

# Hortonworks Data Platform

## Apache Ambari Upgrade

(June 13, 2018)

## Hortonworks Data Platform: Apache Ambari Upgrade

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

The Hortonworks Data Platform, powered by Apache Hadoop, is a massively scalable and 100% open source platform for storing, processing and analyzing large volumes of data. It is designed to deal with data from many sources and formats in a very quick, easy and cost-effective manner. The Hortonworks Data Platform consists of the essential set of Apache Hadoop projects including MapReduce, Hadoop Distributed File System (HDFS), HCatalog, Pig, Hive, HBase, ZooKeeper and Ambari. Hortonworks is the major contributor of code and patches to many of these projects. These projects have been integrated and tested as part of the Hortonworks Data Platform release process and installation and configuration tools have also been included.

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

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



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

## Table of Contents

1. Upgrading Ambari and HDP .....	1
2. Getting Ready to Upgrade Ambari and HDP .....	2
3. Upgrading the Cluster's Underlying OS .....	4
4. Upgrading Ambari .....	6
4.1. Preparing to Upgrade Ambari .....	6
4.2. Upgrade Ambari .....	8
4.3. <i>Mandatory</i> Post-Upgrade Tasks .....	13
4.3.1. Upgrading Ambari Infra .....	14
4.3.2. Upgrading Ambari Log Search .....	15
4.3.3. Upgrading Ambari Metrics .....	16
4.3.4. Upgrading Configurations .....	18
4.3.5. Upgrading SmartSense .....	32
5. Upgrading HDP .....	33
5.1. Prerequisites .....	35
5.1.1. Rolling Upgrade Prerequisites .....	37
5.2. Prepare to Upgrade .....	38
5.2.1. Checkpoint HDFS .....	39
5.2.2. Preparing for Upgrade to Kafka 1.0 .....	41
5.3. Register and Install Target Version .....	44
5.3.1. Performing a Patch Upgrade .....	46
5.3.2. Limitation: Patch Upgrades in Mixed-OS Environments .....	48
5.3.3. Reverting a Patch Upgrade .....	48
5.4. Optional: Remove Duplicate Ranger Entries .....	50
5.5. Perform the Upgrade .....	53
5.5.1. Perform Rolling Upgrade .....	54
5.5.2. Perform Express Upgrade .....	56
5.6. Post-upgrade Tasks .....	58

# List of Tables

5.1. HDP Upgrade Options ..... 34

# 1. Upgrading Ambari and HDP

Ambari and the HDP Stack being managed by Ambari can be upgraded independently.

This guide provides information on:

- [Getting Ready to Upgrade Ambari and HDP \[2\]](#)
- [Upgrading Ambari \[6\]](#)
- [Upgrading HDP \[33\]](#)



## Important

Ambari 2.6 **does not support managing an HDP 2.3, 2.2, 2.1 or 2.0** cluster. If you are running HDP 2.3, 2.2, 2.1 or 2.0, in order to use Ambari 2.6 **you must first upgrade to HDP 2.4 or higher** using Ambari 2.5.2, 2.4.2, 2.2.2, 2.2.1, 2.2, 2.1, or 2.0 **before upgrading to Ambari 2.6**. Once completed, upgrade your current Ambari version to Ambari 2.6.

## More Information

[Hortonworks Support Matrix](#)

[Preparing to Upgrade Ambari \[6\]](#)

## 2. Getting Ready to Upgrade Ambari and HDP

When preparing to upgrade Ambari and the HDP Cluster, we strongly recommend you review this checklist of items to confirm your cluster operation is healthy. Attempting to upgrade a cluster that is operating in an unhealthy state can produce unexpected results.



### Important

If planning to upgrade Ambari and upgrade to a new HDP minor version (for example: moving from HDP 2.2 to HDP 2.3), upgrade Ambari to the latest version before upgrading the cluster.

- 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.
- 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 will be required on all nodes in the cluster.
- Ensure point-in-time backups are taken of all databases that support the cluster. This includes (among others) Ambari, Hive Metastore, Ranger and Oozie.



### Note

If you are upgrading to Ambari 2.5.x, you must make sure that the Ranger db root password is NOT set to blank before performing any HDP upgrade. Ambari requires that the Ranger db root password has a value. If you upgrade Ambari to 2.2 and upgrade HDP without setting a value for the Ranger db root password, Ranger Admin will fail to start after the upgrade.

To prepare Ranger for upgrades, set the password for the Ranger DB root user to a non-blank value. Then, set the Ranger db root password field in Ambari Web to match this value. Finally, restart Ranger Admin using Ambari Web.

### For Large Clusters

In a large cluster, NameNode startup processes can take a long time. NameNode startup time depends not only on host properties, but also on data volume and network parameters. To ensure that the Ambari requests to start the NameNode do not timeout during an upgrade, you should configure the Ambari NameNode restart timeout

parameter, `upgrade.parameter.nn-restart.timeout` in `/etc/ambari-server/conf/ambari.properties` on the Ambari Server host. You may need to add the restart timeout parameter and value to the Ambari server host, following a default installation. For a large cluster, you should add ten percent to the usual time (in seconds) required to restart your NameNode. Although no standard way to determine an appropriate value exists, you may use the following guidance:

For example, record the time (seconds) required to restart the active NameNode for your current Ambari server version. If restarting takes 10 minutes, (600 seconds) and your cluster is large, then add

```
upgrade.parameter.nn-restart.timeout=660
```

to the `/etc/ambari-server/conf/ambari.properties` file on the Ambari Server host.

After adding or resetting the Ambari Namenode restart parameter, restart your Ambari server before upgrading it to a newer version.

```
ambari-server restart
```

### For Ambari Upgrades

- This (Ambari 2.6.x) Upgrade Guide will help you upgrade your existing Ambari server to version 2.6.x. If you are upgrading to another Ambari version, please be sure to use the Ambari Upgrade Guide for that version.
- Be sure to review the Known Issues and Behavioral Changes for this, Ambari-2.6.x release.

### For HDP Cluster Upgrades

- Ensure sufficient disk space on `/usr/hdp/<version>` (roughly 3GB for each additional HDP release).
- If you plan to add new services available with HDP to your cluster, the new services might include new service accounts. Any operational procedures required to support these new service accounts should be performed prior to the upgrade. The accounts will typically be required on all nodes in the cluster.
- If your cluster includes Storm, document any running Storm topologies.

### More Information

[Hortonworks Support Matrix](#)

[Using a Local Repository](#)

[Ambari 2.6.2 Release Notes](#)

[Preparing to Upgrade Ambari \[6\]](#)

## 3. Upgrading the Cluster's Underlying OS

Hortonworks supports using Ambari in a multi-OS version environment. Specifically, an environment in which both RHEL/CentOS 6 and RHEL/CentOS 7 nodes are being used in the cluster. This configuration is common, as many organizations are migrating from RHEL/CentOS 6 to RHEL/CentOS 7. This chapter describes specific caveats and requirements related to using Ambari in a mixed OS version environment.



### Note

No Ambari package upgrades are necessary, only ensuring the Ambari Server database is handled properly, and that individual cluster host OS packages are validated post-upgrade.

Prior to upgrading your cluster nodes that are running RHEL/CentOS 6 to RHEL/CentOS 7, please perform the following steps on the server that is running Ambari Server instance *if, and only if* you are using the embedded Postgres instance for the Ambari Server's backing database.

#### Before Upgrading Ambari Server Host OS

1. Create a backup of your `pg_hba.conf` file.

```
cd /var/lib/pgsql/data/
cp pg_hba.conf pg_hba.conf.backup
```

2. On the Ambari Server host, stop the original Ambari Server.

```
ambari-server stop
```

3. Create a directory to hold the database backups.

```
cd /var/tmp
mkdir dbdumps
cd dbdumps
```

4. Create the database backups.

```
pg_dump -U {ambari.db.username} -f ambari.sql
Password: {ambari.db.password}
```

where:

Variable	Description	Default
<code>ambari.db.username</code>	The database username	<code>ambari</code>
<code>ambari.db.password</code>	The database password	<code>bigdata</code>

5. Create a backup of the Ambari Server meta info.

```
ambari-server backup
```

#### After Upgrading Ambari Server Host OS



During the OS upgrade from RHEL/CentOS 6 to RHEL/CentOS 7 OS packages are upgraded, including the Postgres instance that the Ambari Server relies upon. Use the following steps to ensure the Ambari Server host OS and database function properly after the OS has been upgraded.

1. Clean up HDP and Ambari related repositories created by upgrade tool.

```
(redhat-upgrade-*)
```

2. Ensure that all OS packages were upgraded appropriately. If not, it may be necessary to remove packages containing 'el6' in their names and install the appropriate el7 replacement packages.

```
yum check dependencies
```



### Note

Ensure that check dependencies returns 0 before continuing.

3. a. Postgres is upgraded during the OS upgrade. You must run the following command to complete the database upgrade:

```
postgresql-setup upgrade
```

4. It is also necessary to restore the original `pg_hba.conf` that contains configuration for the Ambari users:

```
cd /var/lib/pgsql/data/
```

```
cp pg_hba.conf pg_hba.conf.after-upgrade
```

```
cp pg_hba.conf.backup pg_hba.conf
```

5. Restart the postgres instance.

```
service postgresql restart
```

6. Start the Ambari Server.

```
ambari-server start
```

### After Upgrading Ambari Agent Host OS

For the rest of the cluster, it is only necessary to ensure the repositories and host OS packages have been fully upgraded, no additional steps are required beyond the following:

1. Cleanup HDP and Ambari related repositories created by upgrade tool.

```
(redhat-upgrade-*)
```

2. Ensure that all OS packages were upgraded appropriately. If not, it may be necessary to remove packages containing 'el6' in their names and install the appropriate el7 replacement packages.

```
yum check dependencies
```



### Note

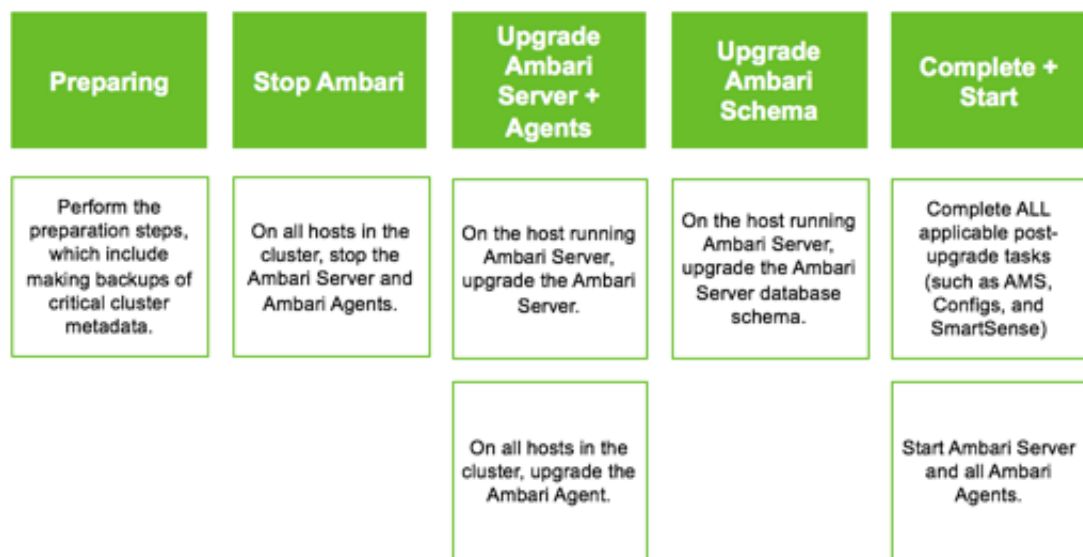
Ensure that check dependencies returns 0 before continuing.

## 4. Upgrading Ambari

Ambari and the HDP cluster being managed by Ambari can be upgraded independently. This section describes the process to upgrade Ambari. You are **strongly encouraged** to read completely through this entire document before starting the upgrade process, so 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 Ambari \[6\]](#)
- [Upgrade Ambari \[8\]](#)
- [Mandatory Post-Upgrade Tasks \[13\]](#)

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



### Important

Completing post-upgrade tasks is mandatory.

### More Information

[Hortonworks Support Matrix](#)

### 4.1. Preparing to Upgrade Ambari

- Be sure to review the Ambari 2.6.2.2 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.
- If your cluster is SSO-enabled, do not stop Knox before upgrading Ambari.
- If you are upgrading to Ambari-2.6.x from a version older than Ambari-2.5.2, you **must** add `kerberos.operation.verify.kdc.trust` in `ambari.properties`.

If you are using SSL, set the value of `kerberos.operation.verify.kdc.trust` to `true`.

If you are NOT using SSL, set the value of `kerberos.operation.verify.kdc.trust` to `false`.

Failing to add this property before upgrading, causes the following error when attempting to install any component, post-upgrade:

```
Failed to connect to KDC - Failed to communicate with the Active Directory
at ldaps:
//my.ldap.com:636: simple bind failed: my.ldap.com:636
Make sure the server's SSL certificate or CA certificates have been imported
into Ambari's truststore.
Please enter admin principal and password.
```



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

From / To	2.5.x	2.6.x
2.0.x	✓	X
2.1.x	✓	X
2.2.x	✓	X
2.4.x	✓	✓
2.5.x	NA	✓

During Ambari upgrade, the existing `/var/lib/ambari-server/ambari-env.sh` file is overwritten and a backup copy of `ambari-env.sh` (with extension `.rpmsave`) is

created. If you have manually modified `ambari-env.sh` (for example, to change Ambari Server heap), you will need to re-apply your changes to the new file.

### Next Steps

[Upgrade Ambari \[8\]](#)

### More Information

[Upgrade Ambari Metrics](#)

[Add Grafana](#)

[Upgrade SmartSense](#)

[Ambari 2.6.2.2 Release Notes](#)

[Upgrade the Ambari Server version to 2.2](#)

## 4.2. Upgrade Ambari

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

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

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



### Important

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

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

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

```
wget -nv https://username:password@archive.cloudera.com/p/ambari/2.x/2.6.2.2/centos6/ambari.repo -O /etc/yum.repos.d/ambari.repo
```

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

```
wget -nv https://username:password@archive.cloudera.com/p/ambari/2.x/2.6.2.2/centos7/ambari.repo -O /etc/yum.repos.d/ambari.repo
```

- **For SLES 11:**

```
wget -nv https://username:password@archive.cloudera.com/p/ambari/2.x/2.6.2.2/suse11/ambari.repo -O /etc/zypp/repos.d/ambari.repo
```

- **For SLES 12:**

```
wget -nv https://username:password@archive.cloudera.com/p/ambari/2.x/2.6.2.2/sles12/ambari.repo -O /etc/zypp/repos.d/ambari.repo
```

- **For Ubuntu 14:**

```
wget -nv https://username:password@archive.cloudera.com/p/ambari/2.x/2.6.2.2/ubuntu14/ambari.list -O /etc/apt/sources.list.d/ambari.list
```

- **For Ubuntu 16:**

```
wget -nv https://username:password@archive.cloudera.com/p/ambari/2.x/2.6.2.2/ubuntu16/ambari.list -O /etc/apt/sources.list.d/ambari.list
```



### Note

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



### Note

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

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

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

```
yum clean all
```

```
yum info ambari-server
```

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

```
yum upgrade ambari-server
```

- **For SLES:**

```
zypper clean
```

```
zypper info ambari-server
```

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

```
zypper up ambari-server
```

- **For Ubuntu/Debian:**

```
apt-get clean all
```

```
apt-get update
```

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

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

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



## Important

When performing upgrade on SLES, you will see a message "There is an update candidate for 'ambari-server', but it is from different vendor. Use 'zypper install ambari-server-2.6.2-155.noarch' to install this candidate". You will need to use yast to update the package, as follows:

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

```
> yast
```

You will see command line UI for YaST program.

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

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

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

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

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

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

- A successful upgrade displays output similar to the following:

```
Updated: ambari-server.noarch 0:2.6.2.2 Complete!
```



## Note

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

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

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

```
yum upgrade ambari-agent
```

- **For SLES:**

```
zypper up ambari-agent
```



### Note

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



### Important

When performing upgrade on SLES, you will see a message "There is an update candidate for 'ambari-agent', but it is from different vendor. Use 'zypper install ambari-agent-2.6-143.noarch' to install this candidate". You will need to use yast to update the package, as follows:

- From the command line run: > yast

```
> yast
```

You will see command line UI for YaST program.

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

- **For Ubuntu/Debian:**

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

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

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

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

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

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

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

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

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

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

```
ambari-server upgrade
```

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

```
ambari-server start
```

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

```
ambari-agent start
```

12 Open Ambari Web.

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

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



### Important

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

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

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

You will see a Restart indicator next to each service after upgrading. Ambari upgrade has added to/adjusted the configuration properties of your cluster based on new configuration types and properties being made available for each service with this release of Ambari. Review these changes by comparing the previous configuration with the latest version created by "ambari-upgrade".

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

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

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

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

to get the JDBC driver .jar file in place.



- 16.If you are running **Ambari Metrics** service in your cluster, you **must** upgrade Ambari Metrics System and add the Grafana component.
- 17.If your cluster includes the SmartSense service, you **must** upgrade SmartSense along with Ambari.
- 18.Perform any other post-upgrade tasks, as necessary.



### Important

Completing post-upgrade tasks is mandatory.

#### Next Steps

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

#### More Information

[Using a Local Repository](#)

[Configuring Network Port Numbers](#)

[Set Up LDAP or Active Directory Authentication](#)

[Using New and Existing Databases - Hive](#)

[Using Existing Databases - Oozie](#)

[Upgrade Ambari Metrics](#)

[Add Grafana](#)

[Upgrade SmartSense](#)

## 4.3. *Mandatory* Post-Upgrade Tasks

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

#### [Upgrading Ambari Infra](#)

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

#### [Upgrading Ambari Log Search](#)

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

#### [Upgrading Ambari Metrics](#)

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

#### [Adding Grafana to Ambari Metrics](#)

Grafana is now included as a component of Ambari Metrics. If you are upgrading from Ambari 2.2.1 or

earlier, and your Ambari Metrics service does not contain Grafana, proceed to add Grafana to Ambari Metrics.

#### Upgrading Configurations

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

#### Upgrading SmartSense

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

#### Next Steps

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

#### More Information

#### [Upgrading Configurations](#)

## 4.3.1. Upgrading Ambari Infra

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

#### Steps

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

#### For RHEL/CentOS/Oracle Linux:

```
yum clean all
```

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

#### For SLES:

```
zypper clean
```

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

#### For Ubuntu/Debian:

```
apt-get clean all
```

```
apt-get update
```

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

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

#### For RHEL/CentOS/Oracle Linux:

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

**For SLES:**

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

**For Ubuntu/Debian:**

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

4. Start the Ambari Infra services.

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

**More Information**

[Ambari Infra](#)

## 4.3.2. Upgrading Ambari Log Search

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

**Prerequisites**

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

**Steps**

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

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

```
yum clean all
```

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

**For SLES:**

```
zypper clean
```

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

**For Ubuntu/Debian:**

```
apt-get clean all
```

```
apt-get update
```

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

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

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

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

**For SLES:**

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

**For Ubuntu/Debian:**

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

**4. Start Log Search Service.**

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

**More Information**

[Upgrading Ambari Infra](#)

### 4.3.3. Upgrading Ambari Metrics

**Prerequisites**

Upgrade to Ambari 2.6 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.

**More Information**

[Add Grafana to Ambari Metrics](#)

### 4.3.3.1. Adding Grafana to Ambari Metrics

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

**Note**

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

1. Upgrade to Ambari 2.6 and perform needed post-upgrade checks. Make sure all services are up and healthy.

2. Add the METRICS\_GRAFANA component to Ambari:

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

3. Add METRICS\_GRAFANA to a host in the cluster.

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

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

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

5. To complete the configuration, click on **Services > Ambari Metrics > Configs** and enter the default Grafana Admin Password in the **General** section. Click **Save**.

6. Browse to **Hosts > host.name** (the **host.name** used in the API call where you added Grafana). You will see the Grafana component is in an **Install Pending...** state. Use the **Install Pending...** action button and select **Re-install**.

 Grafana / Ambari Metrics

Install Pending... ▼

7. Once the install operation completes, select **Start** to start Grafana.

8. To access Grafana, browse to **Services > Ambari Metrics**, select **Quick Links** and then click **Grafana**.

### More Information

[Upgrade to Ambari 2.6](#)

[Using Grafana](#)

## 4.3.4. Upgrading Configurations

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

### More Information

[Upgrading Kerberos krb5.conf \[18\]](#)

[Upgrading Log Rotation Configuration \[19\]](#)

### 4.3.4.1. Upgrading Kerberos krb5.conf

Ambari has added support for handling more than one KDC host . Only one kadmin host is supported by the Kerberos infrastructure. This required modifications for the **krb5.conf**

template. In order for Ambari to properly construct the `krb5.conf` configuration file, make the following configuration change if your cluster meets all of these criteria:

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

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

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

### More Information

[Configure Ambari for Automated Setup](#)

## 4.3.4.2. Upgrading Log Rotation Configuration

Starting with Ambari 2.6.0, log rotation configuration capability is enabled. Ambari 2.6.0 provides a simplified log rotation configuration. These changes will be made automatically during your next stack upgrade, but are not automatically made during the Ambari upgrade. After upgrading Ambari from version 2.x to 2.6.0, 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

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

```
zookeeper_log_max_backup_size=10
```

```
zookeeper_log_number_of_backup_files=10
```

For example:

ambari-upgrade authored on Tue, Jan 17, 2017 15:21

Discard Save

- Advanced zookeeper-log4j
- Advanced zookeeper-logsearch-conf
- Custom zoo.cfg
- Custom zookeeper-log4j
  - zookeeper\_log\_max\_backup\_size: 10
  - zookeeper\_log\_number\_of\_backup\_files: 10
  - Add Property ...

- Click **Add**.
- Browse to **Advanced zookeeper-log4j**.
- 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\_max\_backup\_size }}MB

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

**Replace:**

log4j.appender.ROLLINGFILE.MaxBackupIndex={{ zookeeper\_log\_number\_of\_backup\_files }}

For example:

```

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

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

#
# Add TRACEFILE to rootLogger to get log file output
  
```

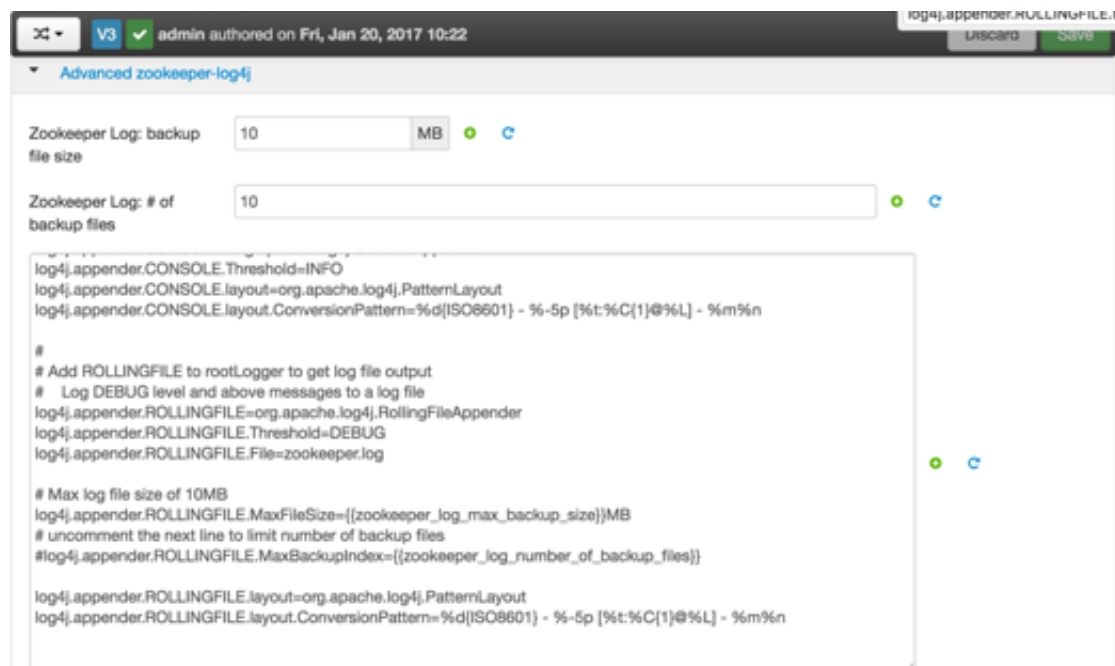
zookeeper-log4j template content

Custom log4j.properties

- 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=<appender type>

**Replace:** log4j.appender.kafkaAppender=org.apache.log4j.RollingFileAppender

**Add:** log4j.appender.kafkaAppender.MaxFileSize = {{kafka\_log\_maxfilesize}}MB

**Add:** log4j.appender.kafkaAppender.MaxBackupIndex =  
{ {kafka\_log\_maxbackupindex} }

**Find:** log4j.appender.controllerAppender=org.apache.log4j.RollingFileAppender

**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. Knox

- a. In **Ambari Web**, browse to **Knox > Configs**.
- b. Scroll down to **Custom gateway-log4j**.
- c. In **Custom gateway-log4j**, click **Add Property**.
- d. In **Add Property**, type the following properties and values:  
  
knox\_gateway\_log\_maxfilesize=256  
  
knox\_gateway\_log\_maxbackupindex=20
- e. Click **Add**.
- f. Browse to **Advanced gateway-log4j**.
- g. In **Advanced gateway-log4j content section**, find and replace the following properties and values:

**Find:** log4j.appender.drfa=org.apache.log4j.DailyRollingFileAppender

**Replace:** log4j.appender.drfa=org.apache.log4j.RollingFileAppender

**Add:** log4j.appender.drfa.MaxFileSize = { {knox\_gateway\_log\_maxfilesize} }MB

**Add:** log4j.appender.drfa.MaxBackupIndex =  
{ {knox\_gateway\_log\_maxbackupindex} }

- h. In **Configs**, click **Save**.
- i. Restart **Knox**.

### 4. Atlas

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

- b. Scroll down to **Custom Atlas-log4j**.
- c. In **Custom Atlas-log4j**, click **Add Property**.
- d. In **Add Property**, type the following properties and values:  
atlas\_log\_max\_backup\_size=256  
atlas\_log\_number\_of\_backup\_files=20
- e. Click **Add**.
- f. Browse to **Advanced atlas-log4j**.
- g. In **Advanced atlas-log4j content section**, find and replace the following properties and values:  
**Find:** <appender name="FILE" class="org.apache.log4j.DailyRollingFileAppender">  
**Replace:** <appender name="FILE" class="org.apache.log4j.RollingFileAppender">  
**Add:** <param name="MaxFileSize" value="{ atlas\_log\_max\_backup\_size } MB" >  
**Add:** <param name="MaxBackupIndex" value="{ atlas\_log\_number\_of\_backup\_files }" >
- h. In **Configs**, click **Save**.
- i. Restart **Atlas**, as prompted.

## 5. 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=<appender type>  
**Replace:** log4j.appender.xa\_log\_appender=org.apache.log4j.RollingFileAppender

**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=<appender type>

**Replace:** log4j.appender.logFile=org.apache.log4j.RollingFileAppender

**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=<appender type>

**Replace:** log4j.appender.logFile=org.apache.log4j.RollingFileAppender

**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.

## 6. 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=<appender type>

**Replace:** log4j.appender.kms=org.apache.log4j.RollingFileAppender

**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.RollingFileAppender

**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**.

## 7. HBase

- a. In **Ambari Web**, browse to **HBase > Configs > Advanced**.
- b. Scroll down to **Custom hbase-log4j**.
- c. In **Custom hbase-log4j**, click **Add Property**.
- d. In **Add Property**, type the following properties and values:  
  
hbase\_log\_maxfilesize=256  
  
hbase\_log\_maxbackupindex=20  
  
hbase\_security\_log\_maxfilesize=256  
  
hbase\_security\_log\_maxbackupindex=20
- e. Click **Add**.
- f. Browse to **Advanced hbase-log4j**.
- g. In **Advanced hbase-log4j content section**, find and replace the following properties and values:  
  
**Find:** hbase.log.maxfilesize=<value>MB  
  
**Replace:** hbase.log.maxfilesize={{hbase\_log\_maxfilesize}}MB  
  
**Find:** hbase.log.maxbackupindex=<value>  
  
**Replace:** hbase.log.maxbackupindex={{hbase\_log\_maxbackupindex}}
- Find:** hbase.security.log.maxfilesize=<value>MB  
  
**Replace:** hbase.security.log.maxfilesize={{hbase\_security\_log\_maxfilesize}}MB  
  
**Find:** hbase.security.log.maxbackupindex=<value>  
  
**Replace:**  
hbase.security.log.maxbackupindex={{hbase\_security\_log\_maxbackupindex}}
- h. In **Configs**, click **Save**.
- i. Restart **HBase**.

## 8. Hive

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

hive\_log\_maxfilesize=256

hive\_log\_maxbackupindex=30

e. Click **Add**.

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

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

**Find:**log4j.appender.DRFA=org.apache.log4j.DailyRollingFileAppender

**Replace:** log4j.appender.DRFA=org.apache.log4j.RollingFileAppender

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

**Replace:** log4j.appender.DRFA.MaxBackupIndex={{hive\_log\_maxbackupindex}}

h. Scroll down to **Custom llap-daemon-log4j**.

i. In **Custom llap-daemon-log4j**, click **Add Property**.

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

hive\_llap\_log\_maxfilesize=256

hive\_log\_maxbackupindex=240

k. Click **Add**.

l. Browse to **llap-daemon-log4j**.

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

**Find:** property.llap.daemon.log.maxfilesize =<value>MB

**Replace:** property.llap.daemon.log.maxfilesize = {{hive\_llap\_log\_maxfilesize}}MB

**Find:** property.llap.daemon.log.maxbackupindex=<value>

**Replace:**

property.llap.daemon.log.maxbackupindex={{hive\_llap\_log\_maxbackupindex}}

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

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

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

webhcat\_log\_maxfilesize=256

webhcat\_log\_maxbackupindex=240

- q. Click **Add**.
- r. Browse to **webhcat-log4j** .
- s. In **webhcat-log4j** *content section*, find and replace the following properties and values:  
**Find:** log4j.appender.standard = org.apache.log4j.RollingFileAppender  
**Add:** log4j.appender.standard.MaxFileSize = {{webhcat\_log\_maxfilesize}}MB  
**Add:** log4j.appender.standard.MaxBackupIndex = {{webhcat\_log\_maxbackupindex}}
- t. In **Configs**, click **Save**.
- u. Restart **Hive**, as prompted.

## 9. 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.

## 10 Falcon

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

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

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

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

```
falcon_log_maxfilesize=256
falcon_log_maxbackupindex=20
falcon_security_log_maxfilesize=256
falcon_security_log_maxbackupindex=20
```

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

**Find:** <appender name="FILE" class="org.apache.log4j.DailyRollingFileAppender">

**Replace:** <appender name="FILE" class="org.apache.log4j.RollingFileAppender">

**Add:** <param name="MaxFileSize" value="{{ falcon\_log\_maxfilesize }}" MB" />

**Add:** <param name="MaxBackupIndex" value="{{ falcon\_log\_maxbackupindex }}" />

**Find:** <appendername="SECURITY" class="org.apache.log4j.RollingFileAppender">

**Add:** <param name="MaxFileSize" value="{{ falcon\_security\_log\_maxfilesize }}" MB"/>

**Add:** <param name="MaxBackupIndex" value="{{ falcon\_security\_log\_maxbackupindex }}" />

- h. In **Configs**, click **Save**.
- i. Restart **Falcon**.

## 11.Oozie

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

```
oozie_log_maxhistory=720
```
- e. Click **Add**.
- f. Browse to **Advanced oozie-log4j** .
- g. In **Advanced oozie-log4j** *content section*, find and replace the following properties and values:

**Find:** log4j.appender.oozie.RollingPolicy.MaxHistory=720

**Replace:** `log4j.appender.oozie.RollingPolicy.MaxHistory={{oozie_log_maxhistory}}`

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

## 12.HDFS

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

`hadoop_security_log_max_backup_size = 256`

`hadoop_security_log_number_of_backup_files = 20`

`hadoop_log_max_backup_size = 256`

`hadoop_log_number_of_backup_files = 10`

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

**Find:** `hadoop.security.log.maxfilesize=<value>MB`

**Replace:**

`hadoop.security.log.maxfilesize={{hadoop_security_log_max_backup_size}}MB`

**Find:** `hadoop.security.log.maxbackupindex=20`

**Replace:**

`hadoop.security.log.maxbackupindex={{hadoop_security_log_number_of_backup_files}}`

**Find:** `log4j.appender.RFA.MaxFileSize=<value>MB`

**Replace:** `log4j.appender.RFA.MaxFileSize={{hadoop_log_max_backup_size}}MB`

**Find:** `log4j.appender.RFA.MaxBackupIndex=<value>`

**Replace:**

`log4j.appender.RFA.MaxBackupIndex={{hadoop_log_number_of_backup_files}}`

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

### 13.YARN

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

```
yarn_rm_summary_log_max_backup_size=256
```

```
yarn_rm_summary_log_number_of_backup_files=20
```
- e. Click **Add**.
- f. Browse to **Advanced yarn-log4j**.
- g. In **Advanced yarn-log4j** *content* section, find and replace the following properties and values:  
**Find:** log4j.appender.RMSUMMARY.MaxFileSize=<value>MB  
**Replace:**  
log4j.appender.RMSUMMARY.MaxFileSize={{yarn\_rm\_summary\_log\_max\_backup\_size}}MB  
**Find:** log4j.appender.RMSUMMARY.MaxBackupIndex=<value>  
**Replace:**  
log4j.appender.RMSUMMARY.MaxBackupIndex={{yarn\_rm\_summary\_log\_number\_of\_backup\_files}}
- h. In **Configs**, click **Save**.
- i. Restart **YARN**, as prompted.

## 4.3.5. Upgrading SmartSense

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

### More Information

[Upgrading SmartSense](#)

### Next Steps

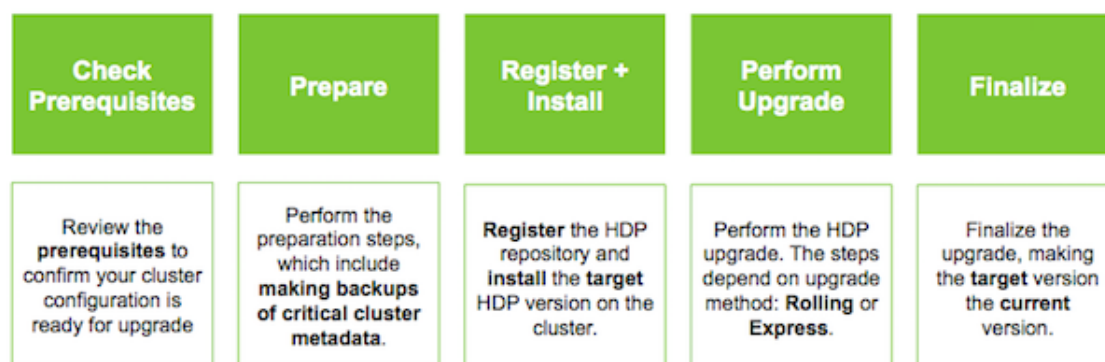
Restart services.

## 5. Upgrading HDP

- [Prerequisites \[35\]](#)
- [Prepare to Upgrade \[38\]](#)
- [Register and Install Target Version \[44\]](#)
- [Perform the Upgrade \[53\]](#)

You may choose from several available upgrade options, based on your current HDP version and the HDP version to which you will upgrade (the target version). This topic describes available upgrade options, their prerequisites, and the overall process. 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 and account for any special configurations for your cluster.

The high-level process for performing an HDP upgrade is as follows:



Ambari will guide you through the steps required to upgrade HDP. Make sure Ambari and the cluster are healthy, operating normally, and all service checks are passing.



### Note

Be sure to review the available HDP upgrade scenarios below. It is **strongly recommended** that you **first upgrade to Ambari 2.6.2** before upgrading HDP unless otherwise noted. Ambari 2.6.2 supports patch and maintenance upgrades to HDP 2.6.5.x stack. After upgrading Ambari, be sure the cluster is operating normally and service checks are passing prior to attempting an HDP upgrade.

Ambari 2.6.2 does not support IOP-HDP migration. Customers migrating from IOP should use Ambari 2.6.1 to migrate IOP to HDP 2.6.4, then upgrade Ambari to 2.6.2 and use it to upgrade HDP 2.6.4 to HDP 2.6.5.

There are two methods for upgrading HDP with Ambari: **Rolling Upgrade** and **Express Upgrade**.

- A **Rolling Upgrade** orchestrates the HDP 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.

- An **Express Upgrade** orchestrates the HDP upgrade in an order that will incur cluster downtime but with less stringent prerequisites.

The HDP Stack version is based on the following format: **major.minor.maintenance.patch-build#**. A major release increments the first digit; a minor release, the second; a maintenance release the third; and a patch release the fourth. This terminology is used in the following table to describe the different upgrade paths. For example: an upgrade from HDP 2.5 to HDP 2.6, is a Minor Upgrade. An upgrade from HDP 2.5.x to HDP 2.5.y, is a Maintenance Upgrade, and an upgrade from 2.6.0.0 to 2.6.0.1 is a patch upgrade. A Maintenance Upgrade may include multiple patch releases.

The following table describes available upgrade options.

**Table 5.1. HDP Upgrade Options**

Upgrade Path	Current HDP*	Target HDP*	Rolling Upgrade	Express Upgrade
Minor **	HDP 2.5	HDP 2.6		
Minor **	HDP 2.4	HDP 2.5 or 2.6		
Patch/Maintenance	HDP 2.6.x.y	HDP 2.6.x.y		
Maintenance	HDP 2.5.x.y	HDP 2.5.x.y		
Maintenance	HDP 2.4.x.y	HDP 2.4.x.y		

\* The instructions in this document sometimes refer to an HDP “version” using the HDP #.#.x.y convention. For example, you will see the HDP 2.6.x.y convention to refer to any HDP 2.6 “maintenance” release. Therefore, to use a specific HDP 2.6 maintenance release, be sure to replace 2.6.x.y in the following instructions with the appropriate maintenance version, such as 2.6.0.0 for the HDP 2.6 GA release, or 2.6.2.0 for an HDP 2.6 maintenance release.

\*\* A Minor upgrade can be performed from any current maintenance release to any target maintenance release in that minor release. For example, you can upgrade from HDP 2.5.x.y to HDP 2.6.x.y as part of a Minor upgrade.



### Important

Because HDP 2.3.6 contains features and fixes that are not applicable to HDP 2.4.0 or HDP 2.4.2, do not upgrade from HDP 2.3.6 to HDP 2.4.0 or HDP 2.4.2.

### More Information

[Preparing to Upgrade Ambari and HDP](#)

[Rolling Upgrade Prerequisites \[37\]](#)

## 5.1. Prerequisites

To perform an HDP upgrade using Ambari, your cluster must meet the following prerequisites, whether you intend to perform a Rolling or Express upgrade. Meeting these prerequisites is essential for Ambari to know the cluster is in a healthy operating mode and can successfully manage the upgrade process. Your cluster must meet additional prerequisites for a Rolling Upgrade, which ensure that the cluster is configured properly to support the rolling upgrade orchestration process.

<b>HDP Version</b>	All hosts must have the target HDP version installed.
<b>Disk Space</b>	Be sure to have adequate space on <code>/usr/hdp</code> for the target HDP version. Each complete install of an HDP version will occupy about 2.5 GB of disk space.
<b>Ambari Agent Heartbeats</b>	All Ambari Agents must be communicating and heartbeating to Ambari Server. Any hosts that are not heartbeating must be in Maintenance Mode.
<b>Host Maintenance Mode</b>	The following two scenarios are checked: <ul style="list-style-type: none"><li>• Any hosts in Maintenance Mode must not be hosting any Service Master Components.</li><li>• 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 <b>before</b> you can finalize the upgrade, you must delete the hosts from the cluster.</li></ul>
<b>Service Maintenance Mode</b>	No Services can be in Maintenance Mode.
<b>Services Started</b>	All Services must be started.
<b>Service Checks</b>	All Service Checks must pass. Be sure to run <b>Service Actions &gt; Run Service Check</b> on all services (and remediate if necessary) prior to attempting an HDP upgrade.
<b>Atlas Check</b>	<p>If you are upgrading to HDP 2.5 and have Atlas installed in your HDP 2.3 or 2.4 cluster, Atlas must be removed prior to upgrading to HDP 2.5 <i>after you have upgraded to Ambari-2.4.x</i>.</p> <p>Do NOT remove Atlas service via REST API.</p> <p>Use the <b>Remove Service</b> capability to completely remove Atlas from your cluster and then perform your cluster upgrade to HDP 2.5 and re-add Atlas service to your cluster.</p>



## Note

If you are using Atlas on HDP-2.3 or 2.4; **Do Not** upgrade to Ambari-2.4.x until you are ready to upgrade to HDP-2.5 with the new version of Atlas.

### Kerberos - Kafka

If your cluster has Kerberos enabled, includes Kafka and you are running HDP 2.2, since Kafka in HDP 2.2 did not support running Kerberized, after upgrade to HDP 2.3 or later (regardless if you perform an Express or Rolling Upgrade), Kafka will then be Kerberos enabled. Plan to adjust your use of Kafka accordingly. Ambari will perform a check to see if your cluster would encounter this issue and warn you prior to performing the HDP upgrade.

### Ranger Policy Export/Import DB Requirements

HDP-2.6 and higher versions include Ranger policy import/export. The minimum database requirements for this feature are as follows:

- MariaDB: 10.1.16+
- MySQL: 5.6.x+
- Oracle: 11gR2+
- PostgreSQL: 8.4+
- MS SQL: 2008 R2+

### Apache Zeppelin Notebooks and Configuration File Storage

In releases of Zeppelin earlier than HDP-2.6.3, notebooks and configuration files were stored on the local disk of the Zeppelin server. In HDP-2.6.3+, the default Zeppelin storage is HDFS.

When upgrading to HDP-2.6.3+ from versions earlier than HDP-2.6.3, perform the steps described in [Enabling HDFS Storage for Zeppelin Notebooks and Configuration in HDP-2.6.3+](#).

### More Information

[Rolling Upgrade Prerequisites \[37\]](#)

[Register and Install Target Version \[44\]](#)

[Remove Service](#)



## 5.1.1. Rolling Upgrade Prerequisites

To perform a **Rolling Upgrade**, your cluster must meet the following prerequisites. If your cluster does not meet these upgrade prerequisites, you can consider an **Express Upgrade**.

### HDFS

NameNode HA	NameNode HA must be enabled and working properly. From Ambari Web, the NameNodes, ZKFailoverControllers and JournalNodes must be "green" and showing no alerts.
NameNode Truncate	In HDP 2.2, truncate <b>must</b> be disabled and this prerequisite is <b>REQUIRED</b> . The hdfs-site configuration dfs.allow.truncate, property is set to true if truncate is enabled. If the property is not set or is false, truncate is disabled.  In HDP 2.3 and later, truncate is always enabled, which is acceptable. Therefore, this check is not applicable and will not be performed with HDP 2.3 maintenance upgrades.

### YARN

Timeline Service State Recovery	YARN state recovery should be enabled. Check the <b>Services &gt; YARN &gt; Configs &gt; Advanced</b> property yarn.timeline-service.recovery.enabled is set to true.
ResourceManager Work Preserving Recovery	YARN work preserving recovery should be enabled. Check the <b>Services &gt; YARN &gt; Configs &gt; Advanced</b> property yarn.resourcemanager.work-preserving-recovery.enabled is set to true.
ResourceManager HA	ResourceManager HA should be enabled to prevent a disruption in service during the upgrade.  This prerequisite is <b>OPTIONAL</b> .

### MapReduce2

MapReduce Distributed Cache	MapReduce should reference Hadoop libraries from the distributed cache in HDFS.
JobHistory State Recovery	JobHistory state recovery should be enabled. Check the following: <b>Services &gt; MapReduce &gt; Configs &gt; Advanced</b> property mapreduce.jobhistory.recovery.enable is set to true . Once enabled, dependent properties for mapreduce.jobhistory.recovery.store.class and mapreduce.jobhistory.recovery.store.leveldb.path should also be set.
Encrypted Shuffle	If encrypted shuffle has been enabled, the ssl-client.xml file must be copied to /etc/hadoop/conf/secure directory on each node in the cluster.

### Tez

Tez Distributed Cache      Tez should reference Hadoop libraries from the distributed cache in HDFS. Check the **Services > Tez > Configs** configuration and confirm `tez.lib.uris` is set with the path in HDFS for the `tez.tar.gz` file.

### Hive

Hive Dynamic Service Discovery      HiveServer2 dynamic service discovery should be configured for Rolling Upgrade.

Hive Client Retry      Hive client retry properties must be configured. Review the **Services > Hive > Configs** configuration and confirm `hive.metastore.failure.retries` and `hive.metastore.client.connect.retry.delay` are specified.

Multiple Hive Metastore      Multiple Hive Metastore instances are recommended for Rolling Upgrade. This ensures that there is at least one Hive Metastore running during the upgrade process. Additional Hive Metastore instances can be added through Ambari.

This prerequisite is **OPTIONAL**.

### Oozie

Oozie Client Retry      Oozie client retry properties must be configured. Review the **Services > Oozie > Configs > oozie-env** configuration and confirm `"export OOZIE_CLIENT_OPTS=${OOZIE_CLIENT_OPTS}-Doozie.connection.retry.count=<number of retries>"` is specified.

### Druid

Druid High Availability      Enable Druid high availability (HA) before performing a rolling upgrade. Multiple instances of Druid services running in the same cluster avoid downtime during the upgrade. If you choose not to enable HA for Druid services, a warning message appears but you can start the upgrade with the risk that Druid might encounter downtime.

### More Information

[Configuring NameNode High Availability.](#)

[ResourceManager High Availability.](#)

[YARN Resource Management](#)

## 5.2. Prepare to Upgrade

Make sure that you have reviewed and completed all [prerequisites](#) described in previous chapters.

It is **strongly** recommended that you perform backups of your databases before beginning upgrade.

- Ambari database
- Hive Metastore database
- Oozie Server database
- Ranger Admin database
- Ranger Audit database



### Important

If you use MySQL 5.6, you must use the InnoDB engine.

If your current MySQL engine is MyISAM, you must migrate to InnoDB before upgrading to Ambari 2.5.

- Export your current, MySQL (MyISAM) data.
- Drop the MySQL (MyISAM) database schema.
- Create a MySQL (InnoDB) database.
- Import your data into the MySQL (InnoDB) database instance.

For example:

```
mysqldump -U ambari -p ambari > /tmp/ambari.original.mysql
```

```
cp /tmp/ambari.original.mysql /tmp/ambari.innodb.mysql
```

```
sed -ie 's/MyISAM/INNODB/g' /tmp/ambari.innodb.mysql
```

```
mysql -u ambari -p ambari
```

```
DROP DATABASE ambari;
```

```
CREATE DATABASE ambari;
```

```
mysql -u ambari "-pbigdata" --force ambari < /tmp/ambari.innodb.mysql
```

Please contact Hortonworks customer support if you have issues running your cluster using MySQL 5.6.

## 5.2.1. Checkpoint HDFS

1. Perform the following steps on the NameNode host. If you are configured for NameNode HA, perform the following steps on the Active NameNode. You can locate the Active NameNode from **Ambari Web > Services > HDFS** in the **Summary** area.
2. Check the NameNode directory to ensure that there is no snapshot of any prior HDFS upgrade. Specifically, using Ambari Web, browse to **Services > HDFS > Configs**, and examine the `dfs.namenode.name.dir` in the NameNode Directories property. Make sure

that only a `/current` directory and no `/previous` directory exists on the NameNode host.

3. Create the following log and other files. Creating these logs allows you to check the integrity of the file system after the Stack upgrade.

As the HDFS user,

```
"su -l <HDFS_USER>"
```

run the following (where `<HDFS_USER>` is the HDFS Service user, for example, `hdfs`):

- Run `fsck` with the following flags and send the results to a log. The resulting file contains a complete block map of the file system. You use this log later to confirm the upgrade.

```
hdfs fsck / -files -blocks -locations > dfs-old-fsck-1.log
```

- Create a list of all the DataNodes in the cluster.

```
hdfs dfsadmin -report > dfs-old-report-1.log
```

- **Optional:** Capture the complete namespace of the file system. The following command does a recursive listing of the root file system:

```
hdfs dfs -ls -R / > dfs-old-lsr-1.log
```

- **Optional:** Copy all unrecoverable data stored in HDFS to a local file system or to a backup instance of HDFS.

4. Save the namespace. As the HDFS user, `"su -l <HDFS_USER> "`, you must put the cluster in Safe Mode.

```
hdfs dfsadmin -safemode enter
```

```
hdfs dfsadmin -saveNamespace
```



### Note

In a highly-available NameNode configuration, the command

```
hdfs dfsadmin -saveNamespace
```

sets a checkpoint in the first NameNode specified in the configuration, in `dfs.ha.namenodes.[nameserviceID]`.

You can also use the

```
dfsadmin -fs
```

option to specify which NameNode to connect. For example, to force a checkpoint in NameNode2:

```
hdfs dfsadmin -fs hdfs://namenode2-hostname:namenode2-port -saveNamespace
```

5. Copy the checkpoint files located in `${dfs.namenode.name.dir}/current` into a backup directory.



### Note

In a highly-available NameNode configuration, the location of the checkpoint depends on where the

```
saveNamespace
```

command is sent, as defined in the preceding step.

6. Store the layoutVersion for the NameNode located at

```
${dfs.namenode.name.dir}/current/VERSION, into a backup directory where
```

```
${dfs.namenode.name.dir} is the value of the config parameter NameNode directories.
```

This file will be used later to verify that the layout version is upgraded.

7. As the HDFS user, " su -l <HDFS\_USER> ", take the NameNode out of Safe Mode.

```
hdfs dfsadmin -safemode leave
```

8. Finalize any prior HDFS upgrade, if you have not done so already. As the HDFS user, " su -l <HDFS\_USER> ", run the following:

```
hdfs dfsadmin -finalizeUpgrade
```



### Important

In HDP-2.5, Audit to DB is no longer supported in Apache Ranger. If you are upgrading from HDP-2.3 or HDP-2.4 to HDP-2.5, you should use Apache Solr for Ranger audits. You should also migrate your audit logs from DB to Solr.

#### Next Steps

[Preparing for Upgrade to Kafka 1.0 \[41\]](#)

[Register and Install Target Version \[44\]](#)

#### More Information

[Getting Ready to Upgrade Ambari and HDP](#)

[Using Apache Solr for Ranger audits](#)

[Migrate your audit logs from DB to Solr](#)

## 5.2.2. Preparing for Upgrade to Kafka 1.0

### About This Task

If you are upgrading and HDP cluster including Kafka, you must perform some manual steps to support the upgrade to Kafka 1.0 before you begin an Ambari-managed Rolling or Express upgrade.

### Steps for a Rolling Upgrade

1. In **Custom kafka-broker** section in Ambari Configs, click **Add Property** and create `inter.broker.protocol.version` with the following values and click **Save**.

```
Key: inter.broker.protocol.version  
Value: CURRENT_KAFKA_VERSION
```

Where `CURRENT_KAFKA_VERSION` is the version of Kafka you are currently running. For example, 0.10.1.

2. In **Custom kafka-broker** section in Ambari Configs, click **Add Property** and create `log.message.format.version` with the following values and click **Save**.

```
Key: log.message.format.version  
Value: CURRENT_MESSAGE_FORMAT_VERSION
```

Where `CURRENT_MESSAGE_FORMAT_VERSION` is the version of the message format you are currently using. For example, 0.10.1.

3. Perform a rolling restart of all Kafka Brokers.



#### Note

Validate the Kafka cluster after performing this restart.

4. Perform a Rolling upgrade of the HDP Stack to HDP 2.6.5.
5. After the Rolling upgrade is complete, change the value for `inter.broker.protocol.version` to 1.0 and save the updated value.
6. Perform rolling restart of all Kafka Brokers.
7. After the Rolling restart is complete, change value for `log.message.format.version` to 1.0 and save the updated value.
8. Perform a final Rolling restart of Kafka Brokers.

### Steps for an Express Upgrade

1. Stop Kafka.
2. In **Custom kafka-broker** section in Ambari Configs, click **Add Property** and create `inter.broker.protocol.version` with the following values and click **Save**.

```
Key: inter.broker.protocol.version  
Value: CURRENT_KAFKA_VERSION
```

Where `CURRENT_KAFKA_VERSION` is the version of Kafka you are currently running. For example, 0.10.1.

3. In **Custom kafka-broker** section in Ambari Configs, click **Add Property** and create `createlog.message.format.version` with the following values and click **Save**.

```
Key: log.message.format.version  
Value: CURRENT_MESSAGE_FORMAT_VERSION
```

Where `CURRENT_MESSAGE_FORMAT_VERSION` is the version of the message format you are currently using. For example, 0.10.1.

- Restart all Kafka Brokers.



### Note

Validate the Kafka cluster after performing this restart.

- Perform an Express upgrade to HDP 2.6.5.
- After the Express upgrade is complete, change the value for `inter.broker.protocol.version` to 1.0 and save the updated value.
- Perform rolling restart of all Kafka Brokers.
- After the Rolling restart is complete, change value for `log.message.format.version` to 1.0 and save the updated value.
- Perform a final Rolling restart of Kafka Brokers.



### Note

Older Scala consumers do not support the new message format introduced in Kafka 0.11. The newer Java consumer must be used to avoid performance issues or leverage exactly-once semantics.

## Additional Upgrade Guidance

"In Kafka versions 0.11.0.0 and later, unclean leader election is disabled by default. If you want to retain the previous default behavior, set the broker configuration `unclean.leader.election.enable` to true.

Due to a bug in code, If you are upgrading from earlier Kafka versions ( 0.9, 0.10.X.X) with `unclean.leader.election` is enabled (by-default it is enabled), then after express upgrade you may see some unavailable partitions.

### Workarounds

- Prior to upgrade, disable `unclean.leader.election` (set `unclean.leader.election.enable = false`) and do a rolling restart.
- Prior to upgrade, enable `unclean.leader.election` explicitly (set `unclean.leader.election.enable = true`) and do a rolling restart. After express upgrade, customers can decide to retain or remove the config, based on requirements.

If your existing cluster contains topics with the replication factor equal to one, you are encouraged to enable unclean leader election for those topics before upgrading the cluster. You can use `kafka-configs.sh` command to add topic specific configurations:

```
bin/kafka-configs.sh
--zookeeper zk.host.name:2181
--entity-type topics
--entity-name TEST_TOPIC
--alter
--add-config unclean.leader.election.enable=true
```

### More Information

[Apache Kafka 1.0 Upgrade Guidance](#)

## 5.3. Register and Install Target Version

Before you use Ambari to perform the stack upgrade, you must register the software repositories for the new target version with Ambari and then install the software on all hosts in the cluster.

### Register Target Version

#### Steps

1. Log in to Ambari.
2. Browse to **Admin > Stack and Versions**.
3. Click the **Versions** tab. You see the version currently running, marked as **Current**.



#### Note

The full version depends on the HDP version you are actually running. For example, if you are currently running the HDP 2.6.4.0 release, you would see something like **HDP-2.6.4.0-91** as the full version number.

4. Click **Manage Versions**.
5. Proceed to register a new version by clicking + **Register Version**.
6. Select the software version and method of delivery for your cluster.

- a. **Choose HDP Stack.**

Available HDP minor versions display on tabs. When you click a tab, Ambari displays available maintenance versions for that HDP Stack on a drop-down list. When you click a specific maintenance version, a list of available Services and their version displays in a table.

- b. **Choose HDP Version.**

If your Ambari host has internet access, available maintenance versions display as options in a drop-down list. If you have a Version Definition File (VDF) for a version that is not listed, you can click **Add Version...** and upload the VDF file. In addition, a **Default Version Definition** is also included in the list if you do not have Internet access



or are not sure which specific version to install. If you choose the **Default Version Definition**, you must enter a "two-digit Version Number" in the **Name** input field.

c. **Choose Repository Delivery Method.**

- Use private Repository

Using a private repository requires internet connectivity. To use the private software repositories, see the list of available HDP Repositories for each OS.

- Use Local Repository

Using a local repository requires that you have configured the software in a repository available in your network. If you are using a local repository, enter the Base URLs for your local repository.

7. Click **Save**.

8. Click **Go To Dashboard**, and browse back to **Stack and Versions > Versions**.

You will see the version currently running, marked **Current**, and the target version you just registered, for example **HDP-2.6.4.0** displaying an **Install** button. A registered Patch upgrade displays a bug fix icon to uniquely identify the patch.

### Next Steps

#### Install Target Version

##### Steps

1. Log in to Ambari.
2. Browse to **Admin > Stack and Versions**.
3. Click the **Versions** tab.
4. On a registered target version, click **Install** and click **OK** to confirm.

The Install version operation starts. This installs the target version on all hosts in the cluster. You can monitor the progress of the install by clicking the **Installing** link.



### Note

If you have Apache Ranger on your cluster, confirm that the `ambari-infra-solr-client` package is installed on the Ranger Admin host. The **Ambari Infra** service contains the default Solr instance used by Apache Ranger.

When the install completes, the **Upgrade** button replaces the **Install** button.

### Next Steps

[Perform the Upgrade \[53\]](#)

## More Information

[Managing Versions](#)

[Performing a Patch Upgrade \[46\]](#)

[Using a Local Repository](#)

[Ambari Infra](#)

### 5.3.1. Performing a Patch Upgrade

A patch release contains a unique software fix that affects one or more, but not all, services in your cluster. A four-digit release number identifies each patch release. Performing a patch upgrade applies changes only to those services in your cluster affected by the patch release.



#### Note

Applying a patch upgrade is supported only for clusters running HDP-2.6.4 and above.

#### Prerequisites

Before performing a patch upgrade, you must obtain from Hortonworks Customer Support, the specific VDF file associated with the patch release.

If a component with upgrade dependencies in the VDF file is not installed in the cluster, the patch upgrade cannot proceed. You must modify the VDF file to remove such upgrade dependencies. For example, in a cluster where a Hive patch upgrade is being performed, if SQOOP is not installed on the cluster, you must make the following change before the upgrade:

```
<available-services>
  <service idref="HIVE-121000"/>
  <service idref="TEZ-070"/>
  <service idref="MAPREDUCE2-273"/>
  <service idref="SQOOP-146"/>
</available-services>
```

must be modified to:

```
<available-services>
  <service idref="HIVE-121000"/>
  <service idref="TEZ-070"/>
  <service idref="MAPREDUCE2-273"/>
</available-services>
```

#### Steps

1. Register the patch release as a target version, using the Add Version option, just as you would during any upgrade.

Patch versions include a "bug" icon to indicate that the patch includes a bug fix.

	Stack	Versions	Upgrade History
	<a href="#">Stack and Versions</a> <a href="#">Service Accounts</a> <a href="#">Kerberos</a> <a href="#">Service Auto Start</a>		<a href="#">Manage Versions</a> Filter: All (2)
		<b>Current</b> <b>HDP-2.6.4.0</b> <a href="#">Show Details</a>	<b>Current</b> <b>HDP-2.6.4.1-23</b> <a href="#">Show Details</a>
HDFS	2.7.3	2.7.3	2.7.3
YARN	2.7.3	2.7.3	2.7.3
MapReduce2	2.7.3	2.7.3	2.7.3
Tez	0.7.0	0.7.0	0.7.0
HBase	1.1.2	1.1.2	1.1.2
ZooKeeper	3.4.6	3.4.6	3.4.6
Ambari Infra	0.1.0	0.1.0	0.1.0
Ambari Metrics	0.1.0	0.1.0	0.1.0
Atlas	0.8.0	0.8.0	0.8.0
Kafka	0.10.1	0.10.1	0.10.1
Ranger	0.7.0	0.7.0	0.7.0
SmartSense	1.4.3.2.6.1.0-134	1.4.3.2.6.1.0-134	1.4.3.2.6.1.0-134
Kerberos	1.10.3-10	1.10.3-10	1.10.3-10

2. Install the patch release as the current version, just as you would during any upgrade.
3. Restart affected services.

When you perform a Patch Express Upgrade of a service, a dependent service may stop and require a manual restart. For example, a patch Express Upgrade of HDFS could stop HBase. In such cases, you must manually re-start HBase using the Ambari Web UI > Service Actions menu.

### More Information

[Limitation: Patch Upgrades in Mixed-OS Environments \[48\]](#)

[Register and Install Target Version \[44\]](#)

[Perform the Upgrade \[53\]](#)

[Performing Service Actions](#)

[Reverting a Patch Upgrade \[48\]](#)

## 5.3.2. Limitation: Patch Upgrades in Mixed-OS Environments

### Summary

Applying a patch upgrade to a cluster with more than one version of an operating system, (mixed OS) may require that you manually rebuild tarballs to include forward-compatible native libraries and upload them to a location accessible by both operating systems.

### Problem Description

When you apply an HDP patch in a mixed OS environment (of the same OS family, for example RHEL 6 and RHEL 7), and if master components like Job History Server or Hive server are located on a RHEL7 machine, native libraries for RHEL7 are packaged in both the `mapreduce.tar.gz` and `tez.tar.gz` tarballs. Worker nodes running RHEL6 in the cluster will be unable to load the RHEL7 native libraries.

### Workaround

You must fetch any `mapreduce.tar.gz` and `tez.tar.gz` tarballs from a RHEL7 machine and rebuild them manually on a RHEL 6 machine. RHEL6 native libraries are forward-compatible. You must then upload the rebuilt tarballs to the correct locations in HDFS.

- Locate and extract `/usr/hdp/<version>/hadoop/mapreduce.tar.gz`
- Copy the files directory `/usr/hdp/<version>/hadoop/lib/native` into the `hadoop/lib` directory from the tarball so that `hadoop/lib/native` exists and contains all of the files and folders from the above location.
- Re-compress the tarball and upload it into HDFS at `/hdp/apps/<version>/mapreduce/mapreduce.tar.gz`
- Locate and extract `/usr/hdp/<version>/tez/lib/tez.tar.gz`
- Copy the files directory `/usr/hdp/<version>/hadoop/lib/native` into the `lib` directory from the tarball so that `lib/native` exists and contains all of the files and folders from the above location.
- Re-compress the tarball and upload it into HDFS at `/hdp/apps/<version>/tez/tez.tar.gz`

## 5.3.3. Reverting a Patch Upgrade

After installing a patch upgrade, you have the option to revert the fix. You can only revert the last installed version. If you have installed multiple patch releases, you must revert them in the reverse order you installed them.

### Steps

1. In the current patch version, choose Revert.

	Current	Current
	<b>HDP-2.6.4.0</b> <a href="#">Show Details</a>	<b>HDP-2.6.4.1-23</b> <a href="#">Show Details</a>
HDFS	2.7.3	2.7.3
YARN	2.7.3	2.7.3
MapReduce2	2.7.3	2.7.3
Tez	0.7.0	0.7.0
Hive	1.2.1000	1.2.1000
HBase	1.1.2	1.1.2
Pig	0.16.0	0.16.0
ZooKeeper	3.4.6	3.4.6
Ambari Infra	0.1.0	0.1.0
Ambari Metrics	0.1.0	0.1.0
Atlas	0.8.0	0.8.0
Kafka	0.10.1	0.10.1
SmartSense	1.4.3.2.6.1.0-134	1.4.3.2.6.1.0-134
Druid	0.10.1	0.10.1
Kerberos	1.10.3-10	1.10.3-10
Slider	0.92.0	0.92.0

2. Restart affected services.

When you perform a Patch Express Upgrade of a service, a dependent service may stop and require a manual restart. For example, a patch Express Upgrade of HDFS could stop HBase. In such cases, you must manually re-start HBase using the Ambari Web UI > Service Actions menu.

**More Information**

[Performing Service Actions](#)

## 5.4. Optional: Remove Duplicate Ranger Entries

### Prerequisites

[Register and Install Target Version \[44\]](#).

### About this task

HDP 2.6.3 introduces unique constraints on a few tables in Ranger DB. Depending upon your environment, Ranger DB may contain duplicate data in these tables prior to the upgrade. To make the upgrade faster, you can manually delete this duplicate data in Ranger DB before performing the upgrade.

These steps are optional, but recommended, and only needed for Ranger users. These should be performed after registering and installing the target HDP version but before actually performing the upgrade.

	HDP-2.5.7.0 <a href="#">Show Details</a>	HDP-2.6.3.0 <a href="#">Show Details</a>
	<b>Current</b>	<b>Upgrade</b>
HDFS	2.7.3	2.7.3
ZooKeeper	3.4.6	3.4.6
Ambari Infra	0.1.0	0.1.0
Ambari Metrics	0.1.0	0.1.0
Ranger	0.6.0	0.7.0
SmartSense	1.4.2.2.5.2.0-298	1.4.2.2.5.2.0-298

### Steps

1. Verify that you can see both ranger versions (in different folders under `/usr/hdp`) before the upgrade:



### Note

Build 198 in `2.6.4.0-198` is an example; note the actual build version from the repository. Use the patch files found in the new version folder for this step.

```
ls -ltr /usr/hdp
ls -ltr /usr/hdp/current/ranger-*
ls -ltr /usr/hdp/2.6.4.0-<198>/ranger-admin/db
ls -ltr /usr/hdp/2.6.4.0-<198>/ranger-admin/db/mysql/patches/028-delete-xgroup-duplicate-references.sql
```

```
[root@pk-source-89877-1 ~]# ls -ltr /usr/hdp/
total 16
drwxr-xr-x. 3 root root 4096 Oct 16 05:04 share
drwxr-xr-x. 16 root root 4096 Oct 16 05:26 2.5.3.0-37
drwxr-xr-x. 2 root root 4096 Oct 16 05:29 current
drwxr-xr-x. 16 root root 4096 Oct 16 06:09 2.6.3.0-198
[root@pk-source-89877-1 ~]# ls -ltr /usr/hdp/current/ranger-*
lrwxrwxrwx. 1 root root 35 Oct 16 05:29 /usr/hdp/current/ranger-usersync -> /usr/hdp/2.5.3.0-37/ranger-usersync
lrwxrwxrwx. 1 root root 34 Oct 16 05:29 /usr/hdp/current/ranger-tagsync -> /usr/hdp/2.5.3.0-37/ranger-tagsync
lrwxrwxrwx. 1 root root 30 Oct 16 05:29 /usr/hdp/current/ranger-kms -> /usr/hdp/2.5.3.0-37/ranger-kms
lrwxrwxrwx. 1 root root 32 Oct 16 05:29 /usr/hdp/current/ranger-admin -> /usr/hdp/2.5.3.0-37/ranger-admin
[root@pk-source-89877-1 ~]# ls -ltr /usr/hdp/2.6.3.0-198/ranger-admin/db
db/
  dba_script.py  dba_script.pyc  dba_script.pyo  db_setup.py  db_setup.pyc  db_setup.pyo
[root@pk-source-89877-1 ~]# ls -ltr /usr/hdp/2.6.3.0-198/ranger-admin/db/mysql/patches/028-delete-xgroup-duplicate-references.sql
-r-xr--r--. 1 root root 3279 Oct 16 03:18 /usr/hdp/2.6.3.0-198/ranger-admin/db/mysql/patches/028-delete-xgroup-duplicate-references.sql
```

## 2. Check for duplicate entries in x\_group and x\_group\_user table:

- If there are duplicate groups entries in the x\_group table, this sql command will return the list of groups that are duplicates: `SELECT group_name, count(1) duplicateCount FROM x_group GROUP BY group_name HAVING duplicateCount>1;`
- If there are duplicate group-user mapping entries in the x\_group\_users table, this sql command will return the list of group-user mapping that are duplicates: `SELECT group_name, user_id, count(1) duplicateCount FROM x_group_users GROUP BY group_name, user_id HAVING duplicateCount>1;`

```
mysql>
mysql> SELECT group_name, count(1) duplicateCount FROM x_group GROUP BY group_name HAVING duplicateCount>1;
+-----+-----+
| group_name | duplicateCount |
+-----+-----+
| group-10 | 3 |
| group-11 | 3 |
| group-12 | 3 |
| group-13 | 3 |
| group-9 | 3 |
+-----+-----+
5 rows in set (0.00 sec)

mysql> SELECT group_name, user_id, count(1) duplicateCount FROM x_group_users GROUP BY group_name, user_id HAVING duplicateCount>1;
+-----+-----+-----+
| group_name | user_id | duplicateCount |
+-----+-----+-----+
| group-10 | 16 | 9 |
| group-11 | 17 | 9 |
| group-12 | 18 | 9 |
| group-13 | 19 | 9 |
| group-9 | 15 | 9 |
+-----+-----+-----+
5 rows in set (0.00 sec)
```

## 3. Delete duplicate groups and its references entries:

- Change your working directory to the current ranger-admin directory: `cd /usr/hdp/current/ranger-admin.`
- Locate the SQL patch files in the new ranger location, according to your DB flavor:



## Note

In these examples, the location of patch `028-delete-xgroup-duplicate-references.sql` at the new ranger location of my cluster: `/usr/hdp/2.6.4.0-198/ranger-admin/db/mysql/patches/028-delete-xgroup-duplicate-references.sql`

- **MySQL DB:**

```
java -cp /usr/share/java/mysql-connector-java.jar:/usr/hdp/current/ranger-admin/jisql/lib/* org.apache.util.sql.Jisql
-driver mysqlconj -cstring jdbc:mysql://localhost/ranger
-u 'rangeradmin' -p 'password' -noheader -trim -c \; -
input /path/to/db/mysql/patches/028-delete-xgroup-duplicate-
references.sql
```

- **Oracle DB:**

```
java -Djava.security.egd=file:///dev/urandom -cp /usr/share/
java/ojdbc6.jar:/usr/hdp/current/ranger-admin/jisql/lib/
* org.apache.util.sql.Jisql -driver oraclethin -cstring
jdbc:oracle:thin:@localhost -u 'rangeradmin' -p 'password' -
noheader -trim -input /path/to/db/oracle/patches/028-delete-
xgroup-duplicate-references.sql -c /
```

- **Postgres DB:**

```
java -cp /usr/share/java/postgresql.jar:/usr/hdp/current/
ranger-admin/jisql/lib/* org.apache.util.sql.Jisql -driver
postgresql -cstring jdbc:postgresql://localhost/ranger -
u rangeradmin -p 'password' -noheader -trim -c \; -input /
path/to/db/posgres/patches/028-delete-xgroup-duplicate-
references.sql
```

- **SQL Anywhere DB:**

```
java -cp /opt/sqlanywhere17/java/sajdbc4.jar:/usr/hdp/
current/ranger-admin/jisql/lib/* org.apache.util.sql.Jisql -
user rangeradmin -password 'password' -driver sapsajdbc4 -
cstring jdbc:sqlanywhere:database=ranger;host=localhost -
noheader -trim -input /path/to/db/sqlanywhere/patches/028-
delete-xgroup-duplicate-references.sql
```

- **MSSQL/SQL Server DB:**

```
java -cp /usr/share/java/sqljdbc4.jar:/usr/hdp/current/
ranger-admin/jisql/lib/* org.apache.util.sql.Jisql -
user rangeradmin -p 'password' -driver mssql -cstring
jdbc:sqlserver://localhost\;databaseName=ranger -noheader -
trim -input /path/to/db/sqlserver/patches/028-delete-xgroup-
duplicate-references.sql
```



4. Verify that the duplicate entries from the `x_group` and `x_group_user` table have been deleted:

```
mysql> use ranger;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> SELECT group_name,user_id,count(1) duplicateCount FROM x_group_users
  GROUP BY group_name,user_id HAVING duplicateCount>1;
Empty set (0.00 sec)

mysql> SELECT group_name,count(1) duplicateCount FROM x_group GROUP BY
  group_name HAVING duplicateCount>1;
Empty set (0.00 sec)
```

```
mysql> use ranger;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> SELECT group_name,user_id,count(1) duplicateCount FROM x_group_users GROUP BY group_name,user_id HAVING duplicateCount>1;
Empty set (0.00 sec)

mysql> SELECT group_name,count(1) duplicateCount FROM x_group GROUP BY group_name HAVING duplicateCount>1;
Empty set (0.00 sec)
```

### Next steps

[Perform the Upgrade \[53\]](#)

## 5.5. Perform the Upgrade

To upgrade your HDP version:

### Steps

1. Log in to Ambari.
2. Browse to **Admin > Stack and Versions**.
3. Click the **Versions** tab.

The registered and installed target HDP version displays a Perform Upgrade button.

4. Click **Perform Upgrade** on the target version.

Based on your current HDP version and the target HDP version, Ambari performs a set of prerequisite checks to determine if you can perform a rolling or an express upgrade. A dialog displays the options available.

5. Select your upgrade method: **Rolling Upgrade** or **Express Upgrade**.

Advanced options are also available.

Skip all Service Check failures

Ambari automatically skips any Service Check failures and completes the task without requiring user actions. After all the Service Checks have run in a

task, you see a summary of the failures and options to continue the upgrade or pause.

**Skip all Slave Component failures** Ambari automatically skips any Slave Component failures and completes the task of upgrading Slave components without requiring user actions. After all Slave Components have been upgraded, you see a summary of the failures and options to continue the upgrade or pause.

#### 6. Click **Proceed**.

Based on your option choice, proceed with the upgrade.

#### Next Steps

[Perform Rolling Upgrade \[54\]](#)

[Perform Express Upgrade \[56\]](#)

#### More Information

[Register and Install Target Version \[44\]](#)

[Prerequisites \[35\]](#)

## 5.5.1. Perform Rolling Upgrade

1. Ambari checks that your cluster meets prerequisites. A dialog displays the results:
  - a. If any *required* prerequisites are not met, the result displays an error.
 

You cannot proceed with the upgrade until you make the appropriate corrections and return to Perform Upgrade again.
  - b. If any *optional* prerequisites are not met, the result displays a warning.
 

You may proceed with the upgrade.
  - c. Ambari displays a list of configuration changes that occur during the upgrade.
2. When the prerequisite checks complete, the upgrade starts. The time required to perform the upgrade depends on many factors. As part of the upgrade process, each component in the cluster restarts in a serial fashion. The stop/start times contribute to the total upgrade time.
3. The upgrade process includes the following stages. Some stages require that you complete an action during normal operation.

If any stage fails, the upgrade stops and prompts you for action.

Stage	Description	Action Required
Prepare Backups	This step prompts you to confirm that you have taken proper backups before proceeding.	You must acknowledge the prompt for database backups.

Stage	Description	Action Required
ZooKeeper	All ZooKeeper servers are upgraded and restarted.	None
Ranger	Ranger Admin and UserSync servers are upgraded and restarted.	None. If Ranger Admin does not function after the upgrade completes, run the upgrade scripts manually, then Retry Upgrading Ranger.
Core Masters	This stage upgrades the master components of core services. This includes JournalNodes & NameNodes (HDFS), HistoryServer (MapReduce2), ResourceManager & ATS (YARN) and HBase Masters (HBase).	None
Druid	This stage upgrades Druid components in the following order: historical, middle manager, broker, router, coordinator, overlord.	None
All Service Checks	All Service Checks are performed against the cluster.	Any service check that fail prompt you to <b>Ignore and Continue, Downgrade or Retry</b> . If you selected the <b>Skip all Service Check failures</b> option, you will only be prompted when all Service Checks complete.
Core Slaves	This stage upgrades the slave components of core services. This includes DataNodes (HDFS), RegionServers (HBase) and NodeManagers (YARN). This is done in two batches: 20% of the slaves first, then the remaining slaves.	After the first 20% batch completes, you are prompted to verify the cluster is operating correctly.
All Service Checks	All Service Checks are performed against the cluster.	Any service check that fail prompt you to <b>Ignore and Continue, Downgrade or Retry</b> . If you selected the <b>Skip all Service Check failures</b> option, you will only be prompted when all Service Checks complete.
Spark	The Spark Job History Server and clients are upgraded.	None
Oozie	The Oozie Server and clients are upgraded.	None
Falcon	The Falcon Server and clients are upgraded.	None
Client Components	All remaining clients are upgraded.	None
All Service Checks	All Service Checks are performed against the cluster.	Any service check that fails prompts you to <b>Ignore and Continue, Downgrade or Retry</b> . If you selected the <b>Skip all Service Check failures</b> option, you will only be prompted when all Service Checks complete.
Kafka	The Kafka Brokers are upgraded.	None
Knox	The Knox Gateways are upgraded.	In an SSO-enabled cluster: If the cookie is lost/session is expired:client should use local login to access the Ambari and proceed further.  For example: <ambari_host:ambari_port>/#/login/local
Storm	Storm does not support rolling upgrade from a previous HDP version to HDP-2.5.	The rolling upgrade process prompts you to stop Storm topologies, perform the upgrade, and redeploy your topologies.
Slider	The Slider clients are upgraded.	None
Flume	The Flume agents are upgraded.	None
Finalize Upgrade Pre-Check	Checks if any hosts were not upgraded, either because the host was in Maintenance Mode, or one or more components on the host failed to upgrade (and were skipped).	Click the list that displays <b># hosts</b> for details on the hosts (and their components) that are not upgraded. You can <b>Pause Upgrade</b> , delete the hosts and return to finalize.

Stage	Description	Action Required
Finalize	The component upgrades are complete. You are prompted to <b>Finalize</b> . Finalizing completes the upgrade process and saves the cluster state.	Prompted to Finalize, Finalize Later or Downgrade.

- When the rolling upgrade stages complete, 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).

- Click **Finalize** to complete the rolling upgrade process.

### More Information

[Rolling Upgrade Prerequisites \[37\]](#)

[Retry Ranger Upgrade](#)

## 5.5.2. Perform Express Upgrade

- Ambari checks that your cluster meets prerequisites. A dialog displays the results:
  - If any *required* prerequisites are not met, the result displays an error.
 

You cannot proceed with the upgrade until you make the appropriate corrections and return to Perform Upgrade again.
  - If any *optional* prerequisites are not met, the result displays a warning.
 

You may proceed with the upgrade.
  - Ambari displays a list of configuration changes that occur during the upgrade.
- When the prerequisite checks complete, the upgrade starts. The time required to perform the upgrade depends on many factors. As part of the upgrade process, each component in the cluster restarts in a serial fashion. The stop/start times contribute to the total upgrade time.
- The upgrade process includes the following stages. Some stages require that you complete an action during normal operation.
 

If any stage fails, the upgrade stops and prompts you for action.

Stage	Description	Action Required
Prepare Upgrade	You should stop all YARN queues, all long-running applications on Slider, and deactivate & kill all running Storm topologies.	Perform the actions to prepare for the upgrade.
Stop Components for High-Level Services	This will stop all components for High-Level Services. This includes all master components <b>except</b> those of HDFS, HBase, ZooKeeper and Ranger.	None
Perform Backups	This step prompts you to confirm that you have taken proper backups before proceeding.	You must acknowledge the prompt for database backups.
Stop Components for Core Service	Stops all components with HDFS, HBase, ZooKeeper and Ranger.	None
Update Target Stack	Updates the stack version in Ambari to the target version. There is no downgrade past this point.	None
Update Service Configs	Updates (i.e. transfers or replaces) any configurations that are necessary for the upgrade.	None
Restart Components	Restarts all core components such as ZooKeeper, Ranger, HDFS, YARN, MapReduce2 and various Clients (Tez, Pig, Sqoop).	In an SSO-enabled cluster: If the cookie is lost/session is expired:client should use local login to access the Ambari and proceed further.  For example: <ambari_host:ambari_port>/#/login/local
All Service Checks	All Service Checks are performed against the cluster.	Any service check that fails prompts you to <b>Ignore and Continue</b> , <b>Downgrade</b> or <b>Retry</b> . If you selected the <b>Skip all Service Check failures</b> option, you will only be prompted when all Service Checks complete.
Restart Components	Restarts the remaining components such as Oozie, Falcon, Hive, Spark and others.	None
Set Version on All Hosts	Sets the HDP version on all hosts to the target HDP version.	None
Finalize Upgrade Pre-Check	Checks if any hosts were not upgraded, either because the host was in Maintenance Mode, or one or more components on the host failed to upgrade (and were skipped).	Click the list that displays # hosts for details on the hosts (and their components) that are not upgraded. You can <b>Pause Upgrade</b> , delete the hosts and return to finalize.
Finalize Upgrade	The component upgrades are complete. You are presented the option to Finalize, which when selected, completes the upgrade process + saves the cluster state.	Prompted to Finalize or Finalize Later or Downgrade.

4. When the rolling upgrade stages complete, 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).

5. Click **Finalize** to complete the express upgrade process.

### More Information

[Register and Install Target Version \[44\]](#)

[Prerequisites \[35\]](#)

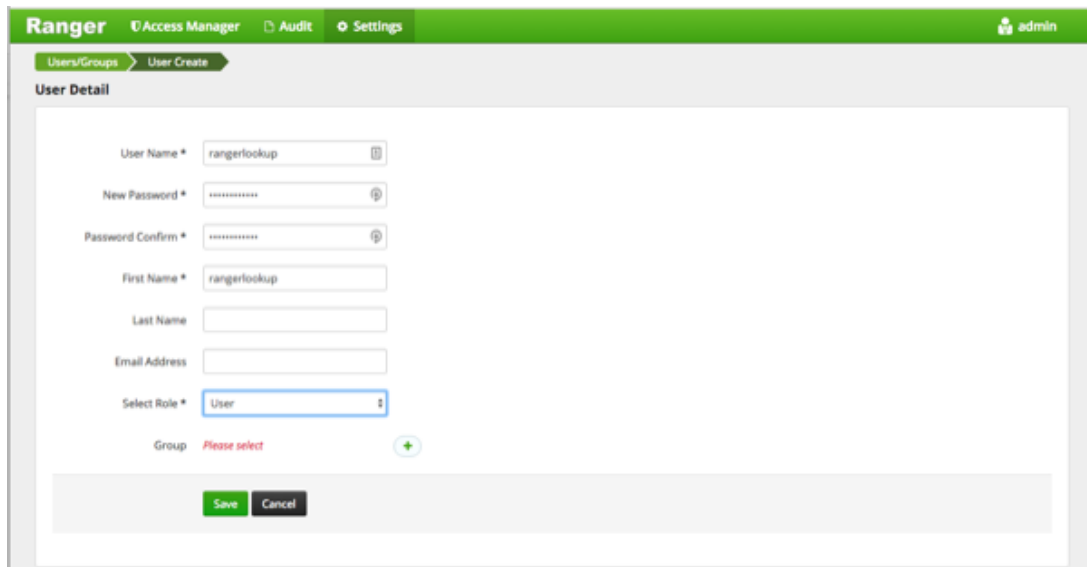
## 5.6. Post-upgrade Tasks

### Post-upgrade Tasks for Ranger Usersync

For HDP-2.6 and higher, Ranger Usersync supports incremental sync for LDAP sync. For users upgrading with LDAP sync, the `ranger.usersync.ldap.deltasync` property (found in Ranger User Info with the label "Incremental Sync") will be set to `false`. This property is set to `true` in new installs of Ranger in HDP-2.6 and higher.

### Post-upgrade Tasks for Ranger with Kerberos

1. If you have not already done so, you should migrate your audit logs from DB to Solr.
2. After successfully upgrading, you must regenerate keytabs. Select the **Only regenerate keytabs for missing hosts and components** check box, confirm the keytab regeneration, then restart all required services.
3. Log in to the Ranger Admin UI and select **Settings > Users/Groups**. Select the **Users** tab, then click **Add New User**. On the **User Detail** page, enter the short-name of the principal `ranger.lookup.kerberos.principal` (if that user does not already exist) in the **User Name** box. For example, if the lookup principal is `rangerlookup/_HOST@${realm}`, enter `rangerlookup` as the user name. Enter the other settings shown below, then click **Save** to create the user. Add the new `rangerlookup` user in Ranger policies and services. You can use **Test Connection** on the **Services** pages to check the connection.



The screenshot shows the Ranger Admin UI interface for creating a new user. The top navigation bar includes 'Ranger', 'Access Manager', 'Audit', and 'Settings', with a user profile icon for 'admin'. The breadcrumb trail is 'Users/Groups > User Create'. The main section is titled 'User Detail' and contains the following fields:

- User Name \*: rangerlookup
- New Password \*: [masked]
- Password Confirm \*: [masked]
- First Name \*: rangerlookup
- Last Name: [empty]
- Email Address: [empty]
- Select Role \*: User
- Group: Please select (with a green plus icon)

At the bottom of the form are 'Save' and 'Cancel' buttons.

## Additional Steps for Ranger HA with Kerberos

If Ranger HA (High Availability) is set up along with Kerberos, perform the following additional steps:

1. Use SSH to connect to the KDC server host. Use the

```
kadmin.local
```

command to access the Kerberos CLI, then check the list of principals for each domain where Ranger Admin and the load-balancer are installed.

```
kadmin.local
kadmin.local: list_principals
```

For example, if Ranger Admin is installed on <host1> and <host2>, and the load-balancer is installed on <host3>, the list returned should include the following entries:

```
HTTP/ <host3>@EXAMPLE.COM HTTP/ <host2>@EXAMPLE.COM HTTP/
<host1>@EXAMPLE.COM
```

If the HTTP principal for any of these hosts is not listed, use the following command to add the principal:

```
kadmin.local: addprinc -randkey HTTP/<host3>@EXAMPLE.COM
```



### Note

This step will need to be performed each time the Spnego keytab is regenerated.

2. Use the following `kadmin.local` commands to add the HTTP Principal of each of the Ranger Admin and load-balancer nodes to the Spnego keytab file:

```
kadmin.local: ktadd -norandkey -kt /etc/security/keytabs/spnego.service.
keytab HTTP/ <host3>@EXAMPLE.COM
kadmin.local: ktadd -norandkey -kt /etc/security/keytabs/spnego.service.
keytab HTTP/ <host2>@EXAMPLE.COM
kadmin.local: ktadd -norandkey -kt /etc/security/keytabs/spnego.service.
keytab HTTP/ <host1>@EXAMPLE.COM
```

Use the `exit` command to exit `ktadmin.local`.

3. Run the following command to check the Spnego keytab file:

```
klist -kt /etc/security/keytabs/spnego.service.keytab
```

The output should include the principals of all of the nodes on which Ranger Admin and the load-balancer are installed. For example:

```
Keytab name: FILE:/etc/security/keytabs/spnego.service.keytab
KVNO Timestamp Principal ----
-----
1 07/22/16 06:27:31 HTTP/ <host3>@EXAMPLE.COM
1 07/22/16 06:27:31 HTTP/ <host3>@EXAMPLE.COM
1 07/22/16 06:27:31 HTTP/ <host3>@EXAMPLE.COM
```

```

1 07/22/16 06:27:31 HTTP/ <host3>@EXAMPLE.COM
1 07/22/16 06:27:31 HTTP/ <host3>@EXAMPLE.COM
1 07/22/16 08:37:23 HTTP/ <host2>@EXAMPLE.COM
1 07/22/16 08:37:23 HTTP/ <host2>@EXAMPLE.COM
1 07/22/16 08:37:23 HTTP/ <host2>@EXAMPLE.COM
1 07/22/16 08:37:23 HTTP/ <host2>@EXAMPLE.COM
1 07/22/16 08:37:23 HTTP/ <host2>@EXAMPLE.COM
1 07/22/16 08:37:23 HTTP/ <host2>@EXAMPLE.COM
1 07/22/16 08:37:23 HTTP/ <host2>@EXAMPLE.COM
1 07/22/16 08:37:35 HTTP/ <host1>@EXAMPLE.COM
1 07/22/16 08:37:36 HTTP/ <host1>@EXAMPLE.COM 1
07/22/16 08:37:36 HTTP/ <host1>@EXAMPLE.COM 1 07/22/16
08:37:36 HTTP/ <host1>@EXAMPLE.COM 1 07/22/16 08:37:36
HTTP/ <host1>@EXAMPLE.COM 1 07/22/16 08:37:36 HTTP/
<host1>@EXAMPLE.COM

```

- Use `scp` to copy the Spnego keytab file to every node in the cluster on which Ranger Admin and the load-balancer are installed. Verify that the `/etc/security/keytabs/spnego.service.keytab` file is present on all Ranger Admin and load-balancer hosts.
- On the Ambari dashboard, select **Ranger > Configs > Advanced**, then select **Advanced ranger-admin-site**. Set the value of the `ranger.spnego.kerberos.principal` property to `*`.



- Click **Save** to save the configuration, then restart Ranger Admin, only if you have completed all applicable post-upgrade tasks.

## Ranger KMS

When upgrading Ranger KMS in an SSL environment to HDP-2.6, the properties listed in Step 2 under "Configuring the Ranger KMS Server" on the [Enabling SSL for Ranger KMS](#) will be moved to the "Advanced ranger-kms-site" section.

## More Information

[Migrate Audit logs from DB to Solr](#)

[How to Regenerate Keytabs](#)