Hortonworks Data Platform

Ambari Upgrade Guide

(Jul 21, 2015)

docs.hortonworks.com

Hortonworks Data Platform: Ambari Upgrade Guide

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.

Table of Contents

List of Tables

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
- Upgrading Ambari
- Upgrading HDP

Important

Ambari 2.1 does **not** support directly upgrading from HDP 2.0 or HDP 2.1 to HDP 2.3. In order to upgrade from HDP 2.0 or HDP 2.1, **you must first upgrade to HDP 2.2 using either Ambari 1.7 or 2.0**. Once completed, upgrade your current Ambari to Ambari 2.1. Then, leverage Ambari 2.1 to complete the upgrade from HDP 2.2 to HDP 2.3.

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 2.1 to HDP 2.2), upgrade Ambari to the lastest version before upgrading the cluster. Refer to the Stack Compatibility Matrix to see which HDP versions each Ambari version supports.

- 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. Refer to Using a Local Repository for more information.
- Ensure point-in-time backups are taken of all DBs supporting the clusters. This includes (among others) Ambari, Hive Metastore, Ranger and Oozie.

Note

If you upgraded Ambari from 2.0x to 2.1x, you must make sure that the Ranger db root password is NOT set to blank before performing any Stack upgrade. Ambari 2.1x requires that the Ranger db root password has a value. If you upgrade Ambari to 2.1x and upgrade the stack 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 Ambari Upgrades

- This (Ambari 2.1) Upgrade Guide will help you upgrade your existing Ambari install to version 2.1. 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 release in the Ambari 2.1 Release Notes.

For HDP Cluster Upgrades

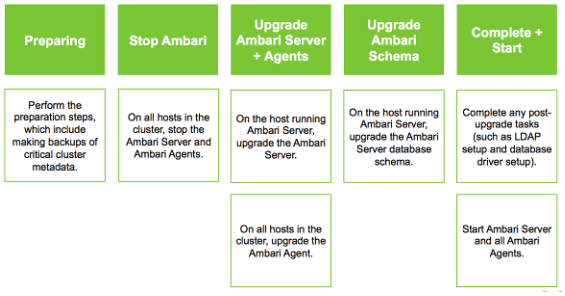
- Ensure sufficient disk space on `/usr/hdp/<version>` (roughly 3GB for each additional HDP release).
- New service user accounts and groups may be required/created during the upgrade. This includes service accounts such as ams, flume, and kafka. Any operational procedures required to support these new 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.

3. Upgrading Ambari

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

- Preparing to Upgrade
- Upgrading Ambari
- Post-Upgrade Tasks

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



3.1. Preparing to Upgrade

- Be sure to review the Release Notes for this Ambari release 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`.
- If you are using Ambari with JDK 1.6, **you must first move to JDK 1.7 before attempting to upgrade to Ambari 2.1**. Refer the the **Ambari Reference Guide** for the Ambari version you are currently running for instructions on **Changing the JDK**.
- **If you are managing Ganglia and Nagios from Ambari:**
  - Support for managing Ganglia and Nagios from Ambari **has been removed**.
  - Record the location of the Nagios server before you begin the upgrade process.
  - Record the location of the Ganglia server before you begin the upgrade process.

- You **must** stop the Nagios and Ganglia services from **Ambari Web**.
- After upgrading Ambari, you **must remove Nagios and Ganglia from your cluster and replace with Ambari Alerts and Metrics**. For more information, see Migrate to Ambari Alerts and Metrics in Ambari.

- **If you are on Ambari 1.7 or earlier:**
  - If your cluster has Kerberos enabled, you **must** review Adjusting for Kerberos and perform the pre- and post-upgrade steps required.
- **If you are on Ambari 2.0, or later:**
  - 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 service.
- Proceed to Upgrade to Ambari 2.1.

Note

If your current Ambari version is 1.4.1 or below, you must upgrade the Ambari Server version to 1.7 before upgrading to version 2.1.

Note

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

3.2. Upgrading Ambari

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

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

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

Important

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

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

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

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

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

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

- **For SLES 11:**

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

Note

If your cluster does not have access to the Internet, set up a local repository with this data before you continue. See Using a Local Repository for more information.

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. For more information about ports, see Configuring Network Port Numbers.

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 "2.1.0"

```
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 "2.1.0"

```
zypper up 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-2.1.0-101.noarch' to install this candidate". You will need to to use yast to update the package, as follows:

a. From the command line run: > yast.

```
> yast
```

You will see command line UI for YaST program.

b. Choose `Software > Software Management`, then click the `Enter` button.

c. In the `Search Phrase` field, enter `ambari-server`, then click the `Enter` button.

d. On the right side you will see the search result `ambari-server 2.1.0 .` 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 0:2.1.0-111
Complete!
```

Note

Confirm there is only one ambari-server*.jar file in `/usr/lib/ambari-server`. If there is more than one JAR file with name ambari-server*.jar, move all JARs except ambari-server-2.1.0.*.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-2.1.0-101.noarch' to install this candidate". You will need to to use yast to update the package, as follows:

   a. From the command line run: > yast

      ```
      > yast
      ```

      You will see command line UI for YaST program.

   b. Choose `Software > Software Management`, then click the `Enter` button.

   c. In the `Search Phrase` field, enter `ambari-agent`, then click the `Enter` button.

   d. On the right side you will see the search result `ambari-agent 2.1.0 .` Click `Actions`, choose `Update`, then click the `Enter` button.

   e. Go to `Accept`, and click `enter`.

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

   ```
   rpm -qa | grep 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`.

14. 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".

15. Review the HDP-UTILS repository Base URL setting in Ambari.

    If you are upgrading from Ambari 1.6.1 or earlier, the HDP-UTILS repository Base URL is no longer set in the `ambari.repo` file.

    If using HDP 2.2 Stack or later:

    - Browse to `Ambari Web > Admin > Stack and Versions`.
    - Click on the `Versions` tab.
    - You will see the current installed HDP Stack version displayed.
    - Click the `Edit` repositories icon in the upper-right of the version display and confirm the value of the HDP-UTILS repository Base URL is correct for your environment.
    - If you are using a local repository for HDP-UTILS, be sure to confirm the Base URL is correct for your locally hosted HDP-UTILS repository.

    If using HDP 2.0 or 2.1 Stack:

    - Browse to `Ambari Web > Admin > Repositories`.
    - Under the Services table, the current Base URL settings are displayed.
    - Confirm the value of the HDP-UTILS repository Base URL is correct for your environment or click the `Edit` button to modify the HDP-UTILS Base URL.
    - If you are using a local repository for HDP-UTILS, be sure to confirm the Base URL is correct for your locally hosted HDP-UTILS repository.

16. If you have configured Ambari to authenticate against an external LDAP or Active Directory, you **must** re-run "ambari-server setup-ldap". For more information, see Set Up LDAP or Active Directory Authentication.

17. 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. For more information, see Using Non-Default Databases - Hive and Using Non-Default Databases - Oozie.

18. **If your cluster includes Ganglia and Nagios**:

    You **must** remove Nagios and Ganglia from your cluster and replace with Ambari Alerts and Metrics. For more information, see Migrate to Ambari Alerts and Metrics.

19. **If you have upgraded from Ambari 1.7 or earlier:**

    - If your cluster was configured for Kerberos, you **must** adjust your cluster for Kerberos. For more information, see Adjusting for Kerberos.
    - If your cluster is running HDP 2.2 Stack, you **must** get the cluster hosts to advertise the "current version". This is done by restarting a master or slave component (such as a DataNode) on each host to have the host. For example, in **Ambari Web**, navigate to the **Hosts** page, select any Host that has the DataNode component and restart that DataNode component on that single host.

20. If you are running **Ambari Metrics** service in your cluster, Upgrade Ambari Metrics service.

3.3. Post-Upgrade Tasks

Depending on the configuration of your cluster and the Ambari version you are starting with, review the following post-upgrade tasks.

Task	Description
Migrate to Ambari Alerts and Metrics	If your cluster includes legacy Nagios and Ganglia services, they must be removed and replaced with Ambari Alerts and Metrics.
Adjusting for Kerberos	If you have upgraded from Ambari 1.7 or earlier and your cluster was configured for Kerberos, you must adjust Ambari.
Upgrading Ambari Metrics Service	If your cluster includes the Ambari Metrics System ("AMS") service, you must upgrade the system along with Ambari.

3.3.1. Migrate to Ambari Alerts and Metrics

Starting with Ambari 2.0, Ambari includes built-in systems for alerting and metrics collection. Therefore, when upgrading to Ambari 2.1, the legacy Nagios and Ganglia services must be removed and replaced with the new systems. **You only need to perform these steps and migrate to Ambari Alerts and Metrics if you are running Nagios and Ganglia**.

Important

We highly recommended that you perform and validate this procedure in a test environment prior to attempting the Ambari upgrade on production systems.

3.3.1.1. Moving from Nagios to Ambari Alerts

After upgrading to Ambari 2.1, the Nagios service will be removed from the cluster. The Nagios server and packages will remain on the existing installed host but Nagios itself is removed from Ambari management.

Important

Nagios used the operating system sendmail utility to dispatch email alerts on changes. With Ambari Alerts, the email dispatch is handled from the Ambari Server via Javamail. Therefore, you must provide SMTP information to Ambari for sending email alerts. Have this information ready. You will use it after the Ambari 2.1 upgrade to get Ambari Alert email notifications configured in the new Ambari Alerts system.

The Ambari Alerts system is configured automatically to replace Nagios but you must:

1. Configure email notifications in Ambari to handle dispatch of alerts. Browse to `Ambari Web > Alerts`.
2. In the Actions menu, select `Manage Notifications`.
3. Click to `Create a new Notification`. Enter information about the SMTP host, port to and from email addresses and select the Alerts to receive notifications.
4. Click `Save`.

Note

(Optional) Remove the Nagios packages (nagios, nagios-www) from the Nagios host.

For more information Ambari Alerts, see Managing Alerts in the Ambari User's Guide.

3.3.1.2. Moving from Ganglia to Ambari Metrics

After upgrading to Ambari 2.1, the Ganglia service stays intact in the cluster. You must perform the following steps to remove Ganglia from the cluster and to move to the new Ambari Metrics system.

Important

- If you are using HDP 2.2 Stack, Storm metrics will not work with Ambari Metrics until you are upgraded to HDP 2.2.4 or later.
- It is recommended that, after moving to Ambari Metrics restart Storm to pick up the new metrics sink classes.
- Do not add the Ambari Metrics service to your cluster until you have removed Ganglia using the steps below.

1. Stop Ganglia service via Ambari Web.

2. Using the Ambari REST API, remove the Ganglia service by executing the following:

```
curl -u admin.user:admin.password -H 'X-Requested-By:ambari' -X DELETE
 'http://ambari.server:8080/api/v1/clusters/cluster.name/services/GANGLIA'
```

   where **admin.user** and **admin.password** are credentials for an Ambari Administrator, **ambari.server** is the Ambari Server host and **cluster.name** is the name of your cluster.

3. Refresh Ambari Web and make sure that Ganglia service is no longer visible.

4. In the Actions menu on the left beneath the list of Services, use the "Add Service" wizard to add `Ambari Metrics` to the cluster.

5. This will install an `Ambari Metrics Collector` into the cluster, and an `Ambari Metrics Monitor` on each host.

6. Pay careful attention to following service configuration for Ambari Metrics.

   By default, Ambari Metrics uses the local filesystem as the default storage backend. This is **embedded mode** and the rootdir for HBase is set to a local filesystem path by default when using Ambari Metrics in **embedded mode**. If you want to run Ambari Metrics in **distributed mode** (i.e. use HDFS instead of local filesystem for storage), set this rootdir value to an HDFS dir. For example: hdfs://namenode.example.org:8020/amshbase. See Advanced Configurations for Ambari Metrics for more information.

Section	Property	Default Value
Advanced ams-hbase-site	hbase.rootdir	file:///var/lib/ambari-metrics-collector/hbase

7. For the cluster services to start sending metrics to Ambari Metrics, restart all services. For example, restart HDFS, YARN, HBase, Flume, Storm and Kafka.

Note

(Optional) Remove the Ganglia packages (ganglia-gmetad and ganglia-gmond) from the hosts.

Important

If you are managing a HDP 2.2 cluster that includes Kafka, you must adjust the Kafka configuration to send metrics to the Ambari Metrics system.

From Ambari Web, browse to `Services > Kafka > Configs` and edit the `kafka-env template` found under `Advanced kafka-env` to include the following:

```
# Add kafka sink to classpath and related dependencies

if [ -e "/usr/lib/ambari-metrics-kafka-sink/ambari-
metrics-kafka-sink.jar" ];

then

export CLASSPATH=$CLASSPATH:/usr/lib/ambari-metrics-kafka-
sink/ambari-metrics-kafka-sink.jar
```

```
export CLASSPATH=$CLASSPATH:/usr/lib/ambari-metrics-kafka-sink/lib/*

fi
```

3.3.2. Adjusting for Kerberos

Important

You **only** need to perform these Kerberos steps **if you are running Ambari 1.7 or earlier**. There is no need to perform these steps if you are running Ambari 2.0.

If you are upgrading to Ambari 2.1 from an Ambari-managed cluster that is already Kerberos-enabled, because of the new Kerberos features, you need perform the following steps after Ambari upgrade.

1. Review the procedure for Configuring Ambari and Hadoop for Kerberos in the Ambari Security Guide.
2. Have your Kerberos environment information readily available, including your KDC Admin account credentials.
3. Take note of current Kerberos security settings for your cluster.
   a. Browse to `Services > HDFS > Configs`.
   b. Record the core-site auth-to-local property value.
4. Upgrade Ambari according to the steps in Upgrading to Ambari 2.1.
5. Ensure your cluster and the Services are healthy.
6. Browse to `Admin > Kerberos` and you'll notice Ambari thinks that Kerberos is not enabled. Run the `Enable Kerberos Wizard`, following the instructions in the Ambari Security Guide. Be sure to pay close attention to the principal names in the Configure Identities step in the wizard to confirm principals names match what you expect for your environment.
7. Ensure your cluster and the Services are healthy.
8. Verify the Kerberos security settings for your cluster are correct.
   - Browse to `Services > HDFS > Configs`.
   - Check the core-site auth-to-local property value.
   - Adjust as necessary, based on the pre-upgrade value recorded in Step 3.

3.3.3. Upgrade Ambari Metrics

1. Upgrade to Ambari 2.1 and perform needed post-upgrade checks. Make sure all services are up and healthy.

2. Make sure Ambari Metrics service is stopped. From **Ambari Web**, browse to **Services > Ambari Metrics** and select **Stop** from the **Service Actions** menu.

3. On **each host** in your cluster running Ambari Metrics, 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
   ```

4. 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
   ```

5. Start Ambari Metrics Service. From `Ambari Web`, browse to `Services > Ambari Metrics` and select `Start` from the `Service Actions` menu.

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

   (For example: HDFS, YARN, Kafka, HBase, Flume, Storm)

4. Upgrading HDP

There are different HDP upgrade options based on your current HDP and the target HDP. This section describes the different upgrade options and their prerequisites.

Note

The HDP Stack version is based on the following format: major.minor.maintenance.patch. A "minor" release is a release that increments the second-digit. A "maintenance" release is one that increments the third-digit. This terminology is used to describe the different upgrade paths. For example: if you want to upgrade from HDP 2.2 to HDP 2.3, that is a Minor Upgrade. If you want to upgrade from HDP 2.2.x to HDP 2.2.y, that is a Maintenance Upgrade.

The following table describes the different upgrade options based on the current and target Stack combinations.

Table 4.1. HDP Stack Upgrade Options

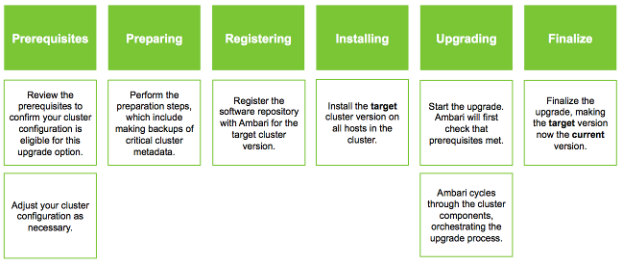
Upgrade Path	Current HDP	Target HDP	Upgrade Options
Minor Upgrade *	HDP 2.2	HDP 2.3	Rolling Upgrade HDP 2.2 to 2.3 or Manual Upgrade HDP 2.2 to 2.3
Maintenance Upgrade	HDP 2.2.x	HDP 2.2.y	Rolling Maintenance Upgrade or Manual Maintenance Upgrade
Maintenance Upgrade	HDP 2.3.x	HDP 2.3.y	Rolling Maintenance Upgrade or Manual Maintenance Upgrade
Minor Upgrade	HDP 2.0	HDP 2.3	Ambari 2.1 does **not** support directly upgrading from H 2.0 to HDP 2.3. In order to upgrade from HDP 2.0, **you first upgrade to HDP 2.2 using either Ambari 1.7 or 2.** Once completed, upgrade your current Ambari to Amb 2.1. Then, leverage Ambari 2.1 to complete the upgrad HDP 2.2 to HDP 2.3.

* Before performing a Minor Upgrade, you **must** upgrade your Ambari **first**. See Preparing to Upgrade Ambari and HDP for more information.

4.1. Rolling Upgrade HDP 2.2 to 2.3

Use this procedure to perform a rolling upgrade from HDP 2.2 to HDP 2.3. You are **strongly encouraged** to read completely through this entire document before starting the upgrade process, to that you understand the interdependencies and order of the steps. It is **highly recommended** you validate these steps in a test environment to adjust + account for any special configurations for your cluster.

The high-level process for performing an HDP 2.2 -> 2.3 Rolling Upgrade is as follows:



Before upgrading to HDP 2.3, **you must first upgrade to Ambari 2.1**. Make sure Ambari is upgraded and the cluster is healthy and operating normally prior to attempting to upgrade from HDP 2.2 to HDP 2.3. See Preparing to Upgrade Ambari and HDP for more information on performing the Ambari upgrade. Once you are running Ambari 2.1, follow these steps to upgrade the Stack from HDP 2.2 to HDP 2.3.

- Prerequisites
- Preparing to Upgrade
- Registering a New Version
- Installing a New Version
- Performing the Upgrade

Note

The HDP packages for a complete installation of HDP 2.3 will occupy about 2.5 GB of disk space.

Note

If you do not meet the upgrade prerequisite requirements listed above, you can consider a Manual Upgrade HDP 2.2 to 2.3 of the cluster.

Note

You can use this procedure to upgrade to any HDP 2.3 maintenance release (i.e. third-or-fourth digit release). The instructions in this document refer to HDP 2.3.x.y as a placeholder version. To use an HDP 2.3.x.y maintenance release, be sure to replace 2.3.x.y in the following instructions with the appropriate maintenance version, such as 2.3.0.0 for the HDP 2.3 GA release, or 2.3.1.0 for an HDP 2.3 maintenance release. Refer to the HDP documentation for the information about the latest HDP 2.3 maintenance releases.

4.1.1. Prerequisites

To perform a rolling upgrade, your cluster must meet the following prerequisites. If you do not meet these upgrade prerequisites, you can consider a Manual Upgrade HDP 2.2 to 2.3 of the cluster.

- General
- HDFS
- YARN
- MapReduce 2
- Tez
- Hive
- Oozie

General

Requirement	Description
Current HDP Version	Must be running HDP 2.2 or later to perform a rolling upgrade. The rolling upgrade capability is not available for clusters running HDP 2.0 or 2.1.
Target HDP Version	All hosts must have the target version installed. See the Register Version and Install Version sections for more information.
Ambari Agent Heartbeats	All Ambari Agents must be heartbeating to Ambari Server. Any hosts that are not heartbeating must be in Maintenance Mode.
Host Maintenance Mode	Any hosts in Maintenance Mode must not be hosting any Service master components.
Service Maintenance Mode	No Services can be in Maintenance Mode.
Services Started	All Services must be started and the Service Check must pass.

HDFS

Requirement	Description
NameNode HA	NameNode HA must be enabled and working properly. See the Ambari User's Guide for more information, Configuring NameNode High Availability.
NameNode Truncate	HDP 2.2.6 introduced the NameNode Truncate option. Truncate must not be enabled.
Client Retry	HDFS client retry should be dfs.client.retry.policy.enabled

YARN

Requirement	Description
ResourceManager HA	YARN ResourceManager HA should be enabled to prevent a disruption in service during the upgrade. See the Ambari User's Guide for more information on Configuring ResourceManager High Availability.
Start Preserving Recovery	YARN start preserving recovery should be enabled. Check the **Services > YARN > Configs** property `yarn.timeline-service.recovery.enabled`.
Work Preserving Restart	YARN Work Preserving Restart must be configured. Check the **Services > YARN > Configs** property `yarn.resourcemanager.work-preserving-recovery.enabled` .

MapReduce 2

Requirement	Description
MapReduce Distributed Cache	MapReduce should reference Hadoop libraries from the distributed cache in HDFS. Refer to the YARN Resource Management guide for more information.
State Preserving Recovery	JobHistory state preserving recovery should be enabled.
Wire Encryption	If encrypted shuffle has been enabled, the `ssl-client.xml` file must be copied to `/etc/hadoop/conf/secure` on each node in the cluster.

Tez

Requirement	Description
Tez Distributed Cache	Tez should reference Hadoop libraries from the distributed cache in HDFS.

Hive

Requirement	Description
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.
Hive Dynamic Service Discovery	HiveServer2 dynamic service discovery is recommended for Rolling Upgrade.
HiveServer2 Port	During the upgrade, Ambari will switch the HiveServer2 port from 10000 to 10010 (or 10011 if using HTTP transport mode).
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.

Oozie

Requirement	Description
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.

4.1.2. Preparing to Upgrade

Review Preparing to Upgrade Ambari and HDP. It is also **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

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:

     ```
     hadoop 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
   ```

Proceed to Registering a New Version.

4.1.3. Registering a New Version

Register HDP 2.3

1. Log in to Ambari.

2. Browse to `Admin > Stack and Versions`.

3. Click on the `Versions` tab. You will see the version currently running. The full version is dependent on the HDP version you are actually running. For example, if you are currently running a maintenance release of HDP 2.2, you would see something like **HDP-2.2.0.0-2041**.

4. Click `Manage Versions`.

5. Proceed to register a new version by clicking `+ Register Version`.

6. Select the HDP 2.3 Stack and enter a two-digit version number. For example, enter **0.0**(which makes the version name **HDP-2.3.0.0**).

7. Select one or more OS families and enter the repository Base URLs.

   To copy a Base URL for a specific OS family, see the list of available HDP Repositories.

8. Click `Save`.

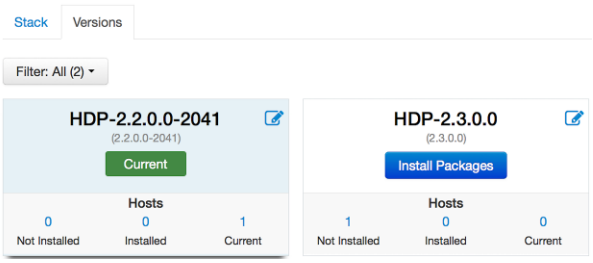
9. You can click `Install on...`, or you can browse back to Admin > Stack and Versions. You will see the version currently running **HDP-2.2.x.y** and the version you just registered **HDP-2.3.0.0**.

   Proceed to Installing a New Version.

4.1.4. Installing a New Version

Install HDP 2.3 on All Hosts

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



4. Click `Install Packages` and click OK to confirm. The Install version operation will start and the new version will be installed on all hosts. You can browse to Hosts and to each host > Versions tab to see the new version is installed.

   Proceed to Performing the Upgrade.

4.1.5. Performing the Upgrade

Perform the Upgrade to HDP 2.3

1. Log in to Ambari.
2. Browse to `Admin > Stack and Versions`.
3. Click on the `Versions` tab.
4. Click `Perform Upgrade.`
5. Ambari will check that your cluster meets the prerequisites. A dialog will be presented with the results:

   a. If any prerequisites are not met but are required, the result will be shown with an error. You will not be allowed to proceed with the upgrade. Make the appropriate corrections and return to Perform Upgrade again.

   b. If any prerequisites are not met but are optional, the result will be shown with a warning. You will be allowed to Proceed with the upgrade.

c. A list of configuration changes (if any) will be displayed.

6. Once the prerequisite checks are complete, the upgrade will start. The time it takes to perform the upgrade dependents on many factors. As part of the upgrade process, each component in the cluster is restarted in a serial fashion so the stop/start time is a big contributor to the overall time.

7. The upgrade process involves a set of stages. This table lists the high-level stages and if the process requires any action by you during normal operation. Note: if any stage fails, the upgrade will halt and prompt for action.

Stage	Description	Action Required
Prepare Upgrade	This stage prepares HDFS and HBase for upgrade and reminds you to perform backups of your database for Hive, Oozie and Ranger.	You must acknowledge the prompt for database backups.
ZooKeeper	All ZooKeeper servers are upgraded and restarted.	None
Ranger	Ranger Admin and UserSync servers are upgraded and restarted.	None
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
Service Checks	All Service Checks are performed against the cluster.	If any service check fails, you will be prompted to Ignore and Continue, Downgrade or Rerowy.
Core Slaves	This stage upgrades the slave components of core services. This includes DataNodes (HDFS), RegionServers (HBase) and NodeManagers (YARN). This is down in two batches: 20% of the slaves first, then the remaining slaves.	After the first 20% batch is complete, you will be prompted to Verify the cluster is operating correctly.
Service Checks	All Service Checks are performed against the cluster.	If any service check fails, you will be prompted to Ignore and Continue, Downgrade or Retry.
Hive	This stage upgrades the Hive Metastore, HiveServer2 and WebHCat components.	Ambari will switch the HiveServer2 port from 10000 to 10010 (or, 10011 if using HTTP transport mode). You will be prompted to confirm prior to the switch.
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
Clients	All remaining clients are upgraded.	None
Service Checks	All Service Checks are performed against the cluster.	If any service check fails, you will be prompted to Ignore and Continue, Downgrade or Rerowy.
Kafka	The Kafka Brokers are upgraded.	None
Knox	The Knox Gateways are upgraded.	None
Storm	The Nimbus, REST API, DRPC Server and UI Server components are upgraded, along with the Supervisors.	None
Slider	The Slider clients are upgraded.	None

Stage	Description	Action Required
Flume	The Flume agents are upgrade.	None
Finalize	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, Finalize Later or Downgrade.

8. Once the upgrade is complete, you have an option 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: the reverse of the upgrade process stages). **Once finalized, you cannot downgrade back to the previous version**.

Note

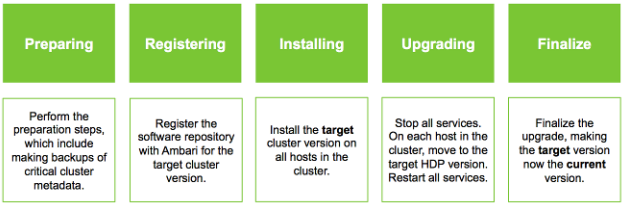
If you choose to finalize later, both versions will be listed on the Stack and Versions tab with the starting version displaying a Current. It is not until you finalize that Ambari makes the target version the current version. Also, until you finalize, you will not be able to perform operational changes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

9. Click **Finalize** and the upgrade process is complete.

4.2. Manual Upgrade HDP 2.2 to 2.3

Use this procedure for upgrading from HDP 2.2 to HDP 2.3. **This procedure will require cluster downtime**. You are **strongly encouraged** to read completely through this entire document before starting the upgrade process, to that you understand the interdependencies and order of the steps. It is **highly recommended** you validate these steps in a test environment to adjust + account for any special configurations for your cluster.

The high-level process for performing an HDP 2.2 -> 2.3 Manual Upgrade is as follows:



Prior to upgrading to HDP 2.3, **you must first upgrade to Ambari 2.1**. Make sure **Ambari is upgraded and the cluster is healthy and operating normally** prior to attempting to upgrade from HDP 2.2 to HDP 2.3. See Upgrading Ambari for more information on performing the Ambari upgrade. Once you are running Ambari 2.1, follow these steps to upgrade the Stack from HDP 2.2 to HDP 2.3.

- Registering a New Version
- Installing a New Version
- Preparing to Upgrade

- Performing the Upgrade

Note

This is an alternative to using the Rolling Upgrade feature of Ambari when using the HDP 2.2 Stack.

Note

The HDP packages for a complete installation of HDP 2.3 will occupy about 2.5 GB of disk space.

Note

You can use this procedure to upgrade to any HDP 2.3 maintenance release (i.e. third-or-fourth digit release). The instructions in this document refer to HDP 2.3.x.y as a placeholder version. To use an HDP 2.3.x.y maintenance release, be sure to replace 2.3.x.y in the following instructions with the appropriate maintenance version, such as 2.3.0.0 for the HDP 2.3 GA release, or 2.3.1.0 for an HDP 2.3 maintenance release. Refer to the HDP documentation for the information about the latest HDP 2.3 maintenance releases.

4.2.1. Registering a New Version

Register the HDP 2.3 Version

1. Log in to Ambari.
2. Browse to `Admin > Stack and Versions`.
3. Click on the `Versions` tab. You will see the version currently running. The full version is dependent on the HDP version you are actually running. For example, if you are currently running a maintenance release of HDP 2.2, you would see something like **HDP-2.2.0.0-2041** .
4. Click `Manage Versions`.
5. Proceed to register a new version by clicking `+ Register Version`.
6. Enter a two-digit version number. For example, enter **0.0** (which makes the version name **HDP-2.3.0.0**).

7. Select one or more OS families and enter the repository Base URLs for that OS.

   To copy a Base URL for a specific OS family, see the list of available HDP Repositories.

8. Click `Save`.

9. Click `Go to Dashboard` and browse back to `Admin > Stack and Versions > Versions`. You will see the current running version **HDP-2.2.0.0-2041** and the version you just registered **HDP-2.3.0.0**.

   Proceed to Installing a New Version.

4.2.2. Installing a New Version

Install HDP 2.3 on All Hosts

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

4. Click `Install Packages` and click OK to confirm.
5. The Install version operation will start and the new version will be installed on all hosts.
6. After installation is complete, you will see the Perform Upgrade button. **Do not click the Perform Upgrade button**. You will be performining this upgrade manually instead.

   Proceed to Preparing to Upgrade.

4.2.3. Preparing to Upgrade

After reviewing Preparing to Upgrade Ambari and HDP, perform the following tasks to prepare to upgrade the HDP Stack.

- Plan for JDK Changes
- Record Component Layout
- Stop Running Jobs
- Disable Kerberos Security
- Stop Cluster and Checkpoint HDFS
- Perform Backups

4.2.3.1. Plan for JDK Changes

Ambari 2.1 does not support JDK 1.6. If you are currently running JDK 1.6, you must upgrade to JDK 1.7 prior to upgrading Ambari and prior to performing this HDP 2.3 Stack

upgrade. If you plan to upgrade your existing JDK to JDK 1.8, do so only after upgrading Ambari and after performing this HDP 2.3 Stack upgrade. Do not move to JDK 1.8 until your cluster is safely on HDP 2.3.

4.2.3.2. Record Component Layout

How you have chosen to install the HDP component across the hosts is unique to your cluster. Before upgrading HDP, use the Ambari Web Hosts and Services tabs to determine the components installed on each host. For more information about using Ambari Web, see the Ambari User's Guide sections Working with Hosts and Viewing Components on a Host. Be sure to record the location of all components, including master components (such as NameNode, ResourceManager, etc) and slave components (such as DataNodes, NodeManagers, etc).

Another option is to use the Ambari REST API and export the list of hosts and their associated components in json format.

`http://`**`ambari.server`**`:8080/api/v1/clusters/`**`cluster.name`**`/hosts?fields=host_components`

where **ambari.server** is the Ambari Server host and **cluster.name** is your cluster name.

4.2.3.3. Stop Running Jobs

Make sure to finish all current jobs running on the system before upgrading the cluster. Client libraries will change during the HDP upgrade. Any jobs remaining active that use the older version libraries will probably fail during the upgrade.

4.2.3.4. Disable Kerberos Security

If your cluster has Kerberos Security enabled, disable Kerberos before performing the Stack upgrade. In Ambari Web, browse to `> Admin > Kerberos`, and click `Disable Kerberos`. You can re-enable Kerberos after performing the HDP upgrade.

4.2.3.5. Stop Cluster and Checkpoint HDFS

1. Using Ambari Web, browse to each Service **except HDFS and ZooKeeper** and perform Stop. Leave HDFS and ZooKeeper running for now.

2. 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.`

3. 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.

4. 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:

  ```
  hadoop 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.

5. 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 -f`option to specify which NameNode to connect. For example, to force a checkpoint in NameNode2: `hdfs dfsadmin -fs hdfs://namenode2-hostname:namenode2-port -saveNamespace`

6. 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.

7. 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.

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
```

9. Using Ambari Web, **stop HDFS service and stop ZooKeeper service**.

10. Using Ambari Web, review each service and **make sure that all services in the cluster are completely stopped**.

4.2.3.6. Perform Backups

It is also **strongly recommended** that you perform backups and checkpoints before beginning upgrade. Checkpoint user metadata by backing-up Hive Metastore and Oozie databases. This step makes rollback and restoration of the original cluster state possible, if necessary.

- If you have the Hive component installed, back up the Hive metastore database.

  These instructions are provided for your convenience. For the latest back up instructions, see your database documentation.

  Hive Metastore Database Backup and Restore

Database Type	Backup	Restore
MySQL	mysqldump $dbname > $outputfilename.sqlsbr For example: mysqldump hive > /tmp/mydir/backup_hive.sql	mysql $dbname < $inputfilename.sqlsbr For example: mysql hive < /tmp/mydir/backup_hive.sql
Postgres	sudo -u $username pg_dump $databasename > $outputfilename.sql sbr For example: sudo -u postgres pg_dump hive > /tmp/mydir/backup_hive.sql	sudo -u $username psql $databasename < $inputfilename.sqlsbr For example: sudo -u postgres psql hive < /tmp/mydir/backup_hive.sql
Oracle	Connect to the Oracle database using sqlplus Export the database: exp username/password@database full=yes file=output_file.dmp mysql $dbname < $inputfilename.sqlsbr For example: mysql hive < /tmp/mydir/backup_hive.sql	 Import the database: imp username/password@database file=input_file.dmp

- If you have the Oozie component installed, back up the Oozie metastore database.

  These instructions are provided for your convenience. For the latest back up instructions, see your database documentation.

  Oozie Database Backup and Restore

Database Type	Backup	Restore
MySQL	mysqldump $dbname > $outputfilename.sql For example:	mysql $dbname < $inputfilename.sql For example:

Database Type	Backup	Restore
	mysqldump oozie > /tmp/mydir/backup_oozie.sql	mysql oozie < /tmp/mydir/backup_oozie.sql
Postgres	sudo -u $username pg_dump $databasename > $outputfilename.sql For example: sudo -u postgres pg_dump oozie > /tmp/mydir/backup_oozie.sql	sudo -u $username psql $databasename < $inputfilename.sql For example: sudo -u postgres psql oozie < /tmp/mydir/backup_oozie.sql
Oracle	Connect to the Oracle database using sqlplus Export the database: exp username/password@database full=yes file=output_file.dmp mysql $dbname < $inputfilename.sqlsbr For example:mysql oozie < /tmp/mydir/backup_oozie.sql	Import the database: imp username/password@database file=input_file.dmp

- If you have the Ranger component installed, back up the Ranger and Ranger_audit databases.

  These instructions are provided for your convenience. For the latest back up instructions, see your database documentation.

  Ranger Database Backup and Restore

Database Type	Backup	Restore
MySQL	mysqldump $dbname > $outputfilename.sql For example: mysqldump ranger > /tmp/mydir/backup_ranger.sql	mysql $dbname < $inputfilename.sql For example: mysql ranger < /tmp/mydir/backup_ranger.sql
Oracle	Connect to the Oracle database using sqlplus Export the database: exp username/password@database full=yes file=output_file.dmp mysql $dbname < $inputfilename.sqlsbr For example:mysql ranger < /tmp/mydir/backup_ranger.sql	Import the database: imp username/password@database file=input_file.dmp

- Verify that edit logs in `${dfs.namenode.name.dir}/current/edits*` are empty.

  1. Run: `hdfs oev -i ${dfs.namenode.name.dir}/current/edits_inprogress_* -o edits.out`

  2. Verify edits.out file. It should only have OP_START_LOG_SEGMENT transaction. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<EDITS>
<EDITS_VERSION>-56</EDITS_VERSION>
<RECORD>
<OPCODE>OP_START_LOG_SEGMENT</OPCODE>
<DATA>
```

```
<TXID>5749</TXID>
</DATA>
</RECORD>
```

3. If edits.out has transactions other than OP_START_LOG_SEGMENT, run the following steps and then verify edit logs are empty.

   - Start the existing version NameNode.
   - Ensure there is a new FS image file.
   - Shut the NameNode down:

     ```
     hdfs dfsadmin –saveNamespace
     ```

- Rename or delete any paths that are reserved in the new version of HDFS.

  When upgrading to a new version of HDFS, it is necessary to rename or delete any paths that are reserved in the new version of HDFS. If the NameNode encounters a reserved path during upgrade, it will print an error such as the following:

  ```
  /.reserved is a reserved path and .snapshot is a reserved path
  component in this version of HDFS.

  Please rollback and delete or rename this path, or upgrade with the
  -renameReserved key-value pairs option to automatically rename these
  paths during upgrade.
  ```

  Specifying `-upgrade -renameReserved` optional key-value pairs causes the NameNode to automatically rename any reserved paths found during startup.

  For example, to rename all paths named `.snapshot` to `.my-snapshot` and change paths named `.reserved` to `.my-reserved`, specify `-upgrade -renameReserved .snapshot=.my-snapshot,.reserved=.my-reserved`.

  If no key-value pairs are specified with `-renameReserved`, the NameNode will then suffix reserved paths with:

  `.<LAYOUT-VERSION>.UPGRADE_RENAMED`

  For example: `.snapshot.-51.UPGRADE_RENAMED`.

  Note

  We recommend that you perform a `-saveNamespace` before renaming paths (running `-saveNamespace` appears in a previous step in this procedure). This is because a data inconsistency can result if an edit log operation refers to the destination of an automatically renamed file.

  Also note that running `-renameReserved` will rename all applicable existing files in the cluster. This may impact cluster applications.

- Backup Hue. If you have the Hue component installed, back up Hue. If you are using the embedded SQLite database, you must perform a backup of the database before you upgrade Hue to prevent data loss.

To make a backup copy of the database, stop Hue and simply do a "dump" and redirect the results to a file.

```
/etc/init.d/hue stop

su - hue

mkdir ~/hue_backup

cd /var/lib/hue

sqlite3 desktop.db .dump > ~/hue_backup/desktop.bak
```

For other databases, follow your vendor-specific instructions to create a backup.

- Backup the Hue Configuration.

```
cp -RL /etc/hue/conf ~/hue_backup
```

Proceed to Performing the Upgrade.

4.2.4. Performing the Upgrade

Perform the following steps to perform the upgrade:

- Run HDP Select
- Update Service Configurations
- Upgrade ZooKeeper
- Upgrade HDFS
- Upgrade YARN and MapReduce2
- Upgrade HBase
- Upgrade Tez
- Upgrade Hive and WebHCat
- Upgrade Pig
- Upgrade Sqoop
- Upgrade Flume
- Upgrade Oozie
- Upgrade Falcon
- Upgrade Knox
- Upgrade Spark
- Upgrade Slider
- Upgrade Kafka

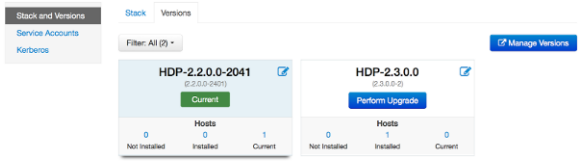
- Upgrade Ranger
- Upgrade Storm
- Finalize Upgrade
- Set Current HDP Version

Important

Be sure you have stopped all services from Ambari. Using Ambari Web, review each service and **make sure that all services in the cluster are completely stopped** before proceeding.

4.2.4.1. Run HDP Select

1. Log in to Ambari.
2. Browse to `Admin > Stack and Versions`.
3. Click on the `Versions` tab.
4. Under the newly registered and installed version **HDP-2.3.0.0**, is the actual software repository version in parenthesis (Ambari determined this repository version during the install). For example, in the picture below the display name is **HDP-2.3.0.0** and the repository version **2.3.0.0-2**. Record this repository version. This is the value of 2.3.x.y-z that you will use later in the upgrade process. **Do not click the Perform Upgrade button**. You will be performing the upgrade manually instead.



5. Go to the command line on each host and move the current HDP version to the newly installed version using the hdp-select utility and repository version number (obtained in Step 4).

```
hdp-select set all {repository-version}
```

For example:

```
hdp-select set all 2.3.0.0-2
```

4.2.4.2. Update Service Configurations

1. Create an "Upgrade Folder". For example, `/work/upgrade_hdp_2`, on the Ambari Server host.

```
mkdir -p /work/upgrade_hdp_2
```

```
cd /work/upgrade_hdp_2
```

2. Download and copy upgrade helper to the Upgrade Folder. Be sure to set the helper to have execute permissions. The helper is available for download here:

```
curl -O https://raw.githubusercontent.com/apache/ambari/
branch-2.1/ambari-server/src/main/python/upgradeHelper.py

chmod +x upgradeHelper.py
```

3. Download and copy upgrade catalog to the Upgrade Folder. The catalog is available for download here:

```
curl -O https://raw.githubusercontent.com/apache/ambari/
branch-2.1/ambari-server/src/main/resources/upgrade/catalog/
UpgradeCatalog_2.2_to_2.3.json
```

4. Run the helper:

```
python upgradeHelper.py --hostname $HOSTNAME --user $USERNAME
 --password $PASSWORD --clustername $CLUSTERNAME
 --fromStack $FROMSTACK --toStack $TOSTACK
 --upgradeCatalog UpgradeCatalog_2.2_to_2.3.json update-configs
```

Variable	Value
$HOSTNAME	Ambari Server hostname. This should be the FQDN for the host running the Ambari Server.
$USERNAME	Ambari Admin user.
$PASSWORD	Password for the user.
$CLUSTERNAME	Name of the cluster. This is the name you provided when you installed the cluster with Ambari. Login to Ambari and the name can be found in the upper-left of the Ambari Web screen. This is case-sensitive.
$FROMSTACK	The "from" stack. For example: 2.2
$TOSTACK	The "to" stack. For example: 2.3

5. Download and copy the "Step 2" upgrade catalog to the Upgrade Folder. **You will not use this catalog now**. You will use this catalog in the Set Current HDP Version section at the end of the upgrade. The "Step 2" catalog is available for download here:

```
curl -O https://raw.githubusercontent.com/apache/ambari/
branch-2.1/ambari-server/src/main/resources/upgrade/catalog/
UpgradeCatalog_2.2_to_2.3_step2.json
```

Note

By default, the upgrade helper will connect to the Ambari Server using port 8080 over http. If your Ambari Server is configured for a different port and/or is configured for ssl/https, you can specify the following options when running the helper:

Option	Description	Example
–port	Specifies the port to connect to Ambari Server with.	–port 8181
–https	Specifies to connect to Ambari Server via https. Defaults to http if option not set.	–https

4.2.4.3. Upgrade ZooKeeper

1. From Ambari Web, browse to `Services > ZooKeeper`.

2. Start ZooKeeper by selecting `Service Actions > Start`. This will push the new ZooKeeper configurations and start the ZooKeeper servers.

3. Save the old ZooKeeper configuration and add symlink from /etc/zookeeper/conf:

   ```
   mv /etc/zookeeper/conf /etc/zookeeper/conf.saved

   ln -s /usr/hdp/current/zookeeper-client/conf /etc/zookeeper/conf

   ls -la /etc/zookeeper
   ```

   ```
   total 4
   drwxr-xr-x 3 root root 4096 Jun 19 21:51 2.3.0.0-2323
   lrwxrwxrwx 1 root root   35 Jun 19 21:54 conf -> /usr/hdp/current/zookeeper-
   client/conf
   drwxr-xr-x 2 root root 4096 Jun 14 00:11 conf.saved
   ```

4.2.4.4. Upgrade HDFS

1. For HDP 2.2, the configuration files were stored in `/etc/hadoop/conf`. Starting with HDP 2.3, the configuration files are stored in `/etc/hadoop` but in a sub-directory specific to the HDP version being used. To perform the HDFS upgrade, we need to copy the existing configuration files into place **on every NameNode and DataNode**:

   ```
   cp /etc/hadoop/conf/* /etc/hadoop/2.3.x.y-z/0/
   ```

   After copying configurations to the 2.3 configuration location, save the old HDFS configuration and add symlink from /etc/hadoop/conf:

   ```
   mv /etc/hadoop/conf /etc/hadoop/conf.saved

   ln -s /usr/hdp/current/hadoop-client/conf /etc/hadoop/conf

   ls -la /etc/hadoop
   ```

   ```
   total 4
   drwxr-xr-x 3 root root 4096 Jun 19 21:51 2.3.0.0-2323
   lrwxrwxrwx 1 root root   35 Jun 19 21:54 conf -> /usr/hdp/current/hadoop-
   client/conf
   drwxr-xr-x 2 root root 4096 Jun 14 00:11 conf.saved
   ```

2. If you are upgrading from an HA NameNode configuration, start all JournalNodes. At each JournalNode host, run the following command:

   ```
   su -l <HDFS_USER> -c "/usr/hdp/current/hadoop-client/sbin/
   hadoop-daemon.sh start journalnode"
   ```

   where <HDFS_USER> is the HDFS Service user. For example, hdfs.

   Important

   All JournalNodes must be running when performing the upgrade, rollback, or finalization operations. If any JournalNodes are down when running any such operation, the operation will fail.

3. If you are upgrading from an HA NameNode configuration, start the ZK Failover Controllers.

   ```
   su -l <HDFS_USER> -c "/usr/hdp/current/hadoop-client/sbin/
   hadoop-daemon.sh start zkfc"
   ```

   where <HDFS_USER> is the HDFS Service user. For example, hdfs.

4. Because the file system version has now changed, you must start the NameNode manually. On the active NameNode host, as the HDFS user,

   ```
   su -l <HDFS_USER> -c "/usr/hdp/current/hadoop-client/sbin/hadoop-
   daemon.sh start namenode -upgrade"
   ```

   where <HDFS_USER> is the HDFS Service user. For example, hdfs.

   Note

   In a large system, this can take a long time to complete. Run this command with the `-upgrade` option only once. After you have completed this step, you can bring up the NameNode using this command without including the `-upgrade` option.

   To check if the Upgrade is progressing, check that the `${dfs.namenode.name.dir}/previous` directory has been created. The `${dfs.namenode.name.dir}/previous` directory contains a snapshot of the data before upgrade.

   Note

   In a NameNode HA configuration, this NameNode does not enter the standby state as usual. Rather, this NameNode immediately enters the active state, upgrades its local storage directories, and upgrades the shared edit log. At this point, the standby NameNode in the HA pair is still down, and not synchronized with the upgraded, active NameNode.

   To re-establish HA, you must synchronize the active and standby NameNodes. To do so, bootstrap the standby NameNode by running the NameNode with the '-bootstrapStandby' flag. Do NOT start the standby NameNode with the '-upgrade' flag.

   At the Standby NameNode,

   ```
   su -l <HDFS_USER> -c "hdfs namenode -bootstrapStandby -
   force"
   ```
   where <HDFS_USER> is the HDFS Service user. For example, hdfs.

   The bootstrapStandby command downloads the most recent fsimage from the active NameNode into the `<dfs.name.dir>` directory on the standby NameNode. Optionally, you can access that directory to make sure the fsimage has been successfully downloaded. After verifying, start the ZKFailoverController, then start the standby NameNode using `Ambari Web > Hosts > Components`.

5. Verify that the NameNode is up and running:

```
ps -ef | grep -i NameNode
```

6. Start all DataNodes.

   At each DataNode, as the HDFS user,

   ```
   su -l <HDFS_USER> -c "/usr/hdp/current/hadoop-client/sbin/
   hadoop-daemon.sh start datanode"
   ```

   where <HDFS_USER> is the HDFS Service user. For example, hdfs.

   The NameNode sends an upgrade command to DataNodes after receiving block reports.

7. Verify that the DataNode process is up and running:

   ```
   ps -ef | grep DataNode
   ```

8. Restart HDFS. Restarting HDFS will push out the upgraded configurations to all HDFS services.

   a. Open the Ambari Web. If the browser in which Ambari is running has been open throughout the process, clear the browser cache, then refresh the browser.

   b. Browse to Services > HDFS, and from the Service Actions menu, select Restart All.

   c. If you are running an HA NameNode configuration, use the following procedure to restart NameNodes.

      i. Browse to `Services > HDFS`. The Summary section of the page shows which host is the active NameNode.

      ii. Hover over the Active NameNode and write down/remember the hostname of the host. You will need this hostname later.

      iii. From the Service Actions menu, select Stop. This stops all of the HDFS Components, including both NameNodes.

      iv. Browse to Hosts and select the host that was running the Active NameNode (as noted in the previous step). Using the Actions menu next to the NameNode component, select Start. This causes the original Active NameNode to re-assume it's role as the Active NameNode .

      v. Browse to `Services > HDFS` and from the Service Actions menu, select Restart All.

   d. After HDFS has started, run the service check. Browse to `Services>HDFS` and from the Service Actions menu, select, `Run Service Check`.

9. After the DataNodes are started, HDFS exits SafeMode. To monitor the status, run the following command, on any DataNode:

   ```
   su -l <HDFS_USER> -c "hdfs dfsadmin -safemode get"
   ```

   where <HDFS_USER> is the HDFS Service user. For example, hdfs.

When HDFS exits SafeMode, the following message displays:

```
Safe mode is OFF
```

Note

In general, it takes 5-10 minutes to get out of safemode. For thousands of nodes with millions of data blocks, getting out of safemode can take up to 45 minutes.

10. Make sure that the HDFS upgrade was successful. Optionally, repeat step 4 in Checkpoint HDFS to create new versions of the logs and reports, substituting "-new" for "-old" in the file names as necessary.

- Compare the old and new versions of the following log files:
  - `dfs-old-fsck-1.log` versus `dfs-new-fsck-1.log`.

    The files should be identical unless the hadoop fsck reporting format has changed in the new version.
  - `dfs-old-lsr-1.log` versus `dfs-new-lsr-1.log`.

    The files should be identical unless the format of hadoop fs -lsr reporting or the data structures have changed in the new version.
  - `dfs-old-report-1.log` versus `fs-new-report-1.log`

    Make sure that all DataNodes in the cluster before upgrading are up and running.

11. From the NameNode WebUI , determine if all DataNodes are up and running.

```
http://<namenode>:<namenodeport>
```

12. If you are on a highly available HDFS cluster, go to the Standby NameNode web UI to see if all DataNodes are up and running:

```
http://<standbynamenode>:<namenodeport>
```

13. If you are not on a highly available HDFS cluster, go to the SecondaryNameNode web UI to see if it the secondary node is up and running:

```
http://<secondarynamenode>:<secondarynamenodeport>
```

4.2.4.5. Upgrade YARN and MapReduce2

1. Prepare MR2 and YARN for work. Execute HDFS commands on any host.

- Create mapreduce dir in hdfs.

```
su -l <HDFS_USER> -c "hdfs dfs -mkdir -p /hdp/apps/2.3.x.y-z/
mapreduce/"
```

- Copy new mapreduce.tar.gz to HDFS mapreduce dir.

```
su -l <HDFS_USER> -c "hdfs dfs -put /usr/hdp/current/hadoop-
client/mapreduce.tar.gz /hdp/apps/2.3.x.y-z/mapreduce/."
```

- Copy new hadoop-streaming.jar to HDFS mapreduce dir.

  ```
  su -l <HDFS_USER> -c "hdfs dfs -put /usr/hdp/current/hadoop-
  mapreduce-client/hadoop-streaming.jar /hdp/apps/2.3.x.y-z/
  mapreduce/."
  ```

- Grant permissions for created mapreduce dir in hdfs.

  ```
  su -l <HDFS_USER> -c "hdfs dfs -chown -R <HDFS_USER>:<HADOOP_GROUP> /hdp"
  su -l <HDFS_USER> -c "hdfs dfs -chmod -R 555 /hdp/apps/2.3.x.y-z/
  mapreduce"
  su -l <HDFS_USER> -c "hdfs dfs -chmod -R 444 /hdp/apps/2.3.x.y-z/
  mapreduce/mapreduce.tar.gz"
  su -l <HDFS_USER> -c "hdfs dfs -chmod -R 444 /hdp/apps/2.3.x.y-z/
  mapreduce/hadoop-streaming.jar"
  ```

- Update YARN Configuration Properties for HDP 2.3

  From Ambari Web, browse to > `Services` > YARN > `Configs`. Filter for the following properties, and add these properties if they do not already exist:

Name	Value
hadoop.registry.zk.quorum	The comma-separated list of ZooKeeper hosts defining the ZooKeeper quorum. For example: ZooKeeperHost1:2181, ZooKeeperHost2:2181
yarn.resourcemanager.zk-address	The comma-separated list of ZooKeeper hosts defining the ZooKeeper quorum. For example: ZooKeeperHost1:2181, ZooKeeperHost2:2181

2. Manually clear the ResourceManager state store.

   ```
   su -l <YARN_USER> -c "yarn resourcemanager -format-state-store"
   ```

   where <YARN_USER> is the YARN Service user. For example, yarn.

3. Using Ambari Web, browse to `Services` > YARN and select Start from the Service Actions menu. Repeat for MapReduce2.

4. Run the smoke checks. Using Ambari Web, browse to Services > YARN and select Run Service Check from the Service Actions menu. Confirm the check passes. Repeat for MapReduce2.

4.2.4.6. Upgrade HBase

1. Save the old HBase configuration and add symlink from /etc/hbase/conf:

   ```
   mv /etc/hbase/conf /etc/hbase/conf.saved
   ```

```
ln -s /usr/hdp/current/hbase-client/conf /etc/hbase/conf

ls -la /etc/hbase
```

```
total 4
drwxr-xr-x 3 root root 4096 Jun 19 21:51 2.3.0.0-2323
lrwxrwxrwx 1 root root   35 Jun 19 21:54 conf -> /usr/hdp/current/hbase-
client/conf
drwxr-xr-x 2 root root 4096 Jun 14 00:11 conf.saved
```

2. From Ambari Web, browse to `Services` > HBase.

3. Start HBase by selecting `Service Actions > Start`.

4. After HBase has started, select Run Service Check from the Service Actions menu. Confirm the Service Check passes.

4.2.4.7. Upgrade Tez

1. Put the Tez tarball libraries in HDFS. Execute on any host that has the Tez client.

```
su -l <HDFS_USER> -c "hdfs dfs -mkdir -p /hdp/apps/2.3.x.y-z/tez/"
su -l <HDFS_USER> -c "hdfs dfs -put -f /usr/hdp/current/tez-client/lib/tez.
tar.gz /hdp/apps/2.3.x.y-z/tez/."
su -l <HDFS_USER> -c "hdfs dfs -chown -R <HDFS_USER>:<HADOOP_GROUP> /hdp"
su -l <HDFS_USER> -c "hdfs dfs -chmod -R 555 /hdp/apps/2.3.x.y-z/tez"
su -l <HDFS_USER> -c "hdfs dfs -chmod -R 444 /hdp/apps/2.3.x.y-z/tez/tez.
tar.gz"
```

   where <HDFS_USER> is the HDFS Service user. For example, hdfs.

2. Save the old Tez configuration and add symlink from /etc/tez/conf:

```
mv /etc/tez/conf /etc/tez/conf.saved

ln -s /usr/hdp/current/tez-client/conf /etc/tez/conf

ls -la /etc/tez
```

```
total 4
drwxr-xr-x 3 root root 4096 Jun 19 21:51 2.3.0.0-2323
lrwxrwxrwx 1 root root   35 Jun 19 21:54 conf -> /usr/hdp/current/tez-
client/conf
drwxr-xr-x 2 root root 4096 Jun 14 00:11 conf.saved
```

3. From Ambari Web, browse to Servces > Tez and select Run Service Check from the Service Actions menu. Confirm the Service Check passes.

4.2.4.8. Upgrade Hive and WebHCat

1. Upgrade the Hive Metastore database schema. Restart the Hive Metastore Database (MySQL, Oracle, or Postgres) **server**, **not the Hive Metastore process**.

   On each HiveServer2 host, run:

```
su -l <HIVE_USER> -c "export HIVE_CONF_DIR=/etc/hive/
conf.server/" "/usr/hdp/current/hive-metastore/bin/schematool -
upgradeSchema -dbType <DATABASE_TYPE>"
```

where <DATABASE_TYPE> is mysql, oracle or postgres and <HIVE_USER> is the Hive Service user. For example, hive.

Note

If you are using Postgres 8 and Postgres 9, you should reset the Hive Metastore database owner to

```
<HIVE_USER>: psql -U <POSTGRES_USER> -c ALTER DATABASE
<HIVE-METASTORE-DB-NAME> OWNER TO <HIVE_USER>
```

2. Save the old Hive configuration and add symlink from /etc/hive/conf:

```
mv /etc/hive/conf /etc/hive/conf.saved;

mv /etc/hive/conf.server /etc/hive/conf.server.saved

ln -s /usr/hdp/current/hive-client/conf /etc/hive/conf

ls -la /etc/hive
```

3. If you use Tez as the Hive execution engine, and if the variable `hive.server2.enabled.doAs` is set to true, you must create a scratch directory on the NameNode host for the username that will run the HiveServer2 service.

```
su -l <HDFS_USER> -c "hdfs dfs -mkdir /tmp/hive-<HIVE_USER>"

su -l <HDFS_USER> -c "hdfs dfs -chmod 777 /tmp/hive-<HIVE_USER>"
```

where <HIVE_USER> is the Hive Service user. For example, hive. And where <HDFS_USER> is the HDFS Service user. For example, hdfs.

4. From Ambari Web, browse to `Services > Hive > Configs`. Under the Advanced tab, add the following properties to Advanced hive-site, only if these properties do not already exist on the cluster:

Name	Value
hive.cluster.delegation.token.store.zookeeper.connectString	The ZooKeeper token store connect string. For example: ZooKeeperHost:2181
hive.zookeeper.quorum	The comma-separated list of ZooKeeper hosts to talk to. For example: ZooKeeperHost1:2181, ZooKeeperHost2:2181

5. For WebHCat , upload new Pig, Hive, and Sqoop tarballs to HDFS. Run the following command from a node that has the HDP clients installed:

```
su -l <HDFS_USER> -c "hdfs dfs -mkdir -p /hdp/apps/2.3.x.y-z/pig/"
su -l <HDFS_USER> -c "hdfs dfs -mkdir -p /hdp/apps/2.3.x.y-z/hive/"
su -l <HDFS_USER> -c "hdfs dfs -mkdir -p /hdp/apps/2.3.x.y-z/sqoop/"
su -l <HDFS_USER> -c "hdfs dfs -put /usr/hdp/current/pig-client/pig.tar.gz /
hdp/apps/2.3.x.y-z/pig/"
su -l <HDFS_USER> -c "hdfs dfs -put /usr/hdp/current/hive-client/hive.tar.
gz /hdp/apps/2.3.x.y-z/hive/"
su -l <HDFS_USER> -c "hdfs dfs -put /usr/hdp/current/sqoop-client/sqoop.tar.
gz /hdp/apps/2.3.x.y-z/sqoop/"
su -l <HDFS_USER> -c "hdfs dfs -chmod -R 555 /hdp/apps/2.3.x.y-z/pig"
su -l <HDFS_USER> -c "hdfs dfs -chmod -R 444 /hdp/apps/2.3.x.y-z/pig/pig.
tar.gz"
su -l <HDFS_USER> -c "hdfs dfs -chmod -R 555 /hdp/apps/2.3.x.y-z/hive"
su -l <HDFS_USER> -c "hdfs dfs -chmod -R 444 /hdp/apps/2.3.x.y-z/hive/hive.
tar.gz"
su -l <HDFS_USER> -c "hdfs dfs -chmod -R 555 /hdp/apps/2.3.x.y-z/sqoop"
su -l <HDFS_USER> -c "hdfs dfs -chmod -R 444 /hdp/apps/2.3.x.y-z/sqoop/
sqoop.tar.gz"
su -l <HDFS_USER> -c "hdfs dfs -chown -R <HDFS_USER>:<HADOOP_GROUP> /hdp"
```

where <HDFS_USER> is the HDFS Service user. For example, hdfs.

6. In Ambari Web, browse to Services > Hive and start Hive.
7. After Hive has started, select Run Service Check from the Service Actions menu. Confirm the check passes.

4.2.4.9. Upgrade Pig

1. From Ambari Web, browse to `Services > Pig.`
2. Save the old Pig configuration and add symlink from /etc/pig/conf:

   ```
   mv /etc/pig/conf /etc/pig/conf.saved
   ```

   ```
   ln -s /usr/hdp/current/pig-client/conf /etc/pig/conf
   ```

   ```
   ls -la /etc/pig
   ```

   ```
   total 4
   drwxr-xr-x 3 root root 4096 Jun 19 21:51 2.3.0.0-2323
   lrwxrwxrwx 1 root root   35 Jun 19 21:54 conf -> /usr/hdp/current/pig-
   client/conf
   drwxr-xr-x 2 root root 4096 Jun 14 00:11 conf.saved
   ```

3. Select Refresh configs from the Service Actions menu.
4. Select Run Service Check from the Service Actions menu. Confirm the Service Check passes.

4.2.4.10. Upgrade Sqoop

1. Upload new Sqoop tarballs to HDFS, if you have not already done so in the prior Hive and WebHCat step. Execute these commands on the host with the Sqoop client:

```
su -l <HDFS_USER> -c "hdfs dfs -mkdir -p /hdp/apps/2.3.x.y-z/sqoop/"
su -l <HDFS_USER> -c "hdfs dfs -put /usr/hdp/current/sqoop-client/sqoop.tar.
gz /hdp/apps/2.3.x.y-z/sqoop/"
su -l <HDFS_USER> -c "hdfs dfs -chmod -R 555 /hdp/apps/2.3.x.y-z/sqoop"
su -l <HDFS_USER> -c "hdfs dfs -chmod -R 444 /hdp/apps/2.3.x.y-z/sqoop/
sqoop.tar.gz"
su -l <HDFS_USER> -c "hdfs dfs -chown -R <HDFS_USER>:<HADOOP_GROUP>/hdp"
```

where <HDFS_USER> is the HDFS Service user. For example, hdfs.

2. Save the old Sqoop configuration and add symlink from /etc/sqoop/conf:

   ```
   mv /etc/sqoop/conf /etc/sqoop/conf.saved
   ```

   ```
   ln -s /usr/hdp/current/sqoop-client/conf /etc/pig/conf
   ```

   ```
   ls -la /etc/sqoop
   ```

   ```
   total 4
   drwxr-xr-x 3 root root 4096 Jun 19 21:51 2.3.0.0-2323
   lrwxrwxrwx 1 root root   35 Jun 19 21:54 conf -> /usr/hdp/current/sqoop-
   client/conf
   drwxr-xr-x 2 root root 4096 Jun 14 00:11 conf.saved
   ```

4.2.4.11. Upgrade Flume

1. Save the old Flume configuration and add symlink from /etc/flume/conf:

   ```
   mv /etc/flume/conf /etc/flume/conf.saved
   ```

   ```
   ln -s /usr/hdp/current/flume-server/conf /etc/flume/conf
   ```

   ```
   ls -la /etc/flume
   ```

   ```
   drwxr-xr-x. 3 root root 4096 Jul 11 22:20 2.3.0.0-2551
   lrwxrwxrwx. 1 root root 34 Jul 14 13:19 conf -> /usr/hdp/current/flume-
   server/conf
   drwxr-xr-x. 2 root root 4096 Jul 11 20:46 conf.saved
   ```

2. From Ambari Web, browse to `Services > Flume`.
3. From the `Service Actions` menu, select `Start`.
4. After Flume has started, select Run Service Check. Confirm the Service Check passes.

4.2.4.12. Upgrade Oozie

1. Prepare each Oozie server host:

   Note

   You must replace your Oozie configuration after upgrading.

   a. Backup and copy from your prior version Oozie configs:

      ```
      mv /etc/oozie/conf /etc/oozie/conf-backup
      ```

```
cp -R /etc/oozie/conf-backup/* /etc/oozie/2.3.x.y-z/0/
```

b. Create symlinks to /etc/oozie/conf:

```
ln -s /usr/hdp/current/oozie-client/conf /etc/oozie/conf

ls -la /etc/oozie
```

```
drwxr-xr-x. 4 root root 4096 Jul 7 14:04 .
drwxr-xr-x. 98 root root 4096 Jul 7 11:31 ..
drwxr-xr-x. 3 root root 4096 Jul 7 08:54 2.3.0.0-2410
lrwxrwxrwx. 1 root root 34 Jul 7 14:04 conf -> /usr/hdp/current/oozie-
client/conf
drwxr-xr-x. 4 oozie hadoop 4096 Jul 7 02:06 conf-backup
[root@hdpone oozie-server]# total 16
```

c. Create `/usr/hdp/2.3.x.y-z/oozie/libext-upgrade23` directory.

```
mkdir /usr/hdp/2.3.x.y-z/oozie/libext-upgrade23
```

d. Copy the JDBC jar of your Oozie database to both `/usr/hdp/2.3.x.y-z/oozie/libext-upgrade22` and `/usr/hdp/2.3.x.y-z/oozie/libtools`. For example, if you are using MySQL, copy your `mysql-connector-java.jar`.

e. Copy these files to `/usr/hdp/2.3.x.y-z/oozie/libext-upgrade23` directory

```
cp /usr/hdp/2.3.x.y-z/hadoop/lib/hadoop*lzo*.jar /usr/hdp/
current/oozie-server/libext-upgrade23/

cp /usr/share/HDP-oozie/ext.zip /usr/hdp/current/oozie-server/
libext-upgrade23/ext-2.2.zip
```

f. Grant read/write access to the Oozie user.

```
chmod -R 777 /usr/hdp/current/oozie-server/libext-upgrade23
```

g. If Falcon was also installed and configured before upgrade in HDP 2.2.x, then you need to do the follo

```
cp /usr/hdp/current/falcon-server/oozie/ext/falcon-oozie-el-
extension-*jar /usr/hdp/current/oozie-server/libext-upgrade23/
```

2. Upgrade steps:

a. On the Services view, make sure that YARN and MapReduce2 services are running.

b. Make sure that the Oozie service is stopped.

c. In `/etc/oozie/conf/oozie-env.sh`, comment out CATALINA_BASE property, also do the same using Ambari Web UI in `Services` > `Oozie` > `Configs` > `Advanced oozie-env`.

d. Extract share-lib.

```
cd /usr/hdp/current/oozie-server
```

```
tar xzvf /usr/hdp/current/oozie-server/oozie-sharelib.tar.gz

su -l <HDFS_USER> -c "hdfs dfs -mkdir -p /user/oozie"

su -l <HDFS_USER> -c "hdfs dfs -put /usr/hdp/current/oozie-
server/share /user/oozie/."
```

You can expect warnings that some files already exist. Delete any existing /oozie/share and replace it with the newly-extracted files.

```
su -l <HDFS_USER> -c "hdfs dfs -chown oozie:hadoop /user/
oozie"

su -l <HDFS_USER> -c "hdfs dfs -chmod -R 755 /user/oozie"
```

e. Upgrade Oozie.

```
su -l <OOZIE_USER> -c "/usr/hdp/current/oozie-server/bin/
ooziedb.sh upgrade -run"
```

f. Prepare the Oozie WAR file.

```
chown <OOZIE_USER>:<HADOOP_GROUP> /usr/hdp/current/oozie-
server/oozie-server/conf/server.xml

su -l <OOZIE_USER> -c "/usr/hdp/current/oozie-server/bin/
oozie-setup.sh prepare-war -d /usr/hdp/current/oozie-server/
libext-upgrade23"
```

3. From Ambari Web, browse to `Services > Oozie`. Select Run Service Check from the Service Actions menu. Confirm the Service Check passes.

4.2.4.13. Upgrade Falcon

1. Save the old Falcon configuation and add symlink from /etc/falcon/conf:

```
mv /etc/falcon/conf /etc/falcon/conf.saved

ln -s /usr/hdp/current/falcon-client/conf /etc/falcon/conf

ls -la /etc/falcon
```

```
total 4
drwxr-xr-x 3 root root 4096 Jun 19 21:51 2.3.0.0-2323
lrwxrwxrwx 1 root root   35 Jun 19 21:54 conf -> /usr/hdp/current/falcon-
client/conf
drwxr-xr-x 2 root root 4096 Jun 14 00:11 conf.saved
```

2. From Ambari Web, browse to `Services > Falcon`.

3. From the Service Actions menu, select Start.

4. After Falcon has started, select Run Service Check. Confirm the Check passes.

4.2.4.14. Upgrade Knox

1. Save the old Knox configuration and add symlink from /etc/knox/conf:

   ```
   mv /etc/knox/conf /etc/knox/conf.saved

   ln -s /usr/hdp/current/knox-server/conf /etc/knox/conf

   ll /etc/knox
   ```

   ```
   total 4
   drwxr-xr-x 3 root root 4096 Jun 19 21:51 2.3.0.0-2323
   lrwxrwxrwx 1 root root   35 Jun 19 21:54 conf -> /usr/hdp/current/knox-
   server/conf
   drwxr-xr-x 2 root root 4096 Jun 14 00:11 conf.saved
   ```

2. From Ambari Web, browse to `Services > Knox`.

3. From the Service Actions menu, select Start.

4. After Knox has started, select Run Service Check. Confirm the Check passes.

4.2.4.15. Upgrade Spark

1. Save the old Spark configuration and add symlink from /etc/spark/conf:

   ```
   mv /etc/spark/conf /etc/spark/conf.saved

   ln -s /usr/hdp/current/spark-client/conf /etc/spark/conf

   ls -la /etc/spark
   ```

   ```
   total 4
   drwxr-xr-x 3 root root 4096 Jun 19 21:51 2.3.0.0-2323
   lrwxrwxrwx 1 root root 35 Jun 19 21:54 conf -> /usr/hdp/current/spark-
   client/conf
   drwxr-xr-x 2 root root 4096 Jun 14 00:11 conf.saved
   ```

2. From Ambari Web, browse to `Services > Spark`.

3. From the Service Actions menu, select Start.

4. After Spark has started, select Run Service Check. Confirm the Service Check passes.

4.2.4.16. Upgrade Slider

1. Save the old Slider configuration and add symlink from /etc/slider/conf:

   ```
   mv /etc/slider/conf /etc/slider/conf.saved

   ln -s /usr/hdp/current/slider-client/conf /etc/kafka/conf

   ls -la /etc/kafka
   ```

```
total 4
drwxr-xr-x 3 root root 4096 Jun 19 21:51 2.3.0.0-2323
lrwxrwxrwx 1 root root   35 Jun 19 21:54 conf -> /usr/hdp/current/slider-
client/conf
drwxr-xr-x 2 root root 4096 Jun 14 00:11 conf.saved
```

2. From Ambari Web, browse to `Services > Slider`.
3. From the `Service Actions` menu, select `Refresh Configs`.
4. After refresh is complete, select Run Service Check from the Service Actions menu. Confirm the Service Check passes.

4.2.4.17. Upgrade Kafka

1. Save the old Kafka configuration and add symlink from /etc/kafka/conf:

   ```
   mv /etc/kafka/conf /etc/kafka/conf.saved
   ```

   ```
   ln -s /usr/hdp/current/kafka-broker/conf /etc/kafka/conf
   ```

   ```
   ls -la /etc/kafka
   ```

   ```
   total 4
   drwxr-xr-x 3 root root 4096 Jun 19 21:51 2.3.0.0-2323
   lrwxrwxrwx 1 root root   35 Jun 19 21:54 conf -> /usr/hdp/current/kafka-
   broker/conf
   drwxr-xr-x 2 root root 4096 Jun 14 00:11 conf.saved
   ```

2. From Ambari Web, browse to `Services > Kafka`.
3. From the `Service Actions` menu, select `Start`.
4. Copy the Kafka .properties files from the HDP 2.2 configuration directory into the HDP 2.3 configuration directory.

   ```
   cp /etc/kafka/conf.saved/tools-log4j.properties /etc/kafka/conf/
   ```

   ```
   cp /etc/kafka/conf.saved/test-log4j.properties /etc/kafka/conf
   ```

   ```
   cp /etc/kafka/conf.saved/zookeeper.properties /etc/kafka/conf/
   ```

   ```
   cp /etc/kafka/conf.saved/producer.properties /etc/kafka/conf/
   ```

   ```
   cp /etc/kafka/conf.saved/consumer.properties /etc/kafka/conf/
   ```

5. After Kafka has started, select Run Service Check. Confirm the Check passes.

4.2.4.18. Upgrade Ranger

1. Back Up Your Ranger Configuration Directories

   Ranger Admin

   ```
   mv /etc/ranger/admin/conf /etc/ranger/admin/conf.saved
   ```

```
ln -s /usr/hdp/current/ranger-admin/conf /etc/ranger/admin/conf

ls -la /etc/ranger/admin/
```

Ranger Usersync

```
mv /etc/ranger/usersync/conf /etc/ranger/usersync/conf.saved

ln -s /usr/hdp/current/ranger-usersync/conf /etc/ranger/
usersync/conf

ls -la /etc/ranger/usersync
```

2. On the Ranger host, update the Ranger Admin install properties file:

   ```
   vi /usr/hdp/2.3.x.y-z/ranger-admin/install.properties
   ```

   using the values from `/usr/hdp/2.2.x.y-z/ranger-admin/install.properties`.

   The following table lists the HDP 2.2 property strings and the HDP 2.3 property labels for Ranger:

HDP 2.2 install.properties	HDP 2.3 Ambari Web UI > Property Names
DB_FLAVOR	DB Settings > DB FLAVOR
SQL_CONNECTOR_JAR	Admin Settings > Sql Connector Jar location
db_root_user	DB Settings > Ranger DB root user
db_root_password	DB Settings > Ranger DB root password
db_host	DB Settings > Ranger DB host
db_name	DB Settings > Ranger DB name
db_user	DB Settings > Ranger DB username
db_password	DB Settings > Ranger DB password
audit_db_name	DB Settings > Ranger Audit DB name
audit_db_user	DB Settings > Ranger Audit DB username
audit_db_password	DB Settings > Ranger Audit DB password
policymgr_external_url	Ranger Settings > External URL

3. Copy the SQL_CONNECTOR_JAR in install.properties to the following directory:

   ```
   /usr/hdp/2.3.x.y-z/ranger-admin/ews/lib
   ```

4. Copy the `ranger-admin-default-site.xml` file:

   ```
   cp /usr/hdp/2.3.x.y-z/ranger-admin/ews/webapp/WEB-INF/classes/
   conf.dist/ranger-admin-default-site.xml /usr/hdp/2.3.x.y-z/
   ranger-admin/conf
   ```

5. Copy the `security-applicationContext.xml` file:

   ```
   cp /usr/hdp/2.3.x.y-z/ranger-admin/ews/webapp/WEB-INF/classes/
   conf.dist/security-applicationContext.xml /usr/hdp/2.3.x.y-z/
   ranger-admin/conf
   ```

6. Export the RANGER_ADMIN_HOME and JAVA_HOME environment variables.

   ```
   export RANGER_ADMIN_HOME=/usr/hdp/2.3.x.y-z/ranger-admin

   export JAVA_HOME=/usr/jdk64/jdk1.7.0_67
   ```

7. On the Ranger host, run the Ranger Admin dba_script and db_setup python scripts, as follows:

   ```
   python /usr/hdp/2.3.x.y-z/ranger-admin/dba_script.py -q

   python /usr/hdp/2.3.x.y-z/ranger-admin/db_setup.py
   ```

8. From Ambari Web UI, select `Ranger > Summary` and click on the `Ranger Admin` component link. Choose `Start` from the `Ranger Admin` drop-down menu.

9. Choose Ok to confirm Ranger Admin start.

10. On the Ranger host, run the dba_setup patch script, as follows:

    ```
    python /usr/hdp/2.3.x.y-z/ranger-admin/db_setup.py -javapatch
    ```

11. Create symbolic links and overwrite any existing destination files for the Ranger UserSync service.

    ```
    ln -sf /usr/hdp/2.3.x.y-z/ranger-usersync/ranger-usersync-
    services.sh /usr/bin/ranger-usersync
    ```

12. From Ambari Web UI, select `Ranger > Summary` and click on the Ranger UserSync component link. Choose `Start` from the `Ranger UserSync` drop-down menu.

13. Choose Ok to confirm Ranger UserSync start.

14. From Ambari Web UI, select `Ranger` and choose `Restart All` from the `Service Actions` menu.

15. After Ranger has started, select `Run Service Check` from the `Service Actions` menu. Confirm the Service Check passes.

4.2.4.19. Upgrade Storm

Prepare the Storm service properties.

1. Save the old Storm configuration and add symlink from /etc/storm/conf:

   ```
   mv /etc/storm/conf /etc/storm/conf.saved

   ln -s /usr/hdp/current/storm-client/conf /etc/storm/conf

   ls -la /etc/storm
   ```

   ```
   total 4
   drwxr-xr-x 3 root root 4096 Jun 19 21:51 2.3.0.0-2323
   lrwxrwxrwx 1 root root   35 Jun 19 21:54 conf -> /usr/hdp/current/storm-
   client/conf
   drwxr-xr-x 2 root root 4096 Jun 14 00:11 conf.saved
   ```

2. Delete all states under ZooKeeper:

   ```
   /usr/hdp/current/zookeeper-client/bin/zkCli.sh rmr /storm
   ```

   Optional: In a secure environment, include the "-server zk.server:port" option.

3. Remove the `storm.local.dir` from every host where the Storm component is installed.

   You can find this property in `Services > Storm > Configs` under the Settings tab.

   `rm -rf` <storm.local.dir>

4. Browse to `Services` > `Storm` > `Configs` under the `Advanced` tab. In the Supervisor section, find the `supervisor.childopts` property. Update the path for the `jmxetric-1.0.4.jar` file to the following:

   ```
   /usr/hdp/current/storm-nimbus/contrib/storm-jmxetric/lib/
   jmxetric-1.0.4.jar
   ```

   If the `supervisor.childopts` property value contains the following setting, delete the setting:

   ```
   -Djava.security.auth.login.config=/etc/storm/conf/
   storm_jaas.conf
   ```

   Retain any other settings in the `supervisor.childopts` box.

5. Browse to `Services` > `Storm` > `Configs` under the `Advanced` tab. In the Advanced storm-site section, add or update the following settings for `worker.childopts`:

   ```
   -Xmx768m

   _JAAS_PLACEHOLDER

   -javaagent:/usr/hdp/current/storm-client/contrib/storm-jmxetric/
   lib/jmxetric-1.0.4.jar
   ```

   Retain any other settings in the `worker.childopts` box.

6. If you are planning to enable secure mode, browse to Services > `Storm` > `Configs` under the `Advanced` tab and add the following property to the Advanced storm-site section:

   ```
   _storm.thrift.secure.transport=backtype.storm.security.auth.kerberos.
   KerberosSaslTransportPlugin
   ```

7. Start Storm, by selecting `Service Actions` > `Start`.

8. After Storm has started, select Run `Service Check` from the `Service Actions` menu. Confirm the Service Check passes.

Using `Ambari Web` > `Services` > `Service Actions`, re-start all stopped services.

4.2.4.20. Finalize Upgrade

The upgrade is now fully functional but not yet finalized. Using the `finalize` command removes the previous version of the NameNode and DataNode storage directories. After the upgrade is finalized, **the system cannot be rolled back**. Perform thorough testing of the upgraded cluster before finalizing the upgrade. The upgrade must be finalized before another upgrade can be performed.

Directories used by Hadoop 1 services set in `/etc/hadoop/conf/taskcontroller.cfg` are not automatically deleted after upgrade. Administrators can choose to delete these directories after the upgrade.

To finalize the upgrade, execute the following command once, on the primary NameNode host in your HDP cluster:

```
sudo su -l <HDFS_USER> -c "hdfs dfsadmin -finalizeUpgrade"
```

where <HDFS_USER> is the HDFS service user. For example, hdfs.

4.2.4.21. Set Current HDP Version

1. Restart all services from Ambari. One by one, browse to each Service in Ambari Web, and in the Service Actions menu select `Restart All`. On the client-only services, such as Tez, Pig, or Sqoop, select Refresh Configs.

2. After all the services are confirmed to be started and healthy, go to the command line on the Ambari Server and run the following to finalize the upgrade, which will move the current version to the new version.

   ```
   ambari-server set-current --cluster-name=$CLUSTERNAME --version-display-
   name=$VERSION_NAME
   Ambari Admin login: admin
   Ambari Admin password: *****
   ```

   where $CLUSTERNAME is the name of your cluster and $VERSION_NAME is the version name for the HDP version you just upgraded to (for example: **HDP-2.3.0.0**).

3. If your cluster includes Ranger: Use the "Step 2" upgrade catalog you downloaded in the Update Service Configurations section and run the helper:

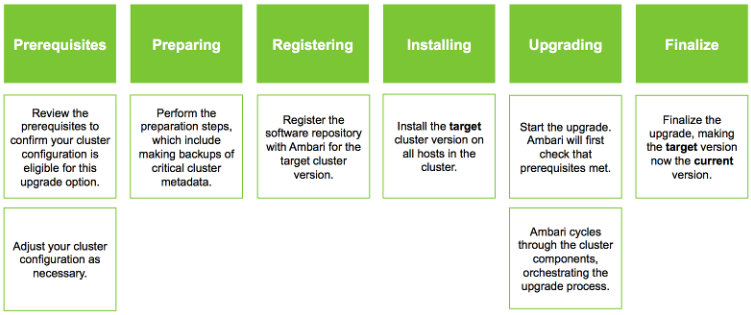
   ```
   python upgradeHelper.py --hostname $HOSTNAME --user
   $USERNAME --password $PASSWORD --clustername $CLUSTERNAME
   --fromStack $FROMSTACK --toStack $TOSTACK --upgradeCatalog
   UpgradeCatalog_2.2_to_2.3_step2.json update-configs [config-
   type]
   ```

4.3. Rolling Maintenance Upgrade

Use this procedure to perform a rolling upgrade from one HDP maintenance release to another. If your cluster is running HDP 2.2 or HDP 2.3, you can perform an upgrade to a later maintenance release of that HDP version. For example: you can upgrade from HDP 2.2.x to HDP 2.2.y OR from HDP 2.3.x to HDP 2.3.y. 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 and account for any special configurations for your cluster.

The high-level process for performing a Rolling HDP Maintenance upgrade is as follows:



- Prerequisites
- Preparing to Upgrade
- Registering New Version
- Installing a New Version
- Performing the Upgrade

Note

If you do not meet the rolling maintenance upgrade Prerequisites , you can consider a Manual Maintenance Upgrade of the cluster.

Important

If you need to upgrade between minor HDP stack releases (such as HDP 2.2 to HDP 2.3), refer the the sections around Minor Upgrades (Rolling or Manual).

4.3.1. Prerequisites

To perform a rolling upgrade, your cluster must meet the following prerequisites. If you do not meet these upgrade prerequisites, you can consider a Manual Upgrade HDP 2.2 to 2.3 of the cluster.

- General
- HDFS
- YARN
- MapReduce 2
- Tez

- Hive
- Oozie

General

Requirement	Description
Current HDP Version	Must be running HDP 2.2 or later to perform a rolling upgrade. The rolling upgrade capability is not available for clusters running HDP 2.0 or 2.1.
Target HDP Version	All hosts must have the target version installed. See the Register Version and Install Version sections for more information.
Ambari Agent Heartbeats	All Ambari Agents must be heartbeating to Ambari Server. Any hosts that are not heartbeating must be in Maintenance Mode.
Host Maintenance Mode	Any hosts in Maintenance Mode must not be hosting any Service master components.
Service Maintenance Mode	No Services can be in Maintenance Mode.
Services Started	All Services must be started and the Service Check must pass.

HDFS

Requirement	Description
NameNode HA	NameNode HA must be enabled and working properly. See the Ambari User's Guide for more information, Configuring NameNode High Availability.
NameNode Truncate	HDP 2.2.6 introduced the NameNode Truncate option. Truncate must not be enabled.
Client Retry	HDFS client retry should be dfs.client.retry.policy.enabled

YARN

Requirement	Description
ResourceManager HA	YARN ResourceManager HA should be enabled to prevent a disruption in service during the upgrade. See the Ambari User's Guide for more information on Configuring ResourceManager High Availability.
Start Preserving Recovery	YARN start preserving recovery should be enabled. Check the **Services > YARN > Configs** property `yarn.timeline-service.recovery.enabled`.
Work Preserving Restart	YARN Work Preserving Restart must be configured. Check the **Services > YARN > Configs** property `yarn.resourcemanager.work-preserving-recovery.enabled` .

MapReduce 2

Requirement	Description
MapReduce Distributed Cache	MapReduce should reference Hadoop libraries from the distributed cache in HDFS. Refer to the YARN Resource Management guide for more information.
State Preserving Recovery	JobHistory state preserving recovery should be enabled.

Tez

Requirement	Description
Tez Distributed Cache	Tez should reference Hadoop libraries from the distributed cache in HDFS.

Hive

Requirement	Description
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.
Hive Dynamic Service Discovery	HiveServer2 dynamic service discovery is recommended for Rolling Upgrade.
HiveServer2 Port	During the upgrade, Ambari will switch the HiveServer2 port from 10000 to 10010 (or 10011 if using HTTP transport mode).
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.

Oozie

Requirement	Description
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.

If you do not meet the upgrade prerequisite requirements listed above, you can consider a Manual Maintenance Upgrade of the cluster.

4.3.2. Preparing to Upgrade

Review Preparing to Upgrade Ambari and HDP. It is also **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

Proceed to Registering a New Version.

4.3.3. Registering a New Version

Register the HDP Maintenance Version

1. Log in to Ambari.
2. Browse to `Admin > Stack and Versions`.
3. Click on the `Versions` tab. You will see the version currently running. The full version is dependent on the HDP version you are actually running. For example, if you are

currently running a maintenance release of HDP 2.2, you would see something like **HDP-2.2.0.0-2041**.

4. Click `Manage Versions`.
5. Proceed to register a new version by clicking `+ Register Version`.
6. Enter a two-digit version number. For example, enter **4.2** (which makes the version name **HDP-2.2.4.2**).

7. Select one or more OS families and enter the respective Base URLs for that OS.

   To copy a Base URL for a specific OS family, see the list of available HDP Repositories.

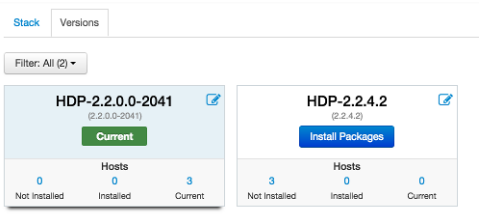
8. Click `Save`.
9. Click `Go to Dashboard` and browse back to `Admin > Stack and Versions > Versions`. You will see the version currently running (for example, **HDP-2.2.0.0-2041**) and the version you just registered, **HDP-2.2.4.2**.

   Proceed to Installing a New Version.

4.3.4. Installing a New Version

Install an HDP Maintenance Version on All Hosts

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



4. Click `Install Packages` and click OK to confirm.
5. The Install version operation will start and the new version will be installed on all hosts.
6. You can browse to Hosts and to each host > Versions tab to see the new version is installed.

   Proceed to Perform Upgrade.

4.3.5. Performing the Upgrade

Perform the HDP Maintenance Upgrade

1. Log in to Ambari.
2. Browse to `Admin > Stack and Versions`.
3. Click on the `Versions` tab.
4. Click `Perform Upgrade.`
5. Ambari will check that your cluster meets the prerequisites. A dialog will be presented with the results:
   a. If any prerequisites are not met but are required, the result will be shown with an error. You will not be allowed to proceed with the upgrade. Make the appropriate corrections and return to Perform Upgrade again.
   b. If any prerequisites are not met but are optional, the result will be shown with a warning. You will be allowed to Proceed with the upgrade.
   c. A list of configuration changes (if any) will be displayed.
6. Once the prerequisite checks are complete, the upgrade will start. The time it takes to perform the upgrade dependents on many factors. As part of the upgrade process, each component in the cluster is restarted in a serial fashion so the stop/start time is a big contributor to the overall time.
7. The upgrade process involves a set of stages. This table lists the high-level stages and if the process requires any action by you during normal operation. Note: if any stage fails, the upgrade will halt and prompt for action.

Stage	Description	Action Required
Prepare Upgrade	This stage prepares HDFS and HBase for upgrade and reminds you to perform backups of your database for Hive, Oozie and Ranger.	You must acknowledge the prompt for database backups.
ZooKeeper	All ZooKeeper servers are upgraded and restarted.	None
Ranger	Ranger Admin and UserSync servers are upgraded and restarted.	None
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
Service Checks	All Service Checks are performed against the cluster.	If any service check fails, you will be prompted to Ignore and Continue, Downgrade or Rerowy.
Core Slaves	This stage upgrades the slave components of core services. This includes DataNodes (HDFS), RegionServers (HBase) and NodeManagers (YARN). This is down	After the first 20% batch is complete, you will be prompted to Verify the cluster is operating correctly.

Stage	Description	Action Required
	in two batches: 20% of the slaves first, then the remaining slaves.	
Service Checks	All Service Checks are performed against the cluster.	If any service check fails, you will be prompted to Ignore and Continue, Downgrade or Retry.
Hive	This stage upgrades the Hive Metastore, HiveServer2 and WebHCat components.	Ambari will switch the HiveServer2 port from 10000 to 10010 (or, 10011 if using HTTP transport mode). You will be prompted to confirm prior to the switch
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
Clients	All remaining clients are upgraded.	None
Service Checks	All Service Checks are performed against the cluster.	If any service check fails, you will be prompted to Ignore and Continue, Downgrade or Rerowy.
Kafka	The Kafka Brokers are upgraded.	None
Knox	The Knox Gateways are upgraded.	None
Storm	The Nimbus, REST API, DRPC Server and UI Server components are upgraded, along with the Supervisors.	None
Slider	The Slider clients are upgraded.	None
Flume	The Flume agents are upgrade.	None
Finalize	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, Finalize Later or Downgrade.

8. Once the upgrade is complete, you have an option 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: the reverse of 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 a 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).

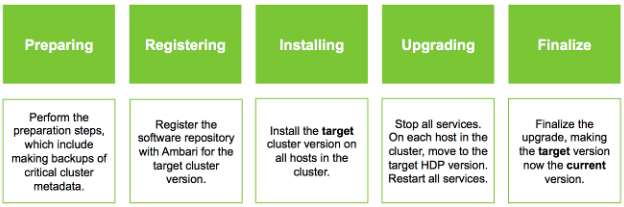
9. Click **Finalize** and the upgrade process is complete.

4.4. Manual Maintenance Upgrade

Use this procedure to perform a manual upgrade from one HDP maintenance release to another. If your cluster is running HDP 2.2 or HDP 2.3, you can perform a manual upgrade to a later maintenance release of that HDP version. For example: you can upgrade from HDP 2.2.x to HDP 2.2.y OR from HDP 2.3.x to HDP 2.3.y. 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 and account for any special configurations for your cluster.

The high-level process for performing a Manual HDP Maintenance upgrade is as follows:



- Preparing to Upgrade
- Registering a New Version
- Installing a New Version
- Performing the Upgrade

Note

This process is an alternative to the Rolling Maintenance Upgrade feature. If you meet the Rolling Maintenance Upgrade Prerequisites, you can consider that process.

Important

If you need to upgrade between minor HDP stack releases (such as HDP 2.2 to HDP 2.3), refer the the sections about Minor Upgrades, Rolling or Manual.

4.4.1. Preparing to Upgrade

Review Preparing to Upgrade Ambari and HDP. It is also **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

Proceed to Registering a New Version.

4.4.2. Registering a New Version

Register HDP Maintenance Version

1. Log in to Ambari.
2. Browse to `Admin > Stack and Versions`.
3. Click on the `Versions` tab. You will see the version currently running. The full version is dependent on the HDP version you are actually running. For example, if you are currently running a maintenance release of HDP 2.2, you would see something like **HDP-2.2.0.0-2041**.
4. Click `Manage Versions`.
5. Proceed to register a new version by clicking `+ Register Version`.
6. Enter a two-digit version number. For example, enter **4.2** (which makes the version name **HDP-2.2.4.2**).

7. Select one or more OS families and enter the repository Base URLs for that OS.

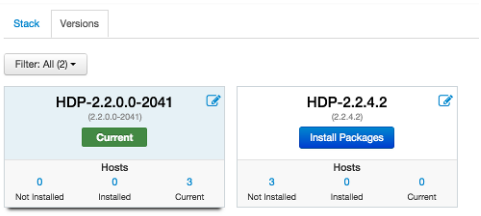
   To copy a Base URL for a specific OS family, see the list of available HDP Repositories.
8. Click `Save`.
9. Click `Go to Dashboard` and browse back to `Admin > Stack and Versions > Versions`. You will see the version currently running (for example, **HDP-2.2.0.0-2041**) and the version you just registered, **HDP-2.2.4.2**.

   Proceed to Installing a New Version.

4.4.3. Installing a New Version

Install HDP Maintenance Version on All Hosts

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



4. Click `Install Packages` and click OK to confirm.
5. The Install version operation will start and the new version will be installed on all hosts.

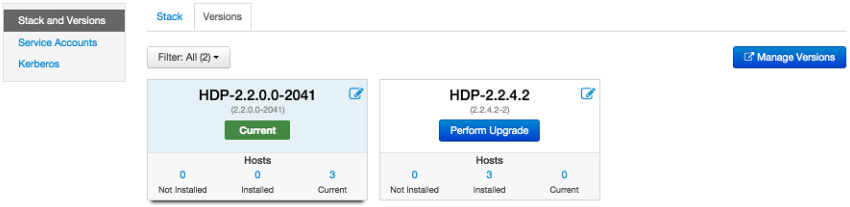
6. You can browse to Hosts and to each host > Versions tab to see the new version is installed.

   Proceed to Performing the Upgrade.

4.4.4. Performing the Upgrade

Perform the HDP Maintenance Upgrade

1. Log in to Ambari.
2. Browse to `Admin > Stack and Versions`.
3. Click on the `Versions` tab.
4. Under the newly registered and installed version **HDP-2.2.4.2**, is the actual software repository version in parenthesis (Ambari determined this repository version during the install). For example, in the picture below the display name is **HDP-2.2.4.2** and the repository version **2.2.4.2-2**. Record this repository version. You will use it later in the manual upgrade process.



5. Stop all services from Ambari. On the Services tab, in the Service navigation area Actions button, select `Stop All` to stop all services.
6. Go to the command line on each host and move the current HDP version to the newly installed version using the hdp-select utility and repository version number (obtained in Step 4).

   ```
   hdp-select set all {repository-version}
   ```

   For example:

   ```
   hdp-select set all 2.2.4.2-2
   ```

7. Restart all services from Ambari, starting with ZooKeeper. From Ambari Web, browse to Services > Zookeeper, and in the Service Actions menu select `Restart All`. Then, one-by-one, restart all other installed services in the same way. On the client-only services, such as Tez, Pig, or Sqoop, select `Refresh Configs`.
8. After all the services are confirmed to be started and healthy, go to the command line on the Ambari Server and run the following to finalize the upgrade, which will move the current version to the new version.

   ```
   ambari-server set-current --cluster-name=MyCluster --version-display-name=
   HDP-2.2.4.2

    Ambari Admin login: admin
   ```

```
Ambari Admin password: *****
```

9. If the `ambari-server set-current` command is not successful, try restarting the Ambari Server and waiting for all agents to re-register before trying again.