

Hortonworks DataFlow

Installing HDF Services on an Existing HDP Cluster

(November 9, 2017)

Hortonworks DataFlow: Installing HDF Services on an Existing HDP Cluster

Copyright © 2012-2017 Hortonworks, Inc. Some rights reserved.

Hortonworks DataFlow (HDF) is powered by Apache NiFi. A version of this documentation originally appeared on the [Apache NiFi website](#).

HDF is the first integrated platform that solves the real time challenges of collecting and transporting data from a multitude of sources and provides interactive command and control of live flows with full and automated data provenance. HDF is a single combined platform that provides the data acquisition, simple event processing, transport and delivery mechanism designed to accommodate the diverse dataflows generated by a world of connected people, systems and things.

Unlike other providers of platforms built using Apache Hadoop, Hortonworks contributes 100% of our code back to the Apache Software Foundation. Hortonworks DataFlow is Apache-licensed and completely open source. We sell only expert technical support, training and partner-enablement services. All of our technology is, and will remain free and open source.

Please visit the [Hortonworks](#) page for more information on Hortonworks technology. For more information on Hortonworks services, please visit either the [Support](#) or [Training](#) page. Feel free to [Contact Us](#) directly to discuss your specific needs.



Except where otherwise noted, this document is licensed under
Creative Commons Attribution ShareAlike 4.0 License.
<http://creativecommons.org/licenses/by-sa/4.0/legalcode>

Table of Contents

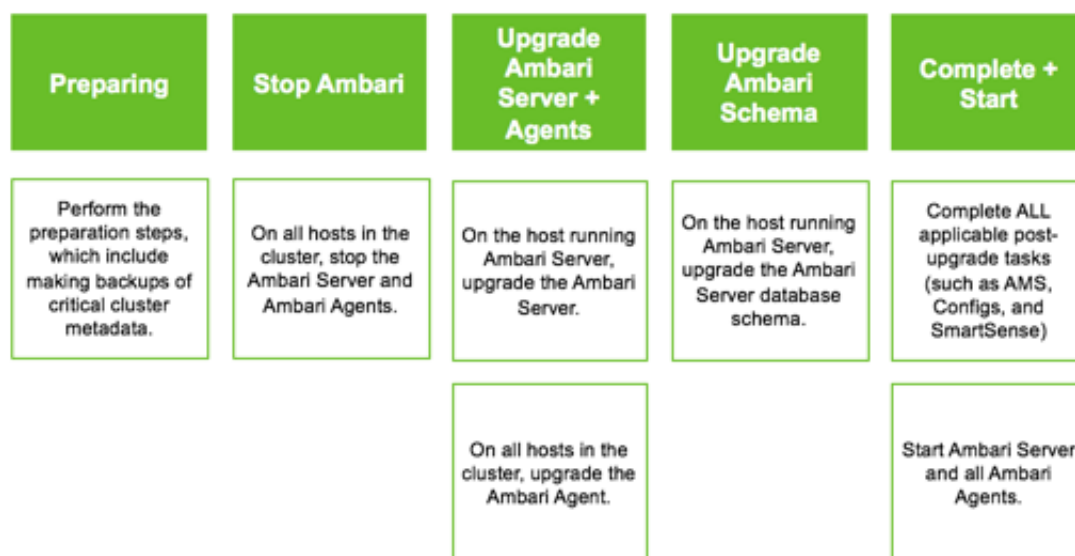
1. Upgrading Ambari	1
1.1. Preparing to Upgrade	1
1.2. Upgrade Ambari	2
1.3. <i>Mandatory</i> Post-Upgrade Tasks	7
1.3.1. Upgrading Ambari Infra	8
1.3.2. Upgrading Ambari Log Search	9
1.3.3. Upgrading Ambari Metrics	10
1.3.4. Upgrading Configurations	12
1.3.5. Upgrading SmartSense	19
2. Upgrading to HDP 2.6.3	20
2.1. Before you begin	20
2.2. Upgrade options	20
3. Installing Databases	21
3.1. Installing MySQL	21
3.2. Configuring SAM and Schema Registry Metadata Stores in MySQL	22
3.3. Configuring Druid and Superset Metadata Stores in MySQL	23
3.4. Install Postgres	23
3.5. Configure Postgres to Allow Remote Connections	24
3.6. Configure SAM and Schema Registry Metadata Stores in Postgres	25
3.7. Configure Druid and Superset Metadata Stores in Postgres	25
3.8. Specifying an Oracle Database to Use with SAM and Schema Registry	26
3.9. Switching to an Oracle Database After Installation	26
4. Installing the HDF Management Pack	28
5. Update the HDF Base URL	29
6. Add HDF Services to an HDP Cluster	30
7. Configure HDF Components	31
7.1. Configure Schema Registry	31
7.2. Configure SAM	32
7.3. Configure NiFi	33
7.4. Configure Kafka	33
7.5. Configure Storm	34
7.6. Deploy the Cluster Services	34
7.7. Access the UI for Deployed Services	35
8. Configuring Schema Registry and SAM for High Availability	36
9. Install the Storm Ambari View	37
10. Using a Local Repository	39
10.1. Setting Up a Local Repository	39
10.2. Getting Started Setting Up a Local Repository	39
10.2.1. Setting Up a Local Repository with No Internet Access	40
10.2.2. Setting up a Local Repository With Temporary Internet Access	42
10.3. Preparing The Ambari Repository Configuration File	44
11. Navigating the HDF Library	46

1. Upgrading Ambari

Ambari and the HDP cluster being managed by Ambari can be upgraded independently. This section describes the process to upgrade Ambari. You are **strongly encouraged** to read completely through this entire document before starting the upgrade process, to that you understand the interdependencies and order of the steps. It is **highly recommended** you validate these steps in a test environment to adjust + account for any special configurations for your cluster.

- [Preparing to Upgrade \[1\]](#)
- [Upgrade Ambari \[2\]](#)
- [Mandatory Post-Upgrade Tasks \[7\]](#)

The high-level process for upgrading Ambari is as follows:



Important

Completing post-upgrade tasks is mandatory.

1.1. Preparing to Upgrade

- Be sure to review the Ambari 2.6.0.0 release notes for Known Issues and Behavioral Changes.
- You **must** have root, administrative, or root-equivalent authorization on the Ambari server host and all servers in the cluster.
- You **must** backup the Ambari Server database.
- You **must** make a safe copy of the Ambari Server configuration file found at `/etc/ambari-server/conf/ambari.properties`.

- **Plan to upgrade the Ambari Metrics service:**
 - Record the location of the **Metrics Collector** component before you begin the upgrade process.
 - You **must** stop the Ambari Metrics service from **Ambari Web**.
 - After upgrading Ambari, you must also upgrade Ambari Metrics System and add the Grafana component.
- After upgrading Ambari, you must also upgrade SmartSense.



Note

During Ambari upgrade, the existing `/var/lib/ambari-server/ambari-env.sh` file is overwritten and a backup copy of `ambari-env.sh` (with extension `.rpmsave`) is created. If you have manually modified `ambari-env.sh` (for example, to change Ambari Server heap), you will need to re-apply your changes to the new file.

Next Steps

[Upgrade Ambari \[2\]](#)

More Information

[Ambari 2.6.0.0 Release Notes](#)

1.2. Upgrade Ambari

1. If you are running Ambari Metrics service in your cluster, stop the service. From **Ambari Web**, browse to **Services > Ambari Metrics** and select **Stop** from the **Service Actions** menu.
2. Stop the Ambari Server. On **the host** running Ambari Server:

```
ambari-server stop
```
3. Stop all Ambari Agents. On **each host** in your cluster running an Ambari Agent:

```
ambari-agent stop
```
4. Fetch the new Ambari repo and replace the old repository file with the new repository file **on all hosts** in your cluster.



Important

Check your current directory before you download the new repository file to make sure that there are no previous versions of the `ambari.repo` file. If you do not, and a previous version exists, the new download will be saved with a numeric extension, such as `ambari.repo.1`. Make sure that the version you copy is the new version.

Select the repository appropriate for your environment from the following list:

- **For RHEL/CentOS/Oracle Linux 6:**

```
wget -nv http://public-repo-1.hortonworks.com/ambari/centos6/2.x/updates/2.6.0.0/ambari.repo -O /etc/yum.repos.d/ambari.repo
```

- **For RHEL/CentOS/Oracle Linux 7:**

```
wget -nv http://public-repo-1.hortonworks.com/ambari/centos7/2.x/updates/2.6.0.0/ambari.repo -O /etc/yum.repos.d/ambari.repo
```

- **For SLES 11:**

```
wget -nv http://public-repo-1.hortonworks.com/ambari/suse11/2.x/updates/2.6.0.0/ambari.repo -O /etc/zypp/repos.d/ambari.repo
```

- **For SLES 12:**

```
wget -nv http://public-repo-1.hortonworks.com/ambari/sles12/2.x/updates/2.6.0.0/ambari.repo -O /etc/zypp/repos.d/ambari.repo
```

- **For Ubuntu 12:**

```
wget -nv http://public-repo-1.hortonworks.com/ambari/ubuntu12/2.x/updates/2.6.0.0/ambari.list -O /etc/apt/sources.list.d/ambari.list
```

- **For Ubuntu 14:**

```
wget -nv http://public-repo-1.hortonworks.com/ambari/ubuntu14/2.x/updates/2.6.0.0/ambari.list -O /etc/apt/sources.list.d/ambari.list
```

- **For Debian 7:**

```
wget -nv http://public-repo-1.hortonworks.com/ambari/debian7/2.x/updates/2.6.0.0/ambari.list -O /etc/apt/sources.list.d/ambari.list
```



Note

If your cluster does not have access to the Internet, set up a local repository with this data before you continue.



Note

Ambari Server does not automatically turn off `iptables`. Check that your installation setup does not depend on `iptables` being disabled. After upgrading the server, you must either disable `iptables` manually or make sure that you have appropriate ports available on all cluster hosts.

5. Upgrade Ambari Server. On the host running Ambari Server:

- **For RHEL/CentOS/Oracle Linux:**

```
yum clean all
```

```
yum info ambari-server
```

In the info output, visually validate that there is an available version containing "2.6"

```
yum upgrade ambari-server
```

- **For SLES:**

```
zypper clean
```

```
zypper info ambari-server
```

In the info output, visually validate that there is an available version containing "2.6"

```
zypper up ambari-server
```

- **For Ubuntu/Debian:**

```
apt-get clean all
```

```
apt-get update
```

```
apt-cache show ambari-server | grep Version
```

In the info output, visually validate that there is an available version containing "2.6"

```
apt-get install ambari-server
```



Important

When performing upgrade on SLES, you will see a message "There is an update candidate for 'ambari-server', but it is from different vendor. Use 'zypper install ambari-server-2.5-101.noarch' to install this candidate". You will need to use yast to update the package, as follows:

- a. From the command line run: > yast.

```
> yast
```

You will see command line UI for YaST program.

- b. Choose **Software > Software Management**, then click the **Enter** button.
- c. In the **Search Phrase** field, enter **ambari-server**, then click the **Enter** button.
- d. On the right side you will see the search result `ambari-server 2.6`. Click **Actions**, choose **Update**, then click the **Enter** button.
- e. Go to **Accept**, and click **enter**.

6. Check for upgrade success by noting progress during the Ambari Server installation process you started in Step 5.

- As the process runs, the console displays output similar, although not identical, to the following:

```
Setting up Upgrade Process Resolving Dependencies --> Running transaction check
```

- If the upgrade fails, the console displays output similar to the following:

```
Setting up Upgrade Process No Packages marked for Update
```

- A successful upgrade displays output similar to the following:

```
Updated: ambari-server.noarch 0:2.6-111 Complete!
```



Note

Confirm there is only one `ambari-server*.jar` file in `/usr/lib/ambari-server`. If there is more than one JAR file with name `ambari-server*.jar`, move all JARs except `ambari-server-2.6.*.jar` to `/tmp` before proceeding with upgrade.

7. Upgrade all Ambari Agents. On **each host** in your cluster running an Ambari Agent:

- For RHEL/CentOS/Oracle Linux:

```
yum upgrade ambari-agent
```

- For SLES:

```
zypper up ambari-agent
```



Note

Ignore the warning that begins with "There are some running programs that use files deleted by recent upgrade".



Important

When performing upgrade on SLES, you will see a message "There is an update candidate for 'ambari-agent', but it is from different vendor. Use 'zypper install ambari-agent-2.5-101.noarch' to install this candidate". You will need to use `yast` to update the package, as follows:

- a. From the command line run: `> yast`

```
> yast
```

You will see command line UI for YaST program.

- b. Choose **Software > Software Management**, then click the **Enter** button.
- c. In the **Search Phrase** field, enter **ambari-agent**, then click the **Enter** button.
- d. On the right side you will see the search result `ambari-agent 2.6`. Click **Actions**, choose **Update**, then click the **Enter** button.
- e. Go to **Accept**, and click **enter**.

- **For Ubuntu/Debian:**

```
apt-get update  
apt-get install ambari-agent
```

8. After the upgrade process completes, check each host to make sure the new files have been installed:

For RHEL/CentOS/Oracle Linux 6: `rpm -qa | grep ambari-agent`

For RHEL/CentOS/Oracle Linux 7: `rpm -qa | grep ambari-agent`

For SLES 11: `rpm -qa | grep ambari-agent`

For SLES 12: `rpm -qa | grep ambari-agent`

For Ubuntu 14: `dpkg -l ambari-agent`

For Ubuntu 16: `dpkg -l ambari-agent`

For Debian 7: `dpkg -l ambari-agent`

9. Upgrade Ambari Server database schema. On **the host** running Ambari Server:

```
ambari-server upgrade
```

- 10 Start the Ambari Server. On **the host** running Ambari Server:

```
ambari-server start
```

- 11 Start all Ambari Agents. On **each host** in your cluster running an Ambari Agent:

```
ambari-agent start
```

- 12 Open Ambari Web.

Point your browser to `http://<your.ambari.server>:8080`

where `<your.ambari.server>` is the name of your ambari server host. For example, `c6401.ambari.apache.org`.



Important

Refresh your browser so that it loads the new version of the Ambari Web code. If you have problems, clear your browser cache manually, then restart Ambari Server.

- 13 Log in, using the Ambari administrator credentials that you have set up.

For example, the default name/password is **admin/admin**.

You will see a Restart indicator next to each service after upgrading. Ambari upgrade has added to/adjusted the configuration properties of your cluster based on new

configuration types and properties being made available for each service with this release of Ambari. Review these changes by comparing the previous configuration with the latest version created by "ambari-upgrade".

- 14.If you have configured Ambari to authenticate against an external LDAP or Active Directory, you **must** re-run

```
ambari-server setup-ldap
```

- 15.If you have configured your cluster for Hive or Oozie with an external database (Oracle, MySQL or PostgreSQL), you **must** re-run

```
ambari-server setup --jdbc-db and --jdbc-driver
```

to get the JDBC driver .jar file in place.

- 16.If you are running **Ambari Metrics** service in your cluster, you **must** upgrade Ambari Metrics System and add the Grafana component.

- 17.If your cluster includes the SmartSense service, you **must** upgrade SmartSense along with Ambari.

- 18.Perform any other post-upgrade tasks, as necessary.



Important

Completing post-upgrade tasks is mandatory.

Next Steps

[Post-Upgrade Tasks](#) **Mandatory**

1.3. Mandatory Post-Upgrade Tasks

Depending on the configuration of your cluster and your current Ambari version, you must upgrade any of the following features in your cluster, as described in the following topics:

Upgrading Ambari Infra

If your cluster includes Ambari Infra service, you must upgrade it along with Ambari.

Upgrading Ambari Log Search

If your cluster includes Ambari Log Search service, you must upgrade it along with Ambari.

Upgrading Ambari Metrics

If your cluster includes the Ambari Metrics System (AMS) service, you must upgrade the system along with Ambari. This will include adding the Grafana component to the system.

Adding Grafana to Ambari Metrics

Grafana is now included as a component of Ambari Metrics. If you are upgrading from Ambari 2.2.1 or earlier, and your Ambari Metrics service does not

contain Grafana, proceed to add Grafana to Ambari Metrics.

Upgrading Configurations

Certain scenarios may require that you modify configurations that Ambari did not upgrade automatically.

Upgrading SmartSense

If your cluster includes the SmartSense service, you must upgrade it after upgrading Ambari.

1.3.1. Upgrading Ambari Infra

If you have Ambari Solr installed, you must upgrade Ambari Infra after upgrading Ambari.

Steps

1. Make sure Ambari Infra services are stopped. From **Ambari Web**, browse to **Services > Ambari Infra** and select **Stop** from the **Service Actions** menu.
2. On every host in your cluster with an Infra Solr Client installed, run the following commands:

For RHEL/CentOS/Oracle Linux:

```
yum clean all
```

```
yum upgrade ambari-infra-solr-client
```

For SLES:

```
zypper clean
```

```
zypper up ambari-infra-solr-client
```

For Ubuntu/Debian:

```
apt-get clean all
```

```
apt-get update
```

```
apt-get install ambari-infra-solr-client
```

3. Execute the following command on all hosts running an Ambari Infra Solr Instance:

For RHEL/CentOS/Oracle Linux:

```
yum upgrade ambari-infra-solr
```

For SLES:

```
zypper up ambari-infra-solr
```

For Ubuntu/Debian:

```
apt-get install ambari-infra-solr
```

4. Start the Ambari Infra services.

From **Ambari Web**, browse to **Services > Ambari Infra** select **Service Actions** then choose **Start**.

1.3.2. Upgrading Ambari Log Search

If you have Ambari Log Search installed, you must upgrade Ambari Log Search after upgrading Ambari.

Prerequisites

Before starting this upgrade, ensure the Ambari Infra components have been upgraded.

Steps

1. Make sure Ambari Log Search service is stopped. From **Ambari Web**, browse to **Services > Log Search** and select **Stop** from the **Service Actions** menu.
2. On every host in your cluster running a Log Feeder, run the following commands:

For RHEL/CentOS/Oracle Linux:

```
yum clean all
```

```
yum upgrade ambari-logsearch-logfeeder
```

For SLES:

```
zypper clean
```

```
zypper up ambari-logsearch-logfeeder
```

For Ubuntu/Debian:

```
apt-get clean all
```

```
apt-get update
```

```
apt-get install ambari-logsearch-logfeeder
```

3. Execute the following command on all hosts running the Log Search Server:

For RHEL/CentOS/Oracle Linux:

```
yum upgrade ambari-logsearch-portal
```

For SLES:

```
zypper up ambari-logsearch-portal
```

For Ubuntu/Debian:

```
apt-get install ambari-logsearch-portal
```

4. Start Log Search Service.

From **Ambari Web**, browse to **Services > Log Search** select **Service Actions** then choose **Start**.

1.3.3. Upgrading Ambari Metrics

Prerequisites

Upgrade to Ambari 2.5 and perform needed post-upgrade checks. Make sure all services are up and healthy.

Steps

1. Make sure Ambari Metrics service is stopped. From **Ambari Web**, browse to **Services > Ambari Metrics** and select **Stop** from the **Service Actions** menu.
2. On every host in your cluster running a Metrics Monitor, run the following commands:

For RHEL/CentOS/Oracle Linux:

```
yum clean all
```

```
yum upgrade ambari-metrics-monitor ambari-metrics-hadoop-sink
```

For SLES:

```
zypper clean
```

```
zypper up ambari-metrics-monitor ambari-metrics-hadoop-sink
```

For Ubuntu/Debian:

```
apt-get clean all
```

```
apt-get update
```

```
apt-get install ambari-metrics-assembly
```

3. Execute the following command on all hosts running the Metrics Collector:

For RHEL/CentOS/Oracle Linux:

```
yum upgrade ambari-metrics-collector
```

For SLES:

```
zypper up ambari-metrics-collector
```

4. Execute the following command on the host running the Grafana component:

For RHEL/CentOS/Oracle Linux:

```
yum upgrade ambari-metrics-grafana
```

For SLES:

5. Start Ambari Metrics Service.

From **Ambari Web**, browse to **Services > Ambari Metrics** select **Service Actions** then choose **Start**.

Updated Ambari Metrics Sink jars will be installed on all hosts and you must restart each service to pick up the latest sink implementations.

Please wait to restart all services until after you have completed all applicable post-upgrade tasks, for example: HDFS, YARN, Kafka, HBase, Flume, Storm.

Next Steps

- Restart services, only after you complete all applicable, post-upgrade tasks.



Note

New Ambari Metrics Sinks will not be activated until all services are restarted.

- If you are upgrading from Ambari 2.2.1 or earlier, and your Ambari Metrics service does not contain Grafana, proceed to add Grafana to Ambari Metrics.

1.3.3.1. Adding Grafana to Ambari Metrics

As of Ambari 2.4, Grafana is included as a component of Ambari Metrics. You must add Grafana to the system and install Grafana on a host in the cluster.



Note

When using the API commands below, be sure to replace the **ambari.server** with the Ambari Server hostname, **cluster.name** with your cluster name and **host.name** with the host where you will run Grafana. This can be the same host that is running the Metrics Collector.

1. Upgrade to Ambari 2.5 and perform needed post-upgrade checks. Make sure all services are up and healthy.
2. Add the METRICS_GRAFANA component to Ambari:

```
curl -u admin:admin -H "X-Requested-By:ambari" -i -X POST
http://ambari.server:8080/api/v1/clusters/cluster.name/services/
AMBARI_METRICS/components/METRICS_GRAFANA
```

3. Add METRICS_GRAFANA to a host in the cluster.

```
curl -u admin:admin -H "X-Requested-By:ambari" -i -X POST -d
'{"host_components":[{"HostRoles":{"component_name":"METRICS_GRAFANA"}}]}'
http://ambari.server:8080/api/v1/clusters/cluster.name/hosts?Hosts/
host_name=host.name
```

4. From **Ambari Web**, browse to **Services > Ambari Metrics** and you will see Grafana is in the **Install Pending...** state.

You need to complete the configuration of Grafana before installing and starting.

5. To complete the configuration, click on **Services > Ambari Metrics > Configs** and enter the default Grafana Admin Password in the **General** section. Click **Save**.
6. Browse to **Hosts > host.name** (the **host.name** used in the API call where you added Grafana). You will see the Grafana component is in an **Install Pending...** state. Use the **Install Pending...** action button and select **Re-install**.

 Grafana / Ambari Metrics

Install Pending... ▾

7. Once the install operation completes, select **Start** to start Grafana.
8. To access Grafana, browse to **Services > Ambari Metrics**, select **Quick Links** and then click **Grafana**.

1.3.4. Upgrading Configurations

This section describes potential cluster configuration updates that may be required.

[Upgrading Kerberos krb5.conf \[12\]](#)

[Upgrading Log Rotation Configuration \[12\]](#)

1.3.4.1. Upgrading Kerberos krb5.conf

Ambari has added support for handling more than one KDC host . Only one kadmin host is supported by the Kerberos infrastructure. This required modifications for the **krb5.conf** template. In order for Ambari to properly construct the krb5.conf configuration file, make the following configuration change if your cluster meets all of these criteria:

- Kerberos is enabled and Ambari is configured for automated setup, and
- Ambari is managing the krb5.conf, and
- You **have modified** the krb5.conf template content from the default content. If you have not modified the default content, Ambari will automatically update the template content as part of upgrade and these configuration updates do not need to be applied manually.

If you meet all of the above criteria, you must update the **krb5.conf** template content found in **Services > Kerberos > Advanced**:

Original Template Entry	Updated Template Entry
admin_server = {{ admin_server_host default(kdc_host, True)}}}	admin_server = {{ admin_server_host default(kdc_host_list[0] trim(), True)}}}
kdc = {{kdc_host}}	{% for kdc_host in kdc_host_list %} kdc = {{kdc_host trim()}} {%- endfor -%}

1.3.4.2. Upgrading Log Rotation Configuration

Ambari 2.5.0 provides a simplified log rotation configuration. These changes will be made automatically during your next stack upgrade, but are not automatically made during the Ambari upgrade. After upgrading Ambari from version 2.x to 2.5.0, if you want to utilize

the simplified log rotation configuration, you must update configurations for all services in your cluster, using the following steps:

Steps

1. ZooKeeper

- a. In **Ambari Web**, browse to **ZooKeeper > Configs**.
- b. Scroll down to **Custom zookeeper-log4j**.
- c. In **Custom zookeeper-log4j**, click **Add Property**.
- d. In **Add Property**, type the following properties and values:

zookeeper_log_max_backup_size=10

zookeeper_log_number_of_backup_files=10

For example:

- e. Click **Add**.
- f. Browse to **Advanced zookeeper-log4j**.
- g. In **Advanced zookeeper-log4j content section**, find and replace the following properties and values:

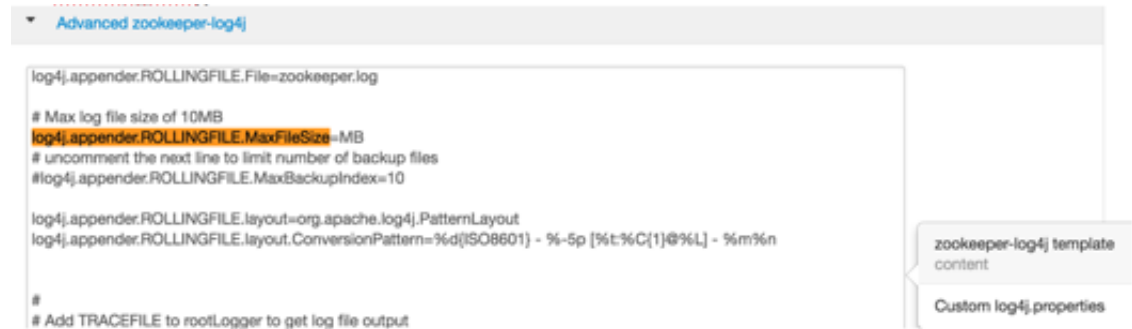
Find: log4j.appender.ROLLINGFILE.MaxFileSize=<value>

Replace:
log4j.appender.ROLLINGFILE.MaxFileSize={{ zookeeper_log_number_of_backup_files }}MB

Find: #log4j.appender.ROLLINGFILE.MaxBackupIndex=<value>MB

Replace:
#log4j.appender.ROLLINGFILE.MaxBackupIndex={{ zookeeper_log_number_of_backup_files }}

For example:



```

log4j.appender.ROLLINGFILE.File=zookeeper.log

# Max log file size of 10MB
log4j.appender.ROLLINGFILE.MaxFileSize=MB
# uncomment the next line to limit number of backup files
#log4j.appender.ROLLINGFILE.MaxBackupIndex=10

log4j.appender.ROLLINGFILE.layout=org.apache.log4j.PatternLayout
log4j.appender.ROLLINGFILE.layout.ConversionPattern=%d{ISO8601} - %-5p [%t:%C{1}@%L] - %m%n

#
# Add TRACEFILE to rootLogger to get log file output

```

h. In **Configs**, click **Save**.

For example:



```

log4j.appender.CONSOLE.Threshold=INFO
log4j.appender.CONSOLE.layout=org.apache.log4j.PatternLayout
log4j.appender.CONSOLE.layout.ConversionPattern=%d{ISO8601} - %-5p [%t:%C{1}@%L] - %m%n

#
# Add ROLLINGFILE to rootLogger to get log file output
# Log DEBUG level and above messages to a log file
log4j.appender.ROLLINGFILE=org.apache.log4j.RollingFileAppender
log4j.appender.ROLLINGFILE.Threshold=DEBUG
log4j.appender.ROLLINGFILE.File=zookeeper.log

# Max log file size of 10MB
log4j.appender.ROLLINGFILE.MaxFileSize={zookeeper_log_max_backup_size}MB
# uncomment the next line to limit number of backup files
#log4j.appender.ROLLINGFILE.MaxBackupIndex={zookeeper_log_number_of_backup_files}

log4j.appender.ROLLINGFILE.layout=org.apache.log4j.PatternLayout
log4j.appender.ROLLINGFILE.layout.ConversionPattern=%d{ISO8601} - %-5p [%t:%C{1}@%L] - %m%n

```

i. Restart **ZooKeeper**, as prompted.

2. Kafka

a. In **Ambari Web**, browse to **Kafka > Configs**.

b. Scroll down to **Custom Kafka-log4j**.

c. In **Custom Kafka-log4j**, click **Add Property**.

d. In **Add Property**, type the following properties and values:

kafka_log_maxfilesize=256

kafka_log_maxbackupindex=20

controller_log_maxfilesize=256

controller_log_maxbackupindex=20

- e. Click **Add**.
- f. Browse to **Advanced kafka-log4j**.
- g. In **Advanced kafka-log4j content section**, find and replace the following properties and values:

Find: log4j.appender.kafkaAppender=org.apache.log4j.DailyRollingFileAppender

Add: log4j.appender.kafkaAppender.MaxFileSize = {{kafka_log_maxfilesize}}MB

Add: log4j.appender.kafkaAppender.MaxBackupIndex =
{{kafka_log_maxbackupindex}}MB

Find: log4j.appender.controllerAppender=org.apache.log4j.DailyRollingFileAppender

Add: log4j.appender.controllerAppender.MaxFileSize =
{{controller_log_maxfilesize}}MB

Add: log4j.appender.controllerAppender.MaxBackupIndex =
{{controller_log_maxbackupindex}}

- h. In **Configs**, click **Save**.
- i. Restart **Kafka**, as prompted.

3. Ranger

- a. In **Ambari Web**, browse to **Ranger > Configs > Advanced**.
- b. Scroll down to **Custom admin-log4j**.
- c. In **Custom admin-log4j**, click **Add Property**.
- d. In **Add Property**, type the following properties and values:

ranger_xa_log_maxfilesize=256

ranger_xa_log_maxbackupindex=20

- e. Click **Add**.
- f. Browse to **Advanced admin-log4j**.
- g. In **Advanced admin-log4j content section**, find and replace the following properties and values:

Find: log4j.appender.xa_log_appender=org.apache.log4j.DailyRollingFileAppender

Add:
log4j.appender.xa_log_appender.MaxFileSize={{ranger_xa_log_maxfilesize}}MB

Add:

```
log4j.appender.xa_log_appender.MaxBackupIndex={{ranger_xa_log_maxbackupindex}}
```

h. Scroll down to **Custom usersync-log4j**.

i. In **Custom usersync-log4j**, click **Add Property**.

j. In **Add Property**, type the following properties and values:

```
ranger_usersync_log_maxfilesize=256
```

```
ranger_usersync_log_number_of_backup_files=20
```

k. Click **Add**.

l. Browse to **Advanced usersync-log4j**.

m. In **Advanced usersync-log4j content section**, find and replace the following properties and values:

Find: log4j.appender.logFile=org.apache.log4j.DailyRollingFileAppender

Add: log4j.appender.logFile.MaxFileSize = {{ranger_usersync_log_maxfilesize}}MB

Add: log4j.appender.logFile.MaxBackupIndex =
{{ranger_usersync_log_number_of_backup_files}}

n. Scroll down to **Custom tagsync-log4j**.

o. In **Custom tagsync-log4j**, click **Add Property**.

p. In **Add Property**, type the following properties and values:

```
ranger_tagsync_log_maxfilesize=256
```

```
ranger_tagsync_log_number_of_backup_files=20
```

q. Click **Add**.

r. Browse to **Advanced tagsync-log4j**.

s. In **Advanced tagsync-log4j content section**, find and replace the following properties and values:

Find: log4j.appender.logFile=org.apache.log4j.DailyRollingFileAppender

Add: log4j.appender.logFile.MaxFileSize = {{ranger_tagsync_log_maxfilesize}}MB

Add: log4j.appender.logFile.MaxBackupIndex =
{{ranger_tagsync_log_number_of_backup_files}}

t. In **Configs**, click **Save**.

u. Restart **Ranger**, as prompted.

4. Ranger-KMS

a. In **Ambari Web**, browse to **Ranger-KMS > Configs > Advanced**.

b. Scroll down to **Custom kms-log4j**.

c. In **Custom kms-log4j**, click **Add Property**.

d. In **Add Property**, type the following properties and values:

```
ranger_kms_log_maxfilesize=256
```

```
ranger_kms_log_maxbackupindex=20
```

```
ranger_kms_audit_log_maxfilesize=256
```

```
ranger_kms_audit_log_maxbackupindex=20
```

e. Click **Add**.

f. Browse to **Advanced kms-log4j**.

g. In **Advanced kms-log4j content section**, find and replace the following properties and values:

Find: log4j.appender.kms=org.apache.log4j.DailyRollingFileAppender

Add: log4j.appender.kms.MaxFileSize = {{ranger_kms_log_maxfilesize}}MB

Add: log4j.appender.kms.MaxBackupIndex = {{ranger_kms_log_maxbackupindex}}

Find: log4j.appender.kms-audit=org.apache.log4j.DailyRollingFileAppender

Add: log4j.appender.kms-audit.MaxFileSize={{ranger_kms_audit_log_maxfilesize}}MB

Add: log4j.appender.kms-audit.MaxBackupIndex = {{ranger_kms_audit_log_maxbackupindex}}

h. In **Configs**, click **Save**.

i. Restart **Ranger-KMS**.

5. Storm

a. In **Ambari Web**, browse to **Storm > Configs**.

b. Scroll down to **Custom cluster-log4j property**.

c. In **Custom cluster-log4j property**, click **Add Property**.

d. In **Add Property**, type the following properties and values:

```
storm_a1_maxfilesize=100
```

```
storm_a1_maxbackupindex=9
```

- e. Click **Add**.
- f. Browse to **Advanced storm-cluster-log4j** .
- g. In **Advanced storm-cluster-log4j** *content section*, find and replace the following properties and values:

Find: In RollingFile="A1"<SizeBasedTriggeringPolicy size="<value>MB"/>

Replace: <SizeBasedTriggeringPolicy size="{{storm_a1_maxfilesize}} MB"/>

Find: In RollingFile="A1"<DefaultRolloverStrategy max="<value>"/>

Replace: <DefaultRolloverStrategy max="{{storm_a1_maxbackupindex}}"/>

- h. Scroll down to **Custom worker-log4j** property.
- i. In **Custom worker-log4j** property, click **Add Property**.
- j. In **Add Property**, type the following properties and values:

```
storm_wrkr_a1_maxfilesize=100
```

```
storm_wrkr_a1_maxbackupindex=9
```

```
storm_wrkr_out_maxfilesize=100
```

```
storm_wrkr_out_maxbackupindex=4
```

```
storm_wrkr_err_maxfilesize=100
```

```
storm_wrkr_err_maxbackupindex=4
```

- k. Click **Add**.
- l. Browse to **Advanced storm-worker-log4j** .
- m. In **Advanced storm-worker-log4j** *content section*, find and replace the following properties and values:

Find: In RollingFile="A1"<SizeBasedTriggeringPolicy size="<value> MB"/>

Replace: <SizeBasedTriggeringPolicy size="{{storm_wrkr_a1_maxfilesize}} MB"/>

Find: In RollingFile="A1"<DefaultRolloverStrategy max="<value>"/>

Replace: <DefaultRolloverStrategy max="{{storm_wrkr_a1_maxbackupindex}}"/>

Find: In RollingFile="STDOUT"<SizeBasedTriggeringPolicy size="<value>" MB/>

Replace: <SizeBasedTriggeringPolicy size="{{ storm_wrkr_out_maxfilesize }}" MB"/>

Find: In RollingFile="STDOUT"<DefaultRolloverStrategy max="<value>"/>

Replace: <DefaultRolloverStrategy max="{{ storm_wrkr_out_maxbackupindex }}" />

Find: In RollingFile="STDERR"<SizeBasedTriggeringPolicy size="<value>" MB/>

Replace: <SizeBasedTriggeringPolicy size="{{ storm_wrkr_err_maxfilesize }}" MB"/>

Find: In RollingFile="STDOUT"<DefaultRolloverStrategy max="<value>"/>

Replace: <DefaultRolloverStrategy max="{{ storm_wrkr_err_maxbackupindex }}" />

n. In **Configs**, click **Save**.

o. Restart **Storm**, as prompted.

1.3.5. Upgrading SmartSense

If your cluster includes the SmartSense service, you must upgrade it after upgrading Ambari.

More Information

[Upgrading SmartSense](#)

Next Steps

Restart services.

2. Upgrading to HDP 2.6.3

If you already have HDP 2.6.0 installed, upgrading your cluster to HDP 2.6.3 means:

- Keeping the same configuration files you used for HDP 2.6.0.
- Keeping the same data and metadata in the same location you used for HDP 2.6.0
- Installing any new components (added for the first time in HDP 2.6.0) side-by-side with existing components

2.1. Before you begin

- Ensure that you know which HDP components you need to upgrade at your installation.
- Decide whether you are going to upgrade using a [local repository](#) or a [remote repository](#).
- If you are using the Falcon service, install the Berkeley DB prior to performing an upgrade.

See the [Prerequisite to Installing or Upgrading Falcon](#) in the Data Movement and Integration guide.

2.2. Upgrade options

- If you are upgrading your cluster manually, use the [Non-Ambari Upgrade Guide](#).
- If you are upgrading your cluster through Ambari, use the [Ambari Upgrade Guide](#)

More information:

- [Upgrading HDP](#)
- [Register and Install HDP Version](#)
- [Obtain the HDP repos](#)

3. Installing Databases

Schema Registry, SAM, Druid, and Superset require a relational data store to store metadata. You can use either MySQL, Postgres, or Oracle for this. This chapter describes how to install either MySQL, Postgres, and Oracle and how create a databases for SAM and 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 or MySQL; both are not necessary. It is recommended that you use MySQL.



Warning

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 and Configuring MySQL

- [Installing MySQL \[21\]](#)
- [Configuring SAM and Schema Registry Metadata Stores in MySQL \[22\]](#)
- [Configuring Druid and Superset Metadata Stores in MySQL \[23\]](#)

Installing and Configuring Postgres

- [Install Postgres \[23\]](#)
- [Configure Postgres to Allow Remote Connections \[24\]](#)
- [Configure SAM and Schema Registry Metadata Stores in Postgres \[25\]](#)
- [Configure Druid and Superset Metadata Stores in Postgres \[25\]](#)

Using an Oracle Database

- [Specifying an Oracle Database to use with SAM and Schema Registry](#)

3.1. Installing MySQL

About This Task

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
```

Steps

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 a 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, followed by the password you obtained in the previous step. MySQL will ask you to change the password.

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

3.2. Configuring SAM and Schema Registry Metadata Stores in MySQL

Steps

1. Launch the MySQL monitor:

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

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

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

3. Create Schema Registry and SAM user accounts, replacing the last 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;
```

3.3. Configuring Druid and Superset Metadata Stores in MySQL

About This Task

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.

Steps

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 last string with your password:

```
CREATE USER 'druid'@'%' IDENTIFIED BY '9oNio)ex1ndL';  
CREATE USER 'superset'@'%' IDENTIFIED BY '9oNio)ex1ndL';
```

4. Assign privileges to the druid account:

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

5. Commit the operation:

```
commit;
```

3.4. Install Postgres

Before You Begin

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



Warning

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.

Steps

1. Install 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
```

2. Install 9.5+ Postgres database:

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

3. 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
```

4. 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
```

5. Verify that you can log in:

```
sudo su postgres  
psql
```

3.5. Configure Postgres to Allow Remote Connections

About This Task

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.

Steps

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
```

3.6. Configure SAM and Schema Registry Metadata Stores in Postgres

About This Task

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.

Steps

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;
```

3.7. Configure Druid and Superset Metadata Stores in Postgres

About This Task

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.

Steps

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;
```

3.8. Specifying an Oracle Database to Use with SAM and Schema Registry

About This Task

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

Prerequisites

You have an Oracle database installed and configured.

Steps

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.

3.9. Switching to an Oracle Database After Installation

About This Task

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

Prerequisites

You have an Oracle database installed and configured.

Steps

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
```

4. 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/.
```

5. 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.

6. Confirm that new tables are created in the Oracle database.
7. From Ambari, restart SAM or Schema Registry.
8. 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
```

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

4. Installing the HDF Management Pack

About This Task

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.

Steps

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
```

More Information

[HDF Release Notes](#)

5. Update the HDF Base URL

About This Task

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.

Steps

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**.

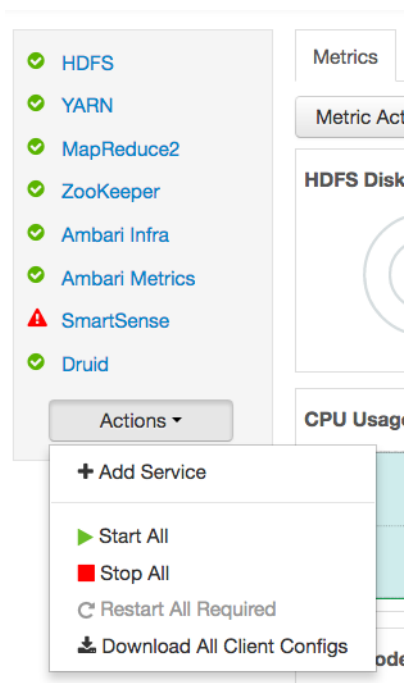
6. Add HDF Services to an HDP Cluster

About This Task

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

Steps

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**.



2. Chose HDF services select the HDF Services needed (NiFi, Storm, Kafka, Streaming Analytics Manager, and Schema Registry)
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.

7. Configure HDF Components

You can customize your Hortonworks DataFlow (HDF) component configurations either during or after installation. During installation, you customize HDF component configurations in the **Customize Services** page of the installation wizard. After installation, you can navigate to Services > Configs in the Ambari dashboard.

- [Configure Schema Registry \[31\]](#)
- [Configure SAM \[32\]](#)
- [Configure NiFi \[33\]](#)
- [Configure Kafka \[33\]](#)
- [Configure Storm \[34\]](#)
- [Deploy the Cluster Services \[34\]](#)
- [Access the UI for Deployed Services \[35\]](#)

7.1. Configure Schema Registry

About This Task

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

Steps

1. In the **Customize Services** step, navigate to the **REGISTRY CONFIG** section of the **Registry** tab.
2. Select **Jar Storage Type** and then the storage type that you want to use.

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. Configure **jar.storage** to the directory in which 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 registry storage connector URL has the fully qualified name of the host on which the database was installed and the connector url and default port for the database selected.

Example

```
MYSQL example:
```

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

```
Postgres Example:
```

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

More Information

[Installing Databases \[21\]](#)

7.2. Configure SAM

About This Task

When you configure Hortonworks Streaming Analytics Manager (SAM), you must provide information about the metadata store database, configure a connection with Schema Registry, and establish the URL for the Druid Supersets.

Steps

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. Configure **jar.storage** to the directory on which 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

- `$FQDN_REGISTRY_HOST` specifies the host on which you are running Schema Registry and
- `$REGISTRY_PORT` specifies the Schema Registry port number, as in the following example:

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

You can find the Schema Registry port in the **REGISTRY_CONFIG** section of the **Registry** tab.

7. Configure the **STREAMLINE STORAGE** configurations based on the database you created to use as a SAM metadata store.
8. Ensure that the registry storage connector URL has the fully qualified name of the host on which the database was installed and the connector url and default port for the database selected.

Example

MYSQL example:

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

Postgres Example:

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

More Information

[Installing Databases \[21\]](#)

7.3. Configure NiFi

About This Task

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 set before proceeding.

Steps

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.

7.4. Configure Kafka

About This Task

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

Steps

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 Apache Storm to meet your operational objectives.

More Information

[Configuring Kafka for Production Environments](#)

7.5. Configure Storm

About This Task

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

Steps

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.

More Information

[Configuring Storm for Production Environments](#)

7.6. Deploy the Cluster Services

After you finish the wizard and deploy the cluster, some services might fail to start. If this is the case, you can start those services individually by launching them from the Ambari dashboard Services pane.

Steps

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**.

Result

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

7.7. Access the UI for Deployed Services

About This Task

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

Steps

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**.

Result

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

8. Configuring Schema Registry and SAM for High Availability

About This Task

You can configure Schema Registry and SAM for high availability.

Steps for Configuring SAM for HA

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**.

Steps for Configuring Schema Registry for HA

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**.

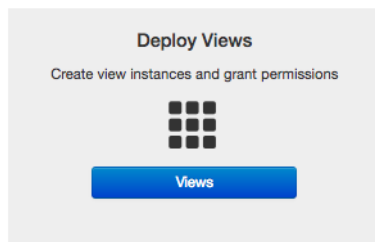
9. Install the Storm Ambari View

About This Task

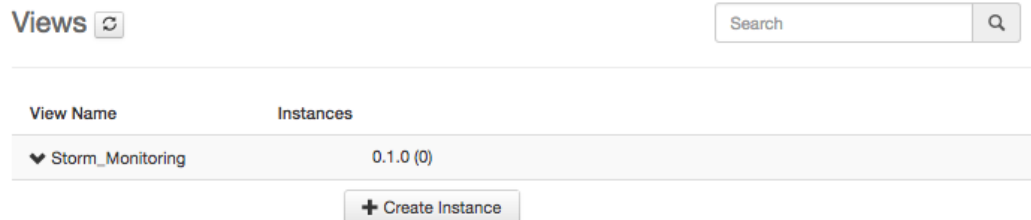
The Storm Ambari view provides you a number of different troubleshooting and debugging tools.

Steps

1. From the **admin** drop-down, select **Manage Ambari**.
2. Click the **Views** button.



3. From the list of available Views, expand **Storm_Monitoring** and click **+ Create Instance**.



4. Configure the Storm Ambari View.

Views / Create Instance

View **Storm_Monitoring**

Version

Details

Instance Name*

Display Name*

Description*

Visible

Settings

Storm Hostname*

Storm Port*

SSL Enabled*

- a. Instance Name and Display Name may not have an spaces.
- b. The Storm Hostname refers to the host where the Storm UI Server is deployed.
- c. The Storm port is the Storm UI port server (keep it as default 8744 if you have not changed it).
- d. Click Save.

Result

After saving it, you should see a menu item for the Storm Ambari View.

The screenshot shows the Ambari dashboard with several widgets. On the left, there are four circular progress indicators: EXECUTOR (0%), TASKS (0%), SUPERVISOR (100%), and SLOTS (0%). Below these is a 'Nimbus Summary' widget showing the host 'vett-hdf-sam1.field.hortonworks.com:6627' with a 'Leader' status and an uptime of '1h 42m 13s'. On the right, there is a 'Topology Listing' table with no data, and a 'Supervisor Summary' table with the following data:

Host	Slots	CPU	Memory	Uptime
172.26.246.18	0%	0%	0%	1h 41m 33s
vett-hdf-sam8.field.hortonworks.com	0%	0%	0%	1h 41m 33s
vett-hdf-sam9.field.hortonworks.com	0%	0%	0%	1h 41m 24s

10. Using a Local Repository

Local repositories are frequently used in enterprise clusters that have limited outbound internet access. In these scenarios, having packages available locally provides more governance, and better installation performance. These repositories are used heavily during installation for package distribution, as well as post-install for routine cluster operations such as service start/restart operations. The following section describes the steps required to setup and use a local repository:

- Obtain Public Repositories from the [HDF Release Notes](#)
- Set up a local repository having:
 - [Setting Up a Local Repository with No Internet Access \[40\]](#)
 - [Setting up a Local Repository With Temporary Internet Access \[42\]](#)
- [Preparing The Ambari Repository Configuration File \[44\]](#)

10.1. Setting Up a Local Repository

Based on your Internet access, choose one of the following options:

- No Internet Access

This option involves downloading the repository tarball, moving the tarball to the selected mirror server in your cluster, and extracting to create the repository.

- Temporary Internet Access

This option involves using your temporary Internet access to sync (using reposync) the software packages to your selected mirror server and creating the repository.

Both options proceed in a similar, straightforward way. Setting up for each option presents some key differences, as described in the following sections:

- [Getting Started Setting Up a Local Repository \[39\]](#)
- [Setting Up a Local Repository with No Internet Access \[40\]](#)
- [Setting up a Local Repository With Temporary Internet Access \[42\]](#)

10.2. Getting Started Setting Up a Local Repository

Before setting up your local repository, complete the following:

Prerequisites

- Select an existing server in, or accessible to the cluster, that runs a supported operating system.

- Enable network access from all hosts in your cluster to the mirror server.
- Ensure the mirror server has a package manager installed such as yum (RHEL / CentOS / Oracle Linux), zypper (SLES), or apt-get (Debian/Ubuntu).
- **Optional:** If your repository has temporary Internet access, and you are using RHEL/CentOS/Oracle Linux as your OS, install yum utilities:

```
yum install yum-utils createrepo
```

To begin setting up your local repository:

Steps

1. Create an HTTP server.
 - a. On the mirror server, install an HTTP server (such as Apache httpd) using the instructions provided on the Apache community website.
 - b. Activate this web server.
 - c. Ensure that any firewall settings allow inbound HTTP access from your cluster nodes to your mirror server.



Note

If you are using Amazon EC2, make sure that SELinux is disabled.

2. On your mirror server, create a directory for your web server.

- For example, from a shell window, type:

For RHEL/CentOS/Oracle Linux: `mkdir -p /var/www/html/`

For SLES: `mkdir -p /srv/www/htdocs/rpms`

For Debian/Ubuntu: `mkdir -p /var/www/html/`

- If you are using a symlink, enable the `followsymlinks` on your web server.

Next Steps

After you have completed the steps in this section, move on to specific set up for your repository internet access type.

More Information

httpd.apache.org/download.cgi

10.2.1. Setting Up a Local Repository with No Internet Access

Prerequisites

Complete the [Getting Started Setting up a Local Repository](#) procedure.

Steps

1. Obtain the tarball for the repository you would like to create.
2. Copy the repository tarballs to the web server directory and untar the archive.
 - a. Browse to the web server directory you created.

For RHEL/CentOS/Oracle Linux: `cd /var/www/html/`

For SLES: `cd /srv/www/htdocs/rpms`

For Debian/Ubuntu: `cd /var/www/html/`

- b. Untar the repository tarballs to the following locations: where `<web.server>`, `<web.server.directory>`, `<OS>`, `<version>`, and `<latest.version>` represent the name, home directory, operating system type, version, and most recent release version, respectively.

Untar Locations for a Local Repository - No Internet Access

Ambari Repository	Untar under <code><web.server.directory></code>
HDF Stack Repositories	Create a directory and untar it under <code><web.server.directory>/hdf</code>
HDP Stack Repositories	Create directory and untar under <code><web.server.directory>/hdp</code>

3. Confirm you can browse to the newly created local repositories.

URLs for a Local Repository - No Internet Access

Ambari Base URL	<code>http://<web.server>/Ambari-2.6.0.0/<OS></code>
HDF Base URL	<code>http://<web.server>/hdf/HDF/<OS>/3.x/updates/<latest.version></code>
HDP Base URL	<code>http://<web.server>/hdp/HDP/<OS>/2.x/updates/<latest.version></code>
HDP-UTILS Base URL	<code>http://<web.server>/hdp/HDP-UTILS-<version>/repos/<OS></code>

Where:

- `<web.server>` – The FQDN of the web server host
- `<version>` – The Hortonworks stack version number
- `<OS>` – centos6, centos7, sles11, sles12, ubuntu14, ubuntu16, or debian7



Important

Be sure to record these Base URLs. You will need them when installing Ambari and the cluster.

4. Optional: If you have multiple repositories configured in your environment, deploy the following plug-in on all the nodes in your cluster.

a. Install the plug-in.

For RHEL and CentOS 7: `yum install yum-plugin-priorities`

For RHEL and CentOS 6: `yum install yum-plugin-priorities`

b. Edit the `/etc/yum/pluginconf.d/priorities.conf` file to add the following:

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

10.2.2. Setting up a Local Repository With Temporary Internet Access

Prerequisites

Complete the [Getting Started Setting up a Local Repository](#) procedure.

Steps

1. Put the repository configuration files for Ambari and the Stack in place on the host.
2. Confirm availability of the repositories.

For RHEL/CentOS/Oracle Linux: `yum repolist`

For SLES: `zypper repos`

For Debian/Ubuntu: `dpkg-list`

3. Synchronize the repository contents to your mirror server.

- Browse to the web server directory:

For RHEL/CentOS/Oracle Linux: `cd /var/www/html`

For SLES: `cd /srv/www/htdocs/rpms`

For Debian/Ubuntu: `cd /var/www/html`

- For Ambari, create `ambari` directory and `reposync`.

```
mkdir -p ambari/<OS>
```

```
cd ambari/<OS>
```

```
reposync -r Updates-Ambari-2.6.0.0
```

where `<OS>` is `centos6`, `centos7`, `sles11`, `sles12`, `ubuntu14`, `ubuntu16`, or `debian7`.

- For HDP Stack Repositories, create `hdp` directory and `reposync`.

```
mkdir -p hdp/<OS>
cd hdp/<OS>
reposync -r HDP-<latest.version>
reposync -r HDP-UTILS-<version>
```

- For HDF Stack Repositories, create an hdf directory and reposync.

```
mkdir -p hdf/<OS>
cd hdf/<OS>
reposync -r HDF-<latest.version>
```

4. Generate the repository metadata.

For Ambari:

```
createrepo <web.server.directory>/ambari/
<OS>/Updates-Ambari-2.6.0.0
```

For HDP Stack Repositories:

```
createrepo <web.server.directory>/hdp/<OS>/
HDP-<latest.version>
```

```
createrepo <web.server.directory>/hdp/<OS>/
HDP-UTILS-<version>
```

For HDF Stack Repositories:

```
createrepo <web.server.directory>/hdf/<OS>/
HDF-<latest.version>
```

5. Confirm that you can browse to the newly created repository.

URLs for the New Repository

Ambari Base URL <http://<web.server>/ambari/<OS>/Updates-Ambari-2.6.0.0>

HDF Base URL <http://<web.server>/hdf/<OS>/HDF-<latest.version>>

HDP Base URL <http://<web.server>/hdp/<OS>/HDP-<latest.version>>

HDP-UTILS Base URL <http://<web.server>/hdp/<OS>/HDP-UTILS-<version>>

Where:

- <web.server> – The FQDN of the web server host
- <version> – The Hortonworks stack version number
- <OS> – centos6, centos7, sles11, sles12, ubuntu14, ubuntu16, or debian7



Important

Be sure to record these Base URLs. You will need them when installing Ambari and the Cluster.

6. Optional. If you have multiple repositories configured in your environment, deploy the following plug-in on all the nodes in your cluster.

- a. Install the plug-in.

For RHEL and CentOS 7: `yum install yum-plugin-priorities`

For RHEL and CentOS 6: `yum install yum-plugin-priorities`

- b. Edit the `/etc/yum/pluginconf.d/priorities.conf` file to add the following:

```
[main]
```

```
enabled=1
```

```
gpgcheck=0
```

10.3. Preparing The Ambari Repository Configuration File

Steps

1. Download the `ambari.repo` file from the public repository.

```
http://public-repo-1.hortonworks.com/ambari/<OS>/2.x/updates/2.6.0.0/ambari.repo
```

where `<OS>` is `centos6`, `centos7`, `sles11`, `sles12`, `ubuntu14`, `ubuntu16`, or `debian7`.

2. Edit the `ambari.repo` file and replace the Ambari Base URL `baseurl` obtained when setting up your local repository.



Note

You can disable the GPG check by setting `gpgcheck=0`. Alternatively, you can keep the check enabled but replace the `gpgkey` with the URL to the GPG-KEY in your local repository.

```
[Updates-Ambari-2.6.0.0]
```

```
name=Ambari-2.6.0.0-Updates
```

```
baseurl=INSERT-BASE-URL
```

```
gpgcheck=1
```

```
gpgkey=http://public-repo-1.hortonworks.com/ambari/centos6/RPM-GPG-KEY/RPM-GPG-KEY-Jenkins
```

```
enabled=1
```

```
priority=1
```

Base URL for a Local Repository

Built with Repository Tarball (No Internet Access) `http://<web.server>/Ambari-2.6.0.0/<OS>`

Built with Repository File <http://<web.server>/ambari/<OS>/Updates-Ambari-2.6.0.0>
(Temporary Internet Access)

where <web.server> = FQDN of the web server host, and <OS> is centos6, centos7, sles11, sles12, ubuntu12, ubuntu14, or debian7.

3. Place the `ambari.repo` file on the machine you plan to use for the Ambari Server.

For RHEL/CentOS/Oracle Linux: `/etc/yum/repos.d/ambari.repo`

For SLES: `/etc/zypp/repos.d/ambari.repo`

For Debian/Ubuntu: `/etc/apt/sources.list.d/ambari.list`

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


```
[main]
```

```
enabled=1
```

```
gpgcheck=0
```


11. Navigating the HDF Library

To navigate the Hortonworks DataFlow (HDF) documentation library, begin by deciding your current goal.

If you want to...	See this document...
Install or upgrade an HDF cluster using Apache Ambari	<ul style="list-style-type: none"> • Release Notes • Support Matrices • Planning Your Deployment • Ambari Upgrade
Manually install or upgrade HDF components  Note This option is not available for Streaming Analytics Manager or Schema Registry.	<ul style="list-style-type: none"> • Command Line Installation • MiNiFi Java Agent Quick Start • Manual Upgrade
Get started with HDF	<ul style="list-style-type: none"> • Getting Started with Apache NFi • Getting Started with Stream Analytics
Use and administer HDF Flow Management capabilities	<ul style="list-style-type: none"> • Apache NiFi User Guide • Apache NiFi Administration Guide • Apache NiFi Developer Guide • Apache NiFi Expression Language Guide • MiNiFi Java Agent Administration Guide
Use and administer HDF Stream Analytics capabilities	<ul style="list-style-type: none"> • Streaming Analytics Manager User Guide • Schema Registry User Guide • Apache Storm Component Guide • Apache Kafka Component Guide