

Installation 1

# DLM Installation and Upgrade

**Date of Publish:** 2019-07-12



<https://docs.hortonworks.com>

# Contents

<b>DLM support requirements.....</b>	<b>4</b>
<b>Installation overview.....</b>	<b>5</b>
<b>Prerequisites.....</b>	<b>6</b>
<b>Version recommendation for DLM and HDP.....</b>	<b>8</b>
<b>Installing DataPlane.....</b>	<b>8</b>
<b>Setting up DLM packages.....</b>	<b>8</b>
Download the packages.....	9
Setting up the local repository.....	9
Set up DLM packages in local repository.....	9
Create the repository configuration file.....	10
<b>Installing the DLM Engine.....</b>	<b>11</b>
Install the DLM Engine MPack.....	11
Set up DLM Engine database.....	12
Configure MySQL external database.....	12
Configure Postgres external database.....	12
Create the DLM Engine service user.....	13
Set up the DLM Engine service in Ambari.....	13
Install DLM Engine service in Ambari.....	14
Configure Knox SSO.....	19
Configure SSL (Optional).....	19
Configure Knox Gateway (Optional).....	23
Complete the installation.....	25
Trusted proxy for DLM.....	25
Configure Kerberos security (Optional).....	27
Enable Ranger Deny policy.....	28
Restart services.....	28
Verify the DLM Engine installation.....	29
About requested events missing in Notification Log table.....	35
<b>Install the DLM Service.....</b>	<b>36</b>
<b>Verify DLM.....</b>	<b>37</b>
<b>Advanced configurations (Optional).....</b>	<b>38</b>

Using TDE with DLM.....	38
Configure TDE for HDFS replication.....	38
Configure TDE for Hive replication.....	39
On-premise replication using different keys and Ranger-KMS instances.....	39
Preparing HDP cluster for Hive cloud replication.....	40
<b>Security considerations.....</b>	<b>40</b>
<b>Upgrading DLM.....</b>	<b>41</b>
Upgrade DataPlane platform.....	44
Upgrade DLM App.....	44
Upgrade DLM Engine.....	45
<b>Migrating DLM Engine from one host to another.....</b>	<b>45</b>
<b>Troubleshooting DLM.....</b>	<b>45</b>
DataPlane platform.....	46
DLM Engine.....	46
Common errors.....	46

## DLM support requirements

Prior to installing Data Lifecycle Manager (DLM), you must consider various aspects of your HDP environment and prepare your clusters prior to DLM installation. The host on which you install DLM is the same host on which you install DataPlane Platform.

### Support Matrix information

You can find the most current information about interoperability for this release on the Support Matrix. The Support Matrix tool provides information about:

- Operating Systems
- Databases
- Browsers
- JDKs

To access the tool, go to: <https://supportmatrix.hortonworks.com>.

### DLM Host requirements

The DLM application is installed on the same host as DP Platform and has no requirements beyond what is required by DP Platform. See the *DP Platform Support Requirements* for details.

### Requirements for clusters used with DLM Engine

- The clusters on which you install the DLM Engine must meet the requirements identified in the following sections. After the DLM Engine is installed and properly configured on a cluster, the cluster can be registered with DP and used for DLM replication.



#### Important:

Clusters used as source and destination in a DLM replication relationship must have exactly the same configurations for LDAP, Kerberos, Ranger, Knox, and HA.

- DLM Engine can be installed on any server class machine starting with 1 GB memory. You may need to increase the memory to about 2 GB or 3 GB, depending on HDFS replication dataset. For example, number of files in the dataset and number of concurrent replication policies.

See the [Support Matrix](#) for supported operating systems and databases.

### Port and Network requirements for clusters

Have the following ports available and open on each cluster:

Default Port Number	Purpose	Comments	Required to be open?
25968	Port for DLM Engine (Beacon) service on hosts	Accessibility is required from all clusters. "Beacon" is the internal name for the DLM Engine. You will see the name Beacon in some paths, commands, etc.	Yes
8020	NameNode host		Yes
50010	All DataNode hosts		Yes
8080	Ambari server host		Yes
10000	HiveServer2 host	Binary mode port (Thrift)	Yes
10001	HiveServer2 host	HTTP mode port	Yes
9083	Hive metastore		Yes

Default Port Number	Purpose	Comments	Required to be open?
2181	ZooKeeper hosts		Yes
6080	Ranger port		Yes
8050	YARN port		Yes
21000	Atlas port		Yes
9292	Ranger KMS		Yes
6182	Secured Ranger		Yes

### HDP component requirements for DLM

The following additional Apache components might be required on your clusters for DLM support, depending on the security configuration and type of replication being performed:

Component	Purpose	Comments
HDFS	For replicating HDFS data.	
Knox	Authentication federation from DPS	Knox must be enabled on clusters before you can register the clusters with DPS.
Ranger	Authorization on clusters during replication	Ranger is optional for HDFS replication, but required for Hive replication.
YARN		
Hive	For replicating Hive database content	Updates via Hive 1 (Based on Apache Hive 1.2.x) and HiveServerInteractive (Based on Apache Hive 2.1.x) are replicated. However, HiveServer2 from Hive 1 is always used for running the replication tasks.
HiveServer 2	Needed for Hive replication	
Hive Metastore	Needed for Hive replication	
Zookeeper	Needed for Hive	
Atlas	Needed for metadata replication	

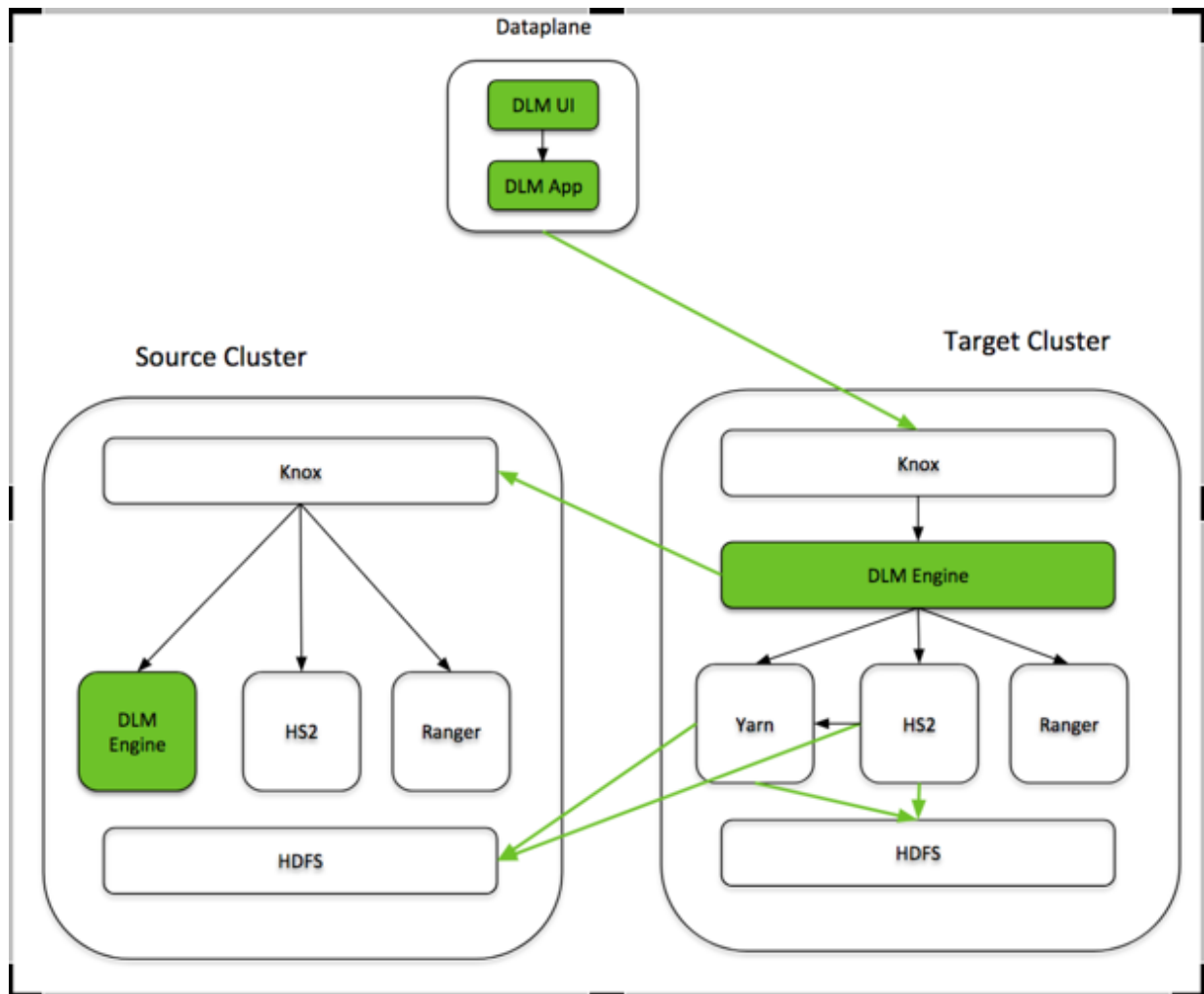
## Installation overview

### About this task

To install Data Life Cycle Manager (DLM), you must install the following components:

### Procedure

1. Install DataPlane Platform (DP): DataPlane Platform hosts services to provide a single pane of glass for managing data across multiple HDP clusters. DataPlane Platform should be installed on a host outside the HDP clusters.
2. Install DLM Service: DLM Service is a DataPlane service which enables DLM functionality. DLM Service hosts DLM UI and sets up DLM functionality across multiple HDP clusters. DLM Service should be installed on the host where DataPlane Platform is installed.
3. Install DLM Engine: DLM engine is the agent in each HDP cluster, which facilitates DLM functionality. DLM engine is installed as an Ambari service in each HDP cluster involved in replication.



## Prerequisites

Before you begin the installation process, verify the following:

- You must have root access to the nodes on which the DLM App and DLM Engine will be installed.
- Ensure required services Knox, Ranger, HDFS, YARN, and Hive are installed.
- Before you install DLM, make sure to verify if you are able to copy files between the Hadoop clusters/endpoints. Depending on various factors, your cluster environment might vary. It is recommended to use distributed copy

command `distcp` to verify if the data between the clusters can be copied successfully. For more information, see [Using DistCp](#).

- Ensure to have one of the following external databases installed: MySQL or Postgres.

See the Hortonworks Support Matrix for the compatible versions of DataPlane (DP) Platform, HDP, and DLM.

- Knox SSO

DP Platform and the DLM leverage Knox SSO to provide users and services with simplified and consistent access to clusters, data and other services. You must configure Knox SSO on the HDP clusters that you plan to use with DLM.



**Note:**

The Knox SSO of your cluster must be configured to use the same LDAP/AD as your DP instance for user identity to match and propagate between the systems.

Refer to the following documentation on how to configure your cluster for Knox SSO:

Resource	Documentation
Install Knox and enable in Ambari	HDP Security Guide, <a href="#">Install Knox</a>
Configure SSO topology	HDP Security Guide, <a href="#">Identity Providers</a>
Configure Knox SSO for Ambari	HDP Security Guide, <a href="#">Setting up Knox SSO for Ambari</a>
Configure LDAP with Ambari	Ambari Security Guide, <a href="#">Configuring Ambari Authentication with LDAP or Active Directory Authentication</a>

- Perform the DataPlane Platform pre-installation tasks. For more information, see [Prepare your clusters](#).
- Install or upgrade to the supported version of Ambari. See Support Matrix for details of the supported Ambari versions. See Apache Ambari installation for more details.
- Install or upgrade to the supported versions of HDP on your cluster using Ambari. See DLM Support Matrix for details of the supported HDP versions. See the HDP installation documentation for more details.
- Ranger

Ranger enables you to create services for specific Hadoop resources (HDFS, HBase, Hive) and add access policies to those services. If you use Ranger for authorisation in your cluster for LDAP users:

- Configure LDAP for Ranger usersync. For more information, see [Advanced Usersync Settings](#).
- Configure LDAP Hadoop group mapping. For more information, see [Setting Up Hadoop Group Mapping](#).
- Knox Gateway

Configuring Knox Gateway is required if your cluster is configured with Kerberos or with wire encryption. This simplifies certificate management for DP and cross-cluster communication, as the only security certificate that needs to be managed is for Knox.

Refer to the following documentation on how to configure your cluster for Knox Gateway:

Resource	Documentation
Configure a reverse proxy with Knox	HDP Security Guide, <a href="#">Configuring the Knox Gateway</a>
Configure LDAP with Knox for proxy authentication	HDP Security Guide, <a href="#">Setting Up LDAP Authentication</a>

- Hive

You must configure Hive with Ranger authoriser. For more information, see

[Authorization using Apache Ranger Policies](#) and `hive.server2.enable.doAs=false`

- YARN

DLM runs the replication jobs using YARN. For on-premise to on-premise replication, the replication job runs on the target cluster. For on-premise to cloud replication, the replication job runs on the source cluster. Make sure YARN is installed on the cluster where the replication job runs.

- Ensure HDP clusters that are involved in replication have symmetric configuration. Each cluster in a replication relationship must be configured exactly the same for security (Kerberos), user management (LDAP/AD), and

Knox Proxy. Cluster services like HDFS, HIVE, Knox, Ranger, and Atlas can have different configurations for High Availability (HA) i.e., source and target clusters have HA and non-HA setup respectively.

- See the Hortonworks Support Matrix for the compatible versions of DP, HDP, and DLM.

#### Related Information

[Hortonworks Support Matrix](#)

## Version recommendation for DLM and HDP

You must ensure that you are using versions of DLM App, DLM Engine, HDP, and Ambari that are supported together.

It is strongly recommended that you install the latest version of DLM, to take advantage of the latest functionality and the stability updates.

DataPlane platform and DLM App should be the latest version. DLM capabilities supported here are the features of the minimum DLM Engine version.

DLM Engine should be on the latest version available for the HDP version that you have installed in your environment. You can upgrade HDP to the version that the latest DLM Engine version requires to perform in a seamless manner.

**DLM 1.5.0** version is compatible with HDP 2.6.5.1175 and HDP 3.1.0.31 releases. **HDP 2.6.5.1175** release has fixes in different components of HDP like Hive, Ranger, and Atlas to enable the DLM functionality. If you are performing replication using Google Cloud Storage and if you are upgrading from less than HDP 2.6.5.1100 release, you must perform a full cluster upgrade to HDP 2.6.5.1175 release.



**Attention:** If you are already on HDP 2.6.5.1100 release, you need not perform a full cluster upgrade.

If you are not using Google Cloud Storage, use [Ambari patch upgrade](#) process, which will upgrade just the required components of HDP stack, so that other HDP components are not affected.

If you are on an earlier release of HDP, use a standard full upgrade to HDP 2.6.5.1175 release. Please reach out to Hortonworks support team to get HDP 2.6.5.1175.

## Installing DataPlane

You must install DataPlane before proceeding with the installation of the DLM App and the DLM Engine. For more information on how to install DataPlane Service (DP), see the DataPlane installation guide.

#### Related Information

[DataPlane installation guide](#)

## Setting up DLM packages

Follow these steps to setup the DLM package:

- Download the packages
- Setup the local repository
- Setup DLM packages in the local repository.
- Create the repository configuration file.



## Download the packages

To install Hortonworks Data Lifecycle Manager (DLM), download the tarballs from the customer portal by following the instructions provided as part of the product procurement process. The list of packages required are:

### Procedure

- beacon-ambari-mpack-X.X.X.X-XX.tar.gz
- DLM-x.X.X.X-XX-<OS>-rpm.tar.gz
- DLM-APP-x.x.x.0-xx-<OS>-rpm.tar.gz

## Setting up the local repository

Hortonworks does not host any public repository for DLM, hence a local repository for the packages is required. As part of the DataPlane setup, you must have set up this local repository already and the same can be used for DLM packages. If not, you can set up a webserver using Apache httpd or Tomcat.

For more information on how to set up the local repository, see the DataPlane Installation Guide.

### Related Information

[DataPlane installation](#)

## Set up DLM packages in local repository

Follow these steps to set up the DLM packages in the local repository.

### Procedure

1. Copy the repository tarball for the DLM Instance to the web server directory and expand (uncompress) the archive file:

- a) Navigate to the web server directory you previously created.

```
cd /var/www/html/
```

All content in this directory is served by the web server.

- b) Move the following tarballs to the current directory and expand the tarballs - DLM-x.X.X.X-XX-<OSplatform>-rpm.tar.gz and DLM-APP-x.x.x.0-xx-centos7-rpm.tar.gz.

Replace <filename> with the actual name of the RPM tarball that you are expanding.

```
tar zxvf <file-name>.tar.gz
```

During expansion of the tarball, subdirectories are created in /var/www/html/, such as DLM/centos7. These directories contain the repositories.

Expanding the tarball might take several seconds.

2. Confirm that you can browse to the newly created local repository by using the *Base URL*:

```
http://<your_webserver>:port/<repo_name>/<OS>/<version>
```

- <your\_webserver>:port

This is the FQDN and port of the web server host.

- <repo\_name>

The repository name of the DLM or DLM-App.

- <OS>  
The operating system, which is centos6 or centos7.
- <version>  
The version number of the downloaded component.

Base URL example for DLM Engine:

```
http://<your_webserver>:port/DLM/centos7/1.1.1.0
```

Base URL example for DLM Service:

```
http://<your_webserver>:port/DLM-App/centos7/1.1.0.0
```

Remember this Base URL. You need it to set up the repository configuration file in subsequent steps.

3. If you have multiple repositories configured in your environment, deploy the following plugin on all the nodes in your cluster.

```
yum install yum-plugin-priorities
```

4. Edit the /etc/yum/pluginconf.d/priorities.conf file to add the following values:

```
[main]
enabled=1
gpgcheck=0
```

## Results

The local repository is now set up and ready for use.

## What to do next

Create the configuration file for the newly created local repository.

## Create the repository configuration file

A repository configuration file (“repo file”) must be created for the DLM Service on the DPS host. The file is required for tasks such as identifying the path to the repository data and establishing whether a GPG signature check should be performed on the repository packages. Only one repository configuration file is needed.

### Procedure

1. Navigate to the repository directory.

```
cd /etc/yum.repos.d/
```

2. Create a repository file.

```
vi dlm-app.repo
```

Alternatively, you can copy an existing repository file to edit.

3. Add the following content to the repository file:



**Important:** Be sure to use the Base URL you created when setting up the local repository.

```
#VERSION_NUMBER=1.1.1.0
```

```
[DLM-1.1.1.0]
name=DLM Version - DLM-1.1.1.0
baseurl=http://<your_webserver>:port/DLM/centos7/1.1.1.0
gpgcheck=1
gpgkey=http://<your_webserver>:port/DLM/centos7/1.1.1.0/RPM-GPG-KEY/RPM-
GPG-KEY-Jenkins
enabled=1
priority=1
```

### What to do next

You are now ready to install the DLM Engine.

## Installing the DLM Engine

DLM requires that an engine be installed on each cluster that is to be used in replication jobs. Complete these steps in every HDP cluster that is involved in replication. “Beacon” is the internal name for the DLM Engine and can be used interchangeably in this document and in the package structure and installation.

Follow these steps to install the DLM Engine:

- Install the DLM Engine MPack.
- Set up the DLM Engine database.
- Create the DLM Engine service user.
- Set up the DLM Engine service in Ambari.

### Install the DLM Engine MPack

DLM requires that an engine be installed on each cluster that is to be used in replication jobs. The engine is installed on the Apache Ambari host, using an Ambari management pack (MPack). An MPack bundles service definitions, stack definitions, and stack add-on service definitions.

#### About this task

- This task must be completed on all clusters to be used with DLM.
- “Beacon” is the internal name for the DLM Engine. If you install DLM, you will see the name Beacon in some paths, commands, etc.

#### Before you begin

You must have root access to the Ambari Server host node to perform this task.



**Important:** Prior to starting installation, you must have downloaded the required repository tarballs from the Hortonworks customer portal, following the instructions provided as part of the product procurement process.

#### Procedure

1. Log in as root to an Ambari host on a cluster.

```
ssh root@<ambari-ip-address>
```

2. Install the DLM Engine MPack by running the following command, replacing <mpack-file-name> with the name of the MPack.

```
ambari-server install-mpack --mpack beacon-ambari-mpack-X.X.X.X-XX.tar.gz --verbose
```

3. Restart the Ambari server.

```
ambari-server restart
```

### What to do next

Set up beacon database.

### Related Tasks

[Create the DLM Engine service user](#)

## Set up DLM Engine database

DLM Engine stores its state in the external relational database. You must configure an external database and add the DLM Engine database user to the database.

Derby database is provided as an easy to use option in test or POC environments. It is strongly recommended to NOT use Derby for production usage. You must use MySQL or PostgreSQL. Make sure to configure backup for this database, so that the DLM state is not lost in case of database host failures.

### Configure MySQL external database

Follow these steps to configure MySQL external database:

#### Procedure

1. Create the database and user for DLM engine. Replace the username, password, and the dbname.

```
CREATE USER beacon IDENTIFIED BY 'beacon';
CREATE DATABASE beaondb;
GRANT ALL PRIVILEGES ON beaondb.* TO 'beacon'@'%' IDENTIFIED BY
'password';
```

2. Setup the mysql connector in Ambari.

If Ambari is not already setup with mysql connector, follow these steps:

- a) Download mysql connector.
- b) Setup the connector in Ambari.

```
ambari-server setup --jdbc-db=mysql --jdbc-driver=/path/to/mysql/mysql-connector-java.jar
```

### Configure Postgres external database

Follow these steps to configure Postgres external database:

#### Procedure

1. Create the database and user for DLM engine. Replace the username, password, and the dbname.

```
CREATE USER beacon WITH PASSWORD 'beacon';
```

```
CREATE DATABASE beaondb;
GRANT ALL PRIVILEGES ON DATABASE beaondb TO beacon;
```

2. Configure the postgresql, so that the DLM engine can connect to this DB instance.

Verify the DB access by running this command on the DLM Engine host: `psql -h <postgres_server_host> -U beacon -d beaondb`

If you cannot connect to the postgres server from DLM Engine host, refer to the [Postgres documentation](#) on setting up the `pg_hba.conf`.

For example:

Add the following entries to `/var/lib/pgsql/data/pg_hba.conf` file:

```
local beaondb          beacon trust
host dlmenginehost    beacon 0.0.0.0/0 trust
host dlmenginehost    beacon ::/0 trust
```

Restart the PostgreSQL service: `service postgresql restart`

Verify client access: `psql -h <POSTGRES_SERVER_HOSTNAME> -U beacon -d beaondb`

## Create the DLM Engine service user

Follow these steps to configure DLM Engine service user:

### Procedure

1. You must configure user. Grant privileges to this user to enable replication of data, metadata, and Ranger policies.
2. If your principal user database is LDAP/AD, create 'DLM Engine service' user in your LDAP/AD setup.
3. Set up the 'DLM Engine service' user as HDFS superuser so that DLM can access HDFS files for replication.

If the hadoop group mapping is set to LDAP,

(`hadoop.security.group.mapping=org.apache.hadoop.security.LdapGroupsMapping`), 'DLM Engine service' user should belong to the HDFS superusergroup (value of `dfs.permissions.superusergroup`).

- a) You can assign HDFS superusergroup to 'DLM Engine service' user in LDAP. or
- b) This can also be set up with static hadoop group mapping (config `hadoop.user.group.static.mapping.overrides=DLM Engine service=<HDFS superusergroup>`).
- c) Refresh the hadoop group mapping.

```
hdfs dfsadmin -refreshSuperUserGroupsConfiguration
hdfs dfsadmin -refreshUserToGroupsMappings
```

4. Verify that 'DLM Engine service' was added as a user to the HDFS superuser group.

`hdfs groups <DLM Engine service user>`

The output should display HDFS or the value of `dfs.permissions.superusergroup` config as one of the groups.

5. The 'DLM Engine service' user requires some set up in Ranger. If the Ranger usersync is set to LDAP/AD, ensure that 'DLM Engine service' user is created in your LDAP/AD setup. Privileges for this user in Ranger will be automatically set up as part of DLM Engine service start.

## Set up the DLM Engine service in Ambari

Follow these steps to set up the DLM Engine service in Ambari.

1. Install the DLM Engine Service in Ambari
2. Configure Knox SSO

3. Configure SSL (Optional)
4. Configure the Knox Gateway (Optional)
5. Complete the installation
6. Trusted Proxy for DLM
7. Configure Kerberos security (Optional)
8. Enable the Ranger Deny policy
9. Restart the services
10. Verify the DLM Engine installation

## Install DLM Engine service in Ambari

Follow these steps to install the DLM Engine in Ambari:

### Procedure

1. Launch Ambari in a browser and log in.

```
http://<ambari-server-host>:8080
Default credentials are:
Username: admin
Password: admin
```

2. Click **Admin > Manage Ambari**.
3. Click **Versions**, and then do the following on the Versions page:

- Click the HDP version in the Name column.
- Change the **Base URL** path for the DLM service to point to the local repository, for example:

```
http://<your_webserver>/DLM/<OS>/<DLM version>
```

4. Click the Ambari logo to return to the main Ambari page.
5. In the Ambari Services navigation pane, click **Actions > Add Service**.

The Add Service Wizard displays.

6. On the **Choose Services** page of the Wizard, select the 'DLM-Engine' service to install in Ambari, and then follow the on-screen instructions.

Other required services are automatically selected.

7. When prompted to confirm addition of dependent services, give a positive confirmation to all.

This adds other required services.

8. On the **Assign Masters** page, you can choose the default settings.
9. On the **Customize Services** page, fill out the required DLM Engine configurations:

- Configure DB

Enter database name as beacon, database user, database password and database host in jdbc url from the database values setup previously.



**Note:** If you have migrated to HDP 3.x version and while adding DLM Engine service, you must enter the **DLM Engine Database** value and also verify the specific **JDBC Driver Class** in the Ambari UI.

The supported DLM Engine database values are:

- EXISTING MYSQL / MARIADB DATABASE
- NEW DERBY DATABASE
- EXISTING POSTGRESQL DATABASE



**Note:** New Derby database option can be used only while setting up the test environment and it must NOT be used for production setup or usage.

If you are upgrading from a previous version of DLM Engine, make sure that your DLM Engine database settings are the same.

### Add Service Wizard

- ADD SERVICE WIZARD
- Choose Services
- Assign Masters
- Assign Slaves and Clients
- Customize Services**
- Configure Identities
- Review
- Install, Start and Test
- Summary

## Customize Services

We have come up with recommended configurations for the services you selected. Customize them as you see fit.

- HDFS
- YARN
- MapReduce2
- Tez
- Hive
- HBase
- Pig
- ZooKeeper
- Ambari Infra
- Ambari Metrics
- Atlas
- Kafka
- Knox
- Ranger
- Ranger KMS
- DLM Engine 1**
- Slider
- Misc

There are 4 configuration changes in 3 services [Show Details](#)

Group:  [Manage Config Groups](#)

Settings **1** [Advanced](#)

### Database

Beacon Database

▾

Database Name

Database Username

JDBC Driver Class

**▲ Database Password**



- Click the Misc tab to modify or change the default Beacon service user or group name, .

### Add Service Wizard x

**ADD SERVICE WIZARD**

- [Choose Services](#)
- [Assign Masters](#)
- [Assign Slaves and Clients](#)
- Customize Services**
- [Configure Identities](#)
- [Review](#)
- [Install, Start and Test](#)
- [Summary](#)

## Customize Services

We have come up with recommended configurations for the services you selected. Customize them as you see fit.

[HDFS](#) [YARN](#) [MapReduce2](#) [Tez](#) [Hive](#) [HBase](#) [Pig](#) [ZooKeeper](#) [Ambari Infra](#) [Ambari Metrics](#) [Atlas](#) [Kafka](#) [Knox](#) [Ranger](#) [Ranger KMS](#) **DLM Engine** [Slider](#) [Misc](#)

There are 3 configuration changes in 3 services [Show Details](#)

▼ **Services Accounts**

Proxy User Group	<input type="text" value="users"/>	
HDFS User	<input type="text" value="hdfs"/>	
Yarn User	<input type="text" value="yarn"/>	
dfs.permissions.superusergroup	<input type="text" value="hdfs"/>	+
Mapreduce User	<input type="text" value="mapred"/>	
Tez User	<input type="text" value="tez"/>	
WebHCat User	<input type="text" value="hcat"/>	
Hive User	<input type="text" value="hive"/>	
HCat User	<input type="text" value="hcat"/>	
ZooKeeper User	<input type="text" value="zookeeper"/>	
HBase User	<input type="text" value="hbase"/>	
Kafka User	<input type="text" value="kafka"/>	
Knox User	<input type="text" value="knox"/>	
Knox Group	<input type="text" value="knox"/>	+
Ranger User	<input type="text" value="ranger"/>	
Kms User	<input type="text" value="kms"/>	
Kms group	<input type="text" value="kms"/>	+
Ranger Group	<input type="text" value="ranger"/>	
Infra Solr User	<input type="text" value="infra-solr"/>	
Ambari Metrics User	<input type="text" value="ams"/>	
Metadata User	<input type="text" value="atlas"/>	
users_group	<input type="text" value="users"/>	🔒 + C
Beacon User	<input type="text" value="custombeaconuser"/>	🔒 + C
Beacon Group Name	<input type="text" value="custombeacongroup"/>	🔒 + C
Hadoop Group	<input type="text" value="hadoop"/>	
Smoke User	<input type="text" value="ambari-qa"/>	
Skip group modifications during install	<input type="checkbox"/>	
Have Ambari manage UIDs	<input checked="" type="checkbox"/>	
Whether to skip creating users and groups in a sysprepped cluster	<input type="checkbox"/>	+



**Note:** Make sure that the new user is added in the Ranger admin as well.

## Configure Knox SSO

If you have the DLM Engine on the cluster, you must take additional steps to set up your Knox SSO configuration.

### About this task

You will perform this DLM Engine Knox SSO setup on your clusters after you perform Dataplane installation. Refer to DP Installation for more information.

### Procedure

1. Export the Knox certificate:
  - a) From the Knox Gateway machine, run the following command: `$JAVA_HOME/bin/keytool -export -alias gateway-identity -rfc -file <cert.pem> -keystore /usr/hdp/current/knox-server/data/security/keystores/gateway.jks`
  - b) When prompted, enter the Knox master password.
  - c) Note the location path where you save the cert.pem file.
2. Enable the Knox SSO topology settings:
  - a) From **Ambari** > **DLM Engine** > **Configs** > **Advanced** > **Advanced beacon-security-site**, click the check-box beside **beacon.sso.knox.authentication.enabled** field.
  - b) Disable basic auth. From **Ambari** > **DLM Engine** > **Configs** > **Advanced** > **Advanced beacon-security-site**, uncheck the check-box beside **beacon.basic.authentication.enabled** field only in case of secured clusters. While using unsecured clusters, check the check-box beside **beacon.basic.authentication.enabled** field.
  - c) Set **beacon.sso.knox.providerurl** to `https://<knox-host>:8443/gateway/knoxssso/api/v1/webssso`.
  - d) Copy the contents of the PEM file exported in Step 1 to **beacon.sso.knox.publicKey**  
Ensure the certificate headers are not copied.



## Configure SSL (Optional)

If your HDP cluster is SSL-enabled, then you can configure SSL. You can use one of the two options to set up SSL certificates.

- Setup trusted CA certificates
- Setup self-signed certificates

### Set up trusted CA certificates

You can enable SSL for the DLM Engine using a certificate from a trusted Certificate Authority (CA). Certificates from a trusted CA are primarily used in production environments. For a test environment, you can use a self-signed certificate.

### Before you begin

- You must have root user access to the clusters on which DLM Engine is installed.
- You must have obtained a certificate from your CA, following their instructions.

### Procedure

1. Log in as root user on the cluster with DLM Engine installed.
2. Import the Certificate Chain Certificate and the certificate you obtained from your CA.

```
keytool -import -alias root -keystore <path_to_keystore_file> -
trustcacerts -file <certificate_chain_certificate>
```

```
keytool -import -alias jetty -keystore <path_to_keystore_file> -file
<certificate_from_CA>
```

### Set up self-signed certificates

You can enable SSL for the DLM Engine using a self-signed certificate. Self-signed certificates are primarily used in test environments. For a production environment, you should use a certificate from a trusted CA.

### Before you begin

You must have root user access to the clusters on which DLM Engine is installed.

### Procedure

1. Log in as root user on the cluster with DLM Engine installed.
2. Generate a key pair and keystore for use with DLM Engine.

```
keytool -genkey -alias jetty -keystore <certificate_file_path>
-storepass <keystore_password> -dname 'CN=beacon.host.com, OU=Eng, O=ABC
Corp,
L=Santa Clara, ST=CA, C=US' -keypass <key_password>
```

Follow the prompts and enter the required information.

- CN must be the FQDN of the DLM Engine host
- Default value for the key password is *password*.

If you change the password then you have to update the DLM configuration.

Following is sample command output:

```
keytool -genkey -alias jetty -keystore ~/tmp/ks -storepass password
What is your first and last name?
[Unknown]: beacon.host.com
What is the name of your organizational unit?
[Unknown]: Eng
What is the name of your organization?
[Unknown]: ABC Corp
What is the name of your City or Locality?
[Unknown]: Santa Clara
What is the name of your State or Province?
[Unknown]: CA
What is the two-letter country code for this unit?
[Unknown]: US
Is CN=beacon.host.com, OU=Eng, O=ABC Corp, L=Santa Clara, ST=CA, C=US
correct?
[no]: yes

Enter key password for <jetty>
(RETURN if same as keystore password):
```



**Note:** You will have to use this keystore file while configuring the DLM Engine for TLS in Ambari.

### 3. Export the certificate.

```
keytool -exportcert -alias jetty -keystore /my/file.keystore -file  
<certificate file path> -storepass <keystore_password> -rfc
```

#### What to do next

Configure the keystore for DataPlane use.

#### Configure TLS

In the Ambari UI, you enable TLS for DLM Engine and update the DLM Engine configuration if settings change.

#### Procedure

1. Navigate to **DLM Engine > Configs > Settings** and scroll to the Wire Encryption settings.
2. Toggle the **Beacon TLS Enabled** switch and enter or modify the appropriate properties:
  - Beacon TLS Port: The TLS listener port.
  - KeyStore Path: Path to the DLM Engine keystore
  - KeyStore Password: Password for the DLM Engine keystore
  - TrustStore Path: Path to the DLM Engine trustStore
  - TrustStore Password: Password for the DLM Engine trustStore
  - Key Password: Password for the DLM Engine key

## Wire Encryption

Beacon TLS Enabled

Yes

Beacon TLS Port

25443

Beacon TLS Port  
beacon\_tls\_port

Beacon TLS listen port.

truststore.jks

KeyStore Path

/etc/security/serverKeys/beamkeystore.jks

TrustStore Password

.....

KeyStore Password

.....

Key Password

.....

### Configure the keyStore for use by DLM Engine and Knox Proxy

While communicating with Knox Proxy, DLM Engine establishes two way SSL connection while acquiring the SSO token and hence the certificate (either self-signed for test setup or received from Certificate Authority) of DLM Engine needs to be imported in the Knox Proxy trustStore and the certificate (either self-signed for test setup or received from Certificate Authority) of Knox Proxy need to be imported in DLM Engine's truststore. For using Atlas, the Atlas certificates have to be copied to the Knox truststore.

### About this task

You must be aware that the Knox instance referred above is the one on the remote cluster DLM Engine needs to communicate with. The location of the trustStore can be configured both in DLM Engine and Knox Proxy. Perform the following steps to export the SSL certificate.

### Procedure

1. To export the SSL certificate of DLM Engine, on DLM Engine host, perform this:

```
$JAVA_HOME/bin/keytool -exportcert -alias jetty -keystore <beacon_keystore_file_path> -file
<beacon_cert.pem> -rfc
```

2. Copy the SSL certificate file <beacon\_cert.pem> to the Knox proxy host.
3. Import the SSL certificate file in the trustStore of Knox Proxy host. On Knox proxy host, perform this:
 

```
$JAVA_HOME/bin/keytool -import -alias jetty -keystore <path_to_knox_truststore_file> -file
<beacon_cert.pem>
```
4. To export the SSL certificate of Knox Proxy, on Knox Proxy host, perform this:
 

```
$JAVA_HOME/bin/keytool -exportcert -alias gateway-identity -rfc -file <knox_gw_cert.pem> -keystore /usr/hdp/
current/knox-server/data/security/keystores/gateway.jks
```
5. Copy the SSL certificate file <knox\_gw\_cert.pem> to the DLM Engine host.
6. Import the SSL certificate file in the trustStore of DLM Engine host. On DLM Engine host, perform this:
 

```
$JAVA_HOME/bin/keytool -import -alias gateway-identity -keystore <path_to_beacon_truststore_file> -file
<knox_gw_cert.pem>
```

## Configure Knox Gateway (Optional)

If you are using TLS (formerly SSL) wire encryption, you must configure DLM so that service requests are proxied through a Knox Gateway. This limits access to cluster services, providing a more secure environment. All cluster services such as Hive, Ambari, Ranger, Atlas, etc. are accessed through a Knox proxy by DP Platform and DLM Engine.

### Configure DLM proxying for TLS wire-encrypted clusters

If you are using TLS (formerly SSL) wire encryption, you must configure DLM so that service requests are proxied through a Knox Gateway. This limits access to cluster services, providing a more secure environment. All cluster services such as Hive, Ambari, Ranger, Atlas, etc. are accessed through a Knox proxy by DP Platform and DLM Engine.

### About this task

To use wire encryption with DLM, you must configure TLS on each cluster running DLM Engine so that the engine can authenticate and communicate with Knox across all paired clusters.

- You must perform this task on all the cluster nodes that have wire encryption enabled.
- If proxying is used, it must be enabled on both clusters in a DLM replication pair.

By default, proxying with Knox is disabled in DLM.

- When proxying is enabled, you cannot pair a cluster running DLM Engine version 1.0 with a cluster running a higher version of the engine.

### Before you begin

- TLS must be configured for Knox before proxying will work with DLM.
- To perform this task, you must have root user privileges on the DLM host and on all nodes that have Knox enabled.
- You must have created the /etc/knox/conf/topologies/dp-proxy.xml file during DPS configuration.

### Before you begin

### Procedure

1. In a terminal, navigate to the Knox topologies directory.

```
cd /etc/knox/conf/topologies
```

2. Log in as root and create a beacon-preauth.xml file.

```
vi beacon-preauth.xml
```

Example beacon-preauth.xml topology file:

You can copy and paste this sample content into your file and modify as needed.

```
<topology>
  <gateway>
    <provider>
      <role>federation</role>
      <name>HeaderPreAuth</name>
      <enabled>true</enabled>
      <param>
        <name>
          preauth.custom.header
        </name>
        <value>
          BEACON_USER
        </value>
      </param>
    </provider>
    <provider>
      <role>identity-assertion</role>
      <name>HadoopGroupProvider</name>
      <enabled>true</enabled>
    </provider>
    <!-- currently validating this acl for authorization -->
    <provider>
      <role>authorization</role>
      <name>AclsAuthz</name>
      <enabled>true</enabled>
      <param>
        <name>knoxtoken.acl</name>
        <value>beacon; *; *</value>
      </param>
    </provider>
  </gateway>
  <service>
    <role>KNOXTOKEN</role>
    <param>
      <name>knox.token.ttl</name>
      <value>120000</value>
    </param>
    <param>
      <name>knox.token.client.cert.required</name>
      <value>true</value>
    </param>
    <param>
      <name>knox.token.allowed.principals</name>
      <value><semicolon separated list of beacon dn names></value>
    </param>
    <param>
      <name>knox.token.client.data</name>
      <value>cookie.name=hadoop-jwt</value>
    </param>
  </service>
</topology>
```

The DN to be configured in `knox.token.allowed.principals` is the DN in the TLS certificate of each beacon host. Example DN: `CN=beacon.host.com, OU=Eng, O=ABC Corp, L=Santa Clara, ST=CA, C=US`



3. Change ownership of the beacon-preauth.xml file to Knox.

```
chown knox:hadoop beacon-preauth.xml
```

4. Open the DPS proxy topology file.

The dp-proxy.xml file was created during installation of the DPS Instance.

```
vi dp-proxy.xml
```

5. Ensure the following service definitions are in the file and configured with the correct FQDN host names.



**Important:** All DLM Engine servers that are registered with DPS must be included in this file. As new wire-encrypted clusters are registered, they must be added to this file manually.

```
<service>
  <role>BEACON</role>
  <url>https://<dlm_engine_host>:25443</url>
</service>
```

```
<service>
  <role>HIVE</role>
  <url>https://<hiveserver_host>:10001/cliservice</url>
</service>
```



**Tip:** You can get the HiveServer host from the default.xml file in the topology directory.

6. Create the truststore password entry using Knox cli:  
 /usr/hdp/current/knox-server/bin/knoxcli.sh create-alias gateway-truststore-password --value <password>The default JRE cacerts password is “changeit” .
7. Repeat this task on all cluster nodes that have Knox Gateway enabled.

## Complete the installation

During the installation, Ambari displays the list of recommended configurations. You need to verify the following configurations:

### Procedure

- hadoop.proxyuser.hive.hosts=\*
- hadoop.proxyuser.beacon.groups=\*
- hadoop.proxyuser.beacon.users=\*
- hadoop.proxyuser.beacon.hosts=\*
- hive.metastore.dml.events=true
- hive.repl.cmrootdir=/apps/hive/cmroot
- hive.repl.cm.enabled=true
- hive.metastore.transactional.event.listeners=org.apache.hive.hcatalog.listener.DbNotificationListener
- hive.repl.rootdir=/apps/hive/repl

Complete the remaining installation wizard steps

## Trusted proxy for DLM

In the existing scenario, communication between DLP App and DLM Engine is through the Knox, using SSO cookie.

Knox routes all the requests to cluster services like Ambari, Beacon, Ranger, Atlas, and others. Each of these services must enable Knox SSO and later setup Knox certificates on each of the requested services, which could be cumbersome and prone to errors. User has to go through Knox SSOs to access any services.

Additionally, when DLM Engine communicates with Source and Target clusters, the Knox SSO cookie is generated using beacon-preauth.xml file. Later, the service request calls are routed through Knox. Again, the cluster services have to enable the Knox SSO, which could be cumbersome and possibly end up being prone to errors.

### New Trusted Proxy

In the new trusted proxy set up, the communication between DLM App and DLM Engine goes through Knox, and the authentication at DLM Engine will be handled using Kerberos. The DLM App to Knox authentication will still be using SSO. Use the recommended HDP stack versions and also install Knox certificate on the DLM Engine service.

### HDP stack for Trusted Proxy

Before setting up the new **Trusted Proxy** for DLM, make a note of the supported stack details.



**Caution:** As of **DLM 1.5.0** release, trusted proxy feature is certified to function only on **HDP 2.6.5 to 2.6.5** clusters.

### HDP 2.6.5 clusters

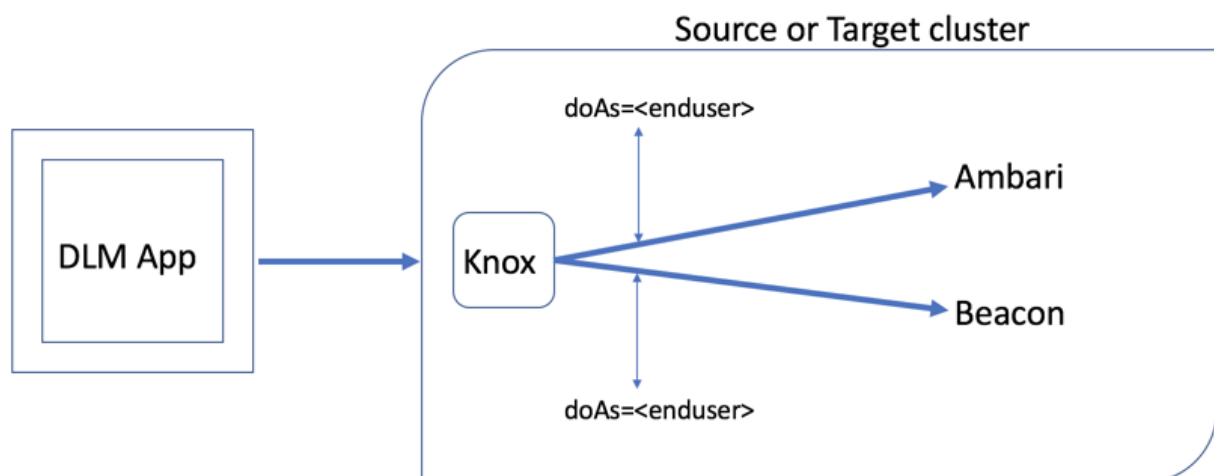
- HDP release - 2.6.5.1175
- Ambari - 2.6.2.33-3
- Dataplane - 1.2.2
- DLM - 1.5.0

### Install Knox certificates

Follow these instructions to download and install Knox certificates on your DLM Engine service:

- Before pairing two secured clusters, for example, Cluster0 and Cluster1, ensure that Beacon is running on Cluster0 on Beacon0 host and Cluster1 on Beacon1 host.
  - Knox is running on Cluster0 on Knox0 host and on Cluster1 on Knox1 host.
  - Later, run the following commands on Beacon0 host:
    - `openssl s_client -connect <knox0>:8443 <<<' | openssl x509 -out /tmp/knox0.crt`
    - `keytool -import -trustcacerts -keystore <java cacert path> -storepass <storepass> -noprompt -alias <alias> -file /tmp/knox0.crt`

For example: `keytool -import -trustcacerts -keystore /usr/lib/jvm/java-openjdk/jre/lib/security/cacerts -storepass changeit -noprompt -alias Knox0 -file /tmp/knox0.crt`
  - `openssl s_client -connect <knox1>:8443 <<<' | openssl x509 -out /tmp/knox1.crt`
  - `keytool -import -trustcacerts -keystore <java cacert path> -storepass <storepass> -noprompt -alias <alias> -file /tmp/knox1.crt`
  - Restart Beacon0 using Ambari.
- Repeat the above steps for Beacon1 host as well.



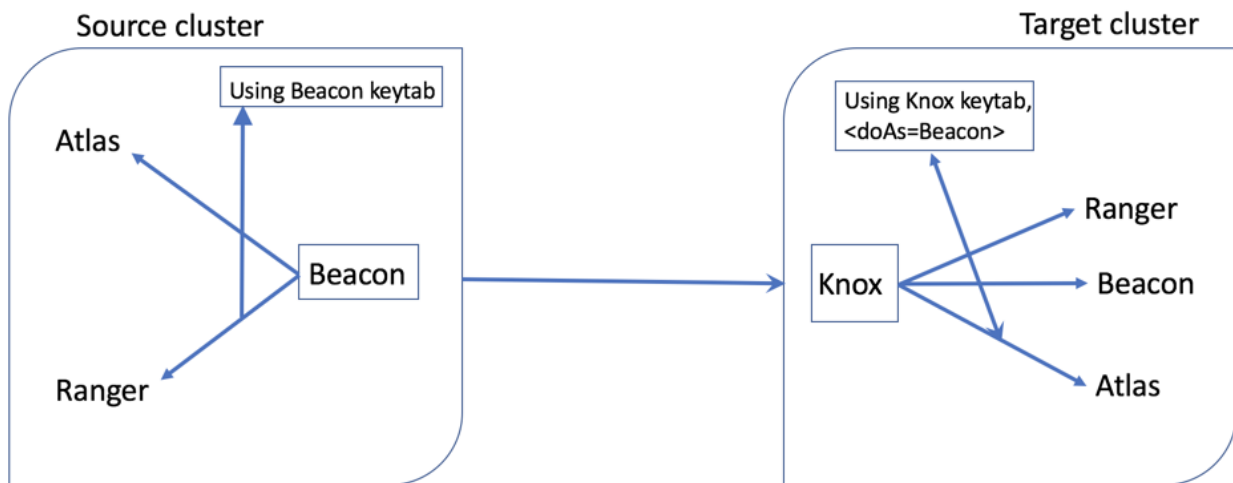
As a prerequisite for this set up to work, the clusters must be kerberized.

When a call is made from DLM App to DLM Engine, the trusted proxy flow takes place in the following manner:

- Knox authenticates the incoming request using SSO.
- Knox initiates a new request to service using 'Knox' keytab with doas=<enduser>
- The service authenticates the requesting user as 'knox', but uses enduser UGI.
- Only 'Knox' can set doas parameter. Any extra validation for the doAs user will be done in services.

Again, in the new trusted proxy environment, DLM Engine to remote cluster service communication happens as follows:

- Knox authenticates the incoming request using Kerberos.
- Knox initiates a new request to service using 'Knox' keytab with doas=<enduser>
- The service authenticates the requesting user as 'knox', but uses enduser UGI.
- Only 'Knox' can set doas parameter. Any extra validation for the doAs user will be done in services.



Use the following information to set up the new trusted proxy for DLM:

- Supported stack versions: Ambari (2.6.2), HDP (2.6.5), and DataPlane Platform (1.2.2)
- Install Knox. Don't enable SSO and Knox proxy
- Enable Kerberos for Ambari service
- Download the script: [DP cluster setup](#) utility
- Run `sudo python dp-cluster-setup-utility.py` utility from the Knox host

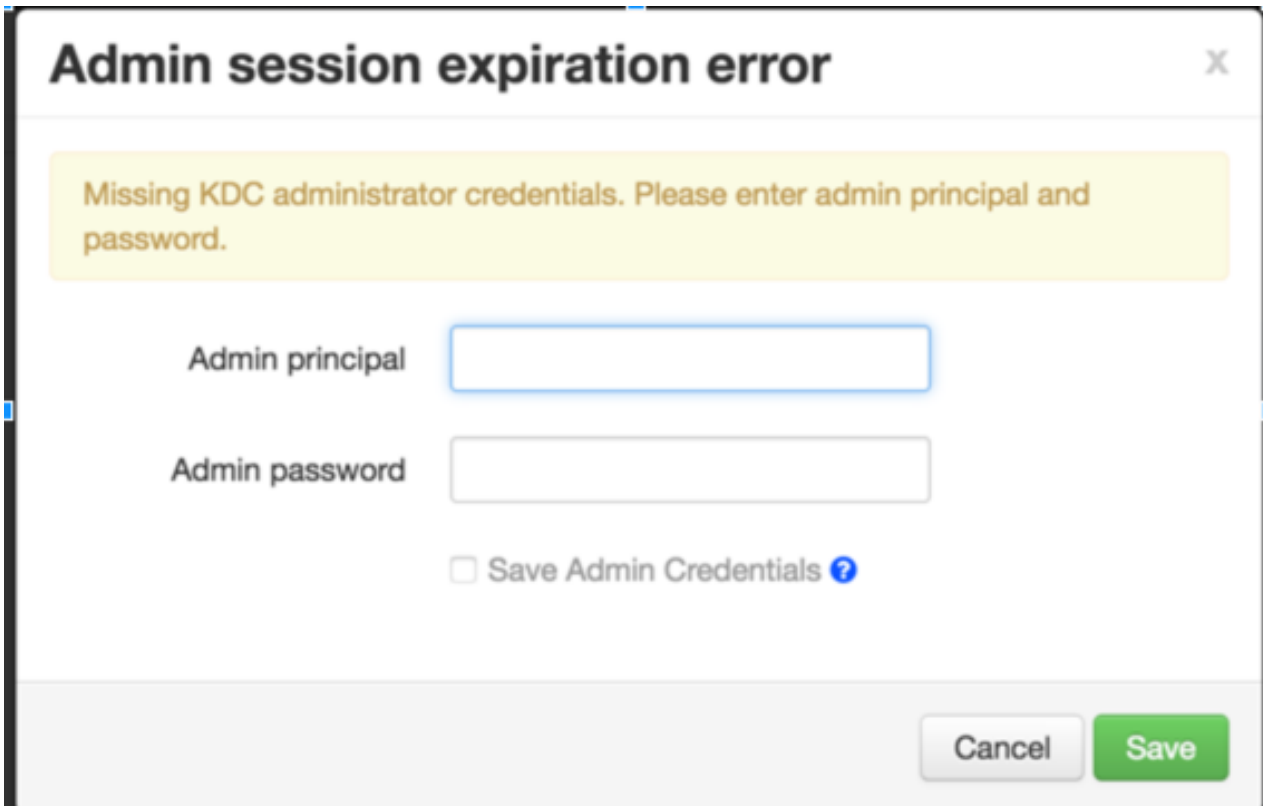


**Note:** If you are using HDP 2.6.5 cluster, you must run `ambari-server setup-trusted-proxy` before running the above script.

- Provide all in-prompt inputs
- Adds trusted proxy configurations to Ambari, Ranger, Atlas, and DLM Engine
- Creates Knox topologies - token, dp-proxy, and beacon-proxy
- Registers the cluster with DataPlane platform
- Import Knox certificates in the DLM Engine service
- Finally, you must restart all the available services with stale configurations on Ambari gateways (both clusters)

### Configure Kerberos security (Optional)

When DLM Engine service is installed on a secured cluster, Ambari requires KDC administrator credential in order to setup DLM Engine service keytab.



The image shows a dialog box titled "Admin session expiration error" with a close button (X) in the top right corner. A yellow warning box contains the text: "Missing KDC administrator credentials. Please enter admin principal and password." Below this, there are two input fields: "Admin principal" and "Admin password". Underneath the "Admin password" field is a checkbox labeled "Save Admin Credentials" with a help icon (question mark in a circle). At the bottom right of the dialog, there are two buttons: "Cancel" and "Save".

### Enable Ranger Deny policy

DLM replication adds deny policy on the target for the dataset that is replicated. By default, Ranger admin UI does not display deny policies.

For more information, see [Providing Authorization with Apache Ranger](#) to enable deny conditions in Ranger.

#### Related Information

[Ranger documentation](#)

### Restart services

Ambari shows 'Restart Required' for dependent services. Restart the services to make sure the configurations are reflected:

#### Procedure

- To refresh 'beacon' user privileges as hadoop superuser, you can use refresh hadoop group mapping command or restart namenode service. Refresh the hadoop group mapping commands:

```
hdfs dfsadmin -refreshSuperUserGroupsConfiguration
hdfs dfsadmin -refreshUserToGroupsMappings
```

- Verify that Beacon was added as a user to the HDFS superuser group.

```
hdfs groups beacon
```

The output should display HDFS (or value of dfs.permissions.superusergroup config) as one of the groups.

Restart Hiveserver2 to refresh Hive configurations.

Restart node managers to create 'beacon' OS user.

## Verify the DLM Engine installation

Ensure that your DLM Engine set up is functional.

### Procedure

- Scan through Ambari logs of DLM Engine service installation for any errors.
- Run the /usr/dlm/current/beacon/bin/verify\_beacon\_knox.sh utility script to ensure that Knox SSO is configured successfully with the DLM Engine.

```
/usr/dlm/current/beacon/bin/verify_beacon_knox.sh https://<knox_hostname>:8443/gateway/knoxssso/api/v1/webssso http://<beacon_hostname>:25968 admin admin
```

- Verify that DLM Engine was added as a user to the HDFS superuser group.

```
hdfs groups beacon
```

The output should display HDFS (or value of the dfs.permissions.superusergroup config) as one of the groups.

- Verify that DLM Engine user is setup as Ranger admin. In secure cluster, this will be 'beacon' user with role 'Admin. In unsecure cluster, this will be 'beacon\_ranger' user with 'Admin' role.

User List

Add New User
Set Visibility
🗑️

<input type="checkbox"/>	User Name	Email Address	Role	User Source	Groups	Visibility
<input type="checkbox"/>	beacon		User	External	--	Visible
<input type="checkbox"/>	beacon_ranger		Admin	Internal	--	Visible

- Verify that the 'DLM Engine' user has Hive access through Ranger policies.

'DLM Engine' user has **repladmin** privileges

'DLM Engine' user has access to all databases, tables, and columns.

**Ranger** Access Manager Audit Settings admin

Service Manager mycluster1\_hive Policies Edit Policy

### Edit Policy

**Policy Details :**

Policy Type: Access

Policy ID: 10

Policy Name \*: all - database, table, column  enabled

database:   include

table:   include

Hive Column \*:   include

Audit Logging: YES

Description: Policy for all - database, table, column

**Allow Conditions :** hide

Select Group	Select User	Permissions	Delegate Admin	
<input type="text" value="Select Group"/>	<input type="text" value="hive rangerlookup ambari-qa"/>	<input type="checkbox"/> select <input type="checkbox"/> update <input type="checkbox"/> Create <input type="checkbox"/> Drop <input type="checkbox"/> Alter <input type="checkbox"/> Index <input type="checkbox"/> Lock <input type="checkbox"/> All <input type="checkbox"/> Read <input type="checkbox"/> Write <input type="checkbox"/> ReplAdmin <input type="checkbox"/> Service Admin	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input type="text" value="Select Group"/>	<input type="text" value="beacon"/>	<input type="checkbox"/> All <input type="checkbox"/> ReplAdmin	<input type="checkbox"/>	<input type="checkbox"/>

DLM Engine service user requires **Service Admin** permission to perform Hive related operations. Make sure that DLM Engine service user is assigned **Service Admin** privilege.

**Edit Policy**

---

**Policy Details :**

Policy Type **Access**

Policy ID **11**

Policy Name \*  **enabled**

\*

Audit Logging **YES**

Description

---

**Allow Conditions :** hide -

Select Group	Select User	Permissions	Delegate Admin	
<input type="text" value="Select Group"/>	<input type="text" value="hive"/> <input type="text" value="rangerlookup"/> <input type="text" value="ambari-qa"/>	<input type="checkbox"/> select <input type="checkbox"/> update <input type="checkbox"/> Create <input type="checkbox"/> Drop <input type="checkbox"/> Alter <input type="checkbox"/> Index <input type="checkbox"/> Lock <input type="checkbox"/> All <input type="checkbox"/> Read <hr/> <input type="checkbox"/> Write <input type="checkbox"/> ReplAdmin <input type="checkbox"/> Service Admin	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input type="text" value="Select Group"/>	<input type="text" value="beacon"/>	<input type="checkbox"/> Service Admin	<input type="checkbox"/>	<input type="checkbox"/>













Verify that Hive service is setup with Ranger authoriser and doas=false.

List of Atlas policies with 'DLM Engine' user on a secure cluster.

List of Policies : mycluster0\_atlas

Search for your policy...

Add New Policy

Policy ID	Policy Name	Status	Audit Logging	Groups	Users	Action
13	all - taxonomy	Enabled	Enabled	--	atlas rangerlookup admin	 
14	all - operation	Enabled	Enabled	--	atlas rangerlookup admin beacon	 
15	all - type	Enabled	Enabled	--	atlas rangerlookup admin	 
16	all - entity	Enabled	Enabled	--	atlas rangerlookup admin beacon	 
17	all - term	Enabled	Enabled	--	atlas rangerlookup admin	 

Atlas entity policy.

**Ranger** Access Manager Audit Settings admin1

Policy ID **16**

Policy Name \* all - entity **enabled**

entity \* \* \* **include**

Audit Logging **YES**

Description Policy for all - entity

**Allow Conditions :** hide

Select Group	Select User	Permissions	Delegate Admin
Select Group	* atlas * rangerlookup * admin	read create update delete All	<input checked="" type="checkbox"/> <span>✖</span>
Select Group	* beacon	read create update delete All	<input type="checkbox"/> <span>✖</span>

**Save** **Cancel** **Delete**

Atlas operation policy.

The screenshot shows the Ranger web interface for editing a policy. The breadcrumb trail is: Service Manager > mycluster0\_atlas Policies > Edit Policy. The page title is 'Edit Policy'.

**Policy Details:**

- Policy Type: Access
- Policy ID: 11
- Policy Name: all - operation (enabled)
- operation (dropdown) with a search box containing '\* \*' (include)
- Audit Logging: YES
- Description: Policy for all - operation

**Allow Conditions:**

Select Group	Select User	Permissions	Delegate Admin
Select Group	x atlas x admin	read create update delete All	<input checked="" type="checkbox"/>
Select Group	x beacon	All create delete read update	<input type="checkbox"/>

Buttons: Save, Cancel, Delete

## About requested events missing in Notification Log table

You must consider all these factors before setting the value of `metastore.event.db.listener.timetolive` parameter.

The `metastore.event.db.listener.timetolive` configuration parameter is used to control the time for which an event will be kept in the database listener queue or the backing RDBMS. Note that, if the configuration value is set too high, the number of events in the queue will increase and can impact performance in terms of normal operation. If the value is set too low, the events might get deleted from the queue, before replication could read it and thus could cause incremental replication to fail. In such a scenario, you must bootstrap the system again to get back to the consistent replicated state.

The value of `metastore.event.db.listener.timetolive` parameter must be large enough to avoid cleaning up of events, when replication of previous events is in progress. During the bootstrap phase, events added once bootstrap starts should be present for the next incremental to succeed. As the bootstrap is performed for whole database, it might consume more time. In case of incremental load, if the replication frequency is too low, incremental load gets triggered that will have large number of events to replicate. This may increase the time required to execute the load.

So, while setting the parameter value, the replication trigger frequency should be taken into consideration to avoid events getting cleaned up before replication finishes. The replication time depends on many factors like the amount of data to be replicated, bandwidth between the clusters, and number of objects like partitions, table, and functions present in the database. For replicating to a cloud-based cluster, the time taken is more as the file system operations takes longer in cloud file system than in HDFS.

## Install the DLM Service

After installing the DP Platform, install the DLM Service App. All service applications are installed as RPMs on the same host as DP Platform. You can install one DP app or a combination of DP apps with DP Platform.

### Before you begin

- You must have root access to the host on which you are installing DLM.
- You must have successfully installed DP Platform.

### Procedure

1. Log in as root to the host on which you set up the DP repositories.

```
sudo su
```

2. Verify that all DataPlane containers are running as expected.
3. Install the RPMs for the DLM service application.

```
yum install dlm-app
```

A folder is created that contains the Docker image tarball files and a configuration script.

If the yum command fails, then the local repository was not set up correctly. Check the repository file `/etc/yum.repos.d/dlm.repo` on the host.

4. Navigate to the directory containing the installation scripts for the DLM service.

```
cd /usr/dlm-app/current/apps/dlm/bin
```

5. Load the DLM Docker images and initialize the environment.

```
./dlmdeploy.sh init
```

Loading the images might take a while.

6. When prompted to enter the previously entered master password for DataPlane apps, specify the same password that you have created while installing DP to protect the secret storage.

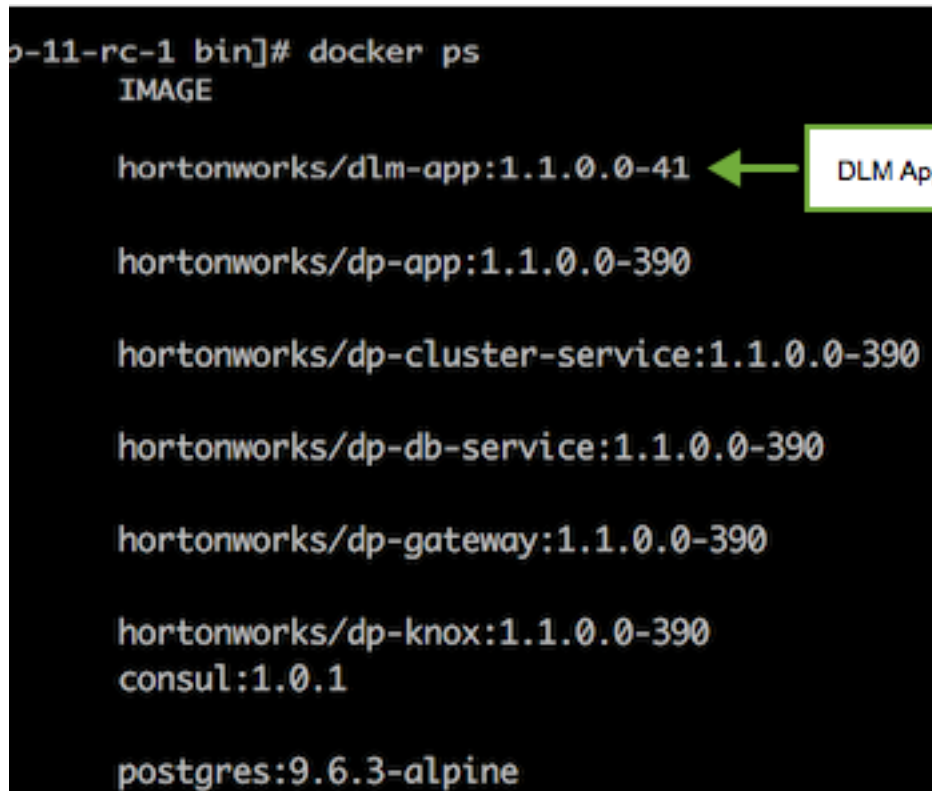
The DLM app requires this to store cloud credentials to the secret storage.

7. Verify that the container you installed is running.

```
docker ps
```

The entry should be similar to the following, displaying `dlm-app:1.1.1.0-25`:

```
p-11-rc-1 bin]# docker ps
IMAGE
hortonworks/dlm-app:1.1.0.0-41
hortonworks/dp-app:1.1.0.0-390
hortonworks/dp-cluster-service:1.1.0.0-390
hortonworks/dp-db-service:1.1.0.0-390
hortonworks/dp-gateway:1.1.0.0-390
hortonworks/dp-knox:1.1.0.0-390
consul:1.0.1
postgres:9.6.3-alpine
```



If any containers are not running, you must destroy the containers and start over, as described in the troubleshooting section of the *DP Installation* guide.

#### Related Tasks

[Install the DLM Engine MPack](#)

#### Related reference

[DLM support requirements](#)

#### Related Information

[DPS Installation](#)

[Hortonworks Support Matrix](#)

## Verify DLM

After the installation is complete, verify the following DLM functionality:

#### Procedure

- Pair the clusters
- Set up a sample replication
- Verify that the replication works

For more information, see *Getting Started with DLM*.

#### Related Information

[Getting Started with DLM](#)

## Advanced configurations (Optional)

In addition to the basic configuration required to set up and use DLM replication, there are some advanced configuration options you might choose to implement. For example, you might want to configure TLS wire encryption for protecting data in motion, or Transparent Data Encryption (TDE) for protecting data at rest.

### Using TDE with DLM

Encryption with Transparent Data Encryption (TDE) is supported in DLM for protecting data at rest. You can use TDE to prevent people from inappropriately gaining access to your data. The source root directory must be either encrypted or unencrypted. DLM Engine does not support replication when part of the data is unencrypted and part encrypted with one or more keys.

#### Replication scenarios for TDE-enabled data

DLM supports replication of HDFS and Hive data when:

- Both source and destination are encrypted with the same key (on-premise to on-premise replication only)
- Both source and destination are encrypted with different keys
- Source is unencrypted, but destination is encrypted

Note that DLM does *not* allow replication when the source is encrypted, but the destination is unencrypted.

The source and target directories can be encrypted with one of the following:

#### Same Key

If the source and destination are encrypted with the same key, DLM engine optimizes replication by replicating the encrypted blocks without decrypting and re-encrypting data during replication.

#### Different Key

During replication, source data is decrypted using the source key and encrypted using the destination key.

#### TDE in HDFS

HDFS implements transparent, end-to-end encryption of data read from and written to HDFS.

- TDE should be configured in the HDFS service, and the directories have to be marked as encryption zones using the encryption keys.

Refer to the [Data Protection: HDFS Encryption](#) in the HDP *Security* guide for more information.

- You can set TDE per directory or per cluster on HDFS.

#### TDE with Hive

- For Hive replication in DLM, any cluster that is using TDE and acts as a source for replication *must* have the entire data warehouse in a single encryption zone.
- You can set TDE only at cluster level for Hive replication.

### Configure TDE for HDFS replication

You set up TDE for HDFS replication using the instructions in the HDP *Security* guide. You can set TDE per directory or per cluster on HDFS. During the replication process, the source data is decrypted using the source key and is encrypted using the destination key.

### Procedure

1. (Optional) Encrypt the source directory and grant the DLM Engine user access to the KMS key in the source Ranger service.  
Refer to [Encryption in HDFS](#) and [Ranger KMS Setup](#) for instructions.
2. Encrypt the destination directory and grant the DLM Engine user access to the KMS key in the destination Ranger service.  
Refer to [Encryption in HDFS](#) and [Ranger KMS Setup](#) for instructions.

### Results

After you configure TDE on the data to be replicated, DLM can identify which directories have TDE enabled. When configuring a replication policy in the DLM App, you can identify and select the TDE-enabled data. You also have the option of replicating data using the same TDE key on both the source and destination, to reduce the overhead of decryption and encryption.

## Configure TDE for Hive replication

You set up TDE for HDFS replication using the instructions in the HDP *Security* guide. You can set TDE only at cluster level for Hive replication. During the replication process, the source data is decrypted using the source key and is encrypted using the destination key.

### Procedure

1. (Optional) Encrypt the source Hive warehouse directory and any additional directories as required by the Hive service and grant the DLM Engine user access to the KMS key in the source Ranger service.  
Refer to [Encryption in Hive](#) and [Ranger KMS Setup](#) for instructions.
2. Encrypt the destination Hive warehouse directory and any additional directories as required by the Hive service and grant the DLM Engine user access to the KMS key in the destination Ranger service.  
Refer to [Encryption in Hive](#) and [Ranger KMS Setup](#) for instructions.

### Results

After you configure TDE on the data to be replicated, DLM can identify which directories have TDE enabled. When configuring a replication policy in the DLM App, you can identify and select the TDE-enabled data. You also have the option of replicating data using the same TDE key on both the source and destination, to reduce the overhead of decryption and encryption.

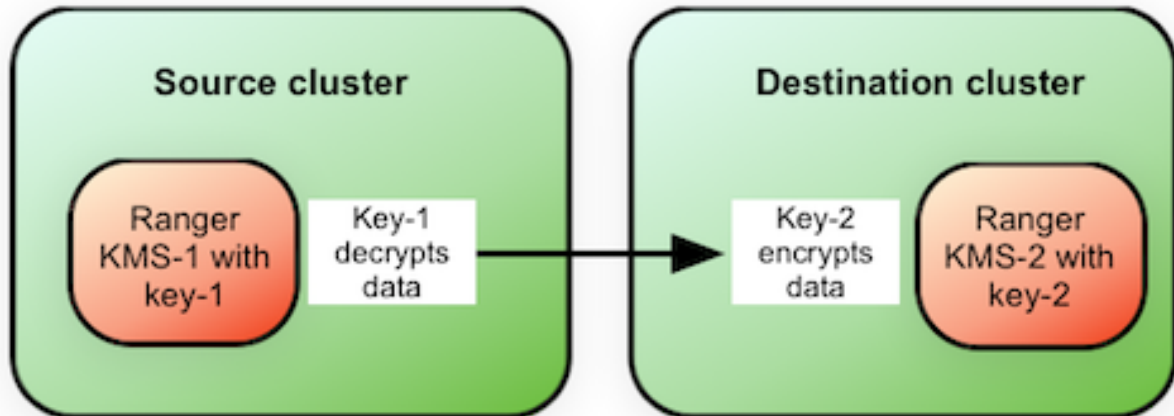
## On-premise replication using different keys and Ranger-KMS instances

For data at rest that is encrypted using transparent data encryption (TDE), DLM can replicate data across different Ranger Key Management Service (KMS) encryption zones and using different encryption keys. This capability applies to replication between on-premise clusters for both HDFS and Hive data.

- Permissions are replicated along with the data.
- Ranger key management and key authorization management must be done external to DLM by an administrator with access to Ranger.
- For Hive replication
  - The entire warehouse must be in one encryption zone.
  - The change management directory (cmroot directory) should also be setup in the same encryption zone as the warehouse directory.

Following is an example of how this replication scenario might apply:

- The source cluster with Ranger-KMS-1 instance uses key-1 to decrypt the data, then passes the data to the destination.
- The destination cluster with Ranger-KMS-2 instance uses key-2 to encrypt the data on the destination.



## Preparing HDP cluster for Hive cloud replication

Hive replication from on-prem cluster to the cloud storage requires minimal cluster on the target with metadata services like HMS, Ranger, Atlas, and DLM engine. HMS should be configured with Hive warehouse directory on cloud storage. Refer to the following steps:

### Procedure

1. Hive Data Locations - Hive metastore requires these specific configurations to point Hive data on cloud storage. Note that both `hive.metastore.warehouse.dir` and `hive.repl.replica.functions.root.dir` should be configured in the same bucket.

```
hive.metastore.warehouse.dir=<cloud storage>
hive.repl.replica.functions.root.dir=<cloud storage>
hive.warehouse.subdir.inherit.perms=false
```

2. Cloud access credentials - When Hive metastore is configured with Hive warehouse directory on cloud storage, Hive will also require the credentials to access the cloud storage. This can be setup with one of the following configurations:
  - Access key and secret key
  - Session token
  - For IAAS clusters, setup instance profiles
3. Cloud encryption configurations - If the bucket is encrypted, setup the [bucket encryption details](#)



#### Note:

Set all these configurations in `hive-site.xml`.

## Security considerations

Ports need to be open for DLM Engine, Knox, Atlas, and DataNodes.

Have the following ports available and open on each cluster:

Default Port Number	Purpose	Comments	Required to be open?
---------------------	---------	----------	----------------------



25968	Port for DLM Engine (Beacon) service on hosts	Accessibility is required from all clusters. “Beacon” is the internal name for the DLM Engine. You will see the name Beacon in some paths, commands, etc.	Yes
8020	NameNode host		Yes
50010	All DataNode hosts		Yes
8080	Ambari server host		Yes
10000	HiveServer2 host	Binary mode port (Thrift)	Yes
10001	HiveServer2 host	HTTP mode port	Yes
9083	Hive metastore		Yes
2181	ZooKeeper hosts		Yes
6080	Ranger port		Yes
21000/21443	Atlas endpoint for Web UI and rest endpoint	Default non-SSL: 21000, SSL: 21443	Yes
8050	YARN port		Yes

## Upgrading DLM

There are multiple components and clusters that are involved while upgrading DLM.

- Dataplane Platform and DLM App which are installed on the Dataplane host.
- HDP and DLM Engine on the multiple HDP clusters.
- Before performing the DLM upgrade make a copy of /usr/dlm/current/beacon/conf directly.

Post-upgrade, make sure all the manual changes that were performed previously must be repeated. You can refer the older configurations from the backup directory.



**Note:** On upgrading HDP version from 2.6.5 to 3.x, DLM Engine would be removed as part of upgrade process. You must install the required version of DLM Engine once the HDP upgrade succeeds.



**Important:** After upgrading from DLM 1.4.x to 1.5.x on a secure cluster, if the Knox gateway is not installed on the cluster, you must set beacon.kerberos.authentication.enabled as false.

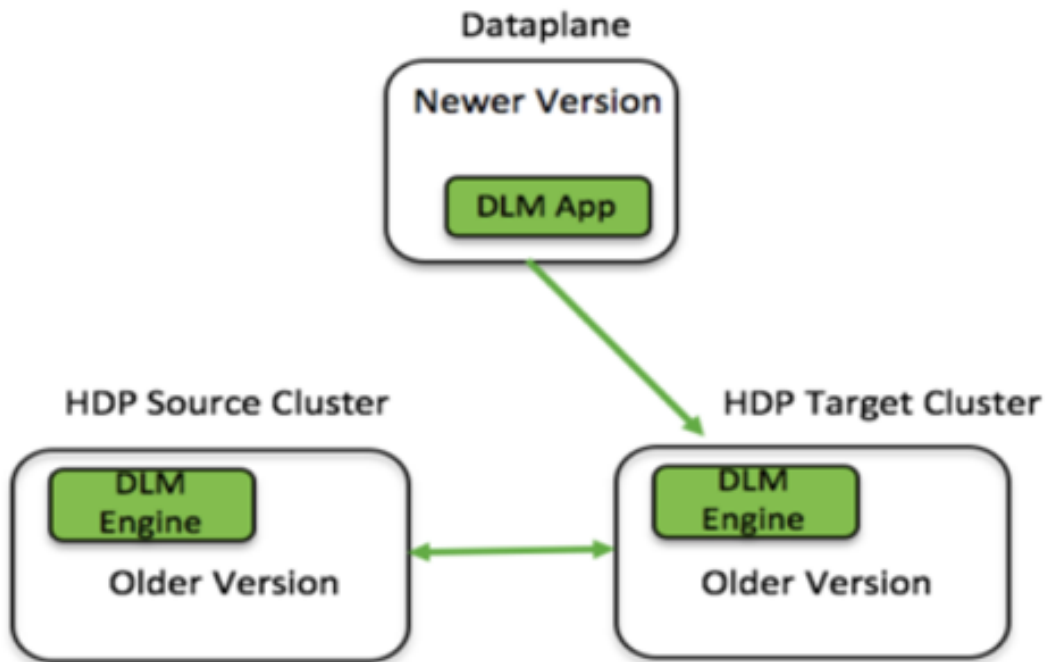


**Caution:** Hive replication from DLM 1.4.x/HDP 2.6.5 to DLM 1.5.x/HDP 3.1 is not possible. You must upgrade DLM on source from 1.4.x to 1.5.x to proceed with this scenario.

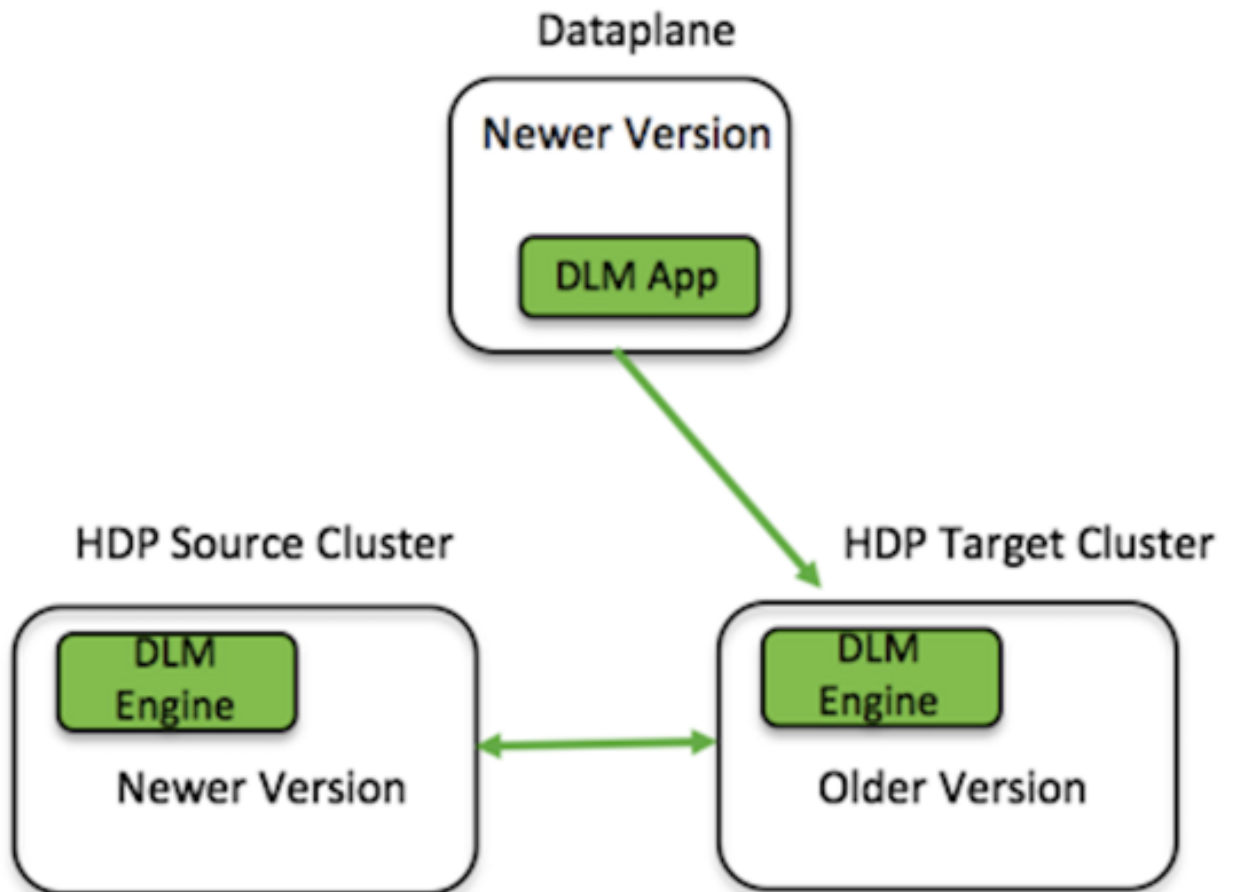
These components and clusters may not be upgraded at once, but can be upgraded over a period of time. DLM is designed to work across versions for these components.

However, the following upgrade order is recommended for performing the upgrade process.

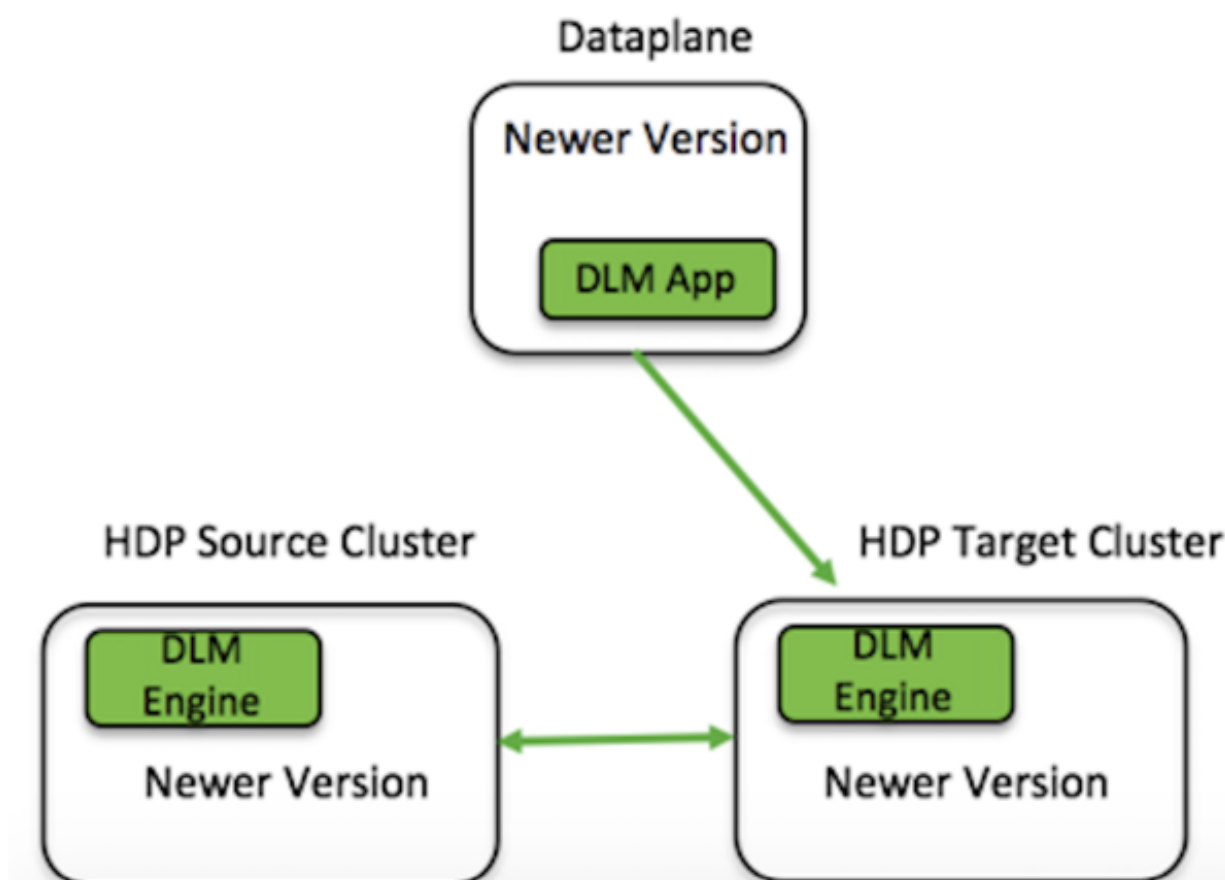
1. Upgrade Dataplane Platform and DLM App on the Dataplane host.



2. Upgrade HDP and DLM Engine on one cluster



3. Upgrade HDP and DLM Engine on another cluster



### Related Information

[DataPlane Installation guide](#)

## Upgrade DataPlane platform

To upgrade from one version of DP to the higher version, see the DataPlane Installation guide.

## Upgrade DLM App

Follow these steps to upgrade from one version of the DLM App to the higher version.

### Procedure

- Run `wget -nv http://s3.amazonaws.com/dev.hortonworks.com/DLM-APP/centos7/1.x/BUILDS/X.X.X.X-XX/dlm-appbn.repo -O /etc/yum.repos.d/dlm-app.repo`.



**Note:** Replace X.X.X.X-XX with 1.5.0.0-44

- Run `yum upgrade dlm-app`.
- Go to the upgraded version directory: [ /usr/dlm-app/X.X.X.X-XX/apps/dlm]
- Run `./bin/dlmdeploy.sh upgrade`.

This will destroy the running DLM container and bring up new laid out updated bits (image).

## Upgrade DLM Engine

DLM 1.5.0 version supports HDP 2.6.5.1175 and HDP 3.1.0.31 releases. If you are on an earlier release of HDP, you must upgrade to HDP 2.6.5 1175 release.

### Procedure

1. Stop DLM Engine service using Ambari UI. **Ambari > DLM Engine > Service Actions > Stop**
2. Remove the DLM Engine service from Ambari. **Ambari > DLM Engine > Service Actions > Delete Service**
3. Login to DLM Engine host using SSH. Remove the beacon RPM on DLM Engine host using:  
`yum remove <beacon_RPM_name>`  
 To get the name of beacon rpm, do: `rpm -qa|grep beacon`
4. On the DLM Engine host, remove /usr/dlm/current and /etc/beacon/conf folder directories.
5. Login to Ambari host using SSH.
6. Uninstall the mpack for beacon in Ambari server: `ambari-server uninstall-mpack --mpack-name beacon-engine.mpack`
7. Install the new beacon mpack:  
`ambari-server install-mpack --mpack=/tmp/beacon-ambari-mpack-<version>.tar.gz --verbose`
8. Restart Ambari server.  
`ambari-server restart`
9. Setup a local repository with new DLM engine binaries.
10. Update the beacon repo URL in the Ambari UI.  
 Go to **Ambari > Admin > Stacks and Versions > Manage > HDP- <version>**  
 Under repositories, update the new repo URL for DLM for the OS of the cluster and click Save.
11. Add the DLM Engine service in Ambari UI by going through steps for adding a service: **Ambari > Actions > Add Service > DLM Engine**

## Migrating DLM Engine from one host to another

When you reinstall the DLM Engine on another host, make sure your configuration is pointing to the existing beacondb. Specify the host name where the beacondb resides during the installation process. Once the DLM Engine service is up and running on your new host, verify that the Knox SSO is configured by running the `/usr/dlm/current/beacon/bin/verify_beacon_knox.sh` utility script:

### Procedure

- In case Knox is not set up, you need to configure Knox and restart the services.
- If Knox is set up successfully, you will see a message that Knox is successfully set up.

## Troubleshooting DLM

You can refer to the troubleshooting details for the following:

- DataPlane
- DLM App
- DLM Engine
- Common Errors

## DataPlane platform

You might encounter few errors while registering a cluster in DP. To know more about the possible causes and possible resolutions, see the Cluster Registration Error Messages section in DataPlane Installation Guide.

## DLM Engine

On an HDP cluster, the host, where DLM Engine is installed, ensure the following:

### Procedure

- Configuration is in directory `/etc/beacon/conf`. `Beacon.yml` and `beacon-security-site.xml` contain the service configurations.
- Service logs are in `/var/log/beacon/beacon-application-<hostname>.log`, the log files are rolled hourly.

## Common errors

This section list the common issues that you might run into during the DLM installation.

- Access denied for user Beacon. Superuser privilege is required.

Beacon user should be setup as HDFS superuser.

- Permission denied: user=beacon, access=READ, inode="<some directory>":<user>:<group>:drwxr-xr-x

Beacon user should be setup as HDFS superuser.

- Authentication failure while communicating to Ranger admin:

Check the ranger admin logs at `/var/log/ranger/admin/xa_portal.log`. If the error is 'Request failed.

`loginId=<beacon user>`, `logMessage=User is not allowed to access the API`, make sure the Beacon user is setup as Ranger admin or even DLM Engine restart will setup the required users and their permissions.

- `org.apache.hadoop.hive.ql.security.authorization.plugin.HiveAccessControlException:Permission denied: user [beacon] does not have [REPLADMIN] privilege on [^<dbname>`]`

Beacon user should have repladmin privileges in Hive, please set up these Hive authorisation rules in Ranger.

- Error while compiling statement: FAILED: SemanticException

`org.apache.hadoop.security.AccessControlException: Permission denied: user=hive, access=EXECUTE, inode="/apps/hive/repl/563db5d9-caee-4015-b020-9cbd7bd8b41b/_dumpmetadata":beacon:hadoop:drwx-----`

Check that Hive is setup with `doas=false`.

- Hive replication fails with the following error in `hiveserver2.log` - 'Unauthorized connection for super-user: hive/<host>@EXAMPLE.COM from IP <ip>'

`Hadoop.proxyuser.hive.hosts` should be set to \*

- `org.apache.hadoop.security.authorize.AuthorizationException: User:beacon not allowed to do 'DECRYPT_EEK' on 'backup-encryption-key'`

Beacon user should have access to encryption key used in TDE setup.

- On DLM UI, error in connecting to DLM engine.
- Replication job failing with Ranger authentication.