

Installation 3

Installing HDF Services on a New HDP Cluster

Date of Publish: 2018-11-15



<https://docs.hortonworks.com/>

Contents

Installing Ambari.....	3
Installing Databases.....	3
Installing MySQL.....	3
Configuring SAM and Schema Registry Metadata Stores in MySQL.....	4
Configuring Druid and Superset Metadata Stores in MySQL.....	4
Install Postgres.....	5
Configure Postgres to Allow Remote Connections.....	5
Configure SAM and Schema Registry Metadata Stores in Postgres.....	6
Configure Druid and Superset Metadata Stores in Postgres.....	6
Specifying an Oracle Database to Use with SAM and Schema Registry.....	7
Switching to an Oracle Database After Installation.....	7
Deploying an HDP Cluster Using Ambari.....	8
Installing an HDP Cluster.....	8
Customize Druid Services.....	9
Configure Superset.....	9
Deploy the Cluster Services using Ambari.....	11
Access the Stream Insight Superset UI.....	11
Installing the HDF Management Pack.....	13
Update the HDF Base URL.....	14
Add HDF Services to an HDP Cluster.....	14
Configure HDF Components.....	16
Configure Schema Registry.....	16
Configure SAM.....	16
Configuring SAM log search and event sampling.....	17
Configure NiFi.....	21
Configure NiFi for Atlas Integration.....	21
Configure Kafka.....	22
Configure Storm.....	22
Configure Log Search.....	22
Deploy the Cluster Services.....	23
Access the UI for Deployed Services.....	23
Configuring Schema Registry and SAM for High Availability.....	23
Configuring SAM for High Availability.....	23
Configuring Schema Registry for High Availability.....	23

Installing Ambari

The first step in installing your HDP cluster is installing Ambari.

[Apache Ambari Installation](#)

Installing Databases

When installing Schema Registry, SAM, Druid, and Superset, you require a relational data store to store metadata. You can use either MySQL, Postgres, Oracle, or MariaDB. These topics describe how to install MySQL, Postgres, and Oracle and how create a databases for SAM and Schema Registry. If you are installing on an existing HDP cluster by using Superset, you can skip the installation instructions, because MySQL was installed with Druid. In this case, configure the databases.

**Note:**

You should install either Postgres, Oracle or MySQL; both are not necessary. It is recommended that you use MySQL.

**Attention:**

If you are installing Postgres, you must install Postgres 9.5 or later for SAM and Schema Registry. Ambari does not install Postgres 9.5, so you must perform a manual Postgres installation.

Installing MySQL

You can install MySQL 5.5 or later.

Before you begin

On the Ambari host, install the JDBC driver for MySQL, and then add it to Ambari:

```
yum install mysql-connector-java* \  
sudo ambari-server setup --jdbc-db=mysql \  
--jdbc-driver=/usr/share/java/mysql-connector-java.jar
```

Procedure

1. Log in to the node on which you want to install the MySQL metastore to use for SAM, Schema Registry, and Druid.
2. Install MySQL and the MySQL community server, and start the MySQL service:

```
yum localinstall \  
https://dev.mysql.com/get/mysql57-community-release-el7-8.noarch.rpm  
  
yum install mysql-community-server  
  
systemctl start mysqld.service
```

3. Obtain the randomly generated MySQL root password.

```
grep 'A temporary password is generated for root@localhost' \  
/var/log/mysqld.log |tail -1
```

4. Reset the MySQL root password. Enter the following command. You are prompted for the password you obtained in the previous step. MySQL then asks you to change the password.

```
/usr/bin/mysql_secure_installation
```

Configuring SAM and Schema Registry Metadata Stores in MySQL

Procedure

1. Launch the MySQL monitor:

```
mysql -u root -p
```

2. Create the database for Schema Registry and SAM metastore:

```
create database registry;  
create database streamline;
```

3. Create Schema Registry and SAM user accounts, replacing the final IDENTIFIED BY string with your password:

```
CREATE USER 'registry'@'%' IDENTIFIED BY 'R12$%34qw';  
CREATE USER 'streamline'@'%' IDENTIFIED BY 'R12$%34qw';
```

4. Assign privileges to the user account:

```
GRANT ALL PRIVILEGES ON registry.* TO 'registry'@'%' WITH GRANT OPTION ;  
GRANT ALL PRIVILEGES ON streamline.* TO 'streamline'@'%' WITH GRANT  
OPTION ;
```

5. Commit the operation:

```
commit;
```

Configuring Druid and Superset Metadata Stores in MySQL

Druid and Superset require a relational data store to store metadata. To use MySQL for this, install MySQL and create a database for the Druid metastore.

Procedure

1. Launch the MySQL monitor:

```
mysql -u root -p
```

2. Create the database for the Druid and Superset metastore:

```
CREATE DATABASE druid DEFAULT CHARACTER SET utf8;  
CREATE DATABASE superset DEFAULT CHARACTER SET utf8;
```

3. Create druid and superset user accounts, replacing the final IDENTIFIED BY string with your password:

```
CREATE USER 'druid'@'%' IDENTIFIED BY '9oNio)exlndL';  
CREATE USER 'superset'@'%' IDENTIFIED BY '9oNio)exlndL';
```

- Assign privileges to the druid account:

```
GRANT ALL PRIVILEGES ON *.* TO 'druid'@'%' WITH GRANT OPTION;
GRANT ALL PRIVILEGES ON *.* TO 'superset'@'%' WITH GRANT OPTION;
```

- Commit the operation:

```
commit;
```

Install Postgres

About this task

If you have already installed a MySQL database, you may skip these steps.



Attention:

You must install Postgres 9.5 or later for SAM and Schema Registry. Ambari does not install Postgres 9.5, so you must perform a manual Postgres installation.

Procedure

- Install Red Hat Package Manager (RPM) according to the requirements of your operating system:

```
yum install https://yum.postgresql.org/9.6/redhat/rhel-7-x86_64/pgdg-redhat96-9.6-3.noarch.rpm
```

- Install Postgres version 9.5 or later:

```
yum install postgresql96-server postgresql96-contrib postgresql96
```

- Initialize the database:

- For CentOS 7, use the following syntax:

```
/usr/pgsql-9.6/bin/postgresql96-setup initdb
```

- For CentOS 6, use the following syntax:

```
sudo service postgresql initdb
```

- Start Postgres.

For example, if you are using CentOS 7, use the following syntax:

```
systemctl enable postgresql-9.6.service
systemctl start postgresql-9.6.service
```

- Verify that you can log in:

```
sudo su postgres
psql
```

Configure Postgres to Allow Remote Connections

It is critical that you configure Postgres to allow remote connections before you deploy a cluster. If you do not perform these steps in advance of installing your cluster, the installation fails.

Procedure

1. Open `/var/lib/pgsql/9.6/data/pg_hba.conf` and update to the following

```
# "local" is for Unix domain socket connections only
local all all trust

# IPv4 local connections:
host all all 0.0.0.0/0 trust

# IPv6 local connections:
host all all ::/0 trust
```

2. Open `/var/lib/pgsql/9.6/data/postgresql.conf` and update to the following:

```
listen_addresses = '*'
```

3. Restart Postgres:

```
systemctl stop postgresql-9.6.service
systemctl start postgresql-9.6.service
```

Configure SAM and Schema Registry Metadata Stores in Postgres

If you have already installed MySQL and configured SAM and Schema Registry metadata stores using MySQL, you do not need to configure additional metadata stores in Postgres.

Procedure

1. Log in to Postgres:

```
sudo su postgres
psql
```

2. Create a database called registry with the password registry:

```
create database registry;
CREATE USER registry WITH PASSWORD 'registry';
GRANT ALL PRIVILEGES ON DATABASE "registry" to registry;
```

3. Create a database called streamline with the password streamline:

```
create database streamline;
CREATE USER streamline WITH PASSWORD 'streamline';
GRANT ALL PRIVILEGES ON DATABASE "streamline" to streamline;
```

Configure Druid and Superset Metadata Stores in Postgres

Druid and Superset require a relational data store to store metadata. To use Postgres for this, install Postgres and create a database for the Druid metastore. If you have already created a data store using MySQL, you do not need to configure additional metadata stores in Postgres.

Procedure

1. Log in to Postgres:

```
sudo su postgres
psql
```

2. Create a database, user, and password, each called druid, and assign database privileges to the user druid:

```
create database druid;
CREATE USER druid WITH PASSWORD 'druid';
GRANT ALL PRIVILEGES ON DATABASE "druid" to druid;
```

3. Create a database, user, and password, each called superset, and assign database privileges to the user superset:

```
create database superset;
CREATE USER superset WITH PASSWORD 'superset';
GRANT ALL PRIVILEGES ON DATABASE "superset" to superset;
```

Specifying an Oracle Database to Use with SAM and Schema Registry

You may use an Oracle database with SAM and Schema Registry. Oracle databases 12c and 11g Release 2 are supported

Before you begin

You have an Oracle database installed and configured.

Procedure

1. Register the Oracle JDBC driver jar.

```
sudo ambari-server setup --jdbc-db=oracle --jdbc-driver=/usr/share/java/
ojdbc.jar
```

2. From the SAM and Schema Registry configuration screen, select Oracle as the database type and provide the necessary Oracle Server JDBC credentials and connection string.

Switching to an Oracle Database After Installation

If you want to use an Oracle database with SAM or Schema Registry after you have performed your initial HDF installation or upgrade, you can switch to an Oracle database. Oracle databases 12c and 11g Release 2 are supported

Before you begin

You have an Oracle database installed and configured.

Procedure

1. Log into Ambari Server and shut down SAM or Schema Registry.
2. From the configuration screen, select Oracle as the database type and provide Oracle credentials, the JDBC connection string and click Save.
3. From the command line where Ambari Server is running, register the Oracle JDBC driver jar:

```
sudo ambari-server setup --jdbc-db=oracle --jdbc-driver=/usr/share/java/
ojdbc.jar
```

- From the host where SAM or Schema Registry are installed, copy the JDBC jar to the following location, depending on which component you are updating.

```
cp ojdbc6.jar /usr/hdf/current/registry/bootstrap/lib/.
cp ojdbc6.jar /usr/hdf/current/streamline/bootstrap/lib/.
```

- From the host where SAM or Schema Registry are installed, run the following command to create the required schemas for SAM or Schema Registry.

```
export JAVA_HOME=/usr/jdk64/jdk1.8.0_112 ; source /usr/hdf/current/
streamline/conf/streamline-env.sh ; /usr/hdf/current/streamline/bootstrap/
bootstrap-storage.sh create

export JAVA_HOME=/usr/jdk64/jdk1.8.0_112 ; source /usr/hdf/current/
registry/conf/registry-env.sh ; /usr/hdf/current/registry/bootstrap/
bootstrap-storage.sh create
```



Note:

You only this command run once, from a single host, to prepare the database.

- Confirm that new tables are created in the Oracle database.
- From Ambari, restart SAM or Schema Registry.
- If you are specifying an Oracle database for SAM, run the following command after you have restarted SAM.

```
export JAVA_HOME=/usr/jdk64/jdk1.8.0_112 ; source /usr/hdf/current/
streamline/conf/streamline-env.sh ; /usr/hdf/current/streamline/bootstrap/
bootstrap.sh
```

- Confirm that Sam or Schema Registry are available and turn off maintenance mode.

Deploying an HDP Cluster Using Ambari

To install and configure an HDP cluster for use with HDF's stream analytics capabilities, perform the following steps.

Installing an HDP Cluster

If you are installing an HDF cluster that includes Stream Analytics Manager (SAM), you must have an existing HDP cluster with Druid installed. This section provides instructions for installing HDP for use with SAM. For complete HDP installation instructions, see the *Ambari Installation for HDP*

Procedure

- Log in to the Apache Ambari UI and start the **Cluster Installation** wizard.
The default Ambari user name and password are admin and admin.
- In the **Select Version** page, under public repositories, remove all Base URLs that do not apply to your operating system.
- Change the HDP Base URL to the URL appropriate for the HDP version you are installing, provided in the *HDF Release Notes*.
- On the **Choose Services** step, you must select the following services to run an HDF cluster with full SAM capabilities.
 - HDFS
 - YARN + MapReduce2
 - ZooKeeper

- Ambari Infra
- Ambari Metrics
- SmartSense
- Druid

You may further customize as required by your use case and operational objectives.

5. In the **Assign Masters** page, distribute master services using the deployment diagrams available in *Planning Your Deployment*.
6. On the **Assign Slaves and Clients** screen, distribute slave services using the deployment image as a guide.

Customize Druid Services

After creating a databases for the Druid and Superset store, configure the Druid and Superset services to use the database store.

Procedure

1. On the **Customize Services** page of the Wizard, navigate to the **META DATA STORAGE CONFIG** section of the Druid tab:
 - a) Select a database for the Metadata Storage type, for both Druid and Superset.
 - b) Add user druid and the password you created for Druid.
 - c) Add user superset and the password you created for superset.
 - d) Set the **Metadata Storage host name** to the FQDN of the node running your database server.
 - MySQL – 3306
 - Postgres – 5432
 - e) Set the Metadata Storage port. The default ports are:
 - f) Set the **Database hostname** to the FQDN of the node running your database server.
2. Set the **SECRET_KEY** value, used to encrypt sensitive properties.

Configure Superset

Configure Superset, the data exploration and visualization UI platform

Procedure

1. Provide values for required fields.



Note:

The password is used to log into the SuperSet UI. You should not use special characters in the password due to a bug.

Advanced superset-env ❗

superset-env template

```
# superset log dir
export SUPERSET_LOG_DIR={{superset_log_dir}}

# Superset Log Dir
export SUPERSET_BIN_DIR={{superset_bin_dir}}
```

superset_admin_email

superset_admin_firstname

superset_admin_lastname

Superset Admin Password

Type password

Retype Password

This is re

Superset Admin Username

admin

Superset log dir

/var/log/superset

Superset pid dir

/var/run/superset

2. One of the more powerful visualizations that Superset offers is integration with [MapBox](#). If you want to use this visualization, do the following:
 - a) Create an account at [MapBox](#).
 - b) After account creation, in MapBox, you can need create a token for API Access. Leave all the defaults for token creation.
 - c) You are given a token API key.
 - d) Enter that token API key in the Druid Service under Ambari for the config called MAPBOX_API_KEY under section **Advanced druid-superset**.

META DATA STORAGE CONFIG

Advanced

Advanced druid-superset

MAPBOX_API_KEY

pk.eyJ1Ijoiz3ZldHRpY2FkZW4iLCJhIjoie2oxbnVoaTNUMDB3ZDJ3bGJoN2VmNTJlYyJ9.

Deploy the Cluster Services using Ambari

Procedure

Complete the **Cluster Install** wizard and deploy all the services. If the Druid Services do not start, verify that the Druid Meta data storage settings are correct. After correcting, choose **Start All** (not **Restart**).

Results

After successfully starting Druid services, you should see something like the following:

The screenshot shows the Ambari web interface. The top navigation bar includes 'Ambari', 'streaminsight', '0 ops', and '1 alert'. The 'Services' tab is selected. On the left, a sidebar lists services: HDFS, YARN, MapReduce2, ZooKeeper, Ambari Infra, Ambari Metrics, SmartSense (with a red '1' alert icon), and **Druid** (highlighted). Below the sidebar is an 'Actions' dropdown. The main content area shows the 'Summary' tab for the Druid service. It lists the following components and their status:

- [Druid Broker](#) Started No alerts
- [Druid Coordinator](#) Started No alerts
- [Druid Overlord](#) Started No alerts
- [Druid Router](#) Started No alerts
- [Superset](#) Started No alerts
- [Druid Historicals](#) 4/5 Druid Historicals Live
- [Druid MiddleManagers](#) 5/5 Druid MiddleManagers Live

Access the Stream Insight Superset UI

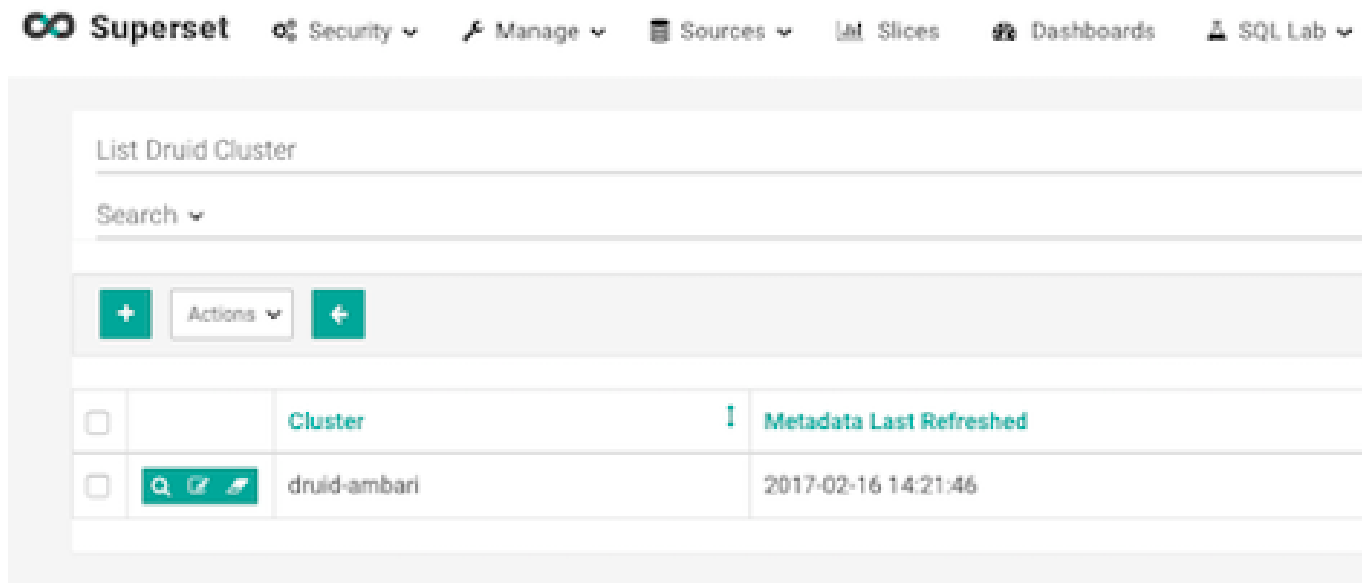
As part of the install, Superset is configured with the Druid cluster that was installed as part of Ambari. To verify installation and access the UI.

Procedure


1. Select the Druid Service in Ambari and under the Quick Links menu, choose Superset to navigate to the Stream Insight UI.

The default user account is admin; the password is what you configured for the Superset service.

2. Click Sources > Druid Clusters.
3. You should see a Druid cluster already configured. It should look like the following:



The screenshot shows the Superset web interface. At the top, there is a navigation bar with the Superset logo and several menu items: Security, Manage, Sources, Slices, Dashboards, and SQL Lab. Below the navigation bar, the main content area is titled 'List Druid Cluster'. There is a search bar and a table with columns for 'Cluster' and 'Metadata Last Refreshed'. The table contains one row with the cluster name 'druid-ambari' and the refresh timestamp '2017-02-16 14:21:46'.

		Cluster	Metadata Last Refreshed
<input type="checkbox"/>		druid-ambari	2017-02-16 14:21:46

4. Once a Druid cluster is set up, you can pull in all of the data sources or cubes associated with that cluster clicking on Sources > Refresh Druid Metadata.

If you have a streaming app that pushes data into data sources, the refresh should pull in all data sources within the Druid cluster. See the following image for an example.



Important:

Whenever you add new data sources to Druid, perform the Refresh Druid Metadata action.

The screenshot shows the Superset web interface. At the top, there is a navigation bar with the Superset logo and several menu items: Security, Manage, Sources, Slices, Dashboards, and SQL Lab. Below the navigation bar, there is a section titled "List Druid Datasource" with a search input field. Underneath, there is a toolbar with a plus sign, an "Actions" dropdown menu, and another plus sign. The main content is a table with the following columns: a checkbox, a search icon, a refresh icon, a delete icon, "Data Source", "Cluster", "Changed By", and "Changed On".

			Data Source	Cluster	Changed By	Changed On	
<input type="checkbox"/>				Alerts-High-Speed-Cube	Streaming Analytics Manager - Stream Insight	George Vetticaden	2017-02-08 05
<input type="checkbox"/>				Alerts-High-Speed-Cube-V2	Streaming Analytics Manager - Stream Insight	George Vetticaden	2017-02-07 15
<input type="checkbox"/>				driver-violations-cube	Streaming Analytics Manager - Stream Insight	George Vetticaden	2017-02-08 05
<input type="checkbox"/>				driver-violations-cube-2	Streaming Analytics Manager - Stream Insight	George Vetticaden	2017-02-07 17

Installing the HDF Management Pack

A management pack (mpack) bundles service definitions, stack definitions, and stack add-on service definitions so they do not need to be included with the Ambari core functionality and can be updated in between major releases.

Procedure

1. Download the Hortonworks HDF management pack. You can find the download location for your operating system in the *HDF Release Notes*.
2. Copy the bundle to /tmp on the node where you installed Ambari.
3. Install the management pack:

```
ambari-server install-mpack \
--mpack=/tmp/hdf-ambari-mpack-<version>.tar.gz \
--verbose
```

4. Restart the Ambari server:

```
ambari-server restart
```

Related Information

[HDF Release Notes](#)

Update the HDF Base URL

Adding the base URL tells Ambari where to look for the HDF repository. This step is necessary when you are using an existing Ambari instance, already managing an HDP cluster, to install and manage an HDF cluster.

Procedure

1. From the Ambari menu, click the **admin** drop-down in the top right of your Ambari Dashboard view. Then select **Manage Ambari**.
2. From the **Clusters** view on the left, click **Versions**, and then click the **HDP version** link.
3. Configure the HDF Base URL to the base URL appropriate for your operating system. Find the HDF Base URLs in the *HDF Release Notes*.
4. Click **Save**.

What to do next

Add HDF Services to an HDP Cluster

You can use the HDF management pack and Ambari to add HDF services to an HDP cluster.

Procedure

1. If you are installing HDF services on an existing HDP Cluster, on the Ambari home page, click the button **Actions** and select + **Add Service**.

The screenshot displays the Ambari interface for managing services. On the left, a list of services is shown with their status: HDFS, YARN, MapReduce2, ZooKeeper, Ambari Infra, Ambari Metrics, SmartSense (with a red warning triangle), and Druid. Below this list is an 'Actions' button with a dropdown arrow. To the right, there are panels for 'Metrics', 'Metric Act', 'HDFS Disk', and 'CPU Usage'. A context menu is open over the 'Actions' button, listing the following options: '+ Add Service', '▶ Start All', '■ Stop All', '⌂ Restart All Required', and '⬇ Download All Client Configs'.

- ✓ HDFS
- ✓ YARN
- ✓ MapReduce2
- ✓ ZooKeeper
- ✓ Ambari Infra
- ✓ Ambari Metrics
- ⚠ SmartSense
- ✓ Druid

Actions ▾

- + Add Service
- ▶ Start All
- Stop All
- ⌂ Restart All Required
- ⬇ Download All Client Configs

2. Chose select the HDF Services (NiFi and NiFi Registry) you want to install.
3. On the Assign Masters screen, distribute master services using the preceding deployment diagram of the Stream Processing cluster.
4. On the Assign Slaves and Clients screen, distribute slave services using the deployment diagram of the Stream Processing cluster.

Configure HDF Components

You can customize your HDF component configurations during installation, or any later time. During installation, customize HDF component configurations in the **Customize Services** step of the Installation Wizard. At any other time, click the service you want to configure in the left-hand Services pane, in the Ambari Dashboard, then click the **Configs** tab.

Configure Schema Registry

The main Schema Registry configuration task you have is to establish a connection between Schema Registry and the database you are going to use as the metadata store.

Procedure

1. In the **Customize Services** step, navigate to the **REGISTRY CONFIG** section of the **Registry** tab.
2. Select **Jar Storage Type**. If you plan to enable HA for Schema Registry on this cluster, you must select **HDFS**.
3. If you selected **HDFS** as the **Jar Storage Type**, configure **Jar Storage HDFS URL**. This specifies the HDFS location where you want the jars to be stored. For example, `hdfs://<<NN_HOST:8020>>/hdfs/registry`.
4. Set **jar.storage** to the directory location where you want to store .jar files for serializers and deserializers.
5. Configure the **REGISTRY STORAGE** configurations based on the database you created to use as the Schema Registry metadata store.
6. Ensure that the Schema Registry storage connector URL has the fully qualified host name for the database installation location, the connector URL, and the default port for the database selected.

Example

```
MYSQL example:
```

```
jdbc:mysql://FQDN_MYSQL:3306/registry
```

```
Postgres Example:
```

```
jdbc:postgresql://FQDN_POSTGRES:5432/registry
```

Configure SAM

When you configure SAM, you need to provide information about the metadata store database, configure a connection with Schema Registry, and establish the URL for Druid's Supersets.

Procedure

1. In the **Customize Services** step, navigate to the **STREAMLINE CONFIG** section of the **Streaming Analytics Manager** tab.
2. Select **Jar Storage Type**. If you plan to enable HA for SAM on this cluster, you must select **HDFS**.

3. If you selected **HDFS** as the **Jar Storage Type**, configure **Jar Storage HDFS URL**. This specifies the HDFS location where you want the jars to be stored. For example, `hdfs://<<NN_HOST:8020:/hdfs/registry`.
4. Set **jar.storage** to the directory location where you want to store `.jar` files for custom processors.
5. Set the **streamline.dashboard.url** to the Superset URL which you can access using **Quick Links** for Druid.
6. Configure **registry.url** to the REST API Endpoint URL for the Registry. The format should be `http://$FQDN_REGISTRY_HOST:$REGISTRY_PORT/api/v1`, where:
 - `$FQDN_REGISTRY_HOST` – Specifies the host on which you are running Schema Registry
 - `$REGISTRY_PORT` – Specifies the Schema Registry port number located in the **REGISTRY_CONFIG** section of the **Registry** tab.

For example: `http://FQDN_REGISTRY_HOST:7788/api/v1`

7. Configure the **STREAMLINE STORAGE** configurations based on the database you created to use as a SAM metadata store.
8. Ensure that the Schema Registry storage connector URL has the fully qualified host name for the database installation location, the connector URL, and the default port for the database selected.

Example

MYSQL example:

```
jdbc:mysql://FQDN_MYSQL:3306/streamline
```

Postgres Example:

```
jdbc:postgresql://FQDN_POSTGRES:5432/streamline
```

Configuring SAM log search and event sampling

You must perform several manual steps to enable log search and event sampling for SAM.

Procedure

1. From the Log Search UI, click the Configuration icon on the top right of your screen. Select **Configuration Editor**.
2. From the left hand **All Configuration** pane, select **Storm**.
3. In the **Edit Configuration** field, replace the existing json text with the following configuration:

```
{
  "input": [
    {
      "type": "storm_drpc",
      "rowtype": "service",
      "path": "/var/log/storm/drpc.log"
    },
    {
      "type": "storm_logviewer",
      "rowtype": "service",
      "path": "/var/log/storm/logviewer.log"
    },
    {
      "type": "storm_nimbus",
      "rowtype": "service",
      "path": "/var/log/storm/nimbus.log"
    },
    {
      "type": "storm_supervisor",
```

```

    "rowtype": "service",
    "path": "/var/log/storm/supervisor.log"
  },
  {
    "type": "storm_ui",
    "rowtype": "service",
    "path": "/var/log/storm/ui.log"
  },
  {
    "type": "storm_worker",
    "rowtype": "service",
    "path": "/var/log/storm/workers-artifacts/**/worker.log",
    "cache_enabled": true,
    "cache_key_field": "log_message",
    "cache_last_dedup_enabled": true,
    "cache_size": 100,
    "cache_dedup_interval": 1000,
    "init_default_fields": true
  },
  {
    "type": "storm_worker_event",
    "rowtype": "service",
    "path": "/var/log/storm/workers-artifacts/**/events.log",
    "add_fields": {
      "level": "INFO"
    },
    "cache_enabled": true,
    "cache_key_field": "log_message",
    "cache_last_dedup_enabled": true,
    "cache_size": 100,
    "cache_dedup_interval": 1000,
    "init_default_fields": true
  }
],
"filter": [
  {
    "filter": "grok",
    "conditions": {
      "fields": {
        "type": [
          "storm_supervisor",
          "storm_logviewer",
          "storm_drpc",
          "storm_worker",
          "storm_ui",
          "storm_nimbus"
        ]
      }
    }
  },
  {
    "skipOnError": false,
    "deepExtract": false,
    "sort_order": 1,
    "post_map_values": {
      "logtime": [
        {
          "map_date": {
            "target_date_pattern": "yyyy-MM-dd HH:mm:ss.SSS"
          }
        }
      ]
    }
  }
],
"log4j_format": "",
"multiline_pattern": "^(%{TIMESTAMP_ISO8601:logtime})"

```

```

        "message_pattern": "(?m)^
%{TIMESTAMP_ISO8601:logtime}%{SPACE}%{JAVACLASS:logger_name}\
\s%{GREEDYDATA:thread_name}\s\[\%{LOGLEVEL:level}\]\s\
%{GREEDYDATA:log_message}"
    },
    {
        "filter": "grok",
        "conditions": {
            "fields": {
                "type": [
                    "storm_worker"
                ]
            }
        },
        "skipOnError": false,
        "deepExtract": false,
        "sort_order": 2,
        "source_field": "thread_name",
        "remove_source_field": false,
        "message_pattern": "(Thread\\-[0-9]+\\-[0-9]+\\-
%{DATA:sdi_streamline_component_name}\\-executor%{DATA}|
%{DATA:thread_name})"
    },
    {
        "filter": "grok",
        "conditions": {
            "fields": {
                "type": [
                    "storm_worker"
                ]
            }
        },
        "skipOnError": false,
        "deepExtract": false,
        "sort_order": 3,
        "source_field": "path",
        "remove_source_field": false,
        "message_pattern": "/var/log/storm/workers-artifacts/
%{DATA:sdi_storm_topology_id}/%{DATA:sdi_storm_worker_port}/worker\\.log"
    },
    {
        "filter": "grok",
        "conditions": {
            "fields": {
                "type": [
                    "storm_worker"
                ]
            }
        },
        "skipOnError": false,
        "deepExtract": false,
        "sort_order": 4,
        "source_field": "sdi_storm_topology_id",
        "remove_source_field": false,
        "message_pattern": "(streamline\\-
%{DATA:sdi_streamline_topology_id}\\-%{DATA:sdi_streamline_topology_name}\\
\\-[0-9]+\\-[0-9]+)|(%{DATA:sdi_storm_topology_id})"
    },
    {
        "filter": "grok",
        "conditions": {
            "fields": {
                "type": [
                    "storm_worker_event"
                ]
            }
        }
    }

```

```

    ]
    },
    "skipOnError":false,
    "deepExtract":false,
    "sort_order":5,
    "post_map_values":{
      "logtime":[
        {
          "map_date":{
            "target_date_pattern":"yyyy-MM-dd HH:mm:ss.SSS"
          }
        }
      ]
    },
    "log4j_format":"","",
    "message_pattern":"^%{TIMESTAMP_ISO8601:logtime}(!_DELIM_!
<STREAMLINE_EVENT>!_DELIM_!%{DATA:sdi_streamline_component_name}!
_DELIM_!%{DATA:sdi_streamline_event_id}!_DELIM_!
%{DATA:sdi_streamline_root_ids}!_DELIM_!%{DATA:sdi_streamline_parent_ids}!
_DELIM_!%{DATA:sdi_streamline_event_fields_and_values}!
_DELIM_!%{DATA:sdi_streamline_event_headers}!_DELIM_!
%{DATA:sdi_streamline_event_aux_fields_and_values})|(%{GREEDYDATA})"
  },
  {
    "filter":"grok",
    "conditions":{
      "fields":{
        "type":[
          "storm_worker_event"
        ]
      }
    }
  },
  "skipOnError":false,
  "deepExtract":false,
  "sort_order":6,
  "source_field":"path",
  "remove_source_field":false,
  "message_pattern":"/var/log/storm/workers-artifacts/
%{DATA:sdi_storm_topology_id}/%{DATA:sdi_storm_worker_port}/events\\.log"
},
{
  "filter":"grok",
  "conditions":{
    "fields":{
      "type":[
        "storm_worker_event"
      ]
    }
  }
},
"skipOnError":false,
"deepExtract":false,
"sort_order":7,
"source_field":"sdi_storm_topology_id",
"remove_source_field":false,
"message_pattern":"(streamline\\-
%{DATA:sdi_streamline_topology_id}\\-%{DATA:sdi_streamline_topology_name}\\
\\-[0-9]+\\-\\-[0-9]+)|(%{DATA:sdi_storm_topology_id})"
}
]
}

```

4. Verify that storm log directory is correct. The above json content is `/var/log/storm/`. You should replace it with the actual log directory path if your cluster uses different log directory.
5. Click **Save**.

Configure NiFi

You use the **NiFi** tab in the **Customize Services** step to configure Apache NiFi. Generally, you can accept the defaults during initial installation. However, there are some settings that you must configure before proceeding.

Procedure

1. From **Advanced-nifi-ambari-config**, specify the **Encrypt Configuration Master Key Passwords**.

This password is used when you generate the master key for sensitive properties encryption in the NiFi properties file when it is written to disk. It must contain at least 12 characters.

2. From **Advanced-nifi-ambari-config**, provide the **Sensitive property values encryption password**.

This is the password used when you encrypt any sensitive property values that are configured in processors. For enhanced security, it should contain at least 10 characters.

Configure NiFi for Atlas Integration

You can integrate NiFi with Apache Atlas to take advantage of robust dataset and application lineage support. You do this by configuring the NiFi ReportLineageToAtlas Reporting Task once you have NiFi configured and running.

Before you begin

If NiFi is installed on an HDP cluster, you must be running HDP 2.6.4 or later. If NiFi is installed on an HDF cluster managed by a separate Ambari instance, you must be running HDP 2.6.1 or later, and Apache Atlas 0.8.0 or later.

Procedure

1. From the Global Menu located in NiFi's upper right corner, select Controller Services and click the Reporting Tasks tab.
2. Click the Add (+) icon to launch the Add Reporting Task dialog.
3. Select ReportLineageToAtlas and click Add.
4. Click the Edit icon to launch the Configure Reporting Task dialog. The following Properties are required:
 - Atlas URLs – a comma-separated list of Atlas Server URLs. Once you have started reporting, you cannot modify an existing Reporting Task to add a new Atlas Server. When you need to add a new Atlas Server, you must create a new reporting task.
 - Atlas Authentication Method – Specifies how to authenticate the Reporting Task to the Atlas Server. Basic authentication is the default.
 - NiFi URL for Atlas – Specifies the NiFi cluster URL.
 - NiFi Lineage Strategy – Specifies the level of granularity for your NiFi dataflow reporting to Atlas. Once you have started reporting, you should not switch between simple and complete lineage reporting strategies.
 - Provenance Record Start Position – Specifies where in the Provenance Events stream the Reporting Task should start.
 - Provenance Record Batch Size – Specifies how many records you want to send in a single batch.
 - Create Atlas Configuration File – If enabled, the atlas-application-properties file and the Atlas Configuration Directory are automatically created when the Reporting Task starts.
 - Kafka Security Protocol – Specifies the protocol used to communicate with Kafka brokers to send Atlas hook notification messages. This value should match Kafka's security.protocol property value.

Results

Once you have ReportLineageToAtlas up and running, you may view dataset level lineage graphs in the Atlas UI.



Note:

The default time interval for the Reporting Task to start sending data to an Atlas Server is 5 minutes so do not expect to see immediate lineage graphs. You can change the default time interval in the Reporting Task property configuration.

What to do next

More Information

For complete information, see the help included with the Reporting Task.

Configure Kafka

You can configure Apache Kafka from the **Kafka** tab in the **Customize Services** step.

Procedure

1. For your initial installation, accept the default values set by Apache Ambari.
2. If Ambari prompts you with Some configurations need your attention before you can proceed, review the list of properties and provide the required information.
3. Review the *Apache Kafka Component Guide* for information about configuring Kafka to meet your operational objectives.
4. If you have enabled Kerberos, ensure that the listener configuration is set to `advertised.listeners=SASL_PLAINTEXT://$HOSTNAME:$PORT`.

Configure Storm

You can configure Storm from the **Storm** tab in the **Customize Services** step.

Procedure

1. For your initial installation, accept the default values set by Ambari.
2. If Ambari prompts you with: Some configurations need your attention before you can proceed. Review the list of properties and provide the required information.
3. Review the *Apache Storm Component Guide* for information about configuring storm to meet your operational objectives.

Configure Log Search

To ensure that you can view logs in the new SAM Log Search, you can manually review and adjust Log Search Settings for `storm_worker` and `storm_worker_event`.

Procedure

1. From the left-hand navigation pane, select **Log Search | Configs**.
2. Manually set the Log Feeder Log Levels Filter for `storm_worker` and `storm_worker_event` to include **Info**, **Debug**, and **Trace**.

Deploy the Cluster Services

Procedure

Finish the wizard and deploy the cluster. After the cluster has been deployed, some services might fail in starting. If this is the case, start those services individually.

Access the UI for Deployed Services

Once you have deployed your Ambari-managed cluster, you can launch the UI for any of the services from Ambari.

Procedure

1. From Ambari's left-hand **Services** pane, click the service you want.
2. From the **Quick Links** drop-down, select the UI option.
3. Find links for the SAM UI under **Streaming Analytics Manager** and for the Schema Registry UI under **Registry**.

Results

The UI for your HDF service opens in a new window.

Configuring Schema Registry and SAM for High Availability

Configuring SAM for High Availability

You can configure SAM for high availability.

Procedure

1. Install two or more instances of SAM on unique nodes.
2. From the Services pane, select **Streaming Analytics Manager** and click the **Configs** tab.
3. In the **Jar Storage Type** drop down, select **HDFS** or **Database**.



Note:

If you are using a MySQL database, ensure that you make adjustments to the database configuration as well. `max_allowed_packet` must be greater than the maximum file size of any custom processor or user defined function that will be uploaded.

Configuring Schema Registry for High Availability

You can configure Schema Registry for high availability.

Procedure

1. Install two or more instances of Schema Registry on unique nodes.
2. From the Services pane, select **Schema Registry** and click the **Configs** tab.
3. In the **Jar Storage Type** drop down, select **HDFS**.