

DSS installation and Upgrade 1

Data Steward Studio Installation and Upgrade

Date of Publish: 2018-05-24



<https://docs.hortonworks.com>

Contents

Installation Overview.....	3
Data Steward Studio Installation Steps.....	3
Supported Configurations for DSS Installation.....	3
Obtaining necessary DSS software.....	5
DSS Installation Artifacts.....	5
Set Up a Local Repository.....	6
Create the Repository Configuration File.....	7
Installing Profiler Agent on Clusters.....	8
Pre-installation tasks for DP Profiler Agent for HDP 2.6.5 version.....	8
Pre-installation tasks for DP Profiler Agent for HDP 3.x versions.....	9
Update Service Specific Properties for HDP 3.X versions.....	10
Configure an external database for DataPlane Profiler Agent.....	11
Configure MySQL external database.....	11
Configure Postgres external database.....	11
Grant Permissions in Ranger and YARN.....	12
Install the Data Plane Profiler Agent.....	12
Ambari Dataplane Profiler Configs.....	14
Tuning Parameters for the Profiler Service.....	20
Update Knox Topologies.....	21
DP Proxy Settings for Atlas HA Configuration.....	21
Installing the Data Steward Studio App.....	23
Set Up a Local Repository.....	24
Create the Repository Configuration File.....	25
Configure an external database for Data Steward Studio App.....	26
Install the Data Steward Studio Service App.....	27
Enable the Data Steward Studio in the Data Plane Platform.....	27
Add Users and Assign Roles for DSS App.....	28
Add Data Lakes to the Data Plane Service.....	29
Upgrade Data Steward Studio.....	29
Pre-upgrade tasks.....	29
Additional pre-upgrade tasks to upgrade from DSS 1.3.1 to DSS 1.5.0.....	30
Run the DataPlane Profiler pre-upgrade script.....	30
Upgrade DataPlane Profiler on Ambari.....	30
Additional post-upgrade tasks to upgrade from DSS 1.3.1 to DSS 1.5.0.....	32
Run the post-upgrade script.....	32
Upgrade the Data Steward Studio App.....	32

Installation Overview

Data Steward Studio Installation Steps

You must install the Data Plane Platform, Data Plane Profiler Agent, and Data Steward Studio app in the specified order.

To install Data Steward Studio, perform the following steps:

1. Get your software.
 - a. Download the DataPlane and Data Steward Studio installation artifacts.
 - b. Set up a local repository.
 - c. Create the repository configuration file.
2. Set up DataPlane Platform.
 - a. Install or upgrade to the supported version of Data Plane Platform.
 - b. Configure DataPlane Platform.
3. Prepare your cluster for use with DSS.
 - a. Install or upgrade Ambari.
 - b. Install or upgrade HDP to the supported version.
 - c. Set up security on your cluster.
 - (Required) Set up Ambari.
 - (Required) Configure Knox SSO topologies.
 - (Required) Install Ranger and set up permissions.
 - (Required) Set up Kerberos for your cluster.
4. Install Data Plane Profiler Agent on the HDP Cluster.
 - a. Configure an external database for DataPlane Profiler Agent.
 - b. Install the DataPlane Profiler Agent using the DP Profiler Agent management pack.
 - c. Update the Knox topologies. Make sure the topologies are already created as specified in the *DataPlane Installation* instructions.
5. Install Data Steward Studio app.
 - a. Configure an external database for Data Steward Studio app.
 - b. Install the Data Steward Studio app.
 - c. Enable clusters for DSS in DataPlane Platform.

You are strongly encouraged to read completely through this entire document before starting the installation process, so that you understand the interdependencies and order of the steps.

Supported Configurations for DSS Installation

Make sure the DSS host and the clusters on which you install the DataPlane Profiler Agent meet the configuration requirements for DSS.

Requirements for the DSS host

The DSS application is installed on the same host as Data Plane Platform and has no requirements beyond what is required by Data Plane Platform.

Cross-version support for DSS application and engine

Table 1: Support by engine or app version

Engine or App Version	Supports...
DP Profiler Agent 1.4	HDP 2.6.5, HDP 3.0.0, HDP 3.1.0
DP 1.2.2 and DSS 1.4 UI application	DP Profiler Agent 1.4

Requirements for clusters used with Dataplane Profiler Agent

The clusters on which you install the Dataplane Profiler Agent must meet the requirements identified in the following sections.

Table 2: Version requirements for clusters used with DSS

Item	Versions
HDP versions	2.6.5, 3.0.0, 3.1.0
Ambari versions	2.6.2.0, 2.7.0

You can find the most current information about your product's 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>

HDP Apache Component Requirements for DSS

The following additional Apache components are required on your clusters for DSS support:

Component	Purpose	Comments
Atlas	Required for metadata search and discovery	DataPlane Profiler Agent only works with file-based authentication of Atlas.
Hive	Only Hive assets are currently supported in DSS for management	
Spark2	Required for running profiler jobs	
Livy2	Required for submitting profiler jobs to the cluster	
Knox	Required for authentication, federation, and proxying	Knox must be enabled on clusters before you can register the clusters with DPS.
Ranger	Required for looking at security policies and mining audit information	

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?
21900	Port for Dataplane Profiler service on hosts.	Accessibility is required from all clusters.	Yes
8080	Ambari server host		Yes

Default Port Number	Purpose	Comments	Required to be open?
6080	Ranger Port		Yes
8443	Knox Port		Yes
21000	Atlas Port		Yes
8999	Livy2 Port		Yes

Obtaining necessary DSS software

DSS Installation Artifacts

Before starting the installation, you must download the DSS repository tarballs and management pack from the Hortonworks Customer Portal following the instructions provided as part of the product procurement process. For DSS, you need to get an RPM tarball for the DSS application, and an RPM tarball and an Ambari management pack for the DP Profiler Agent.

The general steps you take to access the DSS installation artifacts are as follows:

1. Log into the Hortonworks Customer Portal. Access instructions are provided as part of your product procurement process.
2. Click the Data Steward Studio link at the bottom of the portal landing page to access the DSS Download Page.
3. Download the following artifacts appropriate to your operating system:
 - Repository Tar Ball for the DSS App
 - Repository Tar Ball for the DP Profiler Agent
 - Ambari Management Pack Tar Ball for the DP Profiler Agent

DSS Artifact Name	Artifact Type	How is it Installed
DSS App	RPM tarball that user needs to create a local repository from	<ul style="list-style-type: none"> • You must create a local repository • From the local repository, do a yum install. • One tarball per OS, and in this case, RHEL/CentOS/OEL 7 to match DataPlane Platform.
DP Profiler Agent	<p>Two pieces:</p> <ol style="list-style-type: none"> 1. RPM tarball from which you can create a local repository 2. Ambari Management Pack tar ball <p>Need #1 and #2 per OS that you will support for clusters.</p>	Ambari management pack scripts



Important: The MPack package for the DataPlane profiler agent includes MPack files for both the HDP 2.6.5 and HDP 3.0 versions. You need to unzip the package and identify the MPack to use depending on your version of HDP. You can identify the MPack from the name of the file. For example, the HDP 3.x MPack will have the string hdp3 in its name and the HDP 2.6.x MPack will have the string hdp2 in its name. Make sure you use the right MPack corresponding to your HDP version for installation. The MPack package also includes the pre-upgrade and post-upgrade scripts. Follow the upgrade instructions and use these scripts if you plan to upgrade from an earlier version.

Set Up a Local Repository

Setting up a local repository involves moving the tarball to the selected mirror server and extracting the tarball to create the repository.

Before you begin

Ensure that you have downloaded the required tarballs from the customer portal, following the instructions provided as part of the product procurement process.

You must have completed the preparatory tasks before setting up a repository.

Procedure

1. Copy the repository tarballs 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 tarballs to the current directory and expand each of the repository tarballs that you downloaded. Replace <file-name> 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 DSS/centos7. These directories contain the repositories.
Expanding the tarballs takes several seconds.
2. Confirm that you can browse to the newly created local repositories by using the base URLs:

```
http://<webserver-host-name>/<repo-name>/<OS>/<service-version-X>
```

 - <webserver-host-name>
This is the FQDN of the web server host.
 - <repo-name>
This is composed of the abbreviated name of the repository, such as DSS.
 - <OS>
This is the operating system version.
 - <service-version-X>
This is the version number of the downloaded repository with an appended unique number.

Base URL Examples

DSS Base URL:

```
http://webserver.com:port/DSS/centos7/1.2.0.0-X
```

Be sure to record these Base URLs, because you need them when installing DSS app on the host, and installing the associated agent on the clusters.

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 repositories for DSS are now prepared for installation.

What to do next

Create the configuration file for the DSS repository.

Create the Repository Configuration File

A repository configuration file must be created for the DSS Service on the DPS host. The file is required to identify the path to the repository data, and establish 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 dss-app.repo
```

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

3. Add the following content in the repository file:

```
#VERSION_NUMBER=<downloaded-version#> [<service-name-abbreviation>]
```

This is composed of the service name abbreviation and version number (includes the build number). Example:
DSS-APP-1.1.0.0-59

```
name=<service-name-abbreviation> Version - <service-name-abbreviation>
```

```
baseurl=http://<webserver-host-name>/<directory-containing-repo>
```

<webserver-host-name> is the FQDN of the web server host that contains the repository. This is the same base URL that you used in the task to prepare the repositories.

<directory-containing-repo> is the path expanded from the tarball.

```
gpgcheck=1
```

```
gpgkey=http://<webserver-host-name>/<directory-containing-repo>/RPM-GPG-KEY/RPM-GPG-KEY-Jenkins
```

```
enabled=1
```

```
priority=1
```

Example Repository File

```
#VERSION_NUMBER=1.2.0.0-1  
[DSS-APP-1.2.0.0-59]  
name=DSS-APP Version - DSS-APP-1.2.0.0-1  
baseurl=http://<your_webserver>:port/DSS-APP/centos7/1.2.0.0  
gpgcheck=1  
gpgkey=http://<your_webserver>:port/DSS-APP/centos7/1.2.0.0/RPM-GPG-KEY/  
RPM-GPG-KEY-Jenkins  
enabled=1
```

```
priority=1
```

Installing Profiler Agent on Clusters

After installing the Data Steward Studio app, you must install the DP Profiler Agent on clusters to complete the installation of Data Steward Studio service.

Procedure

1. Make sure all the prerequisites are met.
2. Complete the pre-installation tasks for Data Plane Profiler.
3. Configure an external database.
4. Install the Data Plane Profiler Agent.
5. Configure the Ambari Data Plane Profiler Properties.
6. Set up Knox topologies.

Pre-installation tasks for DP Profiler Agent for HDP 2.6.5 version

Perform these tasks before you try to install the Data Profiler agent on the cluster.

Procedure

1. Ensure that you have downloaded the required software from the customer portal, following the instructions provided as part of the product procurement process.

DSS includes the following parts:

- a. DSS app that needs to be installed on the DataPlane host
- b. Cluster agent software that needs to be installed on every cluster that is managed by DSS. The cluster agent software consists of an MPack package and the profiler service package.



Important: The MPack package for the DataPlane profiler agent includes MPack files for both the HDP 2.6.5 and HDP 3.0 versions. You need to unzip the package and identify the MPack to use depending on your version of HDP. You can identify the MPack from the name of the file. For example, the HDP 3.x MPack will have the string hdp3 in its name and the HDP 2.6.x MPack will have the string hdp2 in its name. Make sure you use the right MPack corresponding to your HDP version for installation.

2. Ensure that the clusters are running the required version of HDP.
3. Ensure that the following HDP components are installed and configured:
 - Atlas
 - Ranger
 - Knox
 - Spark2 and Livy Server2
4. Make sure the clusters are Kerberos-enabled.
5. If you plan to sync users from LDAP into Ranger, ensure a dpprofiler user is created in LDAP and synced into Ranger.
6. Ensure that Ranger integration for HDFS and Hive is enabled.
7. Make sure that HDFS Audit logging for Ranger is enabled.
8. Make sure you install Hive client and HDFS client on the machine where you plan to install DataPlane Profiler Agent.
9. Add the following proxy users details in the custom core-site.xml file as follows:


```
hadoop.proxyuser.livy.groups=* hadoop.proxyuser.livy.hosts=* hadoop.proxyuser.knox.groups=*
hadoop.proxyuser.knox.hosts=*
```

10. Restart the services as required.

11. Make sure the resource requirements for YARN queues for a default DSS configurations are as follows:

- RAM should be greater than or equal to 24 GB.
- CPU Cores should be greater than equal to 12.

Update the YARN parameters as follows:

Set the `yarn.scheduler.capacity.maximum-am-resource-percent` parameter on `YARN > Scheduler` (let this be `x`) such that, when multiplied with the total memory in YARN, it should be greater than or equal to 8G.

The equation appears as follows:

$$(x * \text{total_memory_in_yarn}) \geq 8\text{G}$$

For example, for 16 GB it is advised to set `x` to 0.5.

All these resources must be allocated exclusively for profiler agent and profilers. It is advisable to have a separate queue.



Note: The requirements mentioned here specify the resource allocation for the installation of DSS without any performance tuning. For more information about tuning, see [Tuning Parameters for the Profiler Service](#).

Pre-installation tasks for DP Profiler Agent for HDP 3.x versions

Perform these tasks before you try to install the Data Profiler agent on the cluster.

Procedure

1. Ensure that you have downloaded the required software from the customer portal, following the instructions provided as part of the product procurement process.

DSS includes the following parts:

- a. DSS app that needs to be installed on the DataPlane host
- b. Cluster agent software that needs to be installed on every cluster that is managed by DSS. The cluster agent software consists of an MPack package and the profiler service package.



Important: The MPack package for the DataPlane profiler agent includes MPack files for both the HDP 2.6.5 and HDP 3.0 versions. You need to unzip the package and identify the MPack to use depending on your version of HDP. You can identify the MPack from the name of the file. For example, the HDP 3.x MPack will have the string `hdp3` in its name and the HDP 2.6.x MPack will have the string `hdp2` in its name. Make sure you use the right MPack corresponding to your HDP version for installation.

2. Ensure that the clusters are running the required version of HDP.

3. Ensure that the following HDP components are installed and configured:

- Atlas
- Ranger
- Knox
- Spark2 with Livy for Spark2 and Spark Thrift Server for Spark2

4. Make sure the clusters are Kerberos-enabled.

5. Ensure that Hive Interactive is enabled.

6. If you plan to sync users from LDAP into Ranger, ensure a `dpprofiler` user is created in LDAP and synced into Ranger.

7. Ensure that Ranger integration for HDFS and Hive is enabled.

8. Make sure that HDFS Audit logging for Ranger is enabled.
9. Make sure you install Hive client and HDFS client on the machine where you plan to install DataPlane Profiler Agent.
10. Restart the services as required.
11. Make sure the resource requirements for YARN queues for a default DSS configurations are as follows:
 - RAM should be greater than or equal to 24 GB.
 - CPU Cores should be greater than equal to 12.

Update the YARN parameters as follows:

Set the `yarn.scheduler.capacity.maximum-am-resource-percent` parameter on YARN > Scheduler (let this be x) such that, when multiplied with the total memory in YARN, it should be greater than or equal to 8G.

The equation appears as follows:

$$(x * \text{total_memory_in_yarn}) \geq 8G$$

For example, for 16 GB it is advised to set x to 0.5.

All these resources must be allocated exclusively for profiler agent and profilers. It is advisable to have a separate queue.



Note: The requirements mentioned here specify the resource allocation for the installation of DSS without any performance tuning. For more information about tuning, see Tuning Parameters for the Profiler Service.

12. Make sure a stable Hive LLAP instance is available with the following minimal requirements.

Considering the following parameters:

- a= Average number of executor for sensitive/tablestats profilers
- b= Average RAM per executor for sensitive/tablestats profilers
- c= Average RAM per application master for sensitive/tablestats profilers
- y= RAM available in yarn for dpprofilers queue

The following formula will determine the minimal requirements:

$$x = y / (c + a * b)$$

LLAP will have x more jobs accessing data in Hive through LLAP with each having a parallelism.

Update Service Specific Properties for HDP 3.X versions

Make sure you update the properties required for services such as Hive and Spark.

Procedure

1. Add the following proxy users details in the custom `core-site.xml` file as follows:


```
hadoop.proxyuser.livy.groups=* hadoop.proxyuser.livy.hosts=* hadoop.proxyuser.knox.groups=*
hadoop.proxyuser.knox.hosts=* hadoop.proxyuser.hive.hosts=*
```
2. As a part of the Spark installation, update the following properties to `spark2-defaults` from Ambari UI:
 - a) `spark.sql.hive.hiveserver2.jdbc.url` - From Ambari, select **Hive** > **Summary**. Get and use the value of this property: `HIVESERVER2 INTERACTIVE JDBC URL`.
 - b) `spark.datasource.hive.warehouse.metastoreUri` - From **Hive** > **General**, get and use the value of this property: `hive.metastore.uris`
 - c) `spark.datasource.hive.warehouse.load.staging.dir` - Set this value as `/tmp`
 - d) `spark.hadoop.hive.llap.daemon.service.hosts` - From **Advanced hive-interactive-site**, get and use the value of this property: `hive.llap.daemon.service.hosts`
 - e) `spark.hadoop.hive.zookeeper.quorum` - From **Advanced hive-site**, get and use the value of this property: `hive.zookeeper.quorum`

- f) spark.sql.hive.hiveserver2.jdbc.url.principal - From **Advanced hive-site**, get and use the value of this property: hive.server2.authentication.kerberos.principal
 - g) spark.security.credentials.hiveserver2.enabled - Set this value as true.
3. Add the following property to Custom livy2-conf:

```
livy.file.local-dir-whitelist : /usr/hdp/current/hive_warehouse_connector/
```

4. To allow the dpprofiler user to submit apps to default queue in YARN Capacity Scheduler section from Ambari UI, update the property as follows:

```
yarn.scheduler.capacity.root.acl_submit_applications=dpprofiler,yarn,yarn-ats,hdfs
```

5. For Kerberos enabled clusters, set the Hadoop HTTP authentication mode to kerberos. Set the following property in HDFS > Configs > core-site:

```
hadoop.http.authentication.type=kerberos
```

6. Restart all the services as suggested by Ambari. Make sure that all the services are up and running after the restart.

Configure an external database for DataPlane Profiler Agent

You must configure an external database and add the profileragent database user to the database.

About this task



Important: Make sure the external database for DataPlane Profiler Agent has a minimum of dedicated 50 GB disk space and 8 cores.

Configure MySQL external database

You must configure an MySQL database and add the profileragent database user to the database.

About this task

MySQL is supported only if Ambari is installed to use on MySQL.

Procedure

1. Log in to your MySQL database client.
2. Create a database user. The default value is profileragent.
3. Create a database name. The default value is profileragent.
4. Grant the user profileragent all rights on the database profileragent.

Configure Postgres external database

You must configure a Postgres database and add the profileragent database user to the database.

Procedure

1. Log in to postgres shell using admin user like postgres:
psql -U postgres
2. Create profileragent database and user and grant all privileges:
CREATE DATABASE \$profileragentdb;
CREATE USER \$profileragentuser WITH PASSWORD '\$password';

```
GRANT ALL PRIVILEGES ON DATABASE $profileragentdb TO $profileragentuser;
```

The default value for \$profileragentdb and \$profileragentuser is profileragent.

3. Add \$profileragentuser to pg_hba.conf to have access from profiler agent host.

Grant Permissions in Ranger and YARN

The DP Profiler user needs permissions to read entities for the profilers to function. Such a policy should be created via Ranger.

About this task

The dpprofiler user needs access to the following:

- Read and list tables from the Hive metastore.
- Read and write types, entities, and classifications in Atlas.
- Run jobs in YARN against a configured queue. If you are using the default installation, make sure that these permissions are granted to the default queue.

Install the Data Plane Profiler Agent

DSS requires that the DP Profiler Agent be installed on all clusters. The Profiler is installed on the 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 DSS.

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. The repository tarballs for the Data Plane Profiler agent are different from the DSS app repository tarballs.

Procedure

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

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

2. Install the Data Plane Profiler MPack by running the following command, replacing <mpack-file-name> with the name of the MPack.

```
ambari-server install-mpack --mpack <mpack-file-name> --verbose
```

3. Restart the Ambari server.

```
ambari-server restart
```

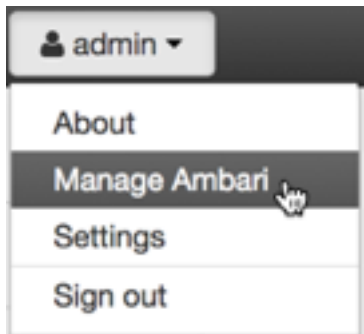
4. Launch Ambari in a browser and log in.
http://<ambari-server-host>:8080

Default credentials are:

Username: admin

Password: admin

5. Click **Admin>Manage Ambari**.

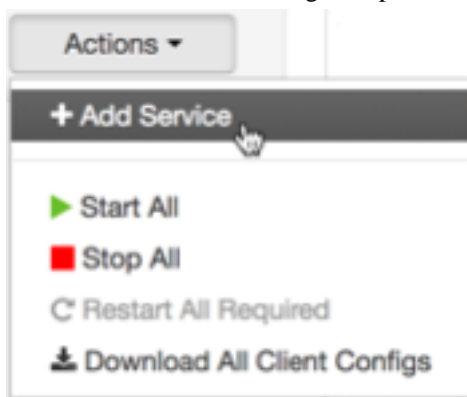


6. Click **Versions**, and then do the following on the Versions page:
- Click the HDP version in the **Name** column.
 - Change the **Base URL** path for the DSS service to point to the local repository, for example:

```
http://webserver.com/DSS/centos7/1.2.0.0-X
```

URLs shown are for example purposes only. Actual URLs might be different.

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



The Add Service Wizard displays.

9. On the **Choose Services** page of the Wizard, select the Dataplane Profiler service to install in Ambari, and then follow the on-screen instructions.

Other required services are automatically selected.

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

This adds other required services.

11. On the **Assign Masters** page, you can choose the default settings.

12. On the **Customize Services** page, fill out the database details and other required fields that are highlighted.

Make sure to enter the credentials that you set while configuring the external database. Change the username profileragent to the values set in the external database.



Note: Make sure to add the database driver to the machine based on the external database that you configured.

13. Complete the remaining installation wizard steps and exit the wizard.

14. Ensure that all components required for your DataPlane Platform have started successfully.



Note: As part of the installation verification screen, an earlier version of DSS repositories might appear in the labels. You can ignore the version number in the version number and proceed further.

15. Enable Knox SSO for DP Profiler Agent.

- a. Set `dpprofiler.sso.knox.enabled` to `true` in Advanced `dpprofiler-env` section in Ambari DP Profiler Configs.
- b. Run the following CLI command to export the Knox certificate:

```
JAVA_HOME/bin/keytool -export -alias gateway-identity -rfc -file knox-
pub-key.cert -keystore /usr/hdp/current/knox-server/data/security/
keystores/gateway.jks
```

When prompted, enter the Knox master password.

- c. After generating the certificate, paste the contents of the certificate in the `dpprofiler.sso.knox.public.key` field under **Advanced dpprofiler-env properties** of DP Profiler Configs in Ambari.

16. Open the quick link of the profiler for service verification.

17. Add `/profilers` to the quick link URL.

If the quick link is `xyz:21900`, change it to `xyz:21900/profilers`.



Note: For non-Kerberized clusters, this request returns the list of all registered profilers. For kerberos-enabled clusters where Knox is not enabled for DP Profiler Agent, you will see an HTTP-401 response which is expected.

18. After installing the profiler agent using **Add Service Wizard** in Ambari, the NodeManager hosts do not have the `dpprofiler` user. For Ambari to automatically create these users, restart all NodeManagers by going to **Services->YARN->Restart NodeManagers** (NodeManagers can be restarted in a rolling fashion - Ambari UI shows restart batching options)



Note: During DP Profiler Agent installation, two new Atlas types - `dss_hive_column_profile_data` and `dss_hive_table_profile_data`, are registered. These types contain attributes to store metrics computed by DSS profilers. In addition, existing Atlas types `hive_table` and `hive_column` are updated to add an additional attribute `profileData`. For `hive_table` type, attribute `profileData` is a reference to `dss_hive_table_profile_data` and for type `hive_column`, attribute `profileData` is a reference to `dss_hive_column_profile_data`.



Important: As part of installation of DataPlane Profiler Agent on HDP 3.x versions, make sure you enter the details of DP Profiler extra JARs when prompted as part of the advanced `dpprofiler-env` properties. To get the value of the version of the JARs, log in to the Livy machine and navigate to this location:

```
///usr/hdp/current/hive-warehouse-connector/hive-warehouse-connector-
assembly-<version>.jar
```

Extract the details of the exact location with specific version details and paste in the Ambari section. Enter the value of the property as follows:

```
file:///usr/hdp/current/hive-warehouse-connector/hive-warehouse-
connector-assembly-<version>.jar
```

19. If TDE zones are set up in the cluster and if any of the following locations fall within the TDE zones, the `dpprofiler` user must have `Decrypt_EEK` access to the Key/Keys used to encrypt that zone.

- `/user/dpprofiler`
- `/ranger/audit/hiveServer2`
- `/apps/dpprofiler`
- all locations of Hive tables

20. In the Advanced `dpprofiler-config` section of the DP Profiler service in Ambari, make sure you enter the Zookeeper Connection String details.




Ambari Dataplane Profiler Configs

From **Ambari > Dataplane Profiler > Configs**, you can view or update your database or advanced configurations.

Dataplane Profiler Database Configs

From **Ambari** > **Dataplane Profiler** > **Configs** > **Database**, you can view or update the DataPlane Profiler Database configurations.

Table 3: Database configs

Value	Description	Example
DP Profiler Database	Database type or flavor used for DSS profiler.	h2 mysql postgres  Note: Make sure that the database type is entered in lower case, such as h2, mysql, or postgres.
Database Username	A Database user needs to be created in the MySQL or Postgres DB that the profiler service would use to connect to the DB. This is name of that database user.	profileragent
Database Name	Name must be "profileragent".  Important: Do not modify.	profileragent
Database URL	The URL of DP profiler database.	H2: jdbc:h2:/var/lib/profiler_agent/h2/profileragent;DATABASE_TO_UPPER=false;DB_CLOSE_DELAY=1000 MySQL: jdbc:mysql://hostname:3306/profileragent?autoreconnect=true POSTGRES: jdbc:postgresql://hostname:5432/profileragent  Note: Make sure that the database name within the URL is in lower case, such as h2, mysql, or postgres.
Database Host	Database host name for Profiler Agent server	<hostname>
Database password	The password for your DP database.	<your_password>



Note: On HDP 3.x versions, Profiler Agent service on Ambari UI does not have a separate tab for configuring database. All database configuration is available as part of Dataplane Profiler Database Configs.

Dataplane Profiler Advanced Configs

From **Ambari** > **Dataplane Profiler** > **Configs** > **Advanced**, you can view or update the DataPlane Profiler advanced configurations.

Table 4: Advanced dpprofiler-config

Value	Description	Example
Dependent Cluster Configurations	Provides various cluster configurations, including: atlasUrl rangerAuditDir metastoreUrl metastoreKeytab metastorePrincipal	atlasUrl=application-properties/ atlas.rest.address;rangerAuditDir=ranger-env/ xasecure.audit.destination.hdfs.dir;metastoreUrl=hive-site/ hive.metastore.uris;metastoreKeytab=hive-site/ hive.metastore.kerberos.keytab.file;metastorePrincipal=hive-site/hive.metastore.kerberos.principal

Value	Description	Example
Additional Cluster Configurations	Additional configuration items of services in the cluster that can be set for use by profilers.	
Profilers local home directory	Local directory for the profilers.	/usr/dss/current/profilers
Profilers shared results directory	The HDFS directory where DSS Profilers will store their metrics output. Ensure the dpprofiler user has full access to this directory.	/user/dpprofiler/dwh
Profilers shared binaries directory	HDFS directory for the profilers.	/apps/dpprofiler/profilers
SPNEGO Cookie Name	Cookie name that is returned to the client after successful SPNEGO authentication.	dpprofiler.spnego.cookie
SPNEGO Signature Secret	Secret for verifying and signing the generated cookie after successful authentication	***some***secret**
Maximum assets submitted per profiler job	Maximum number of assets to be submitted in one profiler job.	50
Maximum number of concurrent profiler jobs	Number of profiler jobs active at a point in time. This is per profiler.	2
Job scan interval	Time in seconds after which the profiler looks for an asset in the queue and schedules the job if the queue is not empty.	30
Maximum number of assets queued for submission	Maximum size of the profiler queue. After which it rejects any new asset submission request.	500
Zookeeper Base Path	Make sure you enter the Zookeeper base path details. This is the path used by DP Profiler Agent for locking functions. The dpprofiler user must have access to this Zookeeper path. Ensure necessary authorization is provided to the dpprofiler user.	/profileragent/stores
Zookeeper Connection String	Enter the connection string of clusters Zookeeper during the installation.	host1:2181,host2:2181

Table 5: Advanced dpprofiler-env

Value	Description	Example
Profiler service local configuration directory	Configuration files directory.	/etc/profiler_agent/conf
Profiler service local data directory	Data directory. If using h2, data is stored here.	/var/lib/profiler_agent
Profiler service HTTP Port	Port where profiler agent runs.	21900
Profiler Knox SSO Enabled	Enable this to use Knox SSO for Profiler	

Value	Description	Example
Profiler Knox SSO Public Key	Knox SSO Public Certificate	<p>Run the following CLI command to export the Knox certificate:</p> <pre>JAVA_HOME/bin/keytool - export -alias gateway- identity -rfc -file knox-pub-key.cert -keystore /usr/ hdp/current/knox- server/data/security/ keystores/gateway.jks</pre> <p>When prompted, enter the Knox master password. After generating the certificate, paste the contents of the certificate in this field.</p>
Profiler Service Keytab	Profiler agent keytab location.	/etc/security/keytabs/ dpprofiler.kerberos.keytab
Profiler Service Principal	Profiler agent kerberos principal.	dpprofiler\${principalSuffix}@REALM.COM principalSuffix is a random string which is generated by Ambari for a cluster. This string is used to uniquely identify services on a cluster in case of multiple clusters being managed by single KDC
Number of retries for refresh of Kerberos ticket	Maximum number of retries allowed for refreshing the Kerberos ticket	5
Profiler service local log directory	Log Directory	/var/log/profiler_agent
Profiler service local PID file directory	Pid Directory	/var/run/profiler_agent
Profiler Service SPNEGO Kerberos Keytab	SPNEGO keytab location.	/etc/security/keytabs/spnego.service.keytab

Value	Description	Example
Profiler Service SPNEGO Kerberos Principal	SPNEGO Kerberos principal.	HTTP/\${FQDN}@REALM.COM FQDN - fully qualified domain name of the machine
Profiler Service Logging configuration	Content for logback.xml.	<pre> <configuration> <conversionRule conversionWord="coloredLevel" converterClass="play.api.libs.logback.ColoredLevel" /> <appender name="FILE" class="ch.qos.logback.core.FileAppender" /> <file>{{dpprofiler_log_dir}}/ application.log</file> <encoder> <pattern>%date [%level] from %logger in %thread - %message%n %xException</pattern> </encoder> </appender> <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender" /> <encoder> <pattern>%coloredLevel %logger{15} - %message %n%xException{10}</ pattern> </encoder> </appender> <appender name="ASYNCFILE" class="ch.qos.logback.classic.AsyncFileAppender" /> <appender-ref ref="FILE" /> </appender> <appender name="ASYNCSTDOUT" class="ch.qos.logback.classic.AsyncConsoleAppender" /> <appender-ref ref="STDOUT" /> </appender> <logger name="play" level="INFO" /> <logger name="application" level="DEBUG" /> <!-- Off these ones as they are annoying, and anyway we manage configuration ourselves --> <logger name="com.avaje.ebean.config.PropertyRetriever" level="OFF" /> <logger name="com.avaje.ebeaninternal.server.core.DefaultServer" level="OFF" /> <logger name="com.avaje.ebeaninternal.server.default.DefaultPersistenceContextImpl" level="OFF" /> </pre>
	18	<pre> name="com.avaje.ebeaninternal.server.default.DefaultPersistenceContextImpl" level="OFF" /> <logger name="com.avaje.ebeaninternal.server.default.DefaultPersistenceContextImpl" level="OFF" /> </pre>

Value	Description	Example
DP Profiler Extra JARs	For HDP 3.x installations, you must enter the location details of DP Profiler extra JAR files.	To get the value of the location of the JARs, log in to the Livy machine and navigate to this location and get the exact value of the location with the version details: <pre>file:///usr/hdp/ current/hive-warehouse- connector/hive- warehouse-connector- assembly-<version>.jar</pre>

Table 6: Advanced dpprofiler-livy-config

Value	Description	Example
Read session driver core count	Number of cores to use for the driver session for the read process.	1
Read session driver memory size	Amount of memory to use for the driver process for the read session.	1g
Read session executor core count	Number of cores to use for each executor for read session.	1
Read session executor memory size	Amount of memory to use per executor for read session.	1g
Read session heartbeat timeout	Timeout in seconds to which read session will be orphaned.	172800
Read session name	Name of the read session.	dpprofiler-read
Read session executor count	Number of executors to launch for the read session.	2
Read session queue name	Name of the YARN queue for the read sessions.	default
Read session timeout	Specifies timeouts for read requests using interactive session.	90
Write session driver core count	Number of cores to use for the driver session for the write process.	1
Write session driver memory size	Amount of memory to use for the driver process for the write session.	1g
Write session executor core count	Number of cores to use for each executor for write session.	1
Write session executor memory size	Amount of memory to use per executor for the write session.	1g
Write session heartbeat timeout	Timeout in seconds to which write session will be orphaned.	172800
Write session name	Name of the write session.	dpprofiler-write
Write session executor count	Number of executors to launch for the write session.	2
Write session queue name	Name of the YARN queue for the write sessions.	default
Write session timeout	Specifies timeouts for write requests using interactive session.	90

Value	Description	Example
Session Lifetime in Minutes	Session lifetime in minutes after its creation before it will be swapped.	2880 For smaller clusters, it is recommended to set this to a smaller value like 240.
Session Lifetime in Requests	Maximum number of requests a session can process before it will swapped.	500
Session creation retry count	Maximum number of attempts for session creation. The session will be declared dead after these many retries.	20

Table 7: Custom dpprofiler-config


Value	Description	Example
dpprofiler.user	User for Profiler Agent  Important: Do not modify.	dpprofiler

Table 8: Custom dpprofiler-env

Value	Description	Example

Table 9: Custom dpprofiler-livy-config

Value	Description	Example

Tuning Parameters for the Profiler Service

You can update the configuration settings to use minimal resources for the Profiler service if you have limited resources in your queue.

About this task



Note: The instructions specified here are not the minimal requirements. When you have limited resources in your queue, you can tune your settings to ensure optimum performance.

If you have resource limitations with your queue, make sure the minimum resource requirements are as follows:

- RAM should be greater than or equal to 16 GB.
- CPU Cores should be greater than equal to 8.

Procedure

1. Set the number of executors for Livy interactive sessions to 1.
 - a) Log in to Ambari.
 - b) From Ambari, select **Dataplane Profiler** and click **Configs**.
 - c) In the Advanced dpprofiler-livy-config section, set the following properties:
 - Number Of Executors For Write Session =1
 - Number Of Executors For Read Session=1
2. Set the number of executors for profiler jobs to 1.
 - a) Log in to the Data Steward Studio app.
 - b) Go to profiler configuration and click **edit**.

- c) Select a profiler. The Profile Configuration tab opens on the right.
 - d) Update the details of the number of executors in the Advanced Options section. Set the number of executors to 1.
3. Update the YARN parameters as follows:

Set the `yarn.scheduler.capacity.maximum-am-resource-percent` parameter on YARN > Scheduler (let this be `x`) such that

$$(x * \text{total_memory_in_yarn}) \geq 8G$$

For example, for 16 GB it is advised to set `x` to 0.5.

All these resources must be allocated exclusively for profiler agent and profilers. It is advisable to have a separate queue.

Update Knox Topologies

After configuring the external database, you must update the Knox topologies for DP Profiler.

About this task

To access the Profiler agent behind Knox gateway, update the `dp-proxy.xml` topology file.

Procedure

1. On each cluster, make sure the `dp-proxy.xml` and `token.xml` are set up in the Knox topologies folder.

- a) Navigate to the topologies folder using this command:

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

- b) Verify that the `dp-proxy.xml` and `token.xml` files are in this folder.

For more information about setting up these files, see [Knox Configuration in DataPlane Installation Guide](#).

2. Add the Profiler Agent Server address in the the `dp-proxy.xml` file as follows:

```
<service>
<role>PROFILER-AGENT</role>
<url>URI of the server address</url>
</service>
```

DP Proxy Settings for Atlas HA Configuration

To support HA configuration, the `dp-proxy.xml` must be set up with additional sections that let Knox know how to work with HA-enabled services.

An example `dp-proxy.xml` appears as follows.

```
<topology>
  <gateway>
    <provider>
      <role>federation</role>
      <name>SSOCookieProvider</name>
      <enabled>>true</enabled>
      <param>
        <name>sso.authentication.provider.url</name>
        <value>Hostname URL</value>
      </param>
    </provider>
  </gateway>
</topology>
```

```

        </provider>
    </provider>
        <role>identity-assertion</role>
        <name>Default</name>
        <enabled>true</enabled>
    </provider>
    <provider>
        <role>ha</role>
        <name>HaProvider</name>
        <enabled>true</enabled>
        <param>
            <name>WEBHDFS</name>

<value>maxFailoverAttempts=3;failoverSleep=1000;enabled=true</value>
        </param>
        <param>
            <name>WEBHCAT</name>

<value>maxFailoverAttempts=3;failoverSleep=1000;enabled=true</value>
        </param>
        <param>
            <name>RANGER</name>

<value>maxFailoverAttempts=3;failoverSleep=1000;enabled=true</value>
        </param>
        <param>
            <name>RESOURCEMANAGER</name>

<value>maxFailoverAttempts=3;failoverSleep=1000;enabled=true</value>
        </param>
        <param>
            <name>WEBHBASE</name>

<value>maxFailoverAttempts=3;failoverSleep=1000;enabled=true</value>
        </param>
        <param>
            <name>HIVE</name>

<value>maxFailoverAttempts=3;failoverSleep=1000;enabled=true</value>
        </param>
        <param>
            <name>OOZIE</name>

<value>maxFailoverAttempts=3;failoverSleep=1000;enabled=true</value>
        </param>
        <param>
            <name>ATLAS</name>

<value>maxFailoverAttempts=3;failoverSleep=1000;enabled=true;zookeeperEnsemble=zookeep
value>
        </param>
        <param>
            <name>ATLAS-API</name>

<value>maxFailoverAttempts=3;failoverSleep=1000;enabled=true;zookeeperEnsemble=zookeep
value>
        </param>
    </provider>
</gateway>
<service>
    <role>WEBHDFS</role>
    <url>URL to access Web HDFS</url>
    <url>URL to access Web HDFS</url>
</service>

```

```
<service>
  <role>WEBHCAT</role>
  <url>URL to access Web HCAT</url>
  <url>URL to access Web HCAT</url>
</service>
<service>
  <role>AMBARI</role>
  <url>URL to access Ambari</url>
</service>
<service>
  <role>RANGER</role>
  <url>Ranger URL</url>
  <url>Ranger URL</url>
</service>
<service>
  <role>RANGERUI</role>
  <url>Ranger UI URL</url>
  <url>Ranger UI URL</url>
</service>
<service>
  <role>ATLAS</role>
</service>
<service>
  <role>ATLAS-API</role>
</service>
<service>
  <role>OOZIE</role>
  <url>none</url>
  <url>none</url>
</service>
<service>
  <role>WEBHBASE</role>
  <url>Web HBase URL</url>
  <url>Web HBase URL</url>
</service>
<service>
  <role>HIVE</role>
  <url>Hive URL</url>
  <url>Hive URL</url>
</service>
<service>
  <role>RESOURCEMANAGER</role>
  <url>Resource Manager URL</url>
  <url>Resource Manager URL</url>
</service>
<service>
  <role>BEACON</role>
  <url>none</url>
</service>
<service>
  <role>PROFILER-AGENT</role>
  <url>Profiler Agent URL</url>
</service>
</topology>
```

Installing the Data Steward Studio App

After installing the DPS Platform, you must install the Data Steward Studio app.

About this task

Data Steward Studio app must be installed on the same host as DPS Platform. You can install one DPS service or any combination of DPS services with the DPS Platform.

Procedure

1. Set up a local repository.
2. Create the repository configuration file.
3. Install the Data Steward Studio app.
4. Enable the Data Steward Studio in the Data Plane Service.
5. Add users and assign roles for DSS app.
6. Add data lakes to the Data Plane service.

Set Up a Local Repository

Setting up a local repository involves moving the tarball to the selected mirror server and extracting the tarball to create the repository.

Before you begin

Ensure that you have downloaded the required tarballs from the customer portal, following the instructions provided as part of the product procurement process.

You must have completed the preparatory tasks before setting up a repository.

Procedure

1. Copy the repository tarballs 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 tarballs to the current directory and expand each of the repository tarballs that you downloaded.
Replace <file-name> 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 DSS/centos7. These directories contain the repositories.
Expanding the tarballs takes several seconds.
2. Confirm that you can browse to the newly created local repositories by using the base URLs:
`http://<webserver-host-name>/<repo-name>/<OS>/<service-version-X>`
 - <webserver-host-name>
This is the FQDN of the web server host.
 - <repo-name>
This is composed of the abbreviated name of the repository, such as DSS.
 - <OS>
This is the operating system version.
 - <service-version-X>
This is the version number of the downloaded repository with an appended unique number.

Base URL Examples

DSS Base URL:

```
http://webserver.com:port/DSS/centos7/1.2.0.0-X
```

Be sure to record these Base URLs, because you need them when installing DSS app on the host, and installing the associated agent on the clusters.

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 repositories for DSS are now prepared for installation.

What to do next

Create the configuration file for the DSS repository.

Create the Repository Configuration File

A repository configuration file must be created for the DSS Service on the DPS host. The file is required to identify the path to the repository data, and establish 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 dss-app.repo
```

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

3. Add the following content in the repository file:

```
#VERSION_NUMBER=<downloaded-version#> [<service-name-abbreviation>]
```

This is composed of the service name abbreviation and version number (includes the build number). Example:

```
DSS-APP-1.1.0.0-59
```

```
name=<service-name-abbreviation> Version - <service-name-abbreviation>
```

```
baseurl=http://<webserver-host-name>/<directory-containing-repo>
```

`<webserver-host-name>` is the FQDN of the web server host that contains the repository. This is the same base URL that you used in the task to prepare the repositories.

<directory-containing-repo> is the path expanded from the tarball.

```
gpgcheck=1
```

```
gpgkey=http://<webserver-host-name>/<directory-containing-repo>/RPM-GPG-KEY/RPM-GPG-KEY-Jenkins
```

```
enabled=1
```

```
priority=1
```

Example Repository File

```
#VERSION_NUMBER=1.2.0.0-1
[DSS-APP-1.2.0.0-59]
name=DSS-APP Version - DSS-APP-1.2.0.0-1
baseurl=http://<your_webserver>:port/DSS-APP/centos7/1.2.0.0
gpgcheck=1
gpgkey=http://<your_webserver>:port/DSS-APP/centos7/1.2.0.0/RPM-GPG-KEY/RPM-GPG-KEY-Jenkins
enabled=1
priority=1
```

Configure an external database for Data Steward Studio App

Although DSS includes an embedded PostgreSQL database, the embedded database is intended for nonproduction use. It is strongly recommended to use an external database for production environments. After installing the database following the instructions provided with the database software, you must set up the database for use with DSS.

About this task

- PostgreSQL database is supported in this version.
- You will be configuring an external database or your own SSL certificate.
- Refer to the *Data Steward Studio Support Matrix* for requirements and supported databases.

Be sure to have the database URI, username, and password available.

Before you begin

- You need root user access on the DP Host to perform this task.
- You must have the proper role to perform this task.
- The PostgreSQL database must have been installed and properly configured for remote access.
- A database called dss must have been created.
- A database user must have been created and assigned permissions for the dss database.

Procedure

1. Open the config.env.sh file for editing.

```
vi /usr/dss-app/current/apps/dss/bin/config.env.sh
```

2. Modify the DB Configs settings to add the appropriate connection information.

```
USE_EXTERNAL_DB="yes"
DATABASE_URI="jdbc:postgresql://<host_name>:5432/dss"
DATABASE_USER="<user_name>"
```

```
DATABASE_PASS=" <password> "
```

Results

Your external database is now set up so that you can configure it for Data Steward Studio during the DSS app installation.

Install the Data Steward Studio Service App

Follow the instructions to install the Data Steward Studio Service app.

Before you begin

You must have successfully installed DPS Platform and DPS is running.

Procedure

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

```
sudo su
```

2. Install the RPMs for the DSS service application.

```
yum install dss-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/dss-app.repo` on the host.

3. Navigate to the directory containing the installation scripts for the DSS service, for example:
`cd /usr/dss-app/current/apps/dss/bin`
4. Load the DSS Docker images and initialize the environment.
`./dssdeploy.sh init`

It prompts for the master password that was used for initializing the Data Plane platform. Make sure you enter the same master password.

Loading the images might take a while.



Note:

If you run into errors while deploying, you must destroy the deployment using `./dssdeploy.sh destroy` command and re-install the app. To check the logs of the dss-app container, you can use the command `./dssdeploy.sh logs`.

5. Verify that the container you installed is running.
`./dssdeploy.sh ps`

Make sure that the container with the name `dss-app` is running.

Enable the Data Steward Studio in the Data Plane Platform

After installing the Data Steward Studio app, you must enable in the Data Plane Platform.

Procedure

1. Log in to the DP Platform as a DataPlane Admin user.
2. From the Admin page, click Services icon on the left panel.
The Services page displays.

3. On the Services page, you can see the available service apps.
4. Click on Data Steward Studio app to view the list of available clusters.



Note: Make sure you install the required components in the clusters before you enable the cluster for DSS.

5. In the Actions column, select the cluster that you want to enable for the service and click **Enable**.

Add Users and Assign Roles for DSS App

After you set up the LDAP configuration for DPS Platform, you need to add users for the DSS app. During LDAP configuration, you added users and groups that can log in as DPS Admin. You must now assign roles to users and groups, which allow users to access the services that plug into DPS.

About this task

You must select the Data Steward role for accessing the Data Steward Studio Service. Users and groups should be assigned this role to access Data Steward Studio service. To enable the Data Steward Studio role, see Role Management section of the *Data Plane Service Administration Guide*.

Before you begin

User accounts must already exist within your corporate LDAP prior to adding the user to DPS Platform.

The DataPlane Admin role is required to perform this task.

Procedure

1. Log in to the DPS Platform.
2. Click the (Users) icon in the DPS Platform navigation pane.
3. On the Users and Groups page, click **Add User**.
4. Enter the name of the user.

With your own LDAP server, the user must already exist within your corporate LDAP. If you are using the packaged LDAP, enter one of the predefined users (guest, sam, tom). The name auto-populates as you type.



Tip:

You must click the name of the user when it displays and ensure it appears in the Username field on a dark background.

If the name appears on a white background, it means the name is not recognized and the action fails.

5. Select the Data Steward role to assign to the user:

Data Steward - Can perform all actions in the Data Steward Studio service UI, and can manage DSS-enabled clusters in DPS Platform.

6. Click **Save**.

You can log in and see Data Steward Studio service inside the DPS Platform. If Data Steward role is the only role assigned, you will be directed to the Data Steward Studio Service. If you have more roles assigned, you can select the Data Steward Studio Service in the navigation menu in the top left corner.



Note: If you assign the Data Steward role to yourself or to the group that you belong to, you must log out and log in again to verify that Data Steward Studio Service is available.

The new user displays in the list on the Users page.

Add Data Lakes to the Data Plane Service

Make sure to add data lakes to the Data Plane Service to access them in the Data Steward Studio Service.

Procedure

Register a cluster in the DPS Platform. For more information, see the *Data Plane Service Administration Guide*.

Data Steward Studio can only work with clusters that are identified as datalakes. Clusters that have Atlas and Ranger installed can be identified as datalakes. For more information, see the *Add Clusters* section in the *DPS Administration Guide*. Users with DPS Admin role can only add clusters on the DPS platform.

Upgrade Data Steward Studio

Make sure you take regular backups of the instance before you proceed with the upgrade procedure.

To upgrade Data Steward Studio, you must perform the following steps in the specified order:

1. Perform the pre-upgrade tasks.
2. Upgrade DataPlane Platform. See the DataPlane Installation for more information.
3. Upgrade DataPlane Profiler on Ambari.
4. Run the DataPlane Profiler Upgrade Script.
5. Upgrade Data Steward Studio app.

Pre-upgrade tasks

Perform the following pre-upgrade tasks before you try to upgrade the DataPlane Profiler and Data Steward Studio app.

Procedure

1. Stop the DSS App container. Log in to DataPlane host and run:
`docker stop dss-app`
2. Stop the DataPlane Profiler service. Log in to Ambari of the HDP cluster and stop the DataPlane Profiler service.
3. From the DataPlane Profiler configuration on Ambari, make a note of the following configurations:
 - DataPlane Profiler Extra JARs
 - Hiveserver Interactive JDBC URL (on HDP3)
 - Enable Knox Single Sign On for Profiler Service
 - Knox SSO Public Key
4. Make a note of the database configuration details used for the DataPlane Profiler service.
5. Delete the DataPlane Profiler service from Ambari.
6. Make sure you retain the Ranger configuration details when prompted.
7. Uncheck all the configurations, when prompted to retain the current values.
8. After stopping and deleting Profiler Agent service, log in to Profiler Agent machine and navigate to the tmp directory.

```
cd /tmp
```

9. Access the upgrade tarball `upgrade.zip` from the installation artifacts already downloaded. Unarchive the contents of downloaded upgrade archive(zip) file:

```
unzip upgrade.zip
```

Additional pre-upgrade tasks to upgrade from DSS 1.3.1 to DSS 1.5.0

Make sure you perform the following additional pre-upgrade tasks if you plan to upgrade from DSS 1.3.1 to DSS 1.5.0.

Procedure

1. Navigate to the upgrade directory to run the pre-upgrade script:

```
cd /tmp/upgrade/1.4.0
```

2. Configure the parameters as required in the `upgrade_configs.json` file. Make sure you set the `previous_dss_version` parameter appropriately.
3. Change the current user to `dpprofiler` user.

```
su dpprofiler
```

4. Run the pre-upgrade data migration script:

```
python dpprofiler_pre_upgrade.py
```

Run the DataPlane Profiler pre-upgrade script

Make sure you download and run the pre-upgrade script before upgrading DataPlane and Data Steward Studio.

Procedure

1. Navigate to the upgrade directory to run the pre-upgrade script:

```
cd upgrade/1.5.0
```

2. Configure the parameters `ranger_audit_dir` and `queue` as required in the `upgrade_configs.json` file.
3. Change the current user to `dpprofiler` user.

```
su dpprofiler
```

4. Run the pre-upgrade data migration script:

```
python dpprofiler_pre_upgrade.py
```

Upgrade DataPlane Profiler on Ambari

Upgrade DataPlane Profiler using Ambari on the same node where the old one is installed.

Procedure

1. Log in to the Ambari node and remove the DataPlane Profiler MPack.

```
ambari-server uninstall-mpack --mpack-name=dpprofiler.mpack
```

2. Log in to DataPlane Profiler node and remove directories associated with DataPlane profiler service.

```
mv /var/lib/profiler_agent /var/lib/profiler_agent.bak
mv /var/log/profiler_agent /var/log/profiler_agent.bak
mv /etc/profiler_agent /etc/profiler_agent.bak
mv /usr/dss /usr/dss.bak
yum remove audit_profiler hive_metastore_profiler profiler_agent
sensitive_info_profiler tablestats_profiler
```

If you encounter any warnings of failed file removals, you can ignore them.

3. Create a backup copy of the DataPlane Profiler warehouse directory on HDFS, as follows:

```
su dpprofiler
hdfs dfs -cp /user/dpprofiler/dwh /user/dpprofiler/dwh.bak
```

4. Log in to Ambari node and download the latest Dataplane Profiler mpack.

```
wget $MPACK_LINK
```

5. Install the new Dataplane Profiler mpack from Ambari.

```
ambari-server install-mpack --mpack=/root/dpprofiler-ambari-
mpack-1.5.0.tar.gz -verbose
ambari-server restart
```

6. Update the repo link for DSS from Ambari.
7. Update and confirm the changed details.
8. Add DataPlane Profiler Agent service from Ambari.
9. Enter the same database credentials that were configured with previous version of DataPlane Profiler.

Depending on existing permissions, the profileragent user on this node might need to be granted permissions on the profileragent database.

Perform the following steps to provide the required permissions.

- If the database is MySQL, run the following command:

```
grant all privileges on profileragent.* to 'profileragent'@'$HOST'
identified by 'profileragent';
```

- If the database is Postgres, run the following command:

```
GRANT ALL PRIVILEGES ON DATABASE "profileragent" to profileragent;
```



Important:

During DSS upgrade on HDP 3.x, enable SSO authentication by setting the **Enable Knox Single Sign On for Profiler Service** property to true. Make sure you also enter the Knox public key for the Knox SSO Public Key property.

10. During installation, when Dependent Configurations window pops up, deselect all and click **OK**.



Note: As part of the installation verification screen, an earlier version of DSS repositories might appear in the labels. You can ignore the version number in the version number and proceed further.

11. In a kerberos-secured cluster, a message may appear asking for principal and password. Enter the required information and click **Save**.

12. Once the installation is complete, log in to the Profiler Agent machine.

```
cd /tmp/upgrade/1.5.0
```

13. Configure Profiler Agent URL in file upgrade_configs.json file.

14. Run the DP Profiler post upgrade script.

```
su dpprofiler
```

```
python dpprofiler_post_upgrade.py
```

Additional post-upgrade tasks to upgrade from DSS 1.3.1 to DSS 1.5.0

Make sure you perform the following additional post-upgrade tasks if you plan to upgrade from DSS 1.3.1 to DSS 1.5.0.

Procedure

1. Navigate to the upgrade directory to run the pre-upgrade script:

```
cd /tmp/upgrade/1.4.0
```

2. Configure the parameters as required in the upgrade_configs.json file. Make you set the atlas_url, cluster_name, db_type, host, database, user and password parameters appropriately.
3. Change the current user to dpprofiler user.

```
su dpprofiler
```

4. Run the post-upgrade script:

```
python dpprofiler_post_upgrade.py
```

Run the post-upgrade script

Make sure you download and run the post upgrade script before proceeding further to upgrade the Data Steward Studio app.

Procedure

1. After the upgrade is complete, log in to DP Profiler agent machine.
2. Navigate to the upgrade directory to run the post-upgrade script:

```
cd /tmp/upgrade/1.5.0
```

3. Configure the parameters as required in the upgrade_configs.json file. Make sure you set the Profiler Agent URL parameter appropriately.
4. Change the current user to dpprofiler user.

```
su dpprofiler
```

5. Run the post-upgrade script:

```
python dpprofiler_post_upgrade.py
```

Upgrade the Data Steward Studio App

Make sure you take regular backups of the instance before your proceed with the upgrade procedure.

About this task

To upgrade from one version to another, you must make note of the existing version and the new upgrade version. You must run the upgrade command from the new upgrade version repository folder and enter the folder details of the existing version.

Procedure

1. Back up your existing Data Steward Studio repository dss-app.repo file in .repo format.
2. Download the upgrade repository tarball to the repository folder:

```
wget -nv <upgrade-repo-URL> -O /etc/yum.repos.d/dss-app.repo
```

3. Verify that the repository is downloaded:

```
yum search dss-app
```

You should see two dss-app repositories.

4. Update the repository by running the following command:

```
yum update dss-app
```

You should see two versions of the DSS app.

5. Navigate to the dss bin directory:

```
cd /usr/dss-app/1.4.0.0-14/apps/dss/bin/
```

6. Run the upgrade command as follows:

```
./dssdeploy.sh upgrade --from /usr/dss-app/1.3.1.0-392/apps/dss/bin/
```

The following message appears:

```
This will update database schema which cannot be reverted. All backups  
need to be made manually.  
Please confirm to proceed (yes/no):
```

7. Enter **yes** to continue.
8. When prompted, enter the master password for DataPlane.



Note: The master password should be the same as the password used in the previous version of DataPlane.

9. Once the upgrade process is successfully completed, add hosts using the following command:

```
./dssdeploy.sh utils add-host <Host URL> <Host FQDN>
```



Note: This step is required only if you want to add the host entries to the DSS environment for the communication to work before the upgrade.