Hortonworks DataFlow

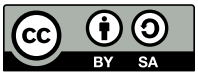
Installing HDF Services on an Existing HDP Cluster

(February 28, 2018)

docs.cloudera.com

Hortonworks DataFlow: Installing HDF Services on an Existing HDP Cluster

Copyright © 2012-2018 Hortonworks, Inc. Some rights reserved.



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

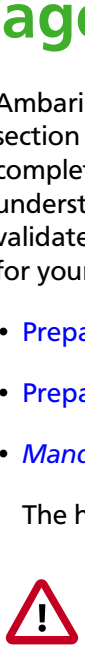
1. Upgrading Ambari and the HDF Management Pack 1
    1.1. Preparing to Upgrade 1
    1.2. Prepare Ambari for Upgrade 2
    1.3. Get the Ambari Repository 3
    1.4. Upgrade Ambari Server 4
    1.5. Upgrade the Ambari Agents 5
    1.6. Upgrade the HDF Management Pack 6
    1.7. Upgrade the Ambari Database Schema 8
    1.8. Restart Ambari 8
    1.9. *Mandatory* Post-Upgrade Tasks 9
        1.9.1. Upgrading Ambari Infra 9
        1.9.2. Upgrading Ambari Log Search 10
        1.9.3. Upgrading Ambari Metrics 11
        1.9.4. Upgrading Configurations 12
        1.9.5. Upgrading SmartSense 19
2. Upgrading to HDP 2.6.4 20
    2.1. Before you begin 20
    2.2. Upgrade options 20
3. Installing the HDF Management Pack 21
4. Update the HDF Base URL 22
5. Add HDF Services to an HDP Cluster 23
6. Configure HDF Components 24
    6.1. Configure NiFi 24
    6.2. Configure NiFi for Atlas Integration 24
    6.3. Configure Kafka 26
    6.4. Configure Storm 26
    6.5. Configure Log Search 26
    6.6. Deploy the Cluster Services 27
    6.7. Access the UI for Deployed Services 27
7. Configuring Schema Registry and SAM for High Availability 28
8. Install the Storm Ambari View 29
9. Using a Local Repository 31
    9.1. Setting Up a Local Repository 31
        9.1.1. Preparing to Set Up a Local Repository 31
        9.1.2. Setting up a Local Repository with Temporary Internet Access 32
        9.1.3. Setting Up a Local Repository with No Internet Access 34
    9.2. Preparing the Ambari Repository Configuration File to Use the Local Repository 36
10. Navigating the HDF Library 38

1. Upgrading Ambari and the HDF Management Pack

Ambari and the 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.

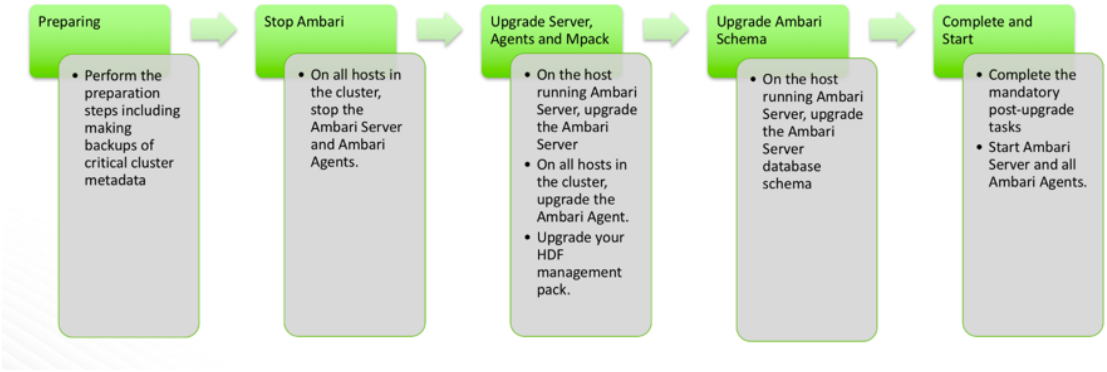
- Preparing to Upgrade [1]
- Prepare Ambari for Upgrade [2]
- *Mandatory* Post-Upgrade Tasks [9]

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 2.6.1.0 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.
- Upgrade Ambari to version 2.5x or 2.6x, based on your current Ambari Server version.

The following table lists recommended (✓), and unsupported (X) upgrade paths.

From / To	2.5.x	2.6.x
2.4.0.x	✓	✓
2.4.2.x	N/A	✓
2.5.x	N/A	✓

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

Prepare Ambari for Upgrade [2]

More Information

Ambari 2.6.1.0 Release Notes

1.2. Prepare Ambari for Upgrade

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
```

1.3. Get the Ambari Repository

1. 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:

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

```
wget -nv http://public-repo-1.hortonworks.com/ambari/centos6/2.x/updates/
2.6.1.0/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/
2.6.1.0/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.
6.1.0/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.
6.1.0/ambari.repo -O /etc/zypp/repos.d/ambari.repo
```

- **For Ubuntu 14:**

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

- **For Ubuntu 16:**

```
wget -nv http://public-repo-1.hortonworks.com/ambari/ubuntu16/2.x/updates/
2.6.1.0/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/
2.6.1.0/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.

1.4. Upgrade Ambari Server

1. 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-2.6.1-143.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**.

2. 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 0:2.6.1-143 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.

1.5. Upgrade the Ambari Agents

1. 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-2.6-143.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
```

2. 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`

1.6. Upgrade the HDF Management Pack

About This Task

A management pack 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. Upgrade the management pack to ensure that you have the latest versions of the available Apache components.

Before You Begin

Get the HDF Management Pack location and build number from the HDF Release Notes.

Steps

1. Back up your Ambari resources folder:

```
cp -r /var/lib/ambari-server/resources /var/lib/ambari-server/resources.
backup
```

2. Upgrade the HDF management pack with the command appropriate for your operating system:

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

```
ambari-server upgrade-mpack \
--mpack=http://public-repo-1.hortonworks.com/HDF/centos6/3.x/updates/
<version>/tars/hdf_ambari_mp/hdf-ambari-mpack-<version>-<build-number>.
tar.gz \
--verbose
```

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

```
ambari-server upgrade-mpack \
--mpack=http://public-repo-1.hortonworks.com/HDF/centos7/3.x/updates/
<version>/tars/hdf_ambari_mp/hdf-ambari-mpack-<version>-<build-number>.
tar.gz \
--verbose
```

   - **SLES 11:**

```
ambari-server upgrade-mpack \
--mpack=http://public-repo-1.hortonworks.com/HDF/suse11sp3/3.x/updates/
<version>/tars/hdf_ambari_mp/hdf-ambari-mpack-<version>-<build-number>.
tar.gz \
--verbose
```

   - **SUSE Linux Enterprise Server (SLES) v12 SP1**

```
ambari-server upgrade-mpack \
--mpack=http://public-repo-1.hortonworks.com/HDF/sles12/3.x/updates/
<version>/tars/hdf_ambari_mp/hdf-ambari-mpack-<version>-<build-number>.
tar.gz \
--verbose
```

   - **Debian 7:**

```
ambari-server upgrade-mpack \
--mpack=http://public-repo-1.hortonworks.com/HDF/debian7/3.x/updates/
<version>/tars/hdf_ambari_mp/hdf-ambari-mpack-<version>-<build-number>.
tar.gz \
--verbose
```

   - **Ubuntu 14:**

```
ambari-server upgrade-mpack \
```

```
--mpack=http://public-repo-1.hortonworks.com/HDF/ubuntu14/3.x/updates/
<version>/tars/hdf_ambari_mp/hdf-ambari-mpack-<version>-<build-number>.
tar.gz \
--verbose
```

- **Ubuntu 16:**

```
ambari-server upgrade-mpack \
--mpack=http://public-repo-1.hortonworks.com/HDF/ubuntu16/3.x/updates/
<version>/tars/hdf_ambari_mp/hdf-ambari-mpack-<version>-<build-number>.
tar.gz \
--verbose
```

1.7. Upgrade the Ambari Database Schema

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

```
ambari-server upgrade
```

2. Start the Ambari Server. On **the host** running Ambari Server:

```
ambari-server start
```

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

```
ambari-agent start
```

4. 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.

1.8. Restart Ambari

1. 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".

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

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

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

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

1.9. *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.
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.

1.9.1. 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**.

1.9.2. 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**.

1.9.3. Upgrading Ambari Metrics

Prerequisites

Ensure 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
```

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

   For RHEL/CentOS/Oracle Linux:

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

   For SLES:

   ```
   zypper up ambari-metrics-grafana
   ```

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.

1.9.4. Upgrading Configurations

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

Upgrading Kerberos krb5.conf [12]

Upgrading Log Rotation Configuration [13]

1.9.4.1. 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 -%}

1.9.4.2. Upgrading Log Rotation Configuration

Ambari 2.6.x provides s 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 2.6.0, 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:

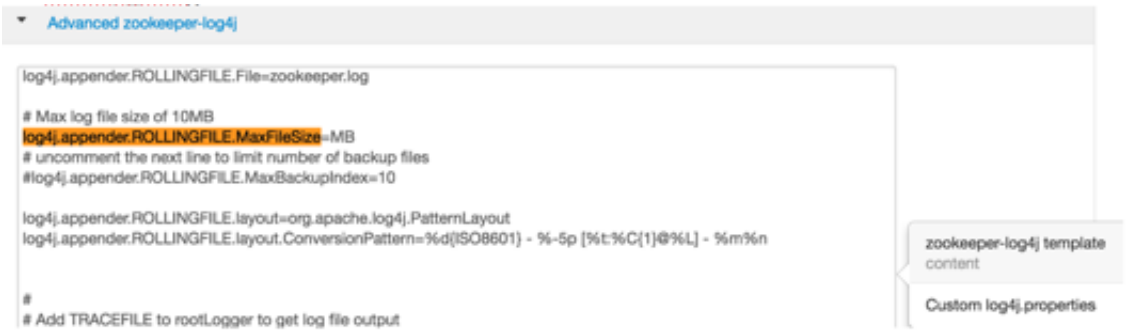
Find: log4j.appender.ROLLINGFILE.MaxFileSize=<value>

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

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

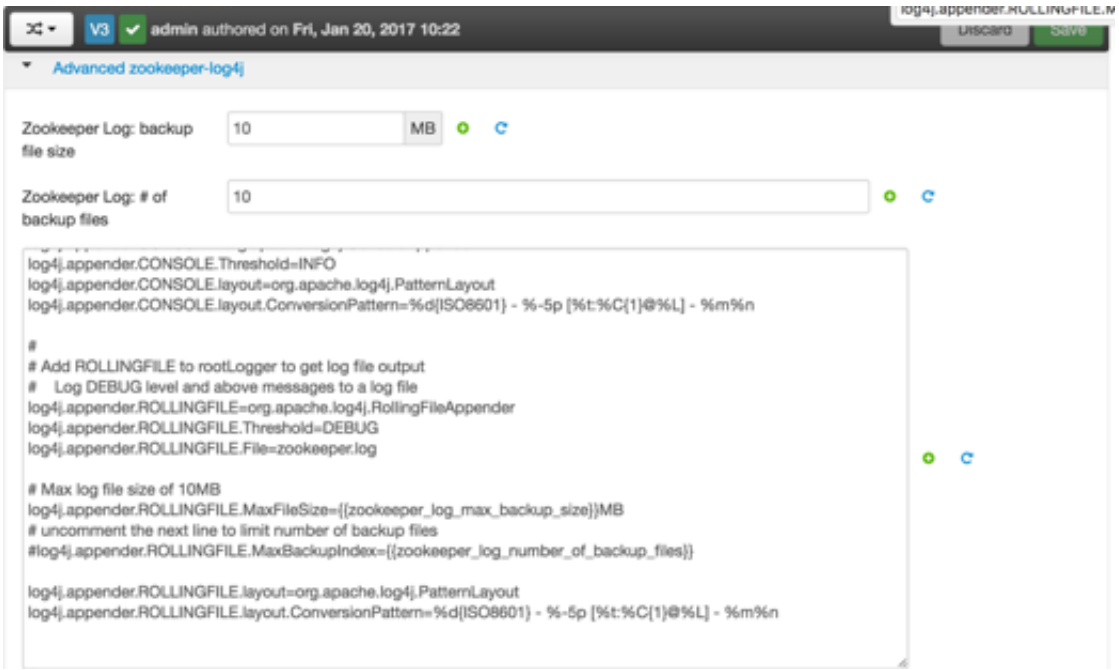
Replace:
#log4j.appender.ROLLINGFILE.MaxBackupIndex={{zookeeper_log_number_of_backup_files}}

For example:



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

For example:



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.

1.9.5. 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 2.6.4

If you already have HDP 2.6.0 installed, upgrading your cluster to HDP 2.6.4 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 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

4. 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**.

5. 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.

Important

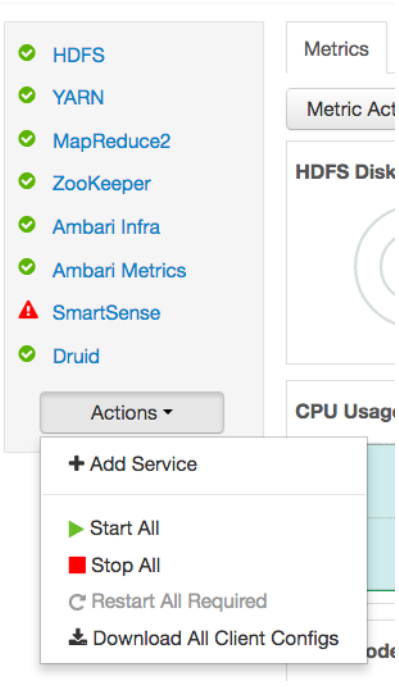
You cannot install SAM and Schema Registry for HDF 3.1 on an HDP 2.6.4 cluster, and you cannot upgrade these services from a previous HDP cluster.

Important

You cannot upgrade your HDF Storm and Kafka versions if they exist on 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 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.

6. 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 NiFi [24]
- Configure NiFi for Atlas Integration [24]
- Configure Kafka [26]
- Configure Storm [26]
- Configure Log Search [26]
- Deploy the Cluster Services [27]
- Access the UI for Deployed Services [27]

6.1. 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.

6.2. Configure NiFi for Atlas Integration

About This Task

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. 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.

Steps

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.

Result

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.

More Information

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

6.3. 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

6.4. 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

6.5. Configure Log Search

About This Task

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.

Steps

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**.

6.6. 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.

6.7. 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.

7. 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** 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.

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**.

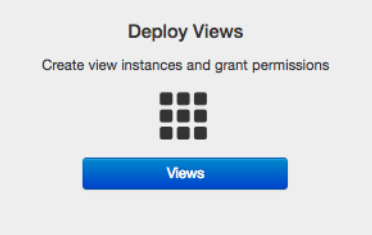
8. Install the Storm Ambari View

About This Task

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

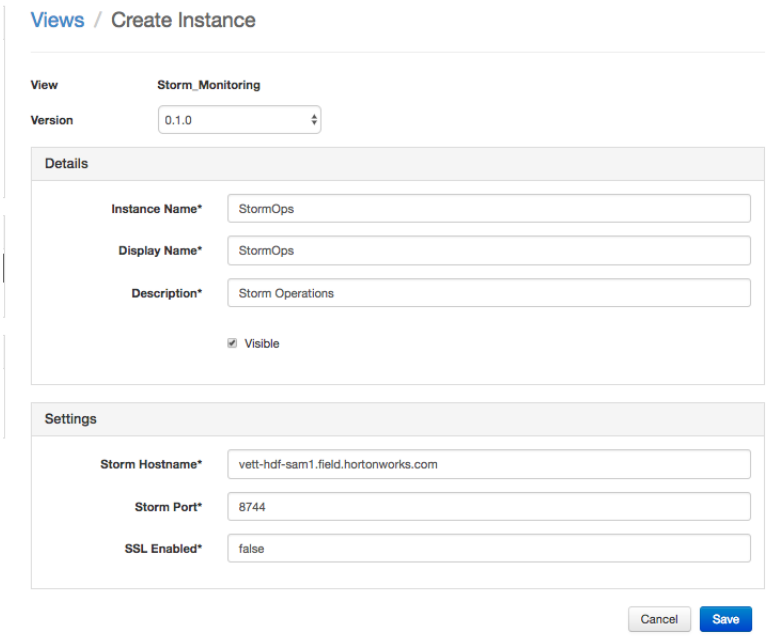
Steps

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.



a. **Instance Name** and **Display Name** may not have an spaces.

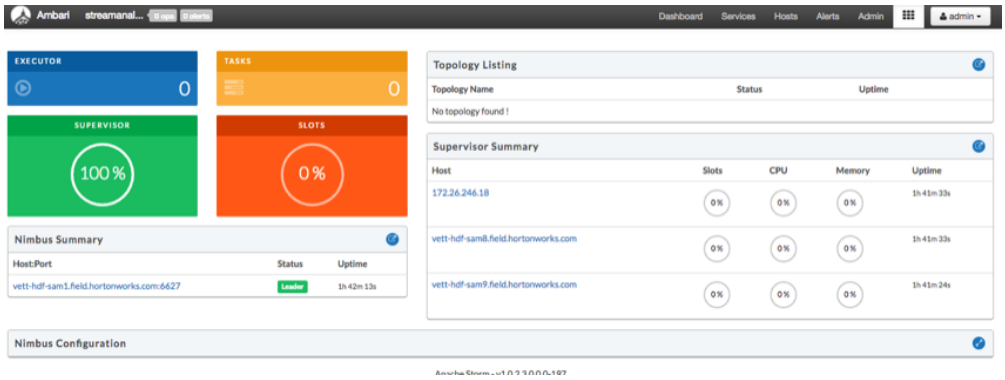
b. The **Storm Hostname** refers to the host where the Storm UI Server is deployed.

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.



9. Using a Local Repository

If your enterprise clusters have limited outbound Internet access, you should consider using a local repository, which enables you to benefit from more governance and better installation performance. You can also use a local repository for routine post-installation cluster operations such as service start and restart operations. Using a local repository includes obtaining public repositories, setting up the repository using either no internet access or limited internet access, and preparing the Apache Ambari repository configuration file to use your new 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 [34]
  - Setting up a Local Repository with Temporary Internet Access [32]
- Preparing the Ambari Repository Configuration File to Use the Local Repository [36]

9.1. 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 the tarball to create the repository.

- Temporary Internet Access

  This option involves using your temporary Internet access to synchronize (using reposync) the software packages to your selected mirror server to create 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:

  - Preparing to Set Up a Local Repository [31]
  - Setting Up a Local Repository with No Internet Access [34]
  - Setting up a Local Repository with Temporary Internet Access [32]

9.1.1. Preparing to Set Up a Local Repository

Before setting up your local repository, you must have met certain requirements.

- Selected an existing server, in or accessible to the cluster, that runs a supported operating system.
- Enabled network access from all hosts in your cluster to the mirror server.

- Ensured that the mirror server has a package manager installed such as yum (for RHEL, CentOS, or Oracle Linux), zypper (for SLES), or apt-get (for Debian and Ubuntu).
- **Optional:** If your repository has temporary Internet access, and you are using RHEL, CentOS, or Oracle Linux as your OS, installed yum utilities:

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

After meeting these requirements, you can take steps to prepare to set 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 the 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

You next must set up your local repository, either with no Internet access or with temporary Internet access.

More Information

httpd.apache.org/download.cgi

9.1.2. Setting up a Local Repository with Temporary Internet Access

Prerequisites

You must have completed the Getting Started Setting up a Local Repository procedure.

Steps

1. Install the repository configuration files for Ambari and the Stack on the host.
2. Confirm repository availability;

   For RHEL, CentOS, or Oracle Linux:
   ```
   yum repolist
   ```

   For SLES:
   ```
   zypper repos
   ```

   For Debian and Ubuntu:
   ```
   dpkg-list
   ```

3. Synchronize the repository contents to your mirror server:
   - Browse to the web server directory:

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

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

     For Debain and Ubuntu:
     ```
     cd /var/www/html
     ```

   - For Ambari, create the `ambari` directory and reposync:

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

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

     ```
     reposync -r Updates-Ambari-2.6.1.0
     ```

     In this syntax, the value of <OS> is centos6, centos7, sles11, sles12, ubuntu14, ubuntu16, or debian7.

   - For Hortonworks Data Platform (HDP) stack repositories, create the `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-2.6.1.0
   ```

   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:

Ambari Base URL	http://<web.server>/ambari/<OS>/Updates-Ambari-2.6.1.0
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
   ```

9.1.3. Setting Up a Local Repository with No Internet Access

Prerequisites

You must have completed the Getting Started Setting up a Local Repository procedure.

Steps

1. Obtain the compressed tape archive file (tarball) for the repository you want to create.

2. Copy the repository tarball to the web server directory and uncompress (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 and move the files 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:

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

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

Ambari Base URL	http://<web.server>/Ambari-2.6.1.0/<OS>
HDF Base URL	http://<web.server>/hdf/HDF/<OS>/3.x/updates/<latest.version>
HDP Base URL	http://<web.server>/hdp/HDP/<OS>/2.x/updates/<latest.version>
HDP-UTILS Base URL	http://<web.server>/hdp/HDP-UTILS-<version>/repos/<OS>

   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. **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 values:

```
[main]

enabled=1

gpgcheck=0
```

9.2. Preparing the Ambari Repository Configuration File to Use the Local Repository

Steps

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

   ```
   http://public-repo-1.hortonworks.com/ambari/<OS>/2.x/updates/2.6.1.0/ambari.repo
   ```

   In this syntax, <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.

   ```
   [Updates-Ambari-2.6.1.0]

   name=Ambari-2.6.1.0-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
   ```

   Note

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

   Base URL for a Local Repository

Built with Repository Tarball (No Internet Access)	http://<web.server>/Ambari-2.6.1.0/<OS>
Built with Repository File (Temporary Internet Access)	http://<web.server>/ambari/<OS>/Updates-Ambari-2.6.1.0

   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 host 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 Debain/Ubuntu: `/etc/apt/sources.list.d/ambari.list`

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

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

10. 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	• Release Notes • Support Matrix • Planning Your Deployment • Ambari Upgrade • MiNiFi Java Agent Quick Start
Get started with HDF	• Getting Started with Apache NiFi • Getting Started with Stream Analytics
Use and administer HDF Flow Management capabilities	• 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	• Streaming Analytics Manager User Guide • Schema Registry User Guide • Apache Storm Component Guide • Apache Kafka Component Guide