","major":1,"minor":7,"micro":0},"locale":null,"client":null}
,"job":null,"action":null}
Login json> {"type":"LoginRequest","username":"delphix_admin","password":"delphix"}
Login Results> {"type":"OKResult","status":"OK","result":"USER-2","job":null,"action":n
ull}
Calling About API ...
About Results> {"type":"OKResult","status":"OK","result":{"type":"PublicSystemInfo","
productType":"standard","productName":"Delphix Engine","buildTitle":"Delphix Engine
5.1.1.0","buildTimestamp":"2016-07-21T07:23:41.000Z","buildVersion":{"type":"VersionI
nfo","major":5,"minor":1,"micro":1,"patch":0},"configured":true,"enabledFeatures":
["XPP","MSSQLHOOKS"],"apiVersion":{"type":"APIVersion","major":1,"minor":8,"micro":0}
,"banner":null,"locales":["en-US"],"currentLocale":"en-US"},"job":null,"action":null}
Converting json string to a PHP Array
stdClass Object
(
    [type] => OKResult
    [status] => OK
    [result] => stdClass Object
        (
            [type] => PublicSystemInfo
            [productType] => standard
            [productName] => Delphix Engine
```

```
            [buildTitle] => Delphix Engine 5.1.1.0
            [buildTimestamp] => 2016-07-21T07:23:41.000Z
            [buildVersion] => stdClass Object
                (
                    [type] => VersionInfo
                    [major] => 5
                    [minor] => 1
                    [micro] => 1
                    [patch] => 0
                )
            [configured] => 1
            [enabledFeatures] => Array
                (
                    [0] => XPP
                    [1] => MSSQLHOOKS
                )
             [apiVersion] => stdClass Object
                (
                    [type] => APIVersion
                    [major] => 1
                    [minor] => 8
                    [micro] => 0
                )
             [banner] =>
             [locales] => Array
                (
                    [0] => en-US
                )
             [currentLocale] => en-US
        )
    [job] =>
    [action] =>
)
```

Perl

Perl provides a couple of methods for working with cURL: operating system calls, WWW::Curl (libcurl) module, or LWP::Curl module. The sample below simply logs into the Delphix Engine and lists the current Delphix Environments.

Filename: perl_curl.pl

Sample Output:

```
$ perl perl_curl.pl
Testing cURL on Perl ...

Session Results: {"type":"OKResult","status":"OK","result":{"type":"APISession","vers
ion":{"type":"APIVersion","major":1,"minor":7,"micro":0},"locale":null,"client":null}
,"job":null,"action":null}
Login Results: {"type":"OKResult","status":"OK","result":"USER-2","job":null,"action":n
ull}
```

```
Enviornment Results: {"type":"ListResult","status":"OK","result":[{"type":"WindowsHos
tEnvironment","reference":"WINDOWS_HOST_ENVIRONMENT-7","namespace":null,"name":
"Window Target","description":null,"primaryUser":"HOST_USER-7","enabled":false,"host":"
WINDOWS_HOST-6","proxy":null},{"type":"UnixHostEnvironment","reference":"UNIX_HOST_EN
VIRONMENT-9","namespace":null,"name":"Oracle Target","description":"","primaryUser":"
HOST_USER-9","enabled":true,"host":"UNIX_HOST-8","aseHostEnvironmentParameters":null
}],"job":null,"action":null,"total":2,"overflow":false}

Done
```

Python

Delphix has an extensive resource library for using Python with the Delphix Engine.

CLI to Python transition

Delphix python module

Blogs

https://github.com/CloudSurgeon/delphixpy-examples

Related Videos

- https://vimeo.com/164779308
- https://vimeo.com/170187276
- https://vimeo.com/170896907

Simple Python program to authenticate and get the "about" API results. This script requires the "request" and "json" modules.

http://stackoverflow.com/questions/17309288/importerror-no-module-named-requests

Filename: auth.py

Sample Output

```
$ python auth.py
Authenticating URL http://172.16.160.195/resources/json/delphix ...
{"type":"OKResult","status":"OK","result":{"type":"APISession","version":{"type":"API
Version","major":1,"minor":7,"micro":0},"locale":null,"client":null},"job":null,
"action":null}
Login ...
{"type":"OKResult","status":"OK","result":"USER-2","job":null,"action":null}
About ...
{"type":"OKResult","status":"OK","result":{"type":"PublicSystemInfo","productType":"s
tandard","productName":"Delphix Engine","buildTitle":"Delphix Engine 5.1.1.0","buildT
imestamp":"2016-07-21T07:23:41.000Z","buildVersion":{"type":"VersionInfo","major":5,"
minor":1,"micro":1,"patch":0},"configured":true,"enabledFeatures":["XPP","MSSQLHOOKS"]
,"apiVersion":{"type":"APIVersion","major":1,"minor":8,"micro":0},"banner":null,"loca
les":["en-US"],"currentLocale":"en-US"},"job":null,"action":null}
JSON Parsing Examples ...
OK
Delphix Engine 5.1.1.0
```

```
1
```

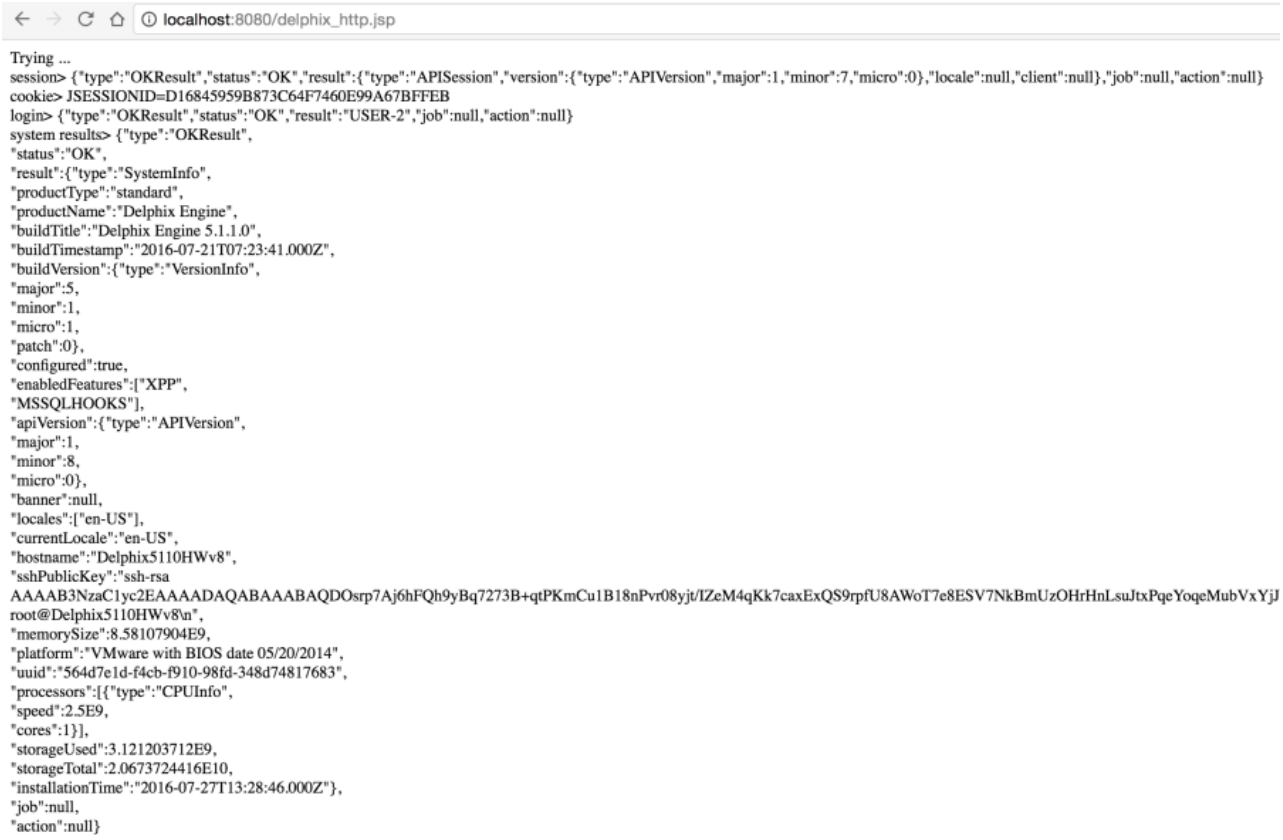
JSP (Java server pages)

Java Server Pages are typically used for the web formatting and output, but you can also use JSP for application logic processing and native Java code integration, although this is scorned by the purest and most logical thinking programmers.

Filename: delphix_http.jsp

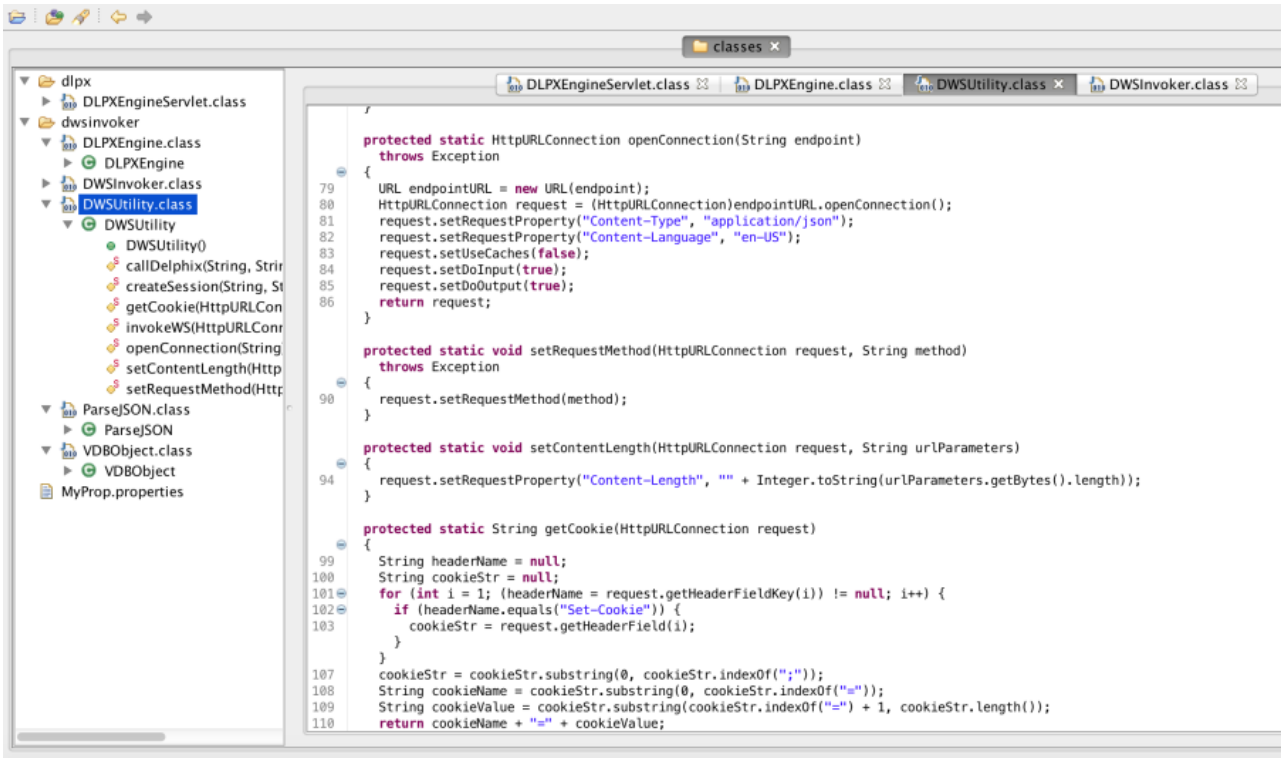
Sample Output:

Browser URL: http://localhost:8080/delphix_http.jsp



Java

Java methods and classes allow coding logic to be effectively re-used, extended and modularized for flexible applications. Using the Java code embedded within the JSP file, code is logically placed into respective classes and methods.


classes
dlpx
DLPXEngineServlet.class
dwsinvoker
DLPXEngine.class
DLPXEngine
DWSInvoker.class
DWSUtility.class
DWSUtility
DWSUtility()
callDelphix(String, Strir
createSession(String, St
getCookie(HttpURLCon
invokeWS(HttpURLConr
openConnection(String
setContentLength(Http
setRequestMethod(Http
ParseJSON.class
ParseJSON
VDBObject.class
VDBObject
MyProp.properties
DLPXEngineServlet.class
DLPXEngine.class
DWSUtility.class
DWSInvoker.class
protected static HttpURLConnection openConnection(String endpoint)
throws Exception
{
URL endpointURL = new URL(endpoint);
HttpURLConnection request = (HttpURLConnection)endpointURL.openConnection();
request.setRequestProperty("Content-Type", "application/json");
request.setRequestProperty("Content-Language", "en-US");
request.setUseCaches(false);
request.setDoInput(true);
request.setDoOutput(true);
return request;
}
protected static void setRequestMethod(HttpURLConnection request, String method)
throws Exception
{
request.setRequestMethod(method);
}
protected static void setContentLength(HttpURLConnection request, String urlParameters)
{
request.setRequestProperty("Content-Length", "" + Integer.toString(urlParameters.getBytes().length));
}
protected static String getCookie(HttpURLConnection request)
{
String headerName = null;
String cookieStr = null;
for (int i = 1; (headerName = request.getHeaderFieldKey(i)) != null; i++) {
if (headerName.equals("Set-Cookie")) {
cookieStr = request.getHeaderField(i);
}
}
cookieStr = cookieStr.substring(0, cookieStr.indexOf(";"));
String cookieName = cookieStr.substring(0, cookieStr.indexOf("="));
String cookieValue = cookieStr.substring(cookieStr.indexOf("=") + 1, cookieStr.length());
return cookieName + "=" + cookieValue;

API timeflows

From earlier, the RESTFul URL for a virtual database refresh will look like:

http://<delphix_engine>/resources/json/delphix/database/MSSQL_DB_CONTAINER-39/refresh

where the **MSSQL_DB_CONTAINER-39** represents the target virtualized database to refresh and we need to POST the JSON data to the URL for processing.

```
{    "type": "RefreshParameters",    "timeflowPointParameters": {        "type":
"TimeflowPointSemantic",        "container": "MSSQL_DB_CONTAINER-38"    }}
```

Timeflow parameters

The **"timeflowPointParameters"** key has 6 **"type": "..."** options which each have their own set of parameters. The type **"TimeflowPointSemantic"** uses the default LATEST_POINT within the source container. Now, for more on timeflowPointParameters.

http://<delphix_engine>/api/#TimeflowPointParameters

TimeflowPointParameters

Parameters indicating a TimeFlow point to use as input to database operations.

TypedObject
TimeflowPointParameters

Direct Known Subclasses:

TimeflowPointTimestamp, TimeflowPointSnapshot, TimeflowPointSemantic, TimeflowPointLocation, TimeflowPointBookmark, TimeflowPointBookmarkTag

TimeflowPointTimestamp

timeflow	Reference to TimeFlow containing this point. Type: Reference to Timeflow Constraints: Required: true
timestamp	The logical time corresponding to the TimeFlow location. Type: date Constraints: Required: true

TimeflowPointSnapshot

snapshot	Reference to the snapshot. Type: Reference to TimeflowSnapshot

TimeflowPointSemantic

Semantic reference to a Timeflow point.

⚠ The reference is relative to a container and not a TimeFlow. If the container contains multiple TimeFlows, the Delphix Engine will evaluate the semantic reference with regards to all TimeFlows in that container.

container	Reference to the container. Type: Reference to Container Constraints: Required: true
location	A semantic description of a TimeFlow location. Type: string Constraints: Default: LATEST_POINT Acceptable values: LATEST_POINT, LATEST_SNAPSHOT Create: optional Update: optional

TimeflowPointLocation

TimeFlow point based on a database-specific identifier (SCN, LSN, etc).

location	The TimeFlow location. Type: string Constraints: Required: true
timeflow	Reference to TimeFlow containing this location. Type: Reference to Timeflow Constraints: Required: true

TimeflowPointBookmark

bookmark	Reference to the bookmark. Type: Reference to TimeflowBookmark Constraints: Required: true

TimeflowPointBookmarkTag

container	Reference to the container. Type: Reference to Container Constraints: Required: true
tag	The name of the tag. Type: string Constraints: Required: true

Timeflow API objects: timeflow, snapshot, timeflowRanges

The sample code provides information on the Timeflow, timeflowRanges and snapshot objects for the respective VDB.

Filename: flows.sh

```
$ ./flows.sh VBITT10Session API Login API Login {"type":"OKResult","status":"OK","res
ult":"USER-2","job":null,"action":null}Source: VBITT10container reference:
ORACLE_DB_CONTAINER-75 Timeflows API timeflow names:DB_PROVISION@2016-09-27T13:15:18D
B_ROLLBACK@2016-09-28T00:33:54DB_ROLLBACK@2016-09-28T00:38:33DB_ROLLBACK@2016-09
-28T00:41:44
```

Select Timeflow Name (copy-n-paste from the above list):

```
DB_ROLLBACK@2016-09-28T00:33:54timeflow reference: ORACLE_TIMEFLOW-97
```

TimeflowRanges for this Timeflow ...

```
{    "type": "ListResult",    "status": "OK",    "result": [        {            "typ
e": "TimeflowRange",            "startPoint": {                "type":
"OracleTimeflowPoint",                "location": "5475918",                "timestam
p": "2016-09-28T04:35:39.000Z",                "timeflow": "ORACLE_TIMEFLOW-97"
      },            "endPoint": {                "type": "OracleTimeflowPoint",
          "location": "5476181",                "timestamp":
"2016-09-28T04:37:53.000Z",                "timeflow": "ORACLE_TIMEFLOW-97"
  },            "provisionable": true        }    ],    "job":null,    "action":null,
    "total": 1,    "overflow": false}
```

Snapshot per Timeflow ...

snapshots:
@2016-09-28T04:35:39.826Z
@2016-09-28T04:37:38.537Z
@2016-09-28T04:37:51.273Z

Select Snapshot Name (copy-n-paste from the above list):

```
@2016-09-28T04:37:38.537Zsnapshot reference: ORACLE_SNAPSHOT-152{    "type":
"OracleSnapshot",    "reference": "ORACLE_SNAPSHOT-152",    "namespace":null,    "nam
e": "@2016-09-28T04:37:38.537Z",    "consistency": "CRASH_CONSISTENT",    "missingNon
LoggedData": false,    "container": "ORACLE_DB_CONTAINER-75",    "creationTime":
"2016-09-28T04:37:38.537Z",    "firstChangePoint": {        "type":
"OracleTimeflowPoint",        "location": "5476152",        "timestamp":null,        "
timeflow": "ORACLE_TIMEFLOW-97"    },    "latestChangePoint": {        "type":
"OracleTimeflowPoint",        "location": "5476157",        "timestamp":
"2016-09-28T04:37:38.000Z",        "timeflow": "ORACLE_TIMEFLOW-97"    },    "retenti
on": 0,    "timeflow": "ORACLE_TIMEFLOW-97",    "timezone": "US/Eastern,EDT-0400",
 "version": "11.2.0.4.0",    "runtime": {        "type": "OracleSnapshotRuntime",
    "provisionable": true,        "missingLogs":null    },    "temporary": false,    "
fromPhysicalStandbyVdb": false,    "fractionTimeflows":null,    "redoLogSizeInBytes":
52428800}Done
```

The following scripts demonstrate REFRESH and RESET/ROLLBACK for the Timeflow types of timestamp, snapshot and scn/lsn.

Filename: vdb_refresh_timestamp.sh
Filename: vdb_refresh_snapshot.sh
Filename: vdb_refresh_scn.sh

Filename: vdb_rollback_timestamp.sh
Filename: vdb_rollback_snapshot.sh
Filename: vdb_rollback_scn.sh

Usage: ./vdb_[*all_the_above_scripts*].sh [source_vdb_name]

Sample Usage: Copy-and-Paste the timeflow name to the "Select timeflow Name:" prompt. Enter any timestamp between the startPoint and endPoint values using the format:
[yyyy] - [MM] - [dd]T[HH] : [mm] : [ss] . [SSS]Z

```
$ ./vdb_rollback_timestamp.sh VBITTSession and Login Successful ...database container
reference: ORACLE_DB_CONTAINER-73 Timeflows API Timeflow Names:DB_REFRESH@2016-09-28T
00:13:13DB_PROVISION@2016-09-26T07:00:08DB_ROLLBACK@2016-09-26T07:45:30DB_REFRESH
@2016-09-26T12:20:57DB_ROLLBACK@2016-09-27T12:29:47
```

Select Timeflow Name (copy-n-paste from above list):

```
DB_ROLLBACK@2016-09-26T07:45:30Timeflow Reference: ORACLE_TIMEFLOW-89TimeflowRanges
for this timeflow ... {    "type": "ListResult",    "status": "OK",    "result":
[        {            "type": "TimeflowRange",            "startPoint":
{                "type": "OracleTimeflowPoint",                "location": "5474012",
               "timestamp": "2016-09-26T15:30:35.000Z",               "timeflow":
"ORACLE_TIMEFLOW-89"            },            "endPoint": {               "type":
"OracleTimeflowPoint",               "location": "5482482",               "timestam
p": "2016-09-26T21:15:17.000Z",               "timeflow": "ORACLE_TIMEFLOW-89"
      },            "provisionable": true        }    ],    "job": null,    "action":
null,    "total": 1,    "overflow": false}Timestamp Format "[yyyy][MM][dd]T[HH]:[mm]:
[ss].[SSS]Z"Enter Timestamp between Start and End Point values (exclude quotes): 2016-0
9-26T15:45:00.000Zjson> {    "type": "OracleRollbackParameters",    "timeflowPointPar
ameters": {        "type": "TimeflowPointTimestamp",        "timeflow":
"ORACLE_TIMEFLOW-89",        "timestamp": "2016-09-26T15:45:00.000Z"    },    "userna
me": ""}Job: JOB-515Current status as of Wed Sep 28 00:49:39 EDT 2016 : RUNNING 0%
Completed. . .Current status as of Wed Sep 28 00:51:10 EDT 2016 : RUNNING 73%
CompletedCurrent status as of Wed Sep 28 00:51:50 EDT 2016 : RUNNING 96%
CompletedJob: JOB-515 COMPLETED 100% Completed ...Done ...
```

API Cookbook: common tasks, workflows, and examples

These topics describe approaches to common tasks and workflows using the Delphix Engine API.

This section covers the following topics:

API cookbook: authentication

This API cookbook recipe describes how to create an authenticated session for using the Delphix Sever web services.

Before you can use any Delphix Web Service API's you need to create a session, and then authenticate the session by providing valid Delphix account credentials.

Create Delphix API Session

```
$ curl -s -X POST -k --data @- http://delphix-server/resources/json/delphix/session \
   -c ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
   "type": "APISession",
   "version": {
       "type": "APIVersion",
       "major": 1,
       "minor": 4,
       "micro": 3
   }
}
EOF

Response
{
   "status":"OK",
   "result": {
       "type":"APISession",
       "version": {
           "type": "APIVersion",
           "major": 1,
           "minor": 4,
           "micro": 3
       },
       "locale": "en_US",
       "client": null
   },
   "job": null
}
EOF
```

Once the session has been established, the next step is to authenticate to the server by executing the `LoginRequest` API. Unauthenticated sessions are prohibited from making any API calls other than this login request. The username can be either a system user or domain user, and the backend will authenticate using the appropriate method. This example illustrates logging in via curl using cookies created when the session was established:

```
$ curl -s -X POST -k --data @- http://delphix-server/resources/json/delphix/login \
-b cookies.txt -c cookies2.txt -H "Content-Type: application/json" <<EOF
{
"type": "LoginRequest",
"username": "delphix_user",
```

```
"password": "delphix_pass",
"target": "DOMAIN"
}
EOF
```

The new cookie (cookie2.txt) will need to be used in subsequent API requests. The login API currently only supports authentication by a password. There is no way to authenticate using any shared key or alternate authentication strategy.

> It is generally recommended to set the API session version to the highest level supported by your Delphix Engine.

API cookbook: host environment details

This API cookbook recipe describes how to obtain host environment details using the Delphix Engine API.

To obtain details about target host environments, list available `Environment` objects on the system. These environments can represent either a single host, or an Oracle cluster.

List Environment

```
curl -X GET -k http://delphix-server/resources/json/delphix/environment \
    -b ~/cookies.txt -H "Content-Type: application/json"
```

For single-host environments, the reference can be used to get information about the associated host. It is also possible to get information about all hosts (regardless of whether they are in a single-host environment or cluster) by omitting the `environment` query parameter.

List UNIX Environment

```
curl -X GET -k http://services.cloud.skytap.com:23173/resources/json/delphix/host?
environment=UNIX_HOST_ENVIRONMENT-1 \
    -b ~/cookies.txt -H "Content-Type: application/json"
```

For more information about the content of host objects, see the `/api/#Host` reference on your local Delphix Engine. Depending on the type of the host, additional information may be available through the following types:

- `UnixHost`
- `WindowsHost`

API cookbook: list alerts and list jobs

This API cookbook recipe describes how to obtain lists of jobs and alerts using the Delphix Engine API.

The `List Alerts` and `List Jobs` API calls can both accept the `toDate` and `fromDate` query parameters to limit rows returned. These parameters require the date to be expressed in ISO 8601 format.

List Alerts

```
$ curl -X GET -k http:/delphix-server/resources/json/delphix/alert \
    -b ~/cookies.txt -H "Content-Type: application/json"
```

For more information about the structure of an alert object, see the "/api/#Alert" link on your local Delphix Engine.

List Jobs (using fromDate)

```
$ curl -X GET -k http://delphix-server/resources/json/delphix/job?
addEvents=true&fromDate=2012-11-08T00:00:00.0000Z \
    -b ~/cookies.txt -H "Content-Type: application/json"
```

For more information about the structure of a job object, see the "/api/#Job" link on your local Delphix Engine.

API cookbook: list dSources and VDBs

This API cookbook recipe describes how to obtain a list of dSources and VDBs using the Delphix Engine API.

To obtain a list of dSources and VDBs, list available `Container` (also known as `database`) objects on the system:

List Databases

```
$ curl -X GET -k http://delphix-server/resources/json/delphix/database  \
    -b ~/cookies.txt -H "Content-Type: application/json"
```

For more information on the structure of a database object, see the `/api/#Container` reference on your local Delphix Engine. The following sub-types are available depending on the type of database:

- `OracleDatabaseContainer`
- `MSSqlDatabaseContainer`

Each database has zero or one source associated with it. This source could be a linked source, indicating that the database is a dSource, or it could be a virtual source, indicating that it is a VDB. If there are no sources, it is a detached dSource. The `parentContainer` property indicates the reference to the parent container, also indicating that the database is a VDB. To get runtime information about the source associated with the dSource or VDB, use the `Source` API with a `database` parameter set to the reference of the database in question.

List Sources

```
$ curl -X GET -k http://delphix-server/resources/json/delphix/source?
database=DB_CONTAINER-13  \
    -b ~/cookies.txt -H "Content-Type: application/json"
```

If the `virtual` flag is true, the source is a VDB, otherwise it is a dSource. For more information about the contents of a source object, see the `/api/#Source` reference on your local Delphix Engine. The following sub-types are available depending on the type of source:

- `OracleSource`
    - `OracleLinkedSource`
    - `OracleVirtualSource`
- `MSSqlSource`
    - `MSSqlLinkedSource`
    - `MSSqlVirtualSource`

API cookbook: list snapshots

This API cookbook recipe describes how to obtain a list of available snapshots for a VDB or dSource.

Snapshots represent points in time where a `sync` operation has occurred on either a dSource or VDB. Provisioning from snapshots is much faster than provisioning between snapshots, as the latter requires replaying LogSync records to arrive at the requested point. Given a reference to a database, the `snapshot` API can be used to retrieve the set of snapshots within the database. See the topic API Cookbook: List dSources and VDBs for information on how to obtain the database reference.

List Snapshots

```
curl -X GET -k http://services.cloud.skytap.com:23173/resources/json/delphix/
snapshot?database=ORACLE_DB_CONTAINER-15  \
    -b ~/cookies.txt -H "Content-Type: application/json"
```

For more information about the structure of a snapshot object, see the `/api/#TimeflowSnapshot` reference on your local Delphix Engine. Snapshots, while representing the point where provisioning will be most efficient, are not the only provisionable points within a database. To get a list of all provisioning points, use the `timeflowRange` API. This API is based on a Timeflow, which is the representation of one timeline within a database. Currently, all databases have a single Timeflow, though this may change in the future. To query for the ranges for a particular database, you will need to use the `currentTimeflow` member of the target database.

List Timeflow Ranges

```
curl -X POST -k --data @- http://services.cloud.skytap.com:23173/resources/json/
delphix/timeflow/ORACLE_TIMEFLOW-11/timeflowRanges  \
    -b ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
    "type": "TimeflowRangeParameters"
}
EOF
```

API cookbook: example provision of an Oracle VDB

This API cookbook recipe demonstrates how to provision an Oracle VDB using the Delphix Engine API.

In order to provision an Oracle VDB using the API, you need to provide a set of parameters of type **OracleProvisionParameters** (having already authenticated as per API Cookbook: Authentication).

There are a number of parameters you will need to know:

- **Group reference** - See the list operation in "/api#group" on your Delphix Engine
- **VDB name** - The name you want the new VDB to be called
- **Mount path** - Where to mount datasets on the target host.
- **DB/unique names** - The Oracle DB and unique names, often the same as the VDB name
- **Instance name/number** - The Oracle instance name and number to use (dictated by your environment, but often VDB name and 1)
- **Repository reference** - See the list operation on "/api#repository" on your Delphix Engine
- **TimeFlow point** - See API Cookbook: List Snapshots for more information on finding a TimeFlow point, as well as the reference at "/api#TimeflowPoint Parameters

You will need to use the structure of the OracleProvisionParameters object to fill it out, see "/api/#OracleProvisionParameters" for details on which fields are mandatory/optional.

Here is a minimal example using curl to communicate with the API, provisioning a VDB called "EGVDB" (authentication omitted)

```
curl -X POST -k --data @- http://delphix1.company.com/resources/json/delphix/database/provision \
    -b cookies.txt -H "Content-Type: application/json" <<EOF
{
    "container": {
        "group": "GROUP-2",
        "name": "EGVDB",
        "type": "OracleDatabaseContainer"
    },
    "source": {
        "type": "OracleVirtualSource",
        "mountBase": "/mnt/provision",
        "allowAutoVDBRestartOnHostReboot": true
    },
    "sourceConfig": {
        "type": "OracleSIConfig",
        "databaseName": "EGVDB",
        "uniqueName": "EGVDB",
        "repository": "ORACLE_INSTALL-3",
        "instance": {
            "type": "OracleInstance",
            "instanceName": "EGVDB",
            "instanceNumber": 1
        }
    },
    "timeflowPointParameters": {
        "type": "TimeflowPointLocation",
        "timeflow": "ORACLE_TIMEFLOW-123",
        "location": "3043123"
```

```
    },
    "type": "OracleProvisionParameters"
}
EOF
```

API cookbook: refresh VDB

This API cookbook recipe describes how to refresh a VDB using the Delphix Engine API.

To refresh a VDB you need a reference to the `Database` object, the location of the point to which you wish to refresh and the reference container associated with the object. See the topic API Cookbook: List dSources and VDBs for information on how to obtain the database reference and current Timeflow. The Timeflow point can be specified either by timestamp, by location (SCN), semantic location or Timeflow bookmark. The TimeflowPointSemantic type allows you to specify a semantically meaningful Timeflow location (i.e. the latest snapshot or the latest Timeflow point). The TimeflowPointBookmark type allows you to reference a previously created Timeflow bookmark. See API Cookbook: List Snapshots topic for information on how to determine provisionable points in the parent database.

Refresh VDB

```
curl -v -X POST -k --data @- http://delphix-server/resources/json/delphix/database/
ORACLE_DB_CONTAINER-13/refresh  \
    -b ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
       "type": "OracleRefreshParameters",
       "timeflowPointParameters": {
               "type": "TimeflowPointSemantic",
               "container": "ORACLE_DB_CONTAINER-1",
               "timeflow": "ORACLE_TIMEFLOW-13",
               "location": "LATEST_SNAPSHOT"
       }
}
EOF
```

For more information about the content of refresh parameters, see the `/api/#RefreshParameters` reference on your local Delphix Engine. Depending on the type of the database, the following parameter types are available:

- `OracleRefreshParameters`
- `MSSqlRefreshParameters`

API cookbook: rewind a VDB

This API cookbook recipe describes how to rewind a VDB using the Delphix Engine API.

To rewind a VDB, you need a reference to the `Database` object. See the topic, API Cookbook: List dSources and VDBs, for information on how to obtain the database reference. The following sample script includes a working example for creating a session, authenticating to the Delphix Engine, and rewinding the VDB. Please update the script variables to match your environment before using it.

```
#!/bin/bash
#
# sample script to start or stop a VDB.
#
# set this to the FQDN or IP address of the Delphix Engine
DE="192.168.2.131"
# set this to the Delphix admin user name
DELPHIX_ADMIN="delphix_admin"
# set this to the password for the Delphix admin user
DELPHIX_PASS="delphix"
# set this to the object reference for the VDB
VDB="ORACLE_DB_CONTAINER-57"
#
# create our session
$ curl -s -X POST -k --data @- http://delphix-server/resources/json/delphix/session \

   -c ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
   "type": "APISession",
   "version": {
       "type": "APIVersion",
       "major": 1,
       "minor": 4,
       "micro": 3
   }
}
EOF
{
   "status":"OK",
   "result": {
       "type":"APISession",
       "version": {
           "type": "APIVersion",
           "major": 1,
           "minor": 4,
           "micro": 3
       },
       "locale": "en_US",
       "client": null
   },
   "job": null
}
EOF
```

```
echo
#
# authenticate to the DE
$ curl -s -X POST -k --data @- http://delphix-server/resources/json/delphix/login \
   -b ~/cookies.txt -c ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
   "type": "LoginRequest",
   "username": "delphix_username",
   "password": "delphix_password"
}

EOF
echo
#
# rewind VDB
curl -s -X POST -k --data @- http://${DE}/resources/json/delphix/database/${VDB}/
rollback \
    -b ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
    "type": "OracleRollbackParameters",
    "timeflowPointParameters": {
    "type" : "TimeflowPointSnapshot",
    "snapshot" : "ORACLE_SNAPSHOT-172"
    }
}
EOF
echo
```

> ⚠ While rewinding a VDB, you can use different parameter types. In the above example, "timeflowPointParameters" type is used as "TimeflowPointSnapshot" and an appropriate snapshot name is provided. Instead of "TimeflowPointSnapshot", you can also choose from "TimeflowPointLocation" or "TimeflowPointTimestamp" or "TimeflowPointBookmark" etc. and pass the relevant parameters.

You can list your Snapshots by following the instructions on API Cookbook: List Snapshots

API cookbook: stop/start a VDB

This API cookbook recipe describes how to stop and start a VDB using the Delphix Engine API.

To stop or start a VDB, you need a reference to the `Database` object. See the topic, API Cookbook: List dSources and VDBs, for information on how to obtain the database reference. The following script example includes working examples for creating a session, authenticating to the Delphix Engine, and stopping or starting a VDB. Please update the script variables to match your environment before using it. This script requires a single argument which is 'start' or 'stop'.

```
#!/bin/bash
#
# sample script to start or stop a VDB.
#
# set this to the FQDN or IP address of the Delphix Engine
DE="192.168.2.131"
# set this to the Delphix admin user name
DELPHIX_ADMIN="delphix_admin"
# set this to the password for the Delphix admin user
DELPHIX_PASS="delphix"
# set this to the object reference for the VDB
VDB="ORACLE_VIRTUAL_SOURCE-5"
#
# create our session
curl -s -X POST -k --data @- http://${DE}/resources/json/delphix/session \
    -c ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
    "type": "APISession",
    "version": {
        "type": "APIVersion",
        "major": 1,
        "minor": 11,
        "micro": 8
    }
}
EOF
echo
#
# authenticate to the DE
curl -s -X POST -k --data @- http://${DE}/resources/json/delphix/login \
    -c ~/cookies.txt -b ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
    "type": "LoginRequest",
    "username": "${DELPHIX_ADMIN}",
    "password": "${DELPHIX_PASS}"
}
EOF
echo
#
# start or stop the vdb based on the argument passed to the script
case $1 in
start)
```

```
  curl -s -X POST -k http://${DE}/resources/json/delphix/source/${VDB}/start \
    -c ~/cookies.txt -b ~/cookies.txt -H "Content-Type: application/json"
;;
stop)
  curl -s -X POST -k http://${DE}/resources/json/delphix/source/${VDB}/stop \
    -c ~/cookies.txt -b ~/cookies.txt -H "Content-Type: application/json"
;;
*)
  echo "Unknown option: $1"
;;
esac
echo
```

API cookbook: creating a database template in Delphix self-service

> ⚠ Jet Stream is now known as Delphix Self-Service.

Delphix Self-Service administrators can use this API cookbook recipe to create a database template on Delphix Self-Service using the Delphix Engine API.

> ⓘ The following script is for educational and demonstration purposes only and is not supported by Delphix.

The following script can be downloaded by selecting **createDBTemplate.sh**

Create Self-Service Database Template

```
#!/bin/bash

# A sample script for calls to the API. This one creates a Jet Stream Template.

##### Constants

# Describes a Delphix software revision.
# Please change version are per your Delphix Engine CLI, if different
VERSION="1.8.0"

##### Default Values. These can be overwriten with optional arguments.
engine="172.16.151.154"
username="delphix_admin"
password="landshark"

##examples##
# Create template with mandatory params
#./createBranch.sh -d 172.16.151.154 -u delphix_admin:landshark -n <sourceName>
<templateName> <containerName>
#Ex ./createBranch.sh -d 172.16.151.154 -u delphix_admin:landshark -n oraclesrc
template1 ORACLE_DB_CONTAINER-191
# Create template with adding optional params, Notes and Description
#./createDBTemplate.sh -n <sourceName> -N "<templateNotes>" -D "<AnyDescription>"
 <templateName> <containerName>

## NOTE: This script is to add one source per template and it will not add any
properties for template, container or source.

##### Functions

# Help Menu
function usage {
    echo "Usage: createDBTemplate.sh [[-h] | options...] <template_name>
<source_container>"
    echo "Create a Jet Stream Dat Template."
    echo ""
    echo "Positional arguments"
    echo "  <template_name>"
```

```
    echo "  <source_container>"
    echo ""
    echo "Optional Arguments:"
    echo "  -h                Show this message and exit"
    echo "  -d                Delphix engine IP address or host name, otherwise
revert to default"
    echo "  -u USER:PASSWORD  Server user and password, otherwise revert to default"
    echo "  -n                source name to display on JS template screen"
    echo "  -N                template notes, if any. Type: String"
    echo "  -D                source description, if any. Type: String"
}

# Create Our Session, including establishing the API version.
function create_session
{
    # Pulling the version into parts. The {} are necessary for string manipulation.
    # Strip out longest match following "."  This leaves only the major version.
    major=${VERSION%%.*}
    # Strip out the shortest match preceding "." This leaves minor.micro.
    minorMicro=${VERSION#*.}
    # Strip out the shortest match followint "." This leaves the minor version.
    minor=${minorMicro%.*}
    # Strip out the longest match preceding "." This leaves the micro version.
    micro=${VERSION##*.}

    # Quick note about the <<-. If the redirection operator << is followed by a -
(dash), all leading TAB from the document data will be
    # ignored. This is useful to have optical nice code also when using here-
documents. Otherwise you must have the EOF be on a line by itself,
    # no parens, no tabs or anything.

    echo "creating session..."
    result=$(curl -s -S -X POST -k --data @- http://${engine}/resources/json/delphix/
session \
        -c ~/cookies.txt -H "Content-Type: application/json" <<-EOF
    {
        "type": "APISession",
        "version": {
            "type": "APIVersion",
            "major": $major,
            "minor": $minor,
            "micro": $micro
        }
    }
    EOF)

    check_result
}

# Authenticate the DE for the provided user.
function authenticate_de
{
    echo "authenticating delphix engine..."
```

```
    echo ${engine}
    echo ${username}
    echo ${password}
    result=$(curl -s -S -X POST -k --data @- http://${engine}/resources/json/delphix/
login \
        -b ~/cookies.txt -H "Content-Type: application/json" <<-EOF
    {
        "type": "LoginRequest",
        "username": "${username}",
        "password": "${password}"
    }
    EOF)

    check_result
}

function create_template
{

   paramString="\"type\": \"JSDataTemplateCreateParameters\",\"name\":
\"$templateName\","

   if [[ -n $templatenotes ]]
    then
       paramString="$paramString \"notes\": \"$templatenotes\","
   fi

    paramString="$paramString \"dataSources\": [{\"type\":
\"JSDataSourceCreateParameters\",
                              \"container\": \"$sourceContainer\",
                              \"source\": {\"type\": \"JSDataSource\",
                              \"priority\": 1,
                              \"name\": \"$sourcename\""

   if [[ -n $sourcedesc ]]
    then
       paramString="$paramString ,\"description\": \"$sourcedesc\"}}]"
    else
       paramString="$paramString }}]"
   fi

    result=$(curl -s -X POST -k --data @- http://${engine}/resources/json/delphix/
jetstream/template \
        -b ~/cookies.txt -H "Content-Type: application/json" <<-EOF
    {
        $paramString
    }
    EOF)

    check_result

    echo "New JetStream template $templateName successfully created"
```

```
}

# Check the result of the curl. If there are problems, inform the user then exit.
function check_result
{
    exitStatus=$?
    if [ $exitStatus -ne 0 ]
    then
        echo "command failed with exit status $exitStatus"
        exit 1
    elif [[ $result != *"OKResult"* ]]
    then
        echo ""
        echo $result
        exit 1
    fi
}

##### Main

while getopts "u:d:n:N:D:h" flag; do
    case "$flag" in
        u )             username=${OPTARG%:*}
                        password=${OPTARG##*:}
                        ;;
        d )             engine=$OPTARG
                        ;;
        n )             sourcename=$OPTARG
                        ;;
        N )             templatenotes=$OPTARG
                        ;;
        D )             sourcedesc=$OPTARG
                        ;;
        h )             usage
                        exit
                        ;;
        * )             usage
                        exit 1

    esac

done


# Shift the parameters so we only have the positional arguments left
shift $((OPTIND-1))

# Check that there are 2 positional arguments
if [ $# != 2 ]
then
  echo "usage1"
    usage
    exit 1
```

```
fi

# Get the two positional arguments
templateName=$1
shift
sourceContainer=$1

create_session
authenticate_de
create_template
```

API cookbook: creating a container in Delphix self-service

Delphix Self-Service administrators can use this API cookbook recipe to create a container on Delphix Self-Service (Jet Stream) using the Delphix Engine API.

> The following script is for educational and demonstration purposes only and is not supported by Delphix.

This script can be downloaded by selecting 

Create Self-Service Container

```
#!/bin/bash

# A sample script for calls to the API. This one creates a Jet Stream container.

##### Constants

# Describes a Delphix software revision.
# Please change version are per your Delphix Engine CLI, if different
VERSION="1.11.10"


##### Default Values. These can be overwriten with optional arguments.
engine="10.110.248.170"
username="admin"
password="delphix"

##examples##
# Create container from latest point in time
#./createContainer.sh -n "testsource" testcont ORACLE_DB_CONTAINER-269
 JS_DATA_TEMPLATE-13
# Create container from specific bookmark
#./createContainer.sh -n "testsource" -b JS_BOOKMARK-77 testcont ORACLE_DB_CONTAINER-2
69 JS_DATA_TEMPLATE-13
# Create container from specific point in time
#./createContainer.sh -n "testsource" -t "2016-08-08T10:00:00.000Z" -B JS_BRANCH-50
 testcont ORACLE_DB_CONTAINER-269 JS_DATA_TEMPLATE-13
```

```
#NOTE: this script will add one container and assign one owner for the container.

##### Functions

# Help Menu
function usage {
    echo "Usage: createContainer.sh [[-h] | options...] <containername> <vdb>
<template>"
    echo "Create a Jet Stream Bookmark on the given branch."
    echo ""
    echo "Positional arguments"
    echo "  <name>"
    echo "  <container> format JS_DATA_CONTAINER-<n>"
    echo ""
    echo "Optional Arguments:"
    echo "  -h                Show this message and exit"
    echo "  -d                Delphix engine IP address or host name, otherwise
revert to default"
    echo "  -u USER:PASSWORD  Server user and password, otherwise revert to default"
    echo "  -n                SourceName need to display for container.(Mandatory)"
    echo "  -b                Bookmark name from which need to create container. If
no bookmark is included, the branch will be created at the latest point in time.
Type: string. Format JS_BOOKMARK-<n> (Optional)"
    echo "  -t                The time at which the branch should be created. This
must be accompanied with branch name from which need to pick up time. Type: date,
must be in ISO 8601 extended format [yyyy]-[MM]-[dd]T[HH]:[mm]:[ss].[SSS]Z"
    echo "  -B                Branch name from which need to create new container, at
specific time. Type: string. Format JS_BRANCH-<n> (Optional)"
    echo "  -N                Optional container notes, if need to add any. Type:
String"
    echo "  -o                Optional owner, to whom we need to assign this
container. Type: String. Format USER-<n>"
}

# Create Our Session, including establishing the API version.
function create_session
{
    # Pulling the version into parts. The {} are necessary for string manipulation.
    # Strip out longest match following "."  This leaves only the major version.
    major=${VERSION%%.*}
    # Strip out the shortest match preceding "." This leaves minor.micro.
    minorMicro=${VERSION#*.}
    # Strip out the shortest match followint "." This leaves the minor version.
    minor=${minorMicro%.*}
    # Strip out the longest match preceding "." This leaves the micro version.
    micro=${VERSION##*.}

    # Quick note about the <<-. If the redirection operator << is followed by a -
(dash), all leading TAB from the document data will be
    # ignored. This is useful to have optical nice code also when using here-
documents. Otherwise you must have the EOF be on a line by itself,
    # no parens, no tabs or anything.
```

```
    echo "creating session..."
    result=$(curl -s -S -X POST -k --data @- http://${engine}/resources/json/delphix/
session \
        -c ~/cookies.txt -H "Content-Type: application/json" <<-EOF
    {
        "type": "APISession",
        "version": {
            "type": "APIVersion",
            "major": $major,
            "minor": $minor,
            "micro": $micro
        }
    }
    EOF)

    check_result
}

# Authenticate the DE for the provided user.
function authenticate_de
{
    echo "authenticating delphix engine..."
    result=$(curl -s -S -X POST -k --data @- http://${engine}/resources/json/delphix/
login \
        -b ~/cookies.txt -c ~/cookies.txt -H "Content-Type: application/json" <<-EOF
    {
        "type": "LoginRequest",
        "username": "${username}",
        "password": "${password}"
    }
    EOF)

    check_result
}

function create_container
{

    # If there is not timeInput and no bookmark name, we need to use
JSTimelinePointLatestTimeInput.
    if [[ -z $inputTime  &&  -z $bookmark ]]
    then
        pointParams="\"timelinePointParameters\":{
            \"sourceDataLayout\": \"${template}\",
            \"type\":\"JSTimelinePointLatestTimeInput\"}"

   # If there is a timeInput and no bookmark name, we need to use Input Time.

    elif [[ -n $inputTime  && -n $branchRef && -z $bookmark ]]
     then
        pointParams="\"timelinePointParameters\":{
            \"time\":\"${inputTime}\",
```

```
            \"branch\":\"${branchRef}\",
            \"type\":\"JSTimelinePointTimeInput\"}"

   # If there is a bookmark name and no time input, we need to use bookmark

   elif [[ -z $inputTime  &&  -n $bookmark ]]
   then
        pointParams="\"timelinePointParameters\":{
        \"bookmark\":\"${bookmark}\",
        \"type\":\"JSTimelinePointBookmarkInput\"}"

    fi

    # These are the required parameters.

    paramString="\"type\": \"JSDataContainerCreateWithRefreshParameters\",
                 \"name\": \"${containerName}\",
                 \"template\": \"${template}\","


    paramString="$paramString \"dataSources\": [{\"type\":
\"JSDataSourceCreateParameters\",
                \"container\": \"${VDB}\",
                \"source\": {
                \"type\": \"JSDataSource\",
                \"priority\": 1,
                \"name\": \"${sourceName}\""

    if [[ -n $sourcedesc ]]
    then
       paramString="$paramString ,\"description\": \"${sourcedesc}\"}}],"
    else
       paramString="$paramString }}],"
   fi

    if [[ -n $containerNotes ]]
    then
       paramString="$paramString \"notes\": \"${containerNotes}\","
   fi

   if [[ -n $owners ]]
   then
      paramString="$paramString \"owners\": [\"${owners}\"],"
   fi

    paramString="$paramString ${pointParams}"


    result=$(curl -s -X POST -k --data @- http://${engine}/resources/json/delphix/
jetstream/container \
        -b ~/cookies.txt -H "Content-Type: application/json" <<-EOF
    {
        $paramString
```

```
    }
    EOF)


    check_result


    echo "confirming job completed successfully..."
    # Get everything in the result that comes after job.
    temp=${result#*\"job\":\"}
    # Get rid of everything after
    jobRef=${temp%%\"*}


    result=$(curl -s -X GET -k http://${engine}/resources/json/delphix/job/${jobRef}
\
    -b ~/cookies.txt -H "Content-Type: application/json")

    # Get everything in the result that comes after job.
    temp=${result#*\"jobState\":\"}
    # Get rid of everything after
    jobState=${temp%%\"*}


    check_result

    while [ $jobState = "RUNNING" ]
    do
        sleep 1
        result=$(curl -s -X GET -k http://${engine}/resources/json/delphix/job/$
{jobRef} \
        -b ~/cookies.txt -H "Content-Type: application/json")

        # Get everything in the result that comes after job.
        temp=${result#*\"jobState\":\"}
        # Get rid of everything after
        jobState=${temp%%\"*}

        check_result

    done

    if [ $jobState = "COMPLETED" ]
    then
        echo "successfully created container $containerName"
    else
        echo "unable to create container"
        echo result
    fi

}

# Check the result of the curl. If there are problems, inform the user then exit.
```

```
function check_result
{
    exitStatus=$?
    if [ $exitStatus -ne 0 ]
    then
        echo "command failed with exit status $exitStatus"
        exit 1
    elif [[ $result != *"OKResult"* ]]
    then
        echo ""
        echo $result
        exit 1
    fi
}

##### Main

while getopts "u:d:b:t:B:D:n:N:o:h" flag; do
    case "$flag" in
        u )             username=${OPTARG%:*}
                        password=${OPTARG##*:}
                        ;;
        d )             engine=$OPTARG
                        ;;
        b )             bookmark=$OPTARG
                        ;;
        t )             inputTime=$OPTARG
                        ;;
        B )             branchRef=$OPTARG
                        ;;
        D )             sourcedesc=$OPTARG
                        ;;
        n )             sourceName=$OPTARG
                        ;;
        N )             containerNotes=$OPTARG
                        ;;
        o )             owners=$OPTARG
                        ;;
        h )             usage
                        exit
                        ;;
        * )             usage
                        exit 1
    esac

done


# Shift the parameters so we only have the positional arguments left
shift $((OPTIND-1))

# Check that there are 3 positional arguments
if [ $# != 3 ]
```

```
then
    usage
    exit 1
fi

# Get the three positional arguments
containerName=$1
shift
VDB=$1
shift
template=$1

create_session
authenticate_de
create_container
```

API cookbook: refreshing a container in Delphix self-service

Delphix Self-Service administrators can use this API cookbook recipe to refresh a container in Delphix Self-Service (Jet Stream) using the Delphix Engine API.

> The following script is for educational and demonstration purposes only and is not supported by Delphix.

This script can be downloaded by selecting **refreshContainer.sh**

Refreshing a Self-Service container

```
#!/bin/bash

# A sample script for calls to the API. This one refresh Jet Stream container.

##### Constants

# Describes a Delphix software revision.
# Please change version are per your Delphix Engine CLI, if different
VERSION="1.8.0"


##### Default Values. These can be overwriten with optional arguments.
engine="172.16.151.154"
username="delphix_admin"
password="landshark"

##examples##
# Refresh container from latest point in time of Template
#./refreshContainer.sh -T JS_DATA_TEMPLATE-13 JS_DATA_CONTAINER-20
# Refresh container from specific bookmark
#./refreshContainer.sh -b JS_BOOKMARK-76 JS_DATA_CONTAINER-20
# Refresh container from specific point in time of branch
#./refreshContainer.sh -t "2016-08-08T10:00:00.000Z" -B JS_BRANCH-50
 JS_DATA_CONTAINER-20

##### Functions

# Help Menu
function usage {
    echo "Usage: refreshContainer.sh [[-h] | options...] <containername> <template>"
    echo "Create a Jet Stream Bookmark on the given branch."
    echo ""
    echo "Positional arguments"
    echo "  <containerName>"
    echo "  <template>"
    echo ""
    echo "Optional Arguments:"
    echo "  -h                Show this message and exit"
    echo "  -d                Delphix engine IP address or host name, otherwise
revert to default"
    echo "  -u USER:PASSWORD  Server user and password, otherwise revert to default"
```

```
    echo "  -T                template reference from which need to refresh from
latest point in time"
    echo "  -b                Bookmark name from which need to refresh container. If
no bookmark is included, the branch will be created at the latest point in time.
Type: string. Format JS_BOOKMARK-<n> (Optional)"
    echo "  -t                The time from where the container should be refreshed.
This must be accompanied with branch name from which need to pick up time. Type:
date, must be in ISO 8601 extended format [yyyy]-[MM]-[dd]T[HH]:[mm]:[ss].[SSS]Z"
    echo "  -B                Branch name from which need to refresh container, at
specific time. Type: string. Format JS_BRANCH-<n> (Optional)"
}

# Create Our Session, including establishing the API version.
function create_session
{
    # Pulling the version into parts. The {} are necessary for string manipulation.
    # Strip out longest match following "."  This leaves only the major version.
    major=${VERSION%%.*}
    # Strip out the shortest match preceding "." This leaves minor.micro.
    minorMicro=${VERSION#*.}
    # Strip out the shortest match followint "." This leaves the minor version.
    minor=${minorMicro%.*}
    # Strip out the longest match preceding "." This leaves the micro version.
    micro=${VERSION##*.}

    # Quick note about the <<-. If the redirection operator << is followed by a -
(dash), all leading TAB from the document data will be
    # ignored. This is useful to have optical nice code also when using here-
documents. Otherwise you must have the EOF be on a line by itself,
    # no parens, no tabs or anything.

    echo "creating session..."
    result=$(curl -s -S -X POST -k --data @- http://${engine}/resources/json/delphix/
session \
        -c ~/cookies.txt -H "Content-Type: application/json" <<-EOF
    {
        "type": "APISession",
        "version": {
            "type": "APIVersion",
            "major": $major,
            "minor": $minor,
            "micro": $micro
        }
    }
    EOF)

    check_result
}

# Authenticate the DE for the provided user.
function authenticate_de
{
    echo "authenticating delphix engine..."
```

```
    result=$(curl -s -S -X POST -k --data @- http://${engine}/resources/json/delphix/
login \
        -b ~/cookies.txt -c ~/cookies.txt -H "Content-Type: application/json" <<-EOF
    {
        "type": "LoginRequest",
        "username": "${username}",
        "password": "${password}"
    }
    EOF)

    check_result
}

function restore_container
{

    # If there is not timeInput and no bookmark name, we need to use
JSTimelinePointLatestTimeInput from template.
    if [[ -n $template && -z $inputTime  &&  -z $bookmark ]]
    then
        pointParams="\"type\": \"JSTimelinePointLatestTimeInput\",
                    \"sourceDataLayout\": \"${template}\""

   # If there is a timeInput and no bookmark name, we need to use Input Time.

    elif [[ -n $inputTime  && -n $branchRef && -z $bookmark && -z $template ]]
     then
        pointParams="\"type\": \"JSTimelinePointTimeInput\",
                 \"branch\": \"${branchRef}\",
                 \"time\": \"${inputTime}\""

   # If there is a bookmark name and no time input, we need to use bookmark

   elif [[ -n $bookmark && -z $template && -z $inputTime ]]
   then
        pointParams="\"type\": \"JSTimelinePointBookmarkInput\",
                    \"bookmark\": \"${bookmark}\""
    fi

    echo "pointParams" $pointParams



    result=$(curl -s -X POST -k --data @- http://${engine}/resources/json/delphix/
jetstream/container/${containerRef}/restore \
        -b ~/cookies.txt -H "Content-Type: application/json" <<-EOF
    {
        $pointParams
    }
    EOF)


    check_result
```

```
    echo "confirming job completed successfully..."
    # Get everything in the result that comes after job.
    temp=${result#*\"job\":\"}
    # Get rid of everything after
    jobRef=${temp%%\"*}


    result=$(curl -s -X GET -k http://${engine}/resources/json/delphix/job/${jobRef}
\
    -b ~/cookies.txt -H "Content-Type: application/json")

    # Get everything in the result that comes after job.
    temp=${result#*\"jobState\":\"}
    # Get rid of everything after
    jobState=${temp%%\"*}


    check_result

    while [ $jobState = "RUNNING" ]
    do
        sleep 1
        result=$(curl -s -X GET -k http://${engine}/resources/json/delphix/job/$
{jobRef} \
        -b ~/cookies.txt -H "Content-Type: application/json")

        # Get everything in the result that comes after job.
        temp=${result#*\"jobState\":\"}
        # Get rid of everything after
        jobState=${temp%%\"*}

        check_result

    done

    if [ $jobState = "COMPLETED" ]
    then
        echo "successfully refresh container $containerName"
    else
        echo "unable to refresh container"
        echo result
    fi

}

# Check the result of the curl. If there are problems, inform the user then exit.
function check_result
{
    exitStatus=$?
    if [ $exitStatus -ne 0 ]
    then
```

```
        echo "command failed with exit status $exitStatus"
        exit 1
    elif [[ $result != *"OKResult"* ]]
    then
        echo ""
        echo $result
        exit 1
    fi
}

##### Main

while getopts "u:d:T:b:t:B:h" flag; do
    case "$flag" in
        u )             username=${OPTARG%:*}
                        password=${OPTARG##*:}
                        ;;
        d )             engine=$OPTARG
                        ;;
        T )             template=$OPTARG
                        ;;
        b )             bookmark=$OPTARG
                        ;;
        t )             inputTime=$OPTARG
                        ;;
        B )             branchRef=$OPTARG
                        ;;
        h )             usage
                        exit
                        ;;
        * )             usage
                        exit 1
    esac

done


# Shift the parameters so we only have the positional arguments left
shift $((OPTIND-1))

# Check that there are 1 positional arguments
if [ $# != 1 ]
then
    usage
    exit 1
fi

# Get the one positional arguments
containerRef=$1

create_session
authenticate_de
restore_container
```

API cookbook: creating a user in Delphix self-service

Delphix Self-Service administrators can use this API cookbook recipe to create a user on Delphix Self-Service (Jet Stream) using the Delphix Engine API.

> The following script is for educational and demonstration purposes only and is not supported by Delphix.

This script can be downloaded by selecting **createJSUser.sh**

Creating a Self-Service User

```
#!/bin/bash

# A sample script for calls to the API. This one creates a Jet Stream user.

##### Constants

# Describes a Delphix software revision.
# Please change version are per your Delphix Engine CLI, if different
VERSION="1.11.9"


##### Default Values. These can be overwriten with optional arguments.
engine="10.43.90.86"
username="admin"
password="delphix"

##examples##
# Create user with NATIVE authentication
#./createJSUser.sh -P <password> NATIVE <username>
# Create user with LDAP authentication
#./createJSUser.sh -r <principal> <LDAP username>

##### Functions

# Help Menu
function usage {
echo "Usage: createJSUser.sh [[-h] | options...] <auth> <newjsuser>"
echo "Create a Jet Stream Only user."
echo ""
echo "Positional arguments"
echo " <auth>"
echo " <newjsuser>"
echo ""
echo "Optional Arguments:"
echo " -h Show this message and exit"
echo " -d Delphix engine IP address or host name, otherwise revert to default"
echo " -u USER:PASSWORD Server user and password, otherwise revert to default"
echo " -P password for NATIVE authentication"
echo " -f firstName of user"
echo " -l lastName of user"
echo " -e emailAddress of user"
```

```
echo " -o homePhoneNumber of user"
echo " -m mobilePhoneNumber of user"
echo " -w workPhoneNumber of user"
echo " -r principal for LDAP authentication"
}

# Create Our Session, including establishing the API version.
function create_session
{
# Pulling the version into parts. The {} are necessary for string manipulation.
# Strip out longest match following "." This leaves only the major version.
major=${VERSION%%.*}
# Strip out the shortest match preceding "." This leaves minor.micro.
minorMicro=${VERSION#*.}
# Strip out the shortest match followint "." This leaves the minor version.
minor=${minorMicro%.*}
# Strip out the longest match preceding "." This leaves the micro version.
micro=${VERSION##*.}

# Quick note about the <<-. If the redirection operator << is followed by a - (dash),
all leading TAB from the document data will be
# ignored. This is useful to have optical nice code also when using here-documents.
Otherwise you must have the EOF be on a line by itself,
# no parens, no tabs or anything.

echo "creating session..."
result=$(curl -s -S -X POST -k --data @- http://${engine}/resources/json/delphix/
session \
-c ~/cookies.txt -H "Content-Type: application/json" <<-EOF
{
"type": "APISession",
"version": {
"type": "APIVersion",
"major": $major,
"minor": $minor,
"micro": $micro
}
}
EOF)

check_result
}

# Authenticate the DE for the provided user.
function authenticate_de
{
echo "authenticating delphix engine..."
result=$(curl -s -S -X POST -k --data @- http://${engine}/resources/json/delphix/
login \
-b ~/cookies.txt -c ~/cookies.txt -H "Content-Type: application/json" <<-EOF
{
"type": "LoginRequest",
"username": "${username}",
```

```
"password": "${password}"
}
EOF)

check_result
}

function create_user
{
# Check on authorization type

if [[ $authtype = "NATIVE" && -n $userpwd ]]
then
pointParams="\"authenticationType\":\"$authtype\",
\"credential\":{
\"type\":\"PasswordCredential\",
\"password\":\"$userpwd\"}"

elif [[ $authtype = "LDAP" && -n $principal ]]
then
pointParams="\"authenticationType\":\"$authtype\",
\"principal\":\"$principal\""

fi

# These are the required parameters.
paramString="
\"type\": \"User\",
\"name\": \"${newjsuser}\","


# Fill in optional parameters if there are any.
if [[ -n $firstname ]]
then
paramString="$paramString \"firstName\": \"$firstname\","
fi

if [[ -n $lastname ]]
then
paramString="$paramString \"lastName\": \"$lastname\","
fi

if [[ -n $emailaddress ]]
then
paramString="$paramString \"emailAddress\": \"$emailaddress\","
fi

if [[ -n $homephone ]]
then
paramString="$paramString \"homePhoneNumber\": \"$homephone\","
fi

if [[ -n $mobilephone ]]
```

```
then
paramString="$paramString \"mobilePhoneNumber\": \"$mobilephone\","
fi

if [[ -n $workphone ]]
then
paramString="$paramString \"workPhoneNumber\": \"$workphone\","
fi


paramString="$paramString
${pointParams}"

result=$(curl -s -X POST -k --data @- http://${engine}/resources/json/delphix/user \
-b ~/cookies.txt -H "Content-Type: application/json" <<-EOF
{
$paramString
}
EOF)

check_result

# Extracting USER ID from result
temp=${result#*\"result\":\"}
userRef=${temp%%\"*}

echo "New user $newjsuser successfully created"

##### ROLE-3 is Jet Stream Role

result=$(curl -s -X POST -k --data @- http://${engine}/resources/json/delphix/
authorization \
-b ~/cookies.txt -H "Content-Type: application/json" <<-EOF
{
"type": "Authorization",
"role": "ROLE-3",
"target": "$userRef",
"user": "$userRef"
}
EOF)

check_result

echo "Assigned Jet Stream Role to user $newjsuser"

}

# Check the result of the curl. If there are problems, inform the user then exit.
function check_result
{
exitStatus=$?
if [ $exitStatus -ne 0 ]
then
```

```
echo "command failed with exit status $exitStatus"
exit 1
elif [[ $result != *"OKResult"* ]]
then
echo ""
echo $result
exit 1
fi
}

##### Main

while getopts "u:d:P:r:f:l:e:o:m:w:h" flag; do
case "$flag" in
u ) username=${OPTARG%:*}
password=${OPTARG##*:}
;;
d ) engine=$OPTARG
;;
P ) userpwd=$OPTARG
;;
r ) principal=$OPTARG
;;
f ) firstname=$OPTARG
;;
l ) lastname=$OPTARG
;;
e ) emailaddress=$OPTARG
;;
o ) homephone=$OPTARG
;;
m ) mobilephone=$OPTARG
;;
w ) workphone=$OPTARG
;;
h ) usage
exit
;;
* ) usage
exit 1

esac

done


# Shift the parameters so we only have the positional arguments left
shift $((OPTIND-1))

# Check that there are 2 positional arguments
if [ $# != 2 ]
then
echo "usage1"
```

```
usage
exit 1
fi

# Get the two positional arguments
authtype=$1
shift
newjsuser=$1

create_session
authenticate_de
create_user
```

API cookbook: create a bookmark in Delphix self-service

Delphix Self-Service administrators can use this API cookbook recipe to create a branch on Delphix Self-Service (Jet Stream) using the Delphix Engine API.

> The following script is for educational and demonstration purposes only and is not supported by Delphix.

This script can be downloaded by selecting **createBranch.sh**

Create Self-Service Branch

```
#!/bin/bash

# A sample script for calls to the API. This one creates a Jet Stream bookmark.

##### Constants

# Describes a Delphix software revision.
# Please change version are per your Delphix Engine CLI, if different
VERSION="1.8.0"


##### Default Values. These can be overwriten with optional arguments.
engine="172.16.151.154"
username="delphix_admin"
password="landshark"

shared=false

##example##
#./createBookmark.sh -d 172.16.151.154 -u delphix_admin:landshark -p "bookmark test"
 -e "2016-07-27T23:38:56.453Z" -t "2016-07-27T01:45:56.453Z" -T
"[tag1,tag2,tag3,tag4,tag5]" bkmrk3 JS_BRANCH-41


##### Functions

# Help Menu
function usage {
    echo "Usage: createBookmark.sh [[-h] | options...] <name> <branch>"
    echo "Create a Jet Stream Bookmark on the given branch."
    echo ""
    echo "Positional arguments"
    echo "  <name>"
    echo "  <branch>"
    echo ""
    echo "Optional Arguments:"
    echo "  -h                Show this message and exit"
    echo "  -d                Delphix engine IP address or host name, otherwise
revert to default"
    echo "  -u USER:PASSWORD  Server user and password, otherwise revert to default"
    echo "  -D                Description of this bookmark. Type: string"
```

```
    echo "  -e                A policy will automatically delete this bookmark at
this time. If not present the bookmark will be kept until manually deleted. Type:
date, must be in ISO 8601 extended format [yyyy]-[MM]-[dd]T[HH]:[mm]:[ss].[SSS]Z"
    echo "  -s                Present if need to make bookmark in shared mode"
    echo "  -t                The time at which the bookmark should be created. If no
time is included, the bookmark will be created at the latest point in time. Type:
date, must be in ISO 8601 extended format [yyyy]-[MM]-[dd]T[HH]:[mm]:[ss].[SSS]Z"
    echo "  -T                A set of user-defined labels for this bookmark. No
spaces allowed. Array of Type: string. In format, [tag1,tag2,..] "
}

# Create Our Session, including establishing the API version.
function create_session
{
    # Pulling the version into parts. The {} are necessary for string manipulation.
    # Strip out longest match following "."  This leaves only the major version.
    major=${VERSION%%.*}
    # Strip out the shortest match preceding "." This leaves minor.micro.
    minorMicro=${VERSION#*.}
    # Strip out the shortest match followint "." This leaves the minor version.
    minor=${minorMicro%.*}
    # Strip out the longest match preceding "." This leaves the micro version.
    micro=${VERSION##*.}

    # Quick note about the <<-. If the redirection operator << is followed by a -
(dash), all leading TAB from the document data will be
    # ignored. This is useful to have optical nice code also when using here-
documents. Otherwise you must have the EOF be on a line by itself,
    # no parens, no tabs or anything.

    echo "creating session..."
    result=$(curl -s -S -X POST -k --data @- http://${engine}/resources/json/delphix/
session \
        -c ~/cookies.txt -H "Content-Type: application/json" <<-EOF
    {
        "type": "APISession",
        "version": {
            "type": "APIVersion",
            "major": $major,
            "minor": $minor,
            "micro": $micro
        }
    }
    EOF)

    check_result
}

# Authenticate the DE for the provided user.
function authenticate_de
{
    echo "authenticating delphix engine..."
```

```
    result=$(curl -s -S -X POST -k --data @- http://${engine}/resources/json/delphix/
login \
        -b ~/cookies.txt -c ~/cookies.txt -H "Content-Type: application/json" <<-EOF
    {
        "type": "LoginRequest",
        "username": "${username}",
        "password": "${password}"
    }
    EOF)

    check_result
}

# Get the branch info so the bookmark to fill in dataLayout
function get_branch
{
    echo "retrieveing branch $branchRef to find Source Data Layout..."
    result=$(curl -s -X GET -k http://${engine}/resources/json/delphix/jetstream/
branch/${branchRef} \
    -b ~/cookies.txt -H "Content-Type: application/json")

    check_result

    # Get everything in the result that comes after dataLayout.
    temp=${result#*\"dataLayout\":\"}
    # Get rid of everything after creat
    dataLayout=${temp%%\"*}

    echo "temp" $temp

    echo "dataLayout" $dataLayout
}

function create_bookmark
{
    get_branch

    # If there is not creation time, we need to use JSTimelinePointLatestTimeInput.
    if [ -z $creationTime ]
    then
        pointParams="\"timelinePointParameters\":{
            \"sourceDataLayout\": \"$dataLayout\",
            \"type\":\"JSTimelinePointLatestTimeInput\"}"

    else
        pointParams="\"timelinePointParameters\":{
            \"sourceDataLayout\": \"$dataLayout\",
            \"time\":\"$creationTime\",
            \"branch\":\"$branchRef\",
            \"type\":\"JSTimelinePointTimeInput\"}"
    fi

    # These are the required parameters.
```

```
    paramString="
        \"bookmark\": {
            \"branch\": \"${branchRef}\",
            \"name\": \"${bookmarkName}\","

    # Fill in optional parameters if there are any.
    if [[ -n $description ]]
    then
        paramString="$paramString \"description\": \"$description\","
    fi

    if [[ -n $expiration ]]
    then
        paramString="$paramString \"expiration\": \"$expiration\","
    fi

    if [[ -n $shared ]]
    then
        paramString="$paramString \"shared\": $shared,"
    fi

    if [[ -n $tags ]]
    then
        # Add quotes back to the passed in tags so they are processed correctly.
        tags=${tags//[/[\"}
        tags=${tags//,/\",\"}
        tags=${tags//]/\"]}

        paramString="$paramString \"tags\": $tags,"
    fi

    paramString="$paramString \"type\": \"JSBookmark\"
        },
        ${pointParams},
        \"type\": \"JSBookmarkCreateParameters\""

    result=$(curl -s -X POST -k --data @- http://${engine}/resources/json/delphix/
jetstream/bookmark \
        -b ~/cookies.txt -H "Content-Type: application/json" <<-EOF
    {
        $paramString
    }
    EOF)

    check_result

    echo "confirming job completed successfully..."
    # Get everything in the result that comes after job.
    temp=${result#*\"job\":\"}
    # Get rid of everything after
    jobRef=${temp%%\"*}
```

```
    result=$(curl -s -X GET -k http://${engine}/resources/json/delphix/job/${jobRef}
\
    -b ~/cookies.txt -H "Content-Type: application/json")

    # Get everything in the result that comes after job.
    temp=${result#*\"jobState\":\"}
    # Get rid of everything after
    jobState=${temp%%\"*}

    check_result

    while [ $jobState = "RUNNING" ]
    do
        sleep 1
        result=$(curl -s -X GET -k http://${engine}/resources/json/delphix/job/$
{jobRef} \
        -b ~/cookies.txt -H "Content-Type: application/json")

        # Get everything in the result that comes after job.
        temp=${result#*\"jobState\":\"}
        # Get rid of everything after
        jobState=${temp%%\"*}

        check_result

    done

    if [ $jobState = "COMPLETED" ]
    then
        echo "successfully created bookmark $bookmarkName"
    else
        echo "unable to create bookmark"
        echo result
    fi

}

# Check the result of the curl. If there are problems, inform the user then exit.
function check_result
{
    exitStatus=$?
    if [ $exitStatus -ne 0 ]
    then
        echo "command failed with exit status $exitStatus"
        exit 1
    elif [[ $result != *"OKResult"* ]]
    then
        echo ""
        echo $result
        exit 1
    fi
}
```

```
##### Main

while getopts "u:d:D:e:s:t:T:h" flag; do
    case "$flag" in
        u )             username=${OPTARG%:*}
                        password=${OPTARG##*:}
                        ;;
        d )             engine=$OPTARG
                        ;;
        D )             description=$OPTARG
                        ;;
        e )             expiration=$OPTARG
                        ;;
        s )             shared=true
                        ;;
        t )             creationTime=$OPTARG
                        ;;
        T )             tags=$OPTARG
                        ;;
        h )             usage
                        exit
                        ;;
        * )             usage
                        exit 1
    esac

echo "OPTARG" $OPTARG ####

done


# Shift the parameters so we only have the positional arguments left
shift $((OPTIND-1))

# Check that there are 2 positional arguments
if [ $# != 2 ]
then
    usage
    exit 1
fi

# Get the two positional arguments
bookmarkName=$1
shift
branchRef=$1

create_session
authenticate_de
create_bookmark
```

API cookbook: delete a bookmark in Delphix self-service

This API cookbook recipe describes how to delete a bookmark in Delphix Self-Service (Jet Stream).

> The following script is for educational and demonstration purposes only and is not supported by Delphix.

Deleting a bookmark in Self-Service

```
#!/bin/bash
#
# sample script to delete a bookmark on a Jet Stream container.
#
# Please set the following variables to suit your purposes.
# set this to the FQDN or IP address of the Delphix Engine
DE="ars-dlpx-6010-3.dlpxdc.co"
# set this to the Delphix admin user name
DELPHIX_ADMIN="admin"
# set this to the password for the Delphix admin user
DELPHIX_PASS="delphix"
# reference of bookmark you want to delete
BOOKMARK_REF="JS_BOOKMARK-2"
#
# create our session
curl -s -X POST -k --data @- http://${DE}/resources/json/delphix/session \
    -c ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
    "type": "APISession",
    "version": {
        "type": "APIVersion",
        "major": 1,
        "minor": 6,
        "micro": 2
    }
}
EOF
echo
#
# authenticate to the DE
curl -s -X POST -k --data @- http://${DE}/resources/json/delphix/login \
    -b ~/cookies.txt -c ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
    "type": "LoginRequest",
    "username": "${DELPHIX_ADMIN}",
    "password": "${DELPHIX_PASS}"
}
EOF
echo
#
# delete the bookmark
curl -s -X DELETE -k http://${DE}/resources/json/delphix/jetstream/bookmark/$
{BOOKMARK_REF} \
    -b ~/cookies.txt -H "Content-Type: application/json"
```

```
echo
```

API cookbook: get a bookmark in Delphix self-service

This API cookbook recipe describes how to get a bookmark in Delphix Self-Service (Jet Stream).

> The following script is for educational and demonstration purposes only and is not supported by Delphix.

```
#!/bin/bash
#
# sample script to get a bookmark on a Jet Stream container.
#
# Please set the following variables to suit your purposes.
# set this to the FQDN or IP address of the Delphix Engine
DE="110.110.200.107"
# set this to the Delphix admin user name
DELPHIX_ADMIN="admin"
# set this to the password for the Delphix admin user
DELPHIX_PASS="delphix"
# reference of bookmark you want to get
BOOKMARK_REF="JS_BOOKMARK-2"
#
# create our session
curl -s -X POST -k --data @- http://${DE}/resources/json/delphix/session \
    -c ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
    "type": "APISession",
    "version": {
        "type": "APIVersion",
        "major": 1,
        "minor": 6,
        "micro": 2
    }
}
EOF
echo
#
# authenticate to the DE
curl -s -X POST -k --data @- http://${DE}/resources/json/delphix/login \
    -b ~/cookies.txt -c ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
    "type": "LoginRequest",
    "username": "${DELPHIX_ADMIN}",
    "password": "${DELPHIX_PASS}"
}
EOF
echo
#
# get the bookmark
curl -s -X GET -k http://${DE}/resources/json/delphix/jetstream/bookmark/$
{BOOKMARK_REF} \
    -b ~/cookies.txt -H "Content-Type: application/json"
echo
```

API cookbook: share a bookmark in Delphix self-service

Delphix Self-Service administrators can use this API cookbook recipe to share a bookmark in Delphix Self-Service (Jet Stream) using the Delphix Engine API.

> The following script is for educational and demonstration purposes only and is not supported by Delphix.

This script can be downloaded by selecting **shareBookmark.sh**

```
#!/bin/bash

# A sample script for calls to the API. This one shares Bookmark across containers in
same template.

##### Constants

# Describes a Delphix software revision.
# Please change version are per your Delphix Engine CLI, if different.
VERSION="1.11.10"


##### Default Values. These can be overwriten with optional arguments.
engine="ars-dlpx-6010-3.dlpxdc.co"
username="admin"
password="delphix"

##examples##
# Share Bookmark
#./shareBookmark.sh -a share JS_BOOKMARK-75
# Unshare Bookmark
#./shareBookmark.sh -a unshare JS_BOOKMARK-75

##### Functions

# Help Menu
function usage {
    echo "Usage: shareBookmark.sh [[-h] | options...] <bookmarkName>"
    echo "Share/Unshare JetStream bookmark"
    echo ""
    echo "Positional arguments"
    echo "bookmarkName. Format: JS_BOOKMARK-<n>"
    echo ""
    echo "Optional Arguments:"
    echo "  -h                Show this message and exit"
    echo "  -d                Delphix engine IP address or host name, otherwise
revert to default"
    echo "  -u USER:PASSWORD  Server user and password, otherwise revert to default"
    echo "  -a                action to perform on bookmark. Type:String.
Values:share/unshare"
}

# Create Our Session, including establishing the API version.
```

```
function create_session
{
    # Pulling the version into parts. The {} are necessary for string manipulation.
    # Strip out longest match following "."  This leaves only the major version.
    major=${VERSION%%.*}
    # Strip out the shortest match preceding "." This leaves minor.micro.
    minorMicro=${VERSION#*.}
    # Strip out the shortest match followint "." This leaves the minor version.
    minor=${minorMicro%.*}
    # Strip out the longest match preceding "." This leaves the micro version.
    micro=${VERSION##*.}

    # Quick note about the <<-. If the redirection operator << is followed by a -
(dash), all leading TAB from the document data will be
    # ignored. This is useful to have optical nice code also when using here-
documents. Otherwise you must have the EOF be on a line by itself,
    # no parens, no tabs or anything.

    echo "creating session..."
    result=$(curl -s -S -X POST -k --data @- http://${engine}/resources/json/delphix/
session \
        -c ~/cookies.txt -H "Content-Type: application/json" <<-EOF
    {
        "type": "APISession",
        "version": {
            "type": "APIVersion",
            "major": $major,
            "minor": $minor,
            "micro": $micro
        }
    }
    EOF)

    check_result
}

# Authenticate the DE for the provided user.
function authenticate_de
{
    echo "authenticating delphix engine..."
    result=$(curl -s -S -X POST -k --data @- http://${engine}/resources/json/delphix/
login \
        -b ~/cookies.txt -c ~/cookies.txt -H "Content-Type: application/json" <<-EOF
    {
        "type": "LoginRequest",
        "username": "${username}",
        "password": "${password}"
    }
    EOF)

    check_result
}
```

```
function bookmark_action
{
    # Change share mode of bookmark

    if [[ $action = "share" ]]
    then
        result=$(curl -s -X POST -k --data @- http://${engine}/resources/json/
delphix/jetstream/bookmark/${bookmarkName}/$action \
       -b ~/cookies.txt -H "Content-Type: application/json" <<-EOF
       {}
       EOF)

    check_result

    echo "Bookmark ${bookmarkName} is now in shared mode"

    elif [[ $action = "unshare" ]]
    then
        result=$(curl -s -X POST -k --data @- http://${engine}/resources/json/
delphix/jetstream/bookmark/${bookmarkName}/$action \
       -b ~/cookies.txt -H "Content-Type: application/json" <<-EOF
       {}
       EOF)

    check_result

    echo "Bookmark ${bookmarkName} is now in not-share mode"

    fi

}

# Check the result of the curl. If there are problems, inform the user then exit.
function check_result
{
    exitStatus=$?
    if [ $exitStatus -ne 0 ]
    then
        echo "command failed with exit status $exitStatus"
        exit 1
    elif [[ $result != *"OKResult"* ]]
    then
        echo ""
        echo $result
        exit 1
    fi
}

##### Main

while getopts "u:d:a:h" flag; do
    case "$flag" in
        u )             username=${OPTARG%:*}
```

```
                        password=${OPTARG##*:}
                        ;;
        d )             engine=$OPTARG
                        ;;
        a )             action=$OPTARG
                        ;;
        h )             usage
                        exit
                        ;;
        * )             usage
                        exit 1
    esac

done


# Shift the parameters so we only have the positional arguments left
shift $((OPTIND-1))

# Check that there is 1 positional arguments
if [ $# != 1 ]
then
    usage
    exit 1
fi

# Get the one positional arguments
bookmarkName=$1

create_session
authenticate_de
bookmark_action
```

API cookbook: update a bookmark in Delphix self-service

This API cookbook recipe describes how to update a Bookmark in Delphix Self-Service (Jet Stream). Note that the following example includes updating the "tags" on a Delphix Self-Service bookmark.

The following script is for educational and demonstration purposes only and is not supported by Delphix.

```
#!/bin/bash
#
# sample script to update a bookmark on a Jet Stream container.
#
# Please set the following variables to suit your purposes.
# set this to the FQDN or IP address of the Delphix Engine
DE="ars-dlpx-6010-3.dlpxdc.co"
# set this to the Delphix admin user name
DELPHIX_ADMIN="admin"
# set this to the password for the Delphix admin user
DELPHIX_PASS="delphix"
# reference of bookmark you want to update
BOOKMARK_REF="JS_BOOKMARK-2"
#
# create our session
curl -s -X POST -k --data @- http://${DE}/resources/json/delphix/session \
    -c ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
    "type": "APISession",
    "version": {
        "type": "APIVersion",
        "major": 1,
        "minor": 6,
        "micro": 2
    }
}
EOF
echo
#
# authenticate to the DE
curl -s -X POST -k --data @- http://${DE}/resources/json/delphix/login \
    -b ~/cookies.txt -c ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
    "type": "LoginRequest",
    "username": "${DELPHIX_ADMIN}",
    "password": "${DELPHIX_PASS}"
}
EOF
echo
#
# Update the bookmark. Note that only fields you want to change must be included in
the bookmark
# json.
```

```
curl -s -X POST -k --data @- http://${DE}/resources/json/delphix/jetstream/bookmark/$
{BOOKMARK_REF} \
    -b ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
    "type": "JSBookmark",
    "tags": ["tagA", "tabB"]
}
EOF
echo
```

API cookbook: delete Delphix self-service container

This API cookbook recipe describes how to delete a container in Delphix Self-Service.

> The following script is for educational and demonstration purposes only and is not supported by Delphix.

This script can be downloaded by selecting **deleteContainer.sh**

Delete Delphix Self-Service Container

```
#!/bin/bash
#
# sample script to delete a bookmark on a Jet Stream container.
#
# Please set the following variables to suit your purposes.
# set this to the FQDN or IP address of the Delphix Engine
DE="ars-6010.dlpxdc.co"
# set this to the Delphix admin user name
DELPHIX_ADMIN="admin"
# set this to the password for the Delphix admin user
DELPHIX_PASS="delphix"
# reference of container you want to delete
CONTAINER_REF="JS_DATA_CONTAINER-1"
#

# create our session
curl -s -X POST -k --data @- http://${DE}/resources/json/delphix/session \
    -c ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
    "type": "APISession",
    "version": {
        "type": "APIVersion",
        "major": 1,
        "minor": 6,
        "micro": 2
    }
}
EOF
echo

#
# authenticate to the DE
curl -s -X POST -k --data @- http://${DE}/resources/json/delphix/login \
    -b ~/cookies.txt -c ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
    "type": "LoginRequest",
    "username": "${DELPHIX_ADMIN}",
    "password": "${DELPHIX_PASS}"
}
EOF
echo
```

```
#
# delete the bookmark
curl -s -X DELETE -k http://${DE}/resources/json/delphix/jetstream/container/$
{CONTAINER_REF} \
    -b ~/cookies.txt -H "Content-Type: application/json"
echo
```

API cookbook: delete Delphix self-service template

This API cookbook recipe describes how to delete a template in Delphix Self-Service

> The following script is for educational and demonstration purposes only and is not supported by Delphix

This script can be downloaded by selecting **deleteTemplate.sh**

Delete Delphix Self-Service Template

```
#!/bin/bash
#
# sample script to delete a bookmark on a Jet Stream container.
#
# Please set the following variables to suit your purposes.
# set this to the FQDN or IP address of the Delphix Engine
DE="ars-dlpx-6010.dlpxdc.co"
# set this to the Delphix admin user name
DELPHIX_ADMIN="admin"
# set this to the password for the Delphix admin user
DELPHIX_PASS="delphix"
# reference of template you want to delete
TEMPLATE_REF="JS_DATA_TEMPLATE-1"
#

# create our session
curl -s -X POST -k --data @- http://${DE}/resources/json/delphix/session \
    -c ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
    "type": "APISession",
    "version": {
        "type": "APIVersion",
        "major": 1,
        "minor": 6,
        "micro": 2
    }
}
EOF
echo

#
# authenticate to the DE
curl -s -X POST -k --data @- http://${DE}/resources/json/delphix/login \
    -b ~/cookies.txt -c ~/cookies.txt -H "Content-Type: application/json" <<EOF
{
    "type": "LoginRequest",
    "username": "${DELPHIX_ADMIN}",
    "password": "${DELPHIX_PASS}"
}
EOF
echo
```

```
#
# delete the bookmark
curl -s -X DELETE -k http://${DE}/resources/json/delphix/jetstream/template/$
{TEMPLATE_REF} \
    -b ~/cookies.txt -H "Content-Type: application/json"
echo
```

API cookbook: uploadUpgrade

This API cookbook recipe describes how to use uploadUpgrade.

```
curl -s -X POST -k --data @- http://delphix.engine/resources/json/delphix/session -c
~/cookies.txt -H "Content-Type: application/json" <<EOF
{
"type": "APISession",
"version": {
"type": "APIVersion",
"major": 1,
"minor": 10,
"micro": 0
}
}
EOF

curl -s -X POST -k --data @- httpk --data @- http://delphix.engine/resources/json/
delphix/login -b ~/cookies.txt -c ~/cookies.txt -H "Content-Type: application/json"
<<EOF
{
"type": "LoginRequest",
"username": "sysadmin",
"password": "sysadmin"
}
EOF


curl -s -X POST -F file=@upgrade_image_path http://delphix.engine/resources/json/
system/uploadUpgrade -b ~/cookies.txt
```

Kerberos APIs

API cookbook: ASEDBConfig

This API cookbook recipe describes how to configure your SAP ASE database using the Delphix Engine API.

```
{
    "name":"ASEDBConfig",
    "description":"A SAP ASE Database Config.",
    "abstract": true,
    "extends": {
        "$ref": "/delphix-source-config.json"
    },
    "properties": {
        "databaseName": {
            "type": "string",
            "description": "The name of the database.",
            "create": "required",
            "update": "optional",
            "pattern": "^[a-zA-Z0-9_]+$",
            "maxLength": 30
        },
        "user": {
            "type": "string",
            "description": "The username of the database user.",
            "update": "optional",
            "maxLength": 256
        },
        "credentials": {
            "type": "object",
            "description": "The password of the database user.",
            "$ref": "/delphix-credential.json",
            "update": "optional"
        },
        "repository": {
            "type": "string",
            "description": "The object reference of the source repository.",
            "format": "objectReference",
            "referenceTo": "/delphix-ase-instance.json",
            "create": "required",
            "update": "optional"
        }
    }
}
```

API cookbook: ASEhostEnvironmentParameters

This API cookbook recipe describes how to configure your SAP ASE host environment parameters using the Delphix Engine API.

```
{
    "name": "ASEHostEnvironmentParameters",
    "description": "SAP ASE host environment parameters.",
    "extends": {
        "$ref": "/delphix-typed-object.json"
    },
    "properties": {
        "dbUser": {
            "type": "string",
            "description": "The username of the database user.",
            "create": "optional",
            "update": "optional",
            "maxLength": 256
        },
        "credentials": {
            "type": "object",
            "description": "The credentials of the database user.",
            "$ref": "/delphix-credential.json",
            "create": "required",
            "update": "optional",
            "properties": {
                "type": {
                    "type": "string",
                    "description": "Object type.",
                    "required": true,
                    "format": "type",
                    "default": "PasswordCredential"
                }
            }
        }
    }
}
```

API cookbook: SAP ASE instance

This API cookbook recipe describes how to configure your SAP ASE instance using the Delphix Engine API.

```
{
    "name": "ASEInstance",
    "description": "The SAP ASE source repository.",
    "extends": {
        "$ref": "/delphix-source-repository.json"
    },
    "properties": {
        "instanceName": {
            "type": "string",
            "description": "The name of the SAP ASE instance.",
            "create": "required"
        },
        "installationPath": {
            "type": "string",
            "description": "The SAP ASE instance home.",
            "create": "required",
            "update": "optional"
        },
        "ports": {
            "type": "array",
            "description": "The network ports for connecting to the SAP ASE
instance.",
            "items": {
                type: "integer"
            },
            "create": "required",
            "update": "optional"
        },
        "instanceOwner": {
            "type": "string",
            "description": "The username of the account the SAP ASE instance is
running as.",
            "create": "required",
            "update": "optional"
        },
        "instanceOwnerUid": {
            "type": "integer",
            "description": "The uid of the account the SAP ASE instance is running
as.",
            "create": "readonly",
            "update": "readonly"
        },
        "instanceOwnerGid": {
            "type": "integer",
            "description": "The gid of the account the SAP ASE instance is running
as.",
            "create": "readonly",
            "update": "readonly"
```

```
        },
        "pageSize": {
            "type": "integer",
            "description": "Database page size for the SAP ASE instance."
        },
        "servicePrincipalName": {
            "type": "string",
            "description": "The Kerberos SPN of the database.",
            "create": "optional",
            "update": "optional"
        },
        "dbUser": {
            "type": "string",
            "description": "The username of the database user.",
            "create": "optional",
            "update": "optional",
            "maxLength": 256
        },
        "isqlPath" : {
            "type" : "string",
            "description" : "The path to the isql binary to use for this SAP ASE
instance.",
            "create" : "optional",
            "update" : "optional"
        },
        "credentials": {
            "type": "object",
            "description": "The credentials of the database user.",
            "$ref": "/delphix-credential.json",
            "create": "optional",
            "update": "optional",
            "properties": {
                "type": {
                    "type": "string",
                    "description": "Object type.",
                    "required": true,
                    "format": "type",
                    "default": "PasswordCredential"
                }
            }
        },
        "discovered": {
            "type": "boolean",
            "description": "True if the SAP ASE instance was automatically
discovered."
        }
    }
}
```

API cookbook: ASELinkData

This API cookbook recipe describes how to configure your SAP ASE link data using the Delphix Engine API.

```
{
    "name": "ASELinkData",
    "description": "SAP ASE specific parameters for a link request.",
    "extends": {
        "$ref": "/delphix-link-data.json"
    },
    "properties": {
        "config": {
            "type": "string",
            "description": "Reference to the configuration for the source.",
            "format": "objectReference",
            "referenceTo": "/delphix-ase-db-config.json",
            "required": true
        },
        "externalFilePath": {
            "type": "string",
            "description": "External file path.",
            "maxLength": 1024,
            "create": "optional",
        },
        "operations": {
            "description": "User-specified operation hooks for this source.",
            "type": "object",
            "$ref": "/delphix-linked-source-operations.json",
            "create": "optional"
        },
        "mountBase" : {
            "type" : "string",
            "description" : "The base mount point to use for the NFS mounts.",
            "maxLength" : 87,
            "create" : "optional"
        },
        "loadBackupPath": {
            "type": "string",
            "description": "Source database backup location.",
            "maxLength": 1024,
            "required": true
        },
        "loadLocation": {
            "type": ["object", "null"],
            "description": "Backup location to use for loading backups from the
source.",
            "$ref": "/delphix-ase-backup-location.json",
            "create": "optional"
        },
        "dumpCredentials": {
            "type": ["object", "null"],
            "description": "The credential for the source DB user.",
```

```
            "$ref": "/delphix-password-credential.json",
            "create": "optional"
        },
        "sourceHostUser": {
            "type": "string",
            "description": "Information about the host OS user on the source to use
for linking.",
            "format": "objectReference",
            "referenceTo": "/delphix-source-environment-user.json",
            "required": true
        },
        "dbUser": {
            "type": "string",
            "description": "The user name for the source DB user.",
            "create": "optional"
        },
        "dbCredentials": {
            "type": "object",
            "description": "The credentials of the database user.",
            "$ref": "/delphix-credential.json",
            "required": true,
            "properties": {
                "type": {
                    "type": "string",
                    "description": "Object type.",
                    "required": true,
                    "format": "type",
                    "default": "PasswordCredential"
                }
            }
        },
        "stagingRepository": {
            "type": "string",
            "description": "The SAP ASE instance on the staging environment that we
want to use for validated sync.",
            "format": "objectReference",
            "referenceTo": "/delphix-ase-instance.json",
            "required": true
        },
       "stagingHostUser": {
            "type": "string",
            "description": "Information about the host OS user on the staging
environment to use for linking.",
            "format": "objectReference",
            "referenceTo": "/delphix-source-environment-user.json",
            "required": true
        },
        "stagingPreScript": {
            "type": "string",
            "description": "A user-provided shell script or executable to run prior
to restoring from a backup during validated sync.",
            "maxLength": 1024,
            "create": "optional"
```

```
        },
        "stagingPostScript": {
             "type": "string",
             "description": "A user-provided shell script or executable to run after
restoring from a backup during validated sync.",
             "maxLength": 1024,
             "create": "optional"
        },
        "syncParameters": {
            "type": "object",
            "description": "Sync parameters for the container.",
            "$ref": "/delphix-ase-sync-parameters.json",
            "required": true,
            "properties": {
                "type": {
                    "type": "string",
                    "description": "Object type.",
                    "required": true,
                    "format": "type",
                    "default": "ASELatestBackupSyncParameters"
                }
            }
        },
        "validatedSyncMode": {
            "type": "string",
            "description": "Specifies the validated sync mode to synchronize the
dSource with the source database.",
            "enum": ["ENABLED", "DISABLED"],
            "default": "ENABLED",
            "create": "optional"
        }
    }
}
```

API cookbook: EnvironmentUser

This API cookbook recipe describes how to configure your environment user using the Delphix Engine API.


```
{
    "root": "/resources/json/delphix/environment/user",
    "name": "EnvironmentUser",
    "description": "The representation of an environment user object.",
    "extends": {
        "$ref": "/delphix-user-object.json"
    },
    "nameParent": "environment",
    "properties": {
        "credential": {
            "type": "object",
            "$ref": "/delphix-credential.json",
            "description": "The credential for the environment user.",
            "create": "required",
            "update": "optional",
            "properties": {
                "type": {
                    "type": "string",
                    "description": "Object type.",
                    "required": true,
                    "format": "type",
                    "default": "PasswordCredential"
                }
            }
        },
        "environment": {
            "type": "string",
            "description": "A reference to the associated environment.",
            "format": "objectReference",
            "referenceTo": "/delphix-source-environment.json",
            "create": "optional"
        },
        "groupId": {
            "type": "integer",
            "description": "Group ID of the user.",
            "create": "optional",
            "update": "optional",
            "minimum": 0,
            "maximum": 4294967295
        },
        "userId": {
            "type": "integer",
            "description": "User ID of the user.",
            "create": "optional",
            "update": "optional",
            "minimum": 0,
            "maximum": 4294967295
        }
```

```
    },
    "create": {
        "description": "Create a new EnvironmentUser object.",
        "payload" : {
            "type": "object",
            "$ref": "/delphix-source-environment-user.json"
         },
         "return": {
             "type": "string",
             "format": "objectReference",
             "referenceTo": "/delphix-source-environment-user.json"
         }
    },
    "read": {
        "description": "Retrieve the specified EnvironmentUser object.",
        "return": {
            "type": "object",
            "$ref": "/delphix-source-environment-user.json"
        }
    },
    "update": {
        "description": "Update the specified EnvironmentUser object.",
        "payload": {
            "type": "object",
            "$ref": "/delphix-source-environment-user.json"
         }
    },
    "delete": {
        "payload": {
            "type": "object",
            "$ref": "/delphix-delete-parameters.json",
            "required": false
        },
        "description" : "Delete the specified EnvironmentUser object."
    },
    "list": {
        "description": "Returns the list of all environment users in the system.",
        "parameters": {
            "environment": {
                "type": "string",
                "description": "Limit results to users within the given environment.",
                "format": "objectReference",
                "referenceTo": "/delphix-source-environment.json",
                "mapsTo": "environment"
            }
        }
        "return": {
            "type": "array",
            "items": {
              "type": "object",
              "$ref": "/delphix-source-environment-user.json"
            }
        }
```

```
    }
}
```

API cookbook: KerberosConfig

This API cookbook recipe describes how to configure Kerberos using the Delphix Engine API.

```
{
    name: "KerberosConfig",
    description: "Kerberos Client Configuration.",
    root: "/resources/json/delphix/service/kerberos",
    singleton: true,
    cliVisibility: [ "DOMAIN", "SYSTEM" ],
    extends: {
        $ref: "/delphix-user-object.json"
    },
    properties: {
        realm: {
            description: "Kerberos Realm name.",
            type: "string",
            create: "required",
            update: "optional"
        },
        kdcs: {
            description: "One of more KDC servers.",
            type: "array",
            create: "required",
            update: "optional",
            minItems: 1,
            items: {
                type: "object",
```

```
            $ref: "/delphix-kerberos-kdc.json"
        }
    },
    keytab: {
        description: "Kerberos keytab file data in base64 encoding.",
        type: "string",
        format: "password",
        create: "required",
        update: "optional"
    },
    principal: {
        description: "Kerberos principal name.",
        type: "string",
        create: "required",
        update: "optional"
    },
    enabled: {
        description: "Indicates whether kerberos has been configured or not.",
        type: "boolean"
    }
},
read: {
    description: "Retrieve the specified KerberosConfig object.",

    return: {

        type: "object",

        $ref: "/delphix-kerberos-config.json"
```

```
        }
    },
    update: {
        description: "Update the specified KerberosConfig object.",

        payload: {

            type: "object",

            $ref: "/delphix-kerberos-config.json"

        }

    },
    rootOperations: {
        reset: {
            description: "Reset kerberos configuration and disable the feature.",
            payload: {}
        }
    }
}
```

API cookbook: KerberosCredential

This API cookbook recipe describes how to configure Kerberos credentials using the Delphix Engine API.

```
{

   "name": "KerberosCredential",

   "description": "Kerberos based security credential.",

   "extends": {

       "$ref": "/delphix-credential.json"

   },

   "properties": {

   }

}
```

API cookbook: KerberosKDC

This API cookbook recipe describes how to configure KerberosKDC using the Delphix Engine API.

```
{
    name: "KerberosKDC",
    description: "Kerberos Client Configuration.",
    extends: {
        $ref: "/delphix-typed-object.json"
    },
    properties: {
        hostname: {
            description: "KDC Server hostname.",
            type: "string",
            format: "host",
            create: "required",
            update: "optional"
        },
        port: {
            description: "KDC Server port number.",
            type: "integer",
            create: "required",
            update: "optional",
            minimum: 0,
            maximum: 65535,
            default: 88
        }
    }
}
```