# Hortonworks DataFlow

## Ambari Managed HDF Upgrade

(January 31, 2018)

docs.hortonworks.com

# Hortonworks DataFlow: Ambari Managed HDF Upgrade

# Table of Contents

1. Upgrading Ambari and HDF 3.1.0 ..... 1
    1.1. Upgrade Ambari and the HDF Management Pack ..... 1
    1.2. Upgrading an HDF Cluster ..... 1
    1.3. Upgrading HDF Services on an HDP Cluster ..... 1
2. Preparing to Upgrade ..... 3
    2.1. General Upgrade Checklist ..... 3
    2.2. Ambari Upgrade Checklist ..... 3
    2.3. HDF Upgrade Checklist ..... 3
    2.4. Pre-Upgrade Tasks for HDF ..... 4
3. Upgrading Ambari and the HDF Management Pack ..... 5
    3.1. Preparing to Upgrade ..... 5
    3.2. Prepare Ambari for Upgrade ..... 6
    3.3. Get the Ambari Repository ..... 7
    3.4. Upgrade Ambari Server ..... 8
    3.5. Upgrade the Ambari Agents ..... 9
    3.6. Upgrade the HDF Management Pack ..... 10
    3.7. Upgrade the Ambari Databases and Restart Ambari ..... 12
    3.8. *Mandatory* Post-Upgrade Tasks ..... 13
        3.8.1. Upgrading Ambari Infra ..... 13
        3.8.2. Upgrading Ambari Log Search ..... 14
        3.8.3. Upgrading Ambari Metrics ..... 15
        3.8.4. Upgrading Configurations ..... 16
        3.8.5. Upgrading SmartSense ..... 23
4. Upgrading HDF ..... 24
    4.1. Prerequisites ..... 24
    4.2. Registering Your Target Version ..... 24
    4.3. Installing Your Target Version ..... 25
    4.4. Verifying Symbolic Links for SAM and Schema Registry ..... 25
    4.5. Performing a Rolling Upgrade ..... 26
    4.6. Performing an Express Upgrade ..... 27
    4.7. Restarting NiFi Certificate Authority ..... 28
    4.8. Review Storm Configurations ..... 28
5. Upgrading HDF Services on an HDP Cluster ..... 30
    5.1. Upgrade to Ambari 2.6.1 ..... 30
    5.2. Upgrading NiFi ..... 32
    5.3. Migrating SAM and Schema Registry data to HDF 3.1.0 ..... 33
6. Post-Upgrade Tasks ..... 35
    6.1. Post-Upgrade Tasks for NiFi ..... 35
    6.2. Post-Upgrade Tasks for SAM ..... 36

# List of Tables

4.1. Ambari-managed HDF Upgrade Prerequisites ..... 24

# 1. Upgrading Ambari and HDF 3.1.0

The steps you take to upgrade to Ambari 2.6.1 and to HDF 3.1.0 depend on your initial deployment scenario. Ensure that you review this chapter before upgrading to HDF 3.1.0, so that you understand your upgrade options and the order in which you must perform them.

## 1.1. Upgrade Ambari and the HDF Management Pack

The first step in upgrading the HDF 3.1 is to upgrade to Ambari 2.6.1.

HDF 3.1.0 is not compatible with Ambari 2.6.0 or earlier releases.

## 1.2. Upgrading an HDF Cluster

If you are upgrading an HDF cluster, you may perform one of the follow tasks:

- Rolling upgrade
- Express upgrade

### Note

Rolling Upgrade is not supported for NiFi. During the Rolling Upgrade, each NiFi instance is stopped, upgraded, and restarted.

## 1.3. Upgrading HDF Services on an HDP Cluster

If you are upgrading HDF services installed on an HDP cluster, you must understand which services you can upgrade and which you cannot.

### 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.

| Component | Version | Upgradeable on an HDP Cluster? |
|---|---|---|
| NiFi | 1.5.0 | Yes |
| NiFi Registry | 0.1.0 | Yes |
| Storm | 1.1.1 | No |
| Kafka | 1.0.0 | No |

| Component | Version | Upgradeable on an HDP Cluster? |
|---|---|---|
| SAM | 0.6.0 | No |
| Schema Registry | 0.4.0 | No |
| Logsearch | 0.5.0 | Yes |
| Ranger | 0.7.0 | Yes |
| Ambari Infra | 0.1.0 | Yes |
| Ambari Metrics | 0.1.0 | Yes |
| ZooKeeper | 3.4.6 | Yes |

# 2. Preparing to Upgrade

When preparing to upgrade, we strongly recommend you review this pre-upgrade checklist of items to confirm your cluster operation is healthy. Attempting to upgrade a cluster that is operating in an unhealthy state may produce unexpected results.

## 2.1. General Upgrade Checklist

- Ensure all services in the cluster are running.
- Run each Service Check (found under the Service Actions menu) and confirm they execute successfully.
- Clear all alerts, or understand why they are being generated. Remediate as necessary.
- Confirm start and stop for all services are executing successfully.
- Time service start and stops. The time to start and stop services is a big contributor to overall upgrade time so having this information handy is useful.
- If you are using a local repository, download the software packages prior to the upgrade. Place them in a local repository and/or consider using a storage proxy since multi-gigabyte downloads is required on all nodes in the cluster.
- Ensure point-in-time backups are taken of all DBs supporting the clusters. This includes Ambari and Ranger databases.

## 2.2. Ambari Upgrade Checklist

- This (Ambari 2.6.1) *Upgrade Guide* helps you upgrade your existing Ambari install to version 2.6.1. If you are upgrading to another Ambari version, use the *Ambari Upgrade Guide* for that version.
- Be sure to review the Known Issues and Behavioral Changes for this Ambari release in the *Release Notes*.

**More Information**

- Ambari 2.6.1 Release Notes

## 2.3. HDF Upgrade Checklist

- If you plan to add new services to your cluster, the new services may include new service accounts. You should perform any operational procedures required to support these new service accounts prior to performing your upgrade. The services accounts are typically required on all nodes in your cluster.
- If your cluster includes Storm, document any running Storm topologies.
- Complete the followin pre-upgrade task

# 2.4. Pre-Upgrade Tasks for HDF

If you have deleted any service pools from your SAM installation, your upgrade may fail as a result of SAM database dangling entries. To remove any remaining entries from your SAM database, run a manual command.

For your MySQL database, run:

```
DELETE FROM `service` WHERE `clusterId` not in (SELECT `id` FROM `cluster`);
DELETE FROM `service_configuration` WHERE `serviceId` not in (SELECT `id` FROM
 `service`);
DELETE FROM `component` WHERE `serviceId` not in (SELECT `id` FROM `service`);
```

For Oracle and Postgres, run:

```
DELETE FROM "service" WHERE "clusterId" not in (SELECT "id" FROM "cluster");
DELETE FROM "service_configuration" WHERE "serviceId" not in (SELECT "id" FROM
 "service");
DELETE FROM "component" WHERE "serviceId" not in (SELECT "id" FROM "service");
```

# 3. Upgrading Ambari and the HDF Management Pack

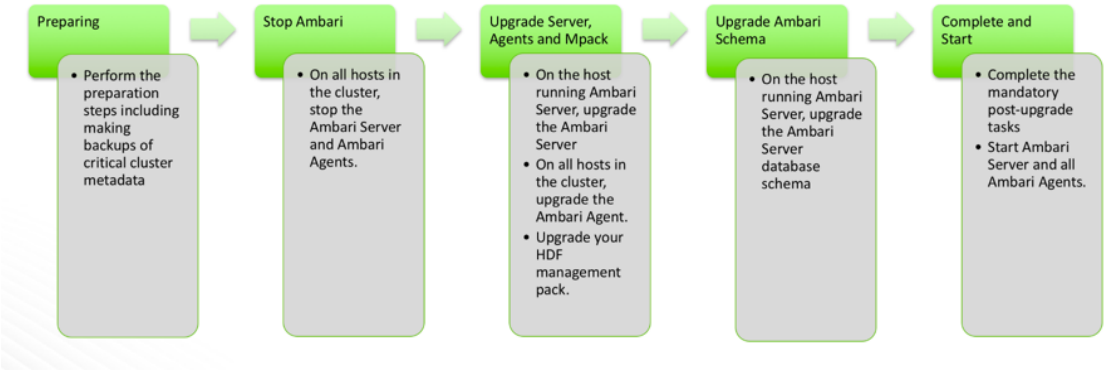
Ambari and the cluster being managed by Ambari can be upgraded independently. This section describes the process to upgrade Ambari. You are **strongly encouraged** to read completely through this entire document before starting the upgrade process, to that you understand the interdependencies and order 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 [5]
- Prepare Ambari for Upgrade [6]
- *Mandatory* Post-Upgrade Tasks [13]

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

### Important

Completing post-upgrade tasks is mandatory.




## 3.1. 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 [6]

**More Information**

Ambari 2.6.1.0 Release Notes

# 3.2. 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
```

# 3.3. 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 from the following list:

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

```
wget -nv http://public-repo-1.hortonworks.com/ambari/centos6/2.x/updates/
2.6.1.0/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.

# 3.4. 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 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.

# 3.5. 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 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` |

# 3.6. 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**

Download the HDF management pack 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 managemet 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/3.1.
0.0/tars/hdf_ambari_mp/hdf-ambari-mpack-3.1.0.0-<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/3.1.
0.0/tars/hdf_ambari_mp/hdf-ambari-mpack-3.1.0.0-<build-number>.tar.gz \
--verbose
```

   - **SLES 11:**

```
ambari-server upgrade-mpack \
--mpack=http://public-repo-1.hortonworks.com/HDF/suse11sp3/3.x/updates/3.
1.0.0/tars/hdf_ambari_mp/hdf-ambari-mpack-3.1.0.0-<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/3.1.0.
0/tars/hdf_ambari_mp/hdf-ambari-mpack-3.1.0.0-<build-number>.tar.gz \
--verbose
```

   - **Debian 7:**

```
ambari-server upgrade-mpack \
--mpack=http://public-repo-1.hortonworks.com/HDF/debian7/3.x/updates/3.1.
0.0/tars/hdf_ambari_mp/hdf-ambari-mpack-3.1.0.0-<build-number>.tar.gz \
--verbose
```

   - **Ubuntu 14:**

```
ambari-server upgrade-mpack \
--mpack=http://public-repo-1.hortonworks.com/HDF/ubuntu14/3.x/updates/3.1.
0.0/tars/hdf_ambari_mp/hdf-ambari-mpack-3.1.0.0-<build-number>.tar.gz \
--verbose
```

   - **Ubuntu 16:**

```
ambari-server upgrade-mpack \
--mpack=http://public-repo-1.hortonworks.com/HDF/ubuntu16/3.x/updates/3.1.
0.0/tars/hdf_ambari_mp/hdf-ambari-mpack-3.1.0.0-<build-number>.tar.gz \
--verbose
```

# 3.7. Upgrade the Ambari Databases and Restart Ambari

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.

5. 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".

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

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

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

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

# 3.8. *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. |

## 3.8.1. 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**.

## 3.8.2. 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**.

## 3.8.3. 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.

## 3.8.4. Upgrading Configurations

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

Upgrading Kerberos krb5.conf [16]

Upgrading Log Rotation Configuration [17]

### 3.8.4.1. 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 %}<br><br>kdc = {{kdc_host\|trim()}}<br><br>{%- endfor -%} |

## 3.8.4.2. Upgrading Log Rotation Configuration

Ambari 2.6.0 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:

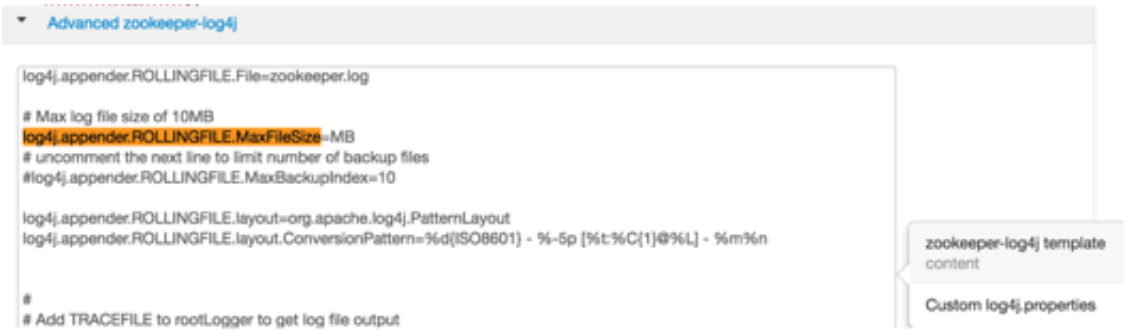
**Find:** log4j.appender.ROLLINGFILE.MaxFileSize=<value>

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

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

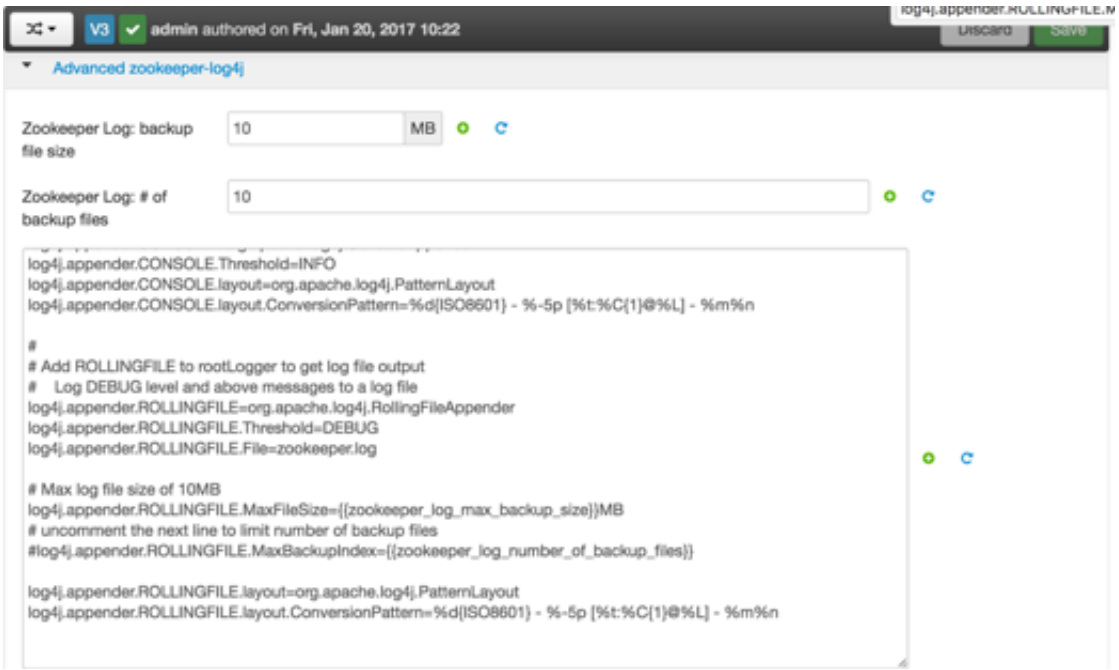
**Replace:**
#log4j.appender.ROLLINGFILE.MaxBackupIndex={{zookeeper_log_number_of_backup_files}}

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.

## 3.8.5. Upgrading SmartSense

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

**More Information**

Upgrading SmartSense

**Next Steps**

Restart services.

# 4. Upgrading HDF

There are two methods for upgrading an HDF cluster using Ambari: **Rolling Upgrade** and **Express Upgrade**.

- A **Rolling Upgrade** orchestrates the upgrade in an order that is meant to preserve cluster operation and minimize service impact during upgrade. This process has more stringent prerequisites (particularly regarding cluster high availability configuration) and can take longer to complete than an Express Upgrade.

**Note**

Rolling Upgrade is not supported for NiFi. During the Rolling Upgrade, each NiFi instance is stopped, upgraded, and restarted.

- An **Express Upgrade** orchestrates the upgrade in an order that incurs cluster downtime but has less stringent prerequisites.

## 4.1. Prerequisites

To perform an HDF upgrade using Ambari, your cluster must meet the following prerequisites. These prerequisites are required because they allow Ambari to know whether the cluster is in a healthy operating mode and can be successfully managed from Ambari.

**Table 4.1. Ambari-managed HDF Upgrade Prerequisites**

| | |
|---|---|
| Disk Space | Be sure to have adequate space on `/usr/hdf` for the target HDF installation. |
| Ambari Agent Heartbeats | All Ambari Agents must be communicating and heartbeating to Ambari Server. Any hosts that are not heartbeating must be in Maintenance Mode. |
| Host Maintenance Mode | The following two scenarios are checked:<br>• Any hosts in Maintenance Mode must not be hosting any Service Master Components.<br>• Any host in Maintenance Mode that is not hosting Master Components is allowed but you will receive a warning. You can proceed with your upgrade but these hosts will not be upgraded and **before** you can finalize the upgrade, you must delete the hosts from the cluster. |
| Service Maintenance Mode | No Services can be in Maintenance Mode. |
| Services Started | All Services must be started. |
| Service Checks | All Service Checks must pass. Be sure to run **Service Actions > Run Service Check** on all services (and remediate if necessary) prior to attempting an HDF upgrade. |

## 4.2. Registering Your Target Version

**About This Task**

Registering your target version makes Ambari aware of the Hortonworks stack to which you want to upgrade, provides the public repository location, and specifies your public or private repository delivery preference.

**Steps**

1. Click the **Admin** tab, and then click **Stack and Versions**.
2. Click the **Versions** tab.
3. Click the **Manage Versions** button.
4. Click the **+ Register Version** button.
5. Select the target version you want to register, specify whether it will be a public or private repository, and select your operating system.
6. Click **Save**.

**Result**

From the **Versions** tab, you now see your target HDF version registered, but not yet installed.

## 4.3. Installing Your Target Version

**About This Task**

Installing your target version downloads the public repositories containing software packages for your target version onto each node in your cluster.

**Steps**

1. From the **Versions** tab, identify the target version you just registered, and click the **Install on ...** button.
2. Click **OK** to confirm.
3. You can monitor the progress of the install by clicking **Installing**.

**Result**

When the installation completes, you are able to see both your current and target HDF versions from **Admin | Stack and Versions | Versions**. Your target version has an active **Upgrade** button.

## 4.4. Verifying Symbolic Links for SAM and Schema Registry

**About This Task**

After you register and install your target version, but before you proceed with a Rolling or Express Upgrade, verify that the symbolic links to the SAM and Schema Registry

configuration directories on each host are still valid. If the links are not valid, fix them before upgrading.

**Steps**

1. Confirm on each host running Registry or SAM that symlinks for the configuration directories are still valid.

```
#For Schema Registry
ls -l /etc/registry/conf

#For SAM
ls -l /etc/streamline/conf
```

2. If the link appears to be broken, perform the following steps to fix it.

   a. Remove the broken symbolic link.

```
#Remove broken Schema Registry link
rm -rf /etc/registry/conf

#Remove broken SAM link
rm -rf /etc/streamline/conf
```

   b. Create a physical SAM or Schema Registry configuration directory.

```
#Create a Schema Registry conf directory
mkdir -p /etc/registry/conf

#Create SAM conf directory
mkdir -p /etc/streamline/conf
```

   c. Copy the configuration files from your backup, into the new directory.

```
#Copy Schema Registry backup files
cp /etc/registry/conf.backup/* /etc/registry/conf

#Copy SAM backup files
cp /etc/streamline/conf.backup/* /etc/streamline/conf
```

# 4.5. Performing a Rolling Upgrade

**About This Task**

You can use a Rolling Upgrade to upgrade a cluster while preserving cluster operations and minimizing service impact. A Rolling Upgrade takes longer to complete an Express Upgrade. You can perform a rolling upgrade from any HDF 3.0.x release. However, Rolling Downgrade is not supported.

**Note**

Rolling Upgrade is not supported for NiFi. During the Rolling Upgrade, each NiFi instance is stopped, upgraded, and restarted.

**Before You Begin**

- You have disabled Auto Start. To do so, go to **Admin | Service Auto Start**, disable the Auto Start option, and click **Save**.
- You have backed up your SAM and Schema Registry databases

**Note**

You cannot edit any SAM applications or schemas managed by Schema Registry while the rolling upgrade is in progress.

**Steps**

1. From **Admin | Stack and Versions | Versions**, click **Upgrade**.
2. In the **Upgrade Options** pop-up window, click **Rolling Upgrade**, and specify if you would like customized upgrade failure tolerance.
3. Review any service Checks by clicking the Checks link from the Rolling Upgrade option.

**Note**

You may encounter a `Storm Downtime During Upgrade` warning. This message displays in error and it is safe to proceed.

4. Click **Proceed** and confirm that you want to proceed.

   The Rolling Upgrade takes some time to proceed. Monitor the progress in the **Upgrade in Progress** status bar.

5. When the Rolling Upgrade stages complete, you may choose to **Finalize** the upgrade, to **Finalize Later** or to **Downgrade**. Finalizing later gives you a chance to perform more validation on the cluster. Downgrade moves the cluster version back to the previous version (basically: reverses the upgrade process stages). **Once finalized, you cannot downgrade back to the previous version**.

**Note**

If you choose to finalize later, both versions will be listed on the Stack and Versions tab with the starting version displaying as Current. It is not until you finalize that Ambari makes the target version the current version. Also, until you finalize, you will not be able to perform operational changes to the cluster (such as move components, change configurations, etc).

# 4.6. Performing an Express Upgrade

**About This Task**

Upgrading HDF installs your target software version onto each node in your cluster.

**Before You Begin**

- You have backed up your SAM and Schema Registry databases

**Steps**

1. From **Admin | Stack and Versions | Versions**, click **Upgrade**.
2. In the **Upgrade Options** pop-up window, click **Express Upgrade**, and specify if you would like customized upgrade failure tolerance. If you select:
   - **Skip all Service Check failures** – Ambari skips any Service Check failures and completes the upgrade without requiring user intervention to continue. After all the Service Checks have run in a task, you are presented with summary of the failures and an option to continue the upgrade or pause.
   - **Skip all Slave Component failures** – Ambari skips any Slave Component failures and completes the Slave components upgrade without requiring user intervention to continue. After all Slave Components have been upgraded, you are presented with a summary of the failures and an option to continue the upgrade or pause.
3. Click **Proceed**.
4. Once the upgrade completes, again confirm that you have performed the required manual steps and click **Finalize**.

**Result**

From **Admin | Stack and Versions | Versions**, you are now able to see only the HDF version to which you upgraded.

# 4.7. Restarting NiFi Certificate Authority

After you have upgraded to your target HDF version, you will need to restart the NiFi Certificate Authority (CA).

1. From **Services | NiFi | Configs**, click **Restart**.
2. Click **Confirm Restart All**.

# 4.8. Review Storm Configurations

**About This Task**

If you have configured STORM_EXT_CLASSPATH or STORM_EXT_CLASSPATH_DAEMON prior to upgrade, you must update the values to add a wild card.

**Steps**

1. From the left-hand services navigation pane, select **Storm | Configs | Advance**
2. Update the STORM_EXT_CLASSPATH or STORM_EXT_CLASSPATH_DAEMON values to include a wild card.

**Example**

If STORM_EXT_CLASSPATH=/foo/bar/lib, update the value to STORM_EXT_CLASSPATH=/foo/bar/lib/*.

# 5. Upgrading HDF Services on an HDP Cluster

**About This Task**

You can only upgrade NiFi and install the NiFi Registry when you are upgrading HDF 3.1.0 services on an HDP cluster. **You cannot upgrade SAM or Schema Registry. You should uninstall these services before upgrading your HDF services on an HDP cluster.** To upgrade HDF services on an HDP cluster, perform the following steps:

1. Upgrade Ambari
2. Upgrade NiFi
3. (Optionally) Migrate SAM and Schema Registry data from a 3.0.x cluster to an HDF 3.1.0 cluster

**Before You Begin**

Before you begin upgrading HDF services on an HDP cluster, ensure that you have done the following:

- Upgraded your HDP cluster to HDP 2.6.4.
- Stop the NiFi service.
- Stop and uninstall SAM and Schema Registry from Ambari. This removes SAM And Schema Registry Ambari configurations. SAM and Schema Registry databases and libraries remain on installed host(s) where applicable. If you want to migrate SAM and Schema Registry to a separate HDF 3.1.0 cluster, back up these databases.

## 5.1. Upgrade to Ambari 2.6.1

**On the Ambari host where the Server and Agent are running**

1. Stop Ambari Server:

```
[root@host ~]# ambari-server stop
```

2. Stop Ambari Agents on all hosts:

```
[root@host ~]# ambari-agent stop
```

3. Update the Ambari repository. See the HDF Release Notes for the Ambari repository appropriate for your Operating System.

   For example:

```
[root@host ~]#  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
```

4. Upgrade Ambari Server binaries:

```
[root@host ~]# yum clean all
[root@host ~]# yum info ambari-server
[root@host ~]# yum upgrade ambari-server
```

5. Upgrade the Ambari Agent binaries:

```
[root@host ~]# yum upgrade ambari-agent
```

6. Upgrade your HDF management pack to 3.1.0. See the HDF Release Notes for the management pack location appropriate for your Operating System.

   For example:

```
[root@host ~]# ambari-server upgrade-mpack
--mpack=mpack=http://public-repo-1.hortonworks.com/HDF/centos7/3.x/updates/
3.1.0.0/tars/hdf_ambari_mp/hdf-ambari-mpack-3.1.0.0-<build-number>.tar.gz
--verbose
```

7. Ensure that the HDF 3.1.0 configurations are enabled for NiFi and NiFi Registry.

```
python
/var/lib/ambari-server/resources/mpacks/hdf-ambari-mpack-3.1.0.0-<build-no>/
hooks/switch_addon_services.py
/var/lib/ambari-server/resources/mpacks/hdf-ambari-mpack-3.1.0.0-<build-no>/
hooks/HDF-3.1.json
```

   **For CentOS 7**

```
cp /var/lib/ambari-server/resources/mpacks/hdf-ambari-mpack-3.1.0.0-<build-
no>/hooks/switch_addon_services.py /usr/lib/python2.6/site-packages/.
```

```
python /usr/lib/python2.6/site-packages/switch_addon_services.py /var/lib/
ambari-server/resources/mpacks/hdf-ambari-mpack-3.1.0.0-<build-no>/hooks/
HDF-3.1.json
```

8. Run the Ambari Server upgrade wizard:

```
[root@host ~]# ambari-server upgrade
```

9. Restart the Ambari Server:

```
[root@host ~]# ambari-server start
```

10. Restart the Ambari Agent:

```
[root@host ~]# ambari-agent start
```

**On the Ambari hosts where only the Ambari Agent is running**

1. Download the Ambari repository and upgrade the Ambari Agent. See the HDF Release Notes for the Ambari repository appropriate for your Operating System.

   For example:

```
[root@host ~]# 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
[root@host ~]# yum clean all
[root@host ~]# yum upgrade ambari-agent
```

2. Restart the Ambari Agent.

```
[root@host ~]# ambari-agent start
```

# 5.2. Upgrading NiFi

**Update the HDF base URL**

1. From Ambari, go to **Manage Versions** and update the HDF base URL. See the HDF Release Notes for the HDF 3.1.0 base URL appropriate for your Operating System.

   For example:

   ```
   http://public-repo-1.hortonworks.com/HDF/centos6/3.x/updates/3.1.0.0
   ```

2. Verify that `/etc/yum.repo.d/ambari-hdp-2.repo` has the updated HDF base URL. If it does not, manually edit this file to set the correct base URL.

**On each host running NiFi, perform the following NiFi upgrade steps**

1. Confirm the current NiFi components and version. For example:

   ```
   [root@host registry]# yum list installed | grep nifi
   nifi_3_0_2_0_76.x86_64
   ```

2. Display the current version associated with each component. For example:

   ```
   [root@host registry]# hdf-select status | grep nifi
   nifi - 3.0.2.0-76
   nifi-registry - None
   ```

3. Install binaries for the HDF services you want to upgrade.

   ```
   [root@host ~]# yum install -y nifi_3_1_0_0_<build-number>*
   ```

   **Note**

   Only run the `yum install` command on Ambari Agent hosts where NiFi is already installed.

4. Ensure that no NiFi processes are present on your cluster nodes:

   ```
   ps aux | grep nifi
   ```

5. If processes are still running, execute the following and ensure that no processes are available:

   ```
   "export JAVA_HOME=/usr/jdk64/{YOUR_JAVA_VERSION};/usr/hdf/current/nifi/bin/
   nifi.sh stop >> /var/log/nifi/nifi-setup.log"
   ```

   Where YOUR_JAVA_VERSION is the Java version you are running on your HDF cluster nodes.

6. Use `hdf-select` to ensure that you have created the appropriate links to the new NiFi installation:

```
[root@host ~]# hdf-select set nifi 3.1.0.0-<build-number>
```

7. Confirm that `hdf-select` displays the new HDF service versions. For example:

```
[root@host ~]# hdf-select status | grep 3.1.0.0-<build-number>
nifi - 3.1.0.0-<build-number>
```

# 5.3. Migrating SAM and Schema Registry data to HDF 3.1.0

**About This Task**

HDF 3.1 does not support installing SAM or Schema Registry services on an HDP cluster. As a result, you cannot upgrade SAM and Schema Registry from HDF 3.0.x, once they are installed on an HDP cluster. If you want to upgrade existing SAM and Schema Registry services installed on an HDP cluster, you must:

1. Backup necessary SAM and Schema Registry data
2. Create an HDF 3.0.2 cluster
3. Migrate SAM and Schema Registry to your new HDF 3.0.2 cluster
4. Upgrade the HDF 3.0.2 cluster to HDF 3.1

**Backing up SAM and Schema Registry data**

1. Backup up your SAM and Schema Registry datastores. See the following database provider documentation for backup instructions:
   - MySQL 5.5 or higher – https://dev.mysql.com/doc/refman/5.5/en/backup-and-recovery.html
   - Postgres 9.5 or higher – https://www.postgresql.org/docs/9.5/static/backup.html
   - Oracle 12c and 11g Release 2
     - https://docs.oracle.com/database/121/BRADV/title.htm
     - https://docs.oracle.com/cd/E11882_01/backup.112/e10642/title.htm
2. If SAM or Schema Registry reference any *jars* using local storage, copy and move these jars to your new server location.
3. Shut down SAM and Schema Registry on the HDF 3.0.2 cluster if it is still running. Ensure that you have also stopped any SAM applications.

**Creating a new HDF 3.0.x cluster**

Create a 3.0.x HDF cluster, restore the database, validate SAM and Schema Registry functionality, and then upgrade the new cluster to HDF 3.1. Installing the a cluster at

the same version from which you are migrating allows you to validate that application metadata was migrated successfully before upgrading.

1. Install a new HDF 3.0.x cluster. For instructions, see the installation documentation available for the appropriate version of HDF 3.0.x.

2. Before you install SAM and Schema Registry on your new HDF cluster, set up the new destination databases and restore from the backups you made. See the database provider documentation for restore instructions.

3. Install SAM and Schema Registry using an Ambari instance configured with the HDF version consistent with your source cluster. Ensure that SAM and Schema Registry are configured to use the databases you just restored.

4. For SAM, complete the following post-migration configuration updates:

   - Confirm that the environment, service pools, and applications were successfully imported.

   - To use services (Storm, Kafka, etc.) on your HDF cluster, create a new Service pool that references the new cluster

   - As needed, edit the imported applications to select the new cluster and update any configurations.

5. For Schema Registry, ensure that it is running with all schemas visible and accessible.

**Upgrade to HDF 3.1**

1. Using Ambari, upgrade this new HDF 3.0.x cluster to HDF 3.1.0. Use the instructions contained in Upgrading an HDF Cluster.

2. After you have completed the upgrade to an HDF 3.1.0 cluster, use SAM to create a new Service Pool and Environment that uses the services from HDF 3.1.0.

3. Edit any SAM applications to use this environment to ensure that you are using the versions of Storm and Kafka that you have deployed with HDF 3.1.0.

# 6. Post-Upgrade Tasks

## 6.1. Post-Upgrade Tasks for NiFi

**About This Task**

If you are upgrading to HDF 3.1.x, you must update some dataflows with a new Controller Service. For flows requiring the new StandardRestrictedSSLContextService, you must create the new context service and associate it with necessary Processors.

**Context**

StandardRestrictedSSLContextService is a new Controller Service that supports TLS version 1.2 and later. It provides the ability to configure keystore and/or truststore properties once and reuse that configuration throughout the application, but only allows you to choose a restricted set of TLS protocols. This service is recommended over StandardSSLContextService unless a component communicates with legacy systems supporting only SSL protocols.

The following Processors require StandardRestrictedSSLContextService:

- ListenBeats
- ListenHTTP
- ListenLumberjack
- ListenRELP
- ListenSMTP
- ListenSyslog
- ListenTCP
- ListenTCPRecord

**Steps**

1. For one of the impacted Processors, double-click to edit the Processor and click the **Properties** tab.
2. Edit the **SSL Context Service** property value.
3. Select **Create new service ...** from the drop-down menu.
4. Use the **Add Controller Service** dialog to create StandardRestrictedSSLContextService.

**Result**

Once you have created StandardRestrictedSSLContextService and associated it with one Processor, you can update any other Processors with the same Controller Service.

# 6.2. Post-Upgrade Tasks for SAM

**About This Task**

In HDF 3.1.x, the package name for the ConfigException class in CustomProcessorRuntime interface has changed from com.hortonworks.streamline.streams.exception to com.hortonworks.streamline.common.exception. If you have custom processors registered and are upgrading to HDF 3.1.x, there are several manual steps you must perform to reflect this change.

**Steps**

1. Update your Custom Processor implementation so that it points to the latest interface with the package name changed to com.hortonworks.streamline.common.exception.

2. In SAM's Application Resources interface, update the Custom Processor with a `.jar` file that also includes the name change.