Configuring Replication

This topic describes how to configure data replication between Delphix Engines.

- Prerequisites
- Configuring the Network
- Configuring the Replication Source Delphix Engine
- Configuring the Target Delphix Engine
- Related Links

Prerequisites

- The replication source and the replication target must be running identical versions of the Delphix Engine (for example, Delphix Engine version 5.0).
- 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.

See also: Replication Prerequisites

Configuring the Network

Replication operates using a private network protocol between two Delphix Engines. Apart from standard network considerations for performance, no additional configuration is required for replication. Replication can run over dedicated networks by configuring routing to direct traffic destined for the target IP address over a specific interface. The replication process can recover from transient network outages, but extended outages may cause the process to start from the previous update.

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:

- Enable port 8415 on the firewall in order to allow connections to this port from the source to the target.
- 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:

```
$ dlpx-engine> service proxy
$ dlpx-engine service proxy> update
dlpx-engine service proxy> set socks.enabled=true
dlpx-engine service proxy> set socks.host=10.2.3.4
$ dlpx-engine service proxy> set socks.username=someuser
$ dlpx-engine service proxy> set socks.password=somepassword
$ dlpx-engine service proxy> commit
$ dlpx-engine service proxy> get

ProxyService
 bred: ProxyConfiguration
 enabled: false
 host: (unset)
 password: (unset)
 port: 8080
 username: (unset)
 socks:
 breed: ProxyConfiguration
 enabled: true
 host: 10.2.3.4
 password: ********
 port: 1080
 username: someuser
```

Note that SOCKS port 1080 is used by default, but can be overridden.
Configuring the Replication 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. For Target Engine, enter the hostname or IP address for the target Delphix Engine.
6. 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.
7. By default, automatic replication is disabled, meaning that you must trigger replication updates manually. To enable automatic replication, click the Enabled checkbox.
8. 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.

   Encrypting Traffic
   Automatic replication uses Quartz for scheduling. Starting with Delphix version 4.2, the Quartz-formatted string is editable via the Advanced option.

9. Under Traffic Options, select whether you want to Encrypt traffic or Limit bandwidth during replication updates.

   Encrypting Traffic
   By default, replication streams are sent unencrypted. This provides maximum performance on a secure network. If the network is insecure, you can enable encryption. Note that encrypting the replication stream will consume additional CPU resources and may limit the maximum bandwidth that can be achieved.

   Compressing Traffic
   All replication streams are compressed via the same mechanism that compresses the data on disk. In environments where network bandwidth is a constrained resource, compression has been shown to conserve bandwidth and optimize overall throughput achieved by replication. Because the data is already stored compressed, there is no CPU overhead for compressing replication streams.

   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 maximum bandwidth that replication can consume.

10. In the right-hand column, under Objects Being Replicated, click the checkboxes next to the objects you want to replicate.
Selected Objects

- Some selected objects may have dependencies – other objects that will be pulled into replication because they share data. For more details, see Replication Overview. Objects that will be replicated are confirmed with a blue chain link icon.

  - Note that this is not guaranteed to be the full set of dependent objects, but rather is a best guess. The full set of objects and their dependants will be calculated at the time of replication.

- When replicating a group, all dSources and VDBs currently in the group, or added to the group at a later time, will be included.

- If you select a Jet Stream 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.

- If you select the entire server, all groups and Jet Stream objects will be included.

- 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.

- 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.

11. 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.

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. See the topics under CLI Cookbook: Replication for more information.

Enabling Configuration of Multiple Replication Profiles

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. Replicated objects will appear in an alternate replica that mirrors the original object layout. These replicas can be viewed through the System > Replication screen under the Received Replicas section (or namespace in the CLI). 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 Failing Over a Replica.

Objects can be created and managed 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 flexible geographical distribution of data. For disaster recovery this is not a recommended practice, as 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.

Related Links

- CLI Cookbook: Replication
- Replication Overview
- Replication User Interface
- Replicas and Failover
- Failing Over a Replica