HDP Administration - Core Lab Guide

ADM-221 HDP Administration I Foundations
Course Rev 2.6.2-1.0

Table of Contents

Lab 0: Environment Setup ... 1
Lab 1: Installing HDP ... 4
Lab 2: Managing Ambari Users and Groups ... 16
Lab 3: Managing Hadoop Services ... 33
Lab 4: Using HDFS Storage ... 49
Lab 5: Using WebHDFS ... 65
Lab 6: Using HDFS Access Control Lists ... 69
Lab 7: Managing Hadoop Storage ... 79
Lab 8: Managing HDFS Quotas ... 91
Lab 9: Configuring Rack Awareness ... 95
Lab 10: Managing HDFS Snapshots ... 101
Lab 11: Using DistCp ... 106
Lab 12: Configuring HDFS Storage Policies ... 110
Lab 13: Configuring HDFS Centralized Cache ... 121
Lab 14: Configuring an NFS Gateway ... 129
Lab 15: Managing the YARN Service Using Ambari Web UI ... 139
Lab 16: Managing the YARN Service Using the CLI ... 149
Lab 17: Running Sample YARN Applications ... 153
Lab 18: Setting Up for Capacity Scheduler Labs ... 167
Lab 19: Managing YARN Containers and Queues ... 171
Lab 20: Managing YARN ACLs and User Limits ... 199
Lab 21: Yarn Node Labels ... 213
Lab 22: Configuring NameNode HA ... 229
Lab 23: Configuring ResourceManager HA ... 243
Lab 24: Adding, Decommissioning, and Recommissioning Worker Nodes ... 253
Lab 25: Managing Ambari Alerts ... 275
Lab 26: Deploy a HDP Cluster using Ambari Blueprints ... 295
Lab 27: Performing a HDP Upgrade ... 305

Lab 0: Environment Setup

About This Lab

Objective:	To connect to AWS nodes.
File locations:	N/A
Successful outcome:	**You will: Connect to your AWS nodes via SSH**
Related lesson:	***Installing the Hortonworks Data Platform***

Accessing your Cluster

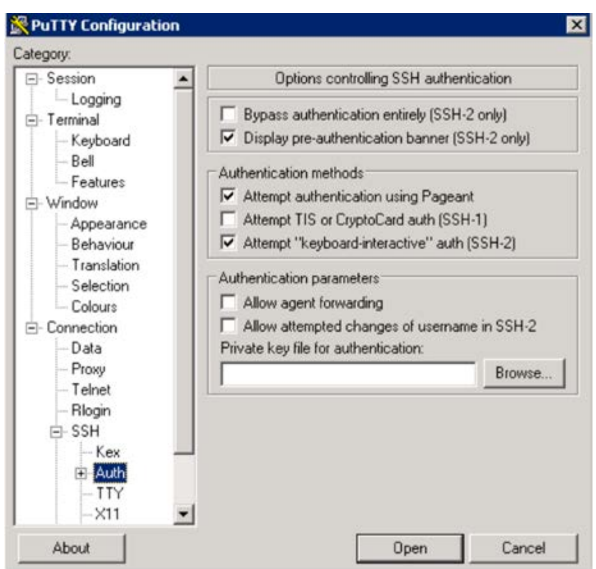
Credentials will be provided for these services by the instructor:

- SSH
- Ambari

To connect using Putty from Windows laptop

1. Download the PPK Key needed for PuTTY
2. Download the Pem Key this will be needed for the HDP Install Lab.
3. Use putty to connect to your node using the ppk key:

    - Connection > SSH > Auth > Private key for authentication > Browse... > Select training-keypair.ppk



- Make sure to click "Save" on the session page before logging in

To connect from Linux/MacOSX laptop

1. Download the Pem key > Save link as > save to Downloads folder
2. SSH into Ambari node of your cluster using below steps:
3. Check .pem and correct permissions

```
ls -l training-keypair.pem
chmod 400 training-keypair.pem
```

- o Login to the Ambari node of the cluster you have been assigned by replacing <**Ambari Server Hostname>** below with Ambari Hostname (your instructor will provide this)

```
ssh -i  training-keypair.pem centos@<Ambari Server External Hostname>
```

- o To change user to root you can run the following command:

```
sudo su -
```

4. Similarly, login via SSH to each of the other nodes in your cluster as you will need to run commands on each node in a future lab
5. Tip: Since in the next labs you will be required to run *the same set of commands* on each of the cluster hosts, now would be a good time to setup your favorite tool to do so: examples here
   - o On OSX, an easy way to do this is to use iTerm: open multiple tabs/splits and then use 'Broadcast input' feature (under Shell -> Broadcast input)
   - o If you are not already familiar with such a tool, you can also just run the commands on the cluster, one host at a time

Login to Ambari

- Login to Ambari web UI by opening http://<Ambari Server Hostname>:8080 and login with admin/BadPass#1
- You will see a Ambari's Management Page showing no clusters running.

- Instructor will provide you with 4 AWS Instances with Public Hostname and Private Hostname.
- Below are examples of the information:

List of External Hostnames: Used to access the AWS Instances from your Laptop

Additional Node ec2-52-37-242-36.us-west-2.compute.amazonaws.com
Additional Node ec2-35-167-218-88.us-west-2.compute.amazonaws.com
Additional Node ec2-35-160-131-246.us-west-2.compute.amazonaws.com
Ambari Node ec2-52-89-118-215.us-west-2.compute.amazonaws.com

List of Internal Hostnames: Used during the Installation of HDP Lab

ip-172-30-0-87.us-west-2.compute.internal
ip-172-30-0-128.us-west-2.compute.internal
ip-172-30-0-126.us-west-2.compute.internal
ip-172-30-0-179.us-west-2.compute.internal

Login to all four of your nodes using ssh client or PuTTy if you are using Microsoft Windows. Below is example from a Linux or Mac OS system.

```
ssh -i training-keypairs.pem centos@<YOUR EXTERNAL HOSTNAME>
```

Switch users to root:

```
sudo su -
```

Result

You have now:
You have logged in to all four of your AWS instances using SSH with a key and switch to the root user. Your lab environment is now ready for next lab “Installing HDP”

Lab 1: Installing HDP

About This Lab

Objective:	To interactively install a multi-node HDP cluster using the Ambari Web UI
File locations:	N/A
Successful outcome:	**You will:** Login to the Ambari Server; and perform an interactive HDP installation using the Ambari Web UI.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Installing the Hortonworks Data Platform***

Installing HDP

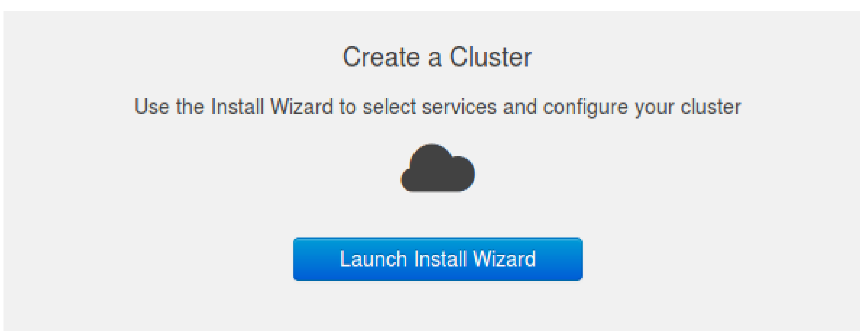
Perform an interactive HDP installation using the Ambari Web UI

1. Open the browser and connect to the Ambari Server at the URL:

   http://<Ambari Server Hostname>:8080

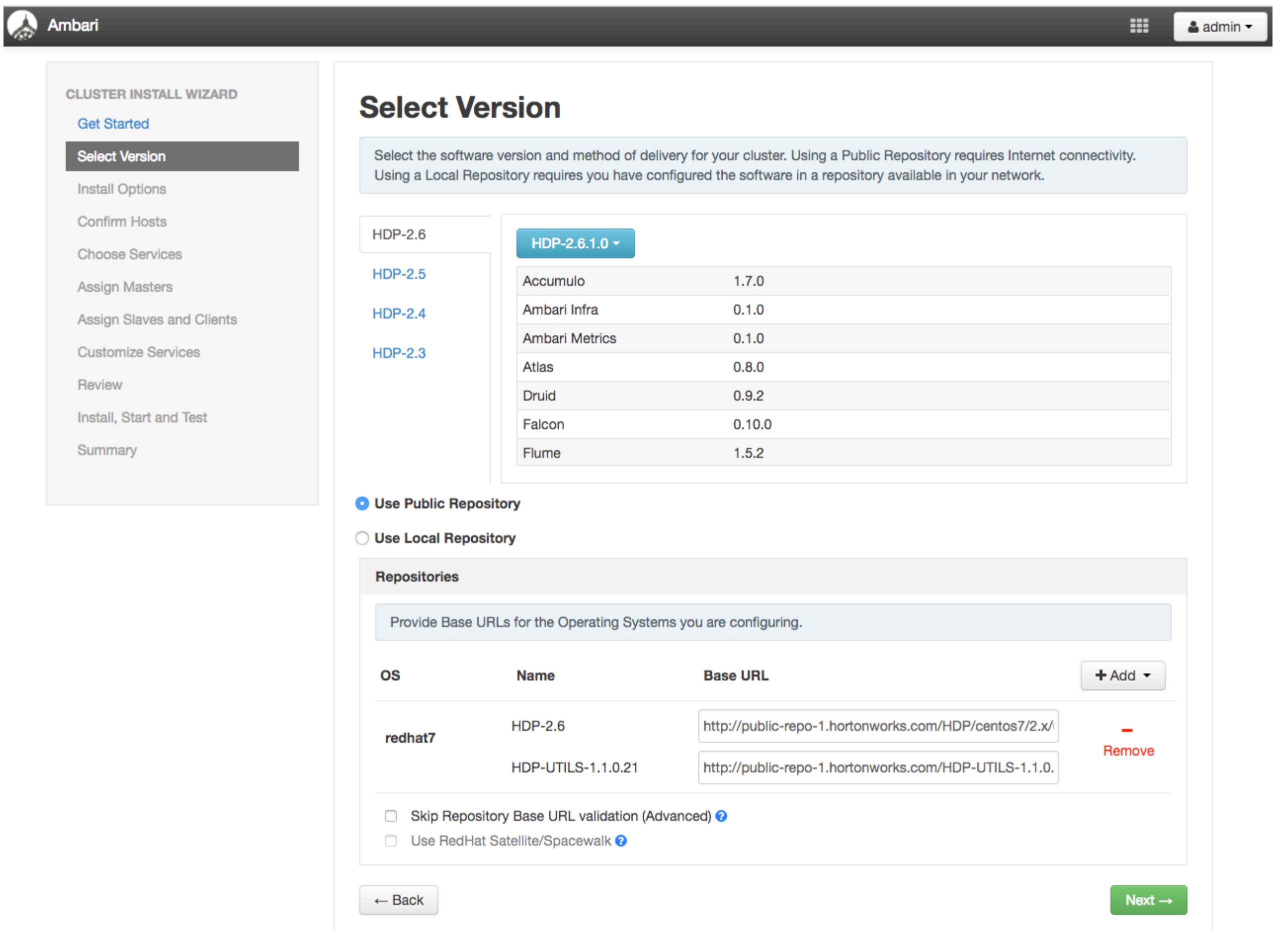
2. Login to the Ambari Web UI using the user name `admin` and the password `BadPass#1`.
3. Because the Ambari Server cannot find an installed cluster, it presents the option to install a cluster.

   Click **Launch Install Wizard**.



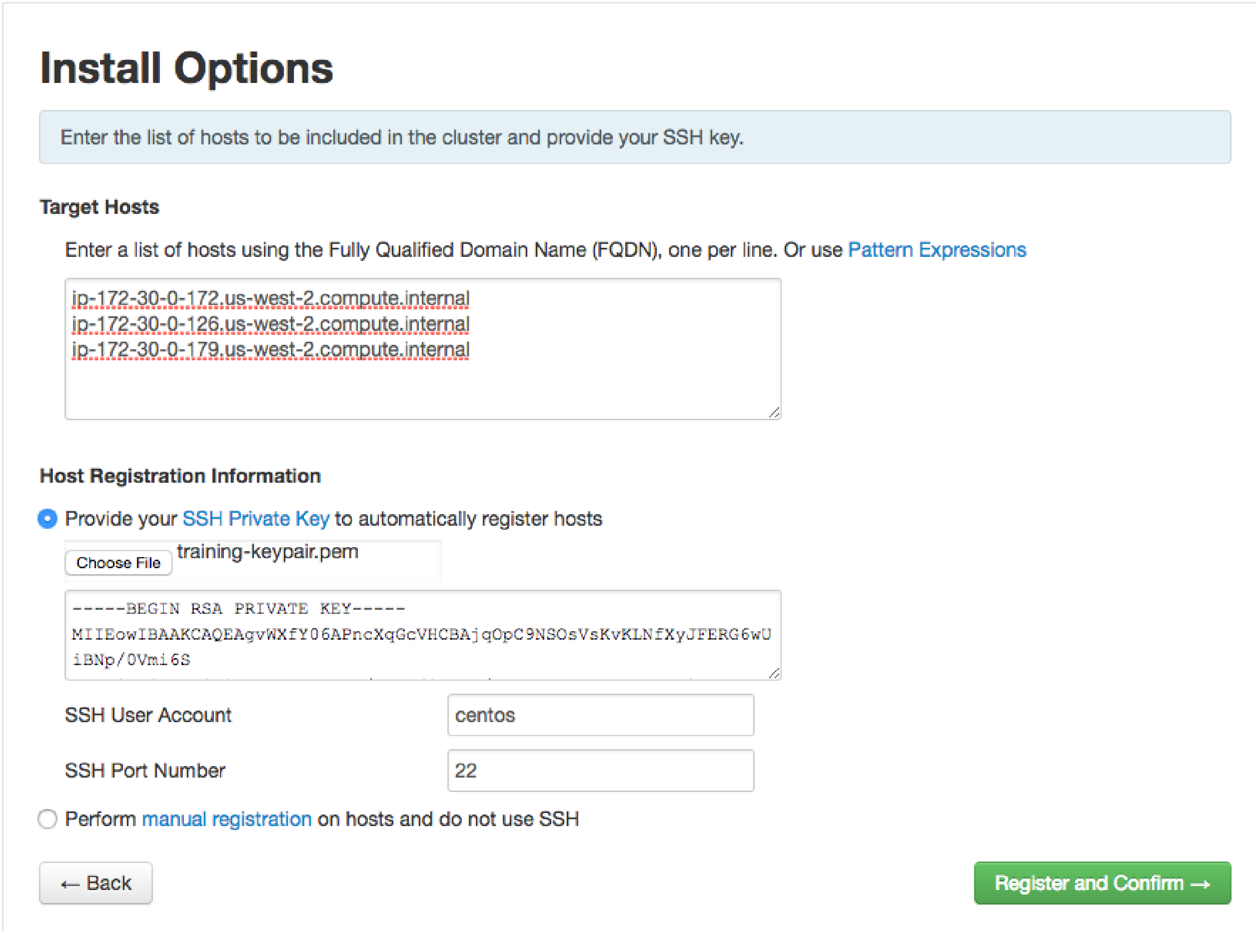
4. Enter a cluster name and click **Next**.

5 . Select the **HDP 2.6** version. Notice you are using the Public Repository.

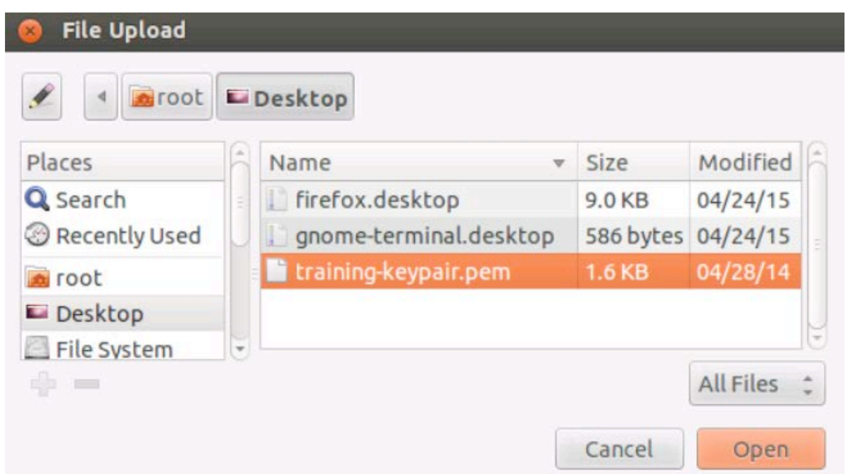


6 . Add three of your AWS hosts in the target hosts box. Use the Internal Hostnames provided to you by your instructor.

In a later lab exercise you will add the remaining AWS host to the cluster.

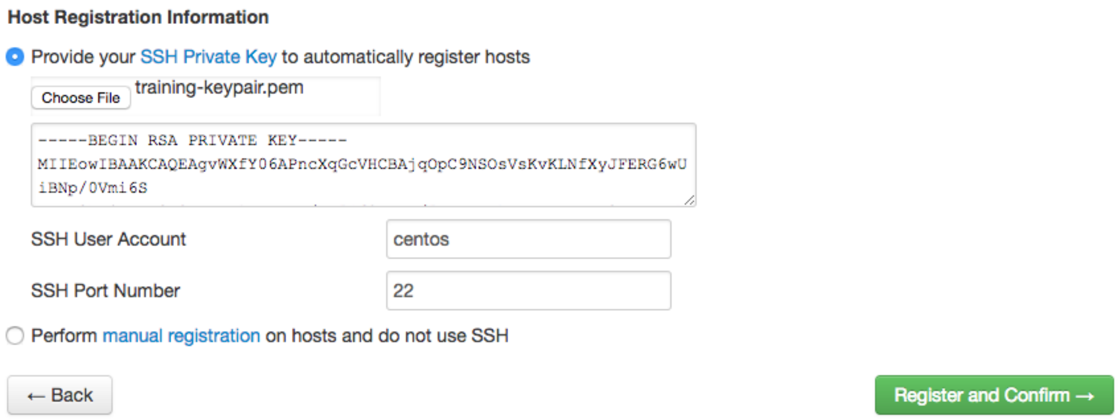


7. In the same installation window, click **Browse** to locate the password-less SSH private key file that is required for automated Ambari agent installation and registration.
8. Locate the `training-keypair.pem` file and **Open** it.



9. Make sure you change the SSH User Account from `root` to `centos`.

10 . Click **Register and Confirm** to install and register the Ambari agent.

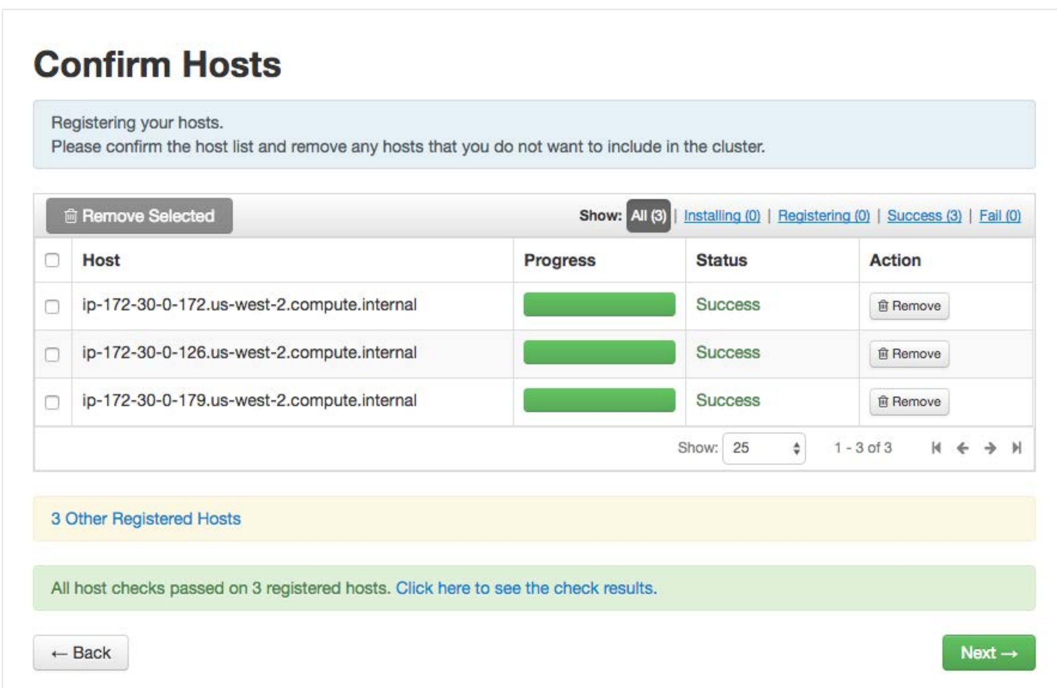


11 . Monitor the agent installation progress window.

The agent is installed and registered and then a series of system checks are performed on the nodes to be installed.

If you receive any warnings these can be ignored in the lab environment.

Click **Next** to continue the installation.

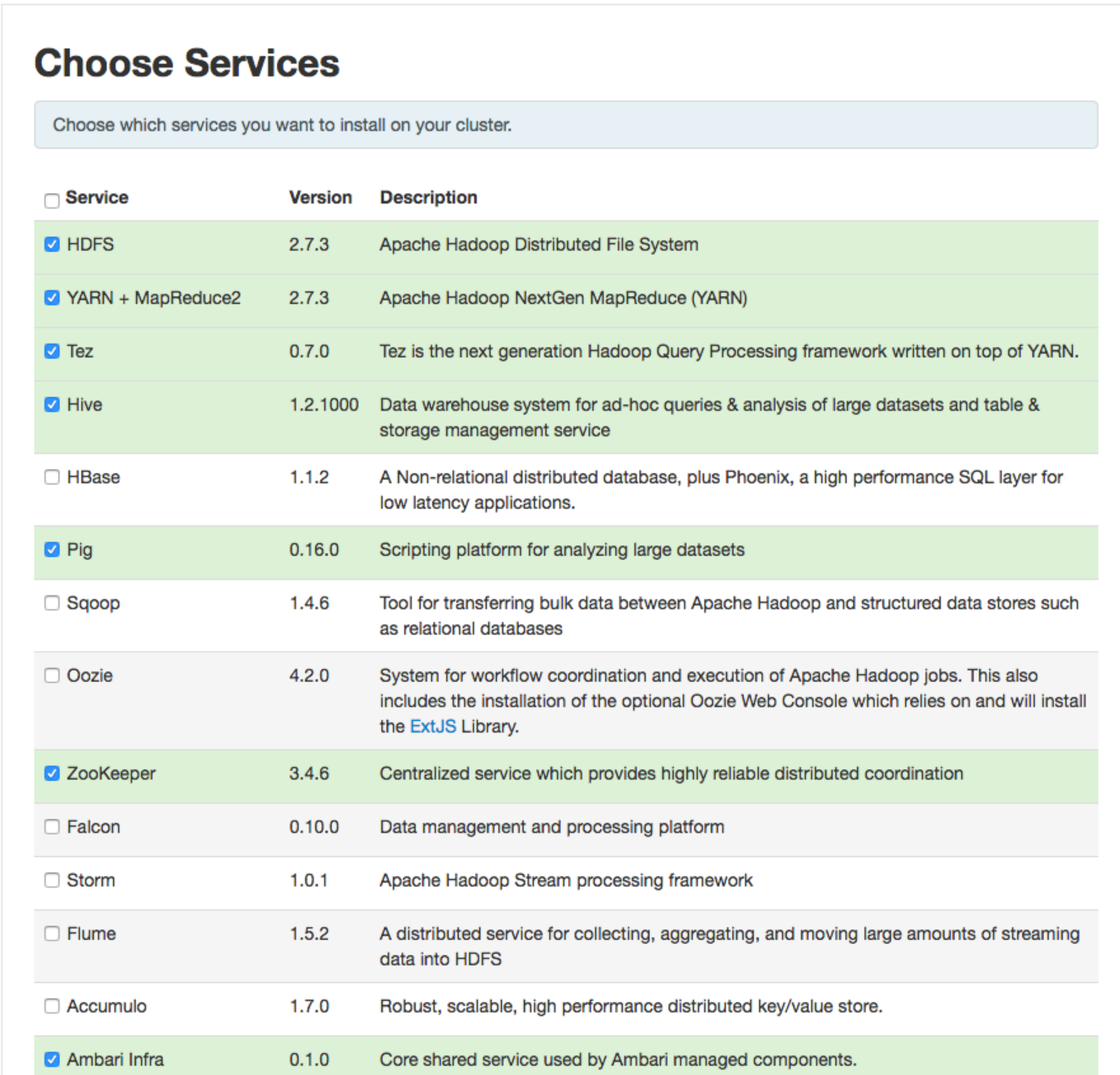


12 . Next, choose which cluster services to install.

To reduce the installation time in lab and to minimize the lab environment hardware requirements, you will only install a few of the available services.

Ensure that only **HDFS**, **YARN + MapReduce2**, **Tez**, **Hive, Pig**, **ZooKeeper**, **Ambari Infra, Ambari Metrics** and **Slider** are selected. SmartSense is autoselected and required component.

Then click **Next**.



Choose Services

Choose which services you want to install on your cluster.

☐ Service	Version	Description
☑ HDFS	2.7.3	Apache Hadoop Distributed File System
☑ YARN + MapReduce2	2.7.3	Apache Hadoop NextGen MapReduce (YARN)
☑ Tez	0.7.0	Tez is the next generation Hadoop Query Processing framework written on top of YARN.
☑ Hive	1.2.1000	Data warehouse system for ad-hoc queries & analysis of large datasets and table & storage management service
☐ HBase	1.1.2	A Non-relational distributed database, plus Phoenix, a high performance SQL layer for low latency applications.
☑ Pig	0.16.0	Scripting platform for analyzing large datasets
☐ Sqoop	1.4.6	Tool for transferring bulk data between Apache Hadoop and structured data stores such as relational databases
☐ Oozie	4.2.0	System for workflow coordination and execution of Apache Hadoop jobs. This also includes the installation of the optional Oozie Web Console which relies on and will install the ExtJS Library.
☑ ZooKeeper	3.4.6	Centralized service which provides highly reliable distributed coordination
☐ Falcon	0.10.0	Data management and processing platform
☐ Storm	1.0.1	Apache Hadoop Stream processing framework
☐ Flume	1.5.2	A distributed service for collecting, aggregating, and moving large amounts of streaming data into HDFS
☐ Accumulo	1.7.0	Robust, scalable, high performance distributed key/value store.
☑ Ambari Infra	0.1.0	Core shared service used by Ambari managed components.

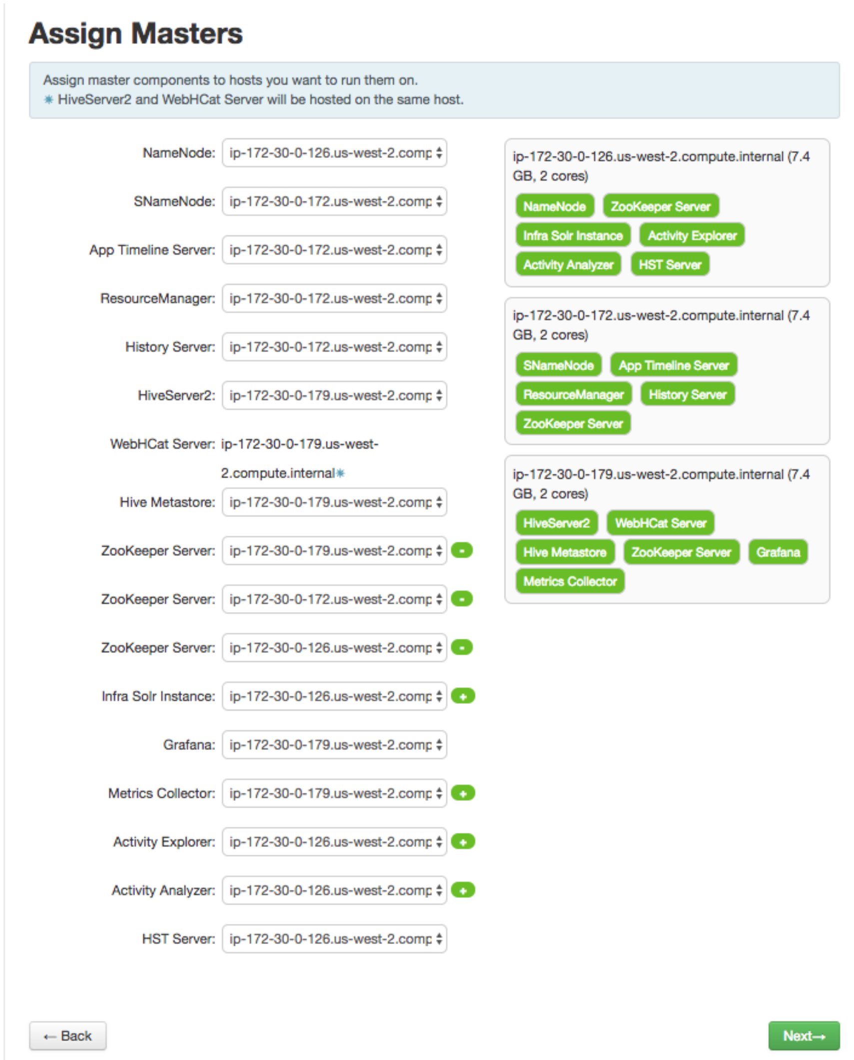
☑ Ambari Metrics	0.1.0	A system for metrics collection that provides storage and retrieval capability for metrics collected from the cluster
☐ Atlas	0.8.0	Atlas Metadata and Governance platform
☐ Kafka	0.10.1	A high-throughput distributed messaging system
☐ Knox	0.12.0	Provides a single point of authentication and access for Apache Hadoop services in a cluster
☐ Log Search	0.5.0	Log aggregation, analysis, and visualization for Ambari managed services. This service is **Technical Preview**.
☑ SmartSense	1.4.2.2.5.2.0-298	SmartSense - Hortonworks SmartSense Tool (HST) helps quickly gather configuration, metrics, logs from common HDP services that aids to quickly troubleshoot support cases and receive cluster-specific recommendations.
☐ Spark	1.6.3	Apache Spark is a fast and general engine for large-scale data processing.
☐ Spark2	2.1.1	Apache Spark is a fast and general engine for large-scale data processing
☐ Zeppelin Notebook	0.7.2	A web-based notebook that enables interactive data analytics. It enables you to make beautiful data-driven, interactive and collaborative documents with SQL, Scala and more.
☐ Druid	0.9.2	A fast column-oriented distributed data store. This service is **Technical Preview**.
☐ Mahout	0.9.0	Project of the Apache Software Foundation to produce free implementations of distributed or otherwise scalable machine learning algorithms focused primarily in the areas of collaborative filtering, clustering and classification
☑ Slider	0.92.0	A framework for deploying, managing and monitoring existing distributed applications on YARN.

← Back Next →

13 . The Assign Masters window is where you would choose which cluster nodes would run which master service components.

The purpose of this step is to evenly balance the master services' workloads across multiple systems. Suggestion - Move HiveServer2 and Hive Metastore to the node that just has ZooKeeper to balance the master nodes.

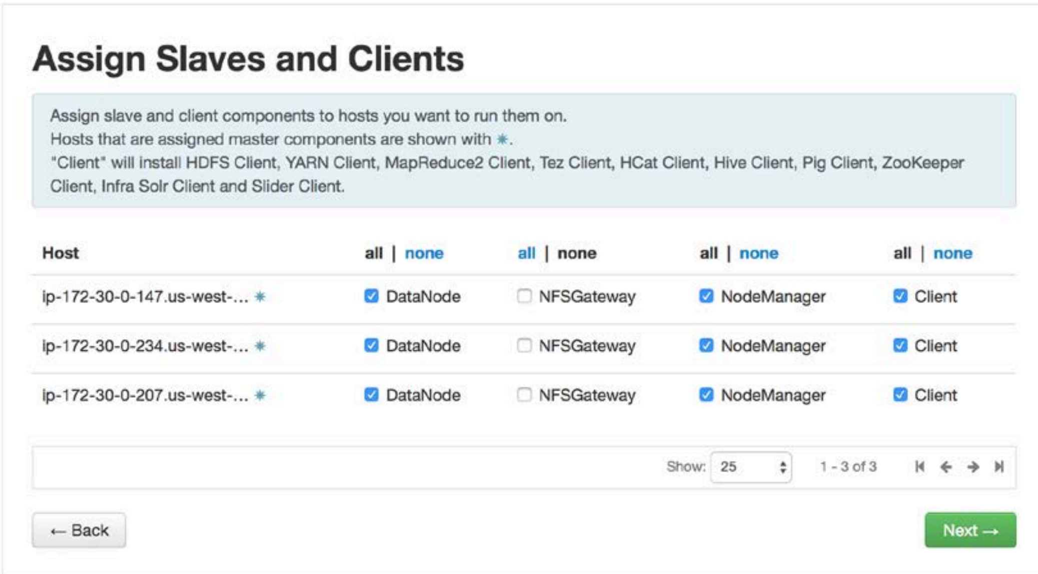
View the window and click **Next** to continue.



14 . The Assign Slaves and Clients window is where you would choose which cluster nodes will run which worker processes.

Worker processes perform most of the data processing in a cluster. The more nodes that you have running worker processes, the more work that can be simultaneously accomplished.

Make sure you select DataNode, NodeManager and Client for all Host listed. See example below.

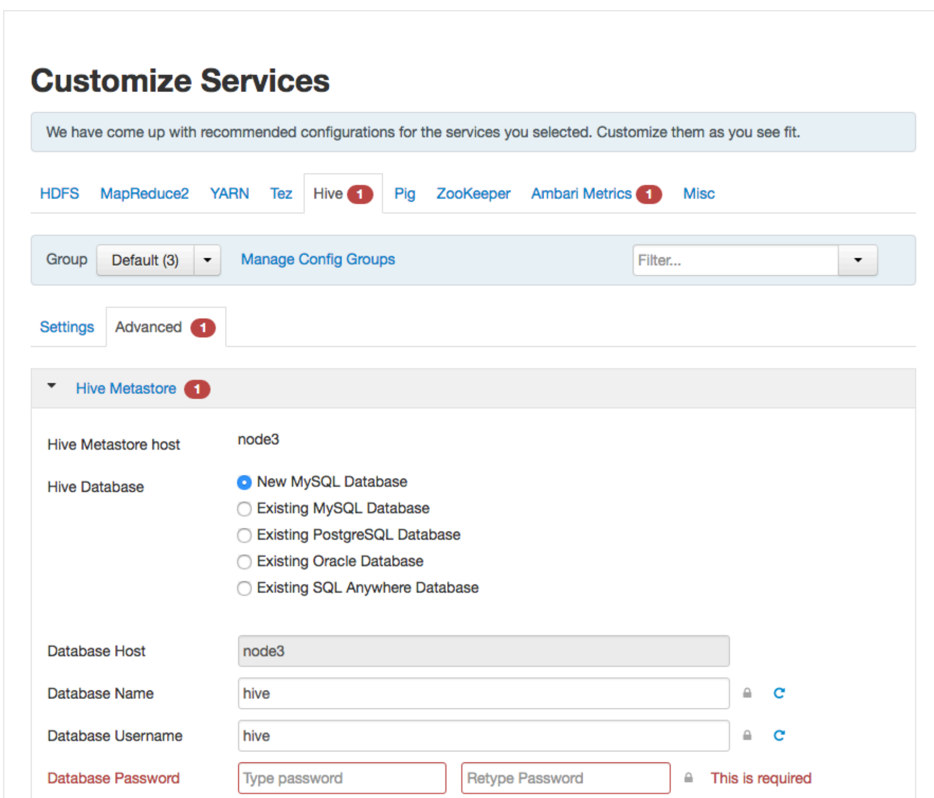


Click **Next** to continue.

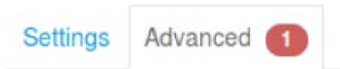
The Customize Services window enables you to configure or customize cluster service property settings during installation.

If there are any property settings that must be configured prior to installation, they are noted by red-colored alert icons. Notice that there is an alert icon next to **Hive** and Ambari Metrics.

Click **Hive** to open the Hive page.



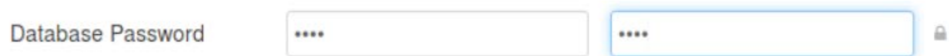
18 . Notice that there is an alert icon on the Hive **Advanced** tab.
Click the **Advanced** tab.



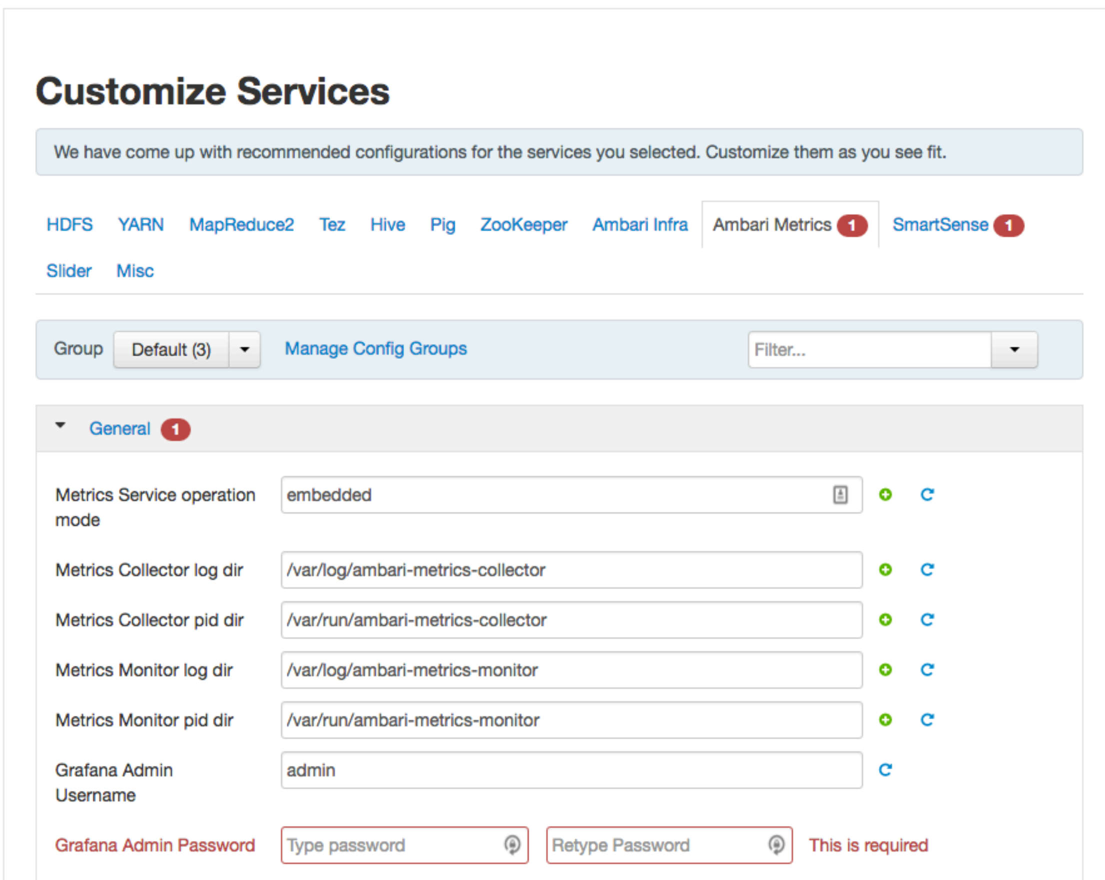
19 . Scroll down in the window until you find the **Database Password** property.

Notice that it is displayed in red. Type and retype `hive` as the database password.

The red lettering will switch to black lettering once the password has been accepted.

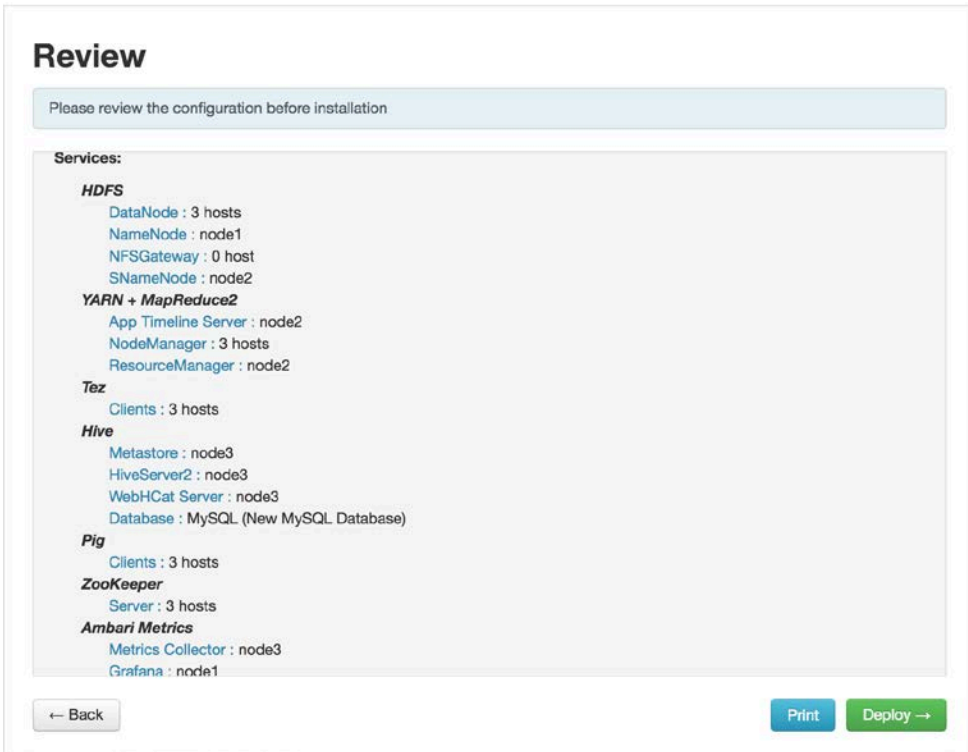


20 . Repeat the same process for Ambari Metrics Alert and enter the Grafana Admin Password that is required. Type and retype `admin` as the password. For SmartSense enter password

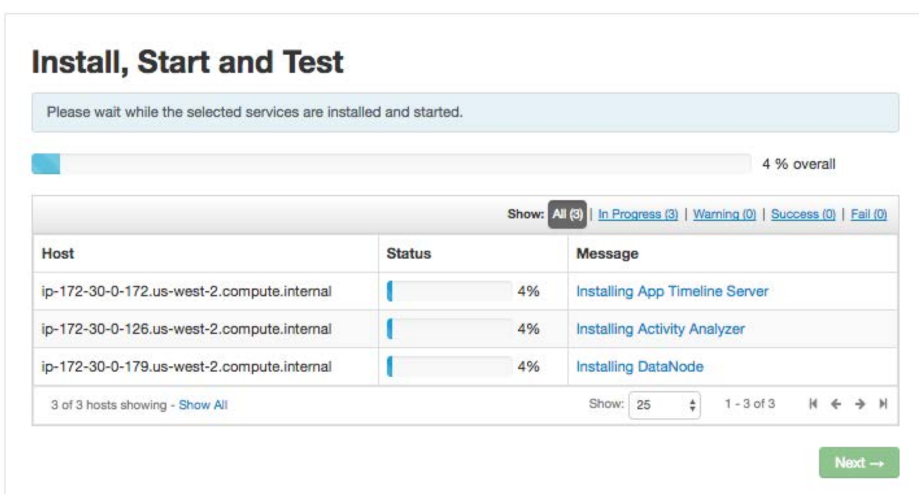


21 . Because there are no more alerts to process, Scroll down in the window and click **Next** to continue.

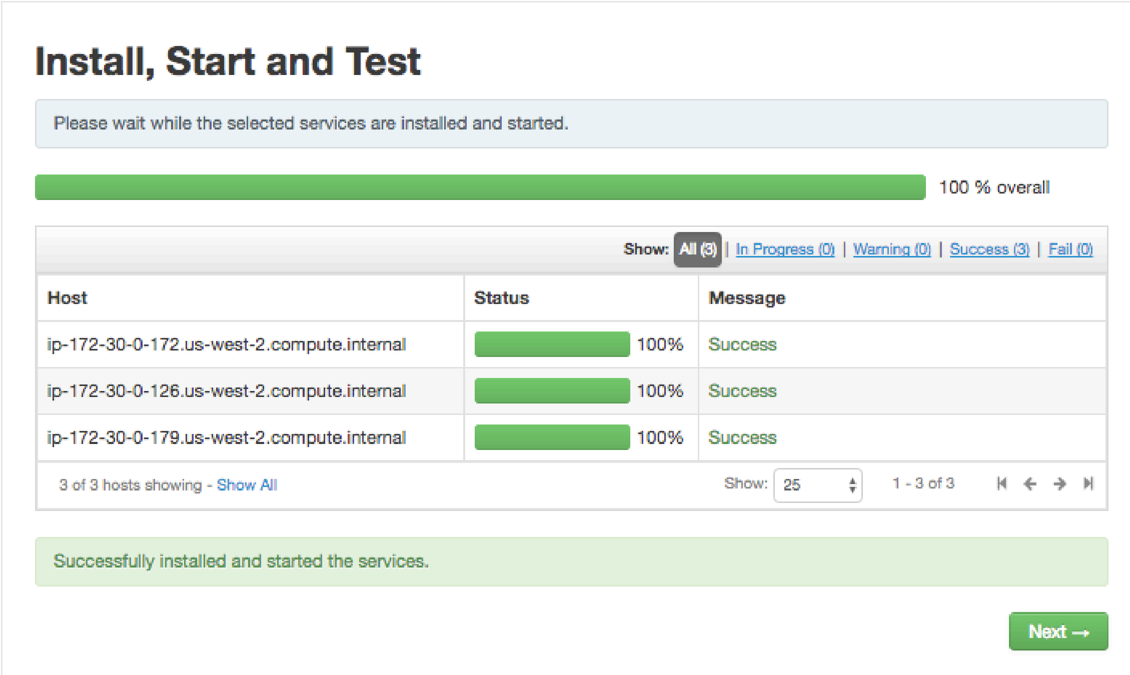
22 . Review your selections for accuracy and then click **Deploy**.



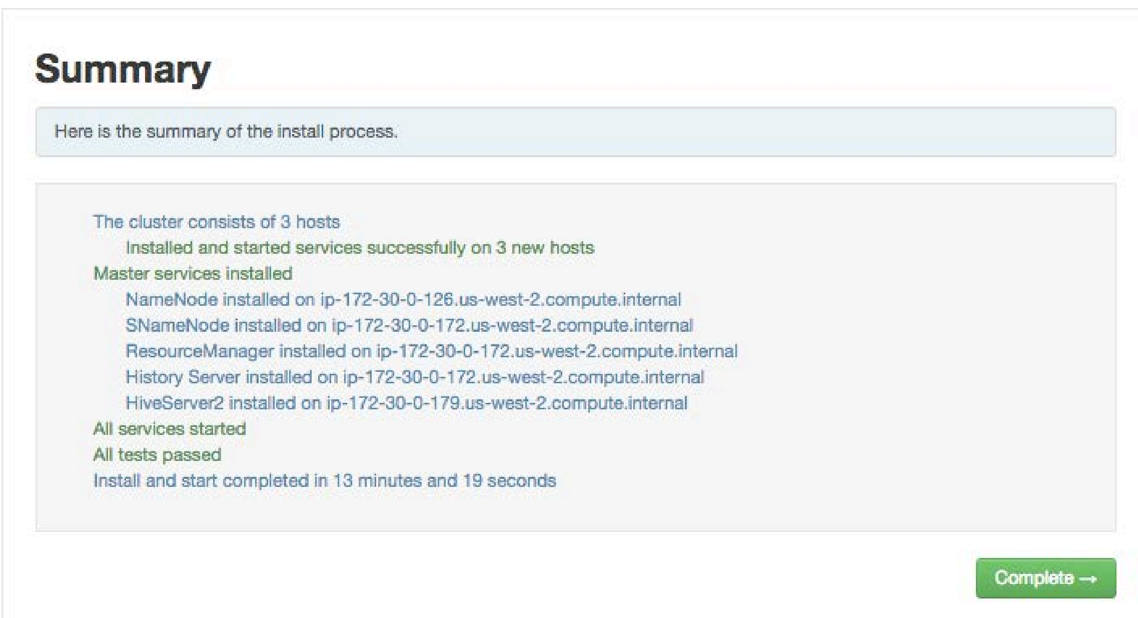
23 . Monitor the installation progress.



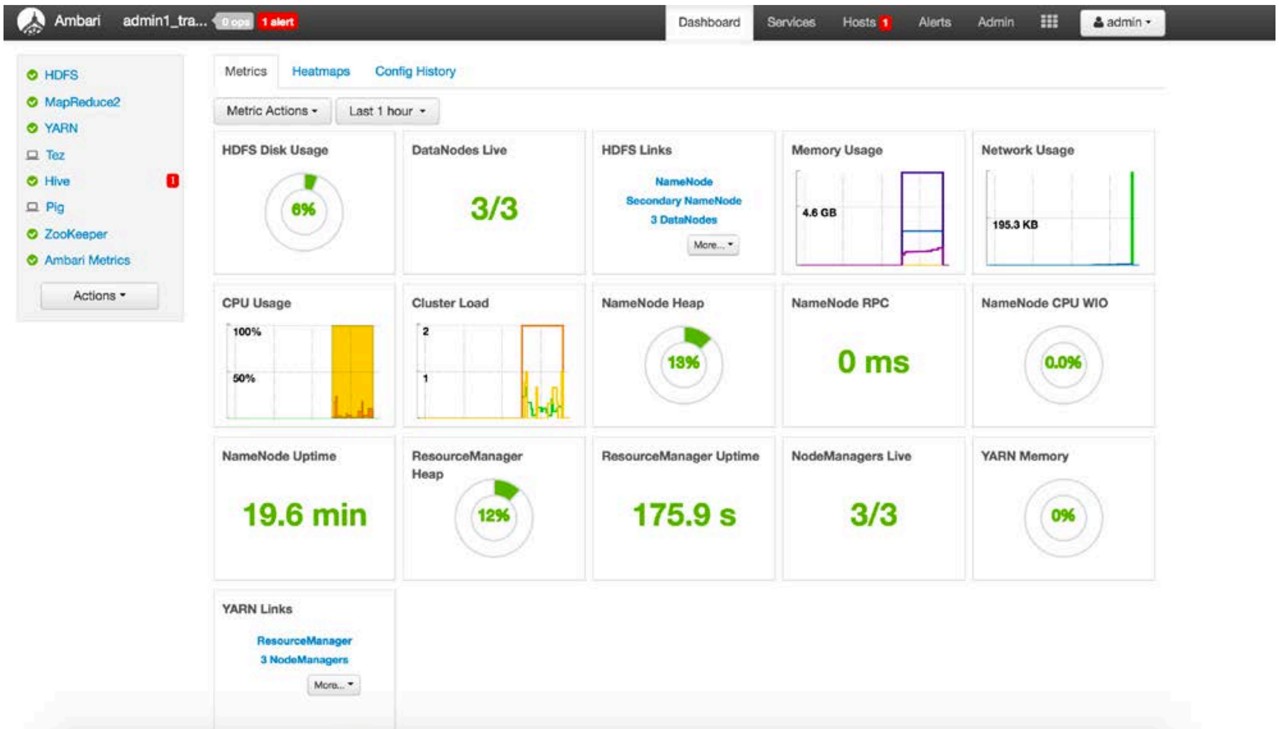
24 . When the progress window indicates that the installation has completed, click **Next**.



25 . Read the Summary window and click **Complete**.



26 . After completing the installation, the Ambari Web UI dashboard is displayed.



Result

You have now:
Installed, configured, and started the Ambari Server; and performed an interactive HDP installation using the AMbari Web UI.

Lab 2: Managing Ambari Users and Groups

About This Lab

Objective:	To manage Ambari users, groups, and permissions in the Ambari Web UI
File locations:	N/A
Successful outcome:	**You will:** View Hadoop service accounts; use the Ambari Web UI to add user accounts; test the functionality of the new user accounts you created; create a new Ambari user group and add accounts and permissions to it; and test the interaction between user permissions and group persmissions.
Before you begin	Connect to your classroom lab environment
Related lesson:	***Managing Ambari Users and Groups***

Viewing Hadoop Service Accounts

View the Hadoop service accounts configured by Ambari during cluster installation.

Which service accounts are configured depends on the services that were installed by Ambari.

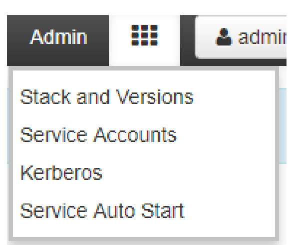
1. Use SSH to login to any cluster node.

2. View the accounts listed in the `/etc/passwd` file.

```
cat /etc/passwd
```

Do you see Hadoop service account names that include hive, zookeeper, ams, ambari, tez, hdfs, and others?

3. In the Ambari Web UI, click the **Admin** menu (not **admin**) and select **Service Accounts**.

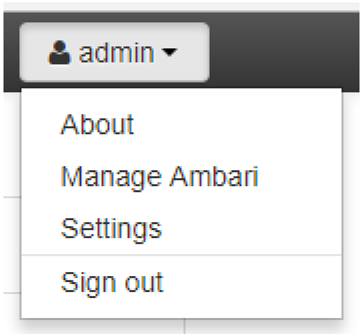


Do you see a list of the same Hadoop service account names that were listed in the `/etc/passwd` file?

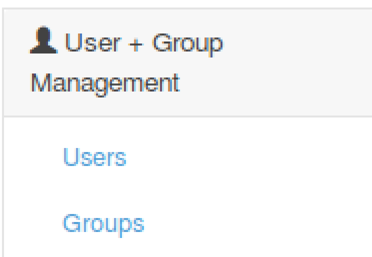
Adding Ambari User Accounts

Use the Ambari Web UI to add four new Ambari user accounts.

1. To add a new Ambari user account, select **Manage Ambari** from the **admin** menu.



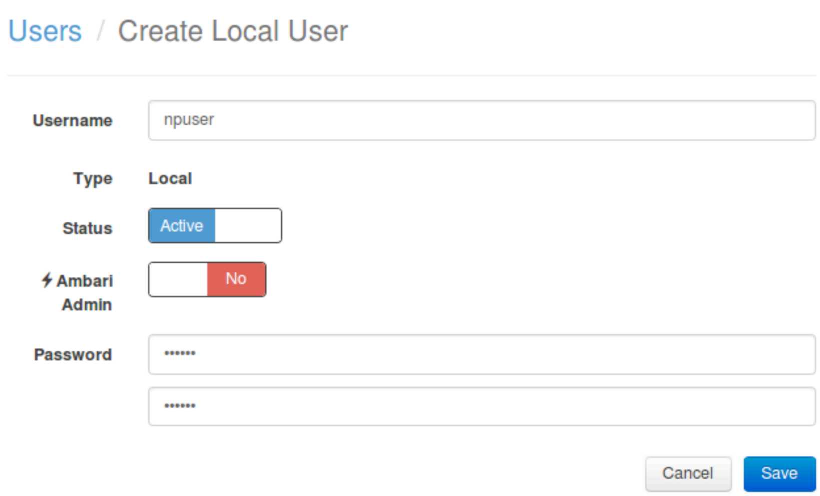
2. Click **Users** to open the Users pane.



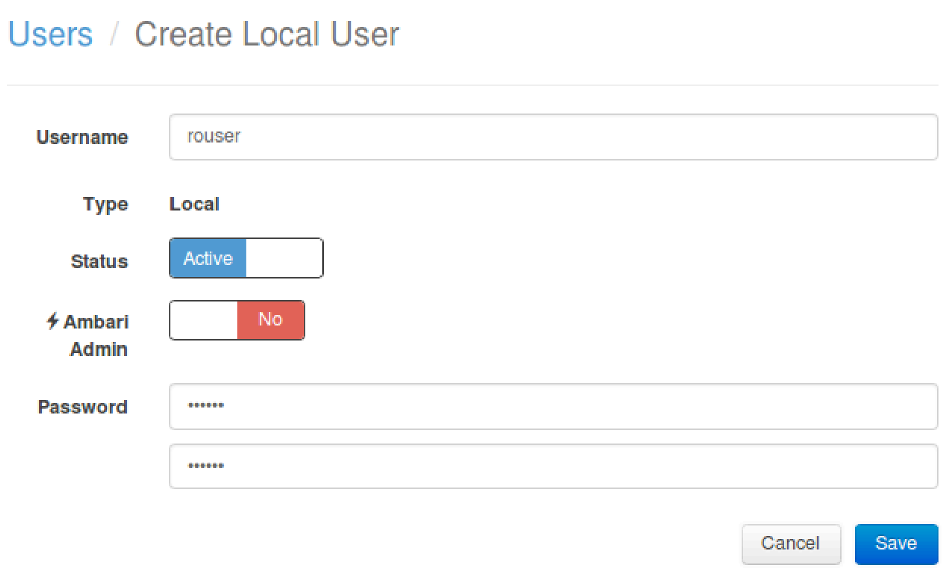
Ambari displays the current list of user accounts. The list should include only the admin user account.

3. Click *+Create Local User*.

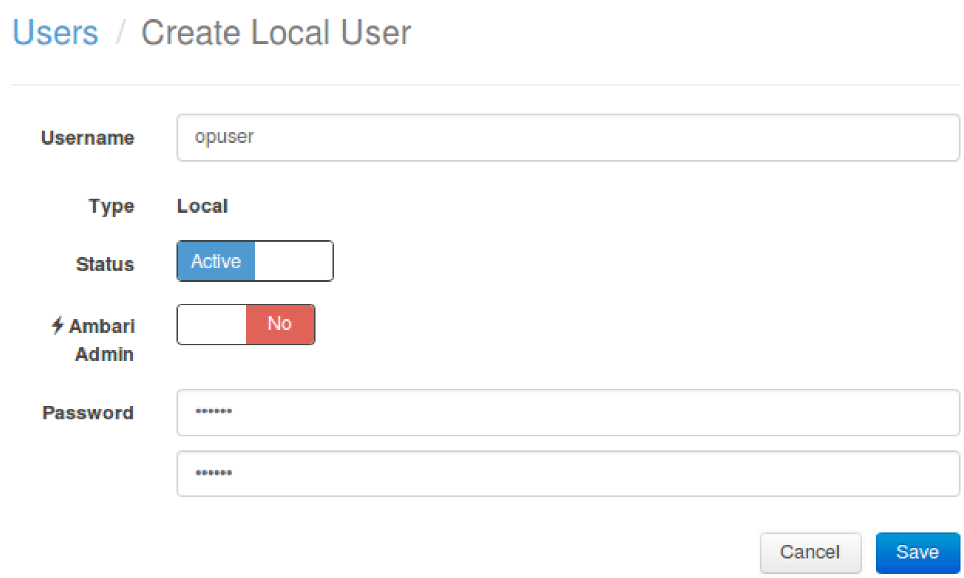
4. Create a local user account named `npuser`. Assign this user account the password `npuser`. Click **Save** when you are finished.



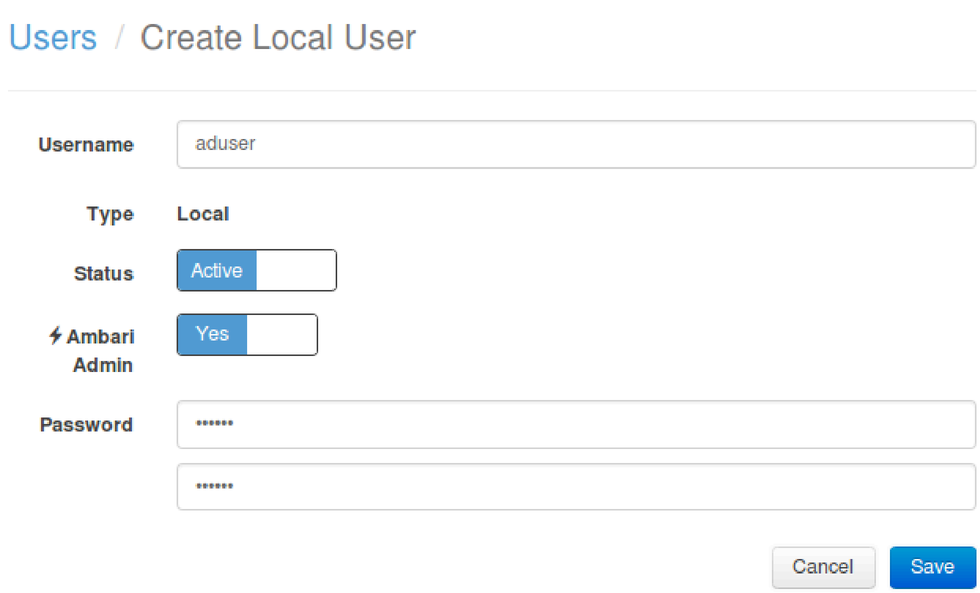
5. Create another local user account named `rouser` and assign it the password `rouser`. Click **Save** when you are finished.



Create another local user account named `opuser` and assign it the password `opuser`. Click **Save** when you are finished.



8. Create another local user account named `aduser` and assign it the password `aduser`. This account will be assigned Ambari Admin permission as part of the creation process when you configure Ambari Admin to **Yes**. Click **Save** when you are finished.



Assign User Account Roles

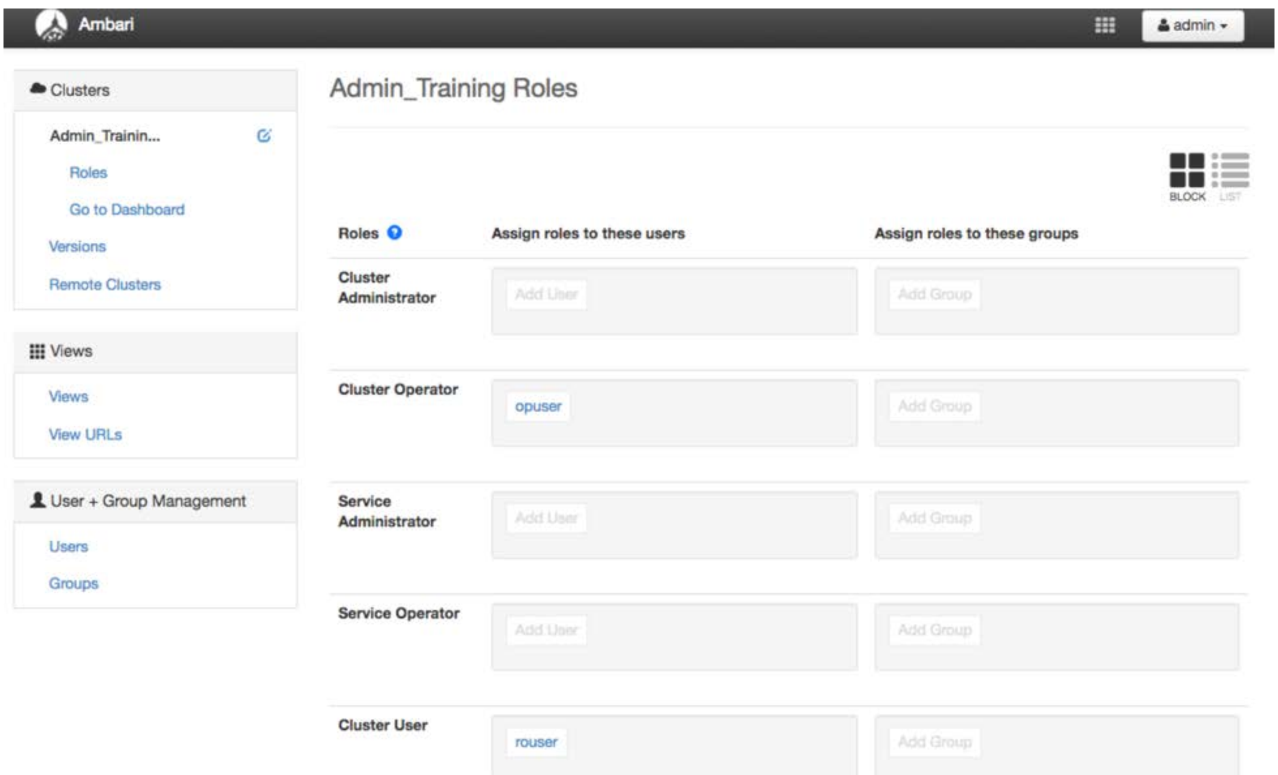
Assign Ambari roles to user accounts.

1. Click **Roles**.

2. In the **Cluster Operator** section beneath **Assign roles to these users c**lick **Add User**, then click **New** and type the user account `opuser`.

   In the **Cluster User** section beneath **Assign roles to these users**, click **Add User,** then click **New** and type the user account `rouser`.

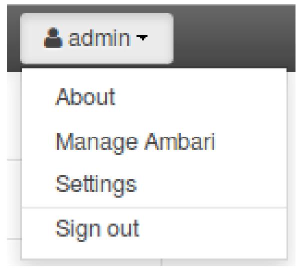
   Click the blue check box icons to confirm and finish adding both users.



Testing User Account Functionality

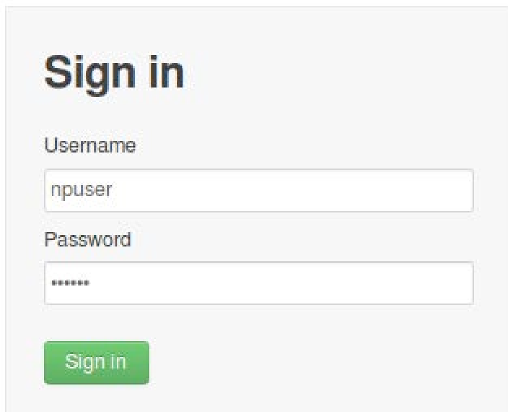
Test the functionality of the four new Ambari user accounts.

1. Log out of the Ambari Web UI by selecting **Sign out** from the **admin** menu.



2. Sign in to the Ambari Web UI using the user name `npuser` and the password `npuser`.

   The npuser is known to Ambari but has no permissions in Ambari.

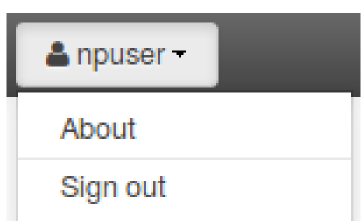


   Do you see the options to navigate to the **Dashboard**, **Services**, **Hosts**, or **Alerts** pages?

   Do you see the **Ambari Views** option?

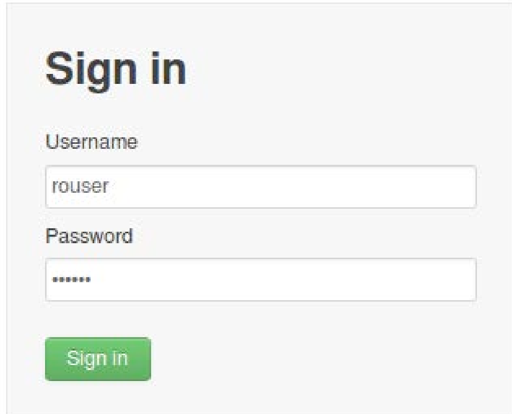
   Are there any Views assigned to the npuser?

3. Log out of the Ambari Web UI by selecting **Sign out** from the **npuser** menu.



4. Sign in to the Ambari Web UI using the user name `rouser` and the password `rouser`.

   The rouser account has been assigned Cluster User role in Ambari.



Do you see the option to navigate to the **Dashboard**, **Services**, **Hosts**, or **Alerts** pages?

Do you see the **Ambari Views** option?

Are there any Views assigned to the rouser?

5. While logged in as rouser, click **Dashboard**.

   Are you able to view Dashboard information?

6. While logged in as rouser, select **HDFS** from the **Services** menu.

   Are you able to view HDFS service information on the Summary page?

   On the **Summary** page, do you see the **Actions** and **Service Actions** buttons?

   Why not?

7. While logged in as rouser, click **Hosts** to open the **Hosts** page.

   Are you able to view host information on the **Hosts** page?

   On the **Hosts** page, do you see the **Actions** button?

   Why not?

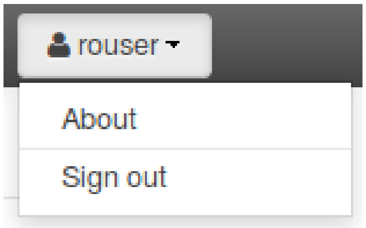
8. While logged in as rouser, click **Alerts** to open the **Alerts** page.

   Are you able to view host information on the **Alerts** page?

   On the **Alerts** page, do you see the **Actions** button?

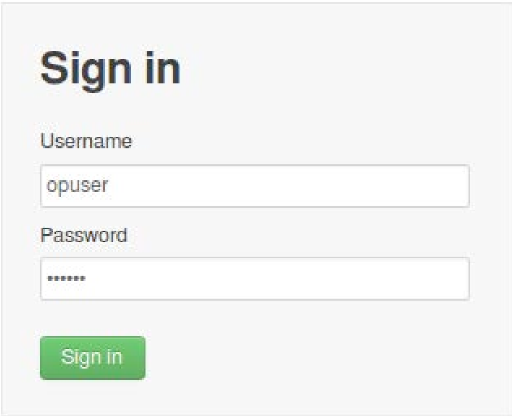
   Why not?

9. Log out of the Ambari Web UI by selecting **Sign out** from the **rouser** menu.



10. Sign in to the Ambari Web UI using the user name `opuser` and the password `opuser`.

    The opuser has been assigned Cluster Operator role in Ambari.



Do you see the option to navigate to the **Dashboard**, **Services**, **Hosts**, or **Alerts**?

Do you see the **Ambari Views** option?

Are there any Views assigned to the opuser?

Are you also able to see the **Admin** page listed at the top of the browser window?

This should be the first time that this page is available as an option.

11 . While logged in as opuser, click **Dashboard**.

Are you able to view Dashboard information?

12 . While logged in as opuser, select **HDFS** from the **Services** menu.

Are you able to view HDFS service information on the **Summary** page?

On the **Summary** page, do you see the **Actions** and **Service Actions** buttons?

Why?

13 . While logged in as opuser, click **Hosts** to open the **Hosts** page.

Are you able to view host information on the **Hosts** page?

On the **Hosts** page, do you see the **Actions** button?

Why?

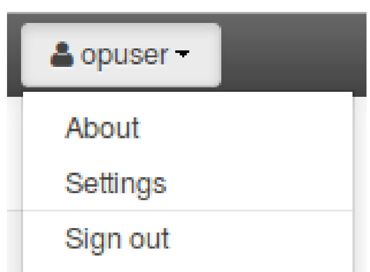
14 . While logged in as opuser, click **Alerts** to open the **Alerts** page.

Are you able to view alert information on the **Alerts** page?

On the **Alerts** page, do you see the **Actions** button?

Why?

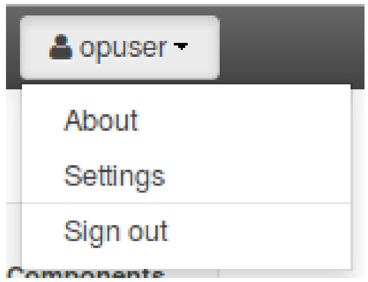
15 . Open the **opuser** menu.



What are the available choices?

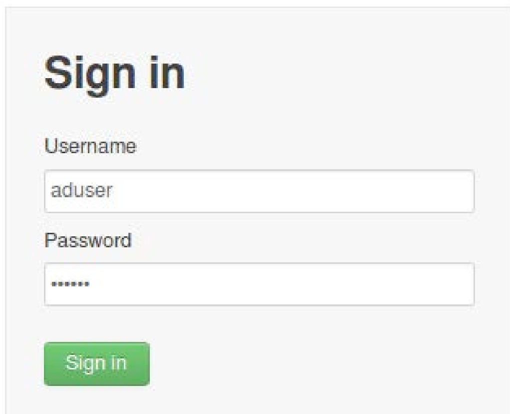
Specifically, notice that there is no **Manage Ambari** menu option.

16 . Log out of the Ambari Web UI by selecting **Sign out** from the **opuser** menu.



17 . Sign in to the Ambari Web UI using the user name `aduser` and the password `aduser`.

The aduser has been assigned admin permission in Ambari.



Do you see the option to navigate to the **Dashboard**, **Services**, **Hosts**, or **Alerts** pages?

Do you see the **Ambari Views** option?

Are there any Views assigned to the aduser?

Are you able to see the **Admin** page listed at the top of the browser window?

18 . While logged in as aduser, click **Dashboard**.

Are you able to view Dashboard information?

19 . While logged in as aduser, select **HDFS** from the **Services** menu.

Are you able to view HDFS service information on the **Summary** page?

On the **Summary** page, do you see the **Actions** and **Service Actions** buttons?

Why?

20 . While logged in as aduser, click **Hosts** to open the **Hosts** page.

Are you able to view host information on the **Hosts** page?

On the **Hosts** page, do you see the **Actions** button?

Why?

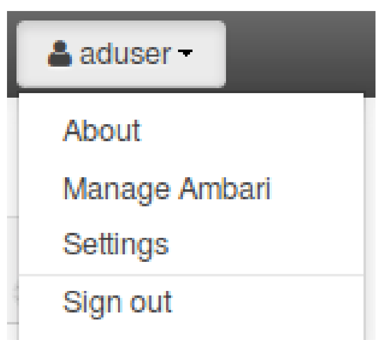
21 . While logged in as aduser, click **Alerts** to open the **Alerts** page.

Are you able to view alert information on the **Alerts** page?

On the **Alerts** page, do you see the **Actions** button?

Why?

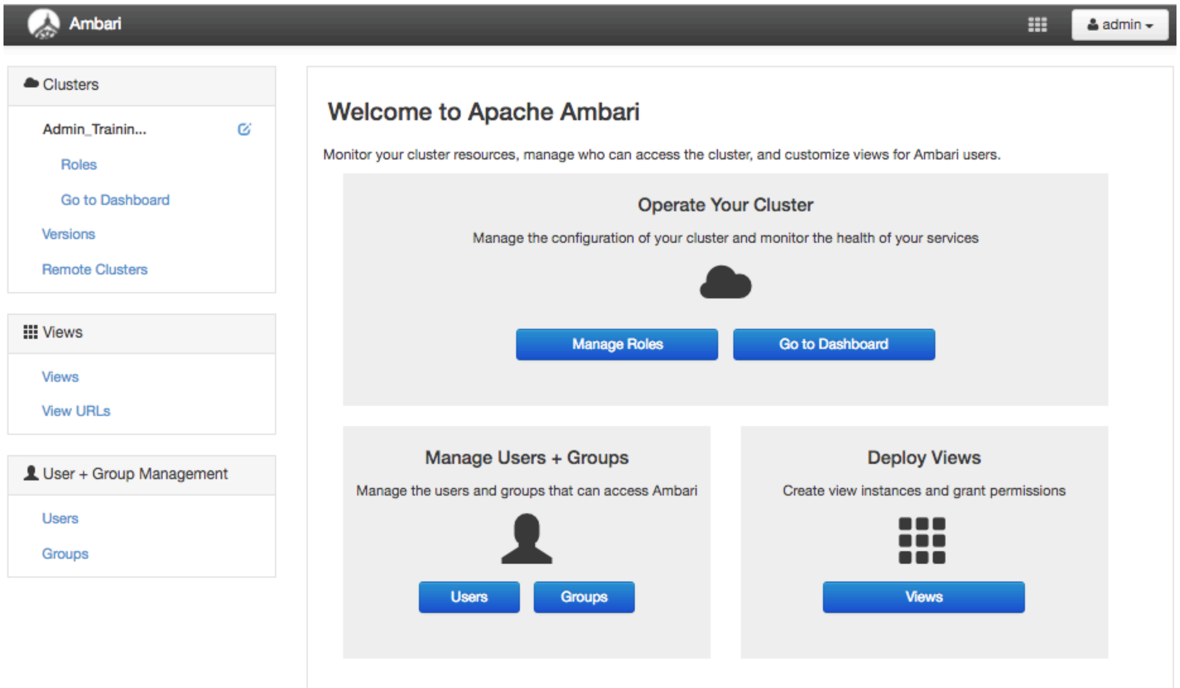
22 . Open the **aduser** menu.



Notice the **Manage Ambari** menu option, which is only available to user accounts with admin permission.

23 . Select **Manage Ambari** from the **aduser** menu.

Do you see the following?

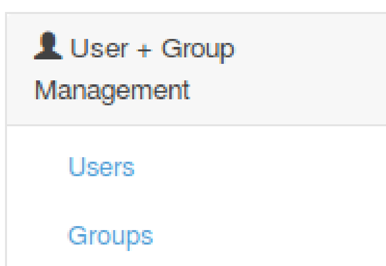


24 . Remain logged in as the aduser.

Creating New Groups

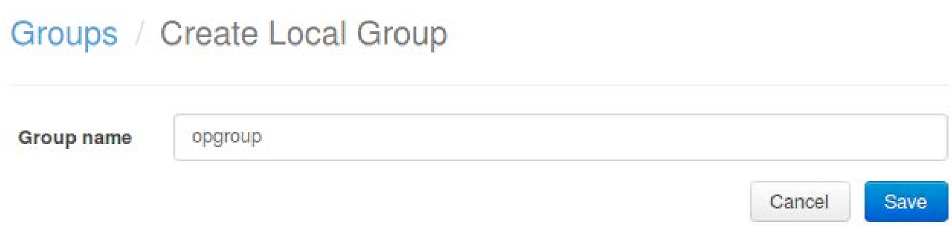
Create a new Ambari group named opgroup, add the npuser, rouser, opuser, and aduser user accounts to the opgroup group, and assign it the group operator permission.

1 . While still logged in as aduser, click **Groups** to create a new Ambari group.



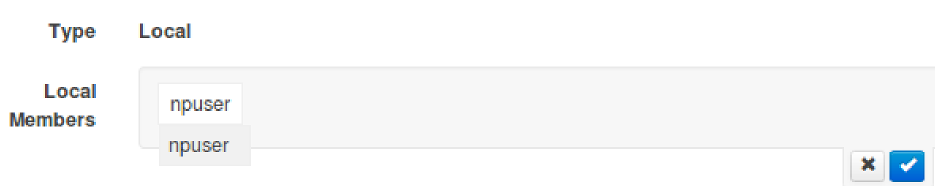
2 . Click **+Create Local Group** to create a new local group.

3. Type `opgroup` as the name and click **Save**.



4. Click **opgroup** to add group members.

5. Click **Add User**, click **New** and type `npuser`, and then click the blue check box to add the user account to the group.

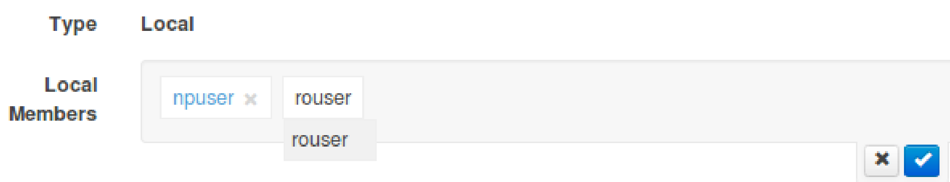


6 . Float the mouse pointer over the newly added **npuser** and notice the pencil icon that appears to the right.

Click the pencil icon to edit the group and add additional users.

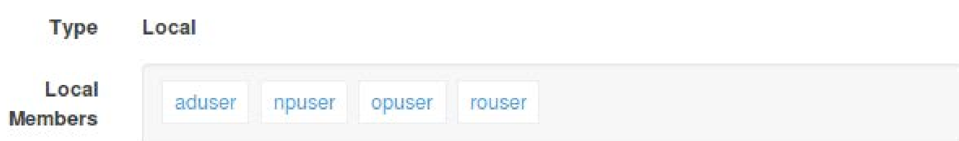
Clicking the pencil icon creates a **New** text box where you are able to type the name of another user.

Type `rouser` and click the blue check box to add the user.

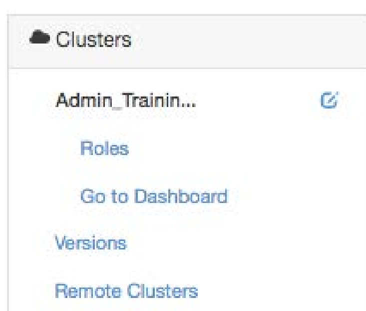


7 . Repeat the previous step to add opuser and aduser to opgroup.

All four user accounts should be members of opgroup when you are finished.

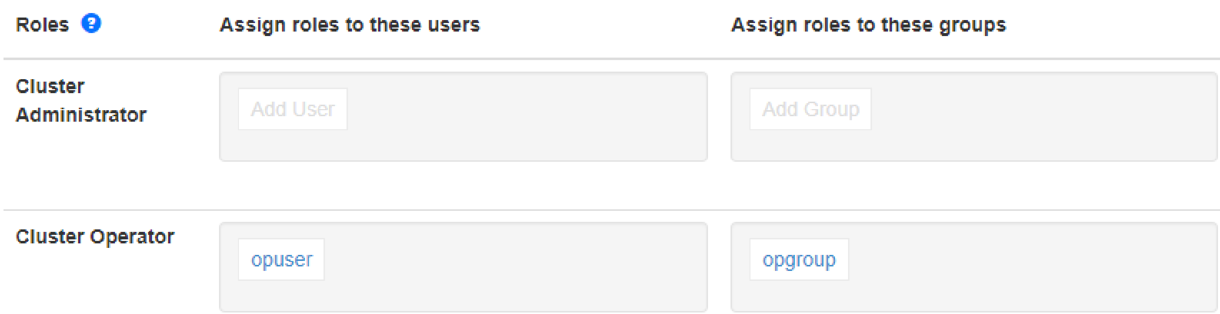


8 . Click **Roles** to assign roles to the opgroup group.



9 . In the **Cluster Operator** roles section, add `opgroup` beneath **Assign roles to these groups**.

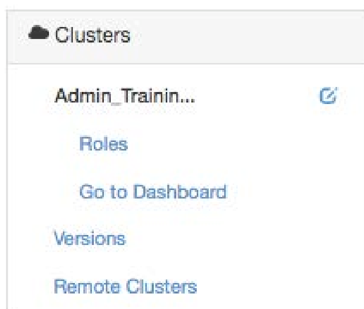
Click the blue check box to add the group.



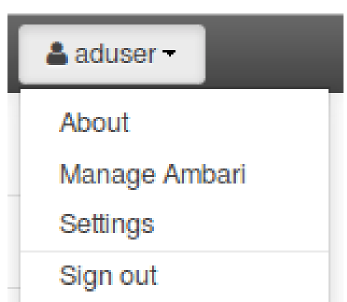
Testing Interaction Between User and Group Permissions

Test the interaction between specific user permissions and group permissions.

1. While still logged in as aduser, click **Go to Dashboard** to return to the main Ambari Web UI window.



2. Click to open the **aduser** menu.



Does the aduser account still have the option on this menu to **Manage Ambari** users, groups, and permissions?

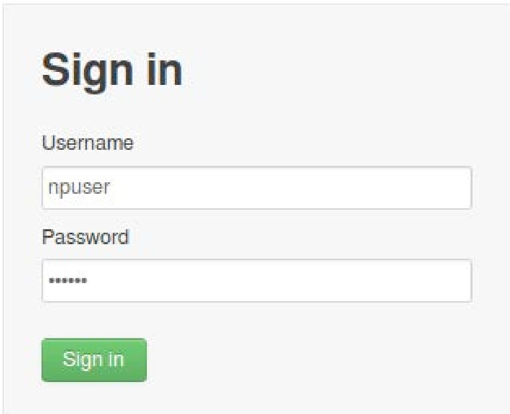
Although the aduser account was added as a member of the opgroup, which has only operator permission, has the aduser account lost its admin permission?

3 . Sign out of the aduser account.

4 . Sign in to the Ambari Web UI using the user name `npuser` and the password `npuser`.

The npuser is known to Ambari but has no permissions in Ambari.

However, the npuser account was added as a member of the opgroup group, which has operator permissions.



What do you see in the browser window?

Do you see the option to navigate to the **Dashboard**, **Services**, **Hosts**, or **Alerts** pages?

Do you see the **Ambari Views** option?

Does the npuser account now appear to have some permissions in Ambari?

5 . While logged in as npuser, select **HDFS** from the **Services** menu.

Are you able to view HDFS service information on the **Summary** page?

On the **Summary** page, do you see the **Actions** and **Service Actions** buttons?

Why?

Would this indicate that the npuser account now has operator permission?

6 . Sign out of the npuser account.

Result

You have now:
Viewed Hadoop service accounts; used the Ambari Web UI to add user accounts; tested the functionality of the new user accounts you created; created a new Ambari user group and added accounts and permissions to it; and tested the interaction between user permissions and group permissions.

Lab 3: Managing Hadoop Services

About This Lab

Objective:	To use the Ambari Web UI to manage Hadoop services
File locations:	N/A
Successful outcome:	**You will:** View the Hadoop service configuration files that were installed with Ambari; tour Ambari Web UI service management and configuration features; change the HDFS service configuration using the Ambari Web UI; create an Ambari Configuration Group; and download HDFS client configuration files.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Managing Hadoop Services***

Viewing Hadoop Service Configuration Files

View the Hadoop service configuration files that were installed by Ambari.

1. Use SSH to login to any cluster node.

2. List the core Hadoop service configuration files in the `/etc/hadoop/conf` directory.

```
sudo su -
ls /etc/hadoop/conf
```

Notice specifically the `core-site.xml`, `hdfs-site.xml`, and `mapred-site.xml` files. These files are some of the more commonly updated Hadoop configuration files.

Do you see service configuration files for Hive, ZooKeeper, AMS (Ambari Metrics System), Ambari, or Tez? *You should not.*

3. List the names of the configuration files for other common Hadoop services.

```
ls /etc/ambari-server/conf #exists only on the Ambari Server
ls /etc/ambari-agent/conf
ls /etc/hive/conf
ls /etc/pig/conf
ls /etc/zookeeper/conf
```

4 . The next lab step will use the *jar* command to display the contents of a `.jar` file. The *jar* command is not typically specified in the shell `PATH` environmental variables. To mitigate this, run the following command:

```
export PATH=$PATH:/usr/java/default/bin
```

5 . Use the *jar* command to display the name of the `hdfs-default.xml` file, which is one of the default configuration files installed as part of core Hadoop. The default files are typically installed as part of Java JAR files.

```
jar tvf /usr/hdp/<VERSION NUMBER>/hadoop-hdfs/hadoop-hdfs.jar | grep xml
```

Ambari Web UI Management and Configuration Features

Tour the Ambari Web UI service management and configuration features.

1 . Open the browser and connect to the Ambari Server at the URL

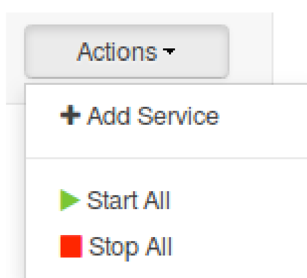
http://<Ambari Server Hostname>:8080

2 . Login to the Ambari Web UI using the user name `admin` and the password `BadPass#1`.

3 . Click **Services** in the Ambari Web UI.

The list of installed services appears on the left side of the **Services** page and includes such services has HDFS, MapReduce2, YARN, and others.

4 . Click to open and view the **Actions** menu button below the list of installed cluster services.

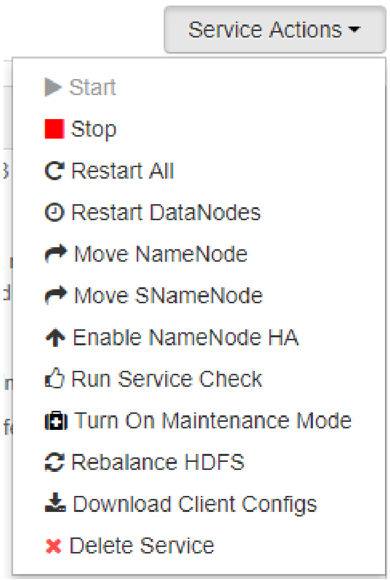


Notice that you can install and configure new cluster services, as well as start or stop *all* currently installed cluster services.

To save time during this lab exercise, you will *not* take the 5-10 minutes to stop and then start all cluster services.

5. Ensure that the **HDFS** service is selected on the **Services** page.

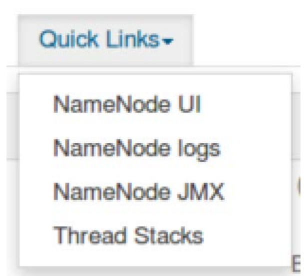
6. Click to open and view the **Service Actions** menu button in the top right-side corner of the **Services** page.



Notice that the actions listed on the **Service Actions** menu button are specific to the currently selected service, in this case HDFS.

Many of the actions listed here are performed in this and other lab exercises.

7. With the **HDFS** service still selected, click **Quick Links** and select **NameNode UI**.



The NameNode UI Web interface opens on another browser tab.

Not all services include their own Web interfaces, but if they do, they can be accessed from the Ambari **Quick Links** menu when that service is selected on the **Services** page.

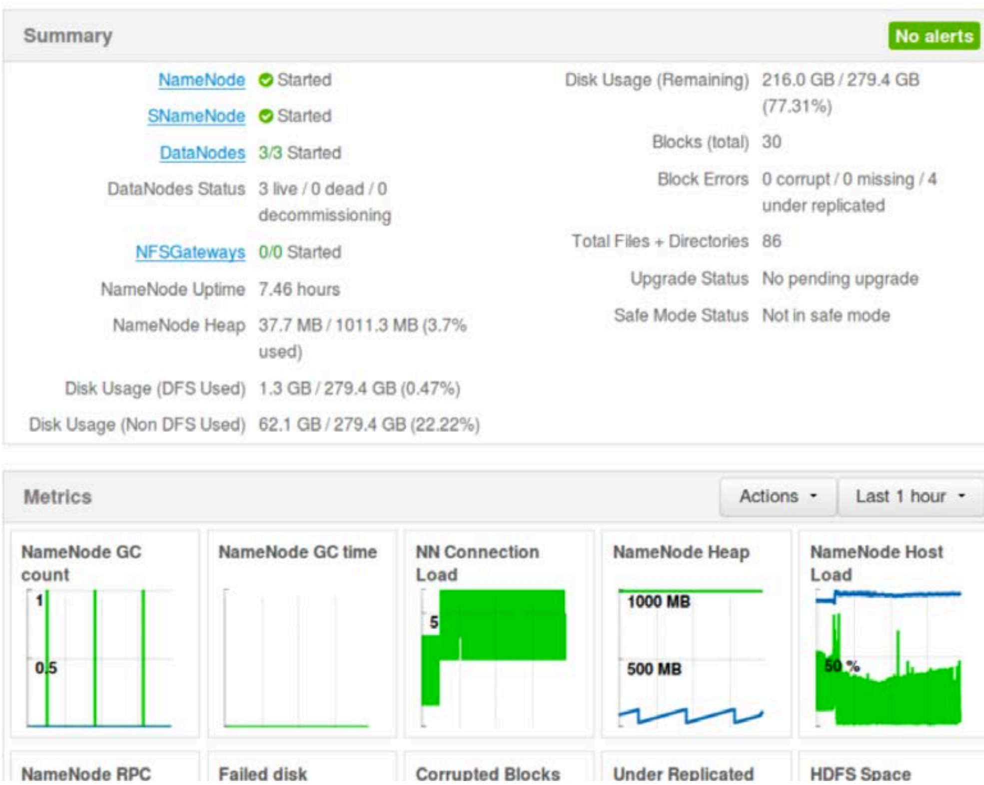
The NameNode UI is described and used in another lesson.

8 . Close the NameNode UI browser tab.

With the **HDFS** service still selected, view the types of information available on the **Summary** page.

The types of information available will vary with each cluster's specific configuration.

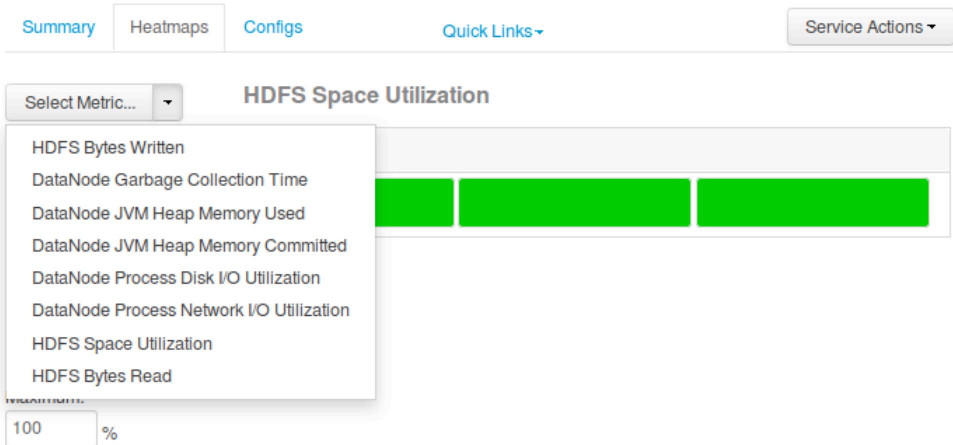
The partial screen capture below might be different from the HDFS information displayed for your lab cluster.



9 . With the **HDFS** service still selected, view the types of metric information available on the **Heatmaps** page.

The types of information available will vary with each cluster's specific configuration.

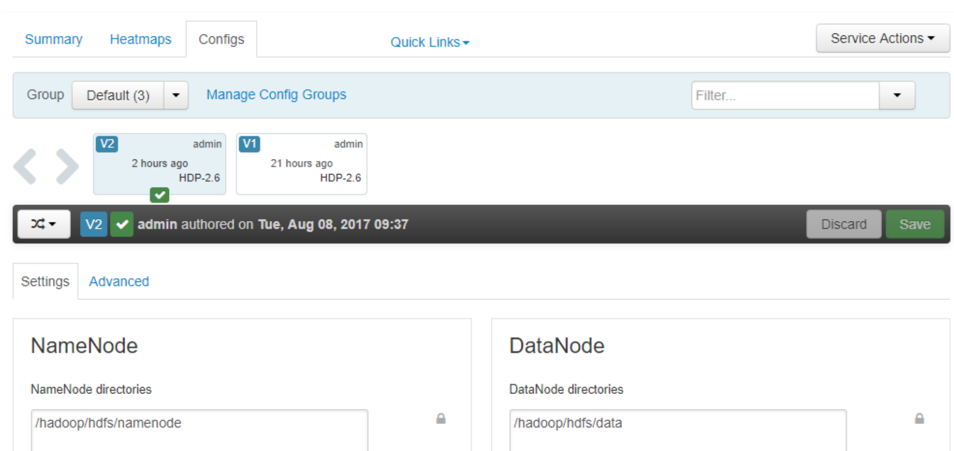
The partial screen capture below could be different from the HDFS information displayed for your lab cluster.



Changing HDFS Service Configuration

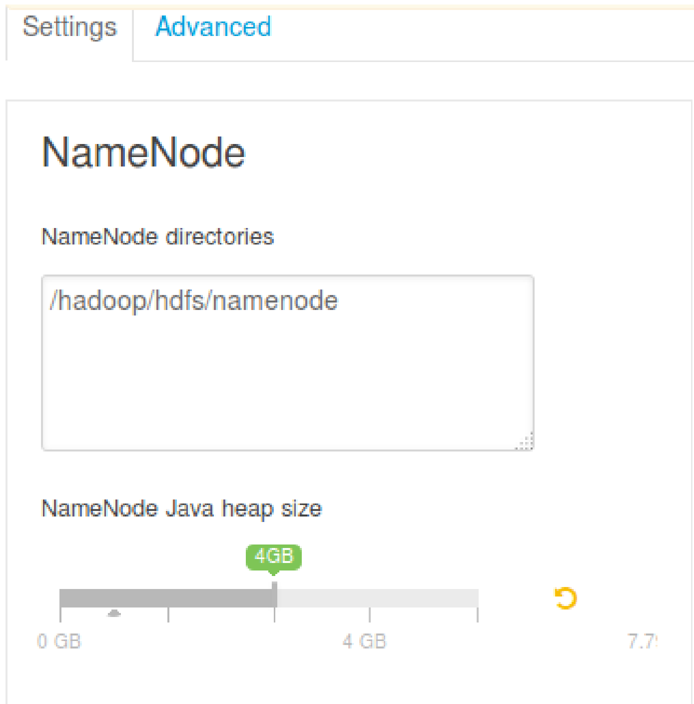
Change the HDFS service configuration using the Ambari Web UI.

1 . With the **HDFS** service still selected, click the **Configs** tab.



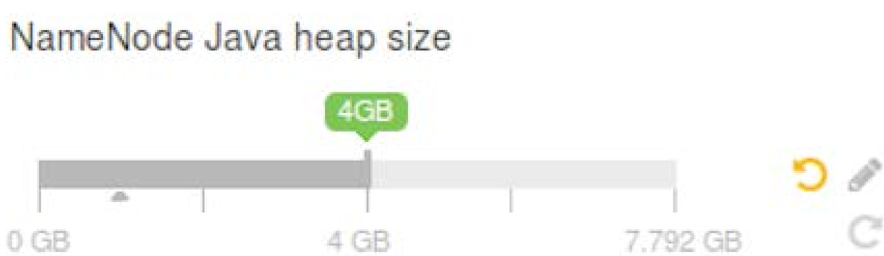
2 . On the **Settings** tab, scroll down to find and change the **NameNode Java heap size** setting. Change it by dragging the slider bar to **4GB**. Do *not* save the change.

NOTE: This is not the size recommended by Ambari Guided Configuration for the current cluster, so you will change this setting back in the next lab step.

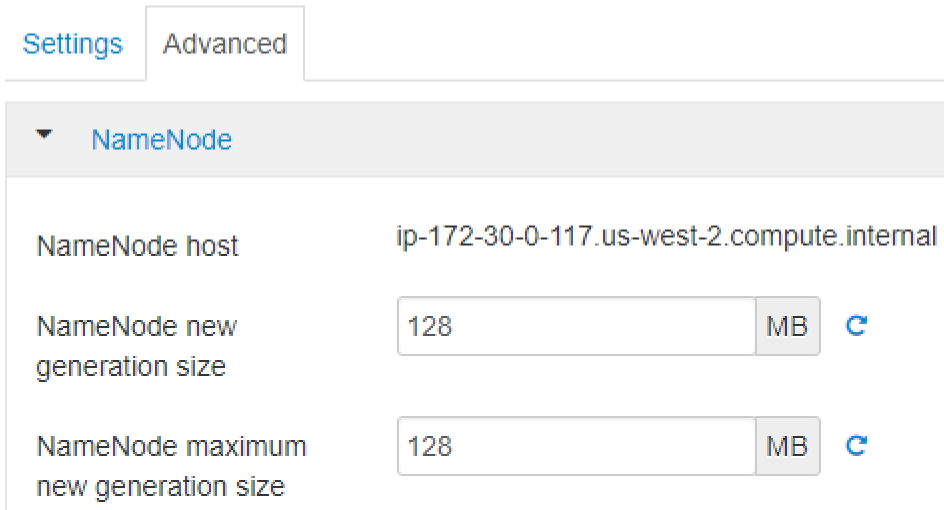


3 . Use Ambari Guided Configuration to return the **NameNode Java heap size** back to its recommended setting.

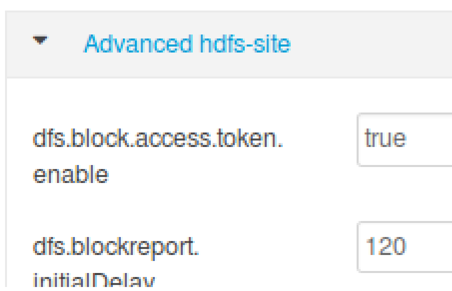
Click the gray-colored, clock-wise circular arrow icon to have Guided Configuration return the heap size to the recommended value.



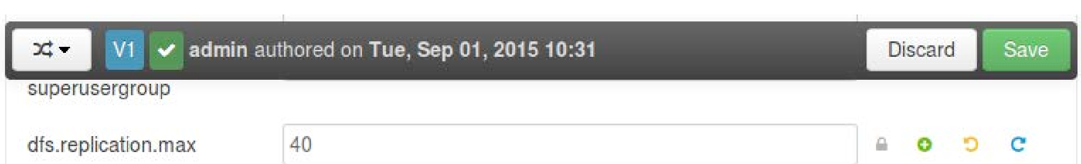
4 . With the **HDFS** service still selected, click the **Advanced** tab.



5 . Scroll down to find and expand the **Advanced hdfs-site** section.

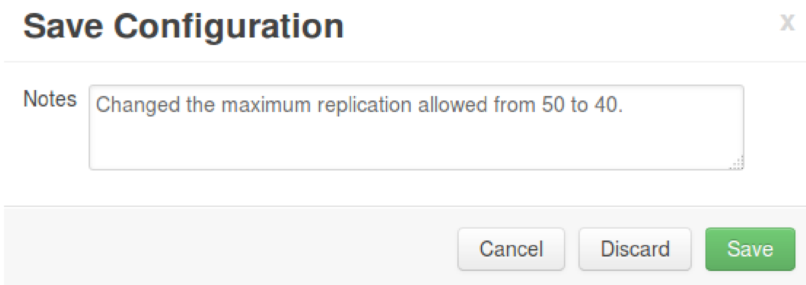


Scroll down to find the parameter `dfs.replication.max` and change it from `50` to `40`, and then click **Save**.



6 . When asked to save the configuration, you can optionally enter Notes and then click **Save** again.

Then click **OK** in the confirmation window.

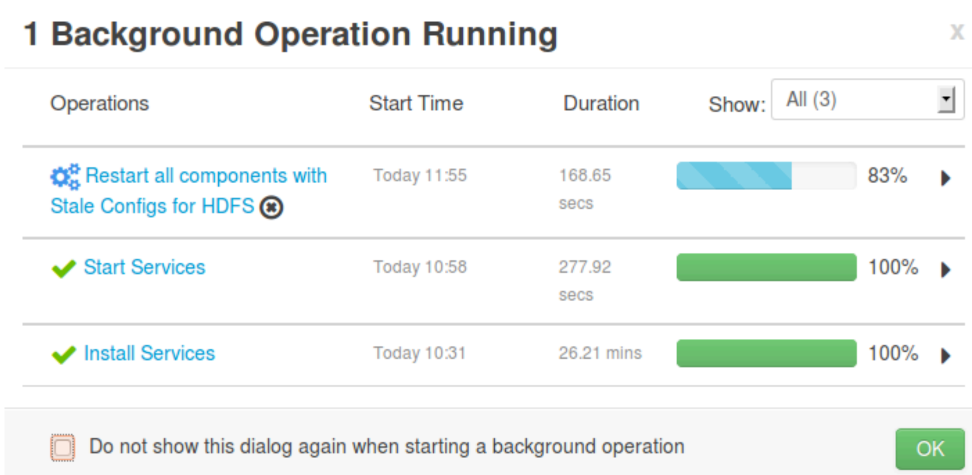


7. At the notification to restart HDFS (and MapReduce2 and YARN), click **Restart** and select **Restart All Affected**.

8. In the confirmation window, click **Confirm Restart All**.

   A Background Operation Running window opens and displays the progress of the restart.

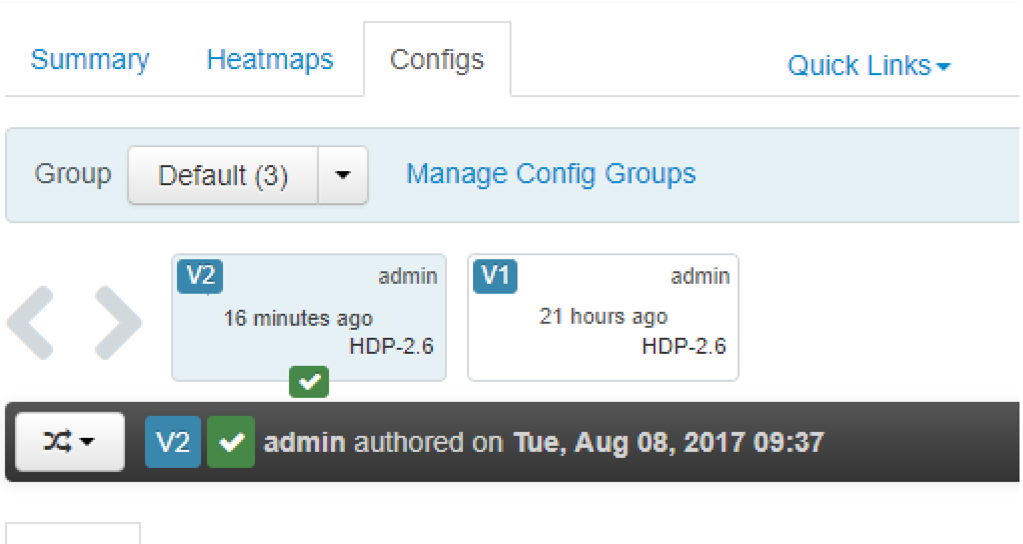
   Click **OK** to dismiss this window once HDFS has been restarted.



9. Ambari displays a restart icon next to any other service or services that must also be restarted.

   Select each service that Ambari indicates must be restarted and then follow the same procedure that you just used to restart HDFS.

10. Reselect the **HDFS** service and ensure that you have clicked its **Configs** tab.



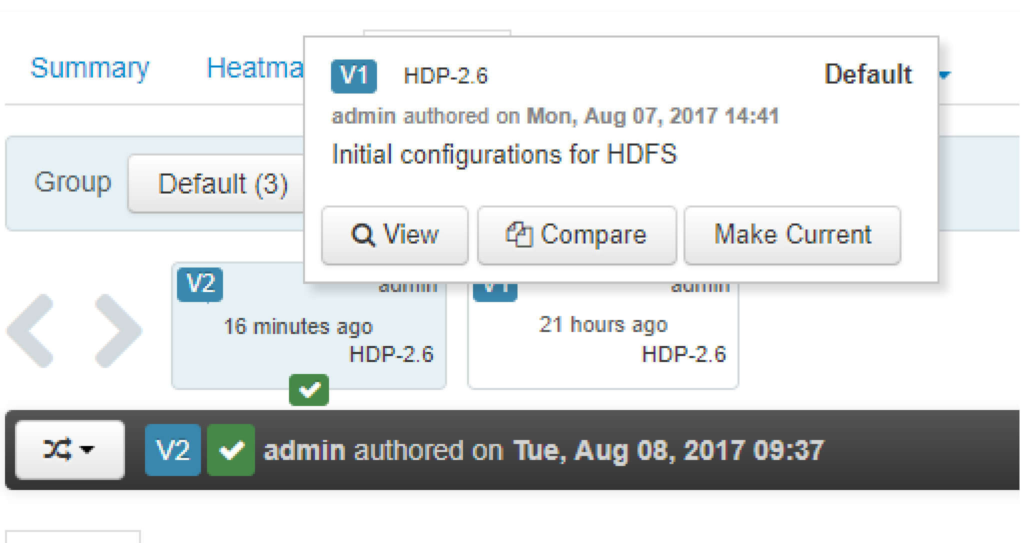
You should now have a configuration change history for HDFS.

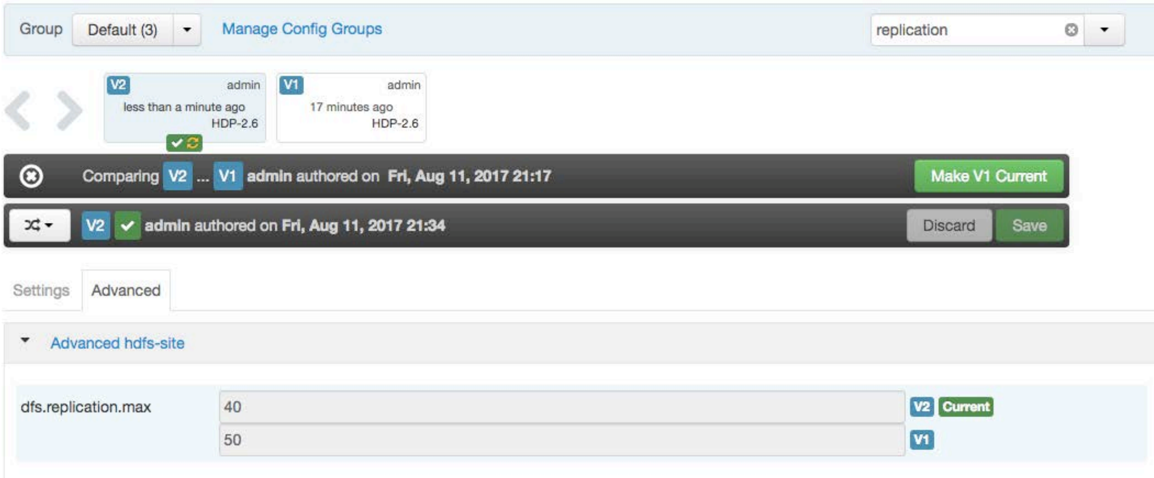
The version 1 configuration was the configuration created during installation.
The version 2 configuration includes the change to the `dfs.replication.max` parameter.

11 . Float the mouse pointer over the **V1** configuration and select **Compare** in the pop-up window.

This enables you to compare the change between the **V1** and the current (**V2**) HDFS configuration.

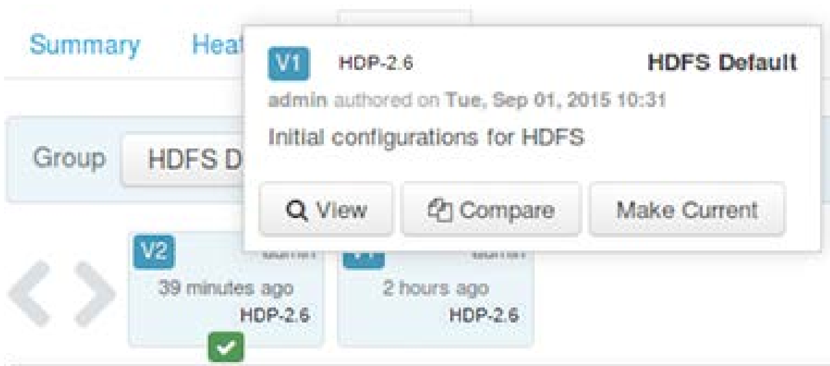
The current configuration is indicated by the green check box.





12 . Float the mouse pointer over the **V1** configuration and select **Make Current** in the pop-up window. This enables you to revert to the **V1** HDFS configuration.

Click **Make Current** again when the confirmation window opens.



13 . Ambari displays a restart icon next to any service or services that must also be restarted as a result of the HDFS configuration change.

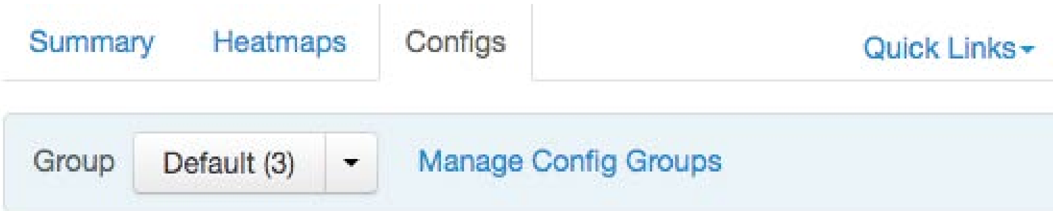
Select each service indicated by Ambari and restart it.

Creating an Ambari Configuration Group

Create an Ambari Configuration Group.

1 . Select the **HDFS** service and click its **Configs** tab.

2 . Click **Manage Config Groups**.

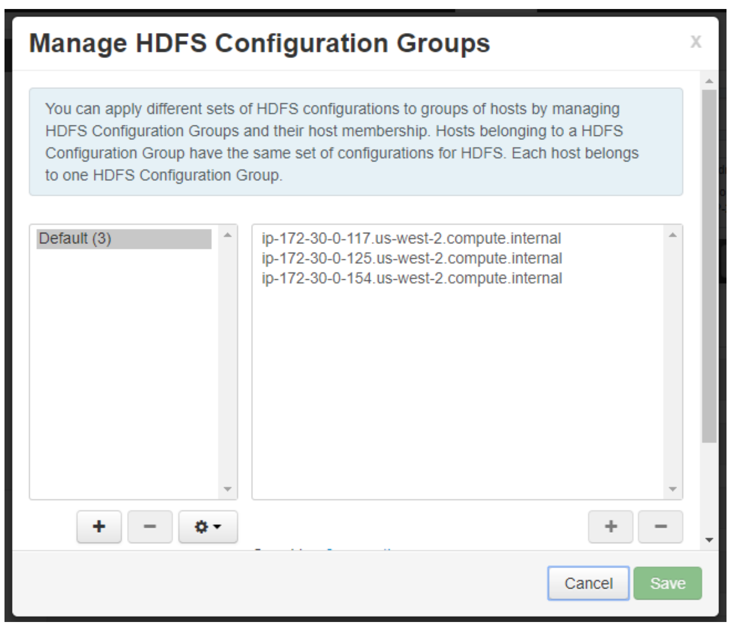


Notice the name of the currently selected configuration group.

It is **Default** and has three cluster node assigned to it.

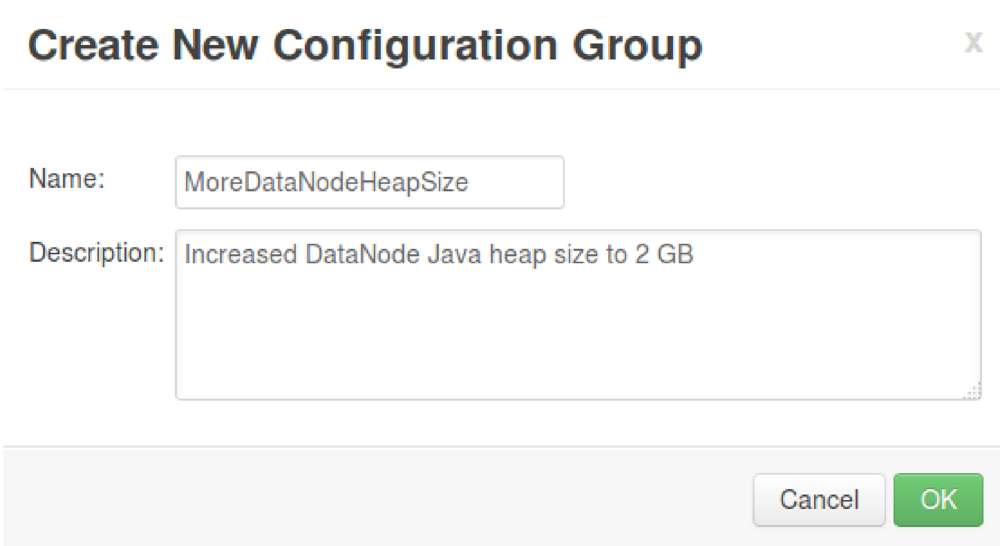
Any configuration change that is made will be made to this configuration group and applied to the single cluster node that is assigned to this configuration group.

3 . In the Manage HDFS Configuration Groups window, click the **+** (plus icon) button beneath the HDFS Default configuration group to create a new configuration group.



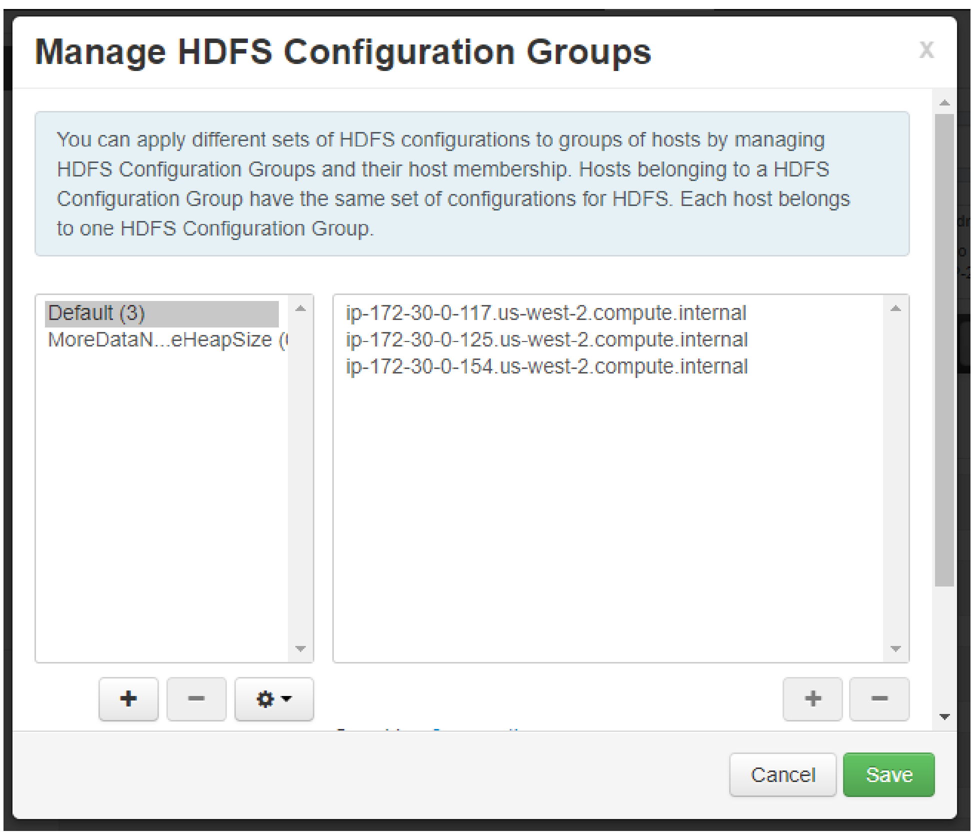
4 . In the Create New Configuration Group window, type `MoreDataNodeHeapSize` as the configuration group name and add an optional description.

Then click **OK**.

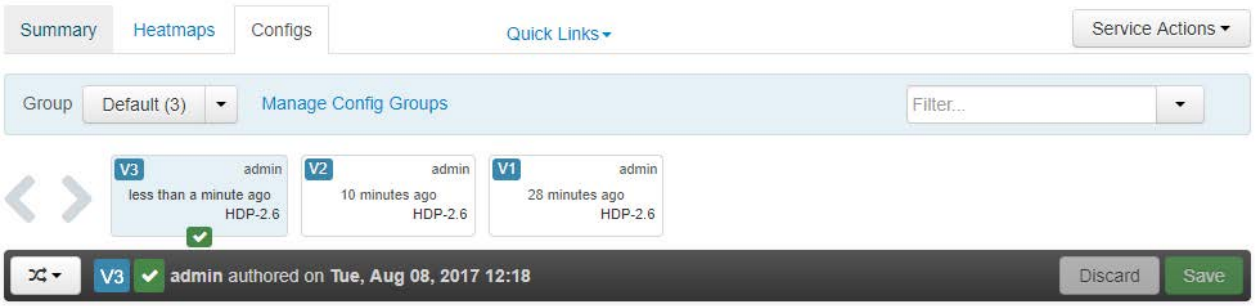


5 . Normally you would then click the (**+**) plus icon on the right side of the window to add some of the HDFS DataNodes to the configuration group.

Instead, just click **Save** to save the new configuration group.

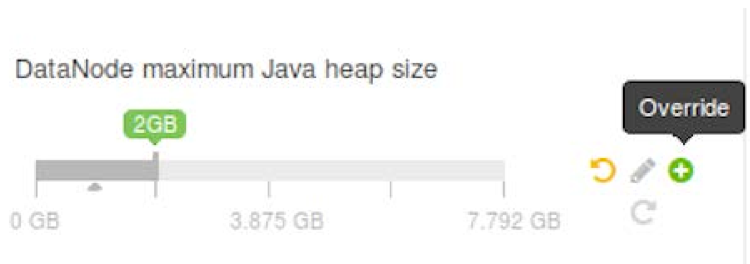


6 . With the **HDFS** service still selected, ensure that you are on the **Settings** tab on the **Configs** page.

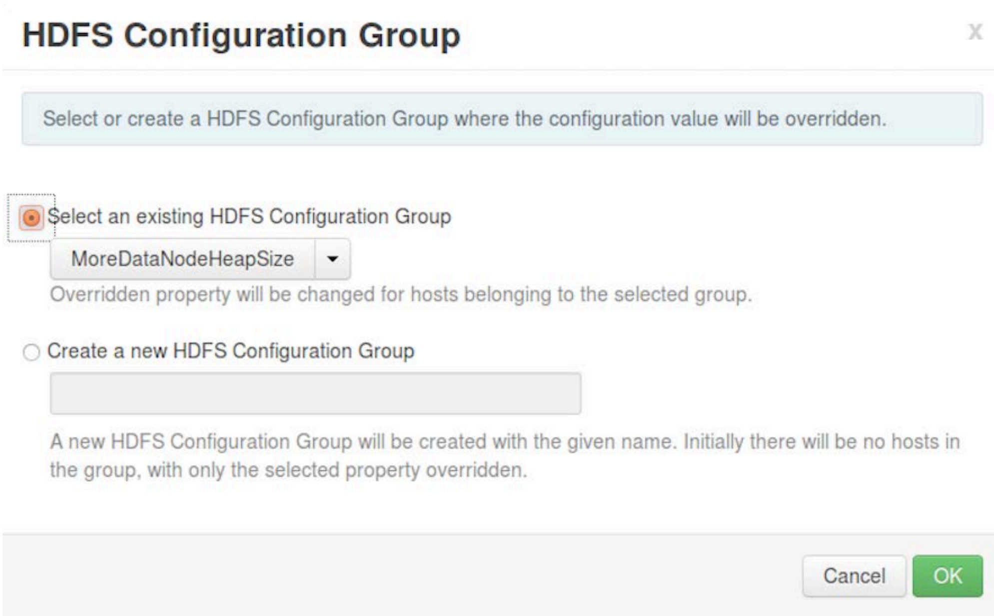


Scroll down, if necessary, to change the **DataNode maximum Java heap size** property to **2GB**.

Then use the green plus sign icon (the **Override** icon) to change this property setting for a specific configuration group.



7 . Select the configuration group created earlier (it is likely already selected) and then click **OK**.



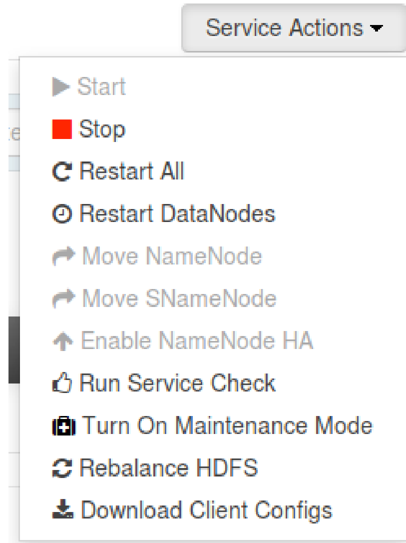
8 . In future labs when making modifications to HDFS, it is important to ensure that the HDFS Default configuration group is selected rather than the new MoreDataNodeHeapSize configuration group.

This will ensure that any HDFS modification will be made to the DataNodes that is assigned to the HDFS Default configuration group.

Downloading HDFS Configuration Files

Download HDFS client configuration files.

1 . With the **HDFS** service still selected, click the **Service Actions** menu button and select **Download Client Configs**.



2 . Choose **Save File** and click **OK**.

3 . Open a new terminal window on your desktop and change directory to the location where the Browser downloaded the file.

4 . Uncompress and untar the file using the following command.

```
tar xvfz HDFS_CLIENT-configs.tar.gz
```

Notice the names of the HDFS client configuration files.

Each service or application has its own set of configuration files.

5 . Close the terminal window when you are finished.

Result

You have now:
Viewed the Hadoop service configuration files that were installed with Ambari; toured Ambari Web UI service management and configuration features; changed the HDFS service configuration using the Ambari Web UI; create an Ambari Configuration Group; and downloaded HDFS client configuration files.

Lab 4: Using HDFS Storage

About This Lab

Objective:	To view and manage HDFS files and directories using the HDFS Shell, the Ambari Files View and the NameNode UI
File locations:	N/A
Successful outcome:	**You will:** Explore the HDFS Shell; use the HDFS Shell to manage files and directories in HDFS; use HDFS Shell command-line options to override the cluster's default HDFS properties; configure the Ambari Files View; and use the NameNodeUI file browser to view files and directories.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Using Hadoop Storage***

Exploring HDFS Shell

Learn to use the HDFS Shell.

1 . Use SSH to login to any cluster node.

```
sudo su -
```

2 . Display usage information for the HDFS Shell.

```
hdfs dfs
```

3 . Display the online manual help for the HDFS Shell.

```
hdfs dfs -help
```

4 . Display the online manual help for a specific HDFS Shell command.

```
hdfs dfs -help ls
```

5 . Use the HDFS Shell to list files and directories on the local file system.

```
hdfs dfs -ls file:///usr
```

6. Use the HDFS Shell to list files and directories in HDFS.

```
hdfs dfs -ls hdfs:///user
hdfs dfs -ls /user
```

Notice that the last two commands list the same HDFS files and directories, but the format of the command output is slightly different

Using the HDFS Shell

Use the HDFS Shell to manage files and directories in HDFS.

1. List files and directories in the HDFS root (`/`) directory.

```
hdfs dfs -ls /
```

You should see a list of directory names.

2. List the root user's HDFS home directory by issuing the same command again, but this time without supplying a directory path argument.

```
hdfs dfs -ls
```

The command fails because the root user does not have an HDFS home directory.

3. Create an HDFS home directory for the root user.

```
hdfs dfs -mkdir /user/root
```

The command fails because the root user does not have HDFS permissions to write to the `/user` directory.

4. View the name of the user that has superuser privileges in HDFS.

```
more /etc/hadoop/conf/hdfs-site.xml
```

Search for the `dfs.cluster.administrators` property.

Which user is an HDFS superuser?

5. Switch to the HDFS superuser account and create the root user's home directory using the command sequence shown below.

```
su - hdfs
hdfs dfs -mkdir /user/root
hdfs dfs -ls /user
```

Notice that the directory owner and group owner are both `hdfs` and should be `root`.

Use the HDFS Shell -*chown* command to change the directory's ownership.

```
hdfs dfs -chown root:root /user/root
hdfs dfs -ls /user
```

Notice that the directory owner and group owner are now `root`.

Now use the *exit* command to switch back to normal root user account permissions.

```
exit
```

6. List the root user's HDFS home directory.

```
hdfs dfs -ls
```

Although the command should execute properly, there are no files or directories to list in the root user's HDFS home directory.

7. Create a subdirectory hierarchy in the root user's HDFS home directory.

```
hdfs dfs -mkdir -p dir1/dir2/dir3
```

8. Recursively list the contents of the root user's HDFS home directory to verify that the directory hierarchy was created.

```
hdfs dfs -ls -R
```

9. Copy the file `constitution.txt` from the local file system to the root user's HDFS home directory.

   Then list the root user's HDFS home directory to verify that the file was copied.

```
wget https://raw.githubusercontent.com/HortonworksUniversity/Ops_Labs/master/const.txt
hdfs dfs -put const.txt constitution.txt
hdfs dfs -ls
```

10. Display the contents of the `constitution.txt` file in the root user's HDFS home directory.

```
hdfs dfs -cat constitution.txt | more
```

11. View only the last 1 KB of the `constitution.txt` file in the root user's HDFS home directory.

```
hdfs dfs -tail constitution.txt
```

12. Copy the `constitution.txt` file from the root user's HDFS home directory to the root user's home directory in the local file system. Then list the root user's home directory in the local file system to verify that the file was copied.

```
hdfs dfs -get constitution.txt /root/constitution.txt
ls /root
```

13. Move the `constitution.txt` file in the root user's HDFS home directory to the HDFS directory `/user/root/dir1/dir2/dir3`.

    Then recursively list the root user's HDFS home directory to verify that the file was moved.

```
hdfs dfs -mv constitution.txt dir1/dir2/dir3
hdfs dfs -ls -R
```

14. Copy the local files `/etc/passwd` and `/etc/hosts` to the root user's HDFS home directory.

    Then copy these two files back to the root user's home directory on the local file system, ensuring that during the copy process the files are merged into a single file named `passwd_hosts`.

    Then view the contents of the merged file to verify that the process properly completed.

```
hdfs dfs -put /etc/passwd passwd
hdfs dfs -put /etc/hosts hosts
hdfs dfs -ls
hdfs dfs -getmerge passwd hosts /root/passwd_hosts
cat /root/passwd_hosts
```

15. Remove the `passwd` file from the root user's HDFS home directory.

    Then list the HDFS home directory to verify that it has been removed.

```
hdfs dfs -rm passwd
hdfs dfs -ls
```

    What message did you receive when you removed the `passwd` file?

    When you listed the contents of the HDFS home directory, did you notice the new `.Trash` directory?

16. Recursively list the contents of the root user's HDFS `.Trash` directory.

```
hdfs dfs -ls -R .Trash
```

    Was the `passwd` file completely removed, or was it temporarily moved to the HDFS `.Trash` directory?

17. Retrieve the `passwd` file from the HDFS `.Trash` directory by moving it back to the root user's HDFS home directory.
18. Once finished, recursively list the contents of the HDFS home directory to verify that the operation completed successfully.

```
hdfs dfs -mv .Trash/Current/user/root/passwd passwd
hdfs dfs -ls -R
```

19. View the contents of the HDFS `/user/root/dir1/dir2/dir3` directory and then remove the `dir3` directory.

    Then verify that the `dir3` directory and its files were removed.

```
hdfs dfs -ls dir1/dir2/dir3
hdfs dfs -rm dir1/dir2/dir3
```

What message did you receive?

The *-rm* command (with no other command options) cannot remove directories.

20. Try to remove the `dir3` directory again, but this time include the *-rm* command's *-R* option.

    List the contents of the `dir2` directory to verify that the `dir3` directory was removed.

```
hdfs dfs -rm -R dir1/dir2/dir3
hdfs dfs -ls dir1/dir2
```

Was the directory removed?
It should be.

Overriding Default HDFS Properties

Use HDFS Shell command-line options to override the cluster's default HDFS properties.

1. View the default HDFS block size as defined by the `dfs.blocksize` property in the `/etc/hadoop/conf/hdfs-site.xml` file.

```
more /etc/hadoop/conf/hdfs-site.xml
```

It should be 134,217,728 bytes (128 MB).

2. Write the `/etc/passwd` file to the root user's HDFS home directory using the non-default block size of 1,000,000 bytes.

```
hdfs dfs -D dfs.blocksize=1000000 -put /etc/passwd test_data
```

The command should fail with a message reporting the reason for the failure.

The minimum block size must be at least 1,048,576 bytes based on the `dfs.namenode.fs-limits.min-block-size` property defined in the `hdfs-default.xml` file.

3. Write the `/etc/passwd` file to the root user's HDFS home directory using the non-default block size of 1,050,000 bytes, which is larger than the minimum block size.

```
hdfs dfs -D dfs.blocksize=1050000 -put /etc/passwd test_data
```

The command should fail with a message reporting the reason for the failure.

While the minimum block size has been met, the block size must also be a multiple of 512 bytes in order to support the checksum system used to verify HDFS data integrity.

This is based on the `dfs.bytes-per-checksum` property defined in the `hdfs-default.xml` file.

4. Write the `/etc/passwd` file to the root user's HDFS home directory using the non-default block size of 1,049,088 bytes.

```
hdfs dfs -D dfs.blocksize=1049088 -put /etc/passwd test_data
```

The command should have completed successfully as the block size of 1,049,088 bytes is exactly the minimum required block size plus an additional 512 bytes.

Configuring Ambari Files View

Configuring Ambari Files View

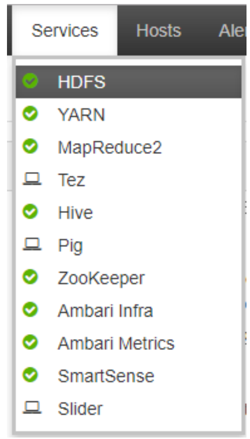
Configure the Ambari Files View

1. Open the browser and connect to the Ambari Server at the URL

   http://<Ambari Server Hostname>:8080

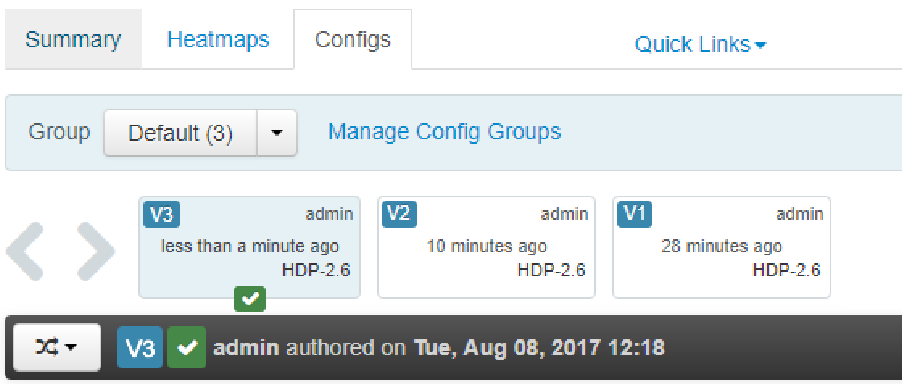
2. Login to the Ambari Web UI using the user name `admin` and the password `BadPass#1`.

3. In the Ambari Web UI, click **Services** and select **HDFS**.

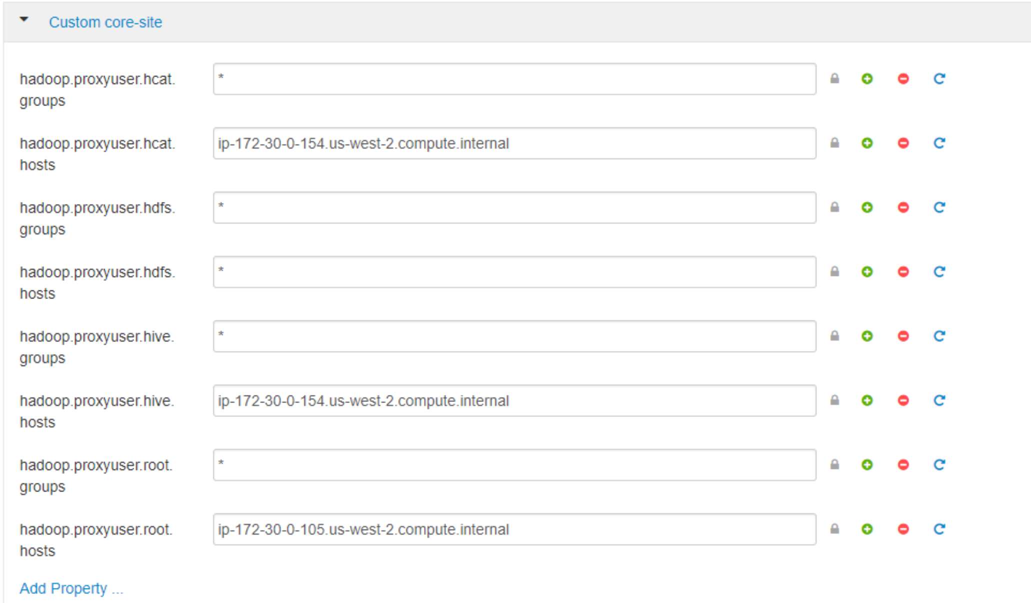


4 . Click **Configs** and then click the **Advanced** tab.

Also ensure that the **HDFS Default** configuration group is selected.



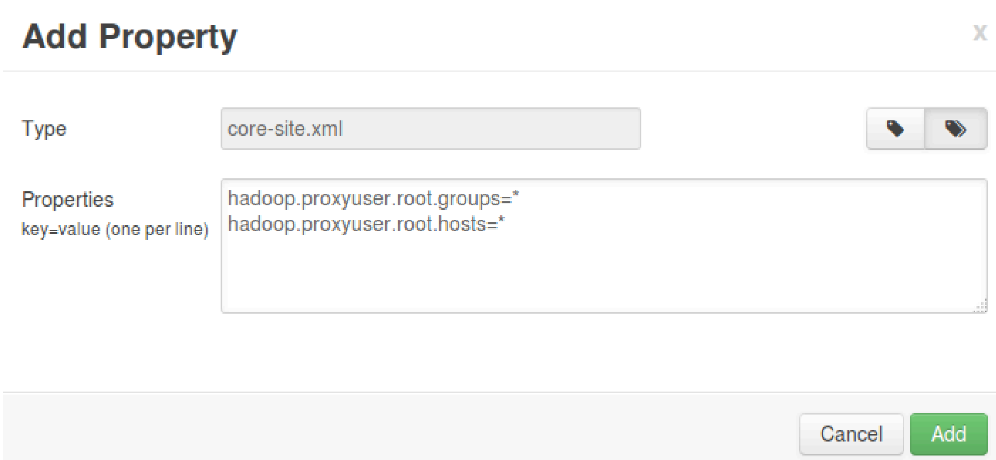
5 . Scroll down to find and expand the **Custom core-site** section and then click **Add Property**.



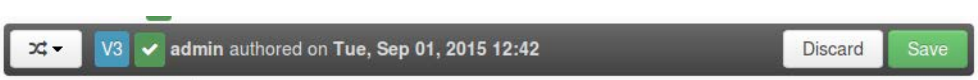
6 . Click the double-tag icon and then add the property and value `hadoop.proxyuser.root.groups=*` and the property and value `hadoop.proxyuser.root.hosts=*` to the **Custom core-site** section.

Then click **Add**.

NOTE: These properties may already be pre-populated by Ambari. If so, edit to ensure they have the value of *.



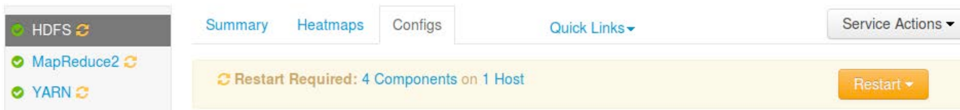
7 . Click **Save** to save the modifications.



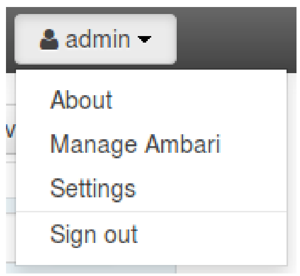
Click **Save** again in the Save Configuration window.

Click **OK** in the confirmation window.

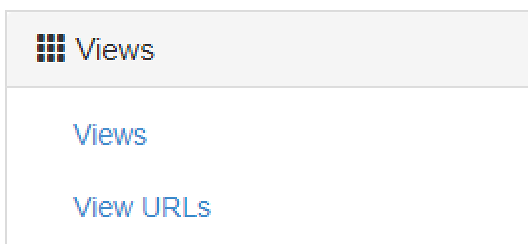
8 . Restart any services as indicated in the Ambari Web UI.



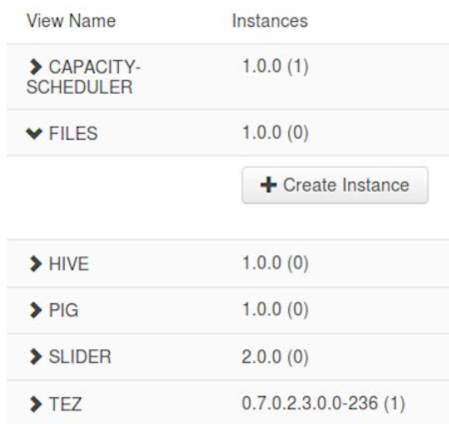
9 . Select **Manage Ambari** from the **admin** menu button in order to start the process of creating an instance of an Ambari View.



10 . Click **Views**.

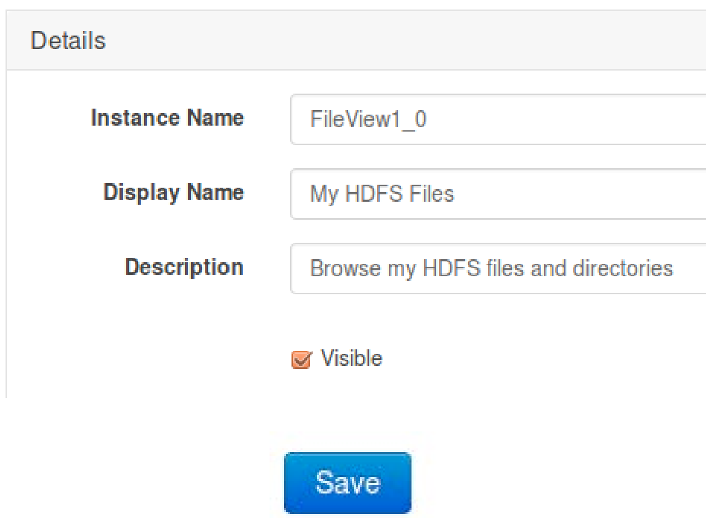


11 . Click to expand **Files** and then click **Create Instance**.



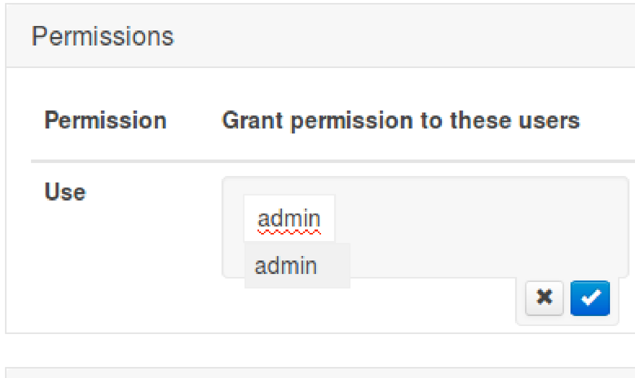
12 . In the **Details section**, enter `FileView1_0` as the Instance Name, `My HDFS Files` as the Display Name, and `Browse my HDFS files and directories` as the Description.

Then scroll down and click **Save**.



13 . Scroll down and locate the **Permissions** section. Beneath **Grant permissions to these users**, then click **New** and type `admin`.

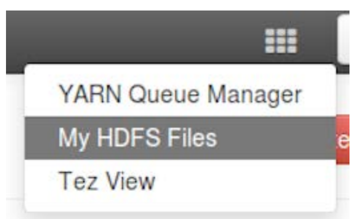
Click the blue check box to save your change.



14 . Use SSH login to any cluster node, create the HDFS home directory `/user/admin` for the admin user account. You will need HDFS superuser privilege to create this home directory.

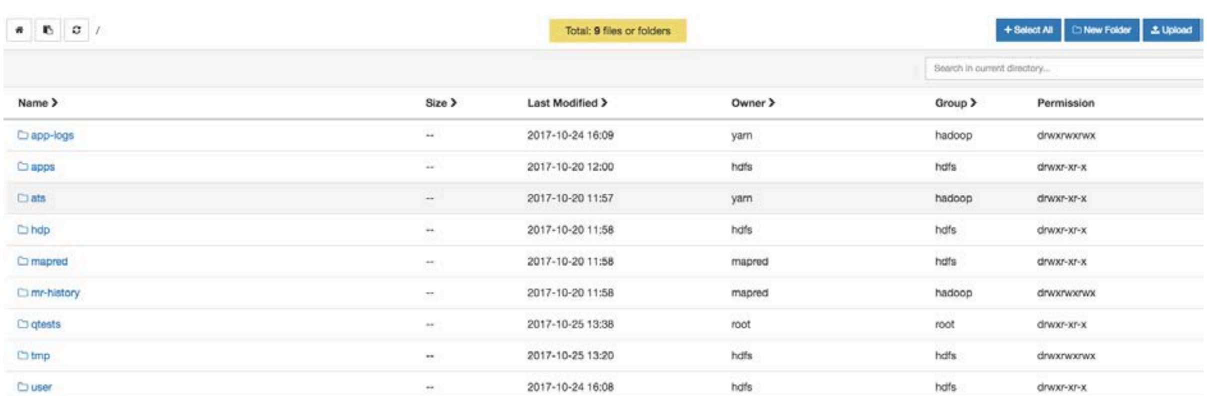
```
su - hdfs
hdfs dfs -mkdir /user/admin
hdfs dfs -chown admin:admin /user/admin
exit
```

15 . Back in the Ambari Web UI,
click the **Views** icon and select the View named **My HDFS Files**.



16 . You will be taken to the `/` directory of HDFS, however the admin user does not have sufficient HDFS permissions to do anything here but view the directory names.

NOTE: The **New Folder** and **Upload** buttons are activated but will not allow the admin user to create a "New Folder" or "Upload a File" due to the lack of necessary HDFS permissions.



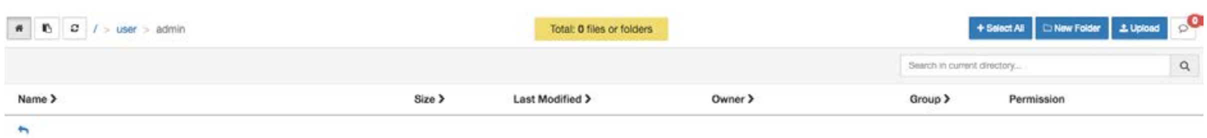
17 . Browse to the `/user/admin` directory by clicking on the **user** folder link at the bottom of the menu.

Name	Size	Last Modified	Owner	Group	Permission
app-logs	--	2017-10-24 16:09	yarn	hadoop	drwxrwxrwx
apps	--	2017-10-20 12:00	hdfs	hdfs	drwxr-xr-x
ats	--	2017-10-20 11:57	yarn	hadoop	drwxr-xr-x
hdp	--	2017-10-20 11:58	hdfs	hdfs	drwxr-xr-x
mapred	--	2017-10-20 11:58	mapred	hdfs	drwxr-xr-x
mr-history	--	2017-10-20 11:58	mapred	hadoop	drwxrwxrwx
qtests	--	2017-10-25 13:38	root	root	drwxr-xr-x
tmp	--	2017-10-25 13:20	hdfs	hdfs	drwxrwxrwx
user	--	2017-10-24 16:08	hdfs	hdfs	drwxr-xr-x

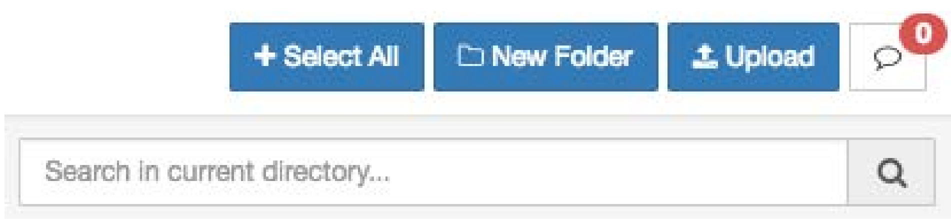
Next click the **admin** folder link.

Name	Size	Last Modified	Owner	Group	Permission
admin	--	2017-10-26 11:26	admin	admin	drwxr-xr-x
ambari-qa	--	2017-10-25 13:19	ambari-qa	hdfs	drwxrwx---
hcat	--	2017-10-20 12:00	hcat	hdfs	drwxr-xr-x
hdfs	--	2017-10-24 11:16	hdfs	hdfs	drwx------
hive	--	2017-10-20 12:00	hive	hdfs	drwxr-xr-x
promo01	--	2017-10-24 16:39	promo01	promo	drwxr-xr-x
qa01	--	2017-10-24 16:12	qa01	qa	drwxr-xr-x
root	--	2017-10-24 15:09	root	root	drwxr-xr-x

You will know you are in the proper place when the **New directory** and **Upload** buttons at top-right are activated.

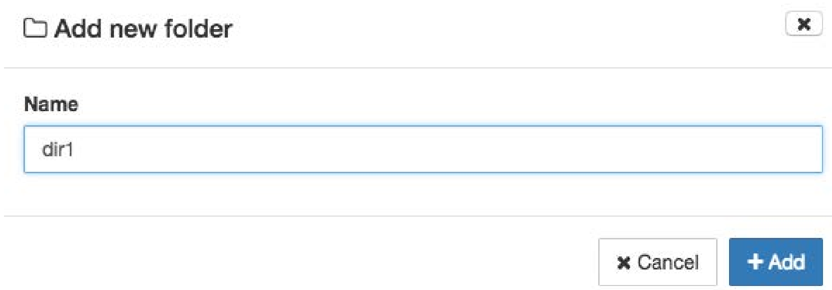


A closer look:

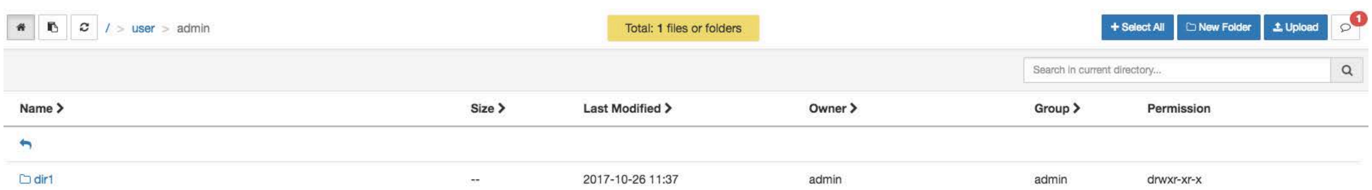


18 . Click **New Folder** to create a new HDFS directory in the admin user's HDFS home directory.

19 . Name the new directory `dir1` and then click **Create**.



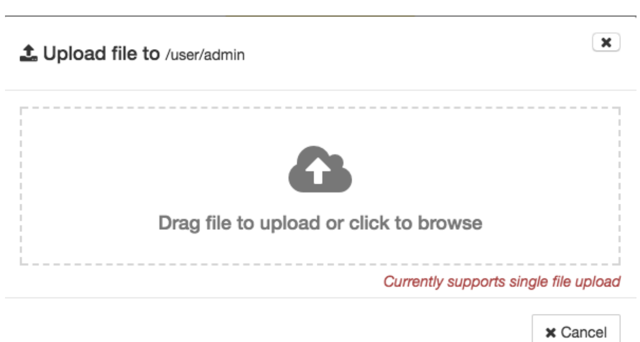
20 . Click **dir1** to open the `dir1` directory.



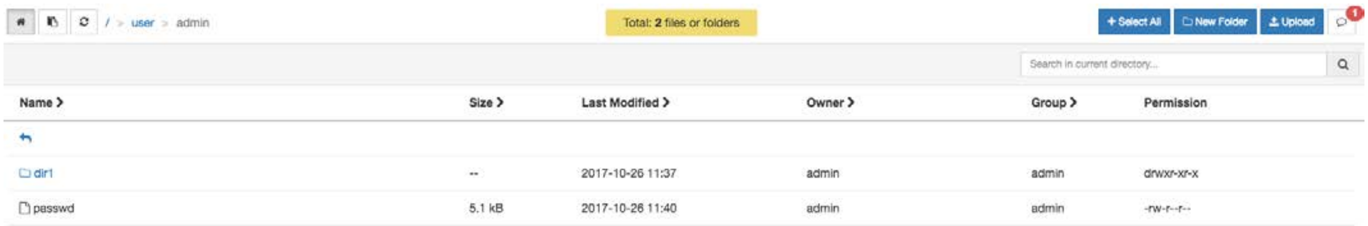
21 . Click **Upload** to upload a file to the `dir1` directory.

22 . Click **Browse** and in the file browser window, select **any file you have**, and then click **Open**.

Once the file has been selected, click **Upload**.



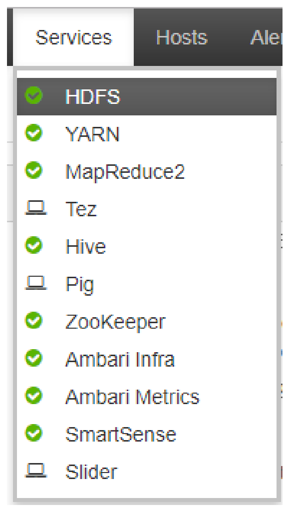
The result should look like the following:



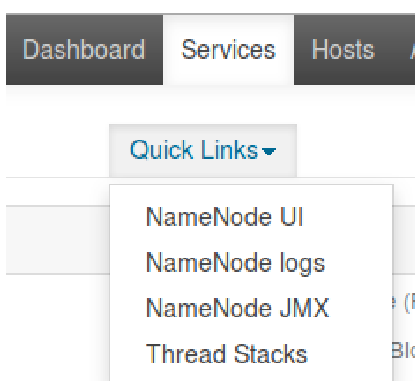
Using the NameNode UI File Browser

Use the NameNode UI file browser to view HDFS files and directories.

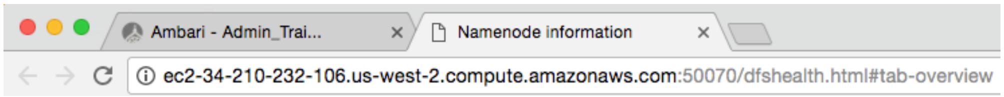
1 . In the Ambari Web UI, click **Services** and select **HDFS**.



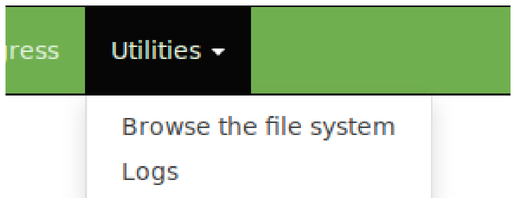
2 . On the **Summary** page, click **Quick Links** and select **NameNode UI.**



3 . In the browser, click the new **Namenode information** tab, if necessary, to move to the NameNode UI.



4. In the NameNode UI, click **Utilities** and select **Browse the file system**.



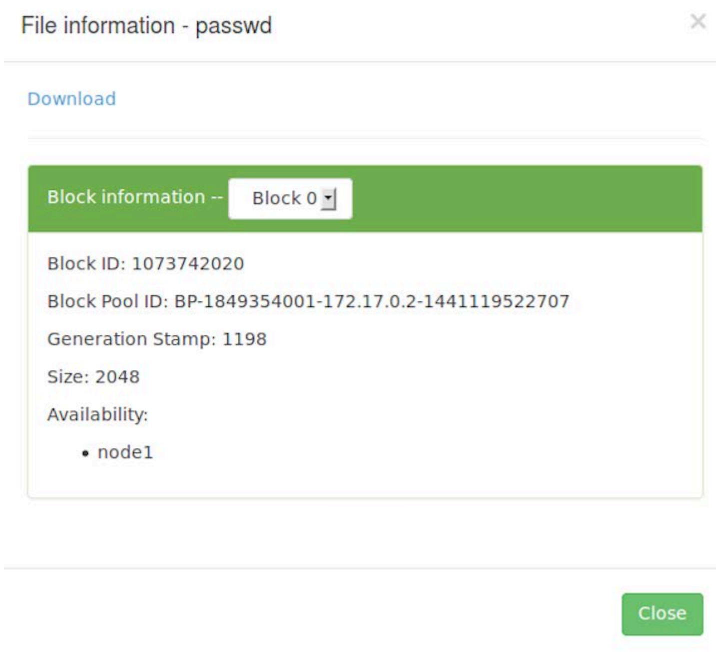
5. To view the contents of the `/user/admin/dir1` directory, click the *user* directory name.

   Then click the **admin** directory name followed by the **dir1** directory name.

drwxr-xr-x	hdfs	hdfs	0 B	9/1/2015, 9:40:57 PM	0	0 B	user
drwxr-xr-x	admin	admin	0 B	9/1/2015, 9:52:37 PM	0	0 B	admin
drwxr-xr-x	admin	admin	0 B	9/1/2015, 10:05:22 PM	0	0 B	dir1

6. To view information about the `passwd` file in HDFS, click the **passwd** file name.

-rw-r--r--	admin	admin	2 KB	9/1/2015, 10:05:22 PM	3	128 MB	passwd

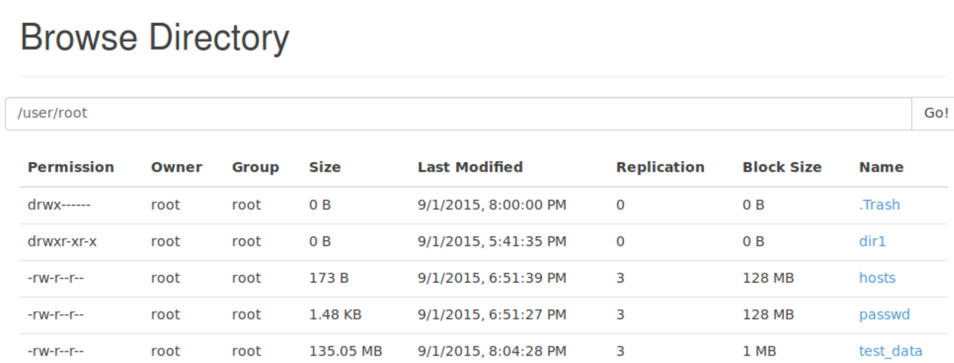


Notice that the only way to view the file contents is to download the file to the local machine. This could be a lengthy or even impossible process for the large files often stored in HDFS.

7. Click **Close** to close the open File information window.

8. You can also easily view directory contents or file information if you know the name and path to a file or directory.

   In the NameNode UI on the **Browse Directory** page, type `/user/root` into the path name text box and then press the **Return** key.



9. Close the NameNode UI browser tab when you are finished with the NameNode UI.

Result

You have now:
Explored the HDFS Shell; used the HDFS Shell to manage files and directories in HDFS; used HDFS Shell command-line options to override the cluster's default HDFS properties; configured the Ambari Files View; and used the NameNodeUI file browser to view files and directories.

Lab 5: Using WebHDFS

About This Lab

Objective:	To gain familiarity with some basic operation of WebHDFS
File locations:	N/A
Successful outcome:	**You will:** Use Ambari Web UI to verify that WebHDFS is enabled in your cluster and used WebHDFS API commands to create an HDFS directory, list an HDFS directory, and read a file from HDFS.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Using Hadoop Storage***

Verifying WebHDFS is Enabled

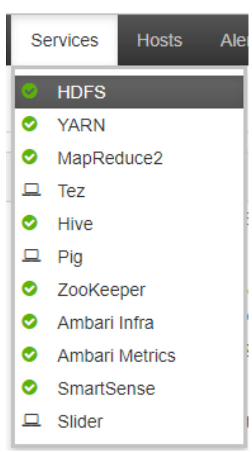
Use the Ambari Web UI to verify that WebHDFS is enabled in your cluster.

1. Open the browser and connect to the Ambari Server at the URL

   http://<Ambari Server Hostname>:8080

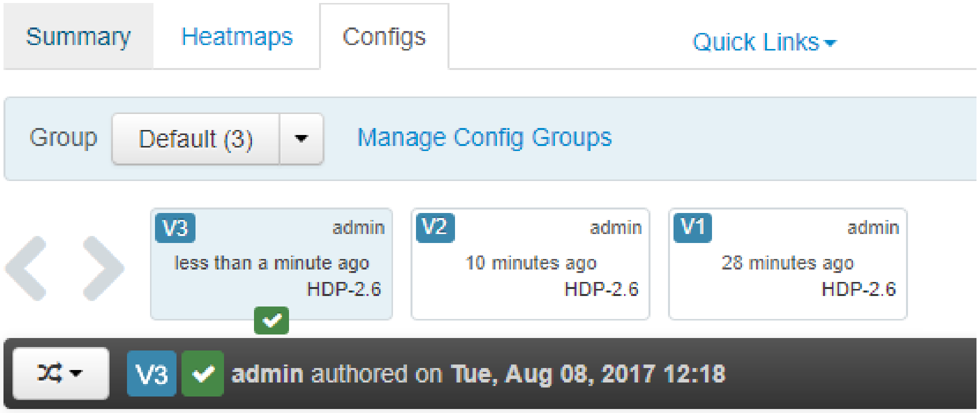
2. Login to the Ambari Web UI using the user name `admin` and the password `BadPass#1`.

3. In the Ambari Web UI, click **Services** and select **HDFS**.



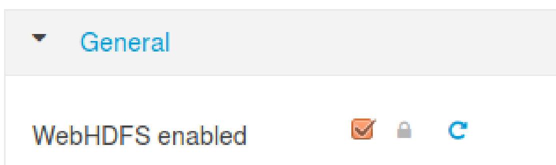
4. Click **Configs** and then click the **Advanced** tab.

   Ensure that the **HFDS Default** configuration group is selected.



5 . Scroll down to find and expand the **General** section and look for the **WebHDFS enabled** check box.

It should already be selected.



6 . You are finished using the Ambari Web UI.

Using WebHDFS API Commands

Use WebHDFS API commands as arguments to the Linux `curl` command.

(This can be useful to ensure that WebHDFS is functioning properly.)

1 Use SSH to login to any cluster node.

```
sudo su -
```

2 Use the *curl* command with WebHDFS API arguments to list the contents of the HDFS `/user/root` directory. Use Ambari UI to locate the internal AWS hostname for your NameNode, example would be "ip-172-30-0-184.us-west-2.compute.internal"

```
curl -i "http://<NameNode Host>:50070/webhdfs/v1/user/root?op=LISTSTATUS"
```

```
[root@ip-172-30-0-184 ~]# curl -i "http://ip-172-30-0-184.us-west-2.compute.inte
rnal:50070/webhdfs/v1/user/root?op=LISTSTATUS"
HTTP/1.1 200 OK
Cache-Control: no-cache
Expires: Mon, 14 Aug 2017 14:10:27 GMT
Date: Mon, 14 Aug 2017 14:10:27 GMT
Pragma: no-cache
Expires: Mon, 14 Aug 2017 14:10:27 GMT
Date: Mon, 14 Aug 2017 14:10:27 GMT
Pragma: no-cache
Content-Type: application/json
X-FRAME-OPTIONS: SAMEORIGIN
Transfer-Encoding: chunked
Server: Jetty(6.1.26.hwx)

{"FileStatuses":{"FileStatus":[
{"accessTime":0,"blockSize":0,"childrenNum":0,"fileId":21451,"group":"root","len
gth":0,"modificationTime":1502719770021,"owner":"root","pathSuffix":"dir0","perm
ission":"755","replication":0,"storagePolicy":0,"type":"DIRECTORY"},
{"accessTime":0,"blockSize":0,"childrenNum":0,"fileId":21452,"group":"root","len
gth":0,"modificationTime":1502719784651,"owner":"root","pathSuffix":"dir1","perm
ission":"755","replication":0,"storagePolicy":0,"type":"DIRECTORY"},
{"accessTime":1502719727751,"blockSize":134217728,"childrenNum":0,"fileId":21445
,"group":"root","length":159,"modificationTime":1502719728301,"owner":"root","pa
thSuffix":"hosts","permission":"644","replication":3,"storagePolicy":0,"type":"F
ILE"},
{"accessTime":1502719756402,"blockSize":134217728,"childrenNum":0,"fileId":21450
,"group":"root","length":2212,"modificationTime":1502719756639,"owner":"root","p
athSuffix":"passwd","permission":"644","replication":3,"storagePolicy":0,"type":
"FILE"},
{"accessTime":1502719814921,"blockSize":134217728,"childrenNum":0,"fileId":21453
,"group":"root","length":22205,"modificationTime":1502719815151,"owner":"root","
pathSuffix":"test_data","permission":"644","replication":3,"storagePolicy":0,"ty
pe":"FILE"}
]}}
[root@ip-172-30-0-184 ~]#
```

You should see a `200 OK` response (you might have to scroll up in the terminal window), along with a JSON object containing the list of files and directories in the `/user/root` directory.

3 Create a new subdirectory named `history` in `/user/root`.

The following command is entered on a single line.

```
curl -i -X PUT "http://<NameNode Host>:50070/webhdfs/v1/user/root/history?op=MKDIRS&user.name=root"
```

```
[root@ip-172-30-0-184 ~]# curl -i -X PUT "http://ip-172-30-0-184.us-west-2.compu
te.internal:50070/webhdfs/v1/user/root/history?op=MKDIRS&user.name=root"
HTTP/1.1 200 OK
Cache-Control: no-cache
Expires: Mon, 14 Aug 2017 14:13:35 GMT
Date: Mon, 14 Aug 2017 14:13:35 GMT
Pragma: no-cache
Expires: Mon, 14 Aug 2017 14:13:35 GMT
Date: Mon, 14 Aug 2017 14:13:35 GMT
Pragma: no-cache
Content-Type: application/json
X-FRAME-OPTIONS: SAMEORIGIN
Set-Cookie: hadoop.auth="u=root&p=root&t=simple&e=1502756015324&s=tlB4TPbffmxNU4
6IUjdK+JvXUS8="; Path=/; HttpOnly
Transfer-Encoding: chunked
Server: Jetty(6.1.26.hwx)

{"boolean":true}[root@ip-172-30-0-184 ~]#
```

The `user.name=root` argument is required to have the necessary permissions to write to the `/user/root` directory.

4 Use the HDFS Shell *-ls* command to verify that the `history` subdirectory was created successfully.

```
hdfs dfs -ls
```

You should see a `history` directory in the output.

5 Read the contents of the `passwd` file using the following *curl* command and WebHDFS API.

```
curl -i -L "http://<NameNode Host>:50070/webhdfs/v1/user/root/passwd?op=OPEN"
```

The contents of the file may scroll out of the terminal window.

Result

You have now:
Used Ambari Web UI to verify that WebHDFS is enabled in your cluster and used WebHDFS API commands to create an HDFS directory, list an HDFS directory, and read a file from HDFS.

Lab 6: Using HDFS Access Control Lists

About This Lab

Objective:	To configure and use HDFS access control lists (ACLs)
File locations:	N/A
Successful outcome:	**You will:** Configure HDFS ACL support; create and manage file ACL entries, manage ACL entries and default ACL entries on directories; and manage user and group access using an access mask.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Using Hadoop Storage***

Configuring HDFS ACLs

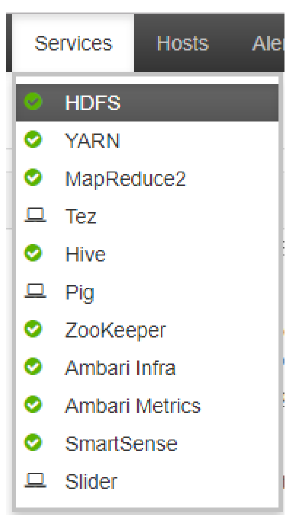
Use the Ambari Web UI to configure HDFS access control list support.

Open the browser and connect to the Ambari Server.

http://<Ambari Server Hostname>:8080

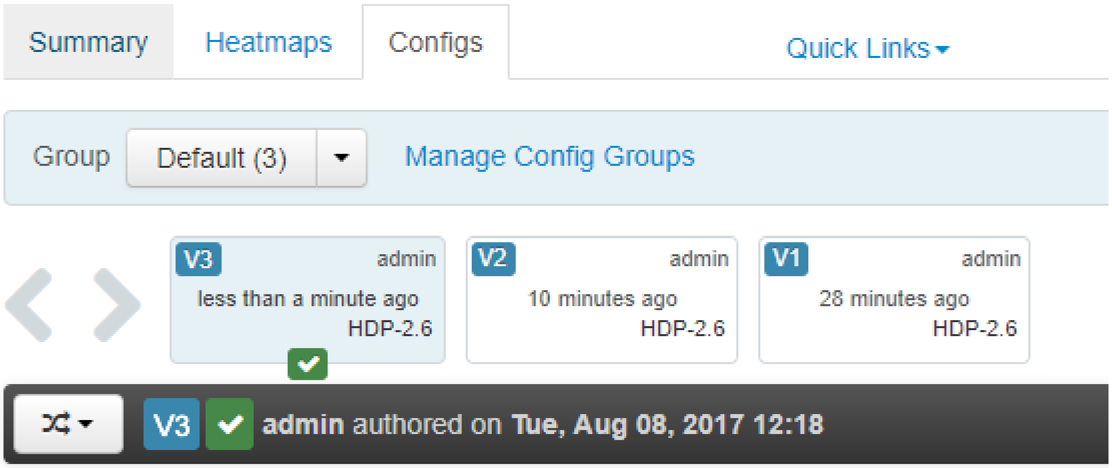
1. Login to the Ambari Web UI using the user name `admin` and the password `BadPass#1`.

2. In the Ambari Web UI, click **Services** and select **HDFS**.



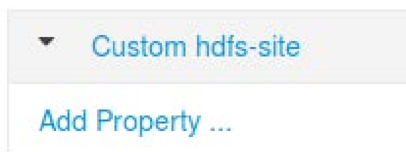
3. Click **Configs** and then click the **Advanced** tab.

   Ensure that the **HDFS Default** configuration group is selected.



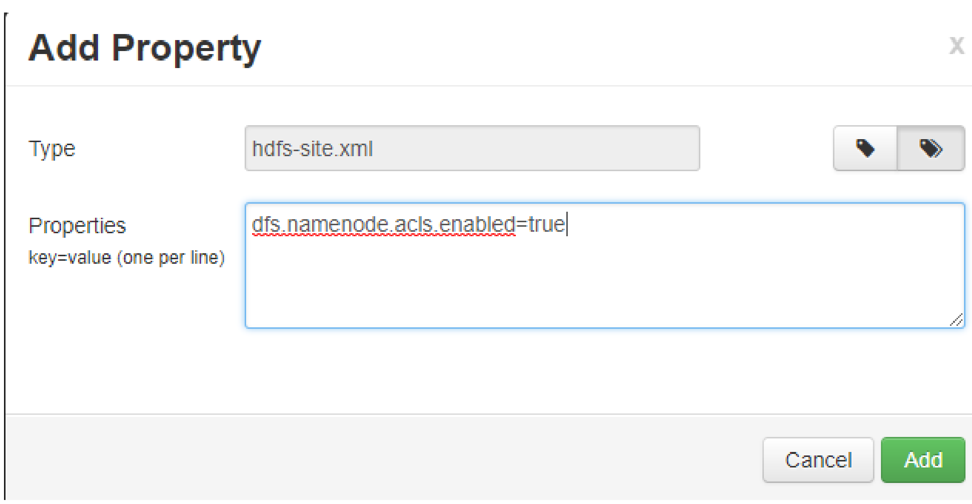
4. Scroll down to find and expand the **Custom hdfs-site** section.

   Click **Add Property**.

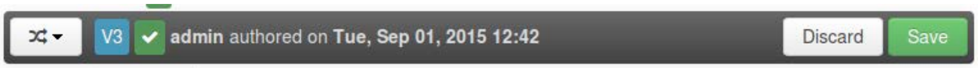


5. Click on the single-tag button, and then add the `dfs.namenode.acls.enabled` property with a value of `true.`

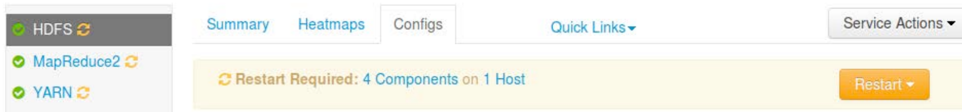
   Click **Add**.



6. Click **Save** to save your changes.



7 . Restart any services as indicated in the Ambari Web UI.

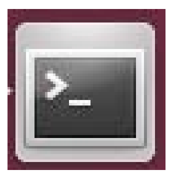


HDFS access control support has now been enabled.

Managing File ACLs

Create and Manage File ACL entries.

1 . Use SSH to login to any cluster node.



```
sudo su -
```

2 . Create a test directory named `/acltests`.

All testing of HDFS ACLs will occur in this directory. You must become the HDFS superuser to configure the test environment.

```
su - hdfs
hdfs dfs -mkdir /acltests
hdfs dfs -ls /
```

Notice that:
- the directory owner is `hdfs`,
- the group is `hdfs`,
- the directory grants read and execute access to anyone who is *not* the hdfs user or belongs to the hdfs group.

For example, the root user could access files in the `/acltests` directory.

3 . In order to have a file for test purposes, upload a copy of the local file `/etc/passwd` to the HDFS `/acltests` directory. Verify that the file was uploaded.

```
hdfs dfs -put /etc/passwd /acltests
hdfs dfs -ls /acltests
```

Notice that:
- the file owner is `hdfs`,
- the group is `hdfs`,
- the file grants read access by anyone who is **not** the hdfs user or does *not* belong to the hdfs group.

4. Remove read permissions from the `/acltests/passwd` file for anyone who is *not* hdfs or does *not* belong to the hdfs group.

```
hdfs dfs -chmod 640 /acltests/passwd
hdfs dfs -ls /acltests
```

The file permissions should now be `rw-r-----`. For example, the root user should *not* be able to read the `passwd` file.

5. To test root user read access to the `/acltests/passwd` file, assume normal root permissions again. Then try to read the file.

```
exit
hdfs dfs -cat /acltests/passwd
```

The *-cat* command should have failed with a message providing information about why the command failed.

6. To continue the testing, assume HDFS superuser privileges again.

```
su - hdfs
```

7. Use the HDFS Shell *-getfacl* command to view any ACL entries that might exist on the `/acltests/passwd` file.

```
hdfs dfs -getfacl /acltests/passwd
```

There should be no ACL entries on the file. It should have only standard HDFS user, group, and other permissions.

8. Use the HDFS Shell *-ls* command on the `/acltests` directory.

```
hdfs dfs -ls /acltests
```

Any file or directory with an ACL entry has a + character appended to the end of the permissions list.

Is there a + character for the `passwd` file?
There should not be.

9. Use the HDFS Shell *-setfacl* command to add an ACL entry for the root user that provides read access to the `/acltests/passwd` file. Then use the HDFS Shell *-getfacl* command to verify the change.

```
hdfs dfs -setfacl -m user:root:r-- /acltests/passwd
hdfs dfs -getfacl /acltests/passwd
```

You should see a new ACL entry for `user:root:r--`.
You should also see a new `mask::r--` entry.

10 . Use the HDFS Shell *-ls* command on the `/acltests` directory.

```
hdfs dfs -ls /acltests
```

Any file or directory with an ACL entry has a `+` character appended to the end of the permissions list.

Is there a `+` character for the `passwd` file?
There should be.

11 . To test root user read access, assume normal root permissions again.

```
exit
```

12 . Use the HDFS Shell *-cat* command to try to read the `/acltests/passwd` file.

```
hdfs dfs -cat /acltests/passwd
```

You should be able to view the contents of the file.

13 . Assume HDFS superuser permissions again.

```
su - hdfs
```

14 . Add another ACL entry to the `/acltests/passwd` file.

This time provide the group hcat read and write access to the file. Verify the change once you have added the new ACL entry.

```
hdfs dfs -setfacl -m group:hcat:rw- /acltests/passwd
hdfs dfs -getfacl /acltests/passwd
```

You should see a new ACL entry for `group:hcat:rw-`.

15 . Remove the ACL entry for the group hcat from the `/acltests/passwd` file. Verify the change once you have removed the ACL entry.

```
$ hdfs dfs -setfacl -x group:hcat /acltests/passwd
$ hdfs dfs -getfacl /acltests/passwd
```

Only the ACL entry for `group:hcat:rw-` should have been removed.

16 . Use the HDFS Shell *-setfacl* command with the *--set* option to completely remove and replace the entire set of ACL entries. Verify the change once you have finished.

```
hdfs dfs -setfacl --set user::rw-,group::rw-,other::---,user:admin:rw-
,group:hcat:rw- /acltests/passwd
hdfs dfs -getfacl /acltests/passwd
```

You should see the entire set of permissions and ACL entries replaced with the permissions and ACL entries specified in the command.

For example, there should no longer be an ACL entry for `user:root:r--`.

17. Remove all ACL entries from the `/acltests/passwd` file. Verify the removal when you have finished.

```
hdfs dfs -setfacl -b /acltests/passwd
hdfs dfs -getfacl /acltests/passwd
```

Only standard HDFS user, group, and other permissions should remain.

18. Upload the `/etc/group` file from the local file system to the HDFS `/acltests` directory. This file will be referenced in the next lab section.

```
hdfs dfs -put /etc/group /acltests/group
hdfs dfs -ls /acltests/
```

Managing Directory ACLs

Manage ACL entries and default ACL entries on directories.

1. While you still have HDFS superuser priveleges, check whether there are ACL entries for the `/acltests` directory.

```
hdfs dfs -getfacl /acltests
hdfs dfs -ls /
```

Both commands should reveal that the directory does not have ACL entries.
(Remember that the HDFS Shell -ls command displays a + character at the end of the permissions list for any file or directory that has ACL entries.)

2. Add a default ACL entry for the root user to the `/acltest` directory.

The default ACL entry should provide the root user with read and write permissions. Verify the changes once you have finished.

```
hdfs dfs -setfacl -m default:user:root:rw- /acltests
hdfs dfs -getfacl /acltests
```

Is there a default ACL entry for `default:user:root:rw-`?
There should be.

There should also be default entries for `user`, `group`, `mask`, and `other` too.

The entries for `user`, `group`, and `other` were automatically created based on the HDFS permissions for `user`, `group`, and `other`. The `mask` was automatically created based on the union of the `user:root:rw-` and `group::r-x` permissions.

3. Upload a copy of the local file `/etc/hosts` to the HDFS `/acltests` directory. Verify that the file was uploaded.

```
hdfs dfs -put /etc/hosts /acltests
hdfs dfs -getfacl /acltests/hosts
```

Did the file inherit the ACL `user:root:rw-` from the parent directory's default ACL entries?
It should have.

Look at the `mask::` that was calculated for the file. Does this access mask limit the effective permissions of any users or groups?
It should.

4. View the ACL entries for the `/acltests/group` file.

```
hdfs dfs -getfacl /acltests/group
```

Is there an ACL entry for `user:root:rw-`?
There should not be.

Default ACL entries on a directory only affect files that are added to the directory after the default ACL entries are created. The `group` file was created in the directory before the default ACL entries were created.

5. To test root user read access to the `hosts` file, assume normal root permissions again.

```
exit
```

6. Use the HDFS Shell *-cat* command to try to read the `/acltests/hosts` file.

```
hdfs dfs -cat /acltests/hosts
```

You should be able to read the `hosts` file as the root user.

7. Assume HDFS superuser permissions again.

```
su - hdfs
```

8. Create a `dir1` subdirectory in the `/acltests` directory. Verify that it was created in the correct location.

```
hdfs dfs -mkdir /acltests/dir1
hdfs dfs -ls /acltests
```

You should see the `dir1` directory listed.

Does the HDFS Shell *-ls* command output indicate that `dir1` has ACL entries? Do you see a + character?
You should have.

9. Display the ACL entries for the `dir1` directory.

```
hdfs dfs -getfacl /acltests/dir1
```

Do you see an ACL entry for `user:root:rw-`?
You should.

Do you see a list of default ACL entries?
You should.

Any files created in `dir1` will inherit its ACL entries. Any directories created in `dir1` will inherit not only its ACL entries, but also its default ACL entries.

Managing Access Using an Access Mask

Manage user and group access using the access mask.

1. While you still have HDFS superuser permissions, change the access mask for `/acltests/hosts` so that the unnamed group or any specific user or groups cannot read the `hosts` file. Verify the change once you have finished.

```
hdfs dfs -setfacl -m mask::--- /acltests/hosts
hdfs dfs -getfacl /acltests/hosts
```

What are the effective permissions for `user:root:` and `group::`?
There should be no effective permissions.

2. To test root user read access to the `hosts` file, assume normal root permissions again.

```
exit
```

3. As the root user, try to read the `/acltests/hosts` file.

```
hdfs dfs -cat /acltests/hosts
```

Were you able to read the file?
You should not have been able to.

Result

You have now:
Configured HDFS ACL support; created and managed file ACL entries, managed ACL entries and default ACL entries on directories; and managed user and group access using an access mask.

Lab 7: Managing Hadoop Storage

About This Lab

Objective:	To manage and monitor HDFS storage using the Ambari Web UI, NameNode UI, DataNode UI, and HDFS command-line commands
File locations:	N/A
Successful outcome:	**You will:** View and manage Ambari Web UI HDFS monitoring widgets; manage and monitor HDFS services and storage using the NameNode UI and command-line commands; check HDFS file system consistency using the *fsck* command; and view a DataNode block scanner report.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Managing Hadoop Storage***

Managing and Monitoring with Ambari

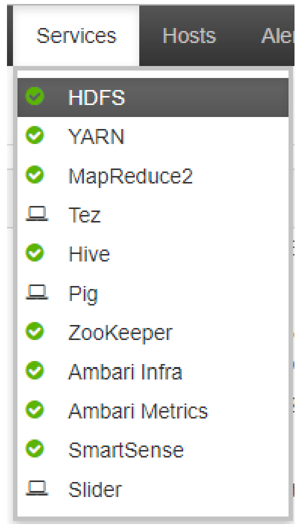
View, monitor, and manage the HDFS service and HDFS storage using the Ambari Web UI and NameNode UI.

Open the browser connect to the Ambari Server at

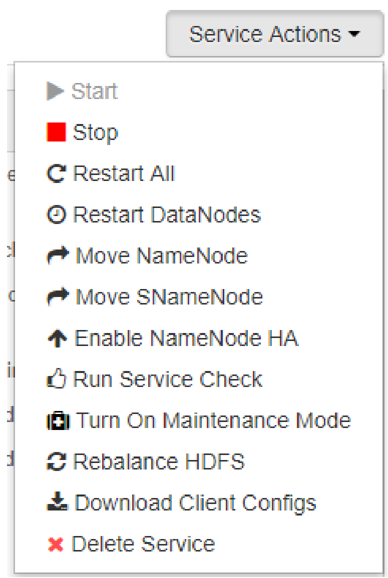
http://<Ambari Server Hostname>:8080

1. Login to the Ambari Web UI using the user name `admin` and the password `BadPass#1`.

2. In the Ambari Web UI, click **Services** and select **HDFS**.



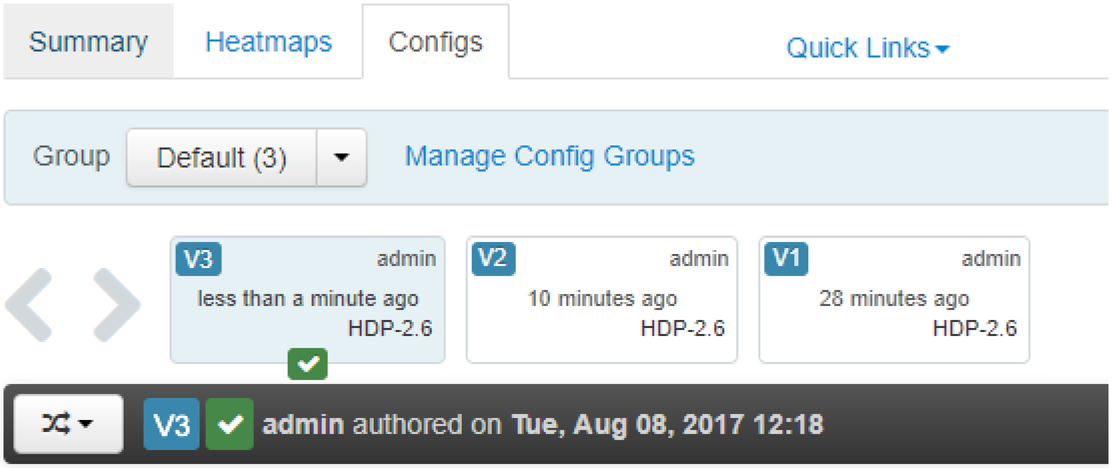
3 . Click to open and view the **Service Actions** menu button in the top right-side corner of the **Services** page.



Notice that the actions listed on the **Service Actions** menu button are specific to the currently selected service, in this case HDFS. Many of the actions listed here are performed other lab exercises.

4 . With **Services** and **HDFS** still selected, click **Configs** and then the **Advanced** tab.

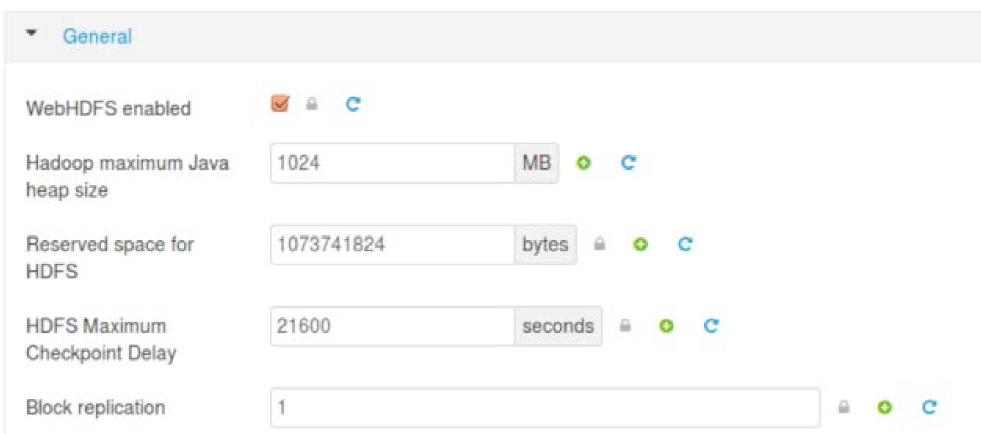
Ensure that the **HDFS Default** configuration group is selected.



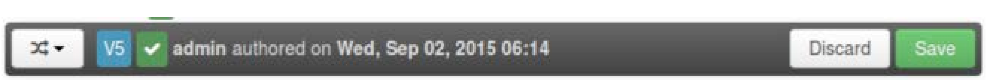
Scroll down to the **General** section. Change the minimum **Block replication** from 3 to 1.

This change means that HDFS will only require one copy of each data block on the system.

NOTE: This change is not retroactive to existing data blocks. It only affects new data blocks written after this change has been made.



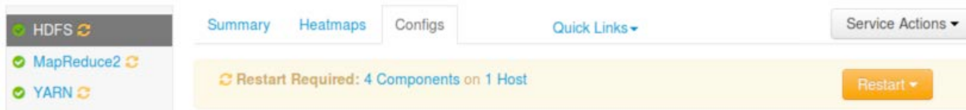
6 . Click **Save** to save your change.



Then click **Save** to confirm the change.

Click **OK** to dismiss the progress window.

7. After the configuration change, restart any affected services as indicated in the Ambari Web UI.



8. Use SSH to login to any cluster node.

   View the replication factor for the existing files in the root user's HDFS home directory.

```
hdfs dfs -ls /user/root
```

   The replication factor for the existing files should be three. Directory names are not replicated.

9. The replication factor of an existing file can be changed.

   Use the HDFS Shell -*setrep* command to change the `/user/root/hosts` file's replication factor from three to one. Verify the change once you are finished. Add another file to HDFS and its replication factor will be one as well.

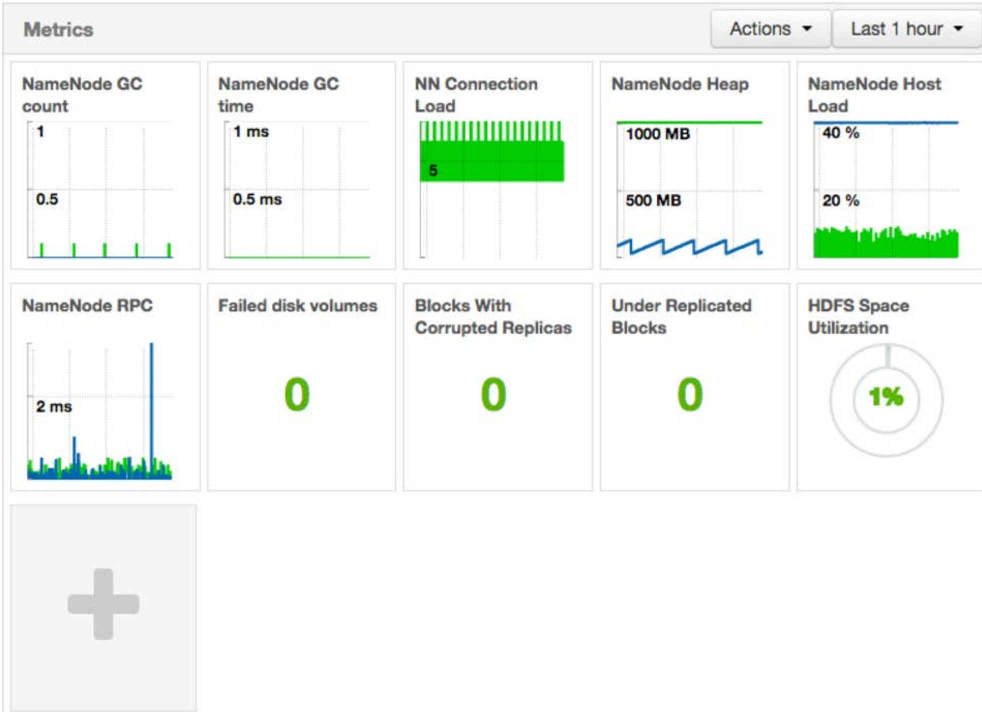
```
hdfs dfs -setrep 1 /user/root/hosts
hdfs dfs -put /etc/group /user/root/group
hdfs dfs -ls /user/root
```

   The replication factor of the `hosts and group` file should be one.

10. Go back to the Ambari Web UI.

    With **Services** and **HDFS** selected, ensure that you are on the **Summary page**.

    Scroll down to the **Metrics** section.



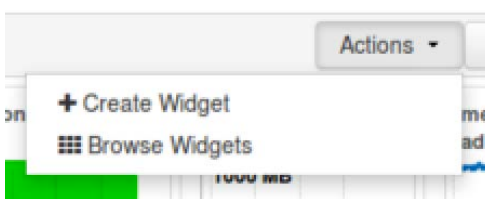
Notice the types of HDFS metrics that are available.

Notice the **Under Replicated Blocks** widget. Even though the minimum replication factor was reduced to one, there are HDFS files that were created before the replication factor was reduced and those files are configured with a replication factor of three. Changing the default HDFS replication factor is not retroactive to existing files and their data blocks.

11 . Float the mouse pointer over the different widgets to view the types of information that is displayed.

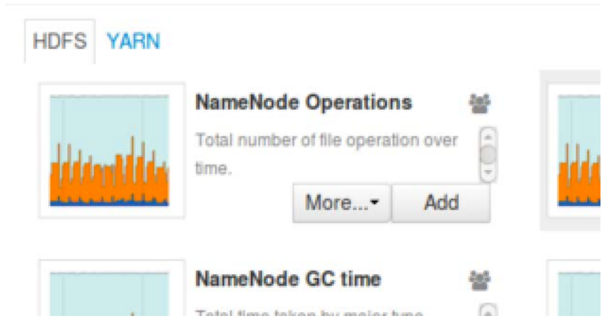
12 . In the Metrics section, click the **Actions** menu button.

Select **Browse Widgets**.

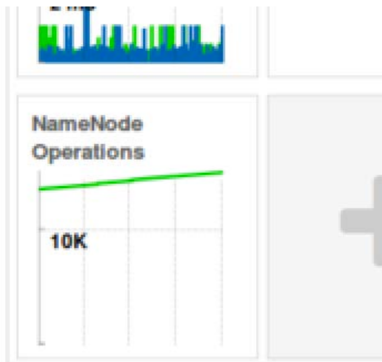


13 . On the **HDFS** tab, click **Add** to add the **NameNode Operations** widget.

Then click **Close**.

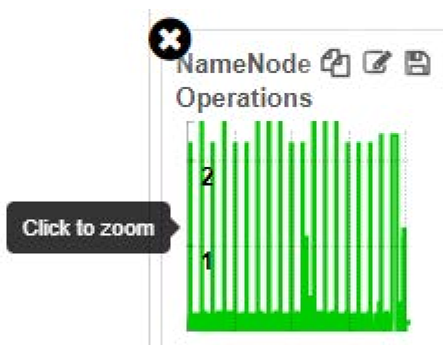


14 . Verify that the NameNode Operation widget is displayed in the Metrics section.



15 . Float the mouse pointer over the NameNode Operations widget.

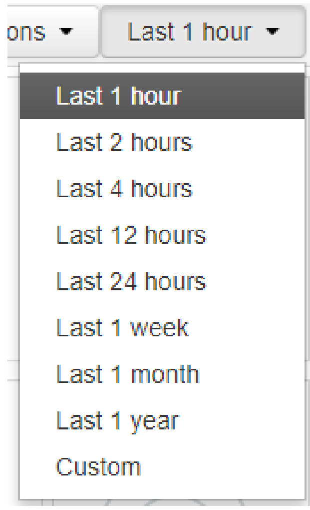
Notice and click the circled **X** icon to delete the widget from the **Metrics section**.



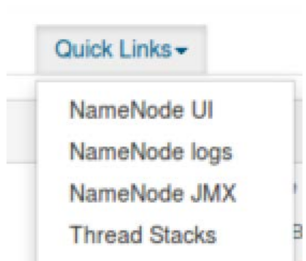
The widget should have been removed.

16 . Click the **Last 1 hour** menu button and notice the available list of time periods.

The name of the menu button will change depending on which option is currently selected.



17 . With **Services** and **HDFS** selected, click **Quick Links** at the top of the page and then select **NameNode UI**.



18 . Using the information displayed in the NameNode UI, answer the following questions:

What date and time was HDFS last started?

Is safemode on or off?

What percentage of your HDFS (DFS) is used?

How many under-replicated blocks are there?

What directory path on the NameNode holds the `fsimage` and `edits` files?
(it is called the NameNode Storage directory in the NameNode UI)

What is the name of the `fsimage` file that was used the last time HDFS was started?

Managing and Monitoring with Commands

Manage and monitor the HDFS service and HDFS storage using HDFS command-line commands.

1 . Use SSH to login to any cluster node.

2. Use the *dfsadmin* command to generate an HDFS report.

```
hdfs dfsadmin -report
```

Did the command work? Why not?

3. Assume HDFS superuser permissions and try to generate a *dfsadmin* report again.

```
su - hdfs
hdfs dfsadmin -report
```

Did it work this time?
It should have.

4. Using the *dfsadmin* report output, answer the following questions:

How much HDFS storage space is configured?

How much HDFS storage space is still available?

How many under-replicated block are there?

How many live DataNodes are there?

Does the *dfsadmin -report* command provide some of the same information as the NameNode UI?

5. Assume normal root user privileges again.

```
exit
```

6. Use the HDFS Shell *-du* (disk usage) command to view HDFS storage usage.

```
hdfs dfs -du /
```

Did the command fail?
It should have.

7. Run the HDFS Shell *-du* command again, but examine only the HDFS `/user/root` directory.

```
hdfs dfs -du /user/root
```

Did the command work this time?
It should have because the root user has the necessary HDFS permissions on its own home directory.

8. Run the HDFS Shell *-du* command again, but add the summary and human-readable command options.

   NOTE: The effect on the displayed output.

```
hdfs dfs -du -h /user/root
hdfs dfs -du -s /user/root
hdfs dfs -du -h -s /user/root
```

9. Run the HDFS Shell *-df* (display file system) command.

```
hdfs dfs -df
```

   What percentage of the available HDFS storage space has been used?

10. Run the HDFS Shell *-df* command again but add the *-h* human-readable command option.

   NOTE: The effect on the displayed output.

```
hdfs dfs -df -h
```

Checking HDFS Consistency

Check HDFS file system consistency.

1. Run the *fsck* command on the HDFS / directory.

```
hdfs fsck /
```

   Did it work?

   Why not?

2. Assume HDFS superuser permissions and try to run the *fsck* command again.

```
su - hdfs
hdfs fsck /
```

   Did it work this time?
   It should have.

3. Assume normal root user privileges again.

```
exit
```

4. Create a big file.

```
xfs_mkfile 200m bigfile
```

5. Upload the local file `bigfile` to the root user's HDFS home directory and then verify that it was uploaded.

```
hdfs dfs -put bigfile /user/root
hdfs dfs -ls /user/root
```

6. Run the *fsck* command again, but run it only on the `/user/root/bigfile` file.

```
hdfs fsck /user/root/bigfile
```

View the command output and answer the following questions:

What are the total number of blocks in the file?
It should be two.

What is the default replication factor for this file?
It should be one.

Are there any under-replicated blocks for this file?
There should not be because the `bigfile` *file was created after the default replication factor was changed to one.*

7. Examine the differences in the displayed output when using different combinations of *fsck* command options. Run the *fsck* command again on the `/user/root/bigfile` file, but this time add the *-files* option.

```
hdfs fsck /user/root/bigfile -files
```

Was the file name added near the beginning of the command output?
The answer should be yes.

8. Run the same command again but this time add the *-files -blocks* options.

```
hdfs fsck /user/root/bigfile -files -blocks
```

What was added to the information displayed by the command?

Do you see a block number?
You should. (It is prefaced by `blk_`*.)*

9. Run the same command again but this time add the *-files -blocks -locations* options.

```
hdfs fsck /user/root/bigfile -files -blocks -locations
```

What was added to the information displayed by the command?

Following the block number information do you have a DataNode IP address and port number inside square brackets?
You should.

10. Use the Linux *find* command to find the local file system file that contains the HDFS data block.

    Use the example syntax below but substitute your actual block number as seen in the output of the previous *fsck* command.

```
find / -name "*blk_107374XXXX*"
```

NOTE: The path to the data block. Does it begin with the path `/hadoop/hdfs/data`?
It should.

11. The HDFS property `dfs.datanode.data.dir` contains one or more local file system paths that are the parent directories of the files containing the HDFS data blocks. Find and view the `dfs.datanode.data.dir` property in the `/etc/hadoop/conf/hdfs-site.xml` file.

```
more /etc/hadoop/conf/hdfs-site.xml
```

Does the path name in the property match the beginning of the path name you saw in the previous lab step?
It should.

Viewing a Block Scanner Report

View a DataNode block scanner report.

1. Open a new tab in the browser and type the following URL:

```
http://<External AWS Hostname of Any cluster node>:50075/blockScannerReport
```

How many blocks were scanned in the current period?
It might be zero.

If there is not any activity recorded, you might need to determine if the block scanner is enabled.

2. To determine if the block scanner is enabled, search the `/etc/hadoop/conf/hdfs-site.xml` file for the `dfs.datanode.scan.period.hours` property.

   A negative value means that the block scanner is disabled while a positive value means that it is enabled.

```
more /etc/hadoop/conf/hdfs-site.xml
```

Did you find the property?
You should not have.

This means that the property has been set in the `hdfs-default.xml` file that is normally packaged as part of a JAR file. To find the default value of an HDFS property you will need to know the version of HDFS that is in use in your cluster.

One way to find your HDFS version number is to use the the Ambari Web UI to access the NameNode UI by selecting **Services** > **HDFS** > **Summary** > **Quick Links** > **NameNode UI.**

The HDFS version is displayed on the NameNode UI **Overview** tab.

In this example the version number is 2.7.3.

3. Use a Web browser to view the default value of an HDFS property by viewing the Apache documentation for the `hdfs-default.xml` file. You can perform a Google search for `hdfs-default.xml` but many of the Google links might not point to the version of the file that matches your version of HDFS. Examine any URL returned by Google and modify the version number within that URL, if necessary, to match your HDFS version. For example, the following URL displays the `hdfs-default.xml` property settings for HDFS version 2.7.3.

```
http://hadoop.apache.org/docs/2.7.3/hadoop-project-dist/hadoop-hdfs/hdfs-default.xml
```

In the 2.7.3 version of the `hdfs-default.xml` file, the default value for the `dfs.datanode.scan.period.hours` property is 504 hours, or 3 weeks. So, the block scanner is enabled but might not have performed any scanning yet.

Result

You have now:
Viewed and managed Ambari Web UI HDFS monitoring widgets; managed and monitored HDFS services and storage using the NameNode UI and command-line commands; checked HDFS file system consistency using the *fsck* command; and viewed a DataNode block scanner report.

Lab 8: Managing HDFS Quotas

About This Lab

Objective:	To configure, test, and monitor HDFS storage name and space quotas
File locations:	N/A
Successful outcome:	**You will:** Create a directory, assign name and space quotas to it, and upload multiple files to the directory until both the space and name quotas are exceeded.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Managing Hadoop Storage***

Creating a Directory

Create a directory.

1 Use SSH to login to any cluster node.

2 Assume HDFS superuser permissions.

HDFS superuser permissions are required to configure HDFS storage name and space quotas.

```
su - hdfs
```

3 Use the HDFS Shell *-mkdir* command to create a directory to test name and space quotas.

Verify that the directory was created.

```
hdfs dfs -mkdir /qtests
hdfs dfs -ls /
```

You should see the `/qtests` directory listed.

Assigning Quotas

Assign name and space quotas to the newly created directory.

1 You will be using the root user account to test the HDFS name and space quotas.

Use the HDFS Shell *-chown* command to change the ownership of the HDFS `/qtests` directory to the root user.

Verify the ownership change once you are finished.

```
hdfs dfs -chown root:root /qtests
hdfs dfs -ls /
```

Does the root user own the HDFS `/qtests` directory?
It should.

2 Use the *dfsadmin -setQuota* command to assign a name quota to the HDFS `/qtests` directory. Set the name quota to 4 for testing purposes.

```
hdfs dfsadmin -setQuota 4 /qtests
```

3 Use the *dfsadmin -setSpaceQuota* command to assign a space quota to the HDFS `/qtests` directory. Set the space quota to `134352500` for testing purposes.

```
hdfs dfsadmin -setSpaceQuota 134352500 /qtests
```

4 Display the current quotas and quota usage for the HDFS `/qtests` directory. (A wide terminal window will make viewing command output easier.)

```
hdfs dfs -count -v -q /qtests
```

Use the command output to answer the following questions:

Do you see a name quota limit of 4?
You should.

Did the directory name itself consume a part of the name quota?
It should have.

So you see a space quota of 134,352,500 bytes?
You should.

Did the directory name consume any of the space quota?
It should not have.

Testing Quotas

Upload multiple files to the directory until both the space and name quotas are exceeded.

1 Assume normal root user privileges again in order to test the name and space quotas.

```
exit
```

2 Upload the local file `constitution.txt` to HDFS as `/qtests/file1`.

Verify the upload once you have finished.

```
hdfs dfs -put constitution.txt /qtests/file1
hdfs dfs -ls /qtests
```

3 Display the current quotas and quota usage for the HDFS `/qtests` directory.

```
hdfs dfs -count -v -q /qtests
```

Use the command output to answer the following questions:

What is the remaining name quota?
It should be 2.

Is there at least enough space for a default data block of 128 MB (134,217,728)?
There should be. This means that a file that is a default block size or less can still be written to the directory.

4 Again, upload the local file `constitution.txt` to HDFS as `/qtests/file2`.

This file will consume less than a data block and should upload.

Verify the upload once you have finished.

```
hdfs dfs -put constitution.txt /qtests/file2
hdfs dfs -ls /qtests
```

5 Again, display the current quotas and quota usage for the HDFS `/qtests` directory.

```
hdfs dfs -count -v -q /qtests
```

Is there still at least enough space for a default data block of 128 MB (134,217,728)?
There should be. This means that a file that is a default block size or less can still be written to the directory.

6 This time try to upload a file that will consume more space than is allowed by the space quota.

Upload the `bigfile` file as `/qtests/file3`.

```
hdfs dfs -put bigfile /qtests/file3
```

Did it fail?
It should have because the file upload exceeded the space quota.

7 Again, upload the local file `constitution.txt` to HDFS as `/qtests/file3`.

This file will consume less than a data block and should upload.

Verify the upload once you have finished.

```
hdfs dfs -put constitution.txt /qtests/file3
hdfs dfs -ls /qtests
```

Did it upload?
It should have.

8 Again, display the current quotas and quota usage for the HDFS `/qtests` directory.

```
hdfs dfs -count -v -q /qtests
```

How much of the name quota is remaining?

It should be zero which means that no more files or directories can be created in the HDFS `/qtests` *directory.*

9 Again, attempt to upload the local file `constitution.txt` to HDFS as `/qtests/file4`.

```
hdfs dfs -put constitution.txt /qtests/file4
```

Did it upload?
It should have failed and displayed a message about exceeding the name quota.

10 Remember that the HDFS replication factor is currently set to one. The default replication factor is normally three. For each file, these two additional data block replicas would consume some of the available space quota.

Result

You have now:
Created a directory, assigned name and space quotas to it, and uploaded multiple files to the directory until both the space and name quotas were exceeded.

Lab 9: Configuring Rack Awareness

About This Lab

Objective:	To use the Ambari Web UI to configure rack awareness
File locations:	N/A
Successful outcome:	**You will:** View default rack awareness information; replace `/default-rack` with `/rack01` on the first host, `/rack02` on the second host and `/rack03` on the third host, and view the results of the new configuration.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Configuring Rack Awareness***

Viewing Rack Awareness Information

View the default rack awareness configuration information.

1. Use SSH to login to any cluster node.

2. List the rack awareness files in the `/etc/hadoop/conf` directory.

```
ls /etc/hadoop/conf
```

Do you see the `topology_mappings.data` and `topology_script.py` files?
You should.

3. View the contents of the `/etc/hadoop/conf/topology_mappings.data` file.

```
cat /etc/hadoop/conf/topology_mappings.data
```

What is the rack name assigned to all three DataNodes?
It should be /default-rack.

4. Assume HDFS superuser permissions.

```
su - hdfs
```

5. Run the HDFS *fsck* command with the *-racks* option.

```
hdfs fsck -racks
```

What is the number of racks that is reported?
It should be one.

6. Run the HDFS *dfsadmin* command with the *-report* option.

```
hdfs dfsadmin -report
```

You should see a `Name:` reported for each DataNode.

Do you see a `Rack:` name reported for any DataNode?
You should not.

7. Remain logged in with HDFS superuser permissions.

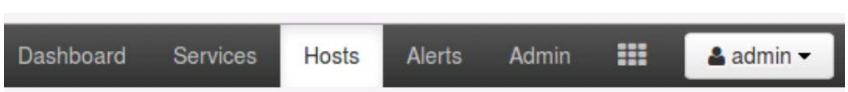
Configuring Rack Awareness

Configure rack awareness using the Ambari Web UI.

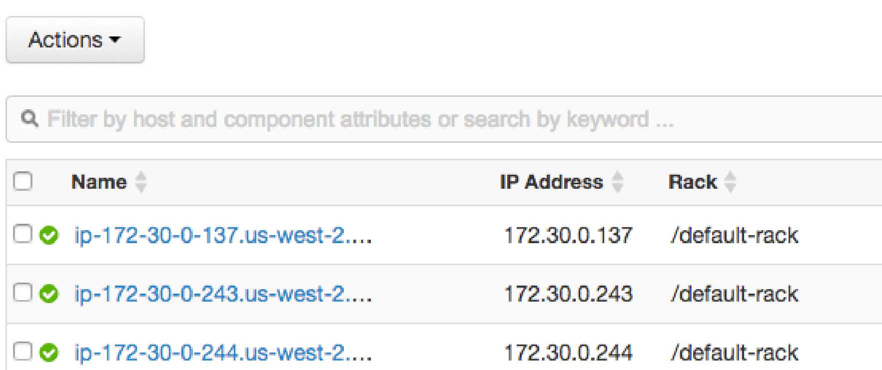
1. Open the browser and connect to the Ambari Server at the URL

   http://<Ambari Server Hostname>:8080

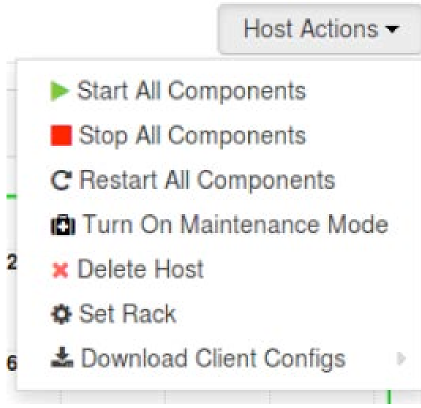
2. Login to the Ambari Web UI using the user name `admin` and the password BadPass#1.
3. Click **Hosts** in the Ambari Web UI.



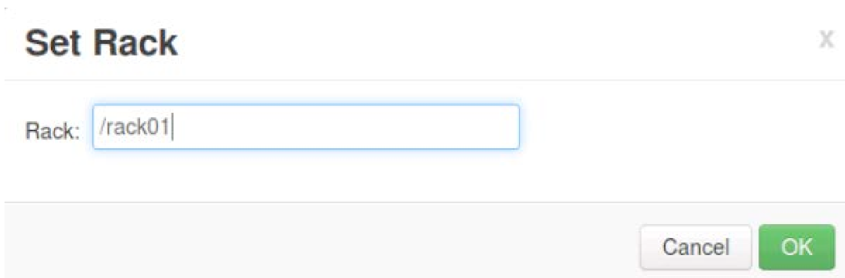
4. Click **on the first name** in the list of nodes.



5. Click the **Host Actions** menu button and select **Set Rack**.

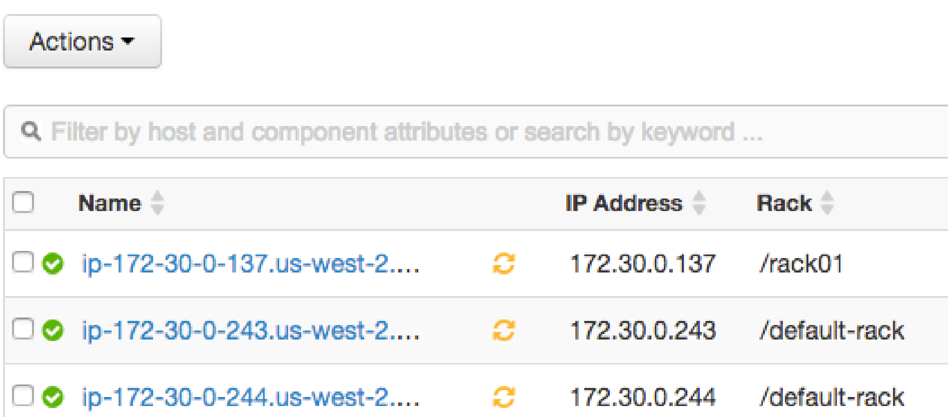


6 . Configure this host's with a rack name of `/rack01`, and click **OK**.

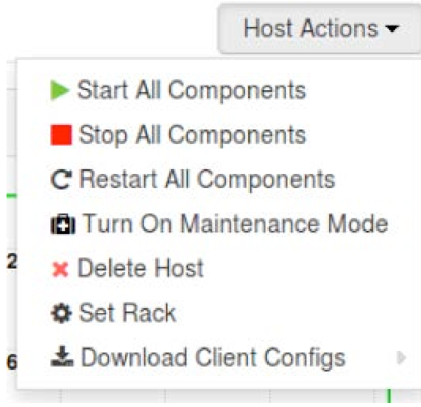


7 . Click **Back** to return to the list of cluster nodes.

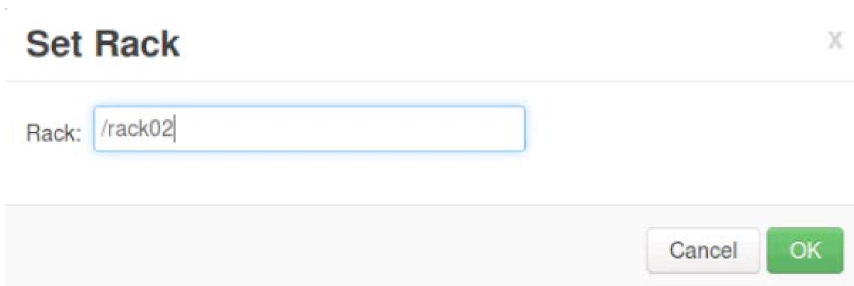
8 . Click **next name** in the list of nodes.



9 . Click the **Host Actions** menu button and select **Set Rack**.

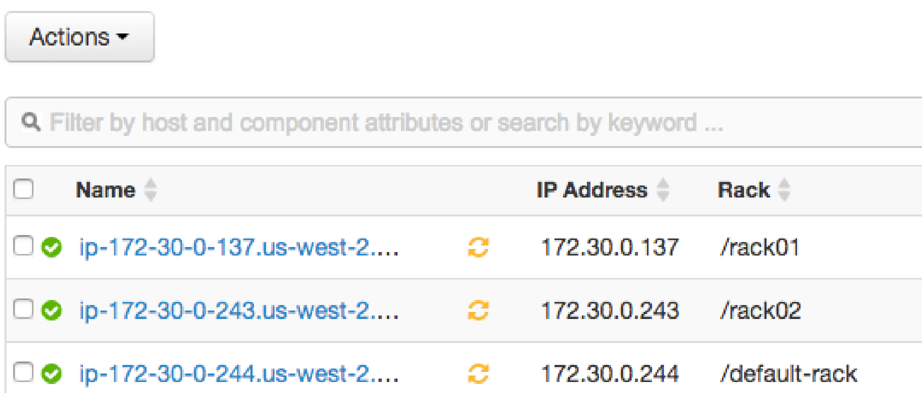


10 . Configure this host's rack name as `/rack02`, and click **OK**.

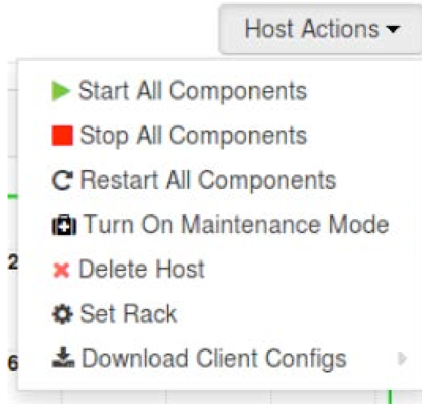


11 . Click **Back** to return to the list of cluster nodes.

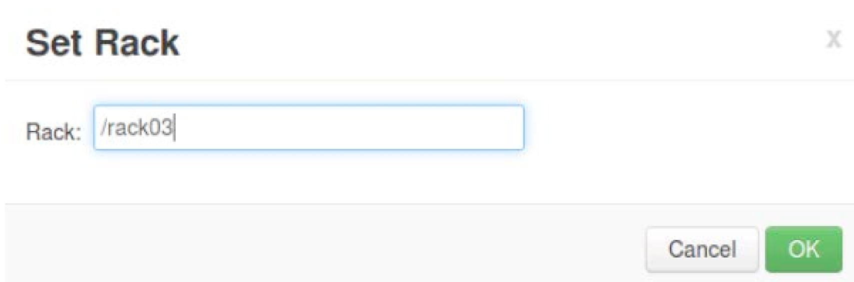
12 . Click **next name** in the list of nodes.



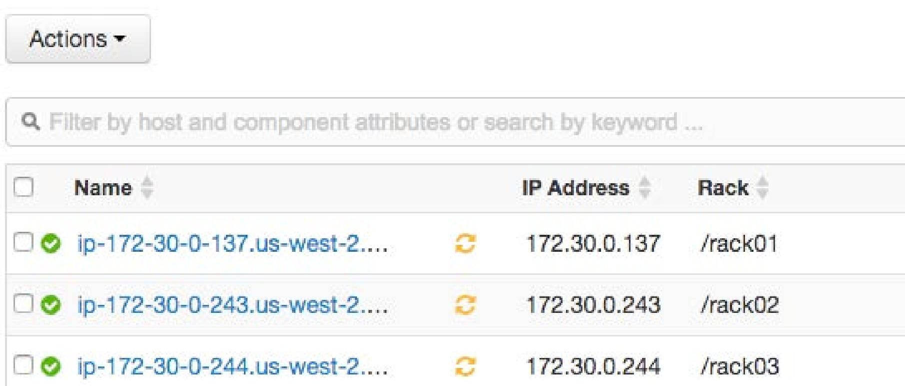
13 . Click the **Host Actions** menu button and select **Set Rack**.



14 . Configure this host's rack name as `/rack03`, and click **OK**.

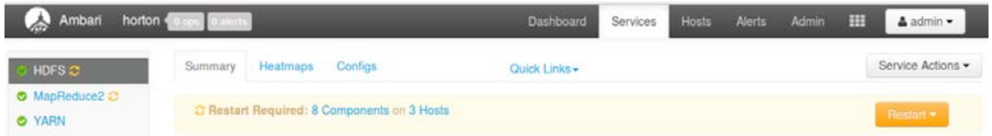


15 . Click **Back** to return to the list of cluster nodes.



16 . Click **Services** and restart any service that requires restarting as indicated in the Ambari Web UI.

HDFS and MapReduce2 should be indicated.



Viewing Results of the Configuration

View the results of configuring rack awareness.

1. Go back to the terminal window logged into at the beginning of this lab.

   View the contents of the `/etc/hadoop/conf/topology_mappings.data` file.

```
cat /etc/hadoop/conf/topology_mappings.data
```

   What are the rack names of your hosts?
   They should be `/rack01`, `/rack02`, and `/rack03`.

2. You should still have HDFS superuser permissions in the terminal window.

   Run the HDFS *fsck* command with the *-racks* option again.

```
hdfs fsck -racks
```

   What is the number of racks that is reported?
   It should be three.

3. Run the *dfsadmin* command again with the *-report* option.

```
hdfs dfsadmin -report
```

   Is the rack name for each DataNode reported?
   It should be.

4. Assume normal root user permissions again.

```
exit
```

Result

You have now:
Viewed default rack awareness information; replaced `/default-rack` with `/rack01, /rack02` and `/rack03` on your hosts; and viewed the results of the new configuration.

Lab 10: Managing HDFS Snapshots

About This Lab

Objective:	To use HDFS command-line utilities and NameNode UI to create and manage HDFS snapshots
File locations:	N/A
Successful outcome:	**You will:** Enable an HDFS directory for snapshots and then create, view, rename and delete a snapshot.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Protecting a Cluster with Backups***

Enabling Snapshots

Enable HDFS snapshots on the /user/root directory.

1. Use SSH to login to any cluster node.

2. Assume HDFS superuser permissions in order to enable HDFS snapshots.

```
su - hdfs
```

3. Use the *dfsadmin* command with the *-allowSnapshot* option to enable snapshots on the `/user/root` directory.

   When you are finished, use the *lsSnapshottableDir* command to view the list of snapshottable HDFS directories.

```
hdfs dfsadmin -allowSnapshot /user/root
hdfs lsSnapshottableDir
```

4. Assume normal root user permissions again.

```
exit
```

Managing Snapshots

Create, view, rename, explore, and delete an HDFS snapshot.

1. Use the HDFS Shell *-ls* command to determine if there are any HDFS snapshots on the `/user/root` directory.

```
hdfs dfs -ls /user/root/.snapshot
```

There should not be any snapshots.

2. Create an HDFS snapshot of the `/user/root` directory using the HDFS Shell *-createSnapshot* command.

```
hdfs dfs -createSnapshot /user/root
```

Notice that you did not use the command option to explicitly name the snapshot, so HDFS assigned the snapshot a default name. The default name is based on the current date and time.

3. Use the HDFS Shell *-ls* command again to determine if there are any HDFS snapshots on the `/user/root` directory.

```
hdfs dfs -ls /user/root/.snapshot
```

You should see a snapshot listed.

4. Use the HDFS Shell *-renameSnapshot* command to rename the `/user/root` snapshot.

Replace the variable name in the command below with your actual snapshot name.

```
hdfs dfs -renameSnapshot /user/root s<yyyymmdd-hhmmss.sss> mysnap
```

You should not see any output from the command.

5. Use the HDFS Shell *-ls* command again to determine if the snapshot was renamed.

```
hdfs dfs -ls /user/root/.snapshot
```

You should see the new `mysnap` *snapshot name.*

6. Use the HDFS Shell *-ls* command to display the contents of the snapshot directory `mysnap`.

Also use the *-ls* command to display the contents of the `/user/root` directory.

How do they compare?


```
hdfs dfs -ls .snapshot/mysnap
hdfs dfs -ls /user/root
```

The two directories should have *identical content.*

7. Use the HDFS Shell *-put* command to upload the local file `/usr/hdp/<VERSION NUMBER>/hadoop-mapreduce/hadoop-mapreduce-examples.jar` to the HDFS `/user/root` directory.

```
hdfs dfs -put /usr/hdp/<VERSION NUMBER>/hadoop-mapreduce/hadoop-mapreduce-
examples.jar /user/root/hadoop-mapreduce-examples.jar
```

You should see no output from this command.

8. Use the HDFS Shell *-ls* command to display the contents of the snapshot directory `mysnap`.

   Also use the *-ls* command to display the contents of the `/user/root` directory.

   How do they compare?

```
hdfs dfs -ls .snapshot/mysnap
hdfs dfs -ls /user/root
```

The `hadoop-mapreduce-example.jar` *file is present only in the* `/user/root` *directory.*

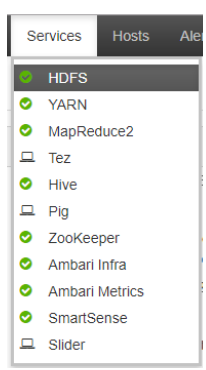
The `hadoop-mapreduce-example.jar` *file should not be present in the* `mysnap` *snapshot.*

Open your browser and connect to the Ambari Server at the URL:

http://<Ambari Server Hostname>:8080

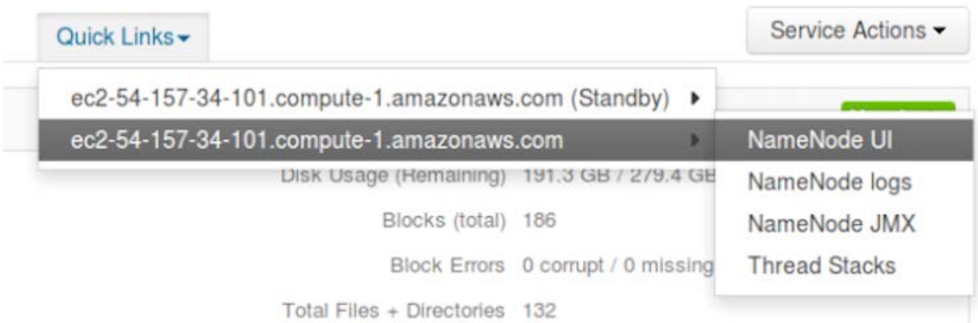
9. Login to the Ambari Web UI using the user name `admin` and the password BadPass#1.

10. Click **Services** and select **HDFS**.

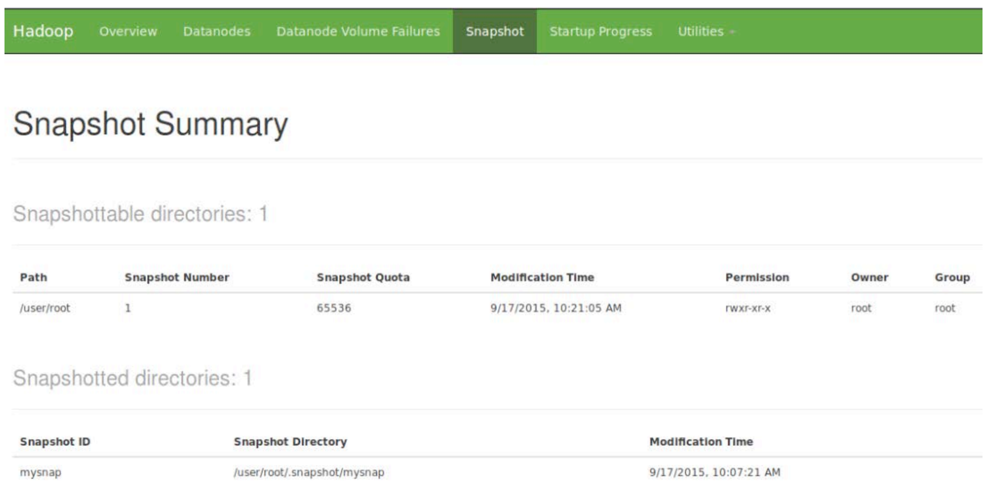


11 . Click **Quick Links**.
Select the active NameNode system, and then select **NameNode UI**.

The NameNode UI opens in a new browser tab.



12 . Click **Snapshot** to view HDFS snapshot information on the **Snapshot** page.



How many snapshottable directories are there?

Which directory is snapshottable?

Are there any current snapshots?

What is the name of the current snapshot?

13 . Move back to the terminal window that is logged in to at the beginning of this lab.

14 . Use the HDFS Shell command *-deleteSnapshot* to delete the `mysnap` snapshot. Then verify that the snapshot has been removed.

```
hdfs dfs -deleteSnapshot /user/root mysnap
hdfs dfs -ls /user/root/.snapshot
```

The snapshot should be gone.

Result

You have now:
Enabled an HDFS directory for snapshots and then created, viewed, renamed and deleted a snapshot.

Lab 11: Using DistCp

About This Lab

Objective:	To use the *DistCp* command and options to copy HDFS files and directories
File locations:	N/A
Successful outcome:	**You will:** View online manual information for the *distcp* command; configure a *DistCp* test environment; and copy files and directories using the *distcp* command and options.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Protecting a Cluster with Backups***

Viewing Online DistCp Information

View the online manual information for the *distcp* command.

1. Use SSH to login to any cluster node.

2. View the online manual page for *distcp*.

```
hadoop distcp -help
```

Configuring a DistCp test environment.

Configure a DistCp test environment on your cluster

1. Use the HDFS Shell *-put* command to upload the any file from `/usr/share/licenses` directory to the HDFS `/user/root` directory. Example below, however any text file will do.

   Then verify that the file has been uploaded.

```
hdfs dfs -put /usr/share/licenses/systemd-219/LICENSE.GPL2
/user/root/LICENSE.txt
hdfs dfs -ls
```

2. Use the HDFS Shell *-mkdir* command to create an HDFS source directory named `/user/root/sourcedir`.

   Then verify that the directory was created.

```
hdfs dfs -mkdir /user/root/sourcedir
hdfs dfs -ls
```

3. Use the HDFS Shell *-put* command to upload other files from `/usr/share/licenses` for example `/usr/share/licenses/systemd-219/LICENSE.LGPL2.1` and `/usr/share/licenses/system-219/LICENSE.MIT` to the HDFS `/user/root/sourcedir` directory. **Example below, again any files will do**.

   Then verify that the files were uploaded.

```
hdfs dfs -put /usr/share/licenses/systemd-219/LICENSE.LGPL2.1
/user/root/sourcedir
hdfs dfs -put /usr/share/licenses/systemd-219/LICENSE.MIT /user/root/sourcedir
hdfs dfs -ls /user/root/sourcedir
```

4. Use the HDFS Shell *-mkdir* command to create an HDFS target directory named `/user/root/targetdir`.

   Then verify that the directory was created.

```
hdfs dfs -mkdir /user/root/targetdir
hdfs dfs -ls
```

5. Use the HDFS Shell *-mkdir* command to create a second HDFS target directory named `/user/root/targetdir2`.

   Then verify that the directory was created.

```
hdfs dfs -mkdir /user/root/targetdir2
hdfs dfs -ls
```

Copying Files and Directories

Copy files and directories to another HDFS directory using *DistCp* and *DistCp* options.

1. Copy the `/user/root/`LICENSE.txt file to the `/user/root/targetdir` directory using the *distcp* command.

   Then verify that the file was copied.

```
hadoop distcp hdfs://<NameNode Hostname>:8020/user/root/LICENSE.txt
hdfs://<NameNode Hostname>:8020/user/root/targetdir
hdfs dfs -ls targetdir
```

   There should be a LICENSE.txt *file in* `targetdir`*.*

2. Use the HDFS Shell *-ls* command to list the contents of the `/user/root/targetdir/sourcedir` directory.

```
hdfs dfs -ls /user/root/sourcedir
```

You should see the other two files you uploaded `LICENSE.LGPL2.1` *and* `LICENSE.MIT` *files.*

Notice that when copying directories with DistCp, the default behavior is to copy the source directory and its files to the target directory.

3. Copy the `/user/root/sourcedir` directory again, but this time copy it to the `/user/root/targetdir2` directory.

   Then verify that the `sourcedir` directory and its two files were copied.

```
hadoop distcp hdfs://<NameNode Hostname>:8020/user/root/sourcedir
hdfs://<NameNode Hostname>:8020/user/root/targetdir2
hdfs dfs -ls /user/root/targetdir2
hdfs dfs -ls /user/root/targetdir2/sourcedir
```

You should see the `sourcedir` *directory in the* `targetdir2` *directory.*
You should also see the `LICENSE.LGPL2.1` *and* `LICENSE.MIT` *files in the* `sourcedir` *directory.*

4. Use the HDFS Shell *-rm -R* command to remove the `sourcedir` directory from the `targetdir2` directory.

```
hdfs dfs -rm -R /user/root/targetdir2/sourcedir
hdfs dfs -ls /user/root/targetdir2
```

The `targetdir2` *directory should be empty.*

5. Again, copy the `/user/root/sourcedir` directory to the `/user/root/targetdir2` directory but this time add the *-update* option.

   List the contents of the `targetdir2` directory when the *distcp* command has finished.

```
hadoop distcp -update hdfs://<NameNode Hostname>:8020/user/root/sourcedir
hdfs://<NameNode Hostname>:8020/user/root/targetdir2
hdfs dfs -ls targetdir2
```

With the addition of the *-update* option, the `LICENSE.LGPL2.1` *and* `LICENSE.MIT` files were copied directly into the `targetdir2` directory. The `sourcedir` directory was not copied.

This same behavior will be seen when the *-overwrite* option is used.

6. Again, copy the `/user/root/sourcedir` directory to the `/user/root/targetdir2` directory but this time add the *-overwrite* option.

   List the contents of the `targetdir2` directory when the *distcp* command has finished.

```

```

```
hadoop distcp -overwrite hdfs://<NameNode Hostname>:8020/user/root/sourcedir
hdfs://<NameNode Hostname>:8020/user/root/targetdir2
hdfs dfs -ls targetdir2
```

With the addition of the *-overwrite* option, the `LICENSE.LGPL2.1` *and* `LICENSE.MIT` files were copied directly into the `targetdir2` directory again. They were copied even though they had not changed.

Once again, the `sourcedir` directory was not copied because the *-overwrite* option was used.

Result

You have now:
Viewed online manual information for the *distcp* command; configured a DistCp test environment; and copied files and directories using the *distcp* command and options.

Lab 12: Configuring HDFS Storage Policies

About This Lab

Objective:	Assign the DISK storage type to DataNodes disks and assign a storage policy to an HDFS directory
File locations:	N/A
Successful outcome:	The DISK storage type is assigned to the DataNode disks and the WARM storage policy is assigned to `/user/root/warmdata`.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Configuring Heterogeneous HDFS Storage***

Lab Steps

Perform the following steps:

1. **Create an HDFS directory and data to use in the lab exercise.**

    a. Use SSH to login to any cluster node.

    b. Use HDFS command-line commands to create a `warmdata` directory in the root user's HDFS home directory. Verify that the directory was created.

```
hdfs dfs -mkdir /user/root/warmdata
hdfs dfs -ls
```

    c. Copy the local `/etc/passwd` file into the new HDFS `warmdata` directory. Verify that the file was copied.

```
hdfs dfs -put /etc/passwd warmdata
hdfs dfs -ls warmdata
```

    d. Leave the terminal window open.

2. **Assign the DISK storage type label to the DataNode disks using the Ambari Web UI.**

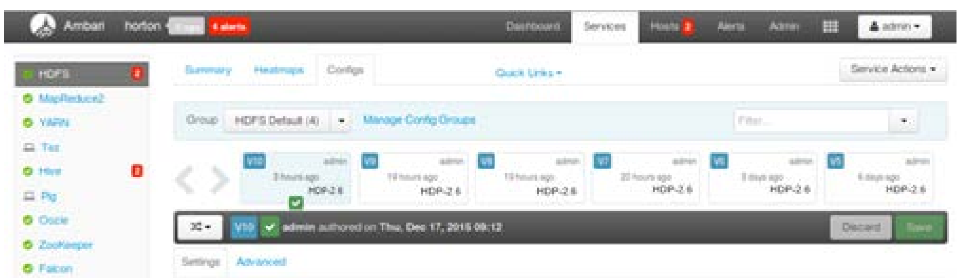
NOTE: The lab environment has only three DataNodes, which will all share the same storage type label configuration. This means that it is not technically necessary to create a new HDFS configuration group in order to add storage type labels to the DataNodes. However, because in the real world this is how different sets of DataNodes would be configured with different storage type labels, it is done in the lab to enable you to practice to necessary configuration steps. Also, DISK is the default storage type but once again, it is explicitly assigned to the DataNode disks in the lab to practice the configuration steps.

Open the browser and connect to the Ambari Server at the URL

http://<Ambari Server Hostname>:8080

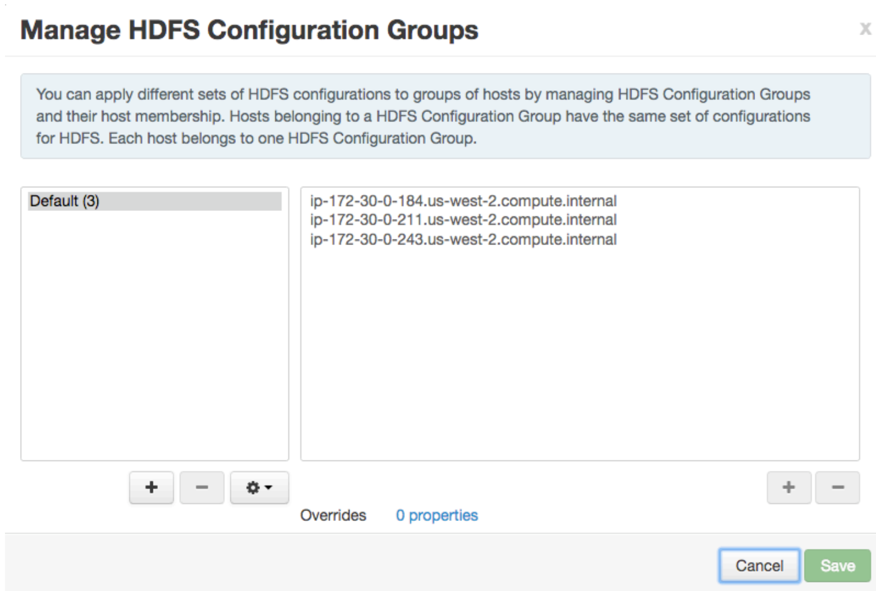
a. Login to the Ambari Web UI using the user name `admin` and the password `BadPass#1`.

b. Browse to **Services** > **HDFS** > **Configs** > **Settings**.

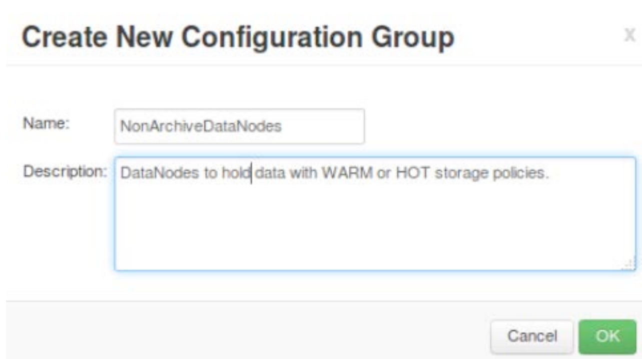


c. Click **Manage Config Groups** to open the Manage HDFS Configuration Groups window.

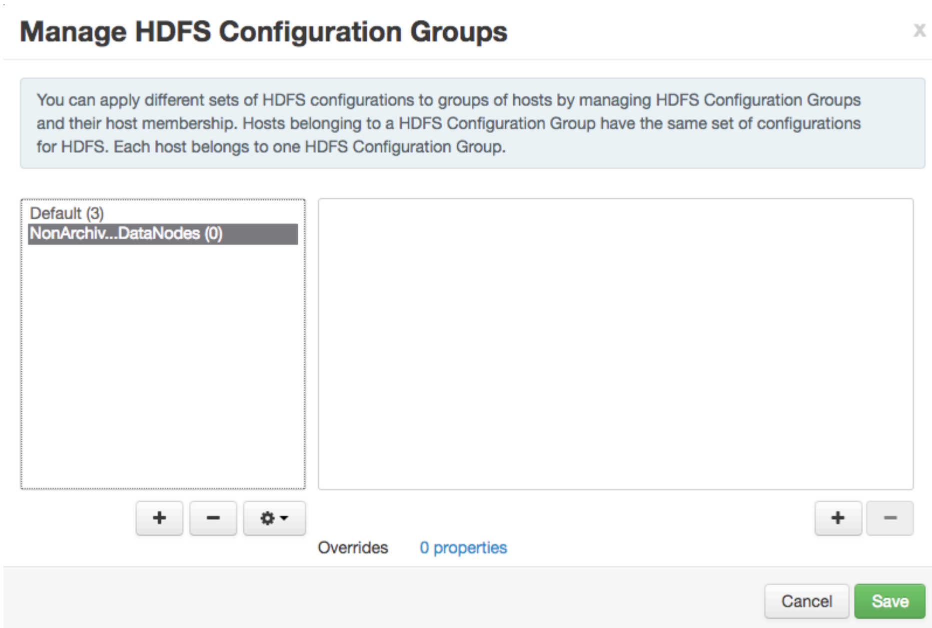
Click the left-hand side plus icon in the Manage HDFS Configuration Groups window to create a new HDFS configuration group.



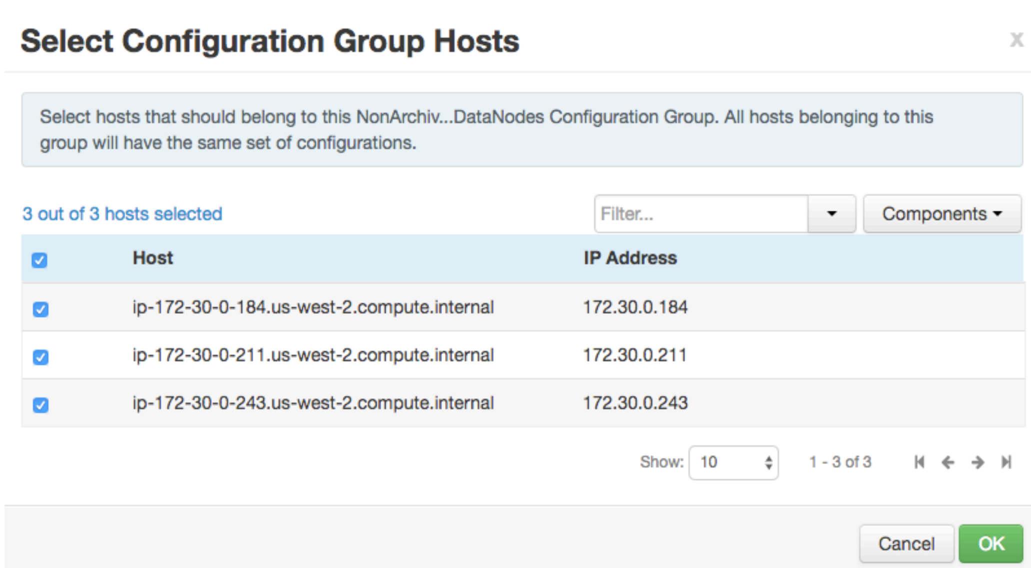
e. Type the name `NonArchiveDataNodes` and optionally enter a description for the new HDFS configuration group, then click **OK**.



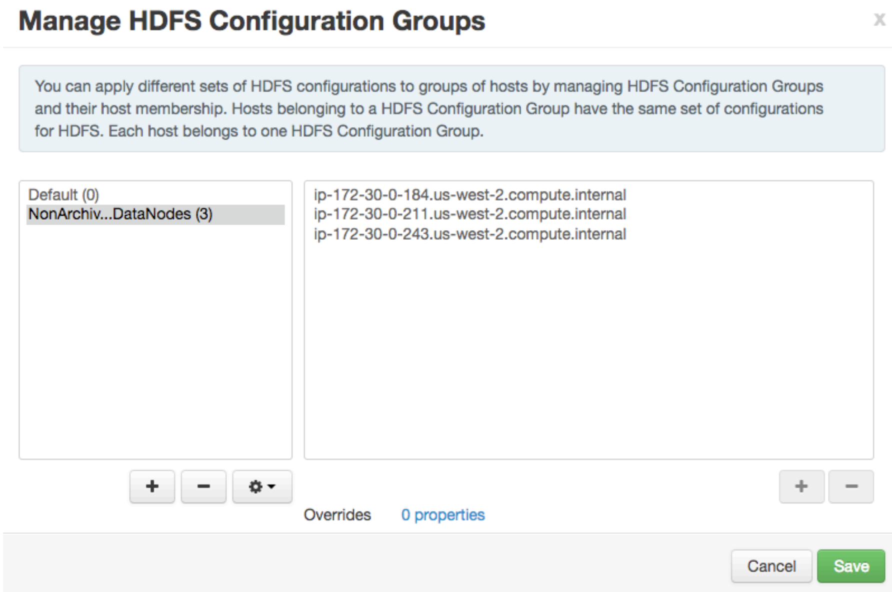
f. Click **NonArchiveDataNodes** and then click the right-hand side plus icon to add DataNodes to the new HDFS configuration group.



g. Select the check boxes for **all your nodes** to add them to the NonArchiveDataNodes configuration group. Then click **OK**.

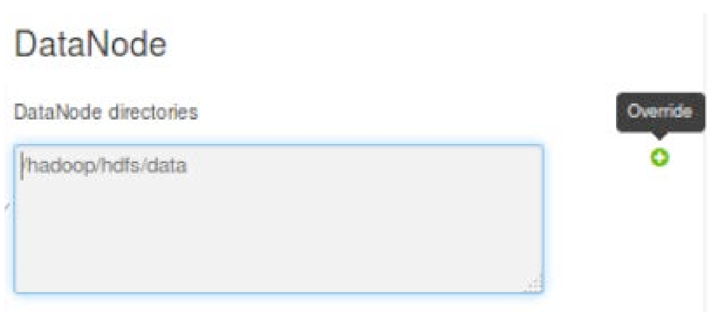


h. View the result and then click **Save**.

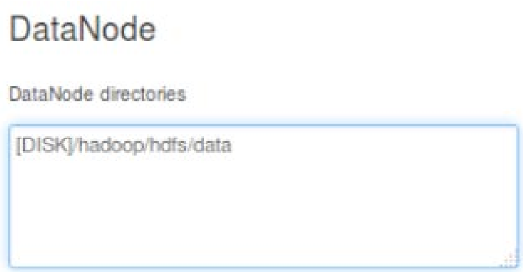


i. Ensure that the **NonArchiveDataNodes** group is selected in the **Group** drop-down menu. This ensures that the upcoming configuration change is applied to the `/etc/hadoop/conf/hdfs-site.xml` file on the correct DataNodes.

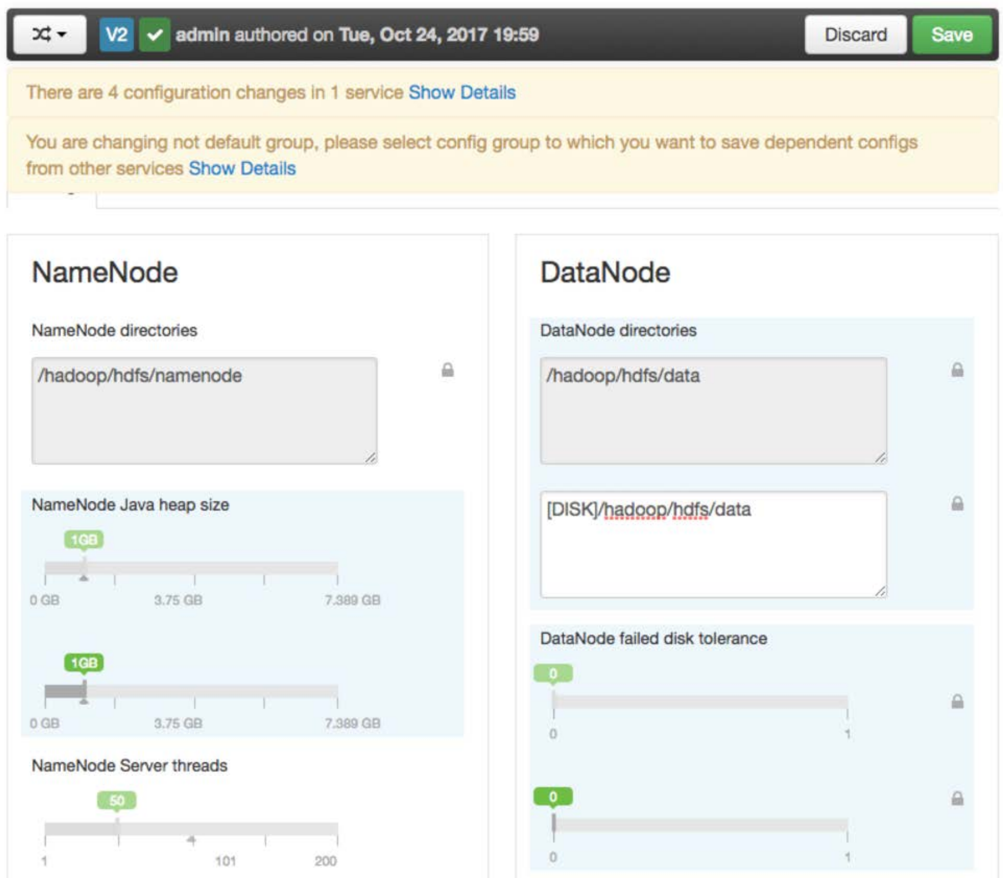
j. In the DataNode section, find the **DataNode directories** property. Click the **Override** icon to associate the change with the DataNodes assigned to a specific configuration group. (You will need to float the mouse pointer or click the text box for the **Override** icon to appear.)



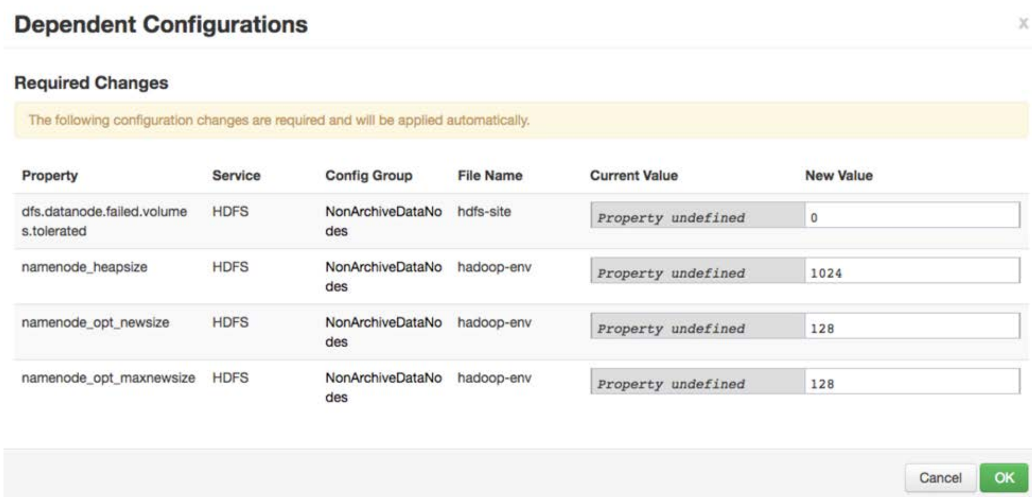
k. In the lower and initially red-colored property text box, modify the directory path value by prepending the storage type label `[DISK]`.



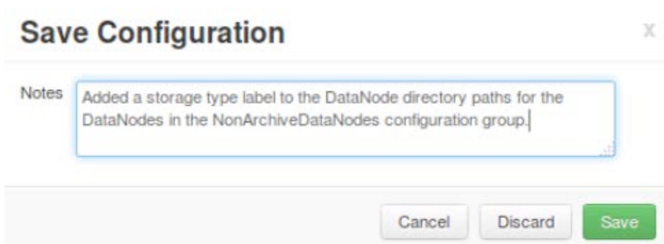
l. Ambari will associate additional configuration change as shown in blue. Click **Save** to save the configuration change.



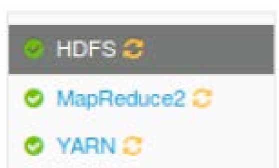
m. A Dependent Configuration pop-up window will appear, review and click OK.



n. In the Save Configuration Changes window, click **OK**.



o. Restart any services indicated in the Ambari Web UI.



3. **Assign a storage policy to an HDFS directory.**

a. Return to the open terminal window logged in to at the beginning of this lab.

b. List the storage policies available in HDFS.

```
hdfs storagepolicies -listPolicies
```

How many storage policies are listed?
What are their names?


```
For the storage policy ALL_SSD, what type of storage is used to write the
first data block if there is no available space left on the SSD device?
```

c. Assign the WARM storage policy to the data in the `/user/root/warmdata` directory.

```
hdfs storagepolicies -setStoragePolicy -path /user/root/warmdata -policy WARM
```

Did the command report any failure message? It should not have.

d. List the storage policy assigned to the `/user/root/warmdata` directory.

```
hdfs storagepolicies -getStoragePolicy -path /user/root/warmdata
```

What policy is assigned to the directory?
Why do you think the storage policy assignment failed?

e. Become the HDFS superuser and once again attempt to assign the WARM storage policy to the `/user/root/warmdata` directory.

```
su - hdfs
hdfs storagepolicies -setStoragePolicy -path /user/root/warmdata -policy WARM
```

f. List the storage policy assigned to the `/user/root/warmdata` directory.

```
hdfs storagepolicies -getStoragePolicy -path /user/root/warmdata
```

Did the command work this time? It should have.

g. Remain logged in with HDFS superuser privileges.

4. **Examine the disk locations and storage types for data blocks controlled by a storage policy.**

   a. Run the HDFS `fsck` utility to display the block locations and their storage types for the `/user/root/warmdata/passwd` file.

```
hdfs fsck /user/root/warmdata/passwd -files -blocks -locations
```

What are the storage types listed for the file's data block replicas?

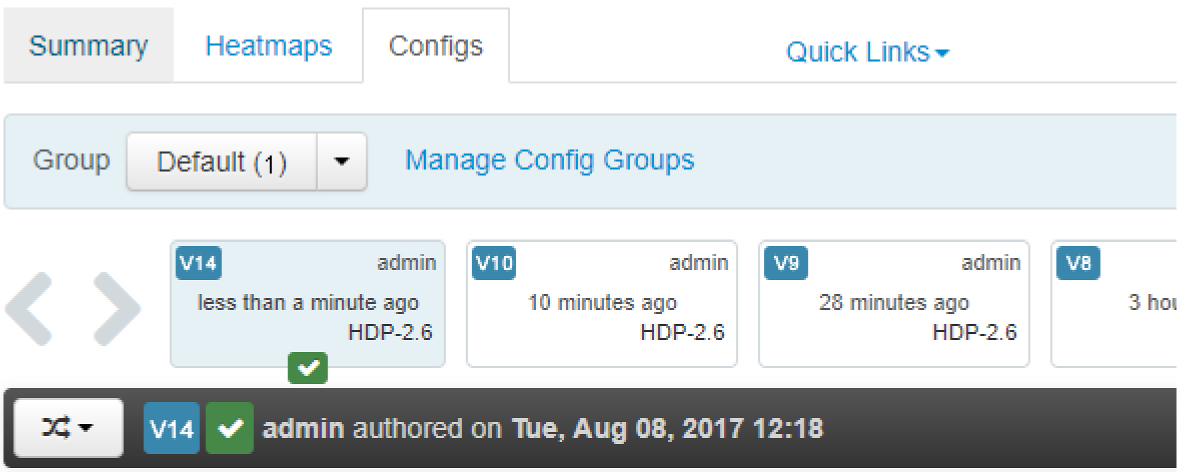
Remember that while the data blocks appear, and actually are, on the correct storage types, this is only because there are no other storage types available. In the real world where there might be multiple DataNodes offering various storage types, the data blocks might have to be moved to the appropriate storage types after assigning a new storage policy.

b. Use the `exit` command to return to normal root user privileges.

```
exit
```

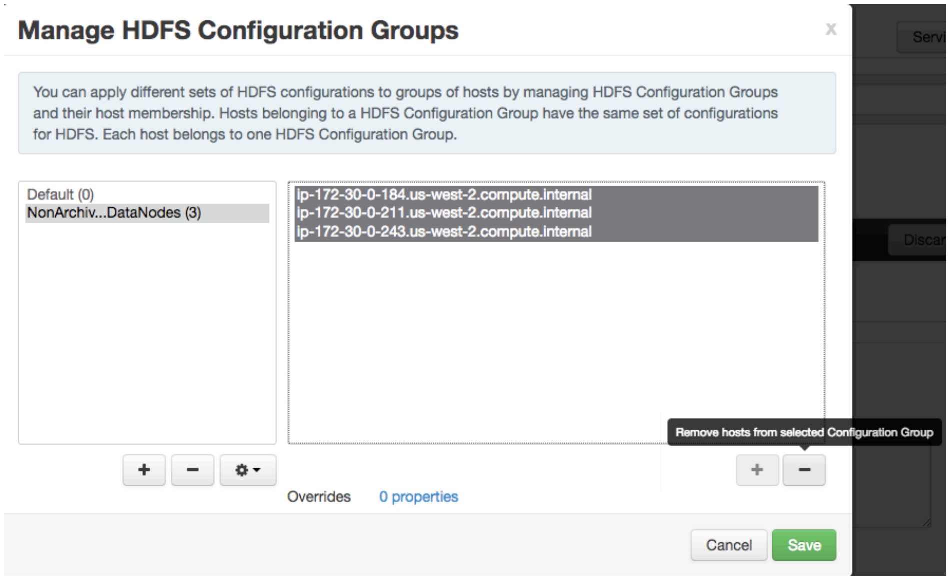
5. **To simplify HDFS configuration in future labs, revert the HDFS configuration back to a single Default Configuration Group.**

a. In the Ambari Web UI, browse to **Services** > **HDFS** > **Configs**.

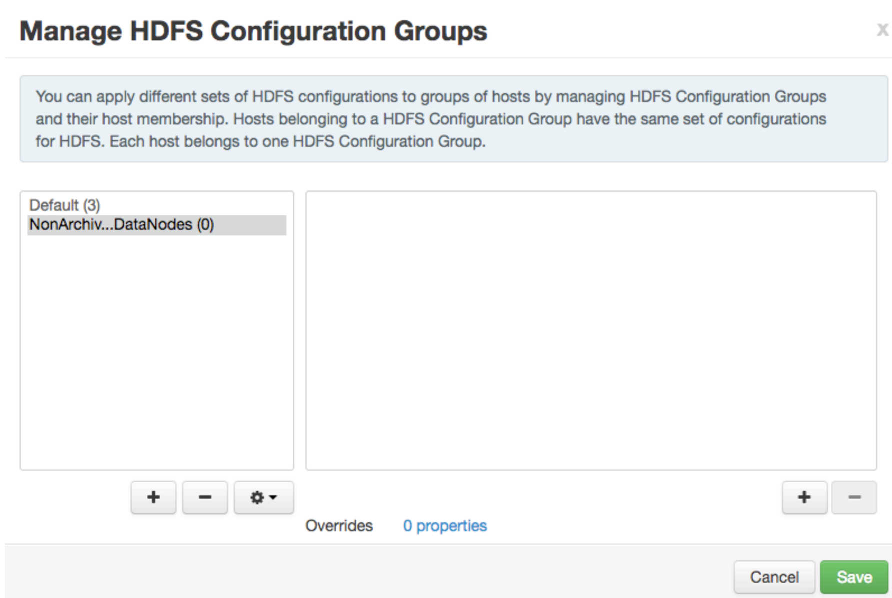


b. Click **Manage Config Groups**.

c. In the Manage HDFS Configuration Groups window, click to select the **NonArchiveDataNodes** group. Then using your Shift key and the mouse pointer, select all three nodes in the NonArchiveDataNodes group. Then click the right-hand side minus button to remove the nodes from the group.

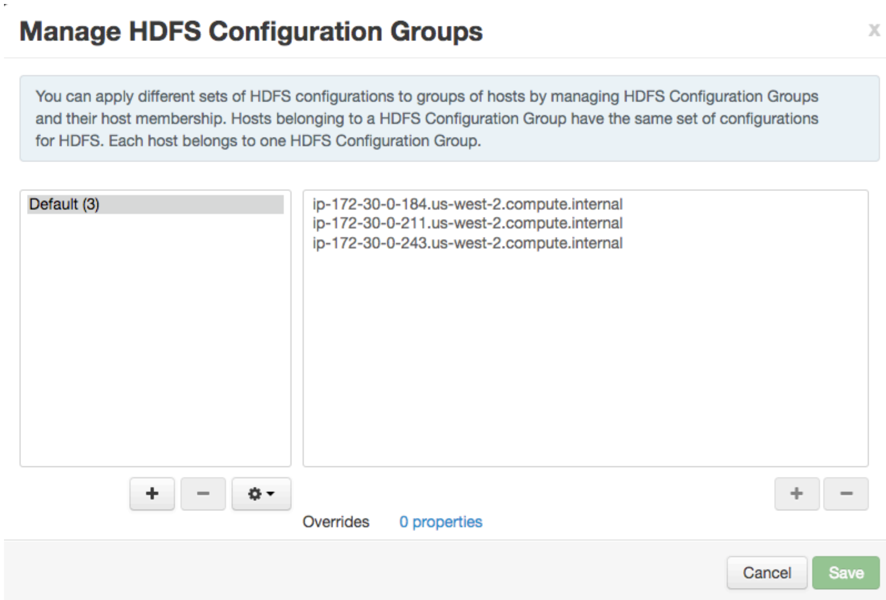


d. Ensure that the **NonArchiveDataNodes** group is selected and then click the left-hand side minus button to remove the configuration group.



e. In the Confirmation window, click **OK**.

f. View the result, which should be your nodes are reassigned to the HDFS Default group. Then click **Save**.



g. Restart all services indicated in the Ambari Web UI.

Result

The HDFS configuration group NonArchiveDataNodes contained your nodes and was used to assign the DISK storage type to the DataNode disks. The storage policy WARM was assigned to the HDFS directory `/user/root/warmdata`. At the end of the lab the HDFS NonArchiveDataNodes configuration group should be deleted and all DataNodes should be assigned to the HDFS Default configuration group.

Lab 13: Configuring HDFS Centralized Cache

About This Lab

Objective:	Create, list, modify, and remove a Cache Pool and a Cache Directive
File locations:	N/A
Successful outcome:	A Cache Pool with two Cache Directives will be created, modified, and removed.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Configuring HDFS Centralized Cache***

Lab Steps

Perform the following steps:

1. **Prepare the lab environment.**

    a. Use SSH to login to any cluster node.

    b. Add a new Linux user to the node. Use your first name as the new user's name.

```
useradd -m <your_first_name>
```

    c. Create a new HDFS home directory for the new Linux user.

```
su - hdfs
hdfs dfs -mkdir /user/<your_first_name>
```

    d. Copy two files to the new HDFS home directory and then change the ownership of the directory and the files to the new user. Only the size, but not the content of the files, is important for this lab. Then verify that the files were copied, renamed, and the file ownerships are correct.

```
hdfs dfs -put /etc/group /user/<your_first_name>/lookuptable
hdfs dfs -put /etc/hosts /user/<your_first_name>/jointable
hdfs dfs -chown -R <your_first_name>:<your_first_name> /user/<your_first_name>
hdfs dfs -ls /user/<your_first_name>
```

e. Return to normal root user privileges.

```
exit
```

2. **Configure the maximum amount of per-DataNode memory that can be used by HDFS caching.**

   a. In the open terminal window as the root user, use the `ulimit` command to determine how much memory the Linux operating system has reserved for locked memory.

```
ulimit -l
```

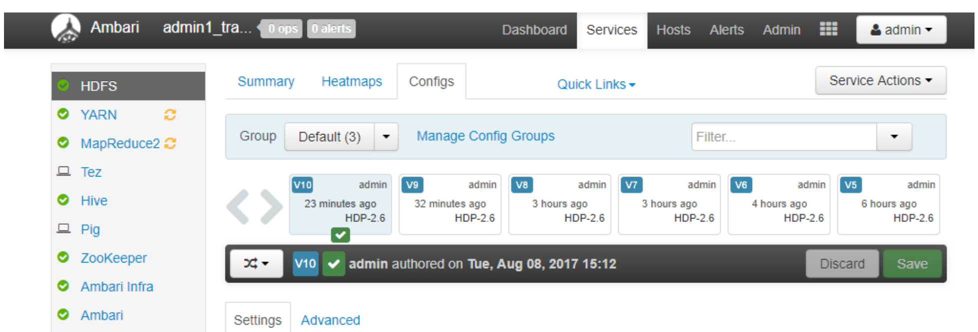
Remember that the number displayed is the number of kilobytes.

Open the browser and connect to the Ambari Server at the URL

http://<Ambari Server Hostname>:8080

   b. Login to the Ambari Web UI using the user name `admin` and the password `BadPass#1`.

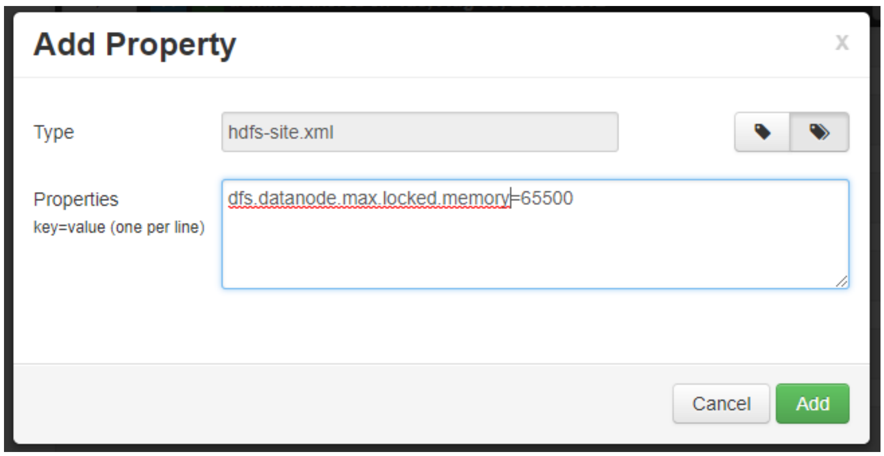
   c. Browse to **Services** > **HDFS** > **Configs** > **Advanced**.



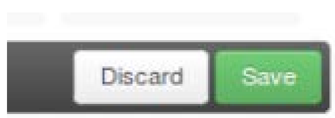
   d. Scroll down and expand **Custom hdfs-site**. Then click **Add Property**.

e. In the Add Property window, add the `dfs.datanode.max.locked.memory` property. Assign it a value slightly less than the 64 kilobyte `ulimit -l` value seen earlier. For example, use the value `65500`.

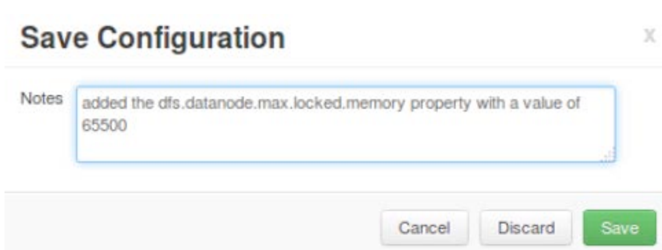
This will be sufficient memory for the limited amount of caching used in this lab exercise. Then click **Add**.



f. Click **Save** to save the configuration change.



g. In the Save Configuration window, enter optional notes and click **Save**.



h. Then click **OK**.

i. Restart HDFS and any other needed services as indicated in the Ambari Web UI.

3. **Create a Cache Pool that is writeable for a single user account.**

   a. Return to the open terminal window logged in as root.

   b. Create a new Cache Pool that is owned and writeable by a normal user and only readable by everyone else. The Cache Pool should be limited to 65500 cached bytes and enforce a maximum time-to-live value of 24 hours.

```
hdfs cacheadmin -addPool <your_first_name>Pool -owner <your_first_name> -mode
644 -limit 65500 -maxTtl 24h
```

Did the command succeed? It should not have.
What was the reported problem?

   c. Repeat the previous step again, but first assume HDFS superuser privileges.

```
su - hdfs
hdfs cacheadmin -addPool <your_first_name>Pool -owner <your_first_name> -mode
644 -limit 65500 -maxTtl 24h
```

Did the command work this time? It should have.

   d. List all existing Cache Pools.

```
hdfs cacheadmin -listPools
```

   e. List only the one existing Cache Pool and add the option that includes current usage information in the output.

```
hdfs cacheadmin -listPools <your_first_name>Pool -stats
```

Why are the values for BYTES_NEEDED and FILES_NEEDED zero?

   f. Modify the Cache Pool’s maximum time to live to five days. Then verify the change.

```
hdfs cacheadmin -modifyPool <your_first_name>Pool -maxTtl 5d
hdfs cacheadmin -listPools <your_first_name>Pool
```

   g. Resume normal root user privileges.

```
exit
```

4. **Add Cache Directives to the new Cache Pool.**

    a. Use a Cache Directive to add (cache) the HDFS file `/user/<your_first_name>/lookuptable` to the <your_first_name>Pool.

```
hdfs cacheadmin -addDirective -path /user/<your_first_name>/lookuptable -pool
<your_first_name>Pool
```

Did the command succeed? It should not have.
What was the reported problem?

    b. Repeat the previous step again, but first assume the normal <your_first_name> user's privileges.

```
su - <your_first_name>
hdfs cacheadmin -addDirective -path /user/<your_first_name>/lookuptable -pool
<your_first_name>Pool
```

Did the command succeed? It should have.

    c. List the Cache Directives in your only Cache Pool.

```
hdfs cacheadmin -listDirectives -pool <your_first_name>Pool
```

Was any usage information reported in the output? There should not be any usage information.
What is the time-to-live value?
How far is that in the future (use the `date` command to print the current date and time)?
Why is the time-to-live set to this date and time?

    d. Repeat the previous command but add the option that includes usage information in the output.

```
hdfs cacheadmin -listDirectives -pool <your_first_name>Pool -stats
```

What are the values of BYTES_CACHED and FILES_CACHED?
If the values are zero, why?
What do you do to “fix” these values so they are non-zero?

    e. List the usage information for your only Cache Pool.

```
hdfs cacheadmin -listPools  <your_first_name>Pool
```

Did the command work for the normal user? It should have.
Why did the command work for the normal user?
Does the Cache Pool usage information match the Cache Directive's usage information? It should.

f. Use a Cache Directive to add (cache) the HDFS file `/user/<your_first_name>/jointable` to the <your_first_name>Pool. Then verify that the command succeeded.

```
hdfs cacheadmin -addDirective -path /user/<your_first_name>/jointable -pool
<your_first_name>Pool
hdfs cacheadmin -listDirectives -pool <your_first_name>Pool
```

Did the command succeed?
It should have.

What is the replication factor for the `jointable` file?
What is the directive's ID number? Remember it.

g. Change the replication factor of the cached `jointable` file to two. Then verify the change.

```
hdfs cacheadmin -modifyDirective -id <directiveID> -replication 2
hdfs cacheadmin -listDirectives -pool <your_first_name>Pool
```

Has the replication factor changed? It should have.

h. Resume normal root user privileges.

```
exit
```

5. **Remove a Cache Directive.**

   a. At the root user, list the Cache Directives in the <your_first_name>Pool.

```
hdfs cacheadmin -listDirectives -pool <your_first_name>Pool
```

Did the command succeed? It should have.

   b. Remove the Cache Directive for the `jointable` file using the Directive's path.

```
hdfs cacheadmin -removeDirectives -path /user/<your_first_name>/jointable
```

Did the command succeed? It should not.
What problem was indicated by the error message?

c. Repeat the previous command but this time assume the Cache Directives owner's privileges first.

```
su - <your_first_name>
hdfs cacheadmin -removeDirectives -path /user/<your_first_name>/jointable
```

Did the command succeed? It should have.

6. **Remove a Cache Pool**

a. As the `<your_first_name>` user, remove the Cache Pool.

```
hdfs cacheadmin -removePool <your_first_name>Pool
```

Did the command succeed? It should not have.
What problem was indicated by the error message?

b. Assume the HDFS superuser privileges.

```
exit
su - hdfs
```

c. How many Cache Directives remain in the Cache Pool?

```
hdfs cacheadmin -listDirectives -pool <your_first_name>Pool
```

d. Remove the Cache Pool.

```
hdfs cacheadmin -removePool <your_first_name>Pool
```

Was the Cache Pool removed even though it contained a Cache Directive? It should have been.

e. Assume normal root user privileges again.

```
exit
```

Result

A Cache Pool with two Cache Directives was added to the system. The Cache Pool and Cache Directive were modified. A Cache Directive and the Cache Pool were removed.

Lab 14: Configuring an NFS Gateway

About This Lab

Objective:	Use the Ambari Web UI to configure an HDFS NFS Gateway
File locations:	N/A
Successful outcome:	The selected HFS Gateway server will NFS export the HDFS file system which will allow any other server to NFS mount HDFS locally.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Managing the HDFS NFS Gateway***

Lab Steps

Perform the following steps:

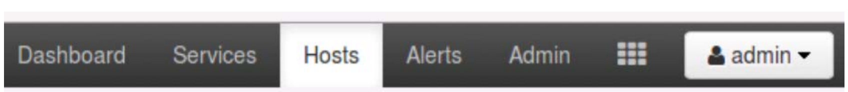
1. **Install and configure an HDFS NFS Gateway using the Ambari Web UI.**

   Open the browser and connect to the Ambari Server at the URL

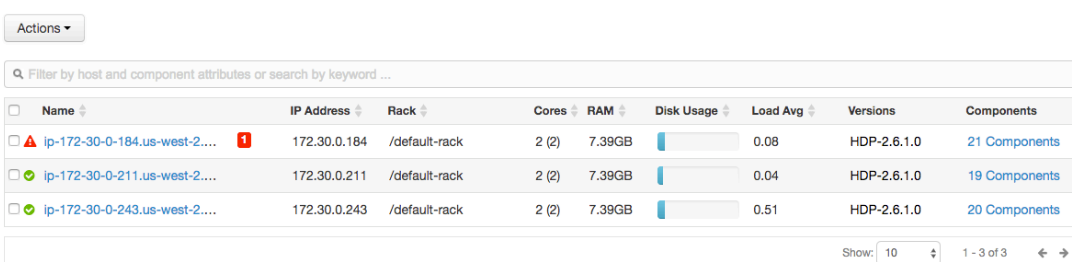
   http://<Ambari Server Hostname>:8080

   a. Login to the Ambari Web UI using the user name `admin` and the password `BadPass#1`.

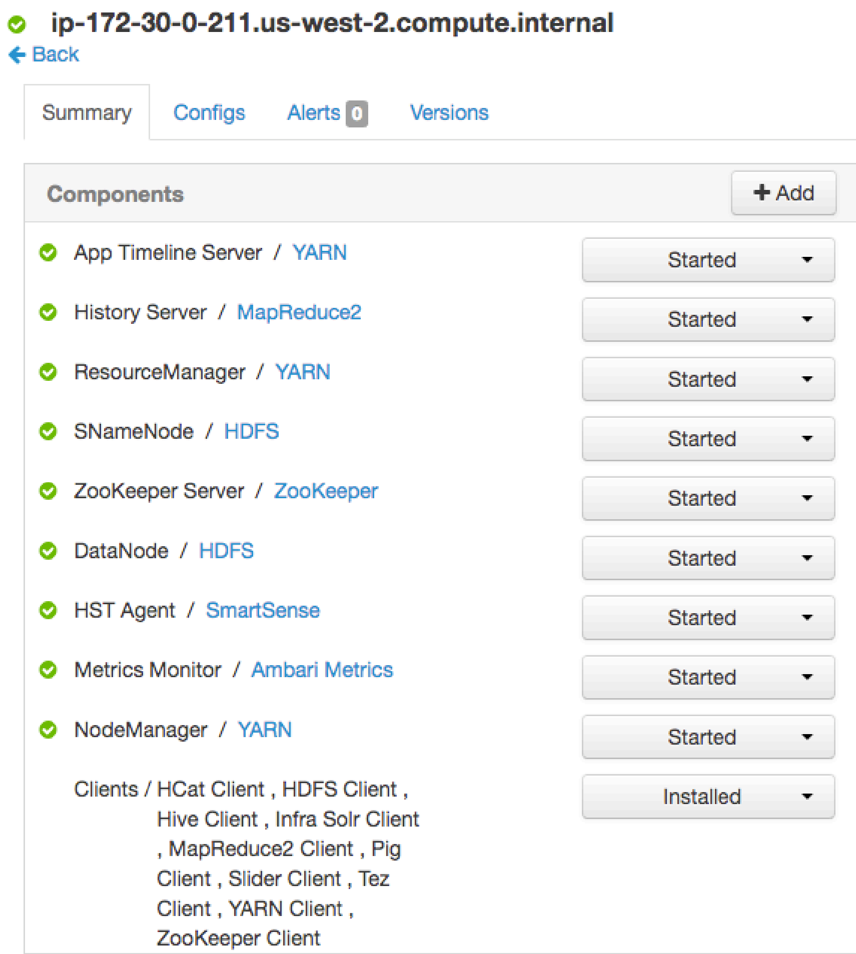
   b. Click Hosts in the Ambari Web UI.



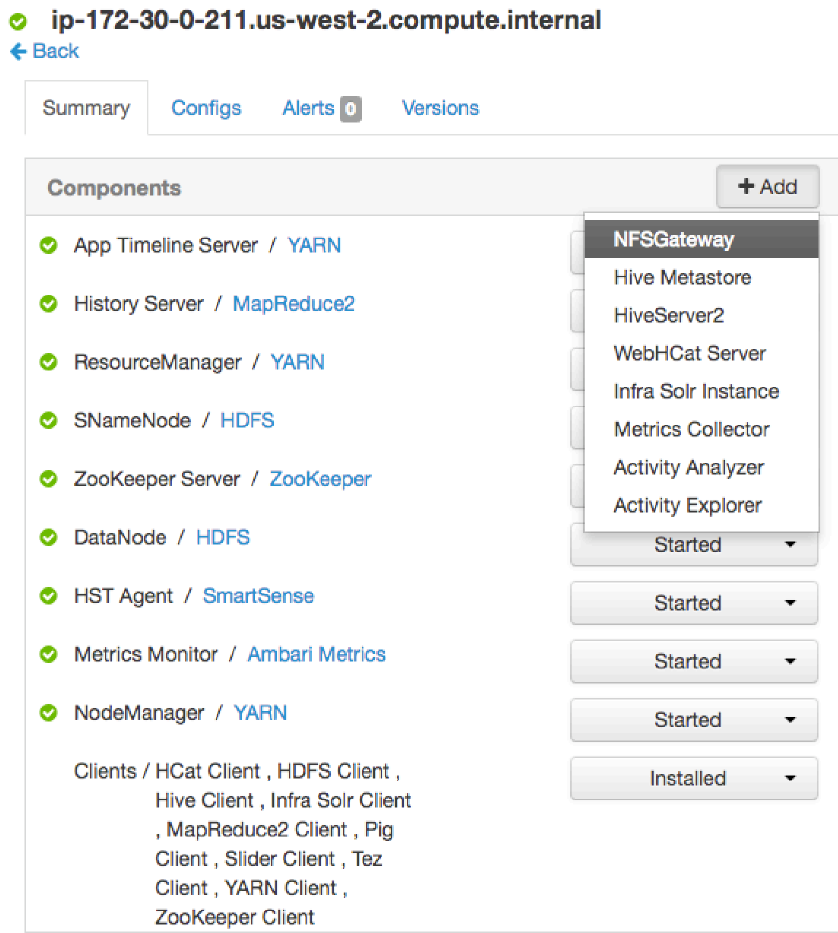
   c. Click on any host.



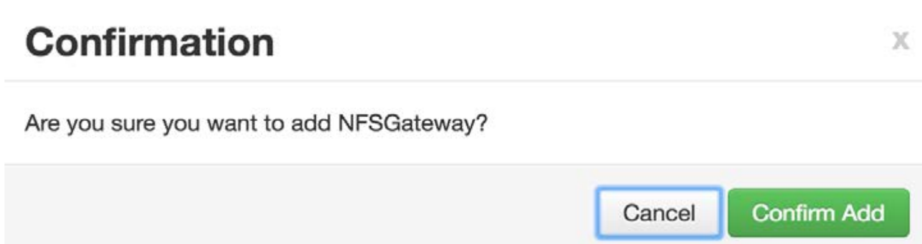
   d. Host Summary shows currently installed components.



e. Click on the "+ Add" button and Select NFSGateway. This will install NFSGateway component.



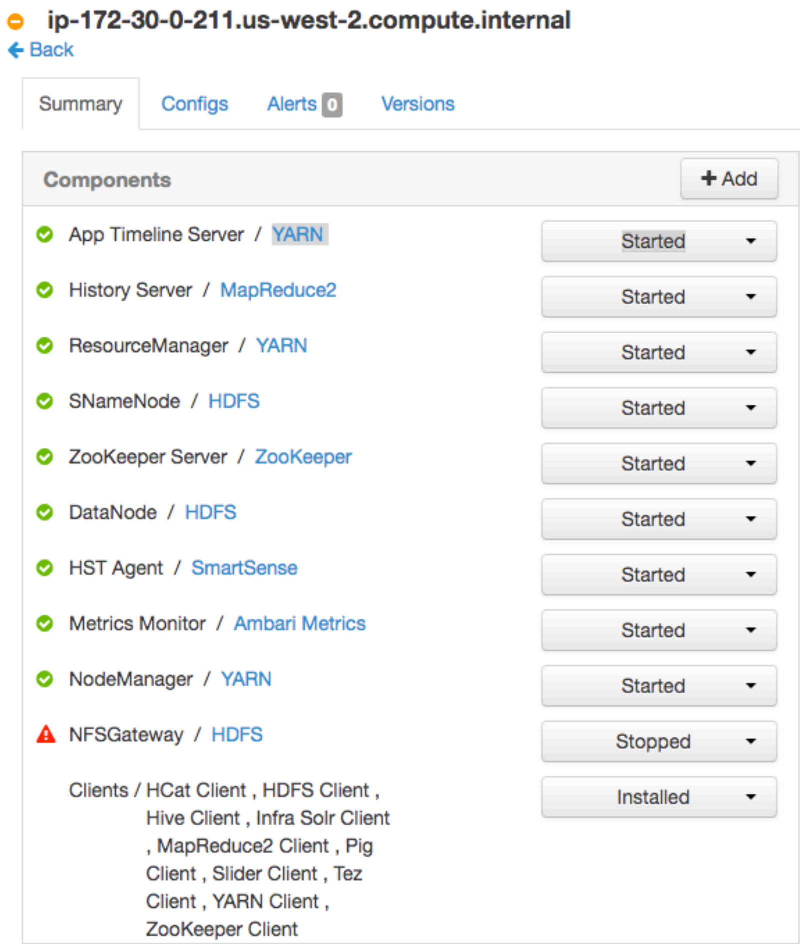
f. In the popup window, click **Confirm Add** to continue.



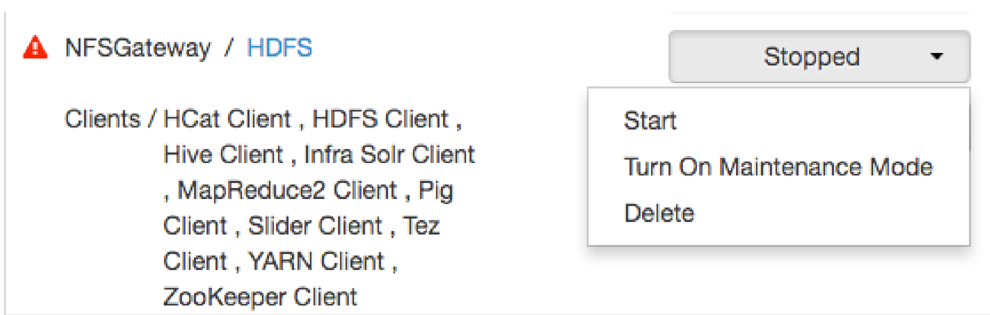
g. Ambari installation progress will display in the Background Operations window.

h. Click **OK** when the installation is complete.

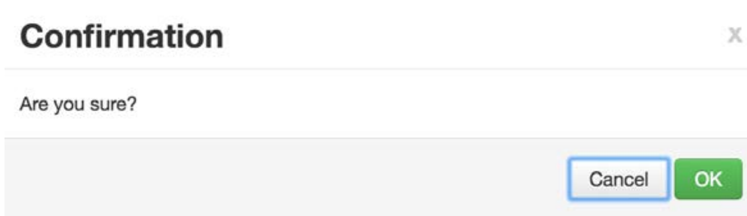
i. The host page will now show NFSGateway installed but not started.



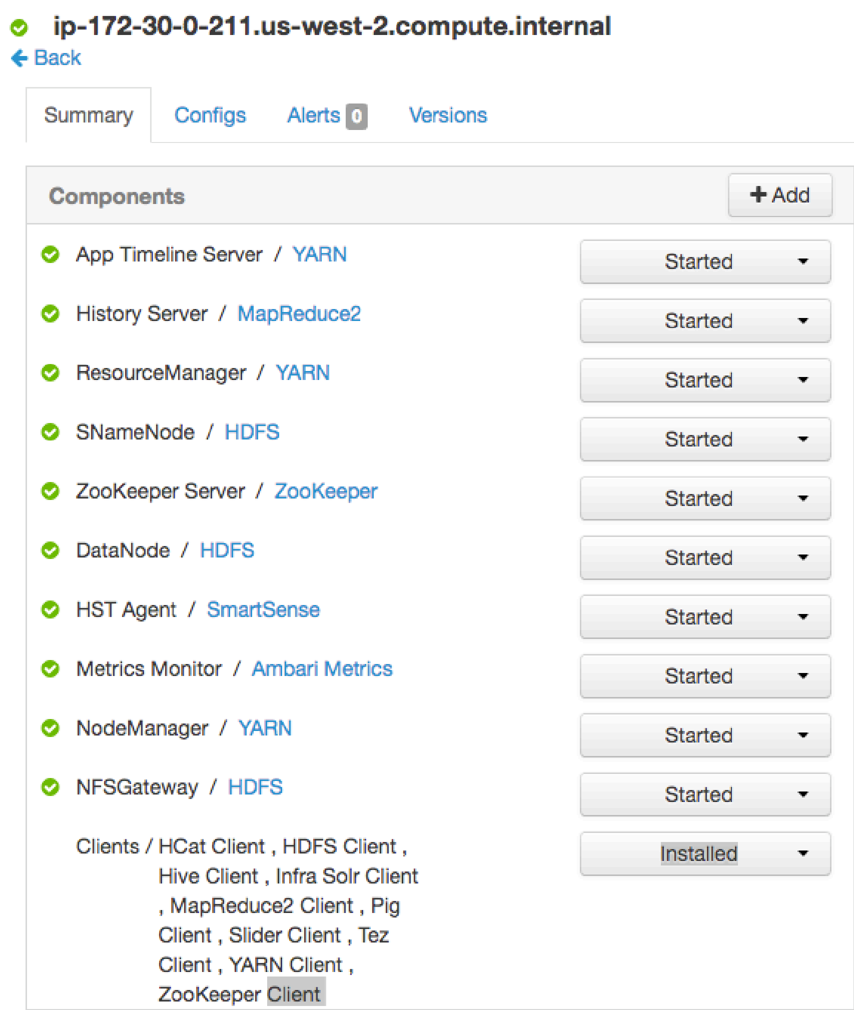
j. Start the NFSGateway



k. Confirm the Start click OK



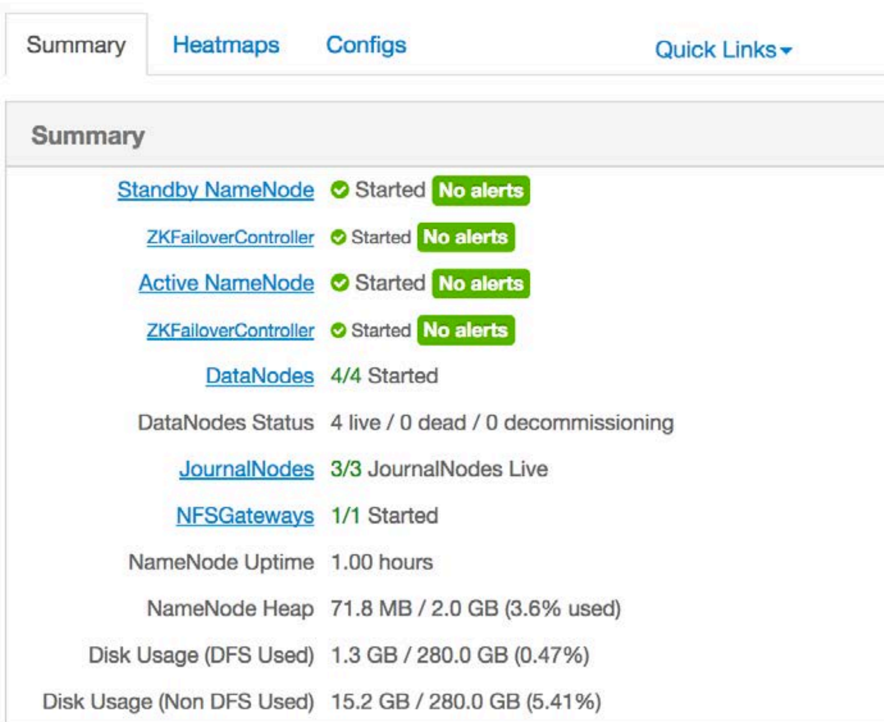
l. NFSGateway is now started



2. **Verify that the HDFS NFS Gateway is installed, started, and has exported the HDFS file system.**

   a. In the Ambari Web UI, click **Services** and select **HDFS**.

   You should see one NFS Gateway has started.



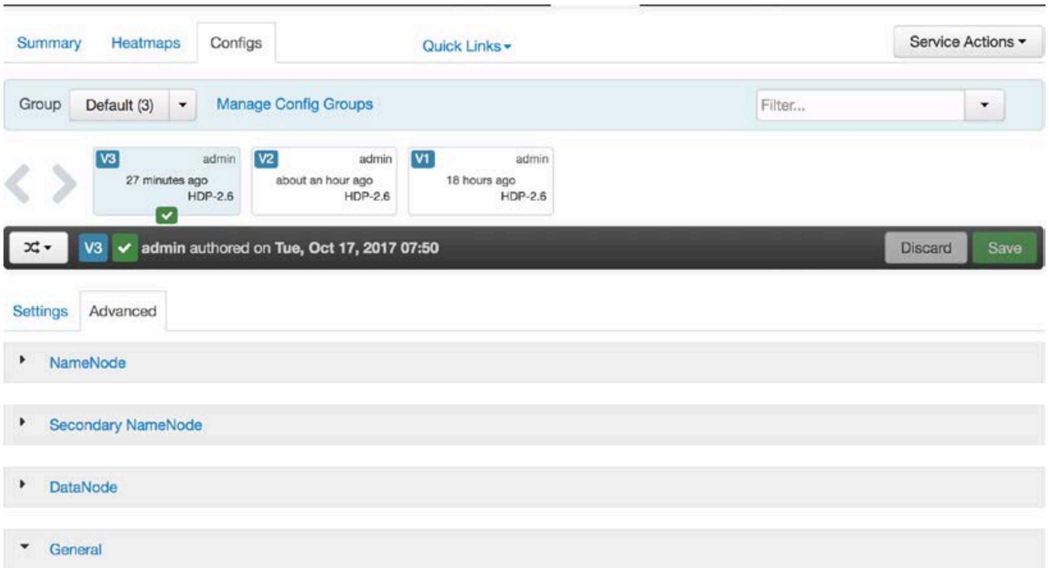
b. Remain logged in to the Ambari Web UI.

c. In the terminal window, use the `rpcinfo` and `showmount` commands to verify that NFS services are running and that the HDFS file system has been exported. Use the internal hostname that you installed the NFS Gateway component service on in the command below.

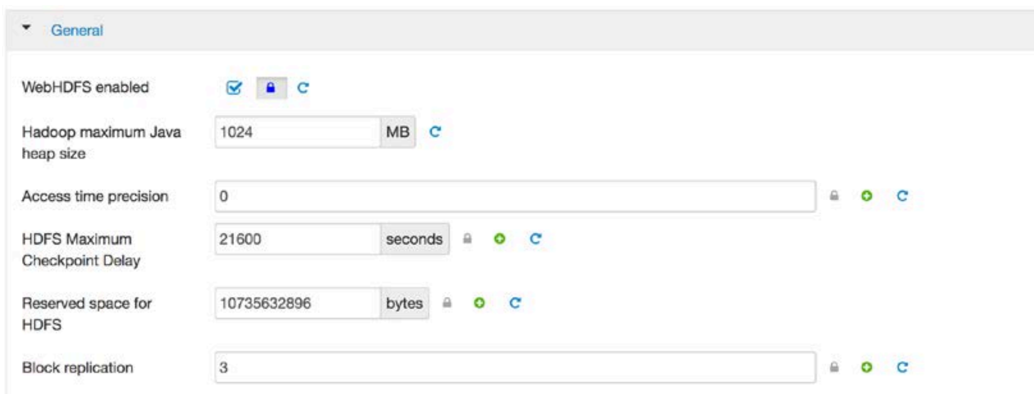
```
rpcinfo -p ip-172-30-0-211.us-west-2.compute.internal
showmount -e ip-172-30-0-211.us-west-2.compute.internal
```

3. **Update the HDFS NFS Gateway Access time precision property using the Ambari Web UI.**

   a. In the Ambari Web UI, browse to **Services** > **HDFS** > **Configs** > **Advanced** > **General**.



b. Change the **Access time precision** property from 0 to 3600000.



c. Click **Save** to save the change. Click **Save** again to confirm and then click **OK**.

d. Restart any services indicated in the Ambari Web UI.

e. Remain logged in to the Ambari Web UI.

4. **Use the command-line interface to NFS mount the HDFS file system.**

   a. In the open terminal window on another node and login.

   b. Create an empty directory to use as an NFS mount point and ensure that it is owned by the user hdfs.


```
mkdir /hdfs_nfs
chown hdfs /hdfs_nfs
```

c. Mount the exported HDFS file system to the mount point previously created. Then verify the mount.

```
mount -t nfs -o vers=3,proto=tcp,nolock,noacl,sync ip-172-30-0-211.us-west-2.compute.internal:/ /hdfs_nfs
mount
```

Do you see the mount? You should.

5. **Access the HDFS file system using the NFS mount.**

   a. List the contents of the `/hdfs_nfs` directory.

```
ls /hdfs_nfs/
```

Is there are `user` subdirectory? There should be.

   b. List the contents of the `/hdfs_nfs/user` directory.

```
ls /hdfs_nfs/user
```

Is there are `root` subdirectory? There should be.

   c. Copy the local `passwd` and `hosts` files to the root user's home directory in HDFS. Verify that the file was copied.

```
cp /etc/passwd /hdfs_nfs/user/root
cp /etc/hosts /hdfs_nfs/user/root
ls /hdfs_nfs/user/root
```

   d. View the contents of the HDFS `passwd` file.

```
cat /hdfs_nfs/user/root/passwd
```

   e. Remove the `hosts` file from HDFS.

```
rm /hdfs_nfs/user/root/hosts
```

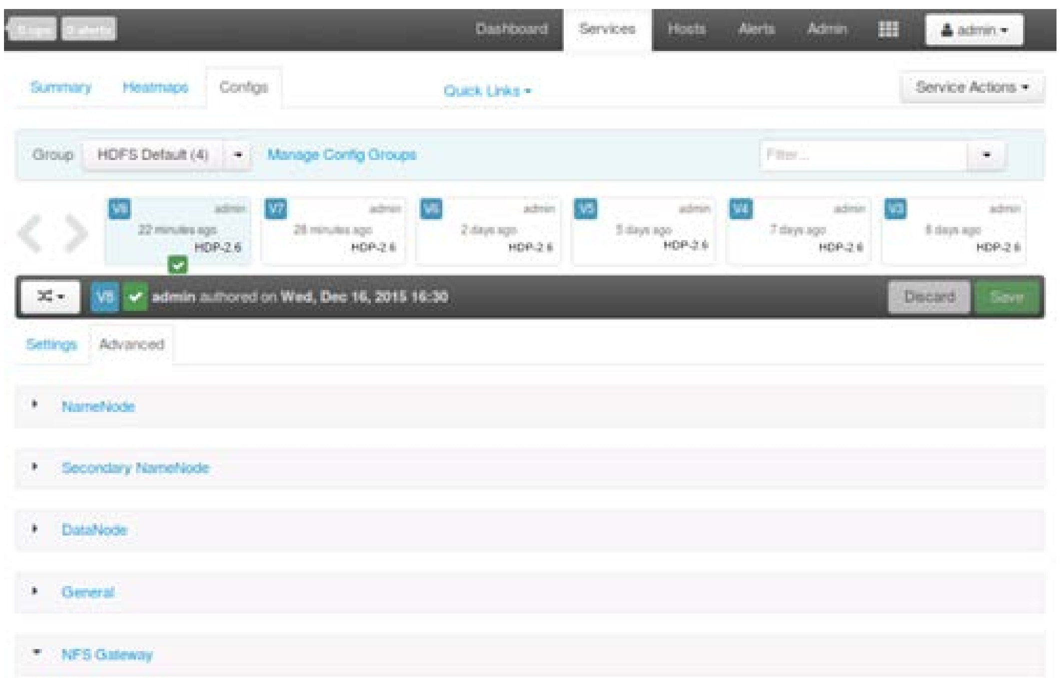
f. Copy the local group file to HDFS `/user` directory.

```
cp /etc/group /hdfs_nfs/user
```

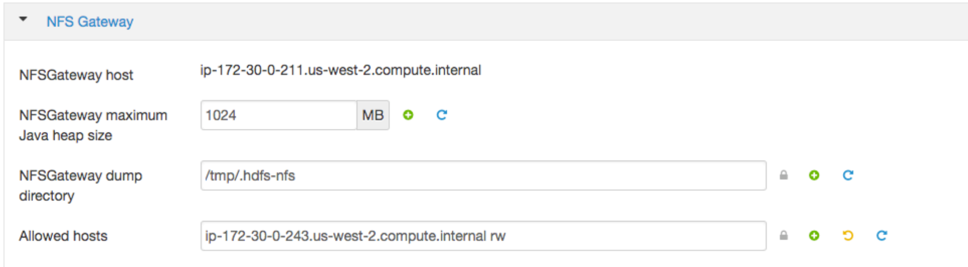
Did it fail?
It should have because the root user does not have write permissions in the HDFS `user` directory.

6. **Restrict NFS client mount permissions using the Ambari Web UI.**

   a. In the Ambari Web UI, browse to **Services** > **HDFS** > **Configs** > **Advanced** > **NFS Gateway**.



   b. Change the **Allowed hosts** property value from `* rw` to restrict access to one of your other nodes example: `ip-172-30-0-243.us-west-2.compute.internal rw`.



c. Click **Save** to save the change. Click **Save** again to confirm and then click **OK**.

d. Restart any services indicated in the Ambari Web UI.

e. In the open terminal window on restricted node and login.

f. Create an empty directory to use as an NFS mount point and ensure that it is owned by the user hdfs.

```
mkdir /hdfs_nfs
chown hdfs /hdfs_nfs
```

g. Mount the exported HDFS file system to the restricted host's mount point.

```
mount -t nfs -o vers=3,proto=tcp,nolock,noacl,sync ip-172-30-0-243.us-west-2.compute.internal:/ /hdfs_nfs
```

Was the NFS mount allowed on this this host? Yes, it was.

h. Return to the previous host that you originally mounted the exported HDFS file system. See if access is still available?

```
ls /hdfs_nfs
```

Did the command return a listing from the NFS mount on this host? No it did not, you get permission denied.

Result

The HDFS NFS Gateway as installed and running on your selected hosts and the HDFS Exported filesystem was mount on another selected host.

Lab 15: Managing the YARN Service Using Ambari Web UI

About This Lab

Objective:	To Use the Ambari Web UI and the ResourceManager UI to view and manage YARN status and configuration properties
File locations:	N/A
Successful outcome:	**You will:** Explore the Ambari Web UI service management and configuration features for YARN and change the YARN service configuration using the Ambari Web UI.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***YARN Resource Management***

Exploring the Ambari Web UI

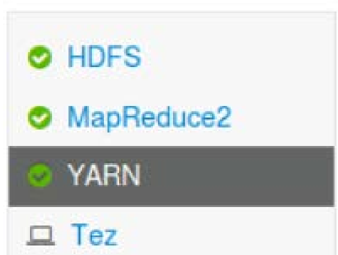
Explore the Ambari Web UI service management and configuration features for YARN.

1. Open the browser and connect to the Ambari Server at the URL

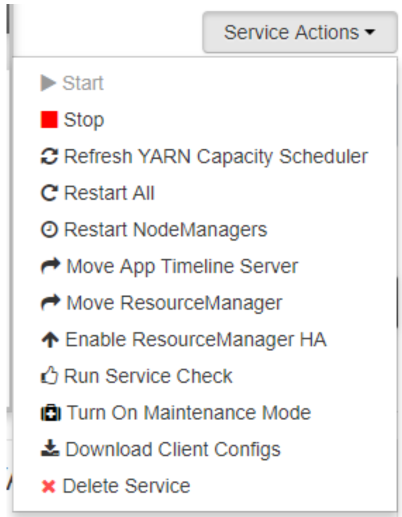
   http://<Ambari Server Hostname>:8080

2. Login to the Ambari Web UI using the user name `admin` and the password `BadPass#1`.
3. Click **Services** in the Ambari Web UI.

4. Select the **YARN** service on the **Services** page.

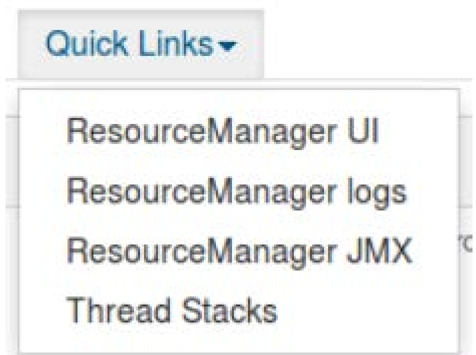


5. Click to open and view the **Service Actions** menu in the top right-side corner of the **Services** page.



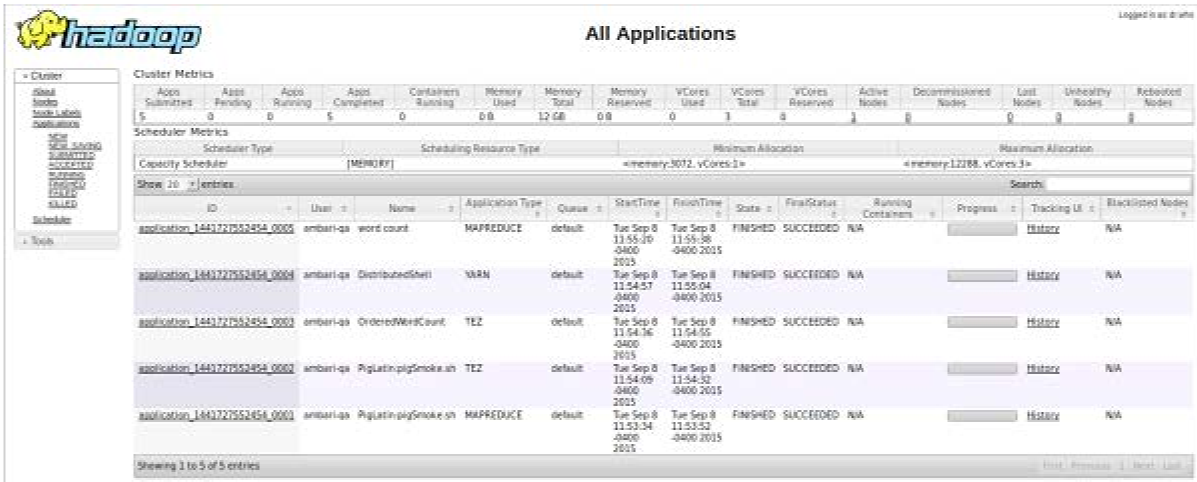
From here you can perform YARN management tasks such as restart YARN, restart NodeManagers, turn on maintenance mode, and download YARN client configuration files.

6. Click **Quick Links** and select **ResourceManager UI**.



The ResourceManager UI Web interface opens on another browser tab.

If simply clicking the quick link fails to open the ResourceManager UI, replace the default URL in the browser tab with `<Your Resource Manager Node IP Address>:8088`.



7. Take a few moments to click on the links in the menu on the left and familiarize yourself with the various pages and the information they provide. Answer the following questions:

   Where can you find information regarding the current version, last restart, and current status of the ResourceManager, as well as the current version of Hadoop and the Cluster ID?

   Where can you find information for a node, such as its rack location, current state, address, HTTP address, Hadoop software version, and available resources?

   Where can you find a list of all jobs that have been submitted but have not yet been accepted/approved to run?

   Where can you find a list of all jobs that have been approved but are not yet running?

   Where can you find information on a job that has failed?

   Where can you find information on a job that has successfully finished?

   Where can you find information on queues and the jobs that are using them?

8. Close the ResourceManager UI browser tab.

9 . With the **YARN** service still selected in the Ambari Web UI, view the types of information available on the **Summary** page. Use this page to answer the following questions:

What is the current status of the ResourceManager?

How many NodeManagers have been configured, and how many are currently started?

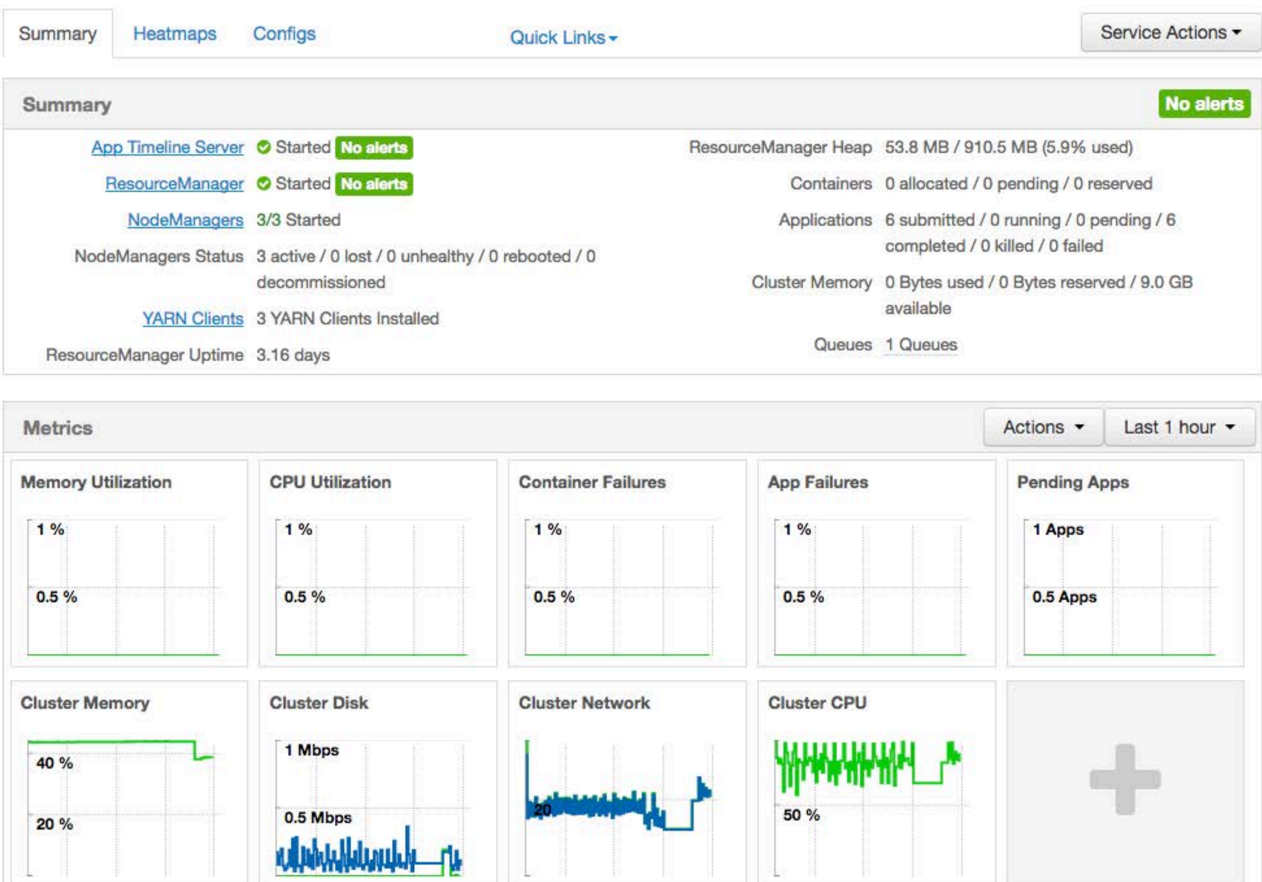
How long has it been since the last time the ResourceManager was restarted?

What percentage of ResourceManager Heap is being used?

How many applications have been submitted? How many are currently running? How many have completed? How many have failed?

How many queues have been configured?

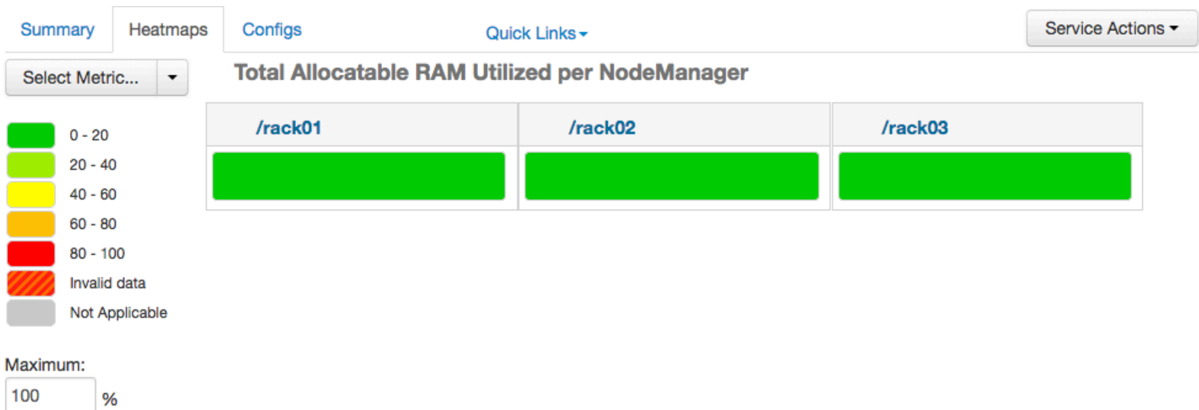
What percentage of cluster memory is currently being utilized? What is the average and max usage?



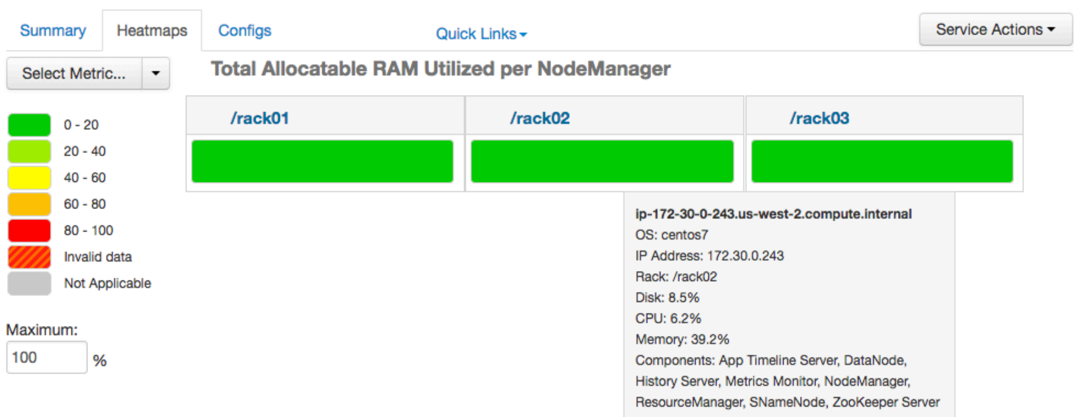
(Make sure you look at Cluster Memory and not just YARN application memory usage.)

10 . With the **YARN** service still selected, view the types of metric information available on the **Heatmaps** page.

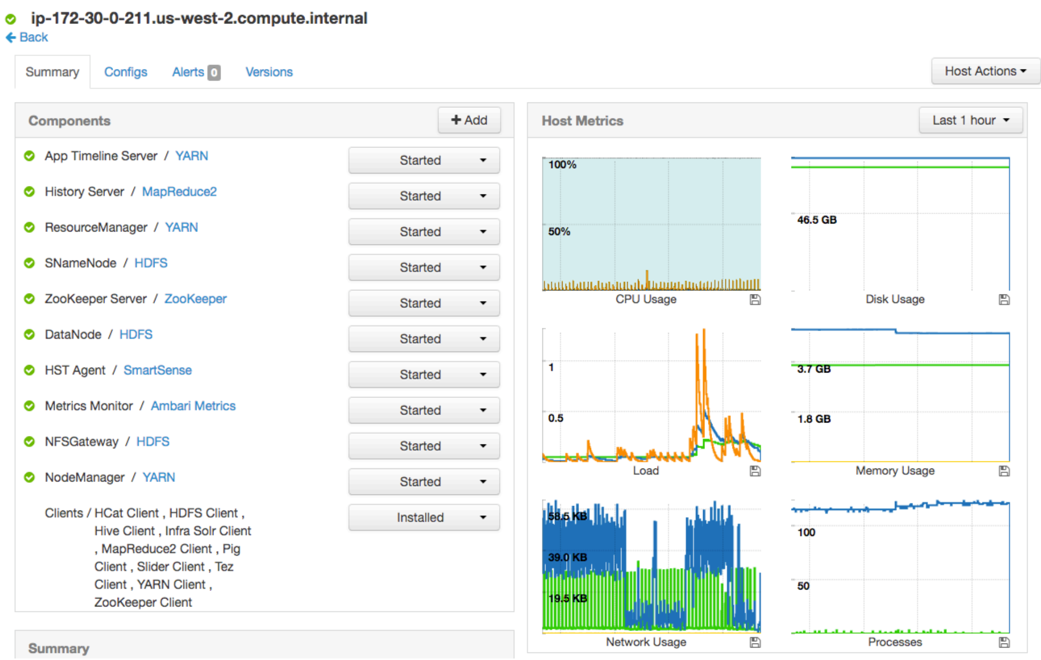
NOTE: That the number of nodes is visually reflected, so a single node is just a single bar, whereas three nodes would be represented with three separate bars.



11 . Hover over a node representation to see a quick view of node information such as the underlying operating system, IP address, resource utilization, rack location and installed components.

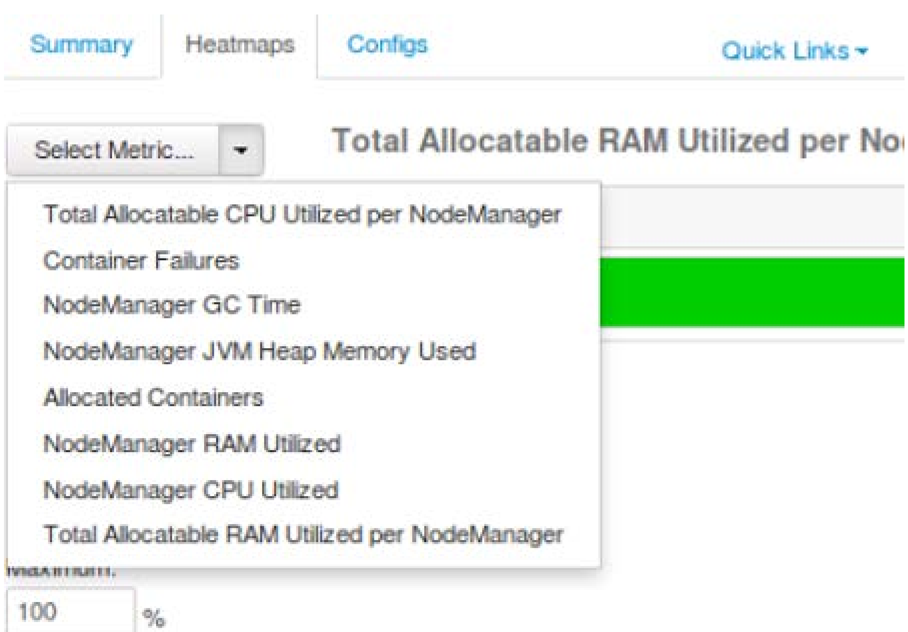


12 . Click the bar representing the node in **Heatmaps** to look at information for that node under the **Hosts** tab **i**n the Ambari Web UI.



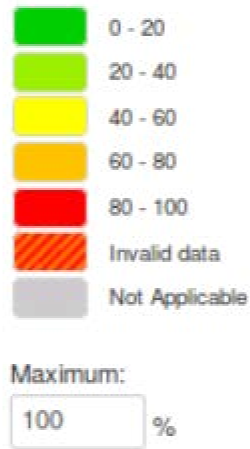
13 . Click the browser back button to go back to the **Heatmaps** tab under YARN **Services**.

Then click the **Select Metric** button to view the available metrics that can be viewed.



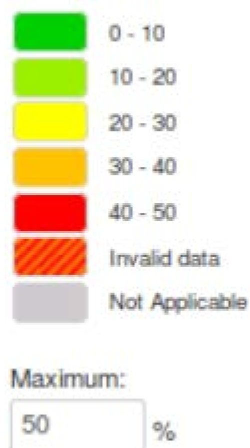
14 . Finally, at the bottom of the **Heatmaps** key, find the **Maximum** value in the text box.

Change this from `100%` to `50%` and note what this does to the key values.



15. The values for what constitutes the top green range change from 0-20 to 0-10.

This feature allows an administrator to determine what constitutes green, yellow, and red cluster nodes from a resource perspective.

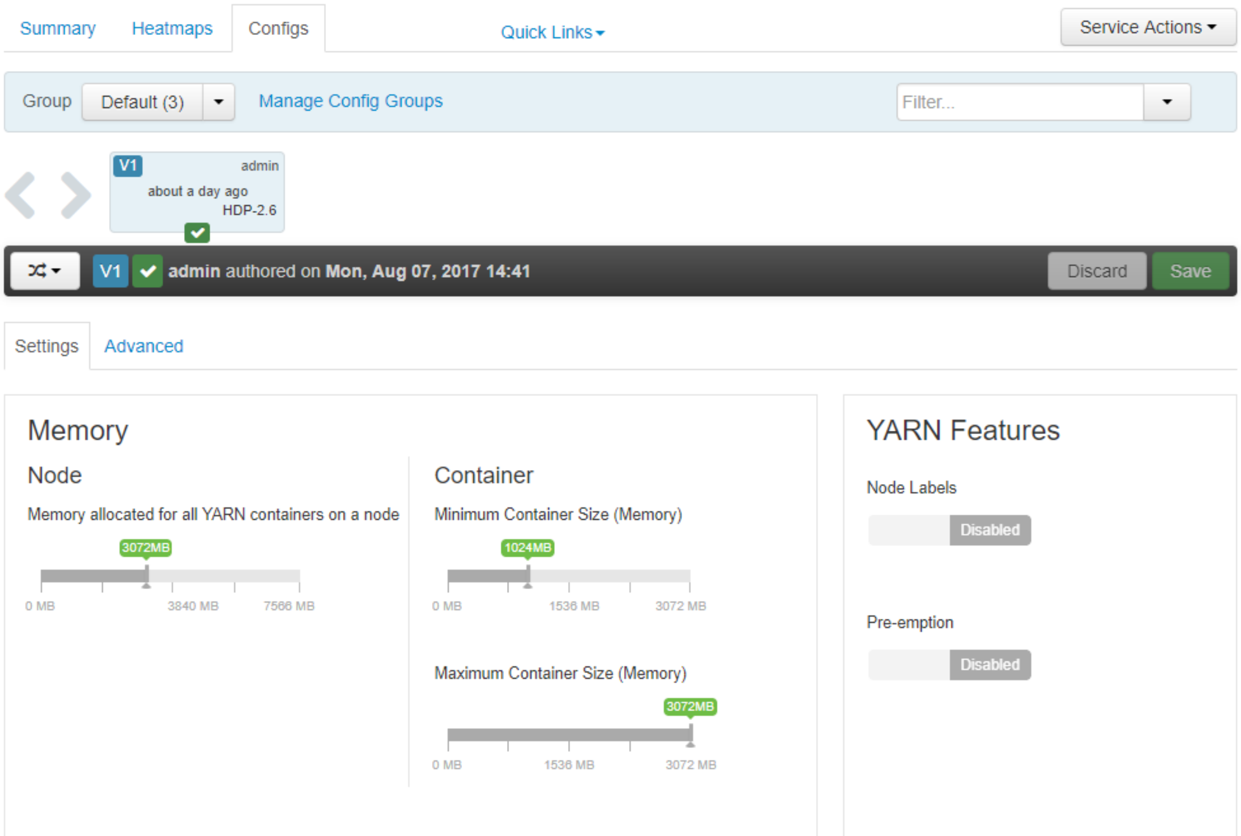


16. Change **Maximum** back to `100%`.

Changing YARN Service Configuration

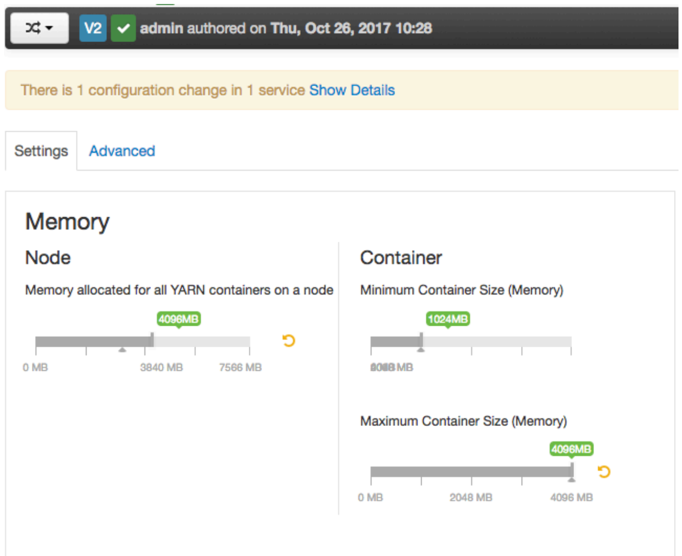
Change the YARN service configuration using the Ambari Web UI.

1. With the **YARN** service still selected, click the **Configs** tab.



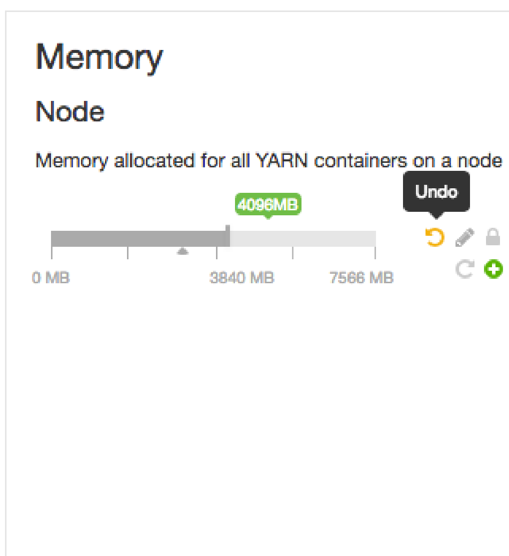
2 . On the **Settings** page, scroll down to find and change the **Memory allocated for all YARN containers on a node** setting. Change it by dragging the slider bar down slightly. Do not save the change.

NOTE: This is not the size recommended by Ambari Guided Configuration for the current cluster, so you will change this setting back in the next lab step.

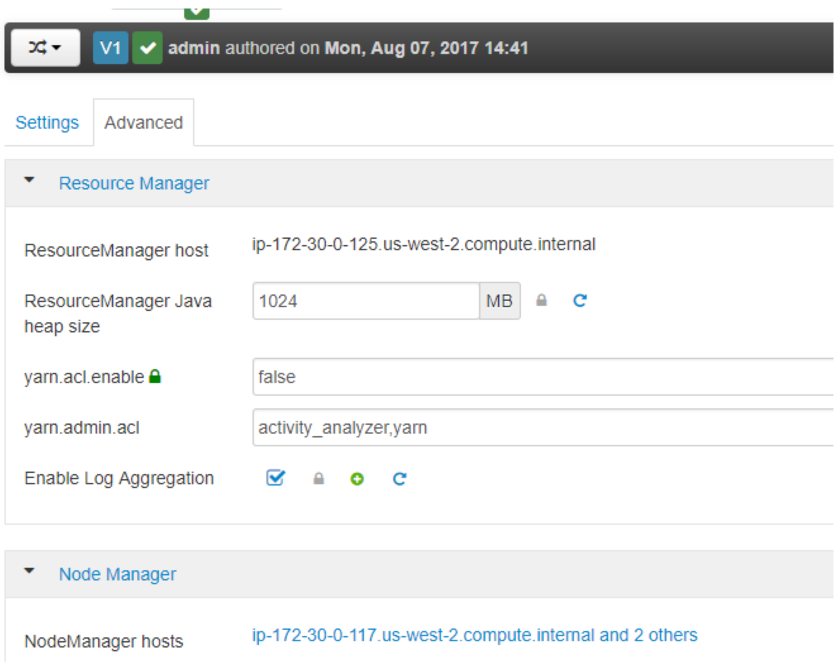


Also note that changing the maximum node memory available for YARN also changed the maximum container size memory (the setting to the right).

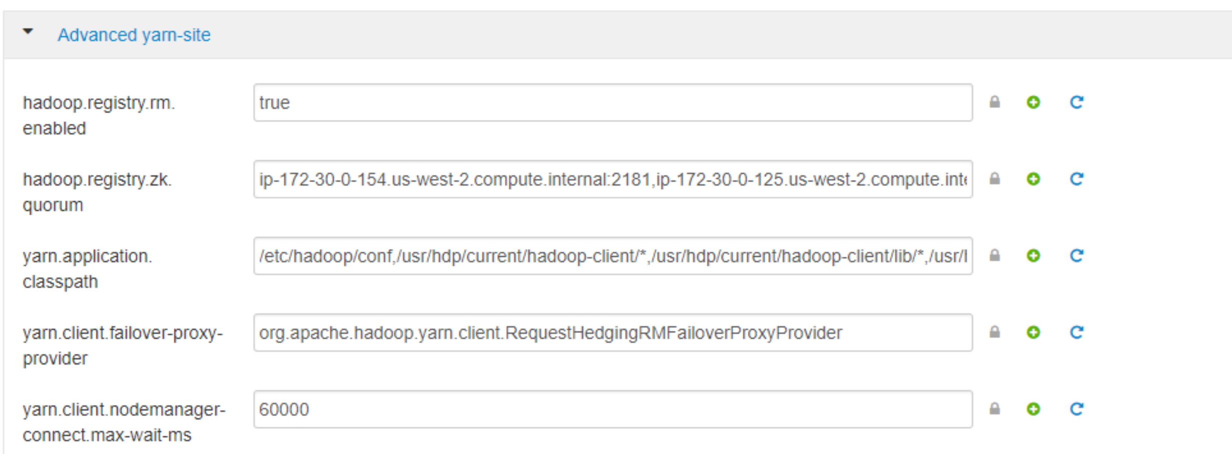
3. Click the yellow, counter-clockwise circular arrow icon to undo the change.



4. With the **YARN** service still selected, click the **Advanced** tab.



5 . Scroll down to find and expand the **Advanced yarn-site** section.



At this point, make no changes. Simply take a moment and familiarize yourself with the settings that are available here.

Result

You have now:
Explored Ambari Web UI service management and configuration features for YARN and changed the YARN service configuration using the Ambari Web UI.

Lab 16: Managing the YARN Service Using the CLI

About This Lab

Objective:	To use CLI commands and the YARN API to manage the YARN service
File locations:	N/A
Successful outcome:	**You will:** Use CLI user and administrator commands and run several simple YARN API calls.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***YARN Resource Management***

YARN CLI User Commands

Use YARN CLI commands useful to users of a Hadoop cluster.

1. Use Ambari UI to determine which host Resource Manager is running on. You will need to use your External AWS Hostname to login.
2. Use *SSH* to login to your Resource Manager node.

3. Find out what version of Hadoop is running on the system.

```
yarn version
```

4. Find out what happens when the *yarn application -list* command is executed without supplying any application types or states.

```
yarn application -list
```

NOTE: That by default, only applications that are in the **SUBMITTED**, **ACCEPTED**, or **RUNNING** state are listed.

```
[centos@ip-172-30-0-211 ~]$ yarn version
Hadoop 2.7.3.2.6.1.0-129
Subversion git@github.com:hortonworks/hadoop.git -r 45e64533cdee3edf67c7b88a0267c64c194f93e5
Compiled by jenkins on 2017-05-31T03:06Z
Compiled with protoc 2.5.0
From source with checksum deba7ab784606611731cd7c37443e1c
This command was run using /usr/hdp/2.6.1.0-129/hadoop/hadoop-common-2.7.3.2.6.1.0-129.jar
[centos@ip-172-30-0-211 ~]$ yarn application -list
17/08/14 16:29:13 INFO client.RMProxy: Connecting to ResourceManager at ip-172-30-0-211.us-west
-2.compute.internal/172.30.0.211:8050
17/08/14 16:29:13 INFO client.AHSProxy: Connecting to Application History server at ip-172-30-0
-211.us-west-2.compute.internal/172.30.0.211:10200
Total number of applications (application-types: [] and states: [SUBMITTED, ACCEPTED, RUNNING])
:0
                Application-Id      Application-Name        Application-Type          User
    Queue                   State             Final-State             Progress
        Tracking-URL
[centos@ip-172-30-0-211 ~]$
```

5 . Generate a list of all applications that are in the **FINISHED** state.

```
yarn application -list -appStates FINISHED
```

```
[centos@ip-172-30-0-211 ~]$ yarn application -list -appStates FINISHED
17/08/14 16:30:25 INFO client.RMProxy: Connecting to ResourceManager at ip-172-30-0-211.us-west
-2.compute.internal/172.30.0.211:8050
17/08/14 16:30:25 INFO client.AHSProxy: Connecting to Application History server at ip-172-30-0
-211.us-west-2.compute.internal/172.30.0.211:10200
Total number of applications (application-types: [] and states: [FINISHED]):6
                Application-Id      Application-Name        Application-Type          User
    Queue                   State             Final-State             Progress
        Tracking-URL
application_1502501149142_0006  TempletonControllerJob            MAPREDUCE     ambari-qa
  default                FINISHED             SUCCEEDED                100% http://ip-172-3
0-0-211.us-west-2.compute.internal:19888/jobhistory/job/job_1502501149142_0006
application_1502501149142_0005            word count              MAPREDUCE     ambari-qa
  default                FINISHED             SUCCEEDED                100% http://ip-172-3
0-0-211.us-west-2.compute.internal:19888/jobhistory/job/job_1502501149142_0005
application_1502501149142_0004      OrderedWordCount                    TEZ     ambari-qa
  default                FINISHED             SUCCEEDED                100% http://ec2-34-2
10-232-106.us-west-2.compute.amazonaws.com:8080/#/main/view/TEZ/tez_cluster_instance?viewPath=%
2F%23%2Ftez-app%2Fapplication_1502501149142_0004
application_1502501149142_0003  PigLatin:pigSmoke.sh                    TEZ     ambari-qa
  default                FINISHED             SUCCEEDED                100% http://ec2-34-2
10-232-106.us-west-2.compute.amazonaws.com:8080/#/main/view/TEZ/tez_cluster_instance?viewPath=%
2F%23%2Ftez-app%2Fapplication_1502501149142_0003
application_1502501149142_0002      DistributedShell                   YARN     ambari-qa
  default                FINISHED             SUCCEEDED                100%
                 N/A
application_1502501149142_0001  PigLatin:pigSmoke.sh                    TEZ     ambari-qa
  default                FINISHED             SUCCEEDED                100% http://ec2-34-2
10-232-106.us-west-2.compute.amazonaws.com:8080/#/main/view/TEZ/tez_cluster_instance?viewPath=%
2F%23%2Ftez-app%2Fapplication_1502501149142_0001
[centos@ip-172-30-0-211 ~]$
```

6 . Generate a list of nodes running the NodeManager daemon.

```
yarn node -list
```

What command would filter the list of nodes by those in the **DECOMMISSIONED** state?

Hint: Use yarn node -help.

7. Run the *yarn logs* command without an application ID (which will cause it to fail), and pipe the resulting help information to the *more* application to read.

```
yarn logs | more
```

Scroll through the information, noting the options that can be applied with this command once an application ID is provided, or simply press q to quit the *more* application.

YARN CLI Administrator Commands

Use YARN CLI commands useful to administrators of a Hadoop cluster.

1. Find out what log level the ResourceManager daemon is running. Use the Internal AWS Hostname. Example ip-172-30-0-211.us-west-2.compute.internal

```
yarn daemonlog -getlevel ip-172-30-0-211.us-west-2.compute.internal:8088
resourcemanager
```

2. The following commands need to be run as the yarn superuser, so switch to that user before moving on

```
su - yarn
```

3. Find out what groups the hdfs user belongs to..

```
yarn rmadmin -getGroups hdfs
```

4. Run the command that will refresh node information for the ResourceManager.

```
yarn rmadmin -refreshNodes
```

5. Return to acting as the root user.

```
exit
```

YARN API Calls

Run some simple YARN API calls using cURL.

1. Generate general cluster information such as the cluster ID and software version.

```
curl -X GET http://ip-172-30-0-211.us-west-2.compute.internal:8088/ws/v1/cluster/info
```

2. Generate cluster metrics information such as number of completed applications, resources available, and decommissioned nodes.

```
curl -X GET http://ip-172-30-0-211.us-west-2.compute.internal:8088/ws/v1/cluster/metrics
```

3. Generate cluster scheduler information such as queues, available capacity, and maximum number of applications allowed.

```
curl -X GET http://ip-172-30-0-211.us-west-2.compute.internal:8088/ws/v1/cluster/scheduler
```

4. Generate information on cluster applications such as application ID and status.

```
curl -X GET http://ip-172-30-0-211.us-west-2.compute.internal:8088/ws/v1/cluster/apps
```

5. Generate cluster information on cluster nodes such as the host names, status, software version, and available resources.

```
curl -X GET http://ip-172-30-0-211.us-west-2.compute.internal:8088/ws/v1/cluster/nodes
```

6. Generate general information about a node running NodeManager.

```
curl -X GET http://ip-172-30-0-211.us-west-2.compute.internal:8042/ws/v1/node/info
```

7. Generate a list of all applications currently running.

```
curl -X GET http://ip-172-30-0-211.us-west-2.compute.internal:8042/ws/v1/node/apps
```

Result

You have now:
Used CLI user and administrator commands and run several simple YARN API calls.

Lab 17: Running Sample YARN Applications

About This Lab

Objective:	To run YARN applications from the CLI and use the Hive Views
File locations:	N/A
Successful outcome:	**You will:** Use the command line to run a sample MapReduce job and interactive Pig and Hive commands; install, configure, and use the Ambari Hive View.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***YARN Applications***

Running YARN Applications from the CLI

Run YARN applications from the command line.

1. Use SSH to login to any cluster node.

2. Change to the `/usr/hdp/<VERSION NUMBER>/hadoop-mapreduce` directory.

```
cd /usr/hdp/<VERSION NUMBER>/hadoop-mapreduce
```

3. Run a sample program that uses MapReduce2 to calculate the number of words in the US Constitution.

```
yarn jar hadoop-mapreduce-examples.jar wordcount /user/root/constitution.txt
/user/root/results1

[root@ip-172-30-0-184 hadoop-mapreduce]# yarn jar hadoop-mapreduce-examples.jar
wordcount /user/root/constitution.txt /user/root/results1
16/11/29 18:27:39 INFO impl.TimelineClientImpl: Timeline service address:
http://ip-172-30-0-184.us-west-2.compute.internal:8188/ws/v1/timeline/
16/11/29 18:27:39 INFO client.RMProxy: Connecting to ResourceManager at ip-172-
30-0-184.us-west-2.compute.internal/172.30.0.250:8050
16/11/29 18:27:39 INFO client.AHSProxy: Connecting to Application History
server at ip-172-30-0-184.us-west-2.compute.internal/172.30.0.250:10200
16/11/29 18:27:40 INFO input.FileInputFormat: Total input paths to process : 1
16/11/29 18:27:40 INFO mapreduce.JobSubmitter: number of splits:1
16/11/29 18:27:40 INFO mapreduce.JobSubmitter: Submitting tokens for job:
job_1480440980637_0002
16/11/29 18:27:41 INFO impl.YarnClientImpl: Submitted application
application_1480440980637_0002
```

```
16/11/29 18:27:41 INFO mapreduce.Job: The url to track the job: http://ip-172-
30-0-184.us-west-2.compute.internal:8088/proxy/application_1480440980637_0002/
16/11/29 18:27:41 INFO mapreduce.Job: Running job: job_1480440980637_0002
16/11/29 18:27:48 INFO mapreduce.Job: Job job_1480440980637_0002 running in
uber mode : false
16/11/29 18:27:48 INFO mapreduce.Job:  map 0% reduce 0%
16/11/29 18:27:54 INFO mapreduce.Job:  map 100% reduce 0%
16/11/29 18:28:00 INFO mapreduce.Job:  map 100% reduce 100%
16/11/29 18:28:00 INFO mapreduce.Job: Job job_1480440980637_0002 completed
successfully
16/11/29 18:28:00 INFO mapreduce.Job: Counters: 49
	File System Counters
		FILE: Number of bytes read=23944
		FILE: Number of bytes written=330261
		FILE: Number of read operations=0
		FILE: Number of large read operations=0
		FILE: Number of write operations=0
		HDFS: Number of bytes read=45234
		HDFS: Number of bytes written=17261
		HDFS: Number of read operations=6
		HDFS: Number of large read operations=0
		HDFS: Number of write operations=2
	Job Counters
		Launched map tasks=1
		Launched reduce tasks=1
		Data-local map tasks=1
		Total time spent by all maps in occupied slots (ms)=7226
		Total time spent by all reduces in occupied slots (ms)=8002
		Total time spent by all map tasks (ms)=3613
		Total time spent by all reduce tasks (ms)=4001
		Total vcore-milliseconds taken by all map tasks=3613
		Total vcore-milliseconds taken by all reduce tasks=4001
		Total megabyte-milliseconds taken by all map tasks=5549568
		Total megabyte-milliseconds taken by all reduce tasks=8194048
	Map-Reduce Framework
		Map input records=872
		Map output records=7652
		Map output bytes=75556
		Map output materialized bytes=23944
		Input split bytes=115
		Combine input records=7652
		Combine output records=1697
		Reduce input groups=1697
		Reduce shuffle bytes=23944
		Reduce input records=1697
		Reduce output records=1697
		Spilled Records=3394
		Shuffled Maps =1
		Failed Shuffles=0
		Merged Map outputs=1
		GC time elapsed (ms)=131
		CPU time spent (ms)=2090
		Physical memory (bytes) snapshot=1316478976
		Virtual memory (bytes) snapshot=6939344896
		Total committed heap usage (bytes)=1188560896
	Shuffle Errors
		BAD_ID=0
```

```
            CONNECTION=0
            IO_ERROR=0
            WRONG_LENGTH=0
            WRONG_MAP=0
            WRONG_REDUCE=0
      File Input Format Counters
            Bytes Read=45119
      File Output Format Counters
            Bytes Written=17261
[root@ip-172-30-0-184 hadoop-mapreduce]#
```

4. The program counts the number of words in the US Constitution. The output is in a file in the /user/root/results1 directory named part-r-00000. You can cat this file to see the results

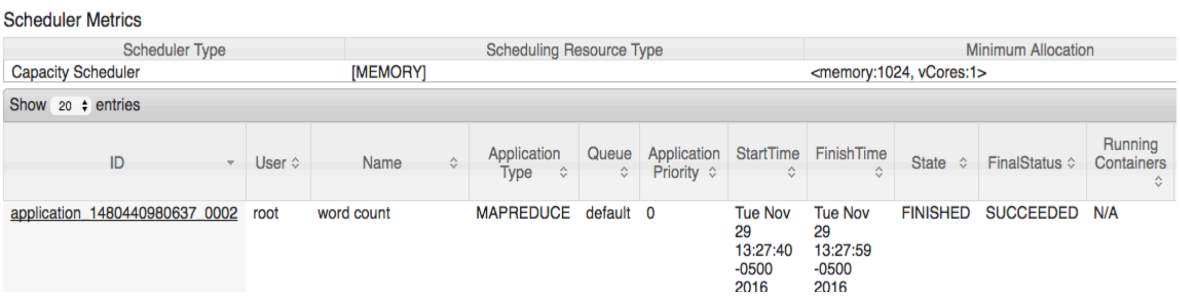
```
hdfs dfs -ls /user/root/results1
hdfs dfs -cat /user/root/results1/part-r-00000
[root@ ip-172-30-0-184 hadoop-mapreduce]# hdfs dfs -cat
/user/root/results1/part-r-00000
"I      1
(Sundays        1
(except 1
(not    1
(or     1
(when   1
-       13
------------------------------          1
1       6
1.      12
10      2
11      1
12      1
13      1
14      1
15      1
15th    1
16      1
17      1
18      1
19      1
2       6
2.      12
20      1
20th    1
21      1
22      1
23      1
24      1
25      1
26      1
27      1
3       5
3.      6
3d      2
4       4
4.      4
```

```
5     2
5.    3
6     2
6.    2
7     2
7.    1
8     2
9     2
A     3
#
# Truncated
#
vacancies   3
vacancies:  1
vacancy     1
vacated     1
valid 3
validity    1
value 1
varying     1
version     1
version.    2
vest  1
vested      4
violated,   1
violation   1
visit 1
void. 1
vote  10
vote. 1
vote; 1
voted 6
votes 5
voting      1
war,  1
was   3
way   1
well  2
were  1
what  2
whatever    1
whatever,   1
whatsoever, 1
when  11
whenever    4
where 2
wherein     3
whereof     3
which 42
which,      1
who   15
who,  1
whole 10
whom  4
whom, 1
whose 1
will  3
```



```
with  17
within      19
within,     1
without     13
witness     1
witnesses   2
work  1
would 2
writing,    1
writs 1
written     6
year  1
year, 1
years 9
years;      1
```

5. Open the browser and type the URL `http://<Resource Manager Hostname>:8088/cluster/apps/FINISHED` to view information about this application. You can also access it via Ambari QuickLinks under the YARN Service.



Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>

Show 20 entries

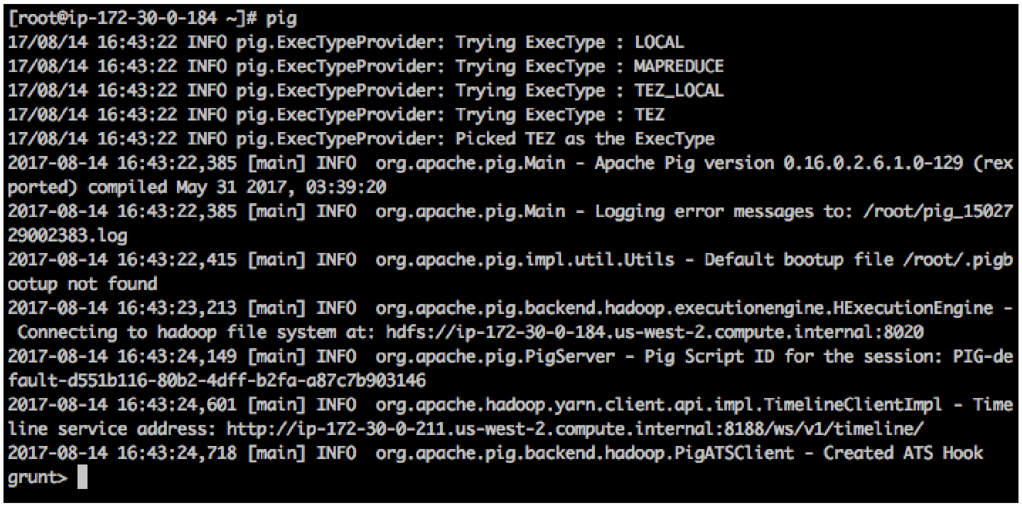
ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State	FinalStatus	Running Containers
application_1480440980637_0002	root	word count	MAPREDUCE	default	0	Tue Nov 29 13:27:40 -0500 2016	Tue Nov 29 13:27:59 -0500 2016	FINISHED	SUCCEEDED	N/A

6. Next download a salaries file that contain the data needed.
7. Put the salaires file in HDFS and now you are ready to execute a Pig script.

```
wget
https://raw.githubusercontent.com/HortonworksUniversity/Ops_Labs/master/salaries.txt

hdfs dfs -put salaries.txt /user/root/salaries.txt
```

8. Launch Pig to get the Grunt shell prompt. This allows you to run interactive Pig commands.

```
pig
```



```
[root@ip-172-30-0-184 ~]# pig
17/08/14 16:43:22 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
17/08/14 16:43:22 INFO pig.ExecTypeProvider: Trying ExecType : MAPREDUCE
17/08/14 16:43:22 INFO pig.ExecTypeProvider: Trying ExecType : TEZ_LOCAL
17/08/14 16:43:22 INFO pig.ExecTypeProvider: Trying ExecType : TEZ
17/08/14 16:43:22 INFO pig.ExecTypeProvider: Picked TEZ as the ExecType
2017-08-14 16:43:22,385 [main] INFO  org.apache.pig.Main - Apache Pig version 0.16.0.2.6.1.0-129 (rex
ported) compiled May 31 2017, 03:39:20
2017-08-14 16:43:22,385 [main] INFO  org.apache.pig.Main - Logging error messages to: /root/pig_15027
29002383.log
2017-08-14 16:43:22,415 [main] INFO  org.apache.pig.impl.util.Utils - Default bootup file /root/.pigb
ootup not found
2017-08-14 16:43:23,213 [main] INFO  org.apache.pig.backend.hadoop.executionengine.HExecutionEngine -
 Connecting to hadoop file system at: hdfs://ip-172-30-0-184.us-west-2.compute.internal:8020
2017-08-14 16:43:24,149 [main] INFO  org.apache.pig.PigServer - Pig Script ID for the session: PIG-de
fault-d551b116-80b2-4dff-b2fa-a87c7b903146
2017-08-14 16:43:24,601 [main] INFO  org.apache.hadoop.yarn.client.api.impl.TimelineClientImpl - Time
line service address: http://ip-172-30-0-211.us-west-2.compute.internal:8188/ws/v1/timeline/
2017-08-14 16:43:24,718 [main] INFO  org.apache.pig.backend.hadoop.PigATSClient - Created ATS Hook
grunt>
```

9. Run the following pig script, this will display Salaries that are greater than 10000.

```
grunt > salaries = LOAD '/user/root/salaries.txt' USING PigStorage(',') AS
(gender, age, income, zip);

grunt > highsalaries = FILTER salaries BY income > 10000;

grunt > Dump highsalaries;


... SNIPPET ...


(F,66,41000,95103)

(M,40,76000,95102)

(F,58,95000,95103)

(F,68,60000,95105)

(M,85,14000,95102)

(M,67,99000,94040)

(F,43,11000,94041)

(F,37,65000,94040)

(M,72,83000,94041)

(M,68,15000,95103)

(F,74,37000,95105)

(M,23,89000,95105)

(M,23,64000,94041)

(M,79,15000,94040)

(F,65,70000,95102)

(F,92,56000,94041)

(M,50,18000,95102)

(M,44,96000,94040)
```

```
(F,73,12000,95102)
(M,55,32000,94040)
(F,33,29000,95050)
(M,67,81000,95101)
(M,31,95000,94041)
(M,34,61000,94040)
(F,22,90000,95102)
(M,66,84000,95103)
(F,97,69000,95103)
(M,48,91000,95102)
(M,45,48000,94041)
(F,84,14000,95051)

grunt > quit;
```

10 . Hive queries can be executed from the CLI by using the Hive shell, which is accessed by issuing the *hive* command.

```
hive
```

11 . Use the Hive shell to create a new table called salaries. Load in the salaries.txt data. Run a query to display Salaries that are greater than 10000.

```
hive > drop table if exists salaries;
hive > create external table salaries (
       gender string,
       age int,
       salary double,
       zip int
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';
```

```
hive > load data inpath '/user/root/salaries.txt' overwrite into table
salaries;

hive > Select gender, count(*) as cnt from salaries where salary > 10000 group
by gender;

Query ID = root_20161205161912_3cbb7c24-9cdf-4501-b932-556375c49664
Total jobs = 1
Launching Job 1 out of 1

Status: Running (Executing on YARN cluster with App id
application_1480440980637_0007)

--------------------------------------------------------------------------------
-
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED
KILLED
--------------------------------------------------------------------------------
-
Map 1 ..........   SUCCEEDED      1          1        0        0       0
0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0
0
--------------------------------------------------------------------------------
-
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 6.49 s
--------------------------------------------------------------------------------
-
OK
F     13
M     17
Time taken: 8.478 seconds, Fetched: 2 row(s)
hive>
```

NOTE: If you forget to put the semi-colon (`;`) at the end of the command, a newline prompt (>) is displayed. If this occurs, type a semi-colon (`;`) at the newline prompt and press **Enter** to finish the command.

12 . Use the *quit* command to exit the Hive shell.

```
hive > quit;
```

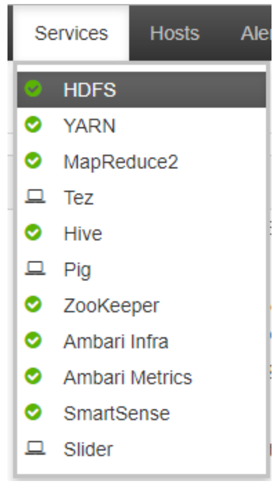
Using Ambari Hive View

Install, configure, and use the Ambari Hive view.

1 . Open the browser and connect to the Ambari Server at the URL

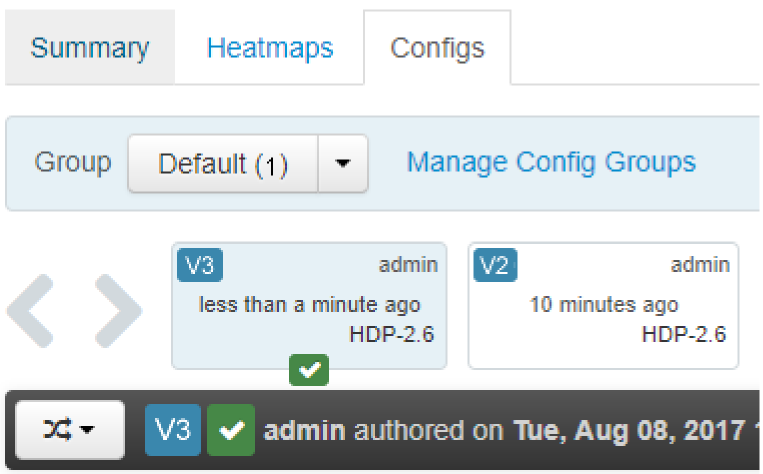
http://<Ambari Server Hostname>:8080

2. Login to the Ambari Web UI using the user name `admin` and the password BadPass#1.
3. In the Ambari Web UI, click **Services** and select **HDFS**.



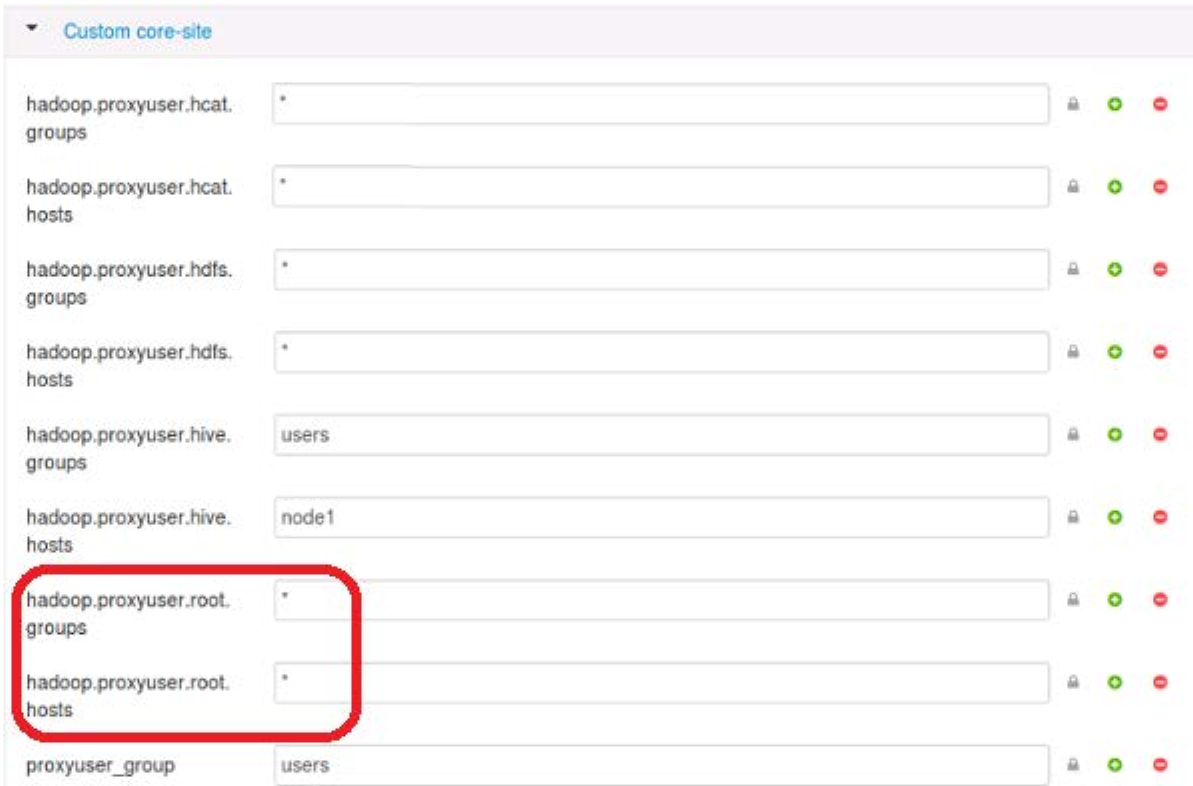
4. Click **Configs** and then click the **Advanced** tab.

   Ensure that the **HDFS Default** configuration group is selected.

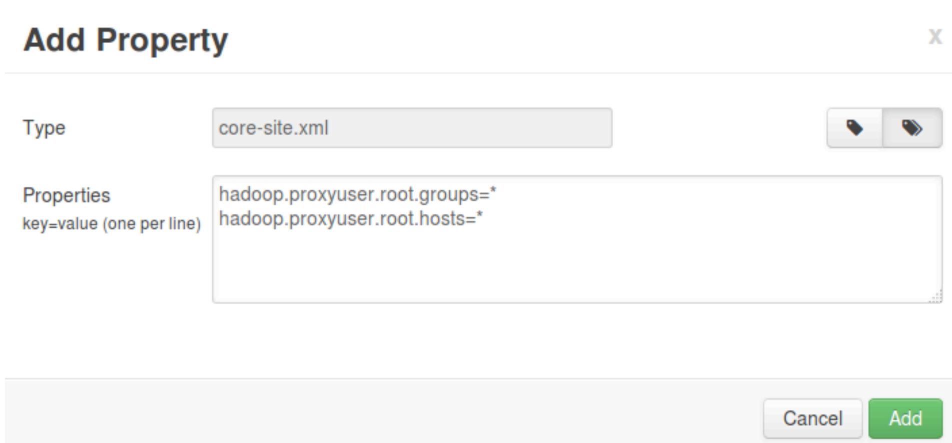


   Scroll down to find and expand the **Custom core-site** section and confirm that the values `hadoop.proxyuser.root.groups=*` and `hadoop.proxyuser.root.hosts=*` exist.

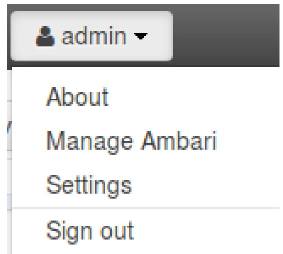
   These should have been added already when configuring the Ambari Files View in an earlier lab, and confirmed in the Ambari Pig View lab.



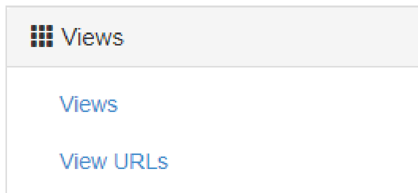
If those labs have not been completed yet, these values need to be added here as well at this point via the **Add Property** link.



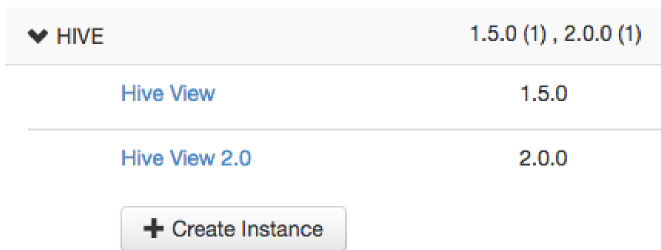
5 . Select **Manage Ambari** from the **admin** menu button in order to initiate the process of creating an instance of an Ambari View.



6 . Click **Views**.



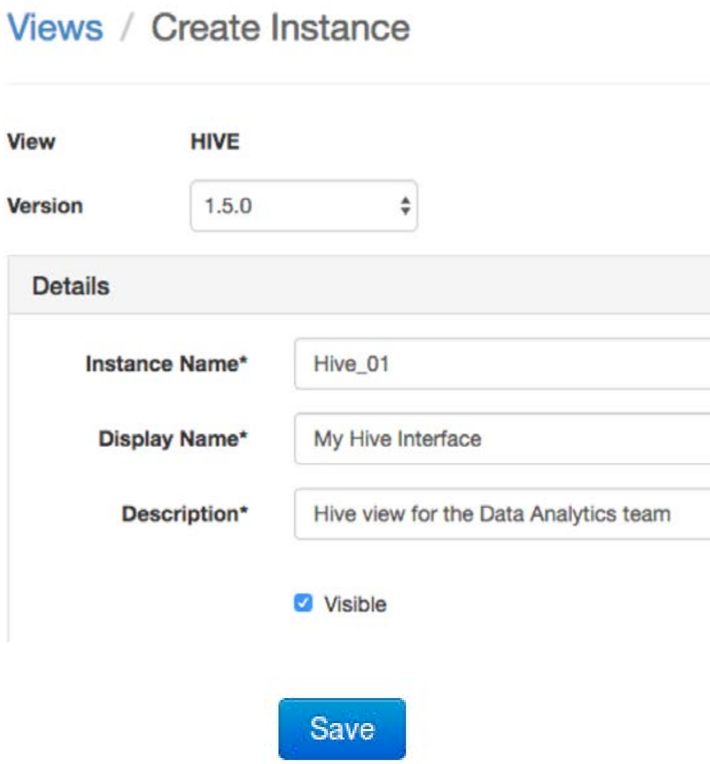
7 . Click to expand **Hive** and then click **Create Instance**.



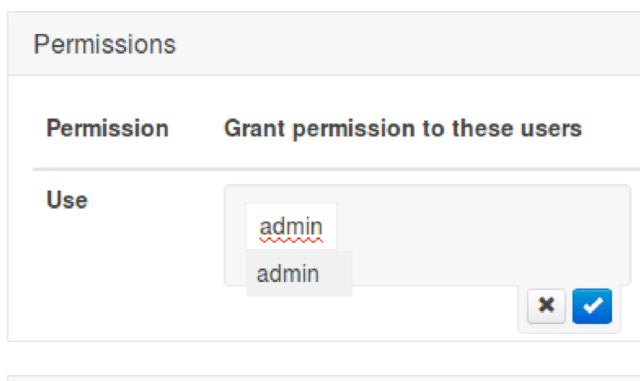
8 . In the **Details section**, enter:

Select `1.5.0` from the Version drop down
Instance Name is `Hive_01`
Display Name `My Hive Interface`
Description `Hive view for the Data Analytics team`

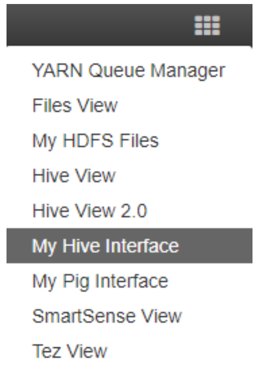
Then scroll down and click **Save**.



9 . Scroll down and locate the **Permissions** section.
Beneath **Grant permission to these users**, click **Add User**, then click **New** and type `admin`.
Click the blue check box to save your change.



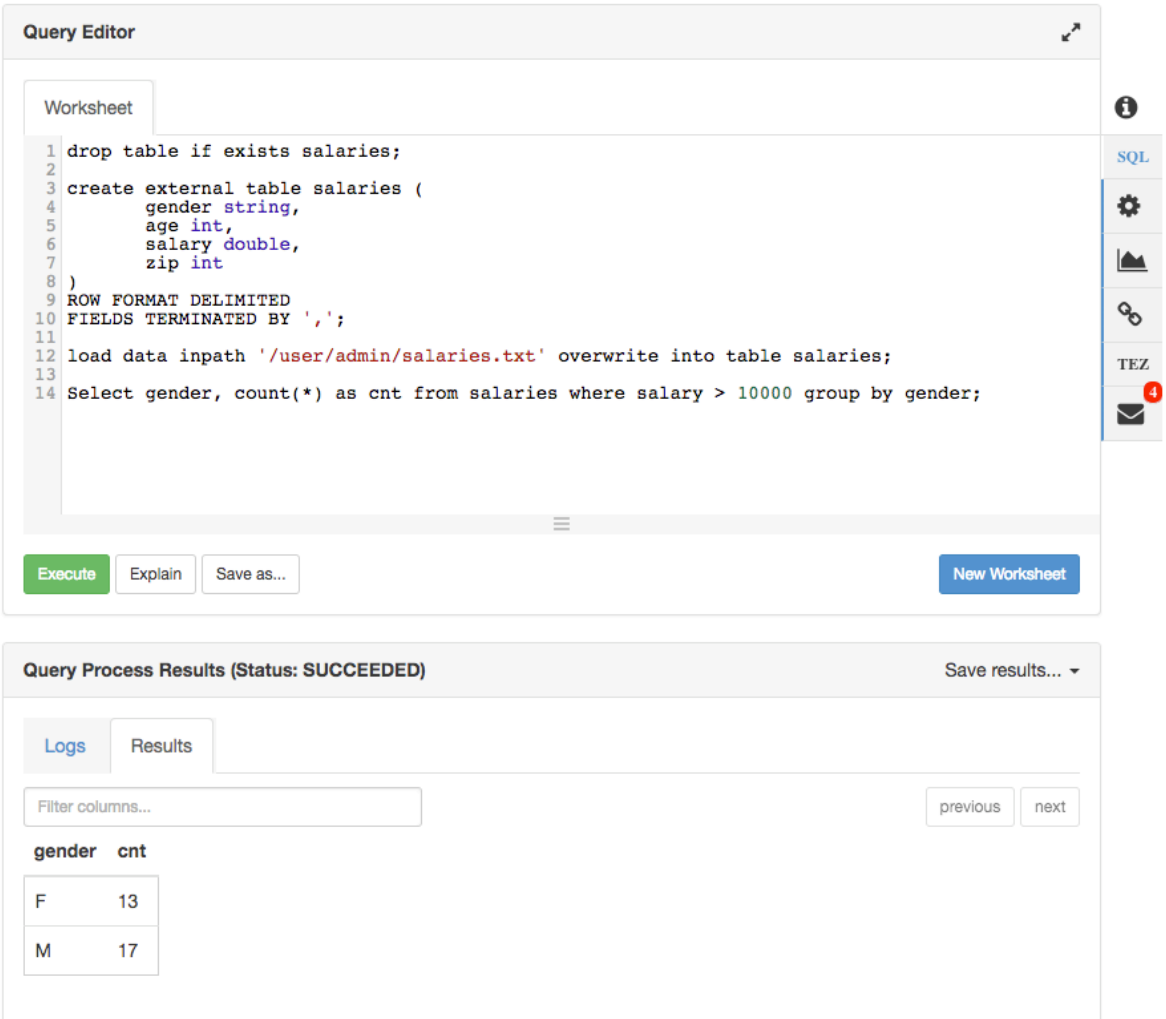
10 . Click the **Views** icon and select the View named **My Hive Interface**.



1. Copy the Hive query below into your Hive View then click on the green "Execute" button. This will show salaries > 10000 within the view. Output will be the same as running these Hive commands from the command line.

```
drop table if exists salaries;

create external table salaries (
        gender string,
        age int,
        salary double,
        zip int
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';

load data inpath '/user/admin/salaries.txt' overwrite into table salaries;

Select gender, count(*) as cnt from salaries where salary > 10000 group by
gender;
```

2 . Once the query completes the View will be updated with the Query Results.



Result

You have now:
Used the command line to run a sample MapReduce job and interactive Pig and Hive commands; installed, configured, and used the Ambari Pig View; and installed, configured, and used the Ambari Hive View.

Lab 18: Setting Up for Capacity Scheduler Labs

About This Lab

Objective:	To create user home directories and Linux users and groups in preparation for capacity scheduler labs
File locations:	N/A
Successful outcome:	**You will:** Create a Linux group matching each home directory; Create Linux accounts for each home directory with memberships in the appropriate groups. Create an HDFS home directory for each user identified in the classroom scenario;
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***The YARN Capacity Scheduler***

Creating Linux Groups

1 Create Linux groups named **promo**, **sales**, **dev**, and **qa** using the *groupadd* command on **ALL NODES in your cluster**. Login to each of your cluster nodes and switch to the root user.

```
sudo su -

groupadd promo
groupadd sales
groupadd dev
groupadd qa
```

Creating Linux User Accounts

1 Use the *useradd* command to create Linux accounts on **ALL NODES** with membership in the appropriate groups.

dev - `dev01, dev02, dev13`
promo - `promo01, promo02`
qa - `qa01, qa02, qa05`
sales - `sales01, sales02, sales04`

It is not necessary to set up a password for the users for lab purposes.

```
useradd <user name> -G <group name>
```

The example below demonstrates the command for user `dev01`:

```
useradd dev01 -G dev
```

Repeat until all users have been created and added to the groups as follows:
Create the three *supportXX* users

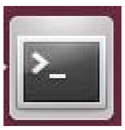
NOTE: When you create the three *supportXX* users, you will not use *-G* or provide a group name because they are not members of a group.

```
useradd support01
useradd support02
useradd support09
```

Creating Home Directories

Create an HDFS home directory for each user identified in the classroom scenario and assign that user ownership of the directory.

1. Use SSH to login to **each of your cluster nodes**.

   On one of your Nodes Create HDFS Home directories for the new users.

2. Switch to the hdfs user

```
su - hdfs
```

3. Run the commands necessary to create an HDFS home directory for each user listed below:

   ```
   dev01, dev02, dev13
   qa01, qa02, qa05
   sales01, sales02, sales04
   promo01, promo02
   support01, support02, support09
   ```

   Syntax:

   ```
   $ hdfs dfs -mkdir /user/<user name>
   $ hdfs dfs -chown <user name>:<user name> /user/<user name>
   ```

   The commands for user dev01 are shown below. These commands should be repeated for each user account created in this lab.

```
hdfs dfs -mkdir /user/dev01
hdfs dfs -chown dev01:dev01 /user/dev01
```

4. When you have created and changed ownership of all of the home directories, type exit to return to normal root user permissions.

```
exit
```

Result

You have now:
Create a Linux group matching each home directory; Create Linux accounts for each home directory with memberships in the appropriate groups. Create an HDFS home directory for each user identified in the classroom scenario on **ALL NODES** of your cluster.

Lab 19: Managing YARN Containers and Queues

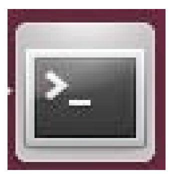
About This Lab

Objective:	To test the consequences of making changes to container sizes, Maximum Application settings, and queue states; configure application preemption and create a set of queues and leaf queues which logically represent the organizations you support and their SLAs
File locations:	N/A
Successful outcome:	**You will:** Perpare for the lab exercise, compare resource usage based on container configuration, test queue settings and state behavior and configure queues to meet service level agreement (SLA) requirements.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***The YARN Capacity Scheduler***

Lab Preparation

Prepare for the lab exercise.

1. Use SSH to login to any cluster node.



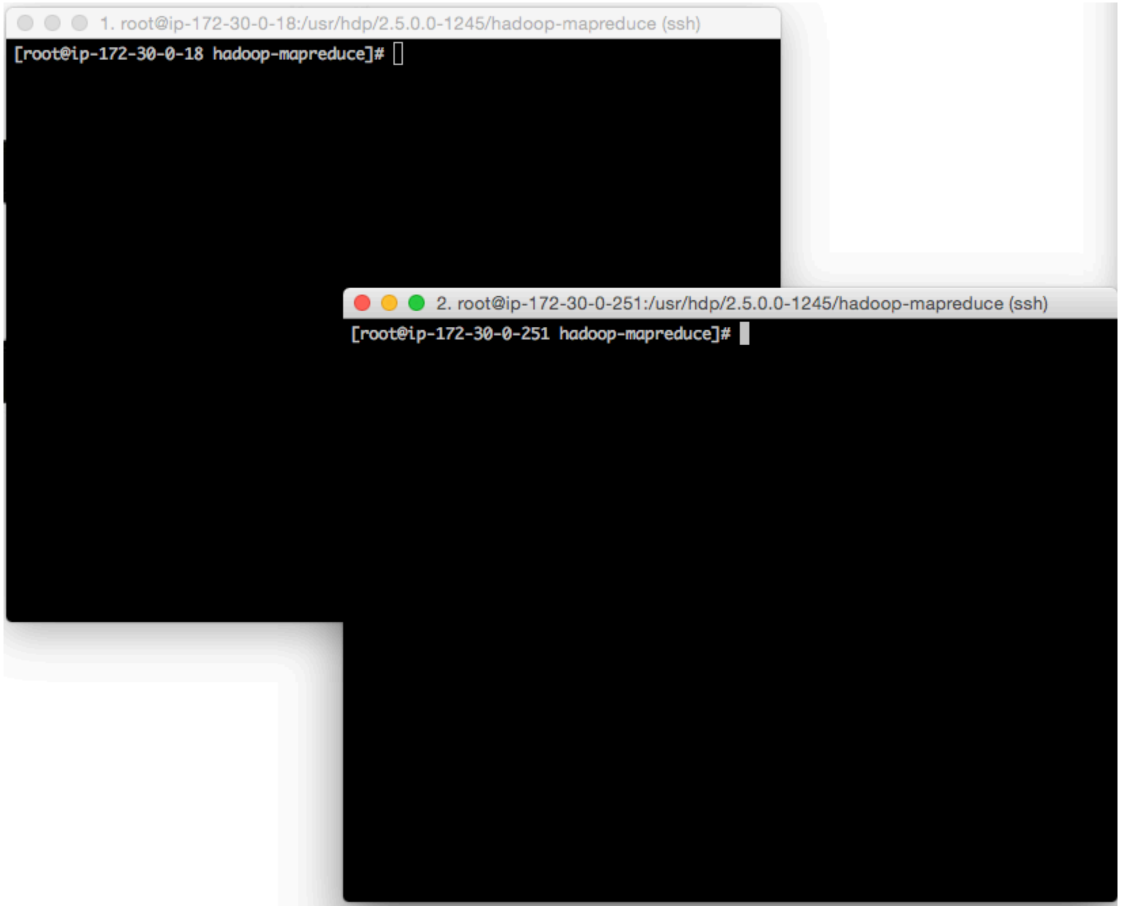
2. Change to the /usr/hdp/<VERSION NUMBER>/hadoop-mapreduce directory.

```
cd /usr/hdp/<VERSION NUMBER>/hadoop-mapreduce
```

3. Open a second terminal window and use SSH to login to another node.
4. Once again, change to the /usr/hdp/<VERSION NUMBER>/hadoop-mapreduce directory.

```
cd /usr/hdp/<VERSION NUMBER>/hadoop-mapreduce
```

5 . Position the two terminal windows so that both are visible on your screen side-by-side.



6 . Type the MapReduce command to execute wordcount in both terminal windows but *do not* press **Enter** to execute it.

```
Terminal One
 yarn jar hadoop-mapreduce-examples.jar wordcount /user/root/constitution.txt
/user/root/output1

Terminal Two
 yarn jar hadoop-mapreduce-examples.jar wordcount /user/root/constitution.txt
/user/root/output2
```

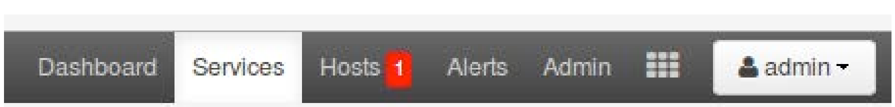
Type, but again *do not* execute, the same command in the second window.

7. Open the browser and connect to the Ambari Server at the URL

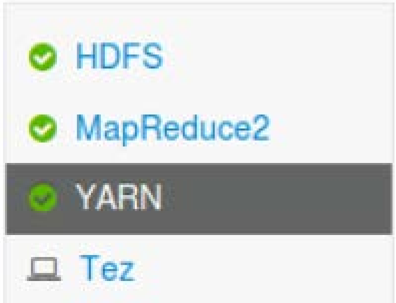
http://<Ambari Server Hostname>:8080

8. Login to the Ambari Web UI using the user name `admin` and the password `BadPass#1`.

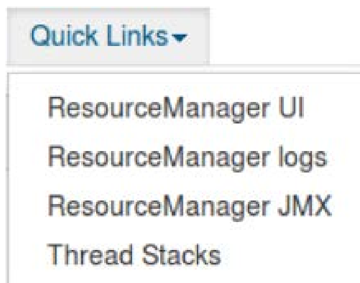
9. Click **Services** in the Ambari Web UI.



10 . Select the **YARN** service on the **Services** page.

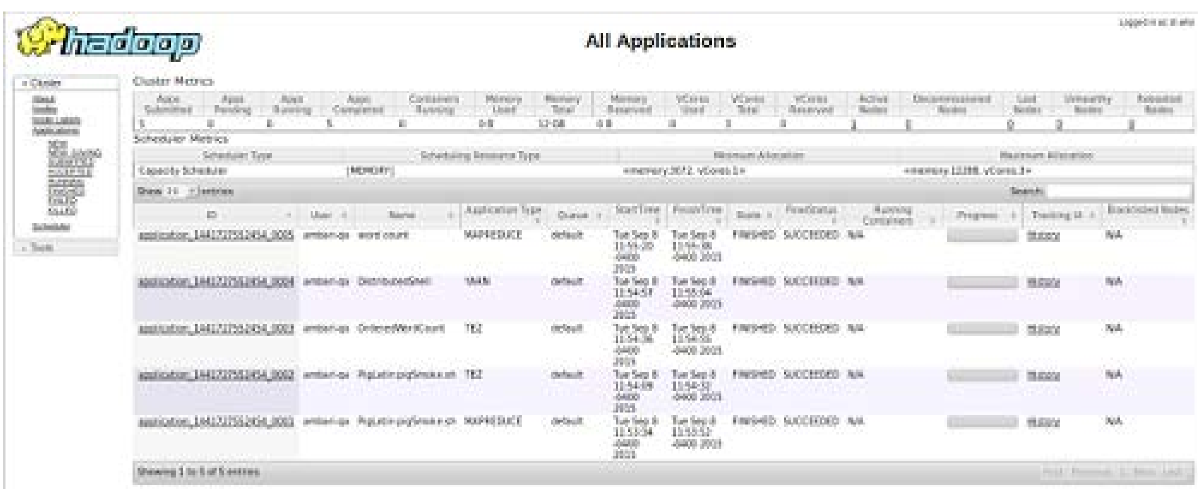


11 . Click **Quick Links** and select **ResourceManager UI**.

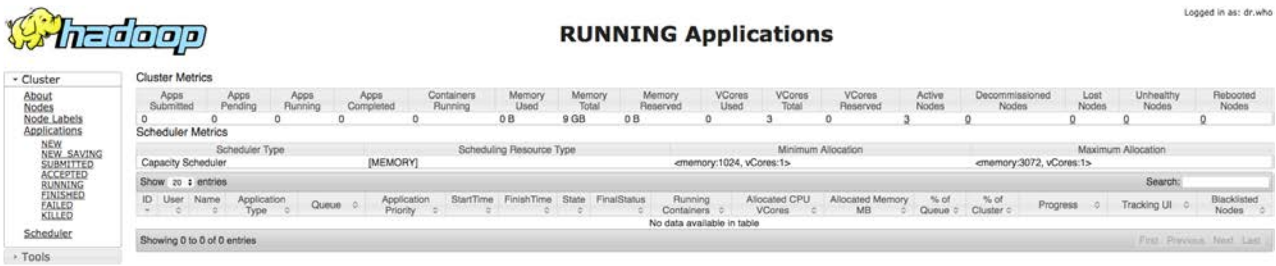


The ResourceManager UI Web interface opens in another browser tab.

NOTE: Depending on your lab environment, the default URL to the ResourceManager UI could be blocked. If simply clicking the quick link fails to open the ResourceManager UI, replace the default URL in the browser tab with `http://<External IP Address of RM>:8088`.

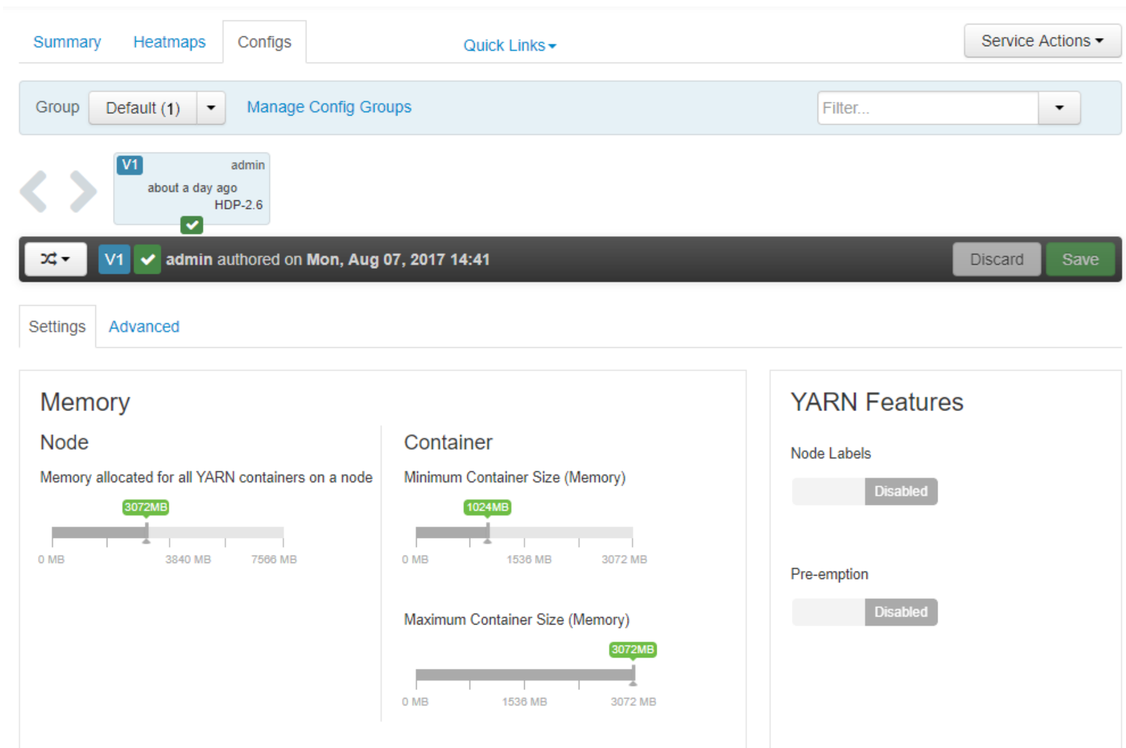


12 . Click on the **RUNNING** link.
No applications are currently running.
This page will be refreshed in a moment once applications are running. Leave this tab open.



13 . Go back to the Ambari Web UI YARN **Services** page and click **Configs**.

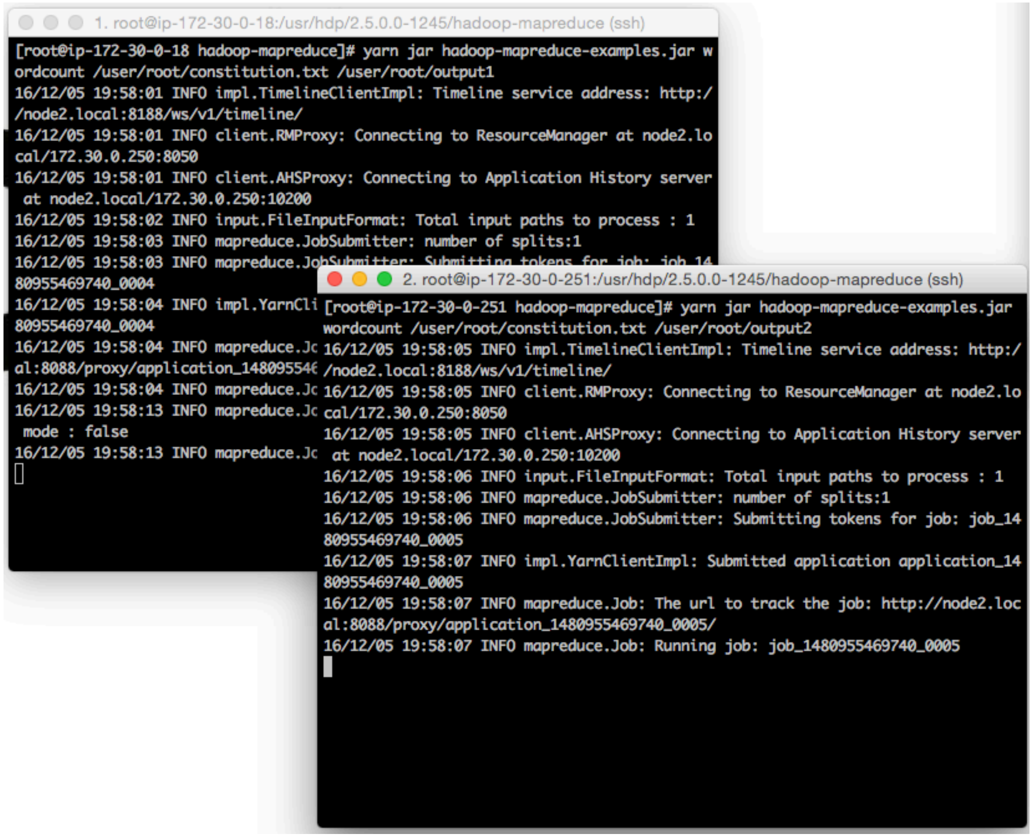
NOTE: The default minimum container size.



Comparing Resource Usage

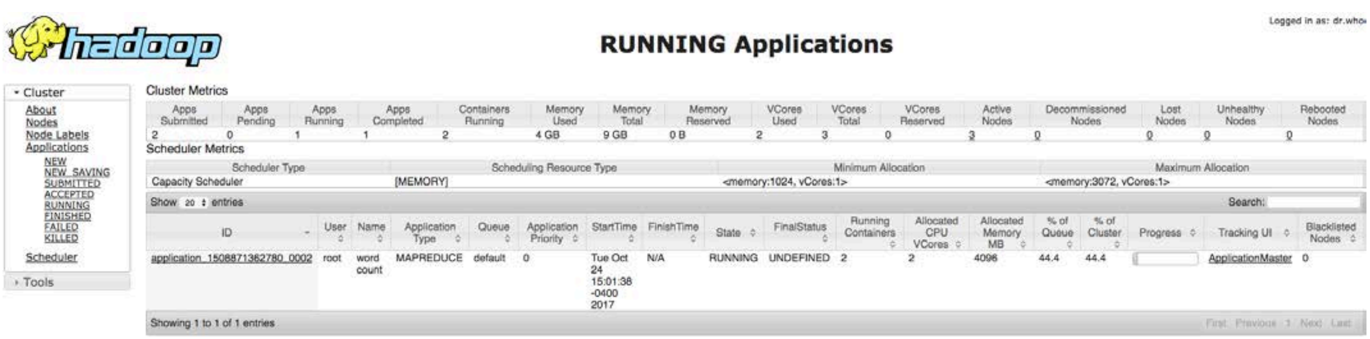
Compare resource usage based on container configuration.

1 . Launch two instances of wordcount YARN job by clicking on the first terminal window and pressing the **Enter** key, then immediately clicking on the second terminal window and pressing the **Enter** key again.



2 . Quickly click the ResourceManager UI browser tab and refresh the **RUNNING** jobs page. Continue refreshing until the page reports more than two containers have been provisioned. Note the maximum total number of containers and the total cluster memory that can be utilized by the applications.

NOTE: Your numbers may differ from the screen captures if the total resources available or default container configuration varies.

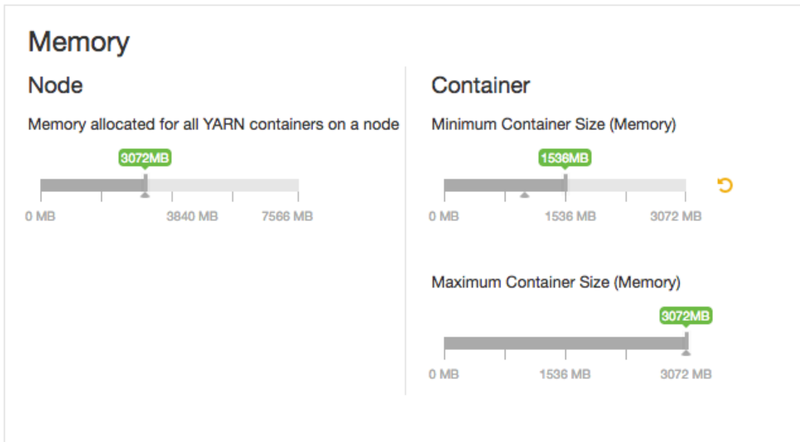


3 . Once the applications have finished running, click on the YARN Services **Configs** tab.

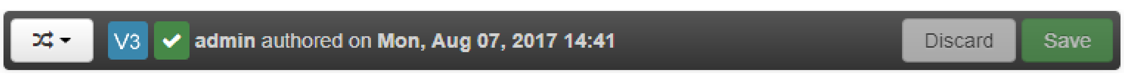
Using the scroll bar, adjust the **Minimum Container Size (Memory)** up to approximately `50%` of the default **Maximum Container Size (Memory)** setting.

NOTE: By default, the maximum container size will equal the total memory available to YARN per node. In the example shown, the maximum container size is approximately 3 GB, so the minimum container size was set to approximately 1.5 GB.

Ask your instructor for the settings to use if your configuration differs from what is shown.



4 . Click **Save** to save the modifications.



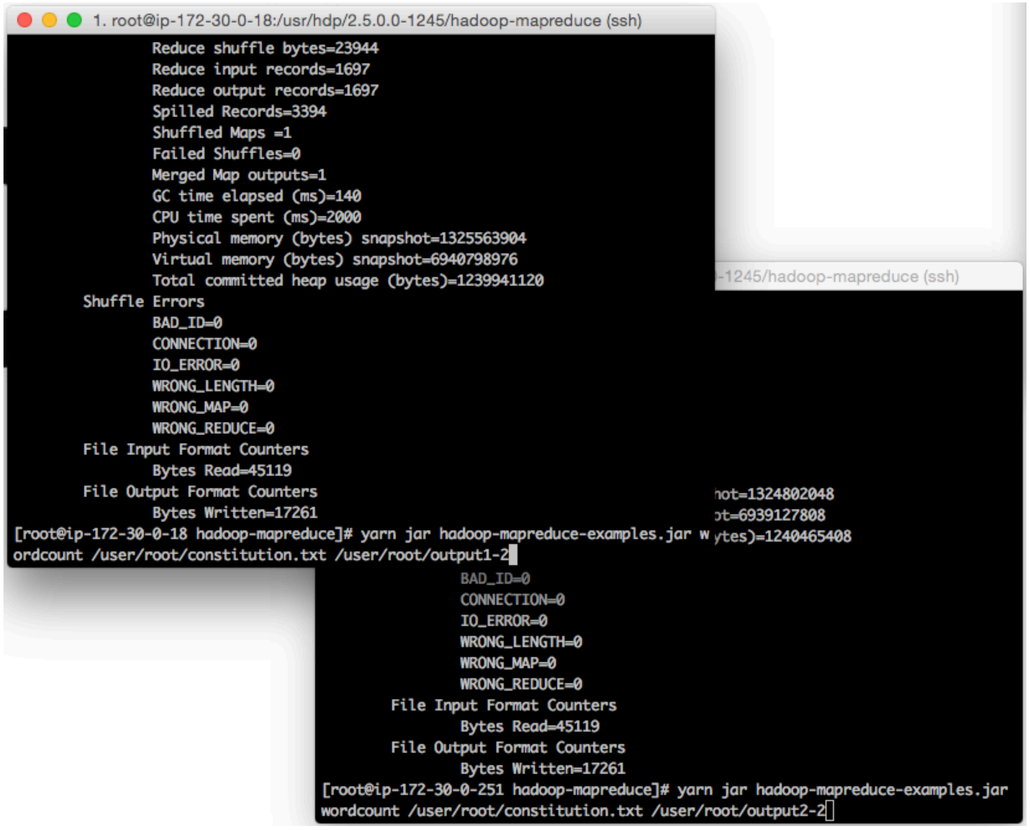
Click **Save** again in the Save Configuration window.

Click **OK** in the Dependent Configurations window presented by the Ambari Guided Configuration feature to accept all suggested changes.
Then click **OK** in the confirmation window.

5 . Restart any services as indicated in the Ambari Web UI.

6 . Return to your two terminal windows.

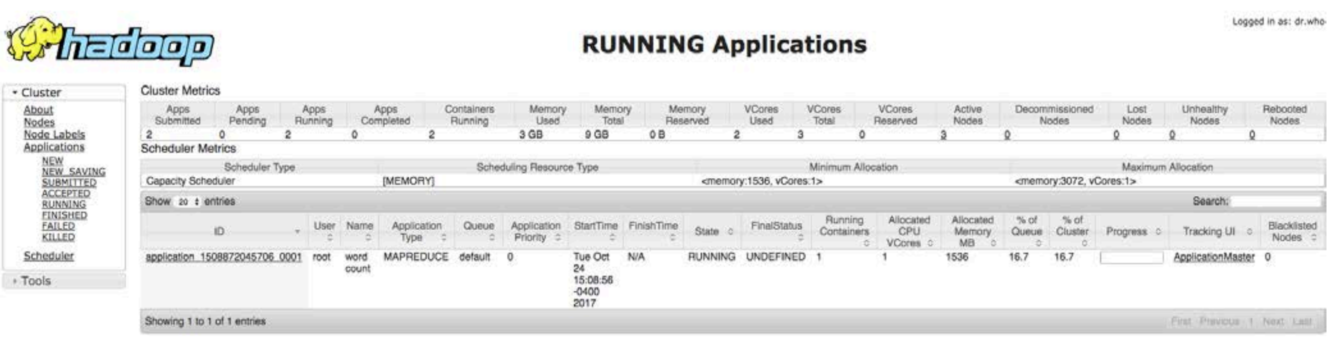
In each one, press the **Up Arrow** on your keyboard to display the previous *yarn jar* command.

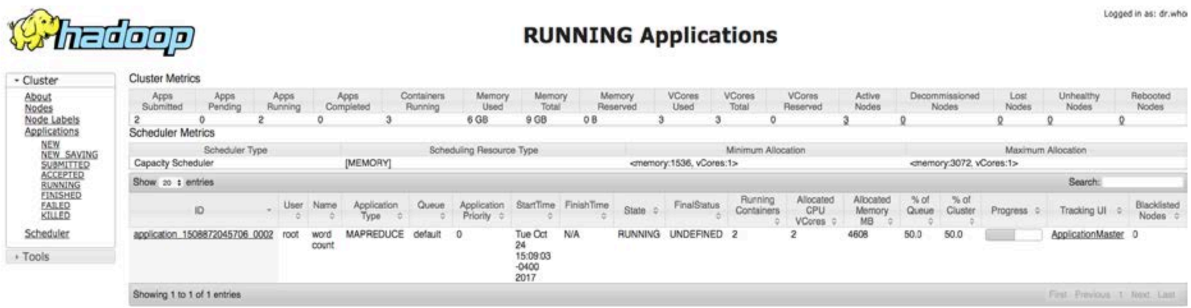


7 . Change the output directories in Terminal One from /user/root/output1 to /user/root/output1-2 and Terminal Two from /user/root/output2 to /user/root/output2-2

8 . As before, press the **Enter** key in each terminal window and then quickly click the ResourceManager UI browser tab, which should still be on the **RUNNING** information page.

Once again, refresh the page until more than two containers are shown as provisioned. Note the maximum total number of containers and total amount of memory the applications are able to use this time.





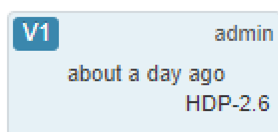
Why are the applications able to access less cluster memory with this set of jobs than they did before?

What is the maximum number of YARN jobs that could be provisioned and running at a time on this cluster given the current configuration?

Remember that each job requires a container for the ApplicationMaster in addition to any scheduled jobs.

9. To undo the YARN configuration changes, click on the Ambari Web UI YARN **Services** page and click **Configs**.

   Find the **V1** configuration settings box and click on it.



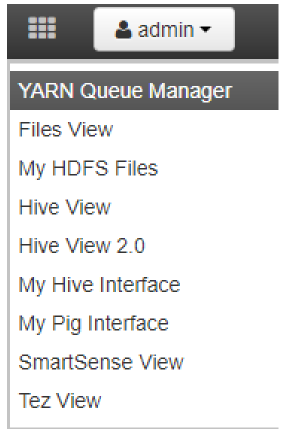
10. Click the green **Make V1 Current** button, accept all dependent configuration suggestions, and then restart any affected services.

   If you want to confirm all settings are back to the default, run the two YARN applications one more time in the terminal windows and confirm the number of containers provisioned and memory used match the results from the first part of this lab section.

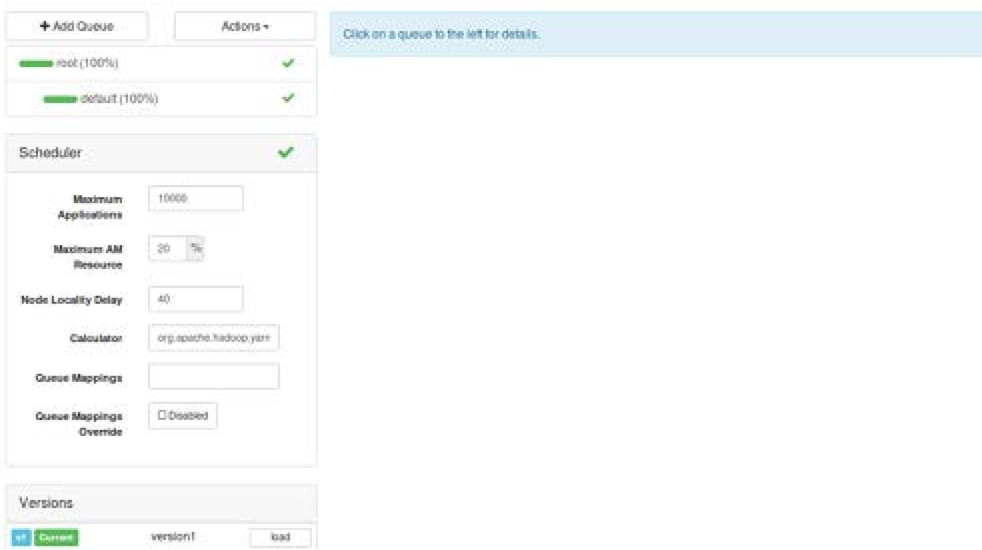
Testing Queue Settings and Behavior

Test queue settings and state behavior.

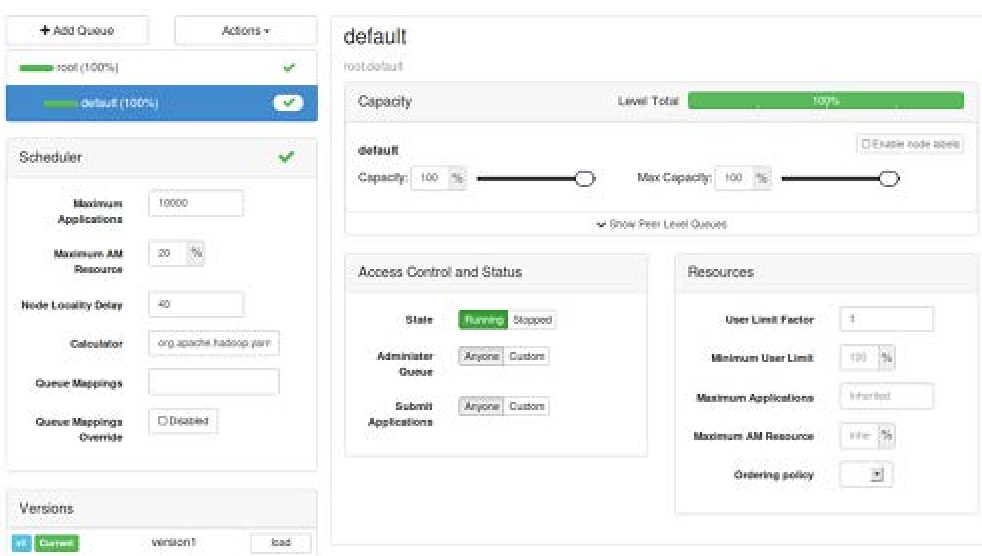
1. Click the **Views** icon and select the View named **YARN Queue Manager**.



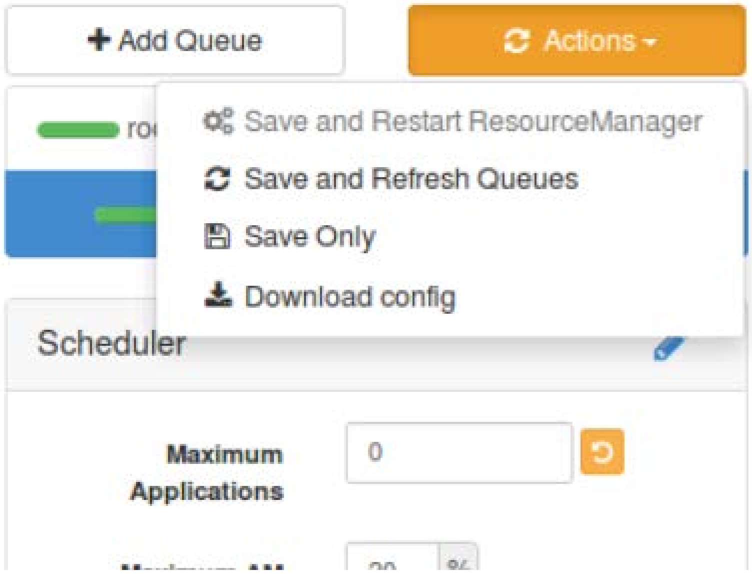
The YARN Queue Manager View opens.



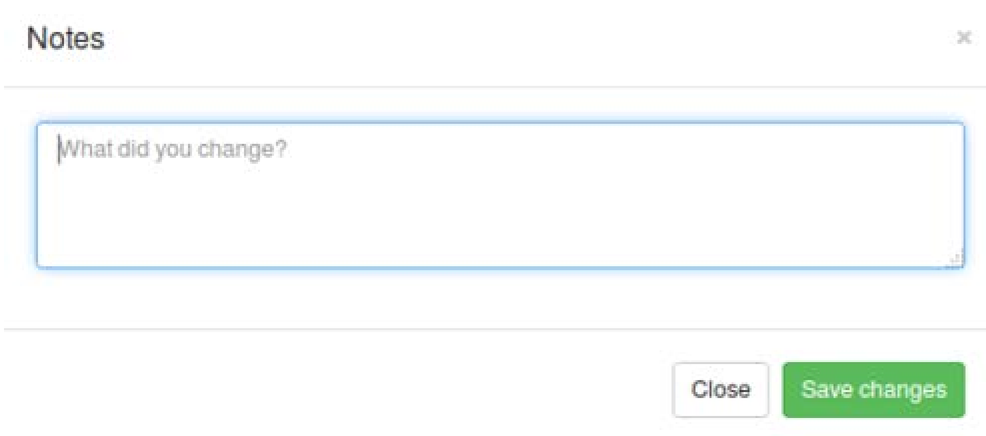
2 . Click the **default** queue to view its settings.



Under Scheduler, find the **Maximum Applications** setting and change it to `0`.
The **Actions** menu button will turn orange.
Click it, and select **Save and Refresh Queues**.

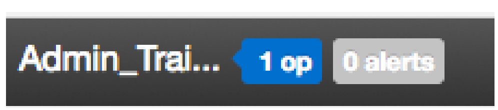


A Notes window opens asking you to describe what you have changed.
Make a note, and then click **Save Changes**.



It will take a few seconds for the queues to refresh.

At the top of the Ambari Web UI, you will see a blue notification box appear that lists **1 op** as running.



When the queue refresh is complete, the box will turn gray and will display **0 ops**.

3 . Return to either open terminal window and press the Up Arrow key to display the previous *yarn jar* command.

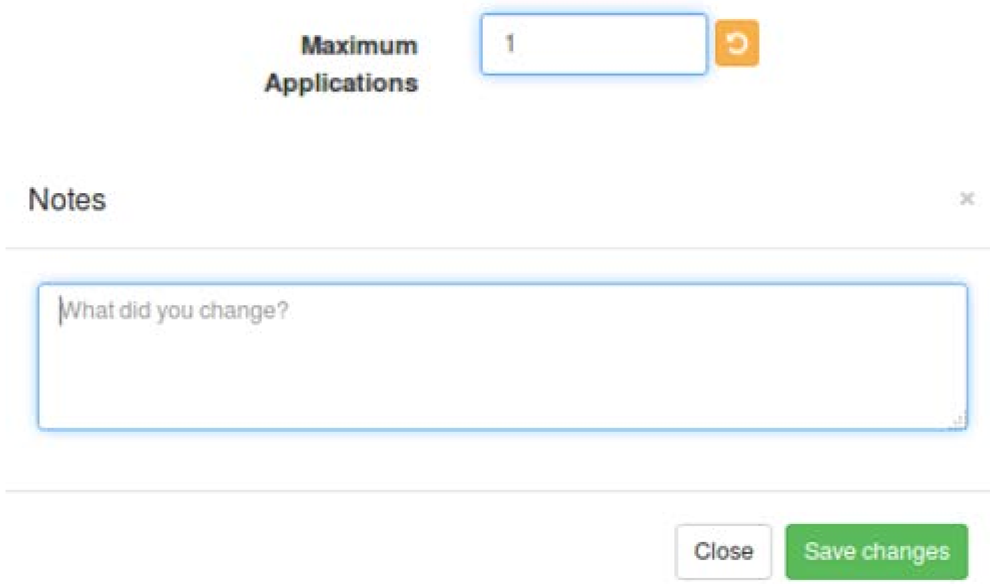
Then press the **Enter** key.

4 . The application should fail to launch.

Scroll up in the command output until you locate the `java.io.IOException` notification. This explains that the queue already has 0 applications, and therefore cannot accept any more.

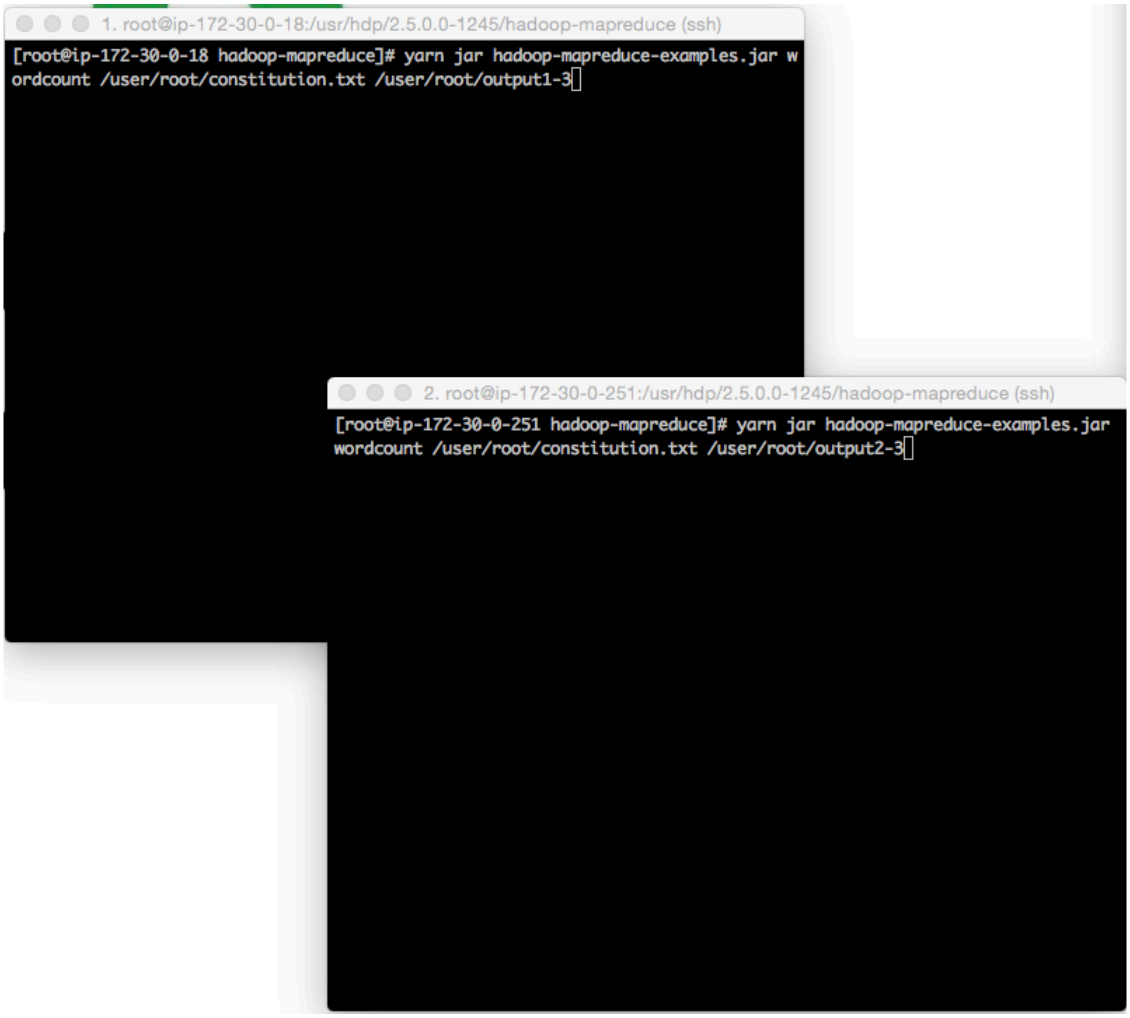
```
java.io.IOException: org.apache.hadoop.yarn.exceptions.YarnException: Failed to submit ap
plication_1442799443779_0042 to YARN : org.apache.hadoop.security.AccessControlException:
 Queue root.default already has 0 applications, cannot accept submission of application:
application_1442799443779_0042
```

5 . Return to the YARN Queue Manager and set **Maximum Applications** to `1`. Then **Save and Refresh Queues** as before.



6 . Open both terminal windows and press the **Up Arrow** key so that both the *yarn jar* command are displayed.

7 . Change the output directories in Terminal One from /user/root/output1-2 to /user/root/output1-3 and Terminal Two from /user/root/output2 to /user/root/output2-3.



8. Press the **Enter** key with one of the terminal windows highlighted,
then quickly switch to the other terminal window and press the **Enter** key there as well.

9. There may be some lag when attempting to switch to the other terminal window, however there should be sufficient time to launch the second application before the first one finishes.

   NOTE: That the first job successfully launches and runs.

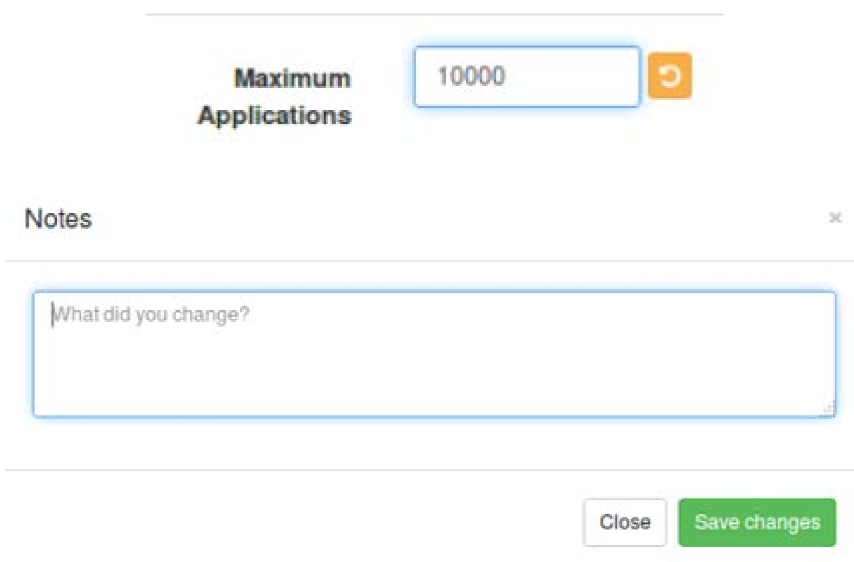
```
Samples per Map = 10
Wrote input for Map #0
Wrote input for Map #1
Wrote input for Map #2
Wrote input for Map #3
Wrote input for Map #4
Starting Job
15/09/21 17:06:19 INFO impl.TimelineClientImpl: Timeline service address: http://node1:81
88/ws/v1/timeline/
15/09/21 17:06:20 INFO client.RMProxy: Connecting to ResourceManager at node1/172.17.0.2:
8050
15/09/21 17:06:20 INFO input.FileInputFormat: Total input paths to process : 5
15/09/21 17:06:20 INFO mapreduce.JobSubmitter: number of splits:5
15/09/21 17:06:21 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_14427994437
79_0043
15/09/21 17:06:21 INFO impl.YarnClientImpl: Submitted application application_14427994437
79_0043
15/09/21 17:06:21 INFO mapreduce.Job: The url to track the job: http://node1:8088/proxy/a
pplication_1442799443779_0043/
15/09/21 17:06:21 INFO mapreduce.Job: Running job: job_1442799443779_0043
15/09/21 17:06:28 INFO mapreduce.Job: Job job_1442799443779_0043 running in uber mode : f
alse
15/09/21 17:06:28 INFO mapreduce.Job:  map 0% reduce 0%
```

The second job, however, fails.

Scroll up in the command output and locate the `java.io.IOException` message. Note that this time it identifies that the queue already has one application running, and cannot accept the submission of an additional application.

```
java.io.IOException: org.apache.hadoop.yarn.exceptions.YarnException: Failed to
submit application_1442799443779_0044 to YARN : org.apache.hadoop.security.Acces
sControlException: Queue root.default already has 1 applications, cannot accept
submission of application: application_1442799443779_0044
```

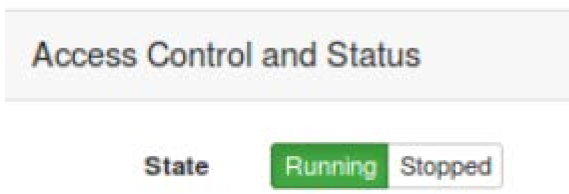
10 . Go back to the YARN Queue Manager and set **Maximum Applications** back to `10000`. (10,000) Once again, click **Save and Refresh Queues**.



11 . Once the queue refresh operation has completed, return to either one of the open terminal windows, click the **Up Arrow** key to display the *yarn jar* command, and press the **Enter** key.

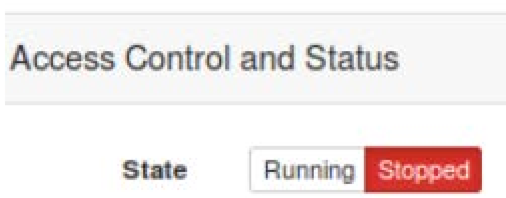
```
Samples per Map = 10
Wrote input for Map #0
Wrote input for Map #1
Wrote input for Map #2
Wrote input for Map #3
Wrote input for Map #4
Starting Job
15/09/21 17:06:19 INFO impl.TimelineClientImpl: Timeline service address: http://node1:81
88/ws/v1/timeline/
15/09/21 17:06:20 INFO client.RMProxy: Connecting to ResourceManager at node1/172.17.0.2:
8050
15/09/21 17:06:20 INFO input.FileInputFormat: Total input paths to process : 5
15/09/21 17:06:20 INFO mapreduce.JobSubmitter: number of splits:5
15/09/21 17:06:21 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_14427994437
79_0043
15/09/21 17:06:21 INFO impl.YarnClientImpl: Submitted application application_14427994437
79_0043
15/09/21 17:06:21 INFO mapreduce.Job: The url to track the job: http://node1:8088/proxy/a
pplication_1442799443779_0043/
15/09/21 17:06:21 INFO mapreduce.Job: Running job: job_1442799443779_0043
15/09/21 17:06:28 INFO mapreduce.Job: Job job_1442799443779_0043 running in uber mode : f
alse
15/09/21 17:06:28 INFO mapreduce.Job:  map 0% reduce 0%
```

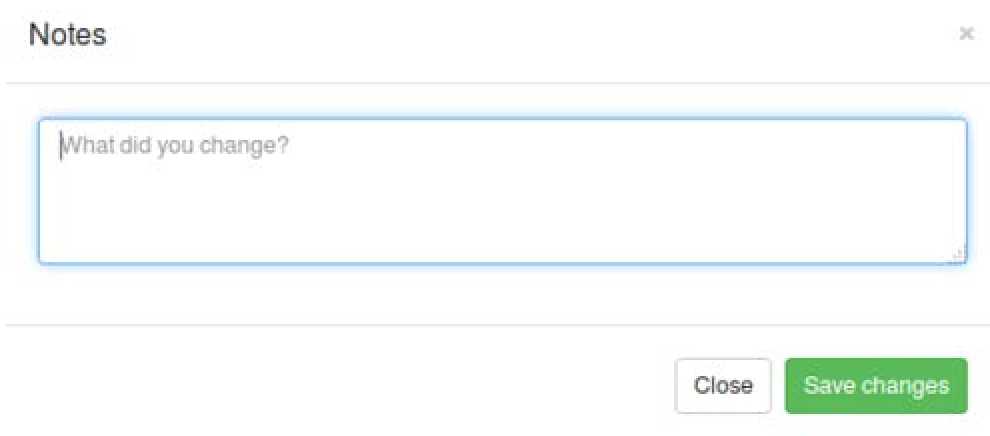
12 . While this job is running, quickly go back to the YARN Queue Manager, and note the current **State** of the queue is **Running**.



13 . Click the **Stopped** button. Then quickly click **Save and Refresh Queues**.

NOTE: You should be able to perform all of these tasks while the job you launched is still running.





14 . Stopping the queue while an application is running might in fact cause a temporary halt...

```
15/09/21 17:16:55 INFO mapreduce.Job:  map 80% reduce 7%
15/09/21 17:16:55 INFO mapreduce.Job: Task Id : attempt_1442799443779_0045_m_000
003_0, Status : FAILED
Container killed on request. Exit code is 137
Container exited with a non-zero exit code 137
Killed by external signal
```

…however the application will continue to run, and should successfully complete.

```
1. root@ip-172-30-0-18:/usr/hdp/2.5.0.0-1245/hadoop-mapreduce (ssh)
                Reduce input groups=1697
                Reduce shuffle bytes=23944
                Reduce input records=1697
                Reduce output records=1697
                Spilled Records=3394
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=137
                CPU time spent (ms)=2170
                Physical memory (bytes) snapshot=1924419584
                Virtual memory (bytes) snapshot=9610948608
                Total committed heap usage (bytes)=1788870656
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=45119
        File Output Format Counters
                Bytes Written=17261
[root@ip-172-30-0-18 hadoop-mapreduce]#
```

15 . After all applications have finished, with the default queue still stopped, click the **Up arrow** in a terminal window and display the *yarn jar* command, then press the **Enter** key to try to start a new application.

The application will fail with a YARN exception that states the queue is **STOPPED** and cannot accept the application submission.

Leave the default queue in the **STOPPED** state.

```
1. root@ip-172-30-0-243:/usr/hdp/2.6.2.0-205/hadoop-mapreduce (ssh)
17/10/24 19:21:31 INFO mapreduce.JobSubmitter: Cleaning up the staging area /user/root/.staging/job_150887
2400198_0002
java.io.IOException: org.apache.hadoop.yarn.exceptions.YarnException: Failed to submit application_1508872
400198_0002 to YARN : org.apache.hadoop.security.AccessControlException: Queue root.default is STOPPED. Ca
nnot accept submission of application: application_1508872400198_0002
        at org.apache.hadoop.mapred.YARNRunner.submitJob(YARNRunner.java:317)
        at org.apache.hadoop.mapreduce.JobSubmitter.submitJobInternal(JobSubmitter.java:240)
        at org.apache.hadoop.mapreduce.Job$10.run(Job.java:1290)
        at org.apache.hadoop.mapreduce.Job$10.run(Job.java:1287)
        at java.security.AccessController.doPrivileged(Native Method)
        at javax.security.auth.Subject.doAs(Subject.java:422)
        at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1866)
        at org.apache.hadoop.mapreduce.Job.submit(Job.java:1287)
        at org.apache.hadoop.mapreduce.Job.waitForCompletion(Job.java:1308)
        at org.apache.hadoop.examples.WordCount.main(WordCount.java:87)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.apache.hadoop.util.ProgramDriver$ProgramDescription.invoke(ProgramDriver.java:71)
        at org.apache.hadoop.util.ProgramDriver.run(ProgramDriver.java:144)
        at org.apache.hadoop.examples.ExampleDriver.main(ExampleDriver.java:74)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.apache.hadoop.util.RunJar.run(RunJar.java:233)
        at org.apache.hadoop.util.RunJar.main(RunJar.java:148)
Caused by: org.apache.hadoop.yarn.exceptions.YarnException: Failed to submit application_1508872400198_000
2 to YARN : org.apache.hadoop.security.AccessControlException: Queue root.default is STOPPED. Cannot accep
t submission of application: application_1508872400198_0002
        at org.apache.hadoop.yarn.client.api.impl.YarnClientImpl.submitApplication(YarnClientImpl.java:271
)
        at org.apache.hadoop.mapred.ResourceMgrDelegate.submitApplication(ResourceMgrDelegate.java:291)
        at org.apache.hadoop.mapred.YARNRunner.submitJob(YARNRunner.java:301)
        ... 22 more
[root@ip-172-30-0-243 hadoop-mapreduce]#
```

Configure queues to meet SLA requirements

Configure queues to meet service level agreement (SLA) requirements according to the following scenario:

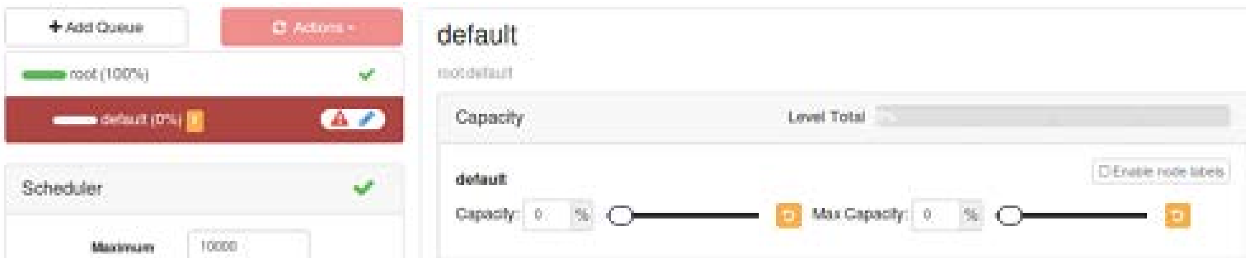
Organization	Minimum Guarantee	Maximum Available
Engineering	60%	100%
Marketing	30%	100%
Support	10%	100%

Preemption is enabled on the cluster so that minimum guarantees will be available to users.

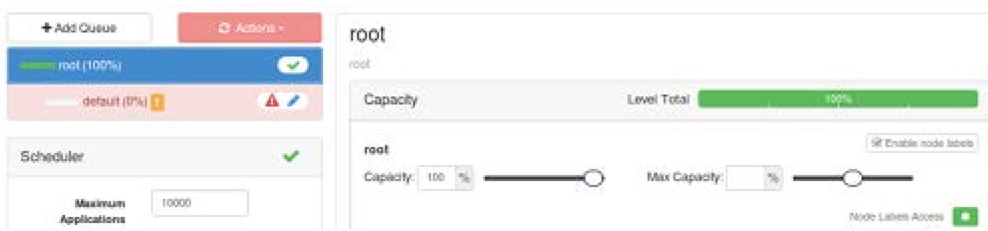
Configure two leaf queues for the Engineering queue with the following characteristics:

Organization	Minimum Guarantee	Maximum Available
Dev	50%	100%
QA	50%	100%

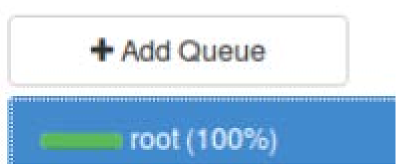
1. Begin by removing all resources from the **default** queue by setting both the Capacity and Maximum Capacity settings to `0%`.



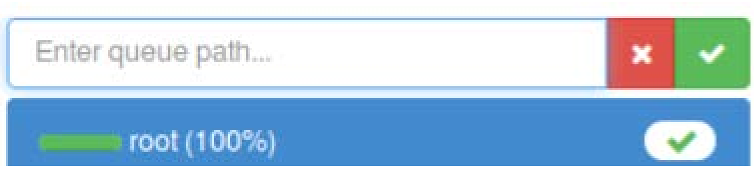
2. Click the **root q**ueue.



3. With the **root** queue highlighted, click the **Add Queue** button above it.

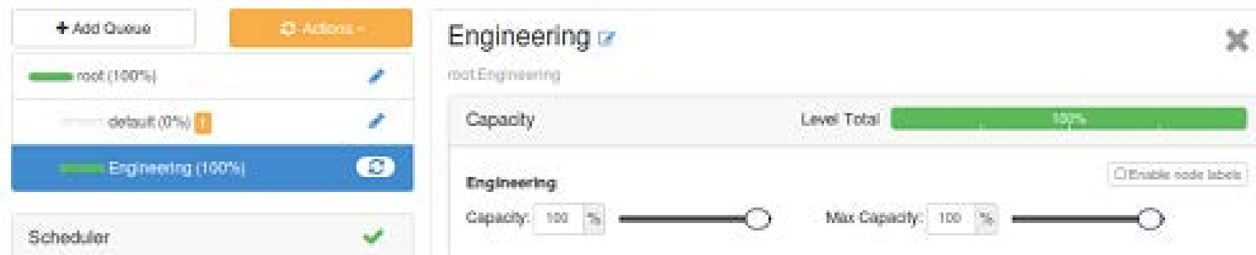


4. The **Enter queue path** text box is displayed.

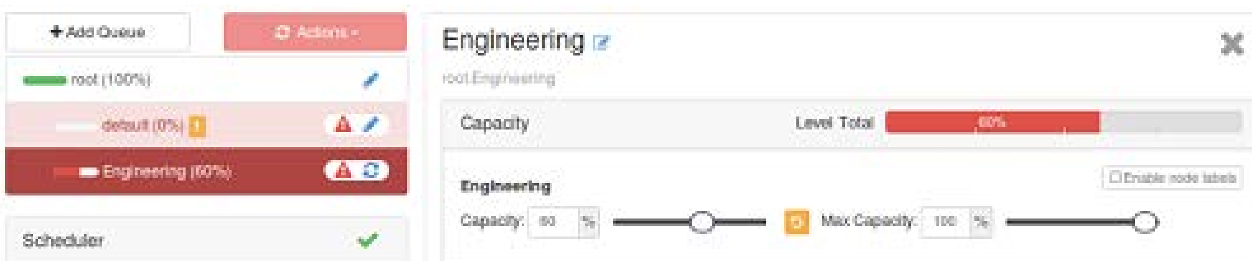


Type `Engineering` in the text box, then click the green checkmark button to the right.

5 . The Engineering queue is created with all available resources assigned to it (currently 100%) by default.

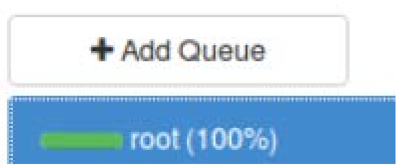


Set the **Capacity** value to `60%` and leave the **Max Capacity** value at `100%`.

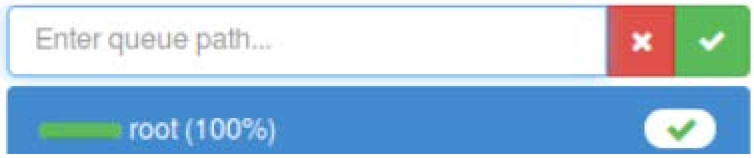


6 . Click the **root** queue.

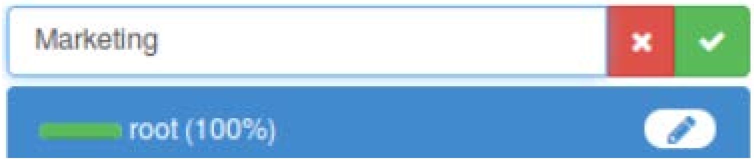
7 . With the **root** queue highlighted, once again click the **Add Queue** button above it.



8 . The **Enter queue path** text box is displayed.



Type `Marketing` in the text box, then click the green checkmark button to the right.

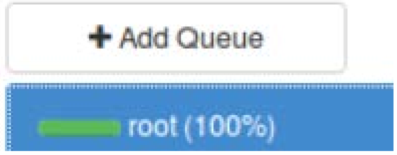


9 . The Marketing queue is created with all available resources assigned to it (currently 40%) by default.

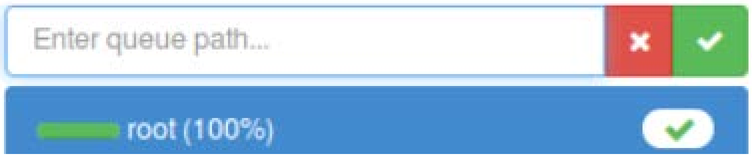
Set the **Capacity** value to `30%` and the **Max Capacity** value to `100%`.

10 . Click the **root** queue.

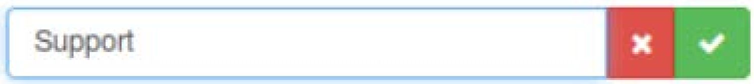
11 . With the **root** queue highlighted, once again click the **Add Queue** button above it.



12 . The **Enter queue path** text box is displayed.



Type `Support` in the text box, then click the green checkmark button to the right.

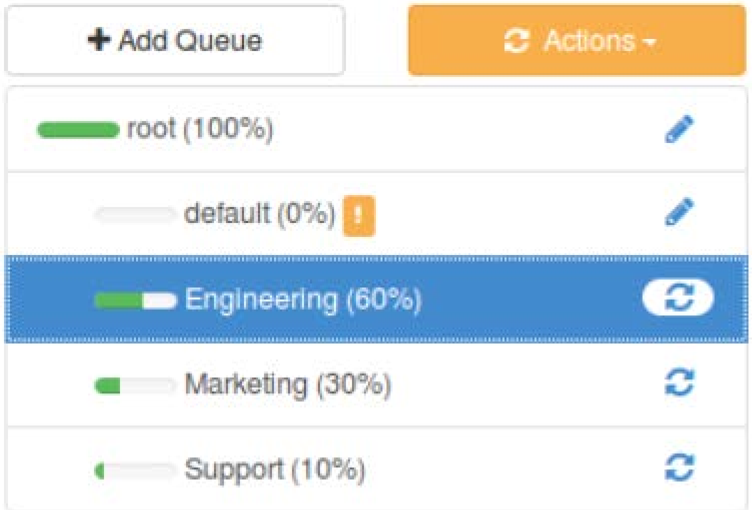


13 . The Support queue is created with all available resources assigned to it (currently 10%) by default.

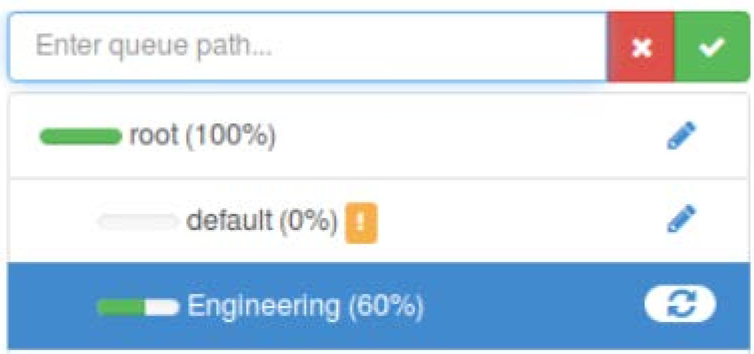
Leave the **Capacity** value at `10%` and set the **Max Capacity** value to `100%`.

14 . Next create the Engineering leaf queues.

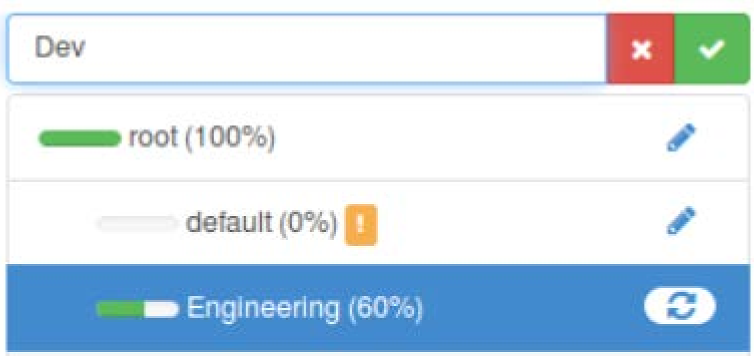
Start by clicking the **Engineering** queue.



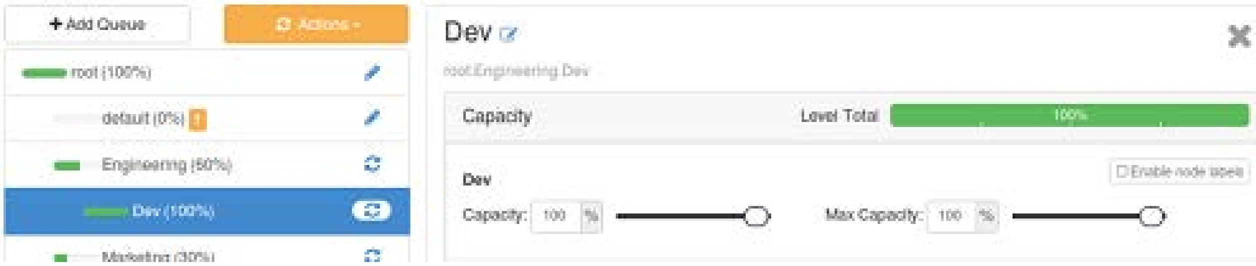
15 . Click the **Add Queue** button.



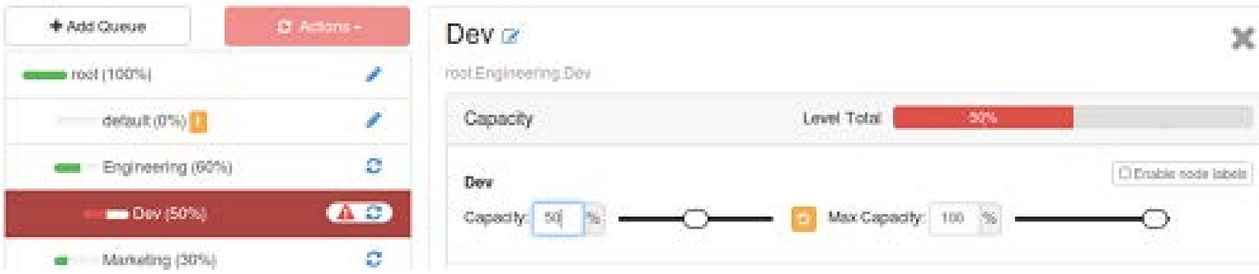
16 . In the text box, type `Dev` and click the green checkmark button.



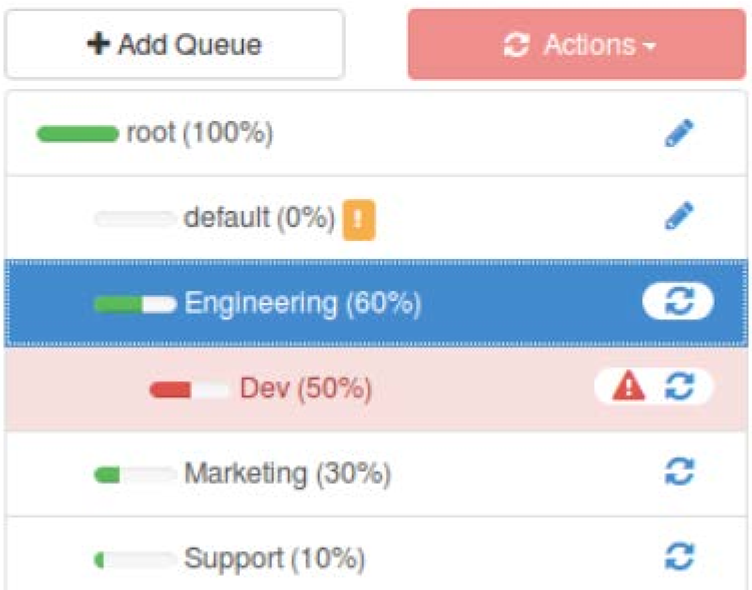
17 . The Dev leaf queue is created with all available resources assigned to it (currently 100% of the resources available to the Engineering queue) by default.



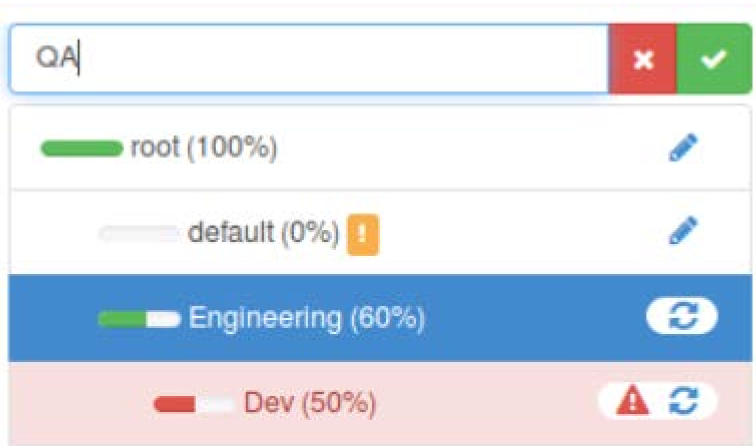
Set the **Capacity** value to `50%` and leave the **Max Capacity** value at `100%`.



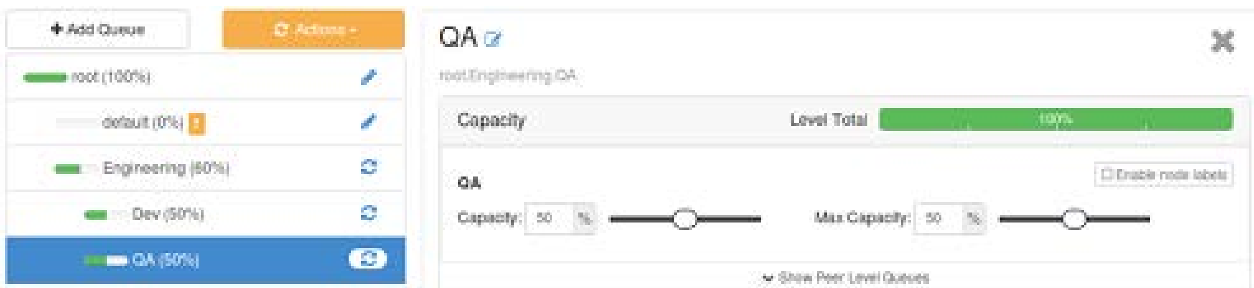
18 . Once again, click the **Engineering** queue.



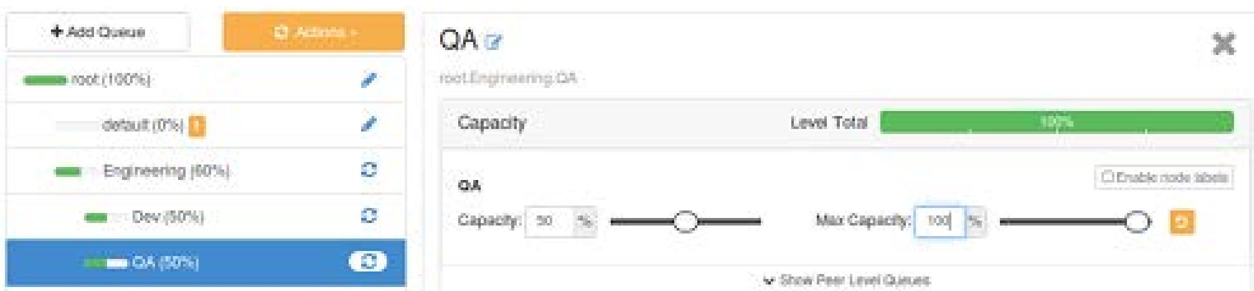
19 . Click the **Add Queue** button and type `QA` in the text box. Then click the green checkmark button.



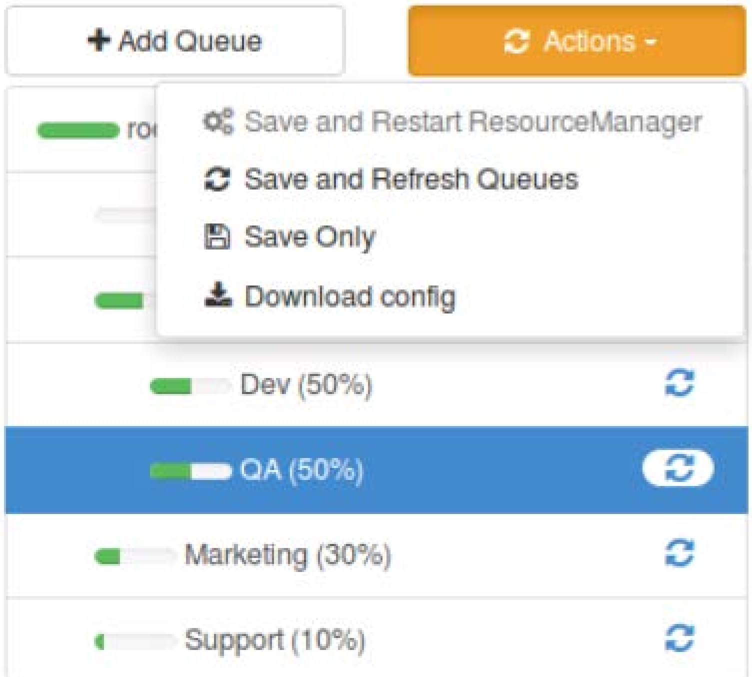
20 . The QA leaf queue is created with all available resources assigned to it (currently 50% of the resources available to the Engineering queue) by default.



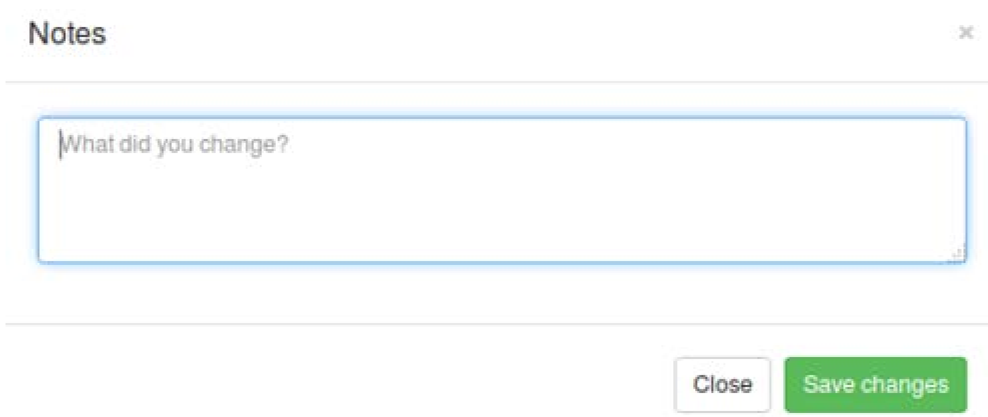
Leave the **Capacity** value at `50%` and set the **Max Capacity** value to `100%`.



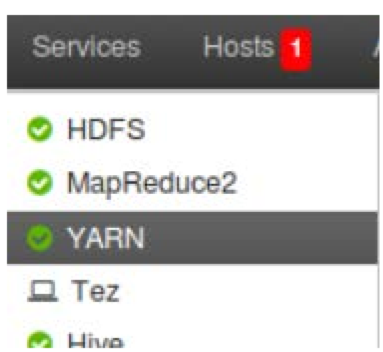
21 . Click the orange **Actions** menu button and select **Save and Refresh Queues**.



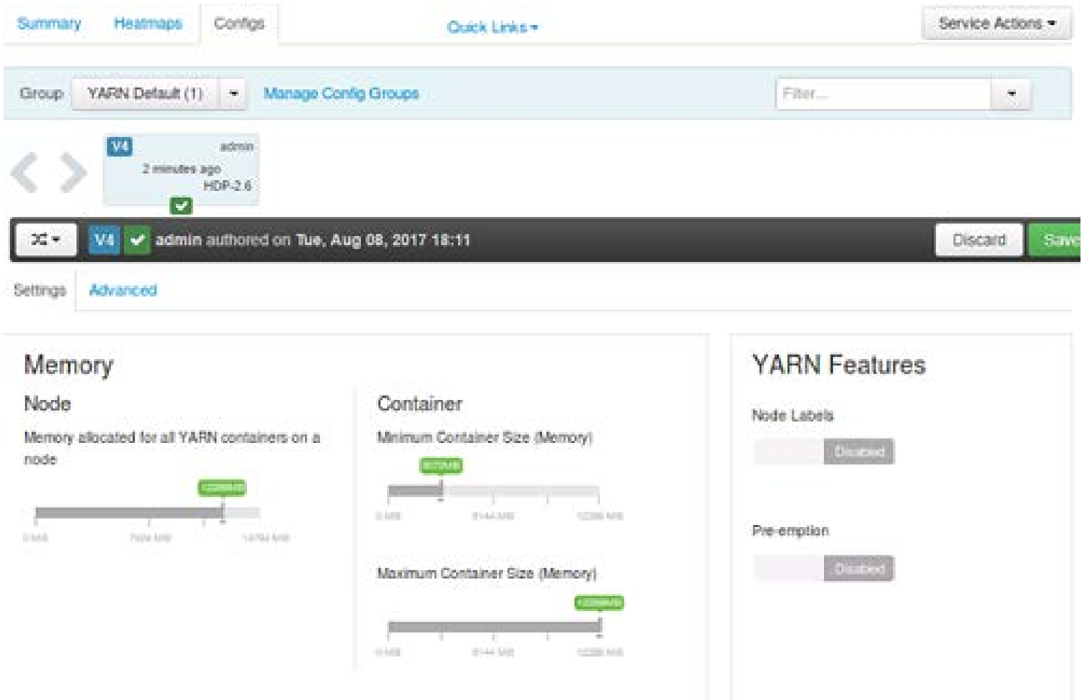
Type notes to describe what was changed, then click **Save changes**.



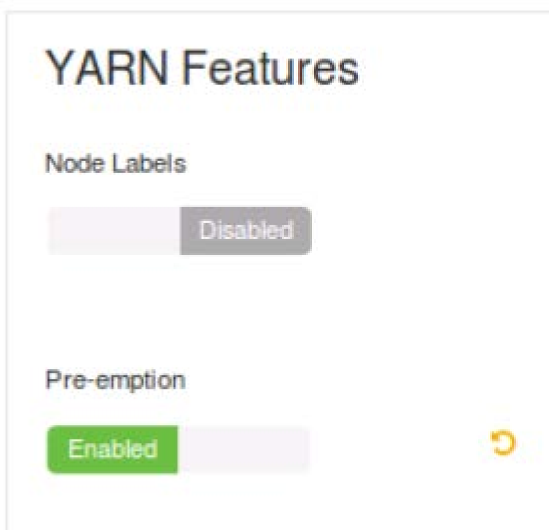
22 . To set preemption, in the Ambari Web UI click **Services** and select **YARN**.



23 . Click the **Configs** tab.



24 . Under YARN Features, set the **Pre-emption** setting to **Enabled**.



25 . Click the green **Save** button.
Add notes to indicate what was changed.
Click **Save**.
Then click **OK**.

26 . Restart any services as indicated in the Ambari Web UI.

Result

You have now:
Perpared for the lab exercise, compared resource usage based on container configuration, tested queue settings and state behavior, and configured queues to meet service level agreement (SLA) requirements.

Lab 20: Managing YARN ACLs and User Limits

About This Lab

Objective:	To configure queue settings so that minimum user limit settings are strictly enforced
File locations:	N/A
Successful outcome:	**You will:** Configure YARN ACLs and default queue mapping; test minimum user limits; and re-establish access to the default queue.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***The YARN Capacity Scheduler***

Lab Preparation

Open multiple windows and login as different user accounts in order to test YARN ACLs and user limits.

1 . Use SSH to login to any cluster node.

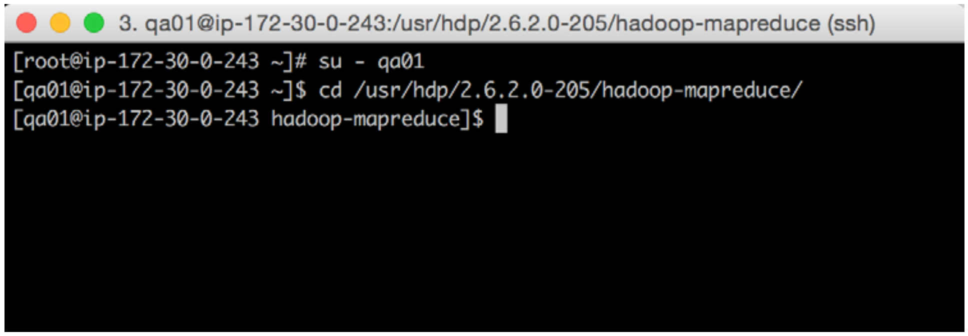
2 . Switch to the dev01 user.

```
su - dev01
```

3 . When you switch to this user, the prompt should look like this:

4 . Open another terminal window. Use SSH to login to your cluster node. Switch to the qa01 user in this window.

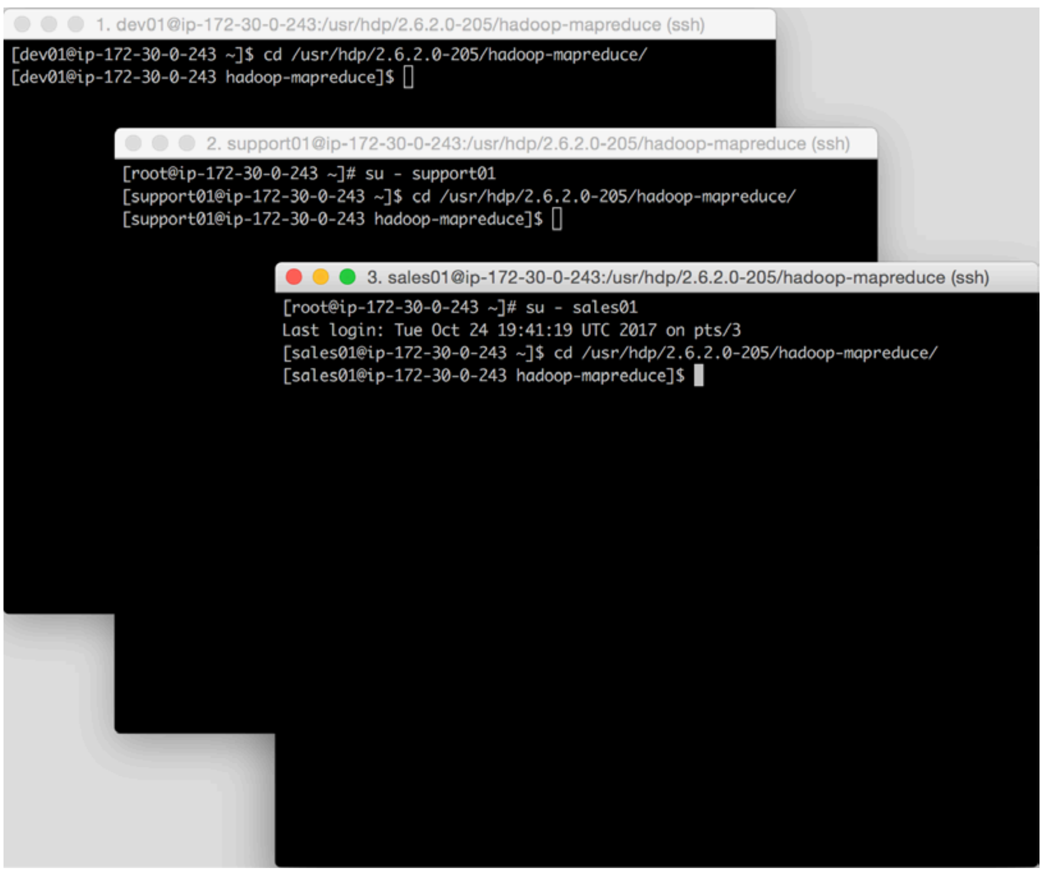
```
su - qa01
```



Repeat this process for all of the following users, each one in a new terminal window.
When you are finished, you should have five open terminals, each one acting as a different user.

promo01
support01
sales01

Position the terminal windows so that you can move between them easily.



5. In each open terminal, change directories to `/usr/hdp/<VERSION NUMBER>/hadoop-mapreduce`.

```
cd /usr/hdp/<VERSION NUMBER>/hadoop-mapreduce
```

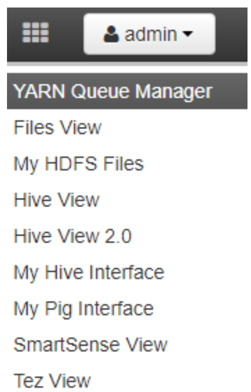
Configuring YARN ACLs and Default Queue Mapping

Configure YARN ACLs and default queue mapping.

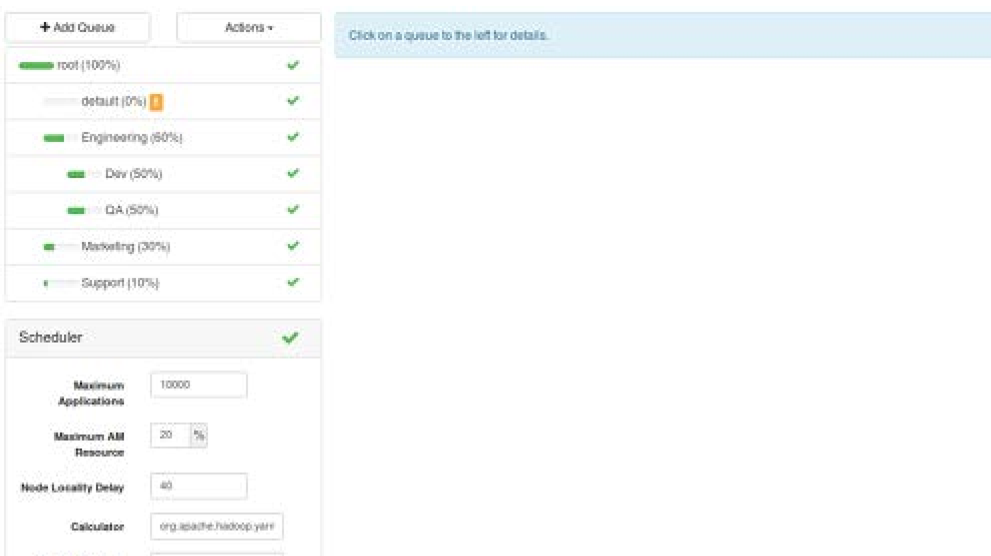
1. Open the browser and connect to the Ambari Server at the URL

   http://<Ambari Server Hostname>:8080

2. Click the **Views** icon and select the View named **YARN Queue Manager**.



The YARN Queue Manager View opens.

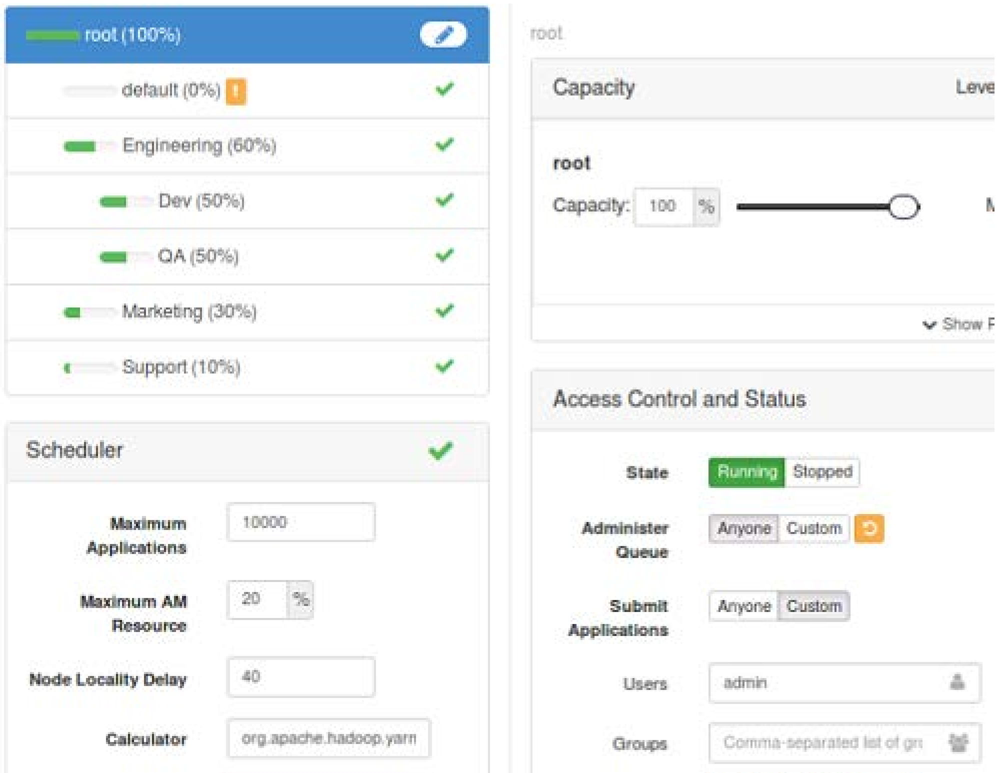


3 . Because queue permissions are cumulative from the top level down, you want to start by removing all permissions from default root queue for everyone except the admin user.

To do this, click on the **root** queue and locate the **Submit Applications** setting.

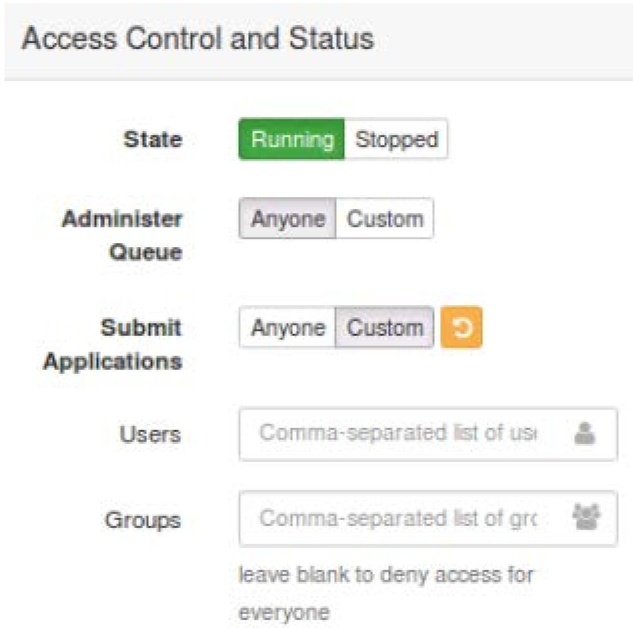
Click the **Custom** button, and in the **Users** field that appears, type the word `admin`.

Leave the **Groups** setting blank.



4 . Click the **Dev** queue under Engineering queue.

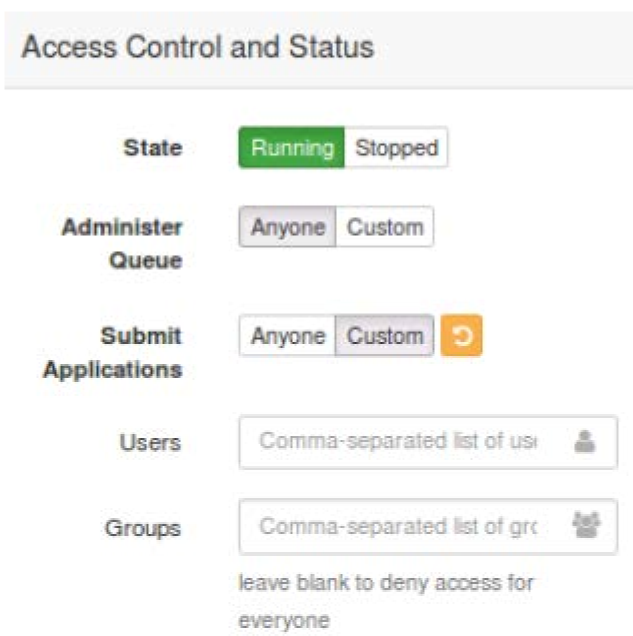
Under Access Control and Status, find the **Submit Applications** setting and click the **Custom** button.



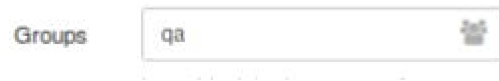
5 . Give members of the dev group access to this by typing `dev` in the **Groups** section.

6 . Click the **QA** queue under Engineering queue.

Under Access Control and Status, find the **Submit Applications** setting and click the **Custom** button.

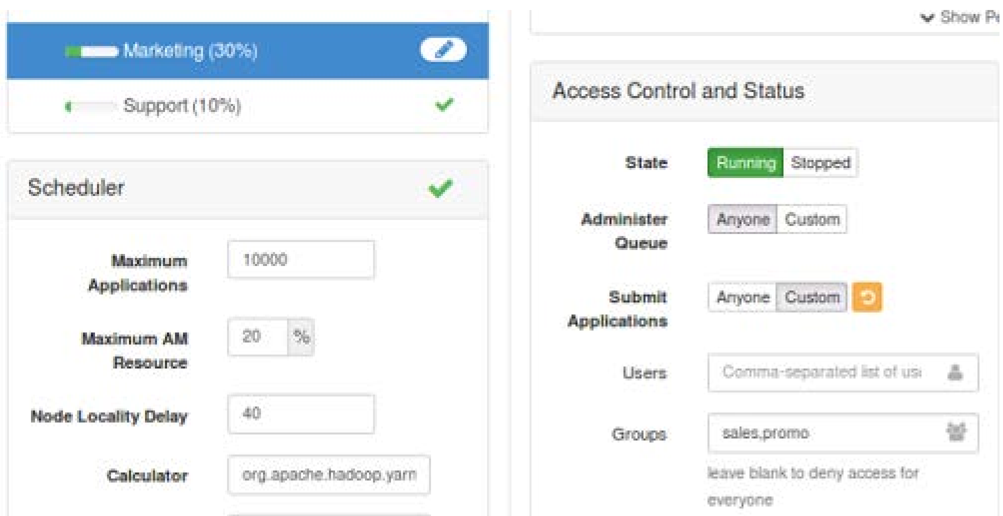


7 . Give members of the qa group access to this by typing `qa` in the **Groups** section.



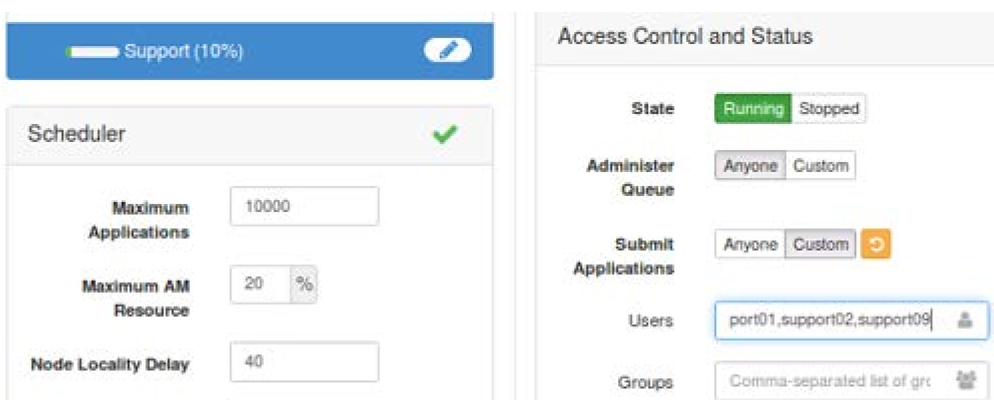
8 . Next click the **Marketing** queue.

Again, under Access Control and Status find **Submit Applications** and click on the **Custom** button. This time, add the sales and promo group to the list of allowed groups via a comma-separated list (no spaces).

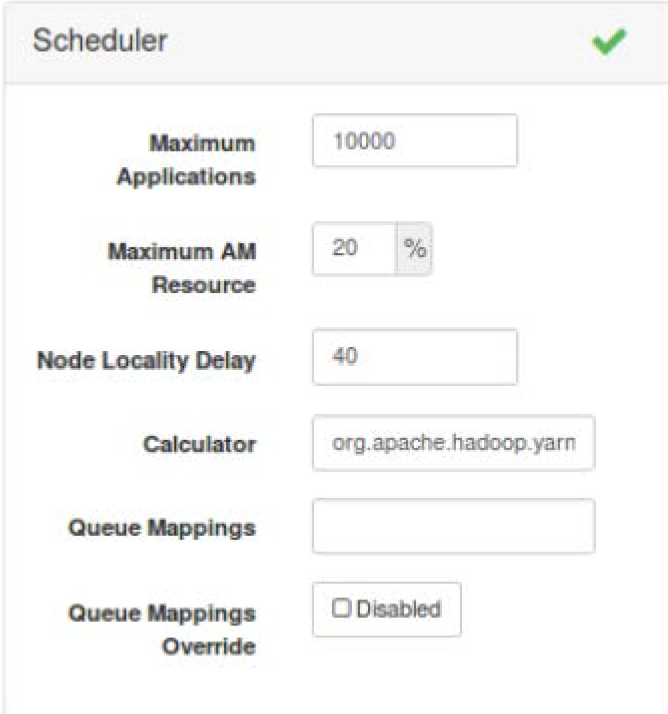


9 . Click the **Support** queue, and again find the **Submit Applications** setting.

Click the **Custom** button, but this time in the section type a comma-separated list of the three Support users we provisioned in a previous lab: `support01,support02,support09`.



10 . Now that ACLs have been set up to control queue access once the queues have been refreshed, locate the **Queue Mappings** setting under **Scheduler**.



11 . Type the following comma-delimited list (no spaces or returns) into the Queue Mappings settings box:

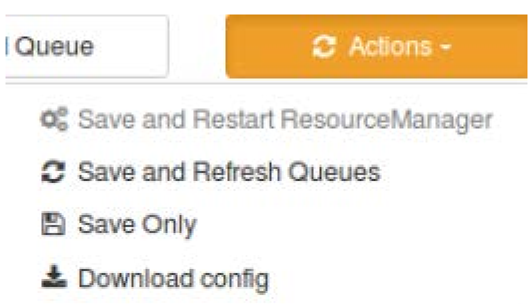
```
u:support01:Support,u:support02:Support,u:support09:Support,
g:promo:Marketing,g:sales:Marketing,g:dev:Dev,g:qa:QA
```

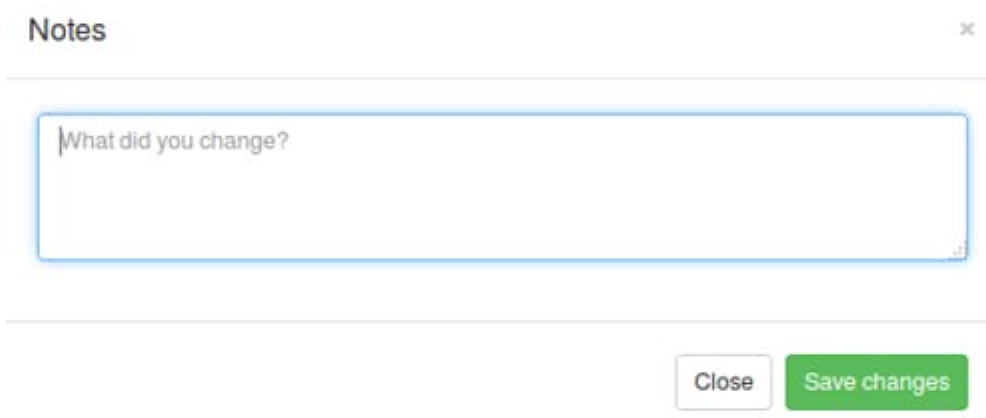
NOTE: The syntax for queue mapping is:

User mappings format - u:<username>:<queuename>
Group mappings format - g:<groupname>:<queuename>

12 . Click the orange **Actions** button and select **Save and Refresh Queues**.

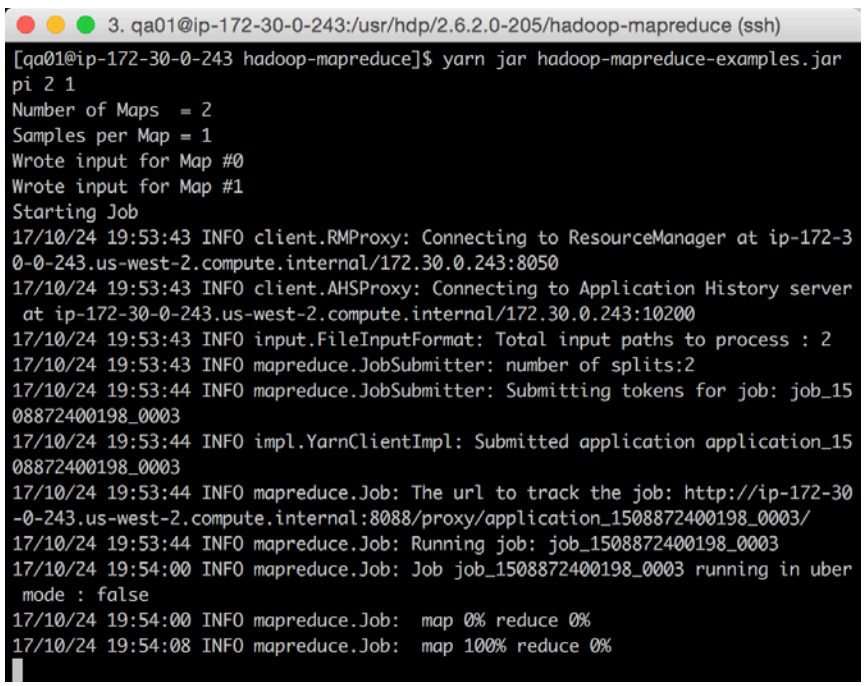
Click the **Save Changes** button in the resulting pop-up.





13 . Now to test your settings and see if they worked, go to one of your open terminal windows and type the *yarn jar* command you have been using in these labs for tests.

```
yarn jar hadoop-mapreduce-examples.jar pi 2 1
```



Repeat this process for at least a couple of users, then open a browser tab to `http://<AWS External Hostname ResourceManage>:8088`.

You should see in the list of applications that various jobs are being or have been performed by different users, and that those jobs have been assigned to the appropriate queues.

Show 20 entries

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus
application_1442990929866_0006	promo02	QuasiMonteCarlo	MAPREDUCE	Marketing	Wed Sep 23 10:09:25 -0400 2015	Wed Sep 23 10:10:46 -0400 2015	FINISHED	SUCCEEDED
application_1442990929866_0005	qa01	QuasiMonteCarlo	MAPREDUCE	QA	Wed Sep 23 04:07:38 -0400 2015	Wed Sep 23 04:08:45 -0400 2015	FINISHED	SUCCEEDED

14 . **OPTIONAL:**
If time allows, go back and change the ACL settings for the Dev queue and remove the dev group from the Submit Applications list of allowed users.

Save and refresh the queues, then go to the terminal window where dev01 is logged in and try to run the *yarn jar* job again.

What do you think will happen?

Testing User Limits

Test minimum user limits.

1 . Open two terminal windows and use SSH to login to any cluster node. In the first one switch users to qa01. In the second one switch users to qa02. Then in both terminal windows, change to the `/usr/hdp/<VERSION NUMBER>/hadoop-mapreduce` directory as before, and type but do not execute the test application *yarn jar* command.

First terminal window:

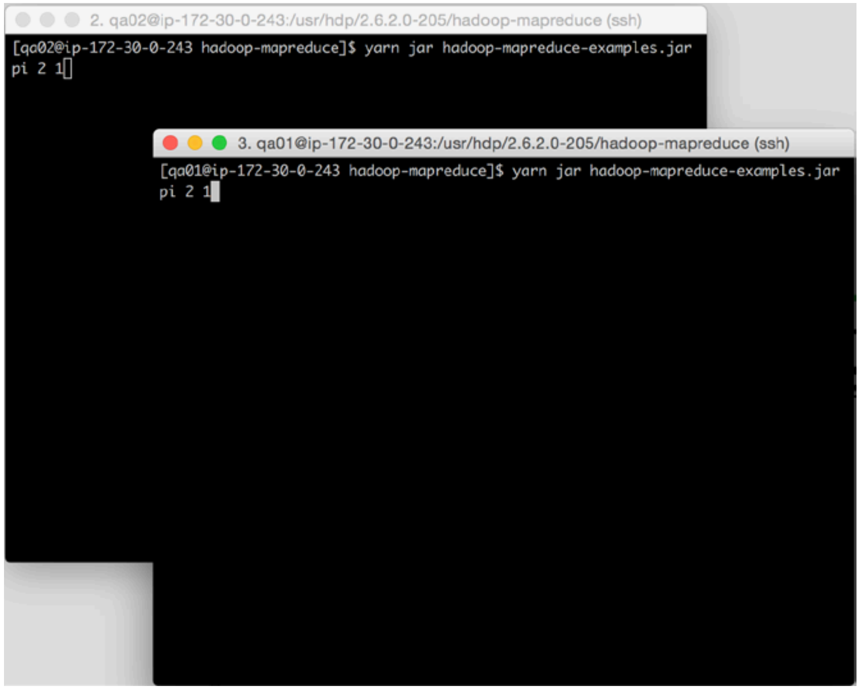
```
su - qa01
cd /usr/hdp/<VERSION NUMBER>/hadoop-mapreduce
yarn jar hadoop-mapreduce-examples.jar pi 2 1
```

(Do not run the yarn jar command yet.)

Second terminal window:

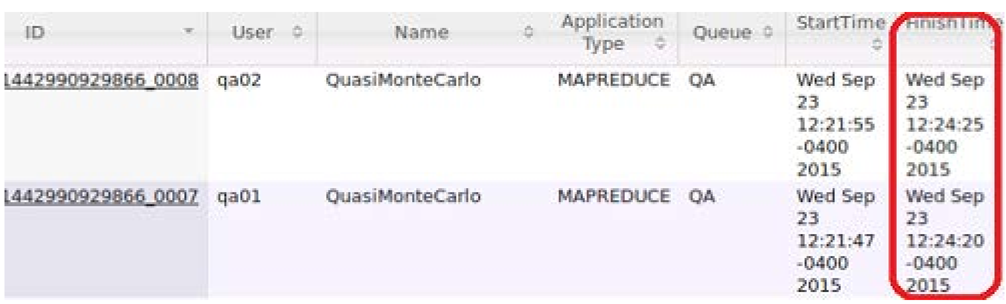
```
su - qa02
cd /usr/hdp/<VERSION NUMBER>/hadoop-mapreduce
yarn jar hadoop-mapreduce-examples.jar pi 5 1
```

(Do not run the yarn jar command yet.)



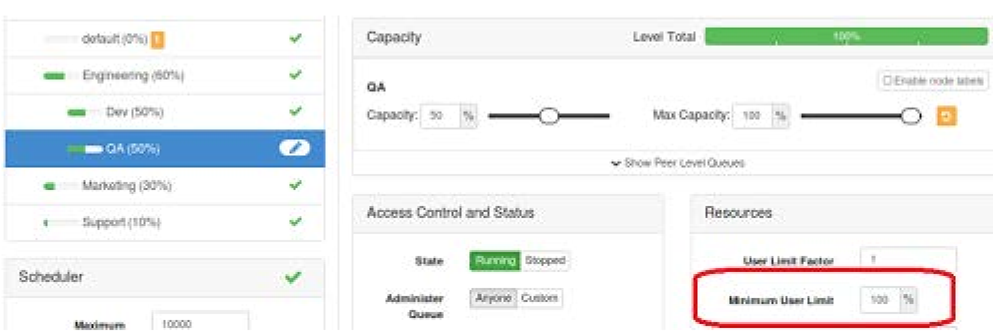
2 . Execute the *yarn jar* command for qa01 and then immediately execute the *yarn jar* command for qa02.

You should observe that both jobs execute at the same time. Then open a browser and point it to the ResourceManager UI at
`http://<AWS External Hostname ResourceManage>:8088.`
Confirm that both jobs ended within just a few seconds of each other.



3 . Go back to the YARN Queue Manager View in the Ambari UI and click the **Engineering.QA** queue.

NOTE: That the default Minimum User Limit setting is `100%`.



A minimum user limit of `100%` should guarantee a single user `100%` of the queue Capacity setting.

Why were two users able to run jobs in this queue at the same time?

4. Disable elasticity settings for the queue.

   To do this, change the **Max Capacity** value from `100%` so that it matches the **Capacity** value of `50%`.

5. Click the **Actions** button and save and refresh the queues, make notes, and save changes.

   Once the operation completes, go back to the two terminal windows, press the **Up Arrow** in each to find the previous *yarn jar* command, and once again run the jobs one right after the other.

   Did the jobs still run at basically the same time?

   Can you figure out why this might have happened?

6. For minimum user limits to be strictly enforced, no elasticity can exist in queue settings. This includes any parent queues.

   Because the Engineering parent queue had elasticity enabled, the second job was allowed to run in that available capacity.

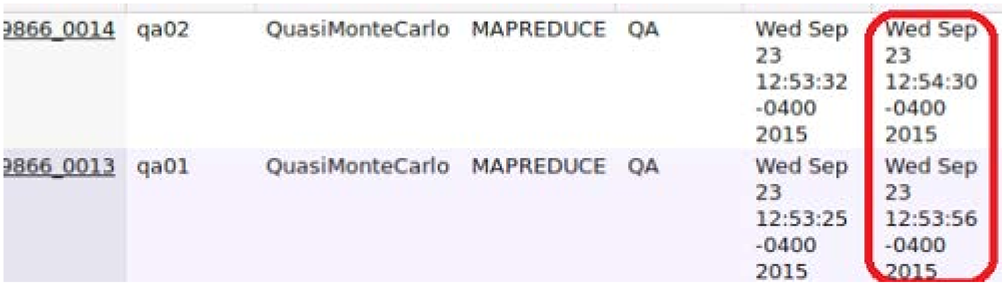
   Disable elasticity settings for the Engineering queue. To do this, in the YARN Queue Manager select the Engineering queue and change the **Max Capacity** value from `100%` to match the **Capacity** value of `60%`.

7. Click the **Actions** button and save and refresh the queues, make notes, and save changes.

   Once the operation completes, go back to the two terminal windows, press the **Up Arrow** in each to find the previous *yarn jar* command, and once again run the jobs one right after the other.

   This time, one job will wait for the other job to complete before it starts running.

   The terminal window should report the second job taking approximately twice as long as the first one, and the ResourceManager UI should show the two jobs finishing farther apart than in our previous runs.



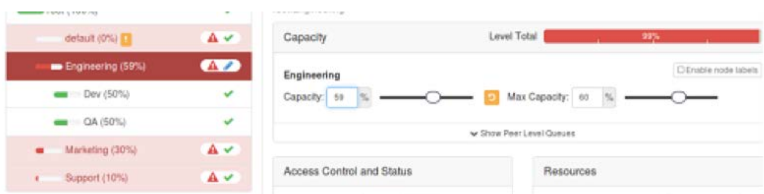
9866_0014	qa02	QuasiMonteCarlo	MAPREDUCE	QA	Wed Sep 23 12:53:32 -0400 2015	Wed Sep 23 12:54:30 -0400 2015
9866_0013	qa01	QuasiMonteCarlo	MAPREDUCE	QA	Wed Sep 23 12:53:25 -0400 2015	Wed Sep 23 12:53:56 -0400 2015

Re-establishing Default Queue Access

Re-establish access to the default queue.

1. Some actions (setting up high availability for the ResourceManager for example) require the ability to run a YARN job in the default queue as part of their self-test. These will report as having failed if this job cannot run properly.

   To enable this, return to the YARN Queue Manager and set the capacity of the Engineering queue to 59%.

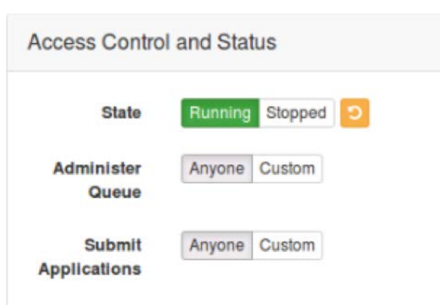
   

2. Then set the default queue **Capacity** setting to `1%` and the **Max Capacity** to `100%`.

   NOTE: In the lab, make sure the **Max Capacity** setting allows for use of all cluster resources. The default will be for the **Max Capacity** value to match the **Capacity** value, which in this case would not be enough resources to allow an application to run.

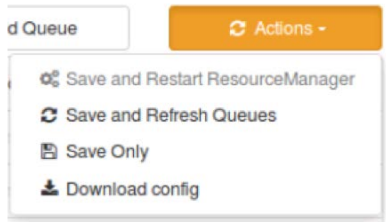
   

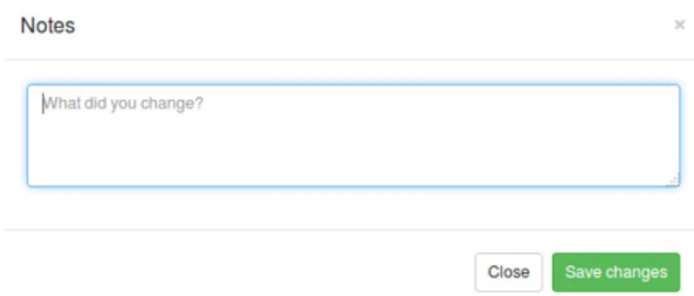
3. With the default queue still selected, change the State to **Running** and make sure that Submit Applications is set to **Anyone**.

4 . Click the orange **Actions** button and select **Save and Refresh Queues**.

Click the **Save Changes** button in the resulting pop-up.





Result

You have now:
Configured YARN ACLs and default queue mapping; tested minimum user limits; and reestablished access to the default queue.

Lab 21: Yarn Node Labels

About This Lab

Objective:	Configure and Test YARN Node Labels
File locations:	
Successful outcome:	Create and test node labels
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Node Labels***

Lab Steps

Perform the following steps:

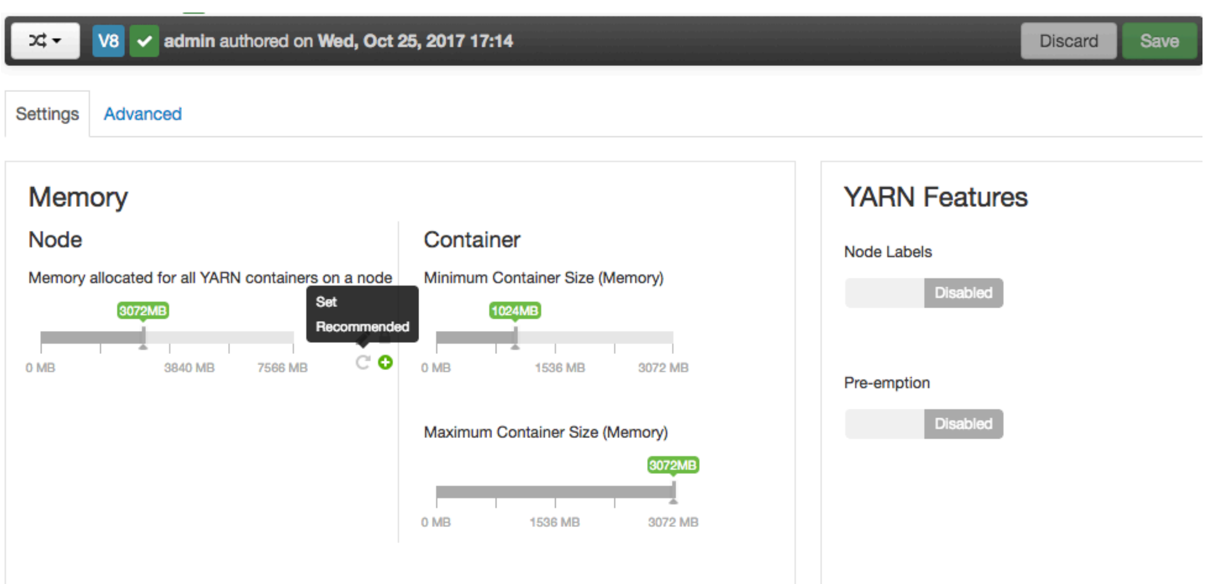
1) **Configure YARN settings and Node Labels using Ambari Web UI**

a) Open the browser and connect to the Ambari Server at the URL

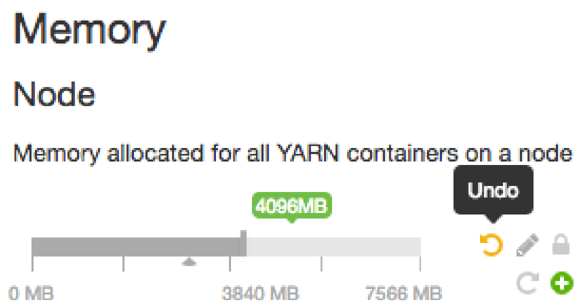
http://<Ambari Server Hostname>:8080

b) Login to the Ambari Web UI using the user name `admin` and the password `BadPass#1`.

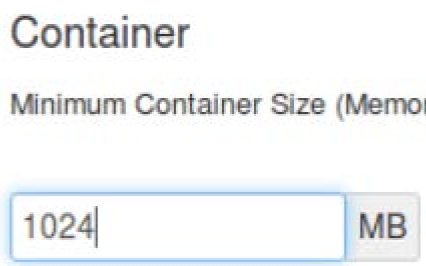
c) Toggle to **Services > Yarn > Configs**. Under **Settings** make sure the node memory is set to **Ambari recommendations** by hovering to the right of the setting bar and clicking on the gray clockwise arrow that appears.



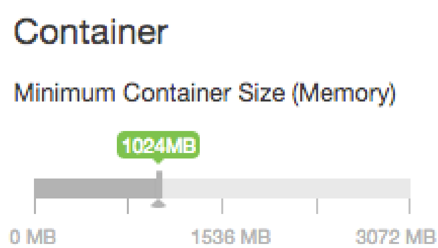
If the value changed, you should see a yellow counter-clockwise undo arrow appear.



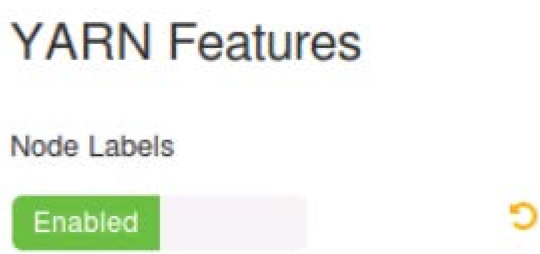
d) On that same screen, make sure the **Minimum Container Size (Memory)** setting is set to 1024 MB. This prevents small applications from taking up more resources than necessary, but still allows them to request containers up to the **Maximum Container Size** setting.



When finished, it should look like this:

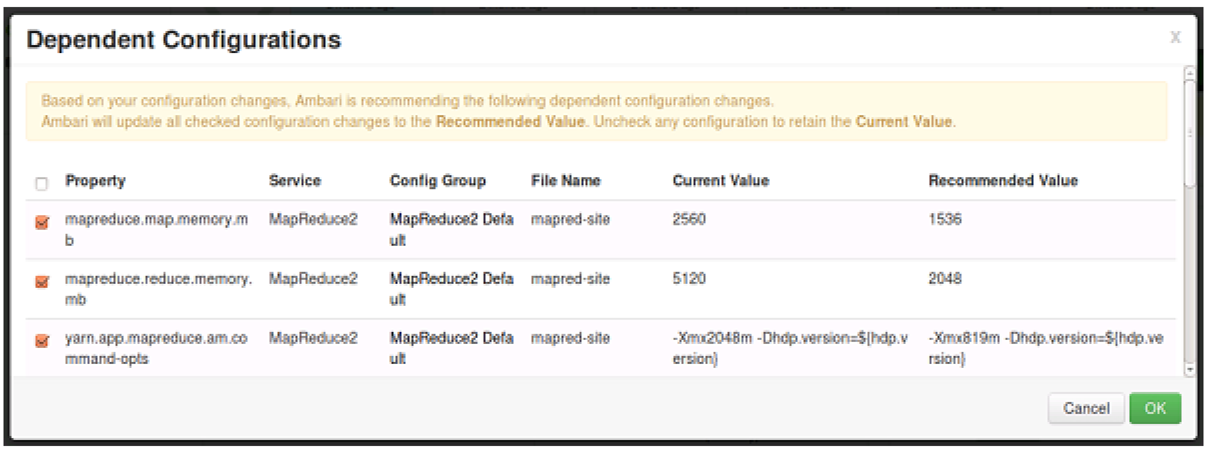


e) Finally on this screen, under **YARN Features**, toggle the **Node Labels** setting from **Disabled** to **Enabled**.

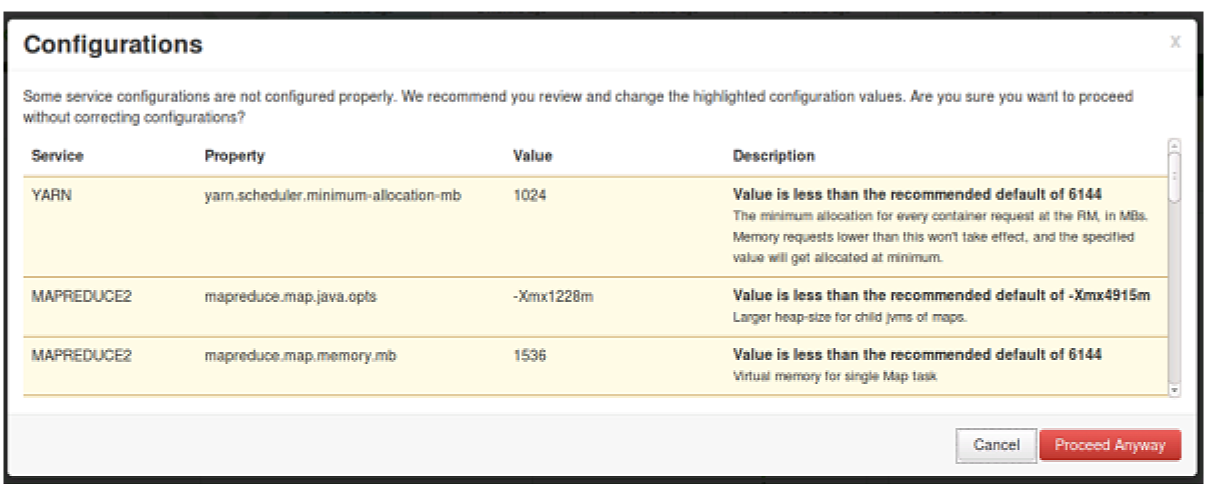


f) Verify all of your settings are correct, then click **Save**.
Click **Save** again on the Save Configuration pop-up window to confirm your changes.

Then click **OK** on the **Dependent Configurations** pop-up window, if one appears.



Click **Proceed Anyway** on the **Configurations** pop-up window, and then click **OK** on the final confirmation pop-up window.



g) Restart all services that need it as indicated by the Ambari UI.

2) **Create a Node Label and assign it to a node.**

a) Use SSH to login to any cluster node.
b) Switch to the built-in yarn administrative user account.

```
su - yarn
```

c) Create a node label named GPU.

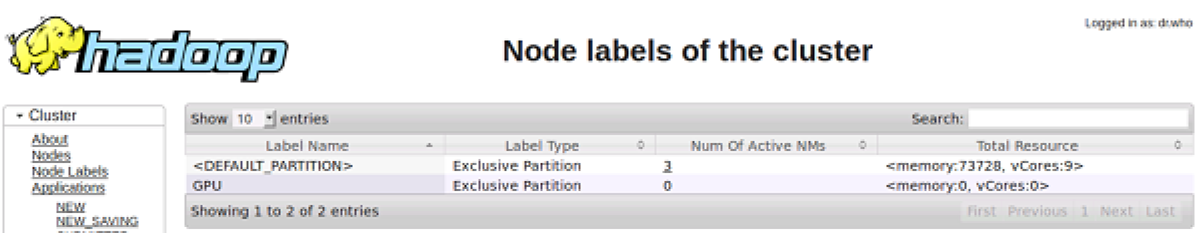
```
yarn rmadmin -addToClusterNodeLabels GPU
```

d) Verify that the node label was created successfully.

```
yarn cluster --list-node-labels
```

```
[yarn@ip-172-30-0-211 ~]$ yarn rmadmin -addToClusterNodeLabels GPU
17/08/14 17:14:24 INFO client.RMProxy: Connecting to ResourceManager at ip-172-30-0-211.us-west-2.compute
.internal/172.30.0.211:8141
[yarn@ip-172-30-0-211 ~]$ yarn cluster --list-node-labels
17/08/14 17:14:40 INFO client.RMProxy: Connecting to ResourceManager at ip-172-30-0-211.us-west-2.compute
.internal/172.30.0.211:8050
17/08/14 17:14:41 INFO client.AHSProxy: Connecting to Application History server at ip-172-30-0-211.us-we
st-2.compute.internal/172.30.0.211:10200
Node Labels: <GPU:exclusivity=true>
[yarn@ip-172-30-0-211 ~]$
```

This can also be verified by going to the Resource Manager UI `http://<AWS External Hostname Resource Manager>:8088` in a web browser and clicking on the **Node Labels** link:



e) In the terminal window logged in as the yarn administrative user, enter the command to assign the GPU node label to a selected node.

```
yarn rmadmin -replaceLabelsOnNode ip-172-30-0-211.us-west-
2.computer.internal=GPU
```

```
[yarn@ip-172-30-0-211 ~]$ yarn rmadmin -replaceLabelsOnNode ip-172-30-0-211.us-west-2.compute.internal=GPU
17/08/14 17:19:42 INFO client.RMProxy: Connecting to ResourceManager at ip-172-30-0-211.us-west-2.compute.internal/172.30
.0.211:8141
[yarn@ip-172-30-0-211 ~]$
```

Verify this was successful by going to the Resource Manager UI and clicking on the Nodes link:

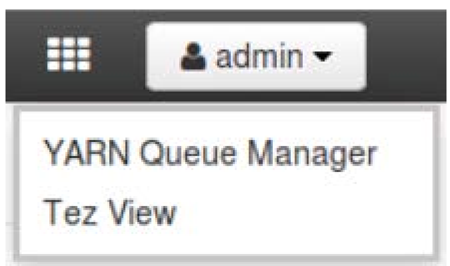
Show 20 entries

Node Labels	Rack	Node State	Node Address	Node HTTP Address
	/default-rack	RUNNING	ip-172-30-0-184.us-west-2.compute.internal:45454	ip-172-30-0-184.us-west-2.compute.internal:8042
	/default-rack	RUNNING	ip-172-30-0-243.us-west-2.compute.internal:45454	ip-172-30-0-243.us-west-2.compute.internal:8042
GPU	/default-rack	RUNNING	ip-172-30-0-211.us-west-2.compute.internal:45454	ip-172-30-0-211.us-west-2.compute.internal:8042

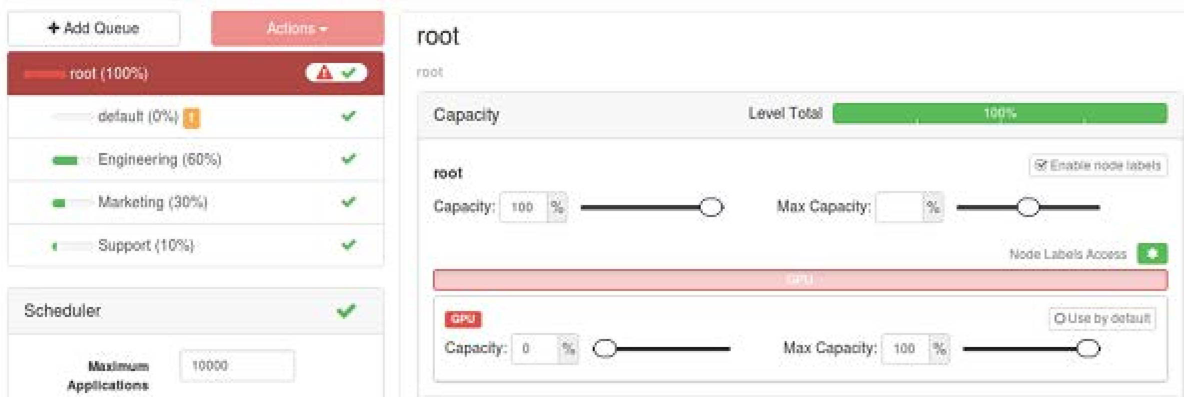
Showing 1 to 3 of 3 entries

3) **Configure the Marketing queue to use labeled nodes.**

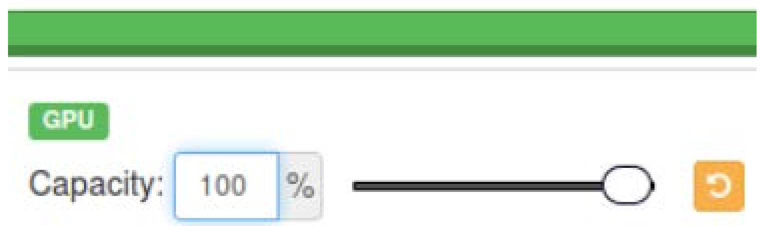
a) In the Ambari UI, click on the Ambari Views drop-down menu and select **YARN Queue Manager**.



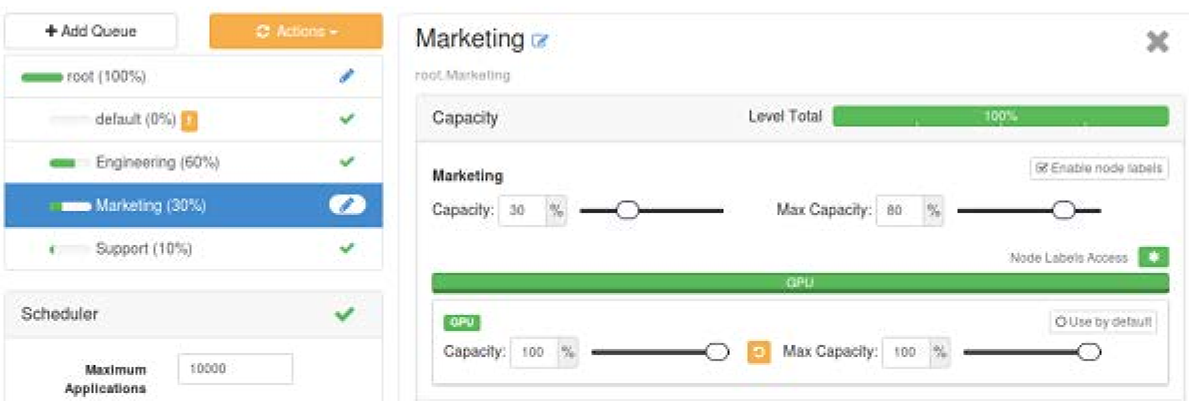
b) Select the root queue and click the **Enable node labels** button on the far right. This may already be selected once YARN detects that node labels have been configured. If not already selected, click the **asterisk** button to the right of **Node Labels Access**, which will make it turn green.



c) Set the capacity for GPU labeled nodes to 100% for the cluster.

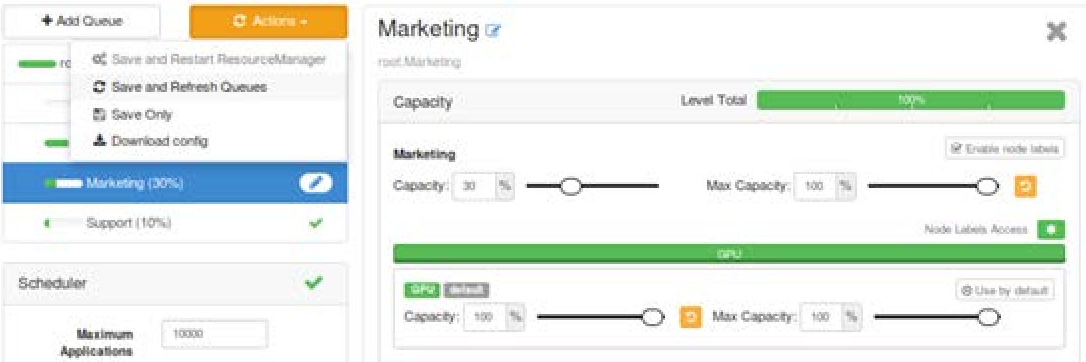


d) Click on the Marketing queue. Just like you did for the root queue, click the **Enable node labels** button, the **asterisk** button to the right of **Node Labels Access**, and set the capacity of the **GPU labeled nodes available** to this queue to 100%.



e) To test whether default mappings work as expected, also click the button labeled **Use by default** for the Marketing queue.

f) **Click on the orange** Actions **button and select** Save and Refresh Queues**. Click** Save Changes **on the resulting pop-up window.**



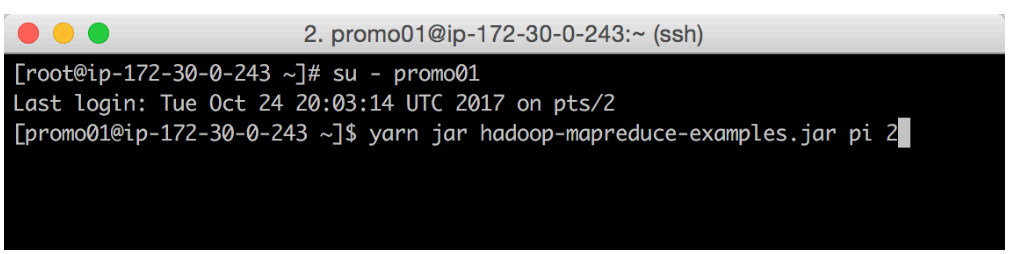
4) **Test interaction between a MapReduce application and labeled nodes.**

a) Use SSH to login to any cluster node, then switch to the promo01 user.

```
sudo su -
su - promo01
```

b) While logged in as promo01, run a sample MapReduce command which approximates the value of pi.

```
cd /usr/hdp/current/hadoop-mapreduce-client
yarn jar hadoop-mapreduce-examples.jar pi 5 10
```



```
2. promo01@ip-172-30-0-243:/usr/hdp/2.6.2.0-205/hadoop-mapreduce (ssh)
                Reduce input records=4
                Reduce output records=0
                Spilled Records=8
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=257
                CPU time spent (ms)=2350
                Physical memory (bytes) snapshot=2403250176
                Virtual memory (bytes) snapshot=11125215232
                Total committed heap usage (bytes)=2310012928
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=236
        File Output Format Counters
                Bytes Written=97
Job Finished in 22.113 seconds
Estimated value of Pi is 4.00000000000000000000
[promo01@ip-172-30-0-243 hadoop-mapreduce]$
```

c) Once the job completes, go back to the Resource Manager UI and click on the **Applications** link. Verify that the job ran, and that it was assigned to the Marketing queue per default queue mapping settings.

Scheduler Type	Scheduling Resource Type	Minimum Allocation
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>

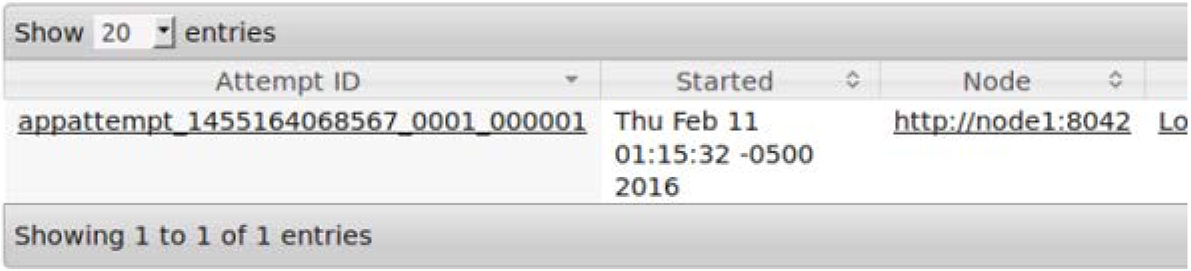
ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus
application_1455164068567_0001	testmar	QuasiMonteCarlo	MAPREDUCE	Marketing	Thu Feb 11	Thu Feb 11	FINISHED	SUCCEEDED

d) Click on the application ID and scroll down to the last section where the Attempt ID is shown. This section also indicates the node on which the job was executed.

e) Did the default queue mapping and the default node label setting result in the MapReduce job being executed on the GPU node?

No, it did not.

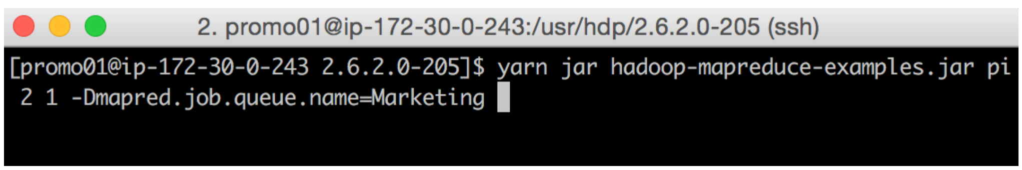
This might be considered a bug, since it is reasonable to expect that the **default queue mapping** and the **default node label** setting for that queue *should* have resulted in the MapReduce job running on the GPU-labeled node.



Even if you opened up three terminal windows and ran three separate instances of the MapReduce application, under no circumstance would any of them be assigned to GPU node.

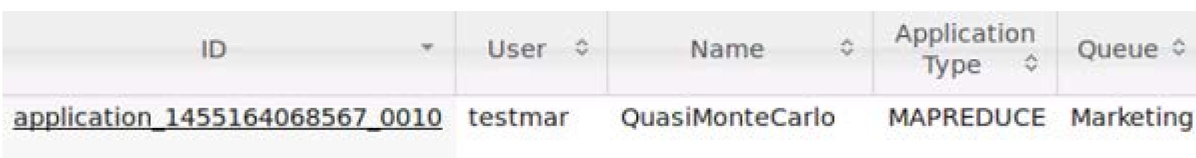
f) MapReduce jobs allow the specification of a queue using the `-Dmapred.job.queue.name=<queue>` tag.
Run the same MapReduce job as before, but this time explicitly call out the use of the Marketing queue in the command.

```
cd /usr/hdp/current/hadoop-mapreduce-client
yarn jar hadoop-mapreduce-examples.jar pi -Dmapred.job.queue.name=Marketing 5
10
```



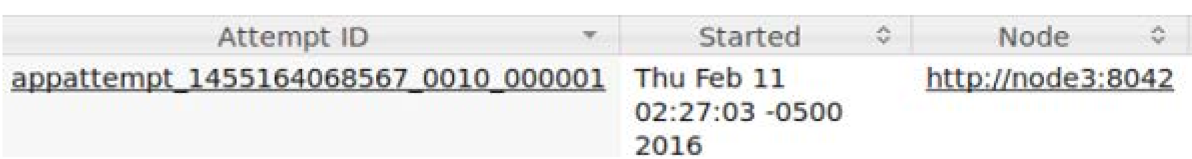
What do you think will happen?

g) Once the job completes, go back to the Resource Manager UI and click on the **Applications** link. Verify that the job ran, and that it was assigned to the Marketing queue just like the previous job.



h) Click on the application ID and scroll down to the last section where the Attempt ID is shown.
Did explicitly specifying the queue result in the MapReduce job being executed on the GPU node? Yes it did!

Default queue mapping does not result in the default node label being applied.
But specifying the queue upon execution did result in the job being assigned to our GPU-labeled node.

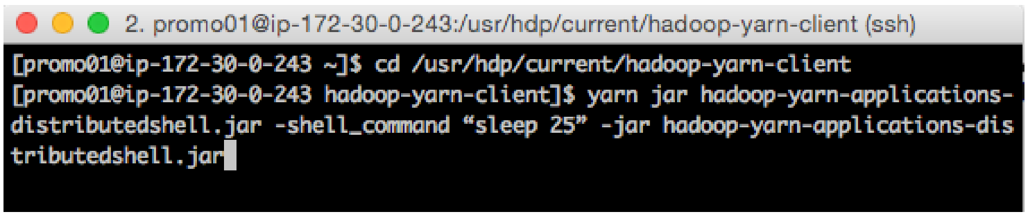


Important
MapReduce jobs do not have an option to specify a specific node label, so the only way to get a MapReduce job to run on a labeled node is to, upon execution, specify a queue that has a default node label assigned.

5) **Test interaction between a YARN application and labeled nodes.**

a) Still logged in as the promo01 user, run a sample YARN distributed shell job. (Don't forget that you can use the **Tab key / autocomplete** every so often when typing out directory paths to assist in typing out this long command.)

```
cd /usr/hdp/current/hadoop-yarn-client
yarn jar hadoop-yarn-applications-distributedshell.jar -shell_command “sleep
25” -jar hadoop-yarn-applications-distributedshell.jar
```



b) When the job completes, go back to the Resource Manager UI and refresh the page. Then click on the **Applications** link. Confirm that a YARN job was executed and assigned to the Marketing queue.

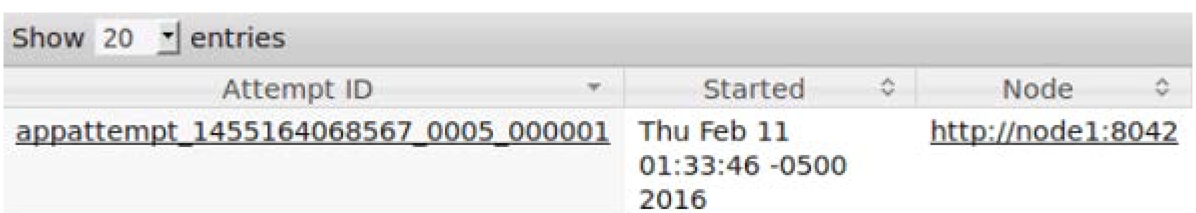
Show 20 entries

ID	User	Name	Application Type	Queue
application_1455164068567_0005	testmar	DistributedShell	YARN	Marketing

c) Click on the application ID and scroll down to check which node the job was assigned to.

Did the default queue mapping and the default node label setting result in the YARN job being executed on the GPU node?

Just as we saw when running the MapReduce job, the answer is "No".



Show 20 entries

Attempt ID	Started	Node
appattempt_1455164068567_0005_000001	Thu Feb 11 01:33:46 -0500 2016	http://node1:8042

d) Run the same YARN job as before, but this time specify in the command that the job shouldbe run in the Marketing queue by appending it with `-queue Marketing`. (You can press the up arrow to bring up the previous command and then edit it to speed up this step.)

```
cd /usr/hdp/current/hadoop-yarn-client
```

```
yarn jar hadoop-yarn-applications-distributedshell.jar -shell_command “sleep 25” -jar hadoop-yarn-applications-distributedshell.jar -queue Marketing
```

```
2. promo01@ip-172-30-0-243:/usr/hdp/current/hadoop-yarn-client (ssh)
[promo01@ip-172-30-0-243 ~]$ cd /usr/hdp/current/hadoop-yarn-client
[promo01@ip-172-30-0-243 hadoop-yarn-client]$ yarn jar hadoop-yarn-applications-distributedshell.jar -shell_command “sleep 25” -jar hadoop-yarn-applications-distributedshell.jar –queue Marketing
```

Given the results of the last test, what do you suspect will happen this time?
Will the YARN job be assigned to the labeled node?

e) Once the job is running or complete, go back to the Resource Manager UI and refresh the Applications page. Verify that another YARN job was executed and assigned to the Marketing queue.

ID	User	Name	Application Type	Queue
application_1455164068567_0006	testmar	DistributedShell	YARN	Marketing

f) Click on the application ID and scroll down to check which node the job was assigned to.

Did specifying the queue and the default node label setting result in the YARN job being executed on the GPU node? Yes it did!

The behavior for YARN jobs is the same as for MapReduce jobs. Default queue mapping does not get default node label assignment, but specifically calling out the queue to use does.

Show 20 entries

Attempt ID	Started	Node
appattempt_1455164068567_0006_000001	Thu Feb 11 01:48:43 -0500 2016	http://node3:8042

g) In addition to specifically calling out the queue, with YARN jobs we can also call out the use of a specific node label by adding the `-node_label_expression` tag and specifying which node label to use.

Test this functionality by re-running the YARN job, removing the -queue setting, and specifying the GPU node label as the promo01 user.

Remember that jobs executed by the promo01 user, by default, will be assigned to the Marketing queue.

```
cd /usr/hdp/current/hadoop-yarn-client
```

```
yarn jar hadoop-yarn-applications-distributedshell.jar -shell_command “sleep
25” -jar hadoop-yarn-applications-distributedshell.jar -node_label_expression
GPU
```

h) Once the job is running or complete, go back to the Resource Manager UI and refresh the **Applications** page.

   Verify that another YARN job was executed and assigned to the Marketing queue.

ID	User	Name	Application Type	Queue
application_1455164068567_0007	testmar	DistributedShell	YARN	Marketing

i) Click on the application ID and scroll down to check which node the job was assigned to.

   Did the default queue mapping result and specifying the node label result in the YARN job being executed on the GPU node? Yes, It did.

Attempt ID	Started	Node
appattempt_1455164068567_0007_000001	Thu Feb 11 01:56:31 -0500 2016	http://node3:8042

NOTE: Since it is possible to assign more than one node label to a queue, it may sometimes be necessary to specify both the desired queue ***and*** the desired node label to assign the job to upon execution to ensure the job gets assigned to an appropriate node.

This is only possible with YARN jobs, since MapReduce jobs lack a mechanism to specify a node label.

j) Next we will test what happens when a user who is not default mapped to the Marketing queue attempts to run a YARN job and assign it to the GPU node label.

   Remember that the Marketing queue was granted 100% of the resources of the GPU-labeled nodes.

To test this, exit the promo01 user, which should take you back to acting as the root user.

Then run the command from the previous test (unfortunately, the up-arrow will not work, so you will need to type it all out this time - although you can use copy and paste) and see whether the YARN job runs on a labeled node or not.

```
exit

cd /usr/hdp/current/hadoop-yarn-client
yarn jar hadoop-yarn-applications-distributedshell.jar -shell_command “sleep
25” -jar hadoop-yarn-applications-distributedshell.jar
-node_label_expression GPU
```

What do you expect will happen?

The command fails.

This is in part because the root user is not default mapped to the Marketing queue, and the default queue is not configured to allow applications to run (0% capacity). Thus, specifying the node label was not enough on its own to force the job to run in the Marketing queue.

```
16/02/11 02:10:13 INFO distributedshell.Client: Submitting application to ASM
16/02/11 02:10:14 FATAL distributedshell.Client: Error running Client
org.apache.hadoop.yarn.exceptions.YarnException: Failed to submit application_14
55164068567_0008 to YARN : org.apache.hadoop.security.AccessControlException: Qu
eue root.default already has 0 applications, cannot accept submission of applica
tion: application_1455164068567_0008
```

The job would fail with the same error under our current queue configuration if no label was specified as well.

k) Next we will test what happens if the root user specifies the job to run in the Marketing queue but does not specify the node label. Do this by replicating the previous command, but replacing `-node_label_expression GPU` with `-queue Marketing`.

```
cd /usr/hdp/current/hadoop-yarn-client
yarn jar hadoop-yarn-applications-distributedshell.jar -shell_command “sleep
25” -jar hadoop-yarn-applications-distributedshell.jar -queue Marketing
```

```
2. root@ip-172-30-0-243:/usr/hdp/current/hadoop-yarn-client (ssh)
[root@ip-172-30-0-243 ~]# cd /usr/hdp/current/hadoop-yarn-client/
[root@ip-172-30-0-243 hadoop-yarn-client]# yarn jar hadoop-yarn-applications-dis
tributedshell.jar -shell_command “sleep 25” -jar hadoop-yarn-applications-distri
butedshell.jar –queue Marketing
```

Assuming ACLs allow the root user to run a job in this queue, what do you expect to happen?

l) Once the job is running or complete, go back to the Resource Manager UI and refresh the **Applications** page. Verify that another YARN job was executed and assigned to the Marketing queue.

ID	User	Name	Application Type	Queue
application_1455164068567_0009	root	DistributedShell	YARN	Marketing

m) Click on the application ID and scroll down to check which node the job was assigned to. Did the default queue mapping result and the default node label setting result in the YARN job being executed on the GPU node? Yes it did, as we would expect.

The default node label setting is followed because the queue was specified during the command, just like before.

Show 20 entries

Attempt ID	Started	Node
appattempt_1455164068567_0009_000001	Thu Feb 11 02:19:45 -0500 2016	http://node3:8042

n) Next we will test what happens when a user who is default mapped to a queue tries to specify the use of a node label only accessible to a different queue.

Switch from root to the dev01 user - the one that has a default queue mapping to the Engineering queue - and run the YARN job specifying only the node label.

```
su - dev01

cd /usr/hdp/current/hadoop-yarn-client
yarn jar hadoop-yarn-applications-distributedshell.jar
-shell_command “sleep 25” -jar hadoop-yarn-applications-distributedshell.jar
-node_label_expression GPU
```

What do you expect will happen?

The job launches and is assigned to the Engineering queue per the default queue mapping, but it does not execute. Because there are no resources in the Engineering queue that have the GPU node label, the job will simply wait until such resources are available.

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus
455164068567_0011	testeng	DistributedShell	YARN	Engineering	Thu Feb 11 02:50:08	N/A	ACCEPTED	UNDEFINED

Press Ctrl+c to exit monitoring of the YARN job, then exit back to the root user.

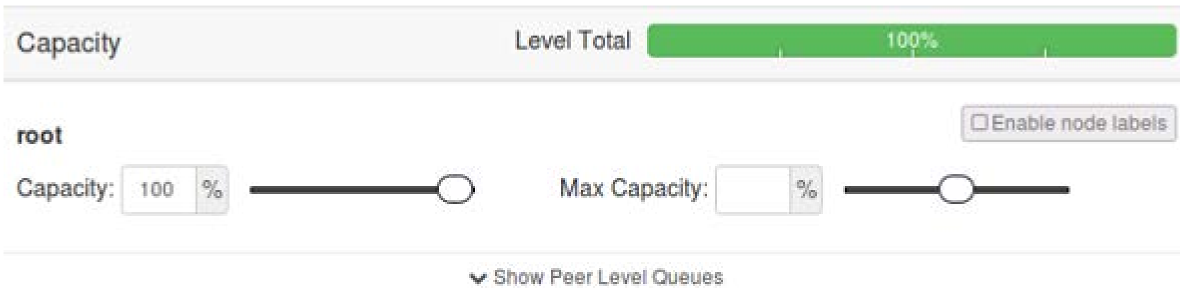
```
exit
```

6) **Remove node label settings from all HDP components.**

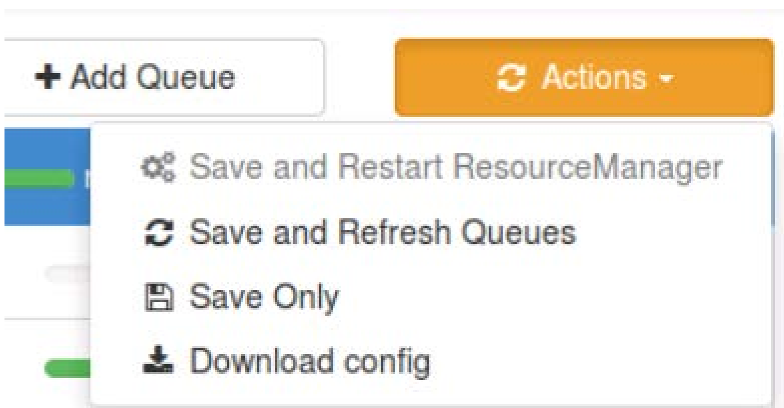
a. In the YARN queue manager, select the Marketing queue, and perform the following configurations in the following order:

- De-select the **Use by default** button
- De-select the **Node Labels Access asterisk** button
- De-select the **Enable node labels** button

b. Select the root queue and perform the following configurations in the following order:
- De-select the **Node Labels Access** asterisk button
- De-select the **Enable node labels** button



c. Click the orange **Actions** button and select **Save and Refresh Queues**. Then click **Save Changes**.



d. In the terminal window that is logged in as the yarn administrative user (switch to that user from root if necessary) remove the node label from node selected earlier.

```
yarn rmadmin -replaceLabelsOnNode ip-172-30-0-211.us-west-2.compute.internal
```

e. Remove the GPU node label from the cluster.

```
yarn rmadmin -removeFromClusterNodeLabels GPU
```

f. Verify that the node label has been removed.

```
yarn cluster --list-node-labels
```

g. Exit back to the root user.

```
exit
```

Result

You have successfully created a node label, assigned it to an HDP node, configured it to be used by default by a YARN queue, and performed various tests to examine node label behavior under different circumstances. You have then backed out of all node label settings on all cluster components.

Lab 22: Configuring NameNode HA

About This Lab

Objective:	To use the Ambari Web UI to configure a cluster with HDFS NameNode HA
File locations:	N/A
Successful outcome:	**You will:** Configure an Active and a Standby NameNode running in your cluster, three ZooKeeper Servers to support NameNode HA, a ZooKeeper FailoverController on each NameNode system, and three running JournalNodes. You will also remove the existing Secondary NameNode and test automatic failover.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Configuring HDFS and YARN High Availability (HA)***

Viewing Configurations

View the NameNode and cluster configuration prior to configuring NameNode HA.

1. Use SSH to login to your node.

2. Search the `/etc/hadoop/conf/core-site.xml` file for the `fs.defaultFS` property. Notice the value listed in this property.

```
more /etc/hadoop/conf/core-site.xml
```

   The value of `fs.defaultFS` should be similar to `hdfs://ip-172-30-0-126.us-west-2.compute.internal:8020`.

3. View the contents of the `/etc/hadoop/conf/hdfs-site.xml` file.

```
more /etc/hadoop/conf/hdfs-site.xml
```

Do any of the following properties exist in the file?

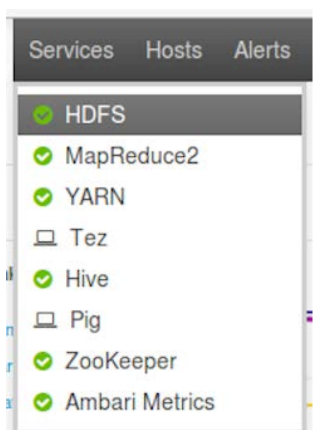
```
dfs.client.failover.proxy.provider.mycluster
dfs.ha.namenode.mycluster
dfs.namenode.http-address.mycluster.nn1
dfs.namenode.http-address.mycluster.nn2
dfs.namenode.rpc-address.mycluster.nn1
dfs.namenode.rpc-address.mycluster.nn2
dfs.nameservices
```

None of these properties should exist. (This is not a complete list of NameNode HA properties.)

4. Open the browser and connect to the Ambari Server at the URL

http://<Ambari Server Hostname>:8080

5. Login to the Ambari Web UI using the user name `admin` and the password `BadPass#1`.
6. Click **Services** and select **HDFS**.



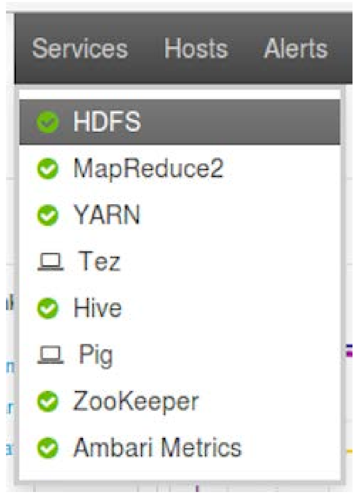
On the **Summary** page do you see a **NameNode** and a Secondary NameNode (**SNameNode**)?
You should see both.

Do you see any ZooKeeper FailoverControllers or JournalNodes?
You should not.

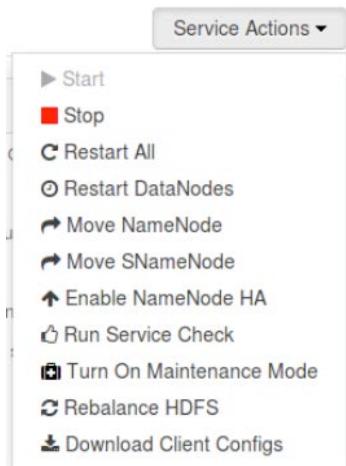
Enabling NameNode HA.

Enable NameNode HA.

1. In the Ambari Web UI, click **Services** and select **HDFS**.



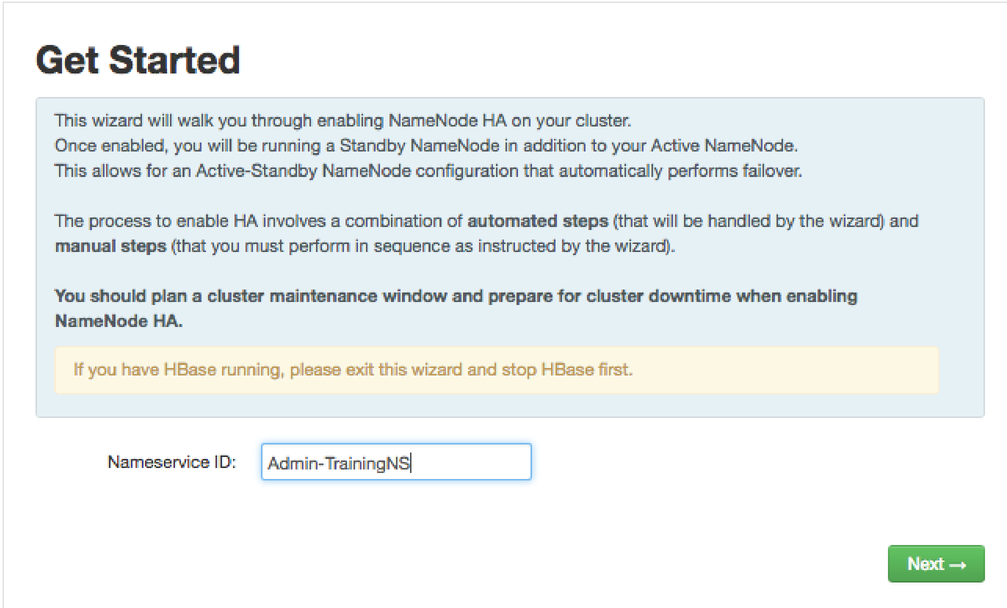
2 . Click the **Service Actions** menu button and select **Enable NameNode HA**.



The Enable NameNode HA Wizard window opens.

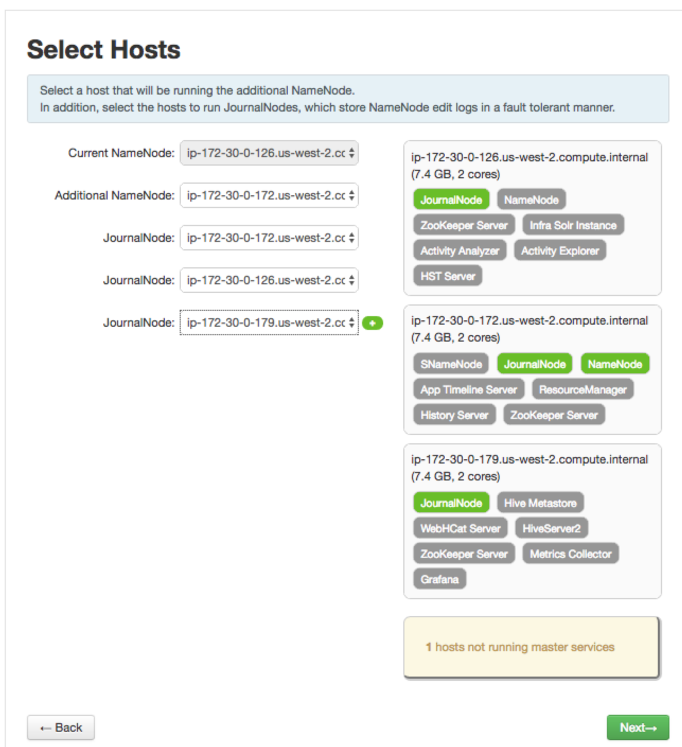
3 . Enter a Nameservice ID and then click **Next**. Example below is Admin-TrainingNS

There is no HBase running on the cluster so ignore the message in the wizard.



4 . Select a host for the **Additional NameNode** a good choice is to use the same node that has "SNameNode" Secondary NameNode listed, and confirm that a **JournalNode** configured on each of the three cluster nodes.

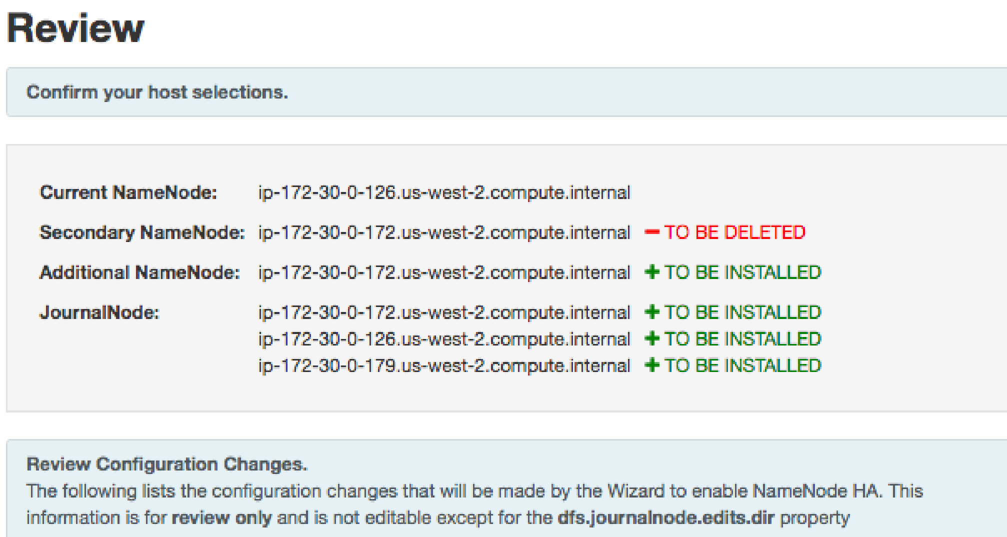
Then click **Next**.



5 . Read the information in the Review window.

Notice that automatic failover is enabled by default.

No changes are necessary so click **Next** (not shown) to continue.



6 . The wizard lists manual configuration steps that are required next. These must be completed or the **Next** button in the wizard will not become active.

In the terminal window logged in to the currently active NameNode system, type the commands shown in the wizard.

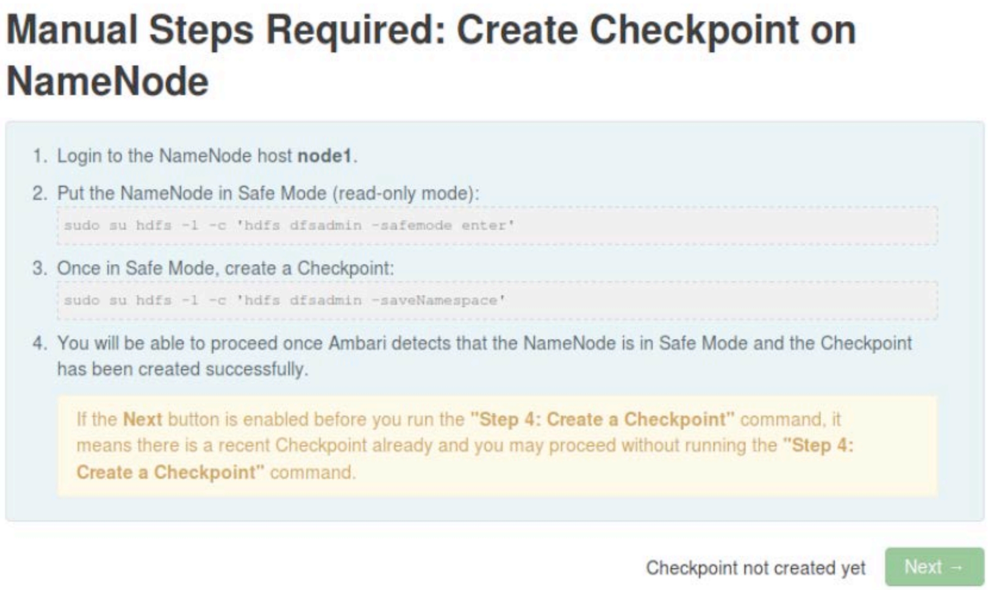
```
sudo su hdfs -l -c ‘hdfs dfsadmin -safemode enter’
```

The command output should report that safemode is ON.

```
sudo su hdfs -l -c ‘hdfs dfsadmin -saveNamespace’
```

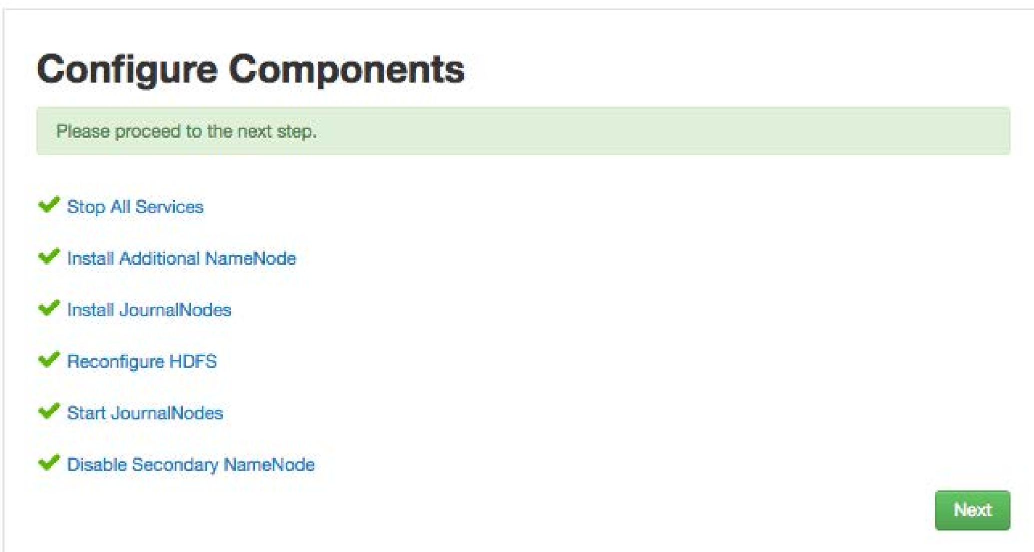
The command output should report that save namespace is successful.

The **Next** button should become active in the wizard, so click **Next**.



7 . In the Configure Components window, monitor the progress of the configuration.

Click **Next** when every item on the list has completed.

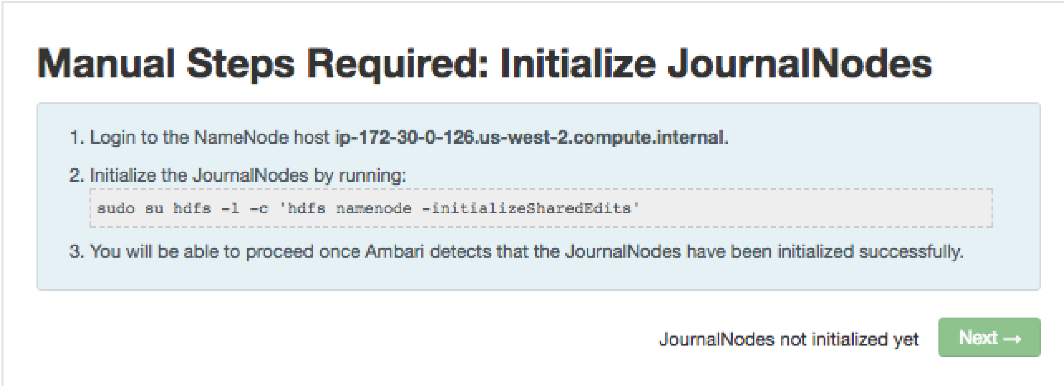


8 . The wizard lists another manual configuration step that is required next.
This command must be completed or the **Next** button in the wizard will not become active.

```
sudo su hdfs -l -c 'hdfs namenode -initializeSharedEdits'
```

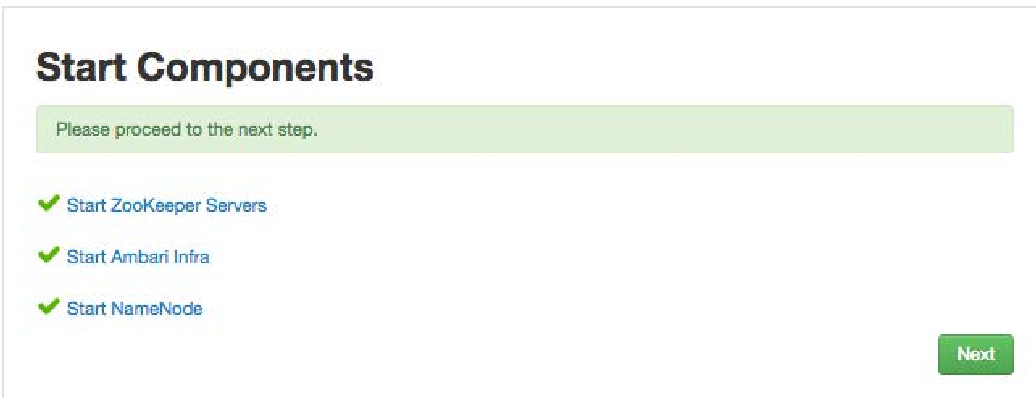
You should see a large amount of screen output. The last message should be a SHUTDOWN_MSG.

The **Next** button should become active in the wizard, so click **Next**.



9 . In the Start Components window, monitor the progress.

When the components have successfully started, click **Next**.



10 . The wizard lists more manual configuration steps that are required next.

The first command must be completed on the currently active NameNode system. The second command must be completed on the NameNode system that is being added.

NOTE: These commands must be completed first, Do *NOT* click **Next** until after manually entering both commands!

In the terminal window logged in to the currently active NameNode system, type the command shown in the wizard.

```
sudo su hdfs -l -c ‘hdfs zkfc -formatZK’
```

You should see a large amount of screen output.

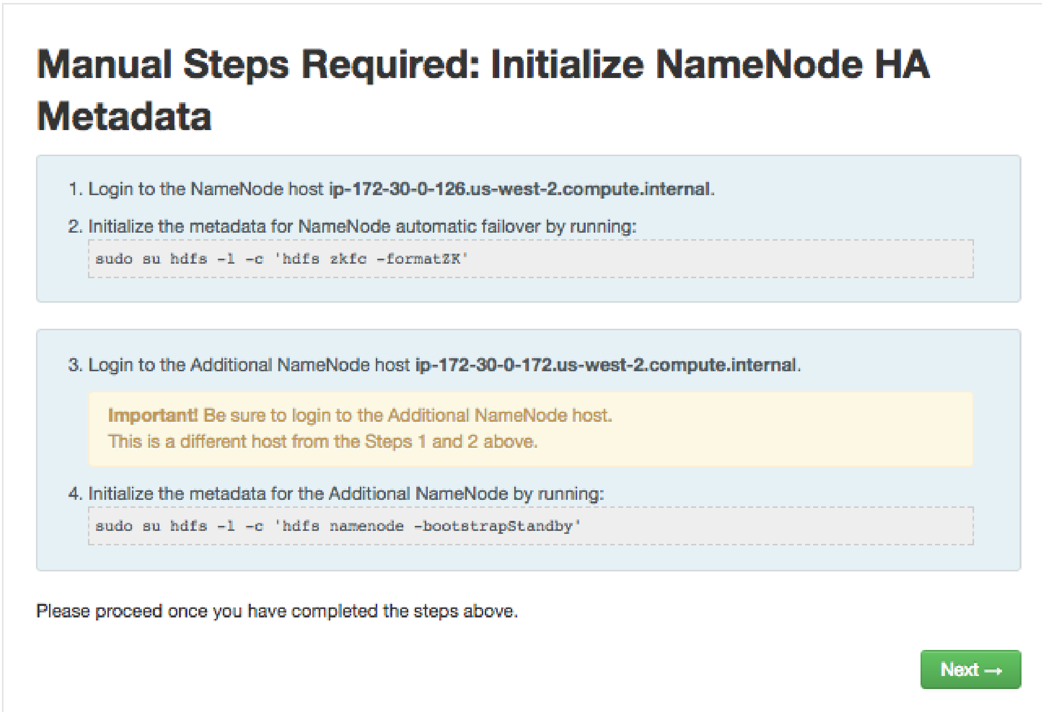
In the terminal window logged in to node with the Additional NameNode, Ambari wizard will show the hostname.

Then run the second command listed in the wizard.

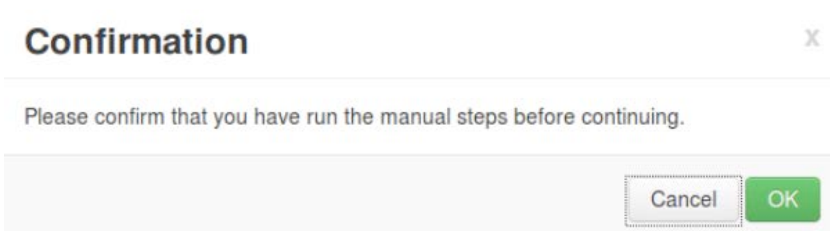
```
sudo su hdfs -l -c 'hdfs namenode -bootstrapStandby'
```

You should see a large amount of screen output. The last message should be a SHUTDOWN_MSG.

Then click **Next**.

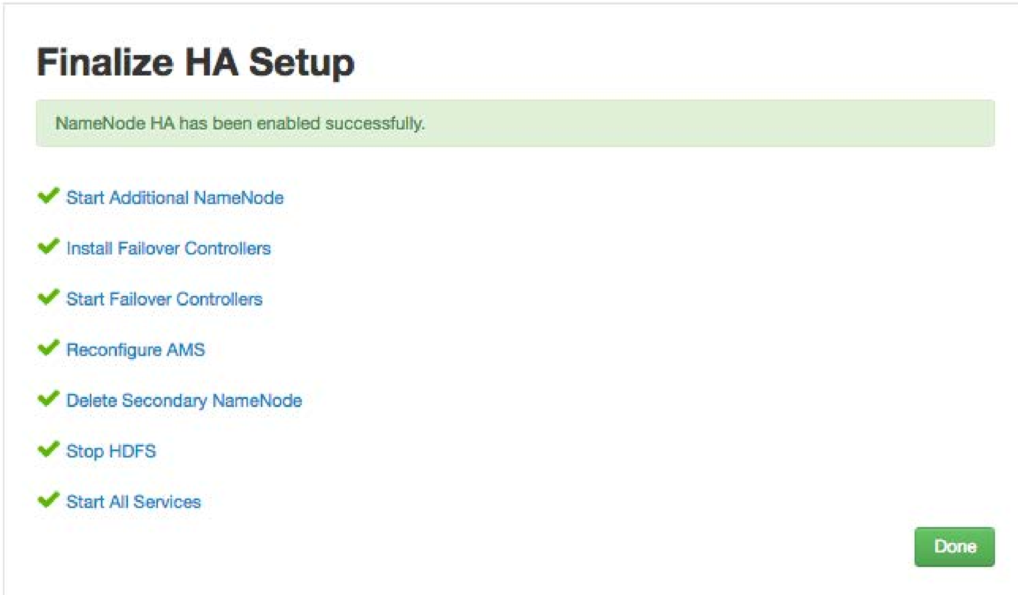


11 . In the Confirmation window click **OK**.



12 . Monitor the progress in the Finalize HA Setup window.

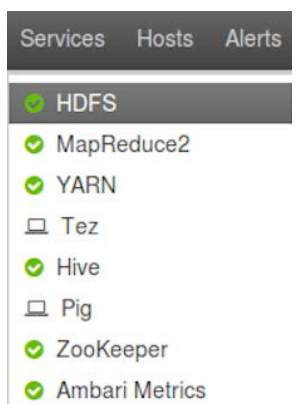
When the components have successfully started, click **Done**.
This step might take several minutes to complete.



Verifying HA Configuration.

Verify the new NameNode HA configuration

1. In the Ambari Web UI, click **Services** and select **HDFS**.



On the **Summary** page:

How many NameNodes are there?
There should be two.

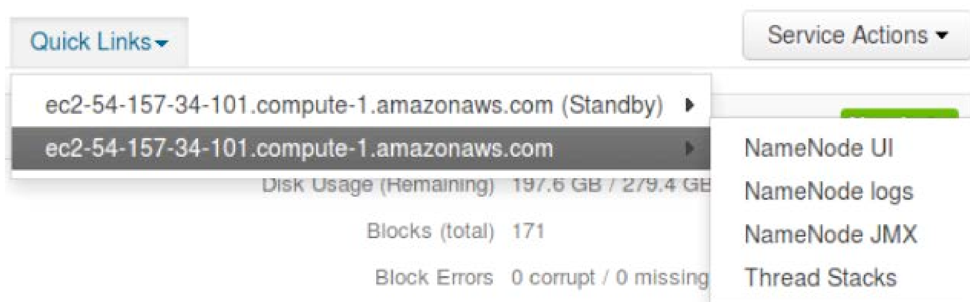
What types of NameNodes are there?
There should be an Active and Standby NameNode.

Is there a ZKFailoverController?
There should be.

Are there any JournalNodes running?
There should be three.

Click the HDFS **Quick Links** menu.

Do **Quick Links** exist to both NameNodes now?
There should be links to both.



2. In the terminal window logged in to at the beginning of this lab, view the `fs.defaultFS` property in the `/etc/hadoop/conf/core-site.xml` file.

```
more /etc/hadoop/conf/core-site.xml
```

What is the value of the property?
It should now be `hdfs://Admin-TrainingNS`*.*

3. View the contents of the `/etc/hadoop/conf/hdfs-site.xml` file.

```
more /etc/hadoop/conf/hdfs-site.xml
```

Do any of the following properties exist in the file?

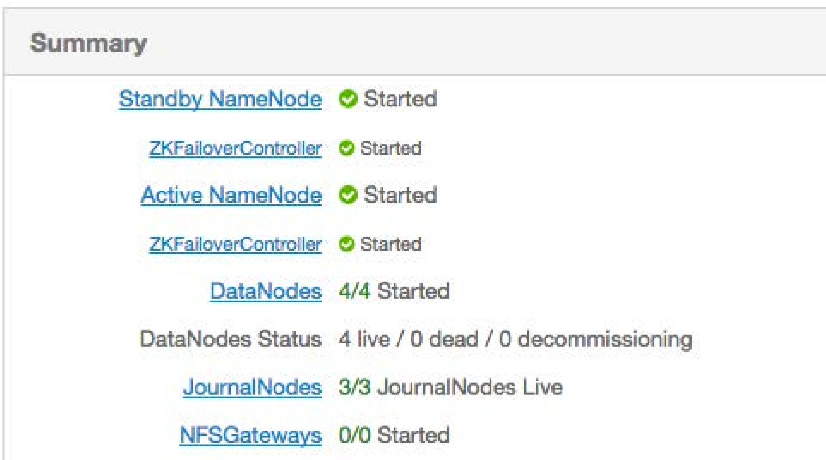
```
dfs.client.failover.proxy.provider.mycluster
dfs.ha.namenode.mycluster
dfs.namenode.http-address.mycluster.nn1
dfs.namenode.http-address.mycluster.nn2
dfs.namenode.rpc-address.mycluster.nn1
dfs.namenode.rpc-address.mycluster.nn2
dfs.nameservices
```

All of these properties should exist now.

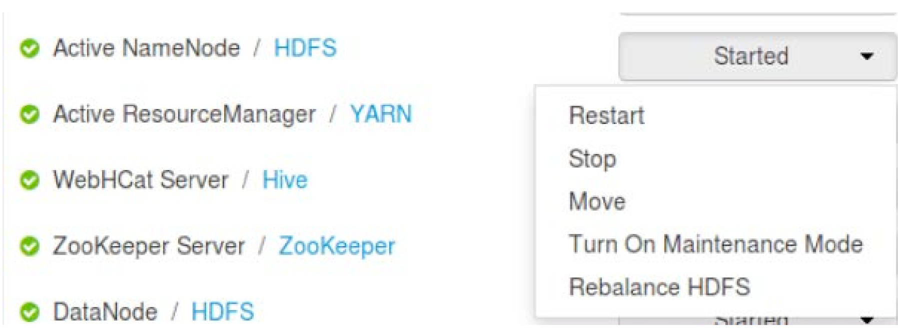
Testing Failover

Test Failover to the Standby NameNode.

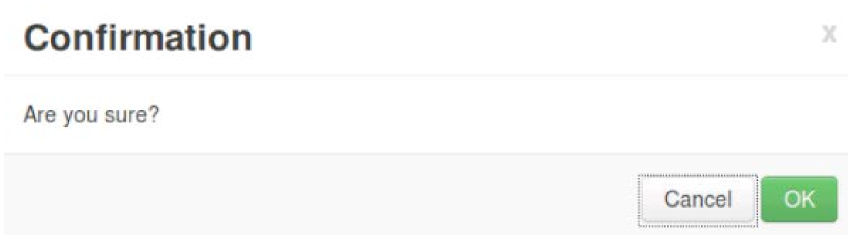
1. In the Ambari Web UI on the **Services** > **HDFS** > **Summary** page. Then click the **Active NameNode** link.



2. Find the Active NameNode in the list of components. Then click the **Started** menu button and select **Stop**.

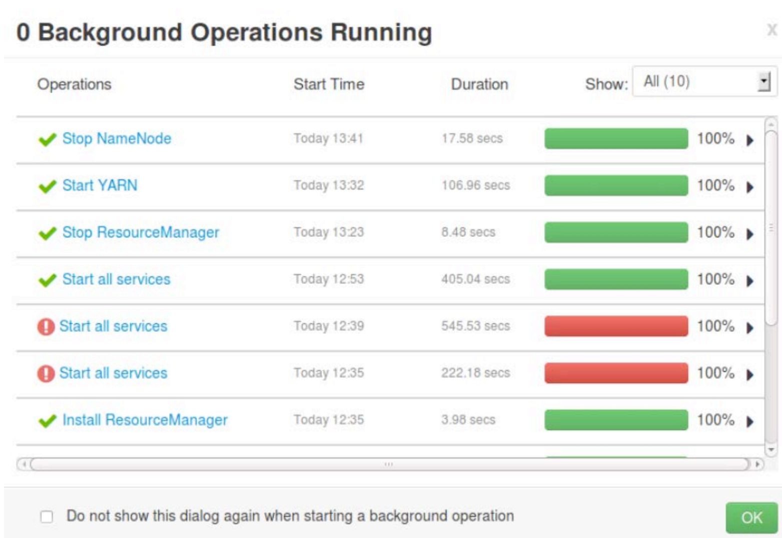


3. In the Confirmation window, click **OK**.



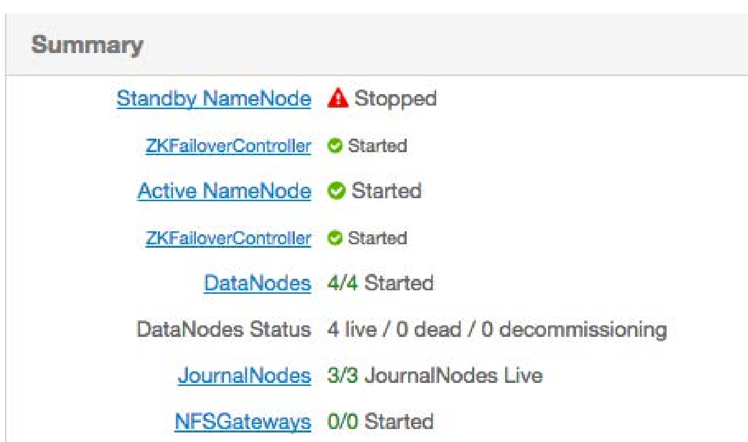
4 . In the Background Operations Running window, monitor the Stop NameNode process.

When it has finished, click **OK**.



The Active NameNode should be listed as Stopped.

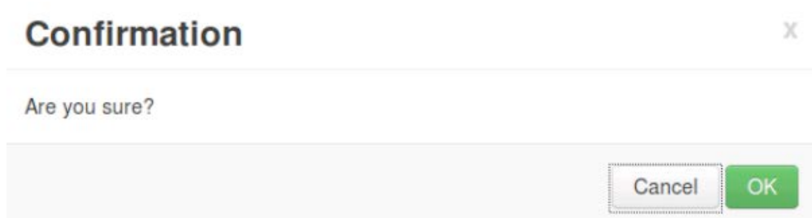
5 . Go back to the **Services** > **HDFS** > **Summary** page and either wait a minute or two or manually refresh the browser. The Summary panel should update and show the former Standby NameNode promoted to the new Active NameNode and former Active NameNode as just a NameNode and stopped.



6 . Click the stopped **NameNode** in the list of components.

7 . Find the NameNode in the list of components, click the **Stopped** menu button, and then select **Start**.

8 . In the **Confirmation** window, click **OK**.

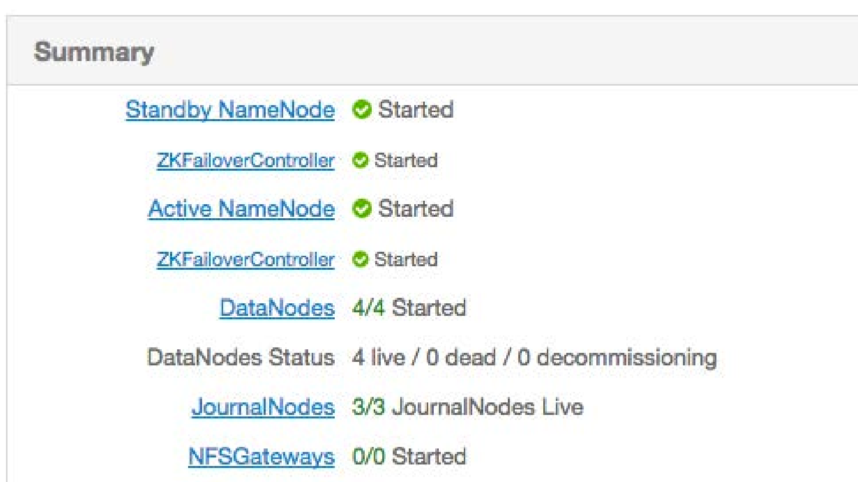


9 . In the Background Operations Running window, monitor the Start NameNode process.

When it has finished, click **OK**.

10 . Use the information on the **Services** > **HDFS** > **Summary** page to ensure that there is an Active and a Standby NameNode.

To see the change either wait for a browser update or manually refresh your browser.



Result

You have now:
Configured an Active and a Standby NameNode running in your cluster, three ZooKeeper Servers to support NameNode HA, a ZooKeeper FailoverController on each NameNode system, and three running JournalNodes. You have also removed the existing Secondary NameNode and tested automatic failover.

Lab 23: Configuring ResourceManager HA

About This Lab

Objective:	To use the Ambari Web UI to configure a cluster with YARN ResourceManager HA in order to have an Active and a Standby ResourceManager running in your cluster
File locations:	N/A
Successful outcome:	**You will:** View ResourceManager cluster configuration, verify there are three ZooKeeper servers supporting ResourceManager HA, enable ResourceManager HA using the Ambari Web UI, verify the new ResourceManager HA configuration and failover to a Standby ResourceManager.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Configuring HDFS and YARN High Availability***

Viewing ResourceManager Configuration Information

View the ResourceManager and cluster configuration prior to configuring ResourceManager HA.

1. Use SSH to login to any cluster node.

2. View the contents of the `/etc/hadoop/conf/yarn-site.xml` file and find the property `yarn.resourcemanager.ha.enabled`.

```
more /etc/hadoop/conf/yarn-site.xml
```

What is the value of this property?
It should be false.

3. While still looking at the `yarn-site.xml` file, do any of the following properties exist?

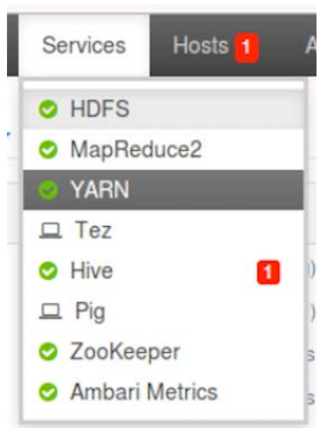
```
yarn.client.failover-proxy-provider
yarn.resourcemanager.cluster-id
yarn.resourcemanager.ha.rm-ids
yarn.resourcemanager.hostname.rm1
yarn.resourcemanager.hostname.rm2
yarn.resourcemanager.webapp.address.rm1
yarn.resourcemanager.webapp.address.rm2
```

None of these properties should exist.
(This is not a complete list of ResourceManager HA properties.)

4. Open the browser and connect to the Ambari Server at the URL

http://<Ambari Server Hostname>:8080

5. Login to the Ambari Web UI using the user name `admin` and the password `BadPass#1`.
6. Click **Services** and select **YARN**.

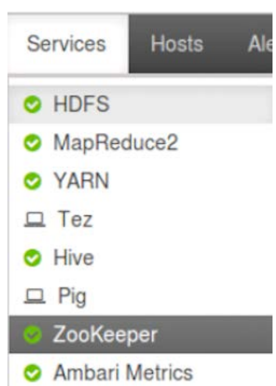


On the **Summary** page do you see a **ResourceManager** and an **App Timeline Server**?
You should see both.

Verifying the Number of ZooKeeper Servers

Verify that there are three ZooKeeper Servers configured to support ResourceManager HA.

1. In the Ambari Web UI, click **Services** and select **ZooKeeper**.

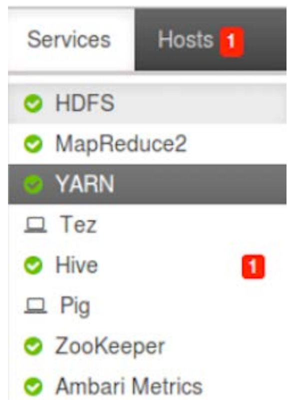


How many ZooKeeper Servers are shown on the **Summary** page?
There should be three.

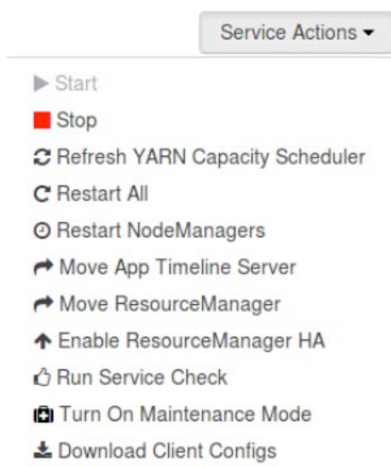
Enabling ResourceManager HA

Enable ResourceManager HA using the Ambari Web UI.

1. In the Ambari Web UI, click **Services** and select **YARN**.

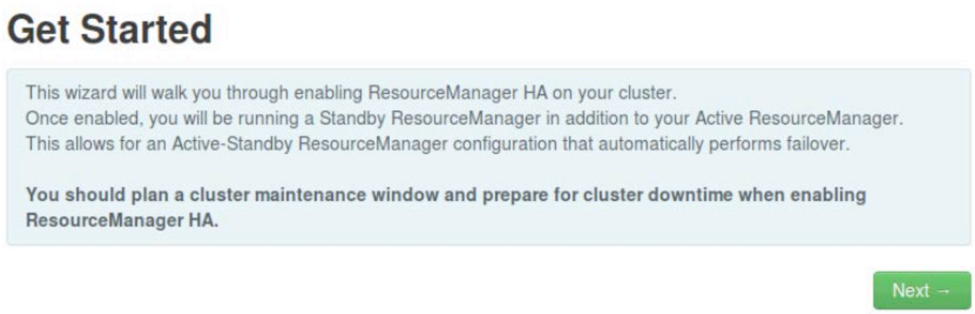


2. Click the **Service Actions** menu button and select **Enable ResourceManager HA**.

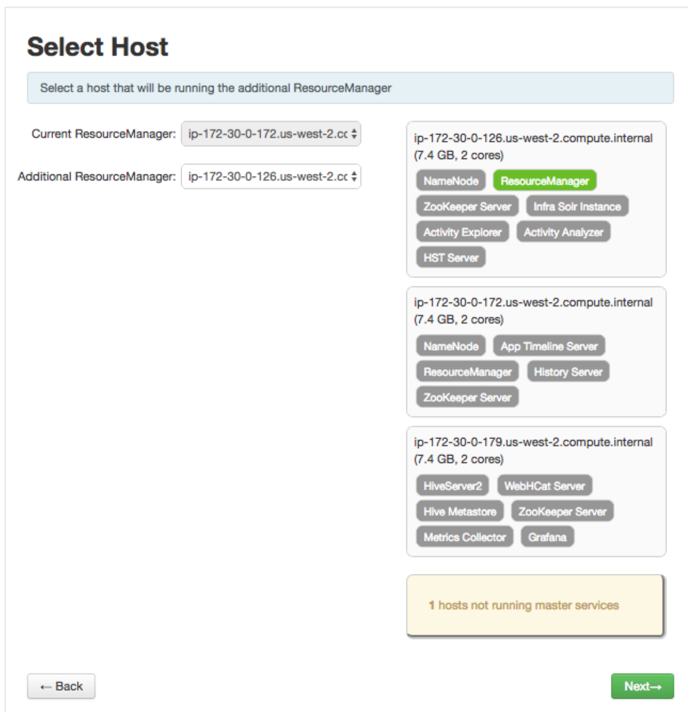


The Enable ResourceManager HA Wizard window opens.

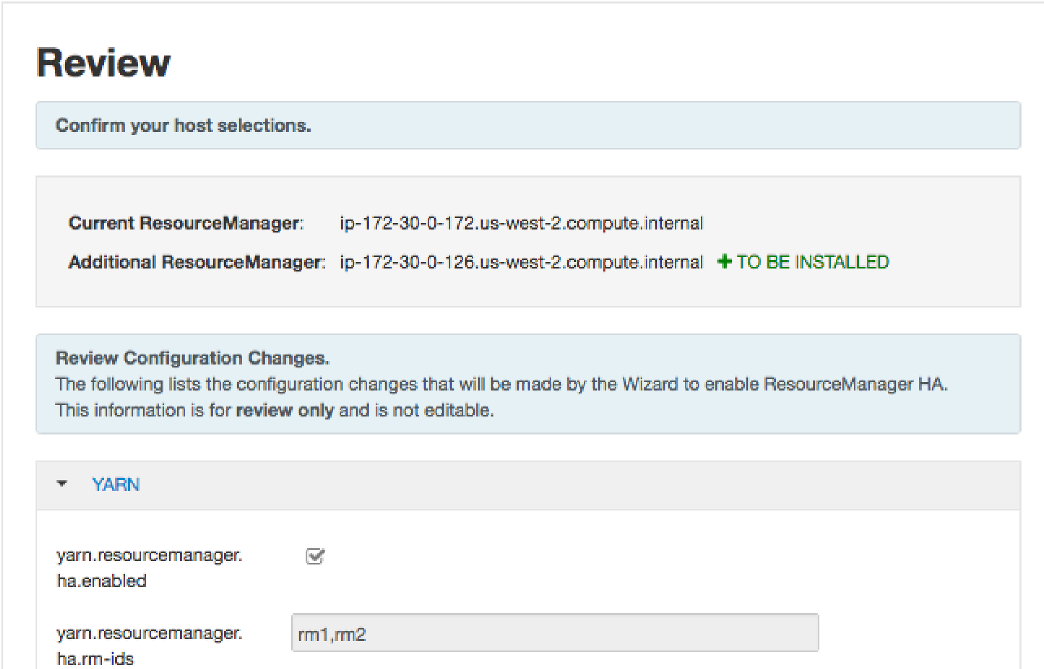
3. Read the information in the Get Started panel and then click **Next**.



4 . Then click **Next**.



5 . Read the information in the Review window. Notice that automatic recovery is enabled by default. No changes are necessary so click **Next** to continue.



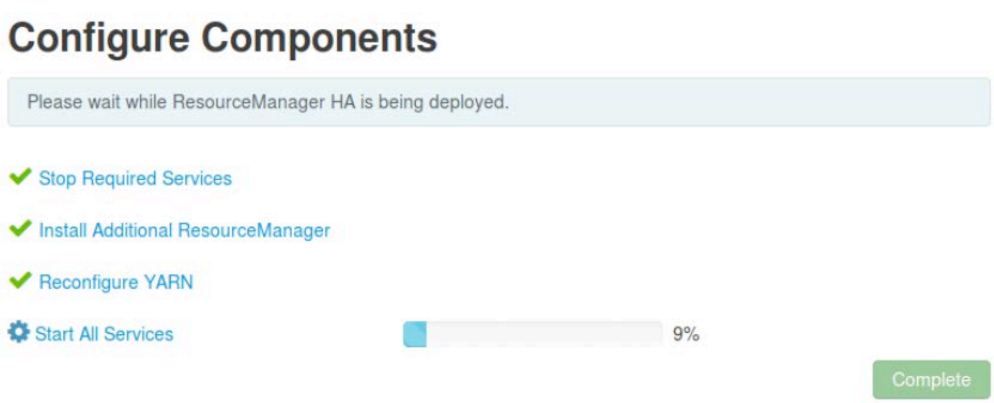
6 . In the Configure Components window, monitor the progress of the configuration.

Click **Complete** when every item on the list has completed.
(This step will take several minutes to complete.)

If there is any failure to start a service, a **Retry** button is presented in the wizard. Click it to continue starting components if necessary.

If this fails more than once or twice, it is likely the default queue has not been configured to allow applications to run, or does not have sufficient resources available. You can view the detailed error messages to confirm look for the Pig Smoke Test job as the failure point.

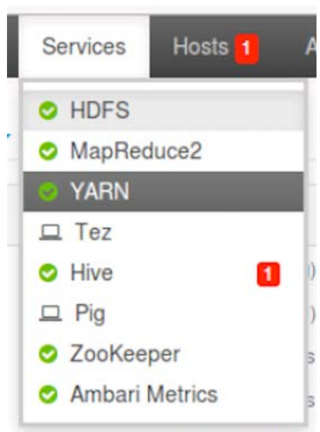
If this happens, simply cancel the wizard. You will see an error message stating that you must manually roll back the configuration changes. However, the failed smoke test was the final step. You will find if you simply restart your cluster services that ResourceManager HA configuration has actually succeeded.



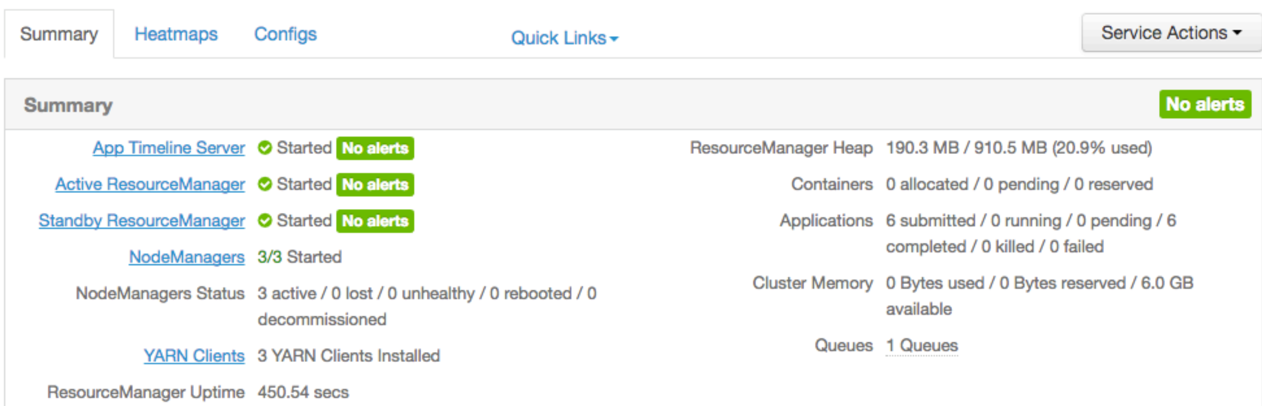
Verifing ResourceManager HA configuration

Verify the new ResourceManager HA configuration.

1 . In the Ambari Web UI, click **Services** and select **YARN**.



On the **Summary** page:

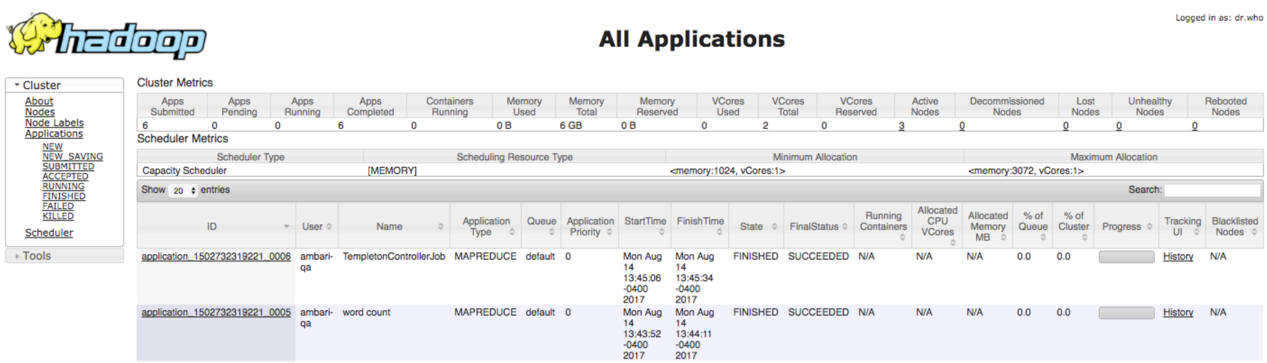


How many ResourceManagers are there?
There should be an Active and Standby ResourceManager.

Which cluster node is running the Active ResourceManager?

2 . Open the **Quick Links** menu, you will see both ResourceManagers one "Active" and one "Stanby". Click on the "Active" to open the ResouceManage UI

This should open the ResourceManager UI.



Did the ResourceManager UI open?
It should have.

3 . If you click on the "StandBy" it will automatically be redirected to the Active ResourceManager.

It was redirected because any Web UI or YARN REST API request received by the Standby ResourceManager is automatically redirected to the Active ResourceManager by ResourceManager HA.

4 . Use SSH to login to any cluster node, view the contents of the `/etc/hadoop/conf/yarn-site.xml` file.

```
more /etc/hadoop/conf/yarn-site.xml
```

What is the value of the `yarn.resourcemanager.ha.enabled` property?

It should be `true`.

Do any of the following properties exist in the file?

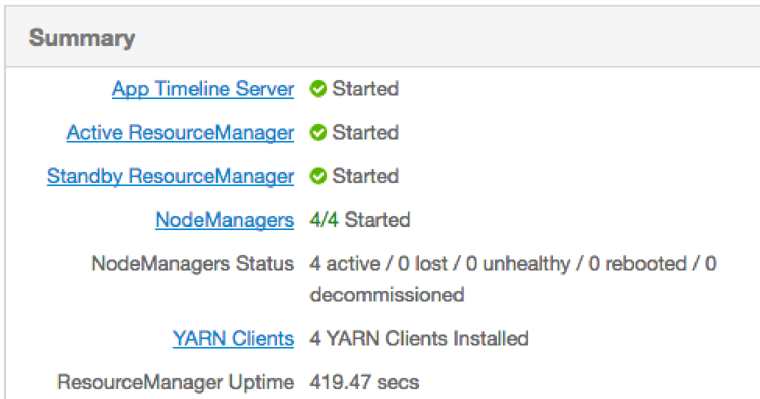
```
yarn.client.failover-proxy-provider
yarn.resourcemanager.cluster-id
yarn.resourcemanager.ha.rm-ids
yarn.resourcemanager.hostname.rm1
yarn.resourcemanager.hostname.rm2
yarn.resourcemanager.webapp.address.rm1
yarn.resourcemanager.webapp.address.rm2
```

All of these properties should exist now.

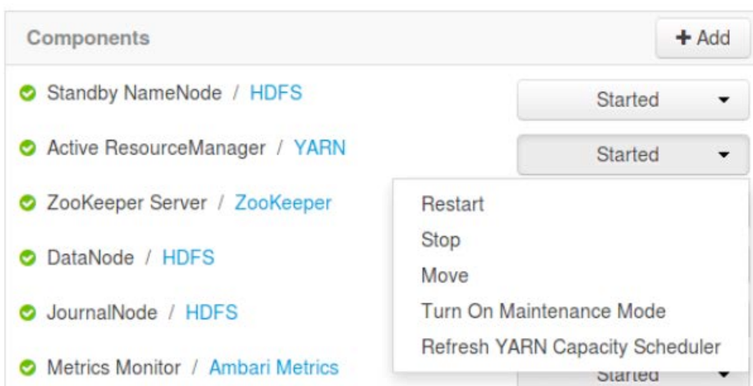
Testing Failover

Failover to the Standby ResourceManager.

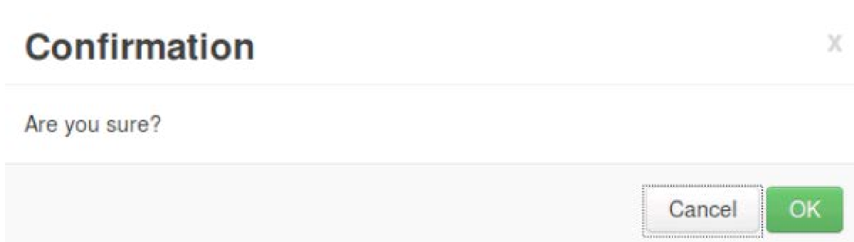
1 . In the Ambari Web UI click on the **Services** > **YARN** > **Summary** page, Click the **Active ResourceManager** link.



2 . Find the Active ResourceManager in the list of components, Click the **Started** menu button, and then select **Stop**.

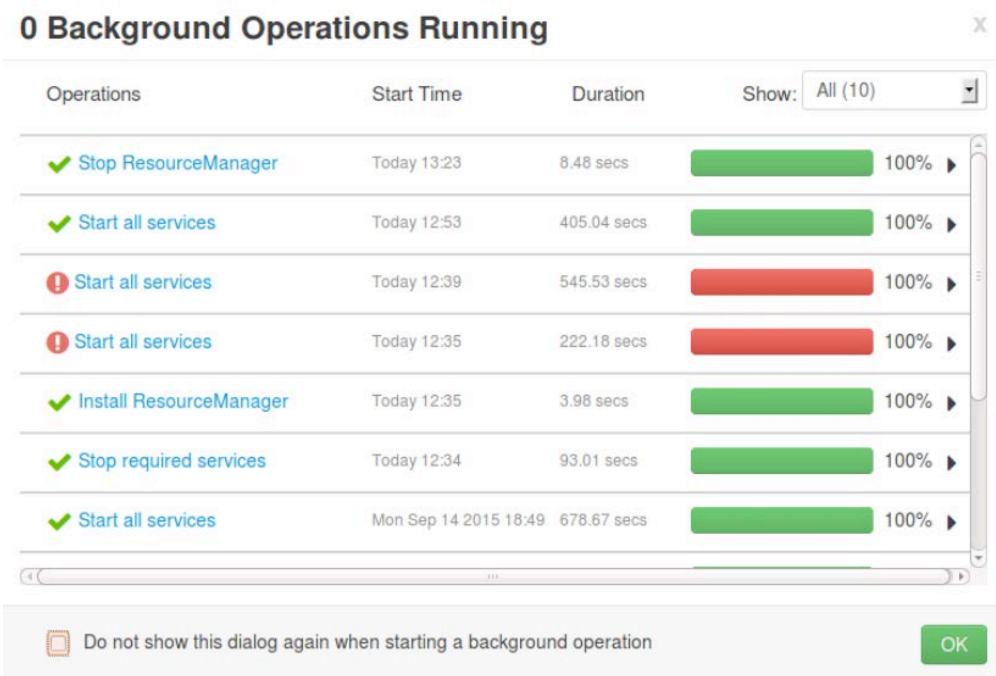


3 . In the Confirmation window, click **OK**.



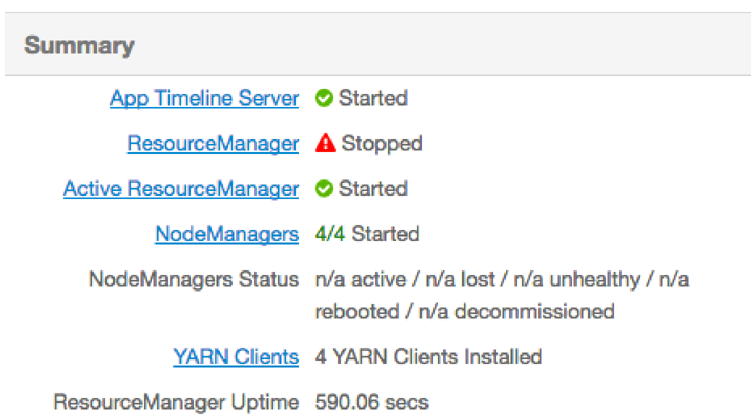
4 . In the Background Operations Running window, monitor the Stop ResourceManager process.

When it has finished, click **OK**.



The Active ResourceManager should be listed as Stopped.

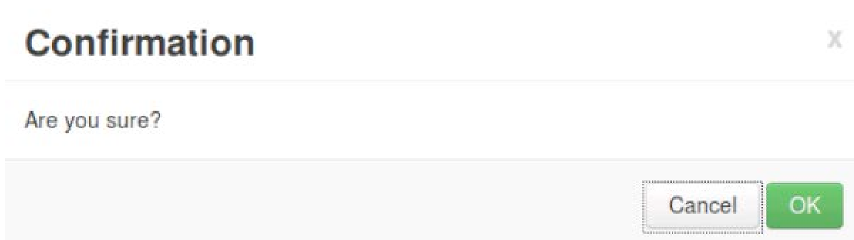
5 . Go back to the **Services** > **YARN** > **Summary** page and either wait a minute or two or manually refresh the browser. The Summary panel should update and show the former Standby ResourceManager promoted to the new Active ResourceManager and former Active ResourceManager as just a stopped ResourceManager.



6 . Click the stopped **ResourceManager** in the list of components.

7 . Find the ResourceManager in the list of components, click the **Stopped** menu button, and then select **Start**.

8 . In the Confirmation window, click **OK**.

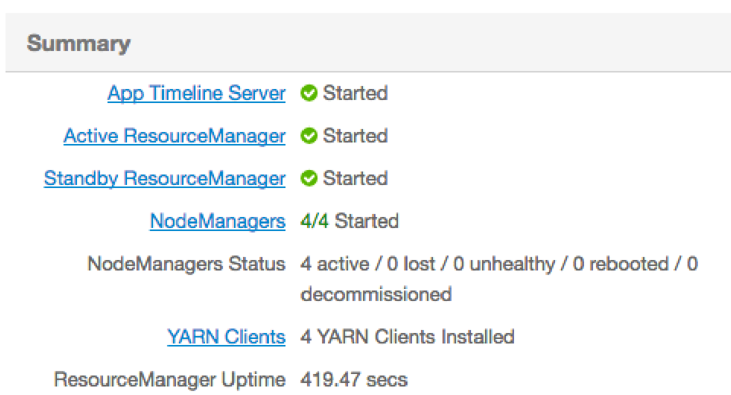


9 . In the Background Operations Running window, monitor the Start ResourceManager process.

When it has finished, click **OK**.

10 . Use the information on the **Services** > **YARN** > **Summary** page to ensure that there is both an Active and a Standby ResourceManager.

You might have to either wait for a browser update or manually refresh your browser.



Result

You have now:
Viewed ResourceManager cluster configuration, verified there are three ZooKeeper servers supporting ResourceManager HA, enabled ResourceManager HA using the Ambari Web UI, verified the new ResourceManager HA configuration and performed a failover to a Standby ResourceManager.

Lab 24: Adding, Decommissioning, and Recommissioning Worker Nodes

About This Lab

Objective:	To use the Ambari Web UI to add one worker node, decommission and recommission a worker node, change the HDFS replication factor to three, and balance HDFS data blocks across DataNodes
File locations:	N/A
Successful outcome:	**You will:** Add one node to the cluster, decommission and then recommission one of the nodes, turn off maintenance mode, change the default replication factor to three, and run the HDFS balancer.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Adding, Deleting, and Replacing Worker Nodes***

Adding Worker Nodes

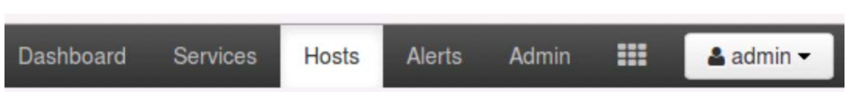
Add two worker nodes to your cluster.

1. Open the browser and connect to the Ambari Server at the URL

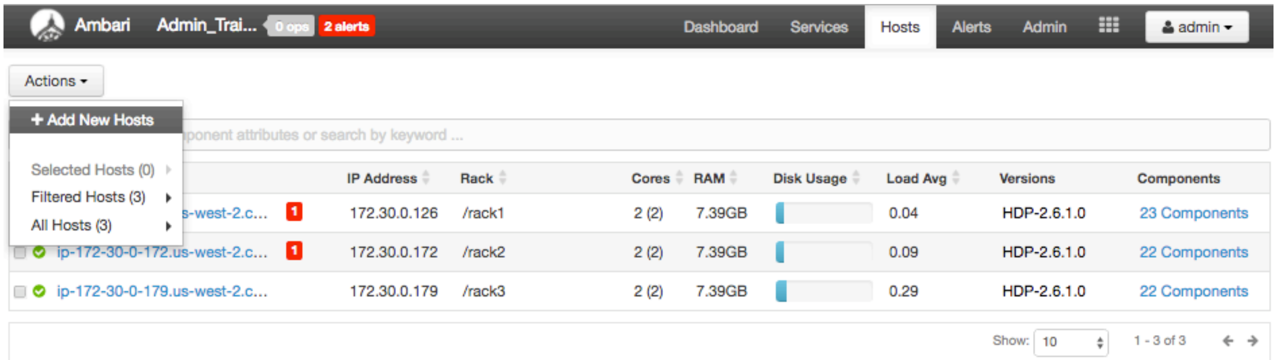
   http://<Ambari Server Hostname>:8080

2. Login to the Ambari Web UI using the user name `admin` and the password `BadPass#1`.

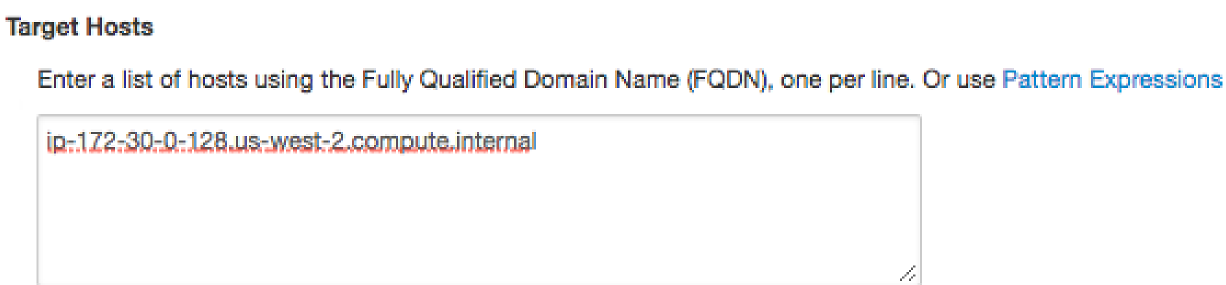
3. Click **Hosts** to open the **Hosts** page.



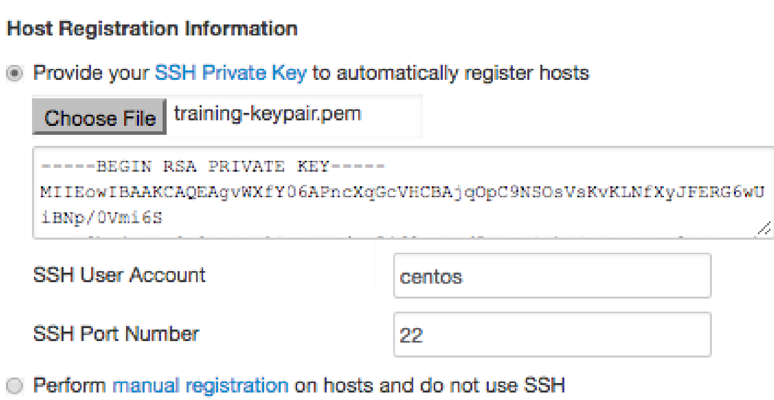
4. Click the **Actions** menu button and select **Add New Hosts**.



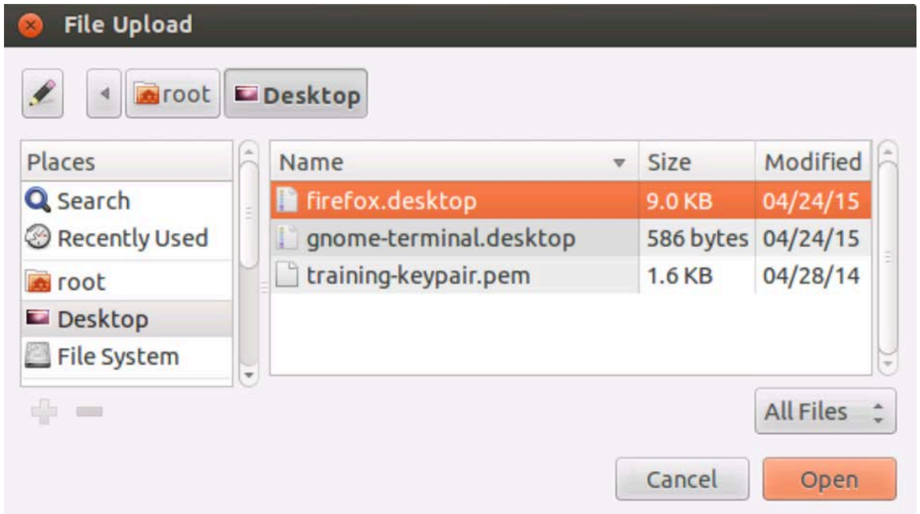
5. Under Target Hosts, type the AWS Internal Hostname of your remaining host.



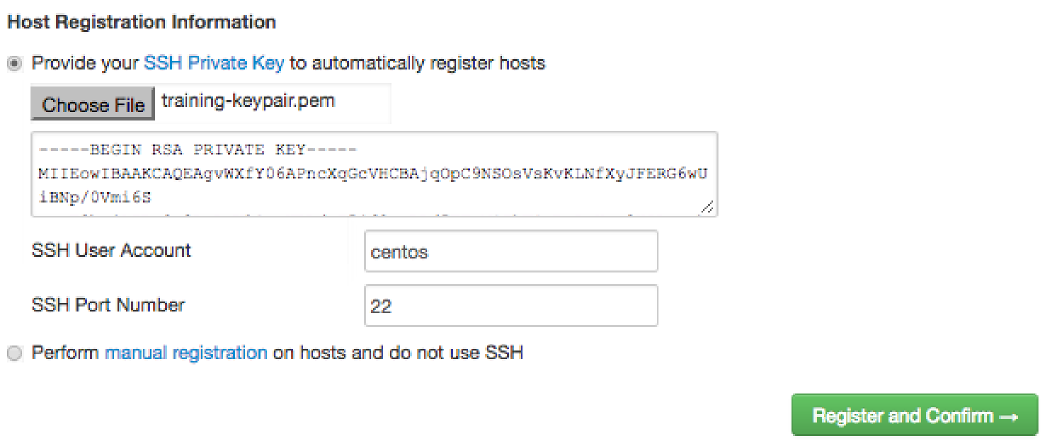
6. Scroll down in the window and under Host Registration Information click **Browse**.



7. Select `training-keypair.pem` and click **Open**. Make sure the `SSH User Account` is set to the `centos` user.



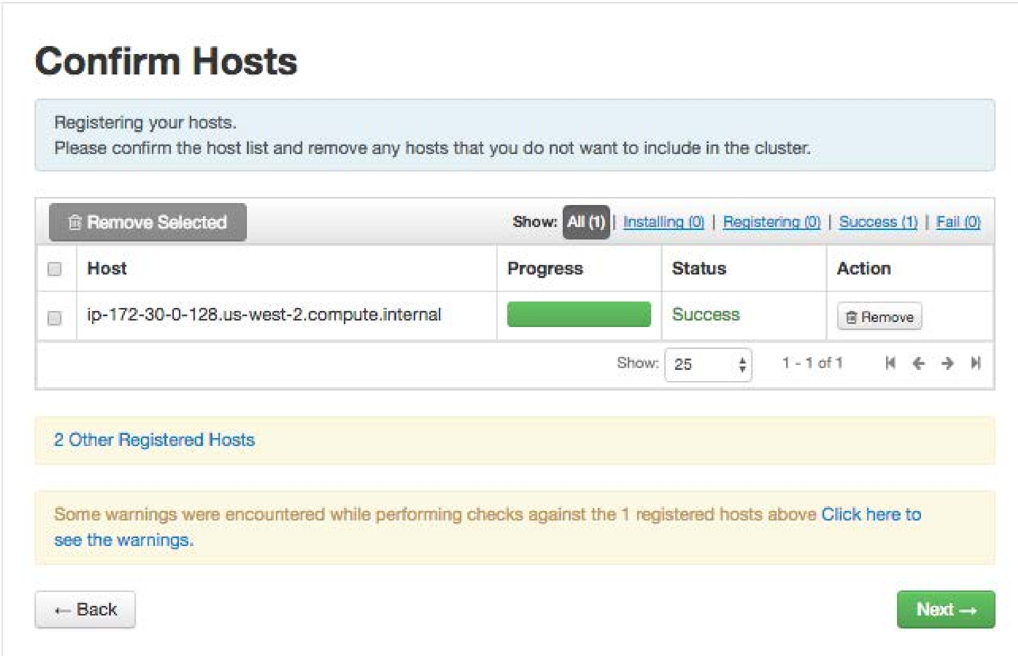
8 . Once you see the private RSA key displayed, click **Register and Confirm**.



9 . Monitor the agent installation progress window.

The agent is installed and registered and then a series of system checks are performed on the host to be installed. You may receive a few warnings but these can be ignored in the lab environment.

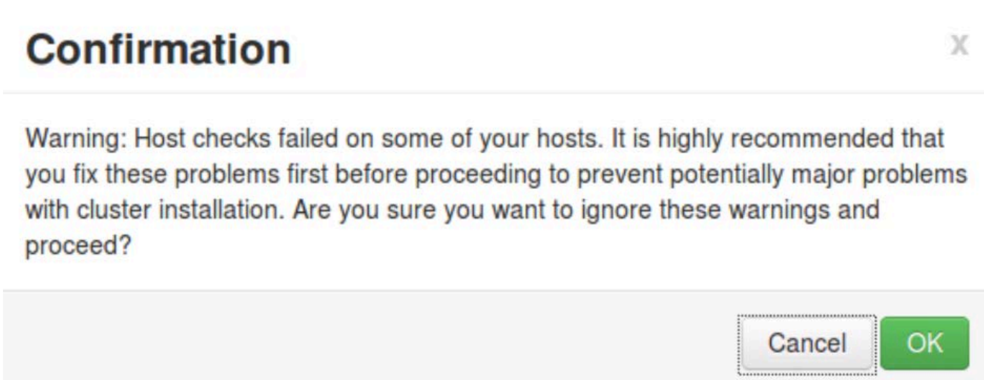
Click **Next** to continue the installation.



10 . A Confirmation window opens and warns that some of the host checks failed.

The failures are minor in the lab environment and will not cause problems.

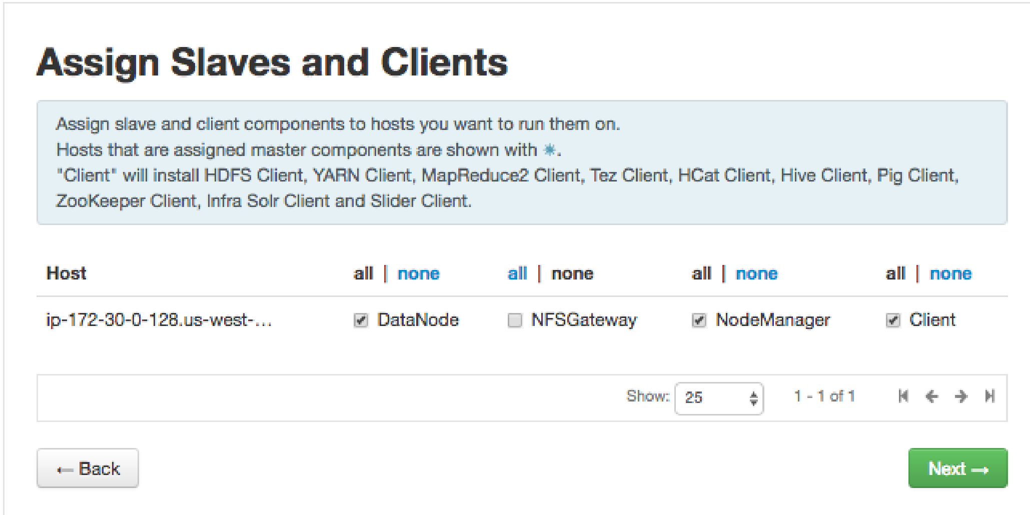
Click *OK* to dismiss the window and continue the installation.



11 . The Assign Slaves and Clients window is where you choose which cluster nodes will run which worker processes. Worker processes perform most of the data processing in a cluster. The more nodes that you have running worker processes, the more work that can be simultaneously accomplished.

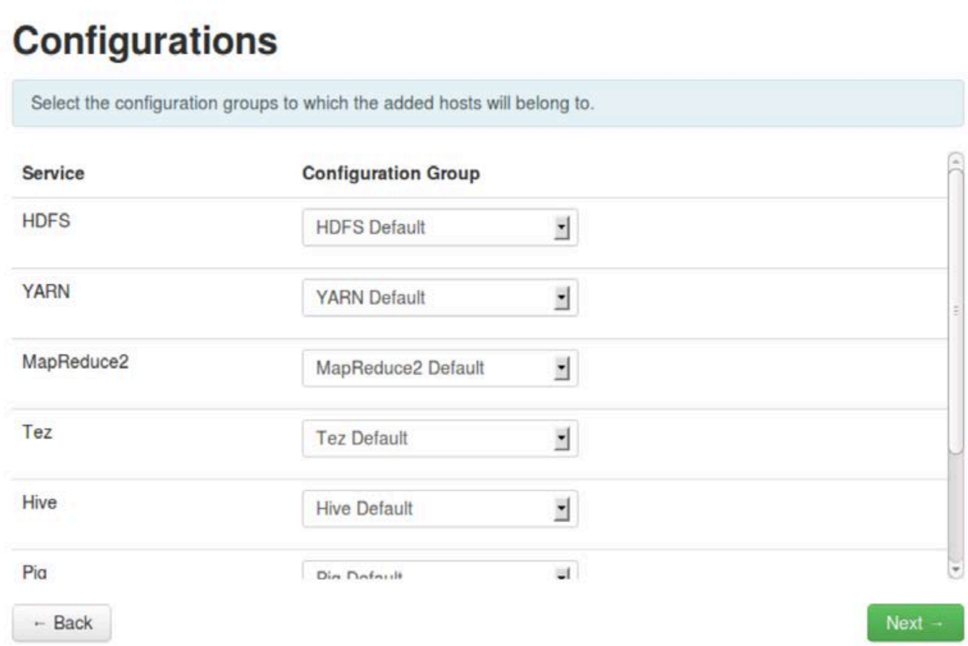
If necessary, click the **DataNode, NodeManager**, and **Client** check boxes for this Host.

Then click **Next** to continue.

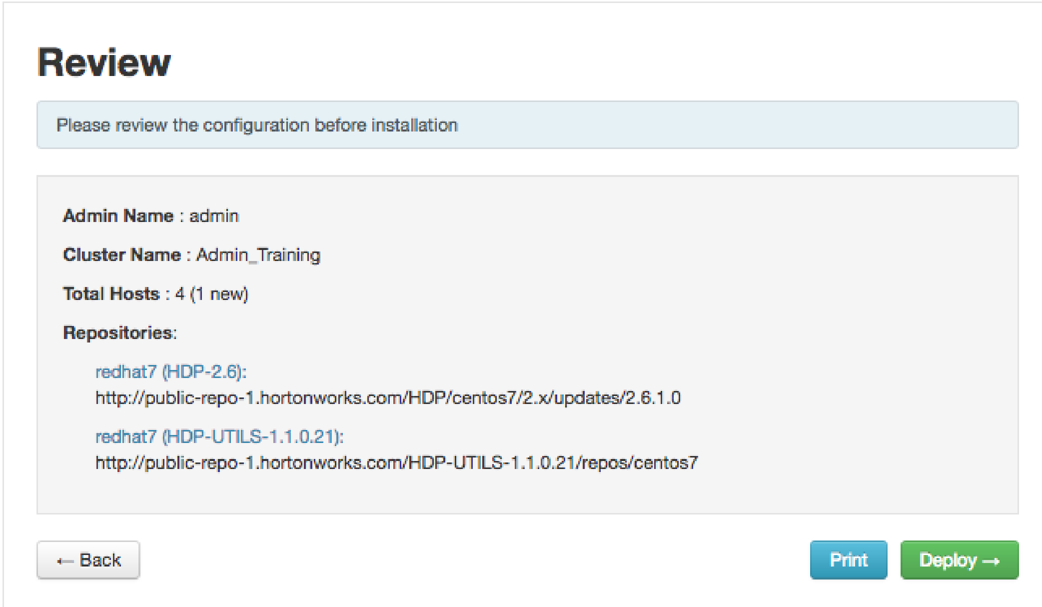


12 . If this host should belong to any specific Ambari Configuration Group you can select it in the Configurations window.

This new host should belong to the default configuration group for each service, so just click **Next**.

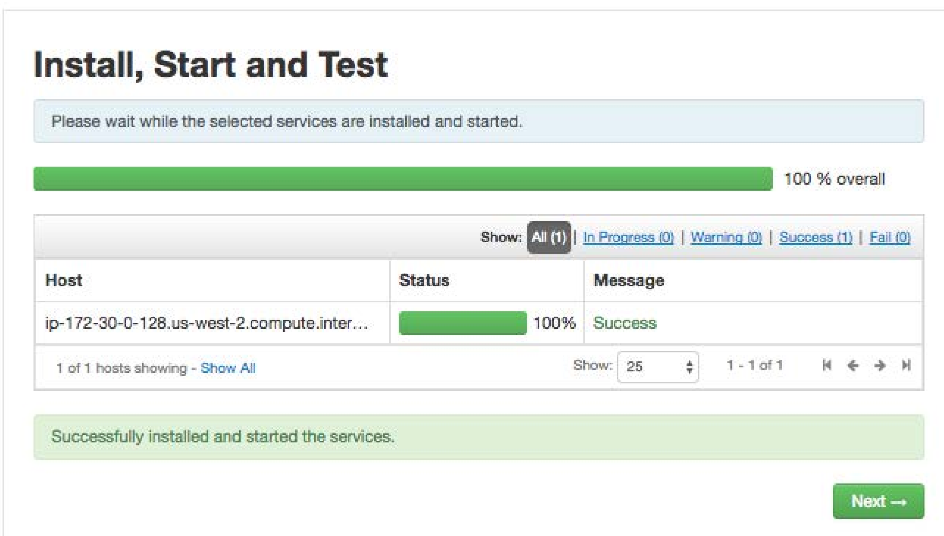


13 . Review your selections for accuracy and then click **Deploy**.

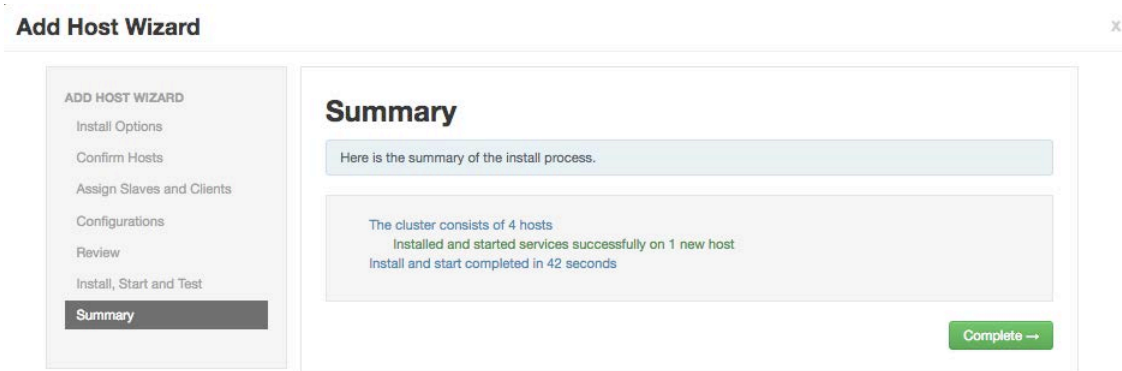


14 . Monitor the installation progress.

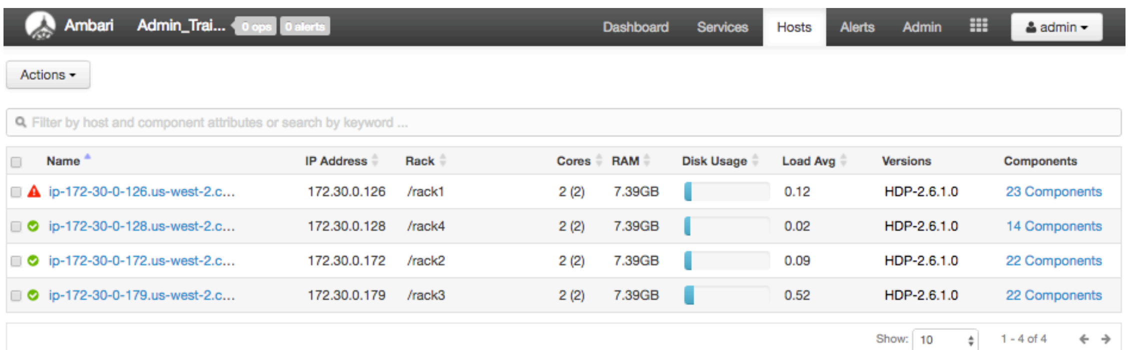
15 . When the progress window indicates that the installation has completed, click **Next**.



16 . Read the Summary window and click **Complete**.



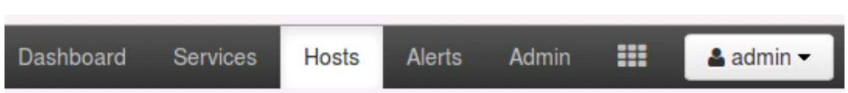
17 . The lists of hosts should now include your new host.



Decommissioning Worker Nodes

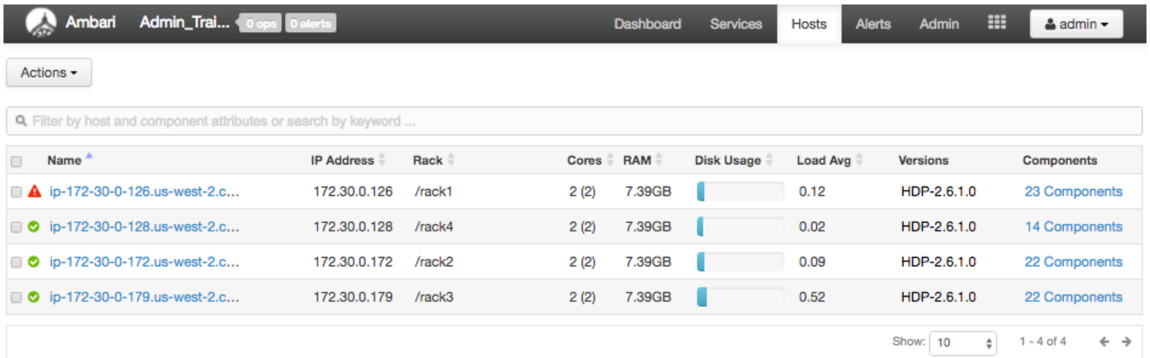
Decommission a worker node.

1 . If necessary, click **Hosts** in the Ambari Web UI.

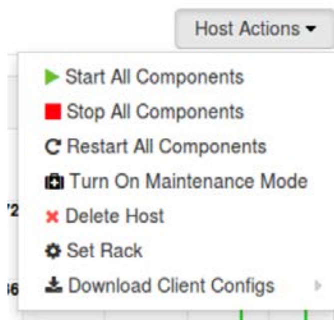


2 . In this lab use the host that was just added in the previous steps. Example

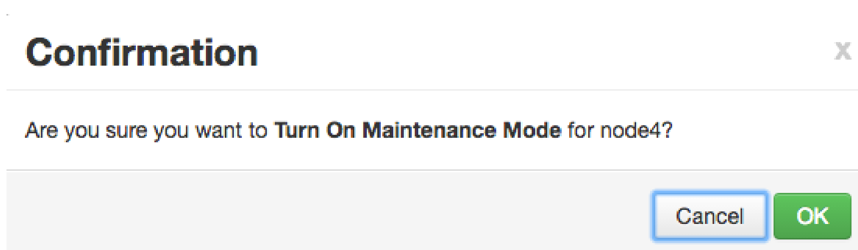
Click on "`ip-172-30-0-128.us-west-2.compute.internal`" in the list of cluster Hosts.



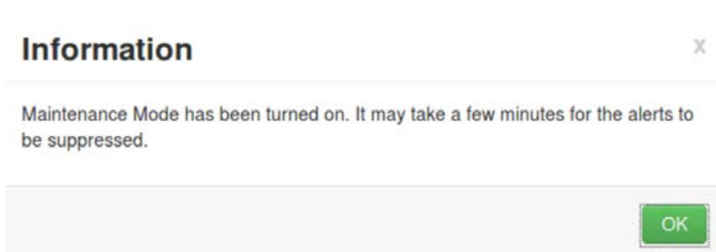
3 . Click the **Host Actions** menu button and select **Turn On Maintenance Mode**.



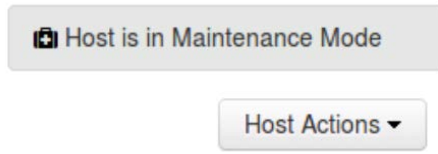
4 . In the Confirmation window, click **OK**.



5 . Read the information in the Information window, and click **OK**.

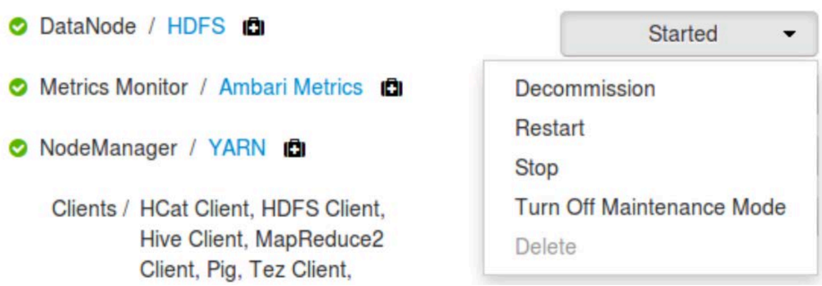


6 . Notice the new message above the **Host Actions** menu button.

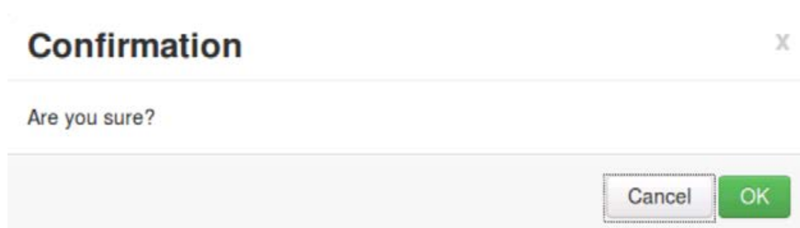


7 . Find the DataNode component listed in the Hosts window of the select host.

Click the **Started** menu button and select **Decommission**.

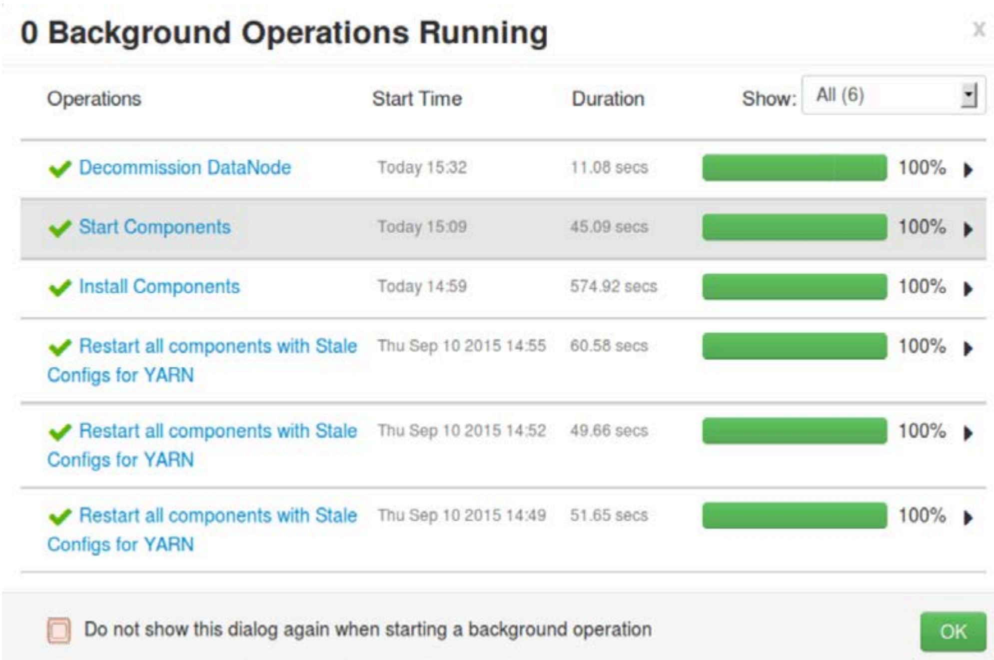


8 . In the **Confirmation** window, click **OK**.



9 . Monitor the Background Operations Running window until the Decommission DataNode process has completed.

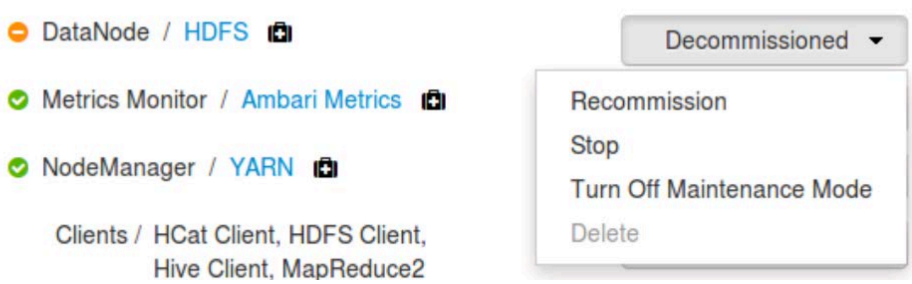
Then click **OK** to dismiss the window.



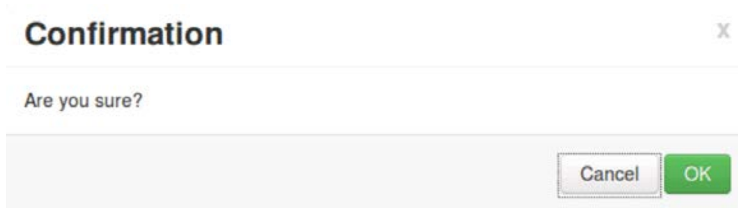
10 . The DataNode should be listed as **Decommissioned**.

11 . If you were going to take the node offline, you can also stop the DataNode process.

Click the **Decommissioned** menu button and select **Stop**.



12 . In the Confirmation window, click **OK**.

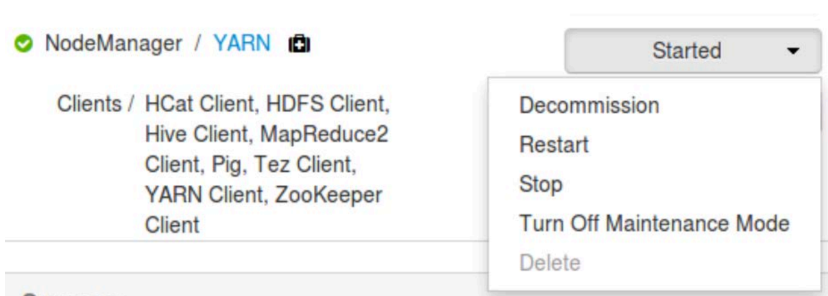


13 . Monitor the Background Operations Running window until the Stop DataNode process has completed.

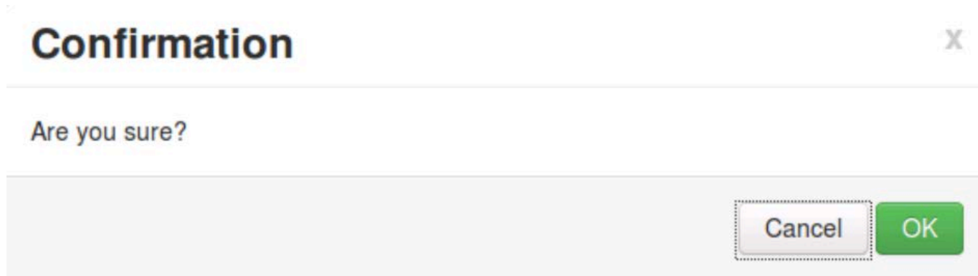
Then click **OK** to dismiss the window.

14 . Find the **NodeManager** component listed in the Hosts window of the select host.

Click the **Started** menu button and select **Decommission**.



15 . In the Confirmation window, click **OK**.



16 . Monitor the Background Operations Running window until the Decommission NodeManager process has completed.

Then click **OK** to dismiss the window.

17 . The DataNode should be listed as **Decommissioned** and stopped.

18 . Ambari updates the `/etc/hadoop/conf/dfs.exclude` and `/etc/hadoop/conf/yarn.exclude` files by adding the host name of the decommissioned DataNode and NodeManager.

View the contents of these files now by opening a terminal window and using SSH to connect to the NameNode and the ResourceManger hosts. Use the Ambari Web UI to determine which hosts are running Namenode and ResourceManager

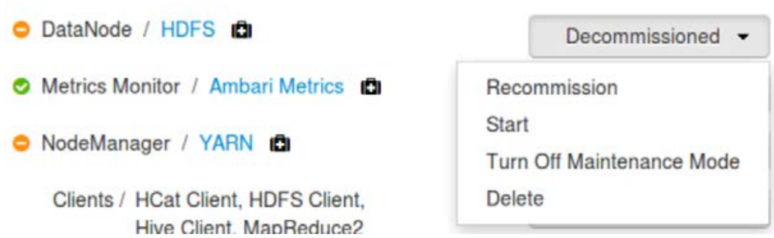
```
ssh <NameNode AWS Hostname>
more /etc/hadoop/conf/dfs.exclude

ssh <ResourceManager AWS Hostname>
more /etc/hadoop/conf/yarn.exclude
```

The host you selected should be listed in each file.

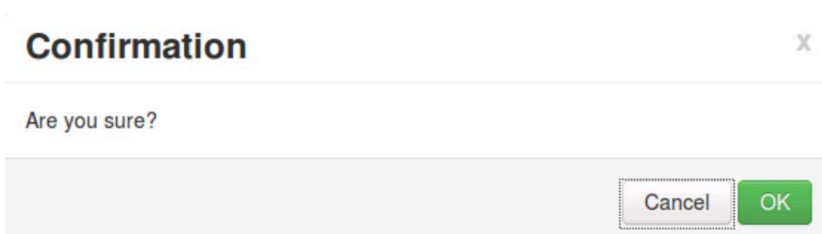
Recommissioning Worker Nodes

Recommission a worker node.

1 . On the selected host page in the Ambari Web UI, start the DataNode process by clicking the **Decommissioned** menu button and selecting **Start**.

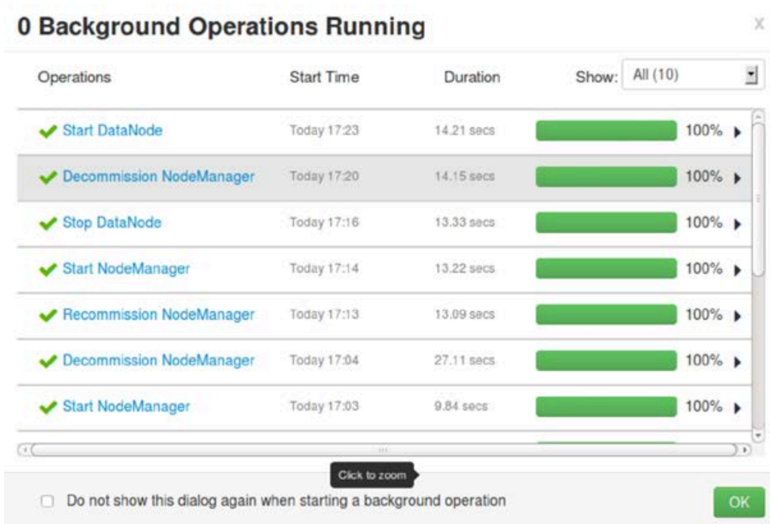


2 . In the Confirmation window, click **OK**.

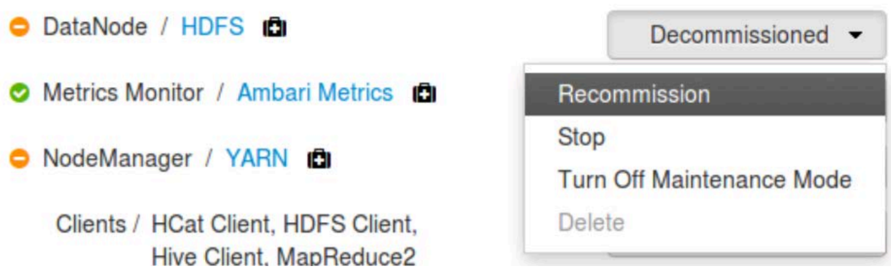


3 . Monitor the Background Operations Running window until the Start DataNode process has completed.

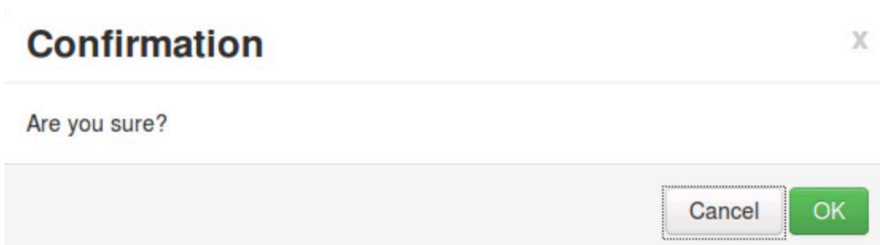
Then click **OK** to dismiss the window.



4 . For the DataNode, click the **Decommissioned** menu button and select **Recommission**.

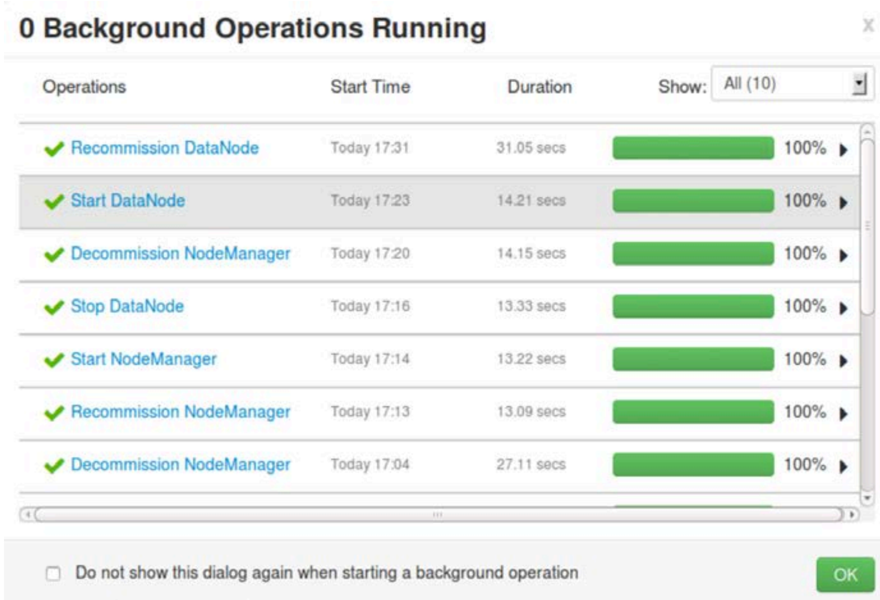


5 . In the Confirmation window, click **OK**.

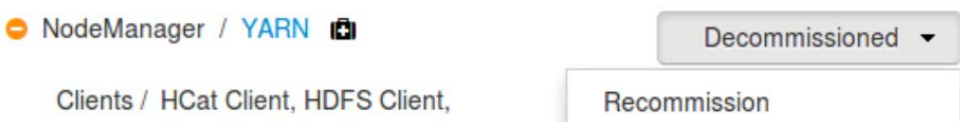


6 . Monitor the Background Operations Running window until the Recommission DataNode process has completed.

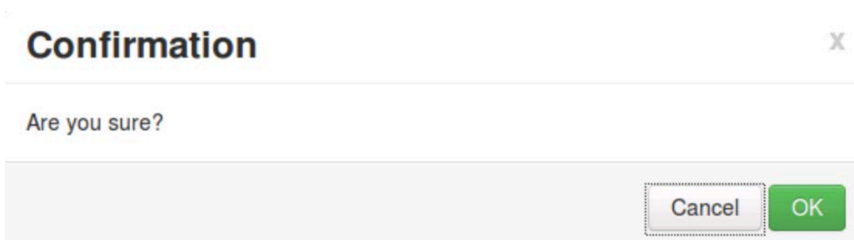
Then click **OK** to dismiss the window.



7 . For the NodeManager, click the **Decommissioned** menu button and select **Recommission**.

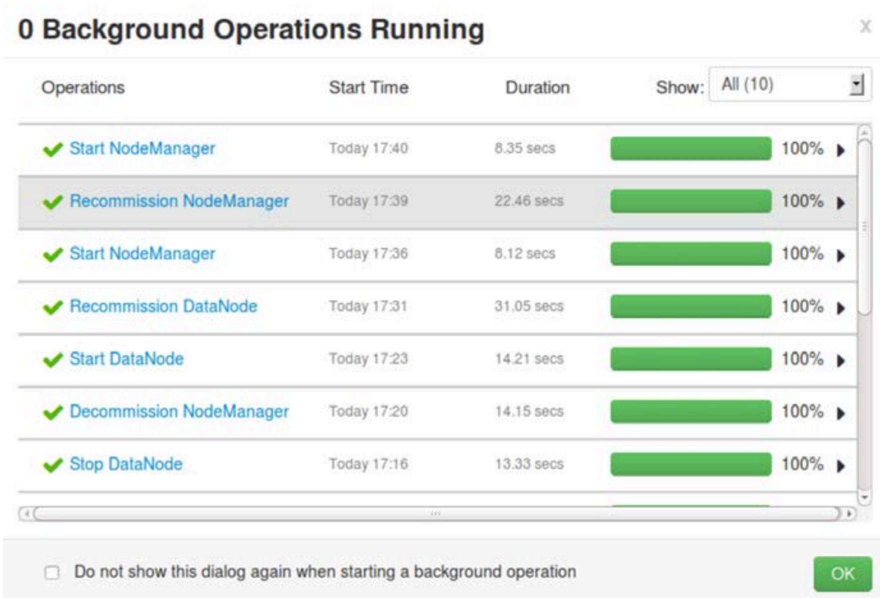


8 . In the Confirmation window, click **OK**.



9 . Monitor the Background Operations Running window until the Recommission DataNode and Start NodeManager processes have completed.

Then click **OK** to dismiss the window.



10. Ambari updates the `/etc/hadoop/conf/dfs.exclude` and `/etc/hadoop/conf/yarn.exclude` files by removing the host name of the decommissioned DataNode and NodeManager.

    View the contents of these files now by opening and then using SSH to connect to Namenode and ResourceManager.

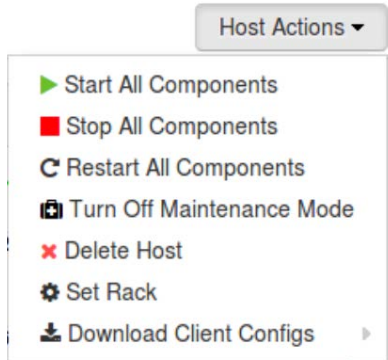
```
ssh <NameNode AWS Hostname>
more /etc/hadoop/conf/dfs.exclude

ssh <ResourceManager AWS Hostname>
more /etc/hadoop/conf/yarn.exclude
```

    The host you selected should not be listed in either file.

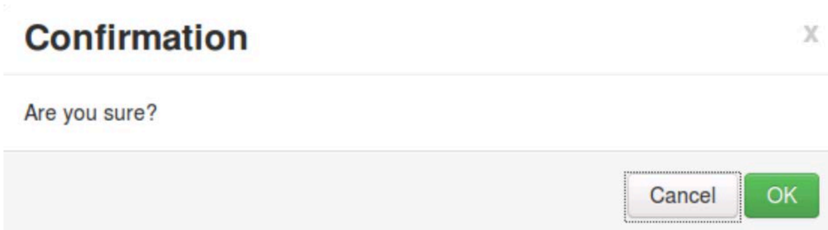
Turning Off Maintentance Mode

Turn off maintenance mode.

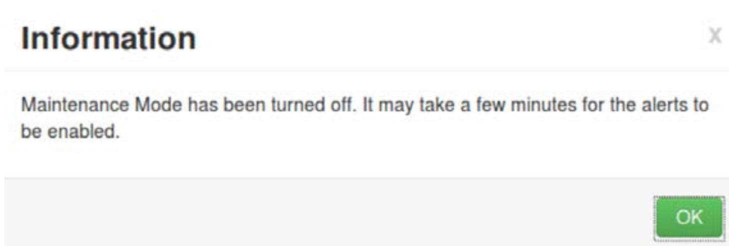
1. On the Hosts page in the Ambari Web UI,
   Click the **Host Actions** menu button and select **Turn Off Maintenance Mode**.



2. In the Confirmation window, click **OK**.



3. Read the information in the Information window and then click **OK**.

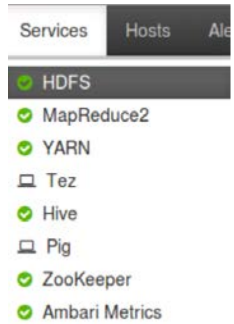


Changing the Replication Factor

Change the default HDFS data block replication factor using the Ambari Web UI.

Now that there are three DataNodes in the cluster, it makes sense for HDFS to automatically replicate each data block across all three DataNodes. This behavior makes HDFS more resilient to individual DataNode failures and can increase HDFS performance as well.

1. In the Ambari Web UI, click **Services** and then select **HDFS**.

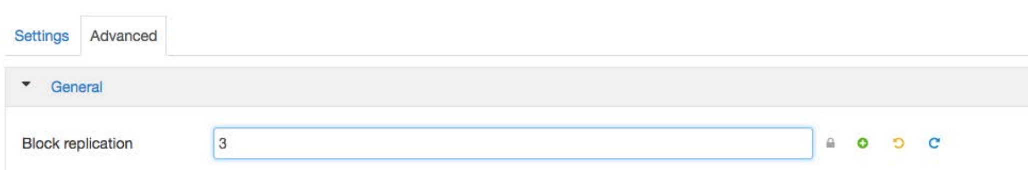


2 . Click the **Configs** page and then click **Advanced**.

Ensure that the **HDFS Default** configuration group is selected.

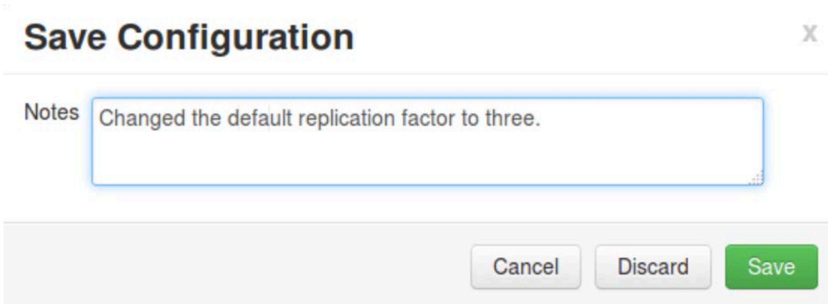
3 . Scroll down to the **General** section and find the **Block replication** property.

Change it from 1 to 3.
(It was changed from the default installation value of 3 *to* 1 *in a previous lab.)*

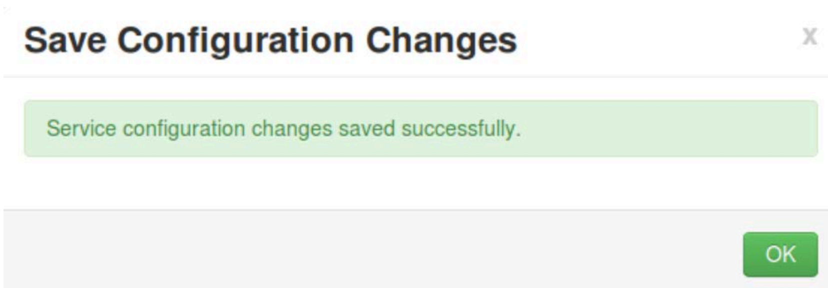


4 . Click **Save** to save the change to the HDFS configuration.

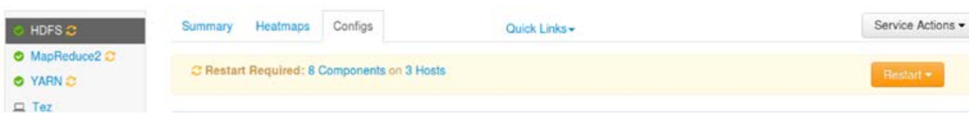
5 . Type an optional description of the change that you made and then click **Save** again.



6 . Click **OK** in the status window.



7 . Restart any services that require restarting as indicated in the Ambari Web UI.

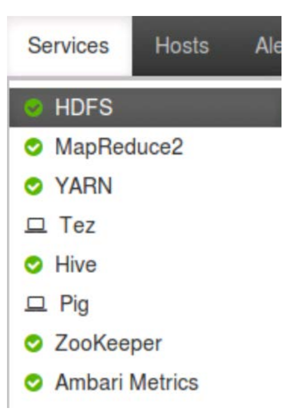


Rebalancing HDFS Storage

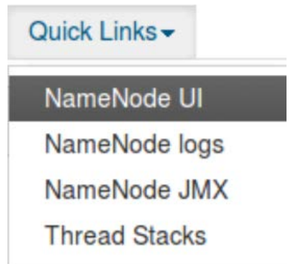
Check and rebalance HDFS storage.

Whenever you add more DataNodes, the new DataNodes are initially under-utilized because they do not contain any data blocks. Running the HDFS balancer will move data blocks to the new DataNodes.

1 . In the Ambari Web UI, click **Services** and select **HDFS**.



2 . Click **Quick Links** and select **NameNode UI**.



3 . Click the **DataNodes** tab in the NameNode UI.

4 . For the three DataNodes, compare the numbers in the **Used**, **Blocks**, and **Block pool used** columns.

Used	Non DFS Used	Remaining	Blocks	Block pool used
577.36 MB	26.44 GB	66.13 GB	170	577.36 MB (0.61%)
440.03 MB	26.57 GB	66.13 GB	26	440.03 MB (0.46%)
440.03 MB	26.57 GB	66.13 GB	26	440.03 MB (0.46%)

The more evenly matched the numbers are across the DataNodes, the more balanced the data blocks are across the DataNodes.

NOTE: There is very little data in HDFS in the current lab environment so any differences between DataNodes will be very small. However, we just added a new host to the cluster, which means that orginial three hosts might have slightly more data blocks.

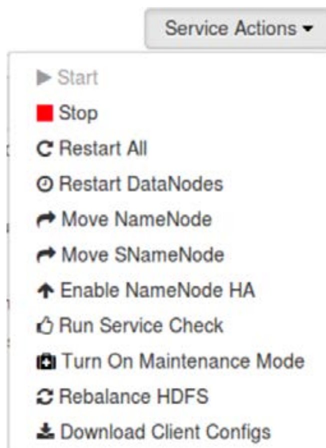
5 . **OPTIONAL**
Use SSH to login to any cluster node, become the HDFS superuser, and run the command *hdfs dfsadmin -report*.


```
su - hdfs
hdfs dfsadmin -report
```

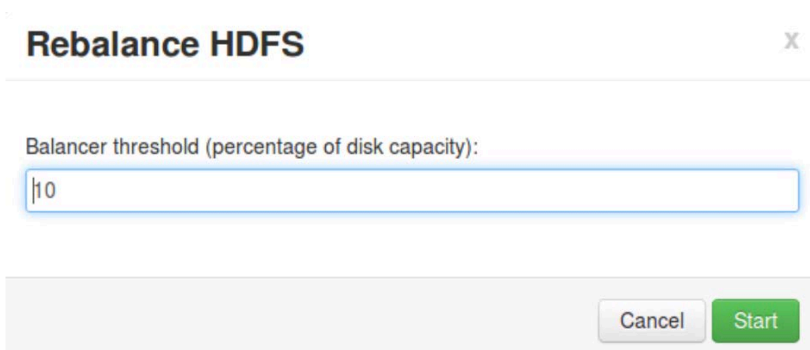
Compare the **DFS Used** and **DFS Remaining** values to determine HDFS balance.

6. In the Ambari Web UI, ensure that **Services** and **HDFS** are selected.

7. Click the **Service Actions** menu button and then select **Rebalance HDFS**.



8. In the Rebalance HDFS window, accept the default of ten percent and click **Start**.

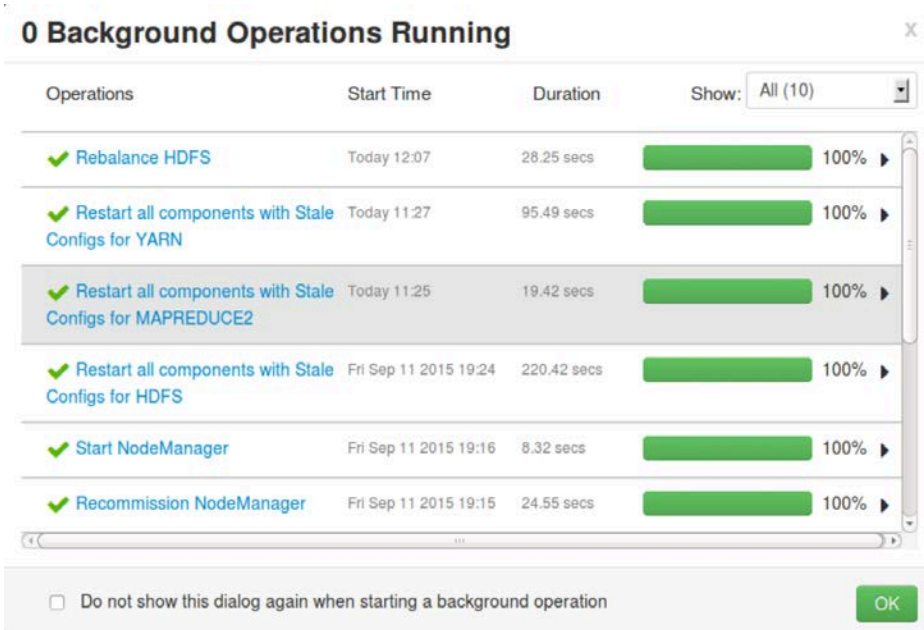


If the DataNodes are balanced to within ten percent the balancer program will not take any action other than just evaluating the DataNodes. No data blocks will be moved.

If the DataNodes are not balanced to within ten percent then the balancer program will shuffle data blocks between DataNodes.

9 . Monitor the Background Operations Running window until the Rebalance HDFS process has completed.

Then click **OK** to dismiss the window.



Because there is so little data, and the default replication factor was one when most of the HDFS files were created,
the balancer might not shuffle any data blocks.

Result

You have now:
Added one nodes to the cluster, decommissioned and recommissioned one of the nodes, turned off maintenance mode, and run the HDFS balancer.

Lab 25: Managing Ambari Alerts

About This Lab

Objective:	To use the Ambari Web UI to manage Ambari alerts
File locations:	N/A
Successful outcome:	**You will:** List alert definitions; disable, enable, and edit an alert definition; list alert instances and trigger an alert; and create a new alert group and notification.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Monitoring a Cluster***

Listing Alert Definitions

List Ambari alert definitions in the Ambari Web UI.

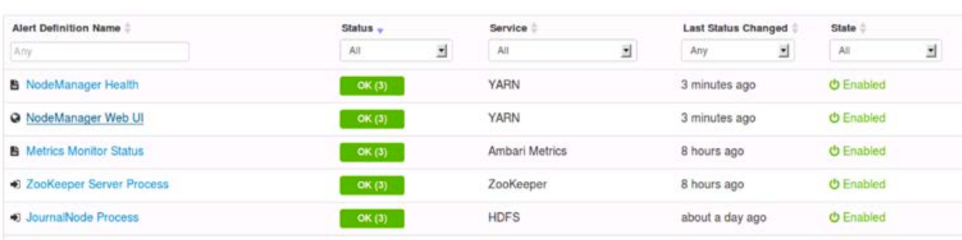
1 . Open the browser and connect to the Ambari Server at the URL

http://<Ambari Server Hostname>:8080

2 . Login to the Ambari Web UI using the user name `admin` and the password `BadPass#1`.

3 . Click **Alerts** in the Ambari Web UI.

4 . View the list of alert definitions on the **Alerts** page.



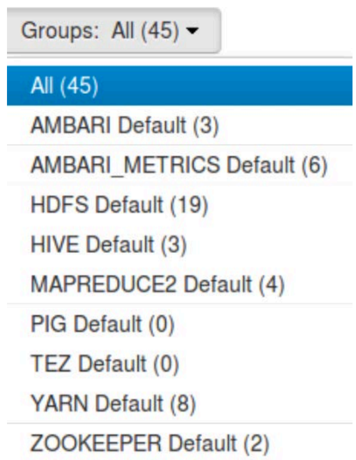
Alert Definition Name	Status	Service	Last Status Changed	State
Any	All	All	Any	All
NodeManager Health	OK (3)	YARN	3 minutes ago	Enabled
NodeManager Web UI	OK (3)	YARN	3 minutes ago	Enabled
Metrics Monitor Status	OK (3)	Ambari Metrics	8 hours ago	Enabled
ZooKeeper Server Process	OK (3)	ZooKeeper	8 hours ago	Enabled
JournalNode Process	OK (3)	HDFS	about a day ago	Enabled
DataNode Process	OK (3)	HDFS	about a day ago	Enabled

How many alert definitions are there?

The number should be shown in the lower right-hand corner of the Alerts page.

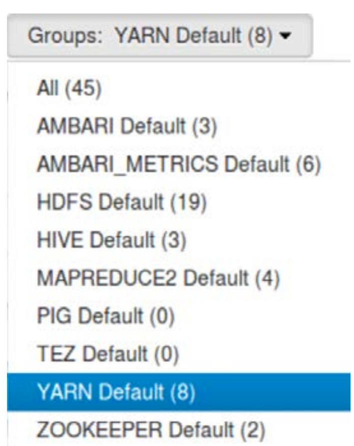
5. Use the forward and back arrows beneath the alert definitions to view the entire list of alert definitions.

6. Filter the list of displayed alert definitions by clicking the **Groups** menu button and selecting **YARN Default (8)**.



Only the eight YARN-related alert definitions are displayed.

7. Click the **Groups** menu button again and select **All**.



All alert definitions are listed again.

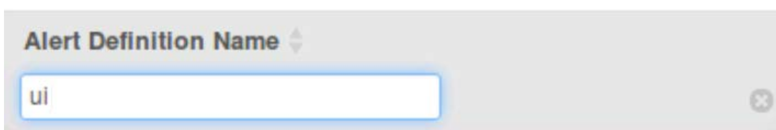
8 . Select **CRITICAL** from the **Status** drop-down menu.

Only alerts with a CRITICAL status are displayed. There might not be any alerts with this status.

9 . Click the **X** icon next to the **Status** drop-down menu in order to clear the status filter.

All alert definitions are displayed again.

10 . Type `ui` in the **Alert Definition Name** text box to filter the list to only those alert definitions that contain `ui`.



The alert definition list should contain only definitions with UI or ui in their name.

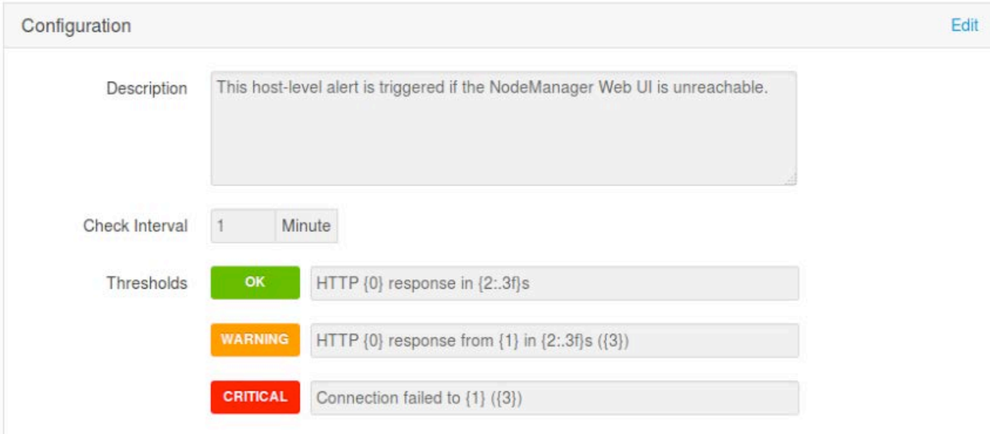
Do not remove `ui` from the Alert Definition Name filter text box.

Working With Alert Definitions

View, disable, enable, and edit an alert definition in the Ambari Web UI.

1 . Click the **NodeManager Web UI** alert definition.

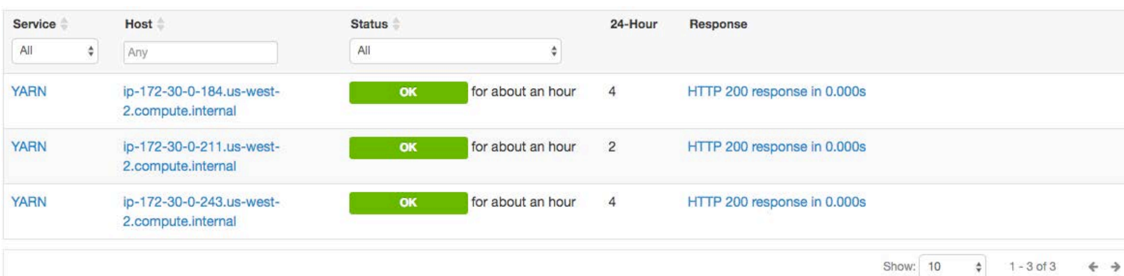
2 . View the alert definition in the Configuration panel.



Notice the Check Interval.
Notice the Thresholds response messages.

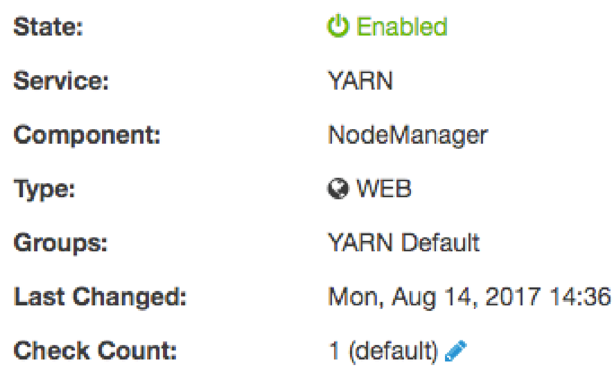
You will see examples of the response messages in other lab steps.
Response messages are editable.

3 . If necessary, scroll down in the browser window to view the alert instances that were created based on the alert definition.

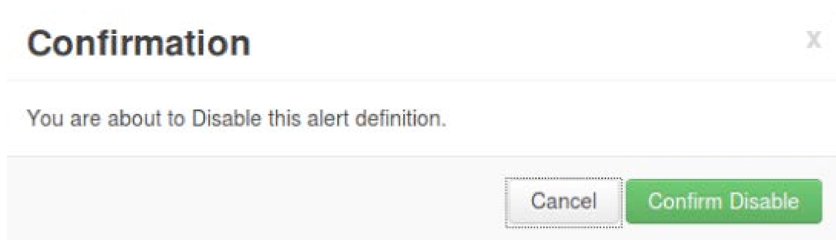


Notice an example of a response message in the response column.
This is the response message for the **OK** status.

4 . If necessary, scroll up in the browser window and click **Enabled**.
The **Enabled** link functions as a toggle and will disable the alert definition.

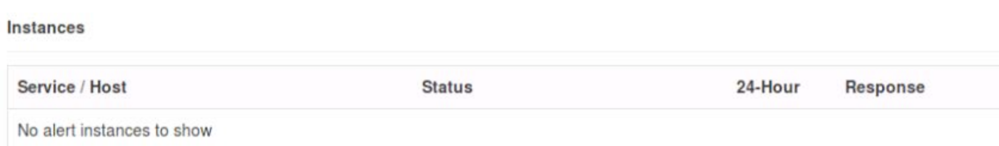


5 . In the Confirmation window, click **Confirm Disable**.

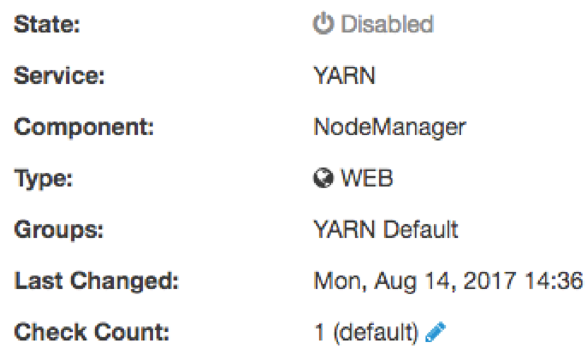


6 . Wait for a minute and then refresh the browser window.
Then scroll down, if necessary, and view the alert instances again.

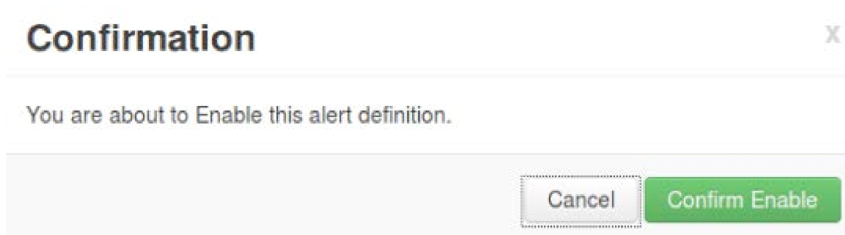
Are there any instances?
All instances should have been deleted.



7 . Click **Disabled** to re-enable the alert definition.

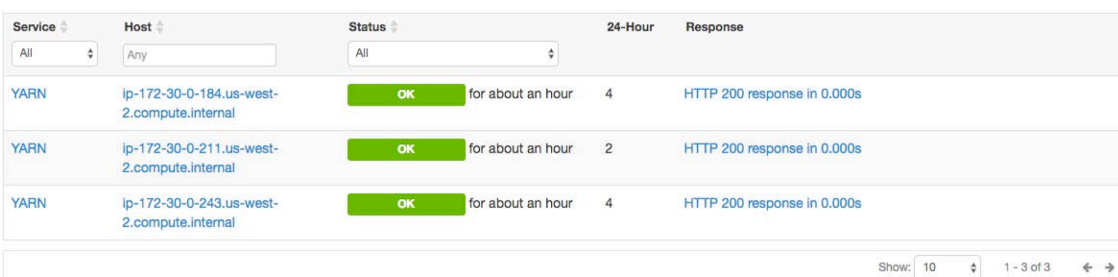


8 . In the Confirmation window, click **Confirm Enable**.



9 . Wait a minute or two and then refresh the browser window.
Then scroll down, if necessary, and view the alert instances again.

Are there any instances?
There should be three instances.



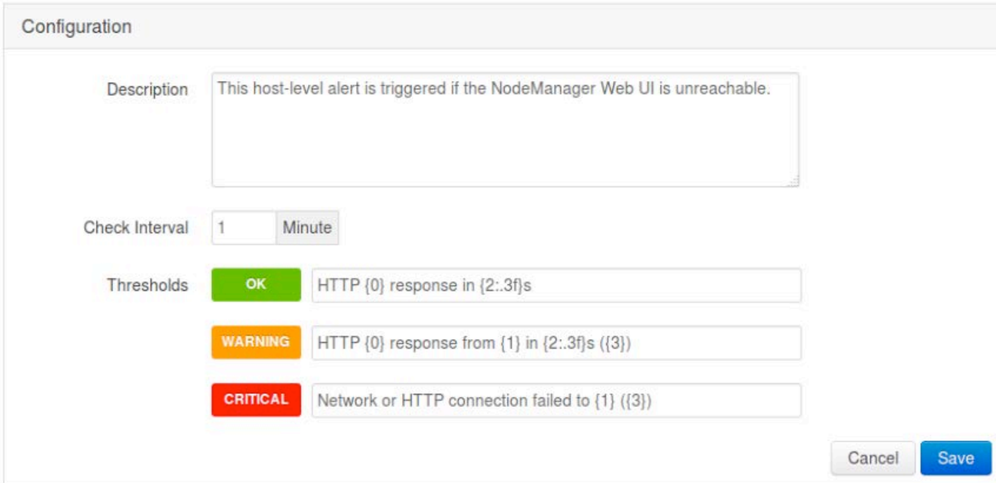
Service	Host	Status		24-Hour	Response
All	Any	All			
YARN	ip-172-30-0-184.us-west-2.compute.internal	OK	for about an hour	4	HTTP 200 response in 0.000s
YARN	ip-172-30-0-211.us-west-2.compute.internal	OK	for about an hour	2	HTTP 200 response in 0.000s
YARN	ip-172-30-0-243.us-west-2.compute.internal	OK	for about an hour	4	HTTP 200 response in 0.000s

Show: 10 1 - 3 of 3

10 . Click **Edit** to change the NodeManager Web UI alert definition.

11 . Change the CRITICAL response message to `Network or HTTP connection failed to {1} ({3})`.

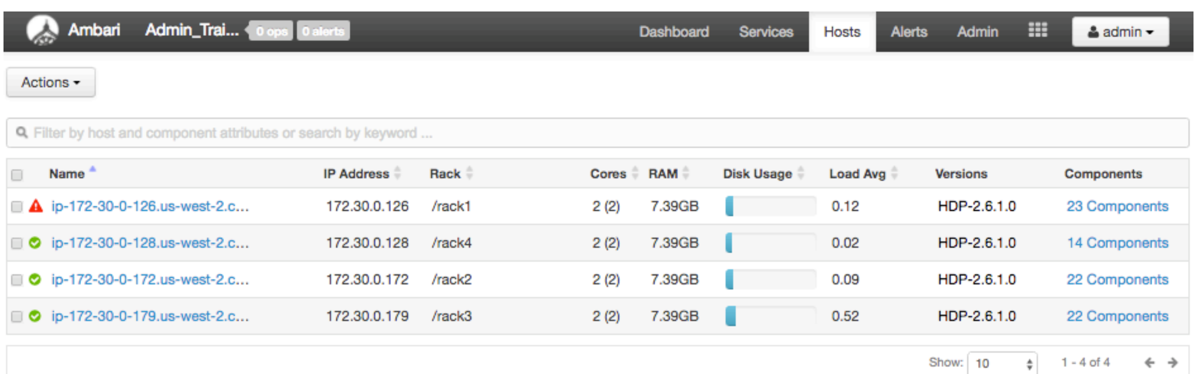
Then click **Save**.



Triggering an Alert

Trigger and view an alert for the NodeManager Web UI.

1 . In the Ambari Web UI, click **Hosts** to open the **Hosts** page.

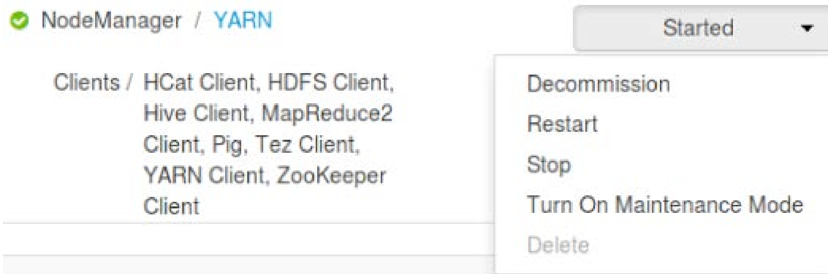


2 . Click on any Name in the list of cluster nodes.

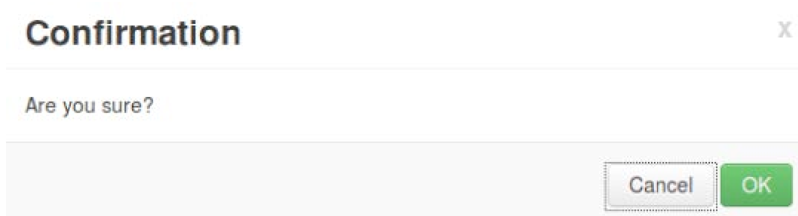
3 . On the **Summary** tab, find the **NodeManager** in the list of components.

Click the **Started** menu button and select **Stop**.

Stopping the NodeManager on this host should trigger the NodeManager Web UI alert, as well as other alerts.

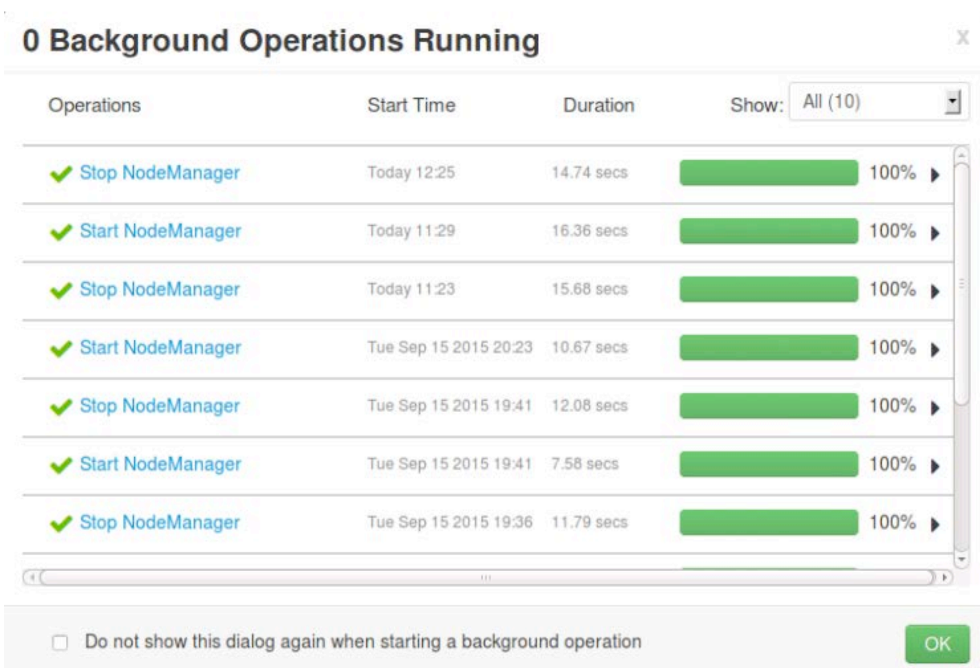


4 . In the Confirmation window, click **OK**.



5 . In the Background Operations Running window, monitor the Stop NodeManager process.

When it has finished, click **OK**.



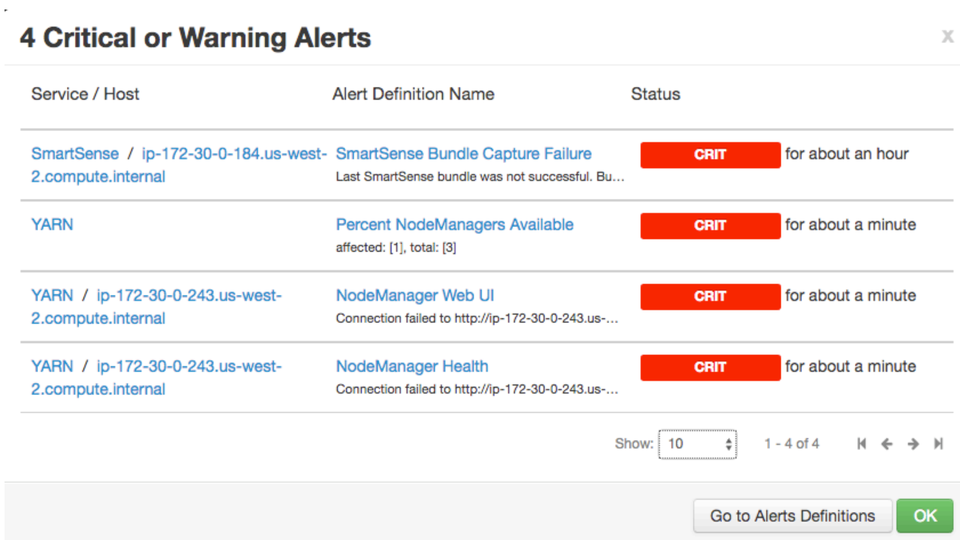
6 . Within a minute you should see alerts triggered.

Click on the red alerts notification at the top of the Ambari Web UI.

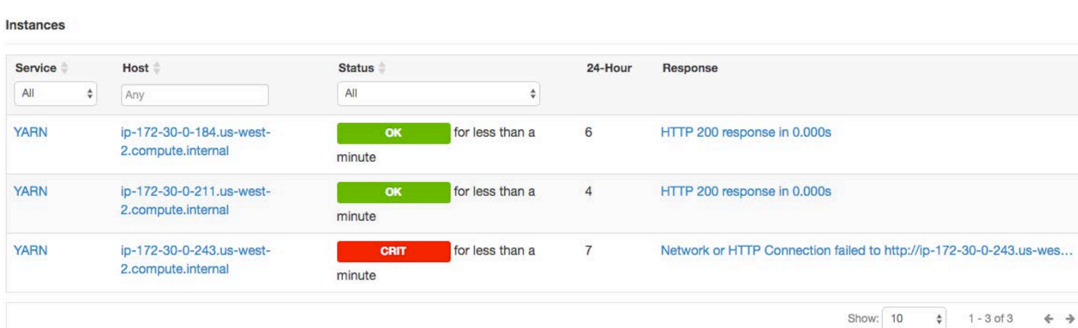
The Critical or Warning Alerts window opens showing all the current alerts.

7 . Notice that the NodeManager Web UI alert has been triggered. Also notice the custom response message.

Click the **NodeManager Web UI** link to open the alert definition and alert instance list.



8 . The alert instance that triggered is shown in the list of instances.



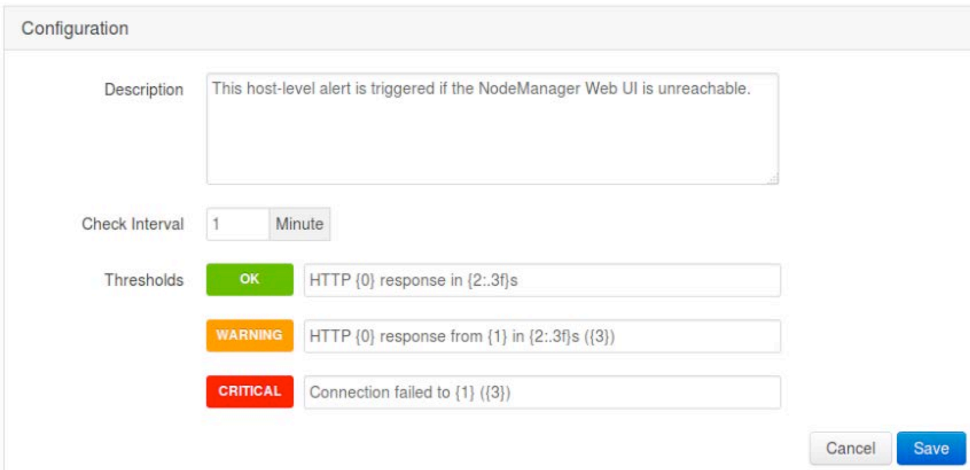
In the screen capture, the alert had been triggered for few minutes.

The custom response message is also displayed. "Network or HTTP Connection failed to..."

9 . Click **Edit** to change the NodeManager Web UI alert definition.

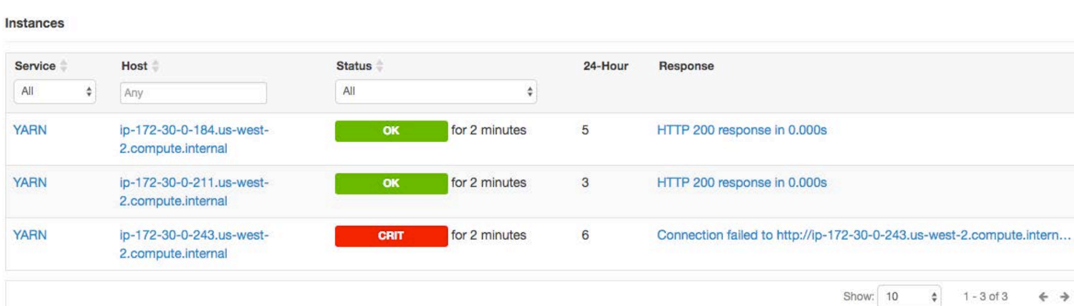
10 . Change the CRITICAL response message back to `Connection failed to {1} ({3})`.

Then click **Save**.



11 . Wait a minute and then look at the response message in the triggered alert.
Did it change?
It should have. It changed because the alert is re-checked every minute.

While the response message is updated, any configured email or SNMP notifications are not resent every minute. Notifications are only sent when the status changes.

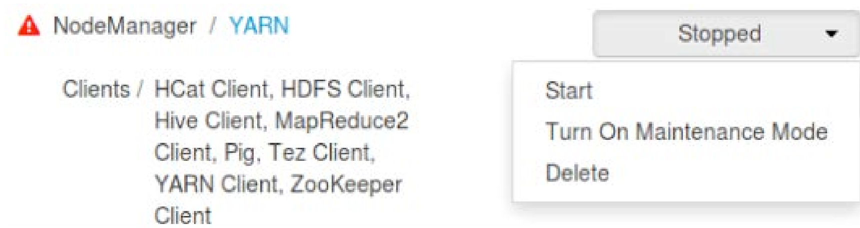


Click on the host that NodeManger has been stopped (not **YARN**) in the list of alert instances.

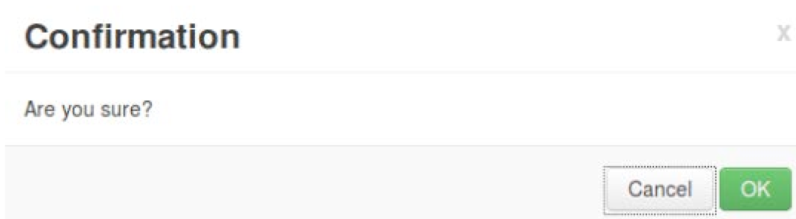
13 . Click the **Summary** tab.

14 . Find the NodeManager in the list of components.
Click the **Stopped** menu button and select **Start**.

Starting the NodeManager on the selected host should remove the NodeManager Web UI alert, as well as other alerts.

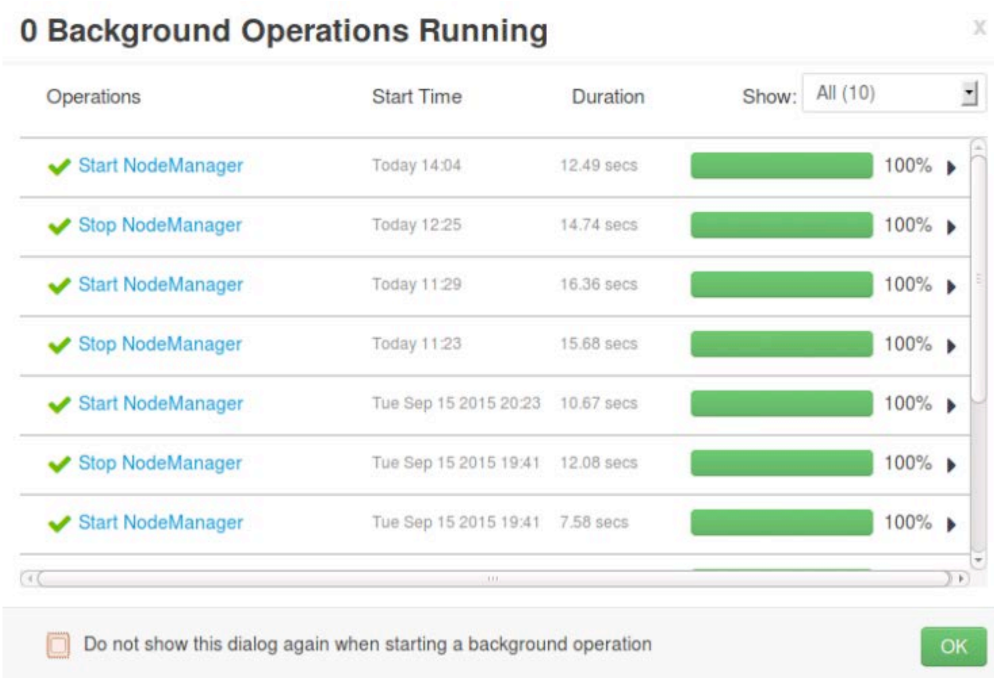


15 . In the Confirmation window, click **OK**.



16 . In the Background Operations Running window, monitor the Start NodeManager process.

When it has finished, click **OK**.



Wait one minute, refresh the browser, and the alerts should be gone.

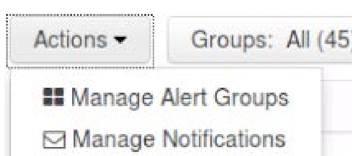
Creating Alert Groups and Notifications

Create an alert group and a notification.

1. In the Ambari Web UI, click **Alerts**.

2. Click the **Actions** menu button and select **Manage Alert Groups**.

   This initiates the process of creating a new alert group.



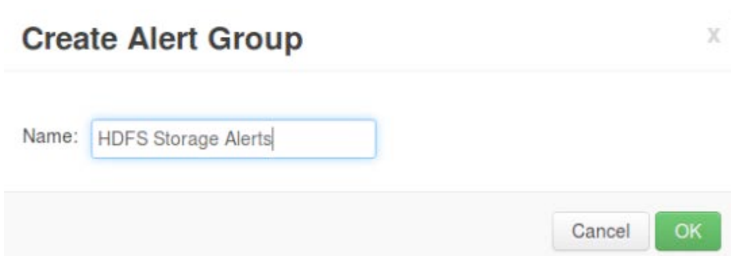
   The Manage Alert Groups window opens.

3 . Click the plus (**+**) button beneath the list of alert groups.

The Create Alert Group window opens.

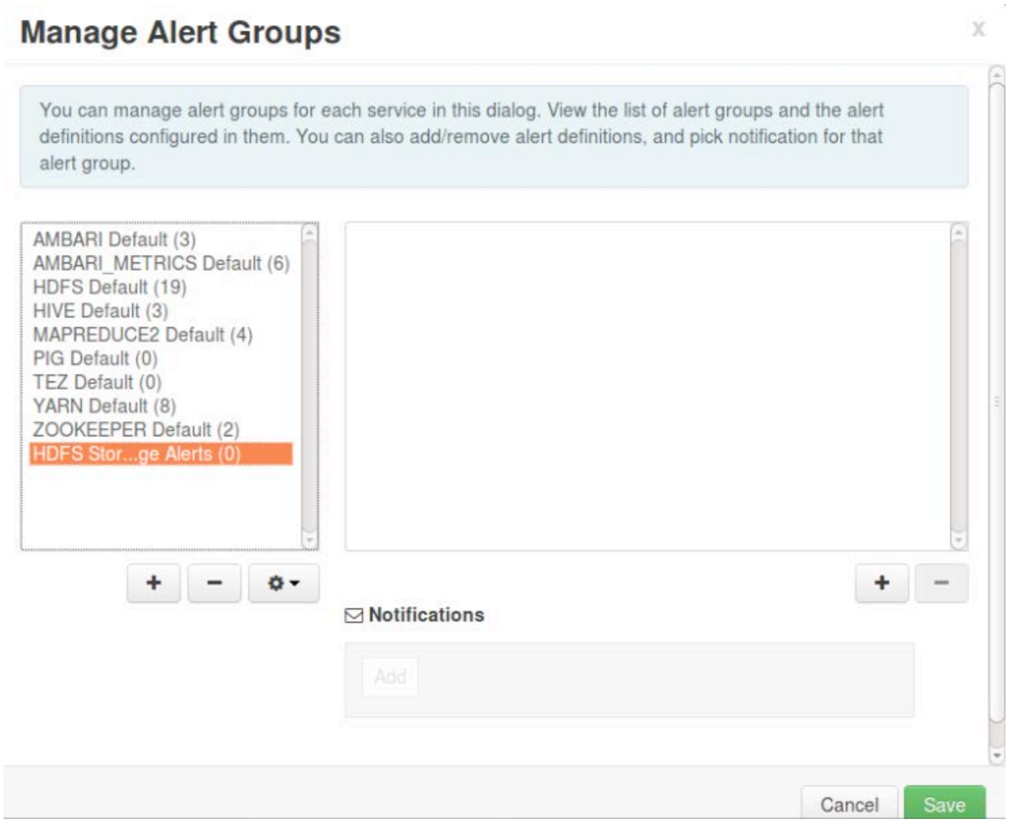
4 . In the Create Alert Group window, type the name `HDFS Storage Alerts.`

Click **OK**.



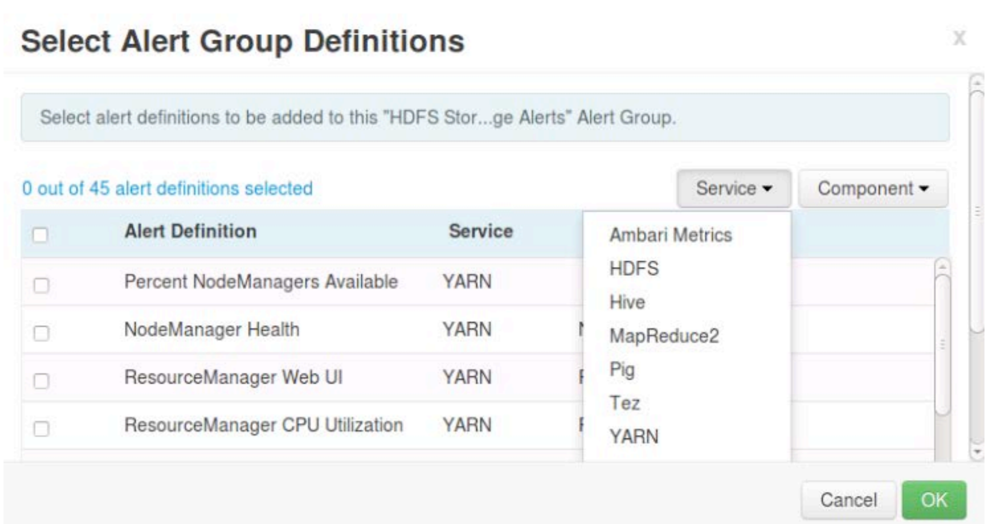
The new alert group appears in the list of alert groups in the Manage Alert Groups window.

5 . Ensure that the **HDFS Storage Alerts** group is selected, and then click the plus (**+**) button on the right side of the window.



The Select Alert Group Definitions window opens.

6 . In the Select Alert Group Definitions window, click the **Service** menu button and select **HDFS**.

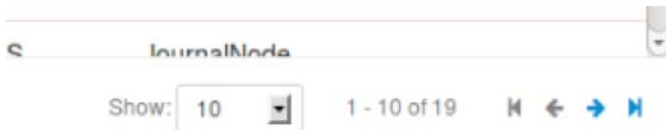


The list of Alert Definitions is filtered to only those that relate to HDFS. The items on the **Service** and **Component** menu buttons can be selected and deselected. Clicking an item, like **HDFS**, selects the item. Clicking the item again deselects the item.

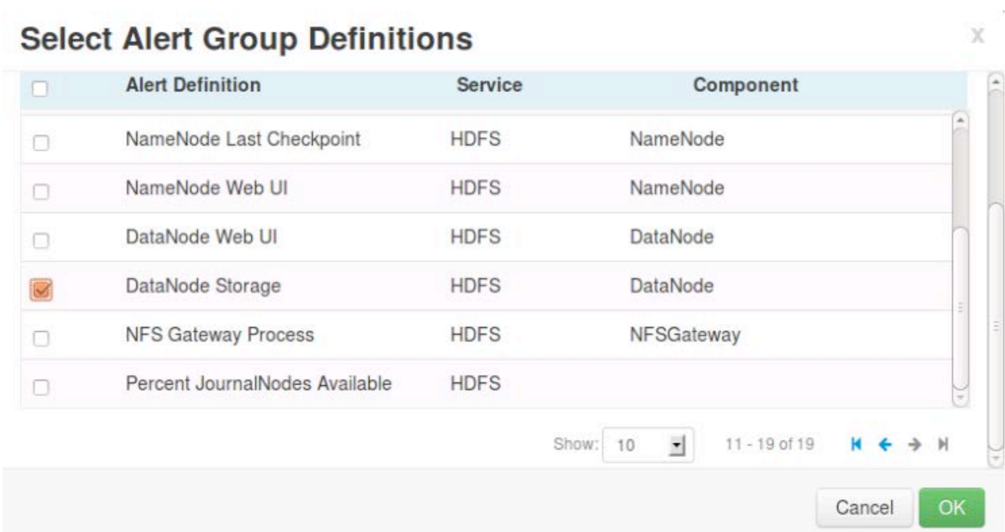
Leave **HDFS** selected.

7 . From the list of HDFS alert definitions, select **Percent DataNodes With Available Space**, **Percent DataNodes Available**, **HDFS Capacity Utilization**, and **DataNode Storage**.

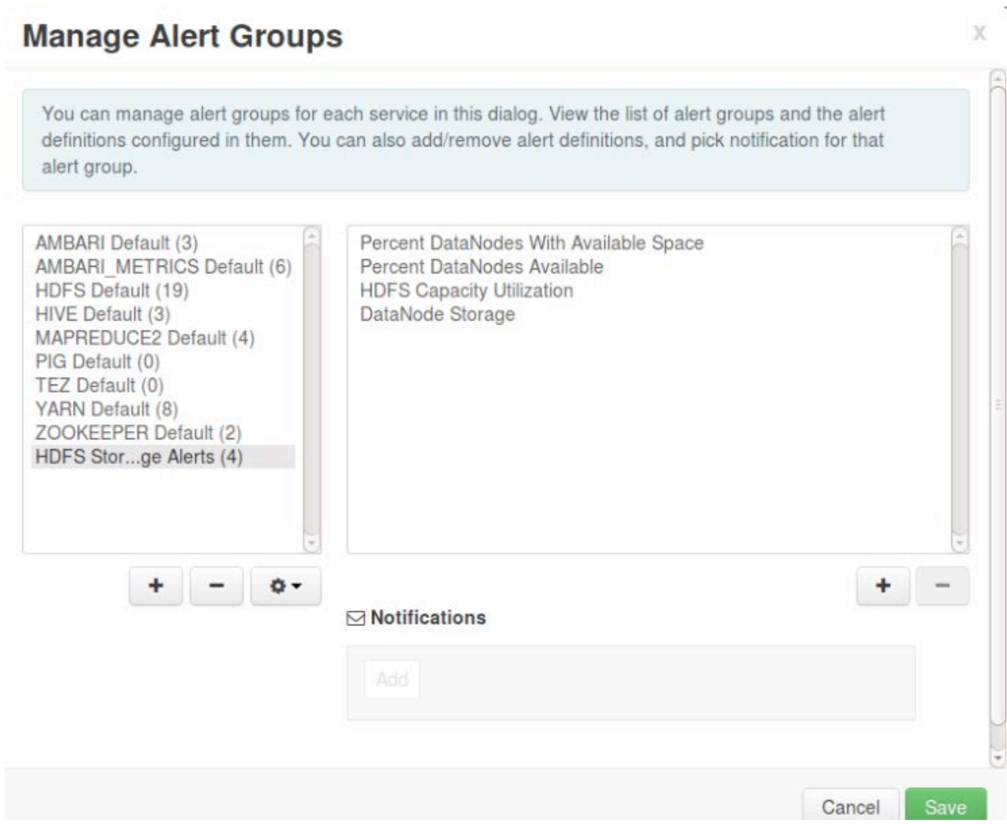
NOTE: By default, only the first 10 alert definitions are visible, so you will need to go to the next page to see all available settings.



Once these four alerts have been selected, click **OK**.



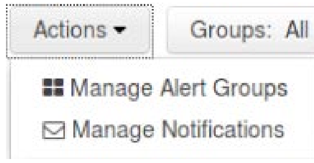
8 . The four alerts are listed in the Manage Alert Groups window.
Click **Save** to save the new alert group.



9 . View the result window and click **OK**.

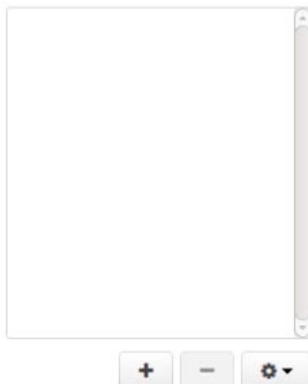
10 . Click the **Actions** menu button and select **Manage Notifications**.

This initiates the process of creating a new notification.



The Manage Alert Notifications window opens.

11 . Click the plus (**+**) button.

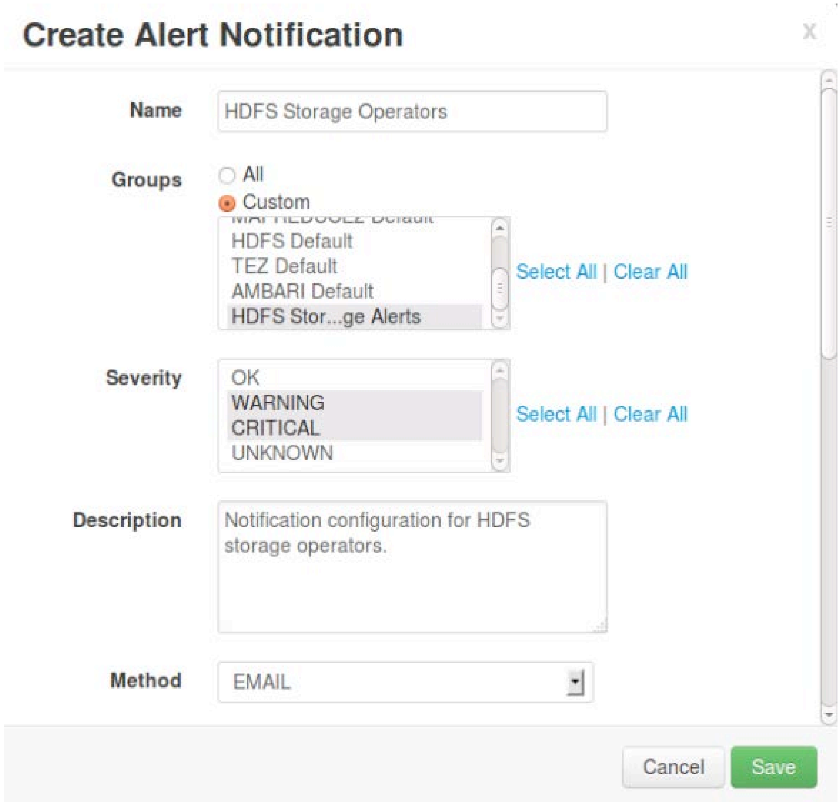


The Create Alert Notification window opens.

12 . In the Create Alert Notification window, make the following selections.

Name: `HDFS Storage Operators`
Groups: Select the **Custom** radio button and then select **HDFS Storage Alerts**
Severity: **WARNING** and **CRITICAL** (use the Shift key to select multiple levels)
Description: `Notification configuration for HDFS storage operators.`
Method: **EMAIL**
Email To: `hdfsstorageops@somecompany.com`
SMTP Server: `smtp.somecompany.com`
SMTP Port: `25`
Email From: `ambarialerts@somecompany.com`
Use authentication: select it
Username: `admin`
Password: `cangetin`
Password Confirmation: `cangetin`
Start TLS: select it

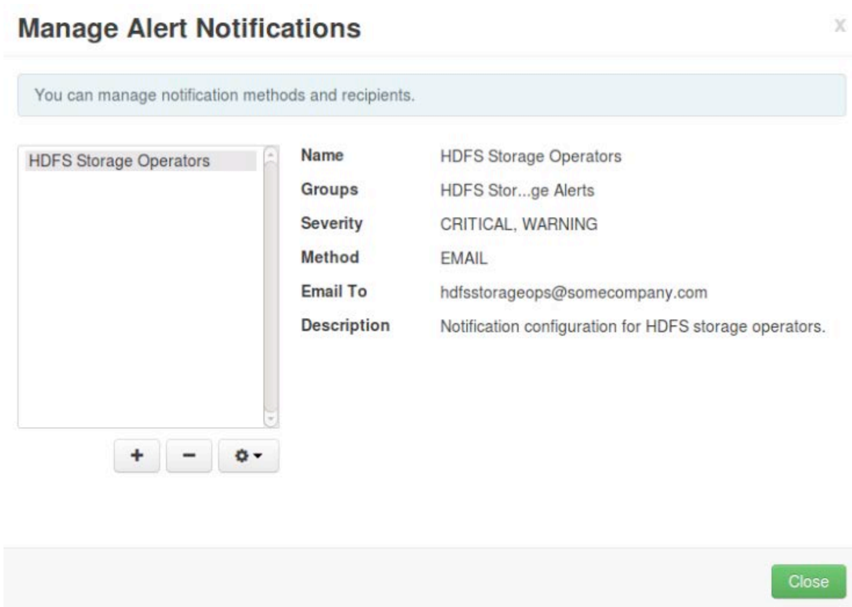
Then click *Save* to create the notification.



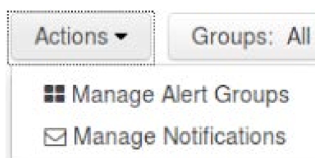
No email server is available in the lab environment so no actual email will be sent.

13 . The result is displayed in the Manage Alert Notifications window.

Click **Close**.



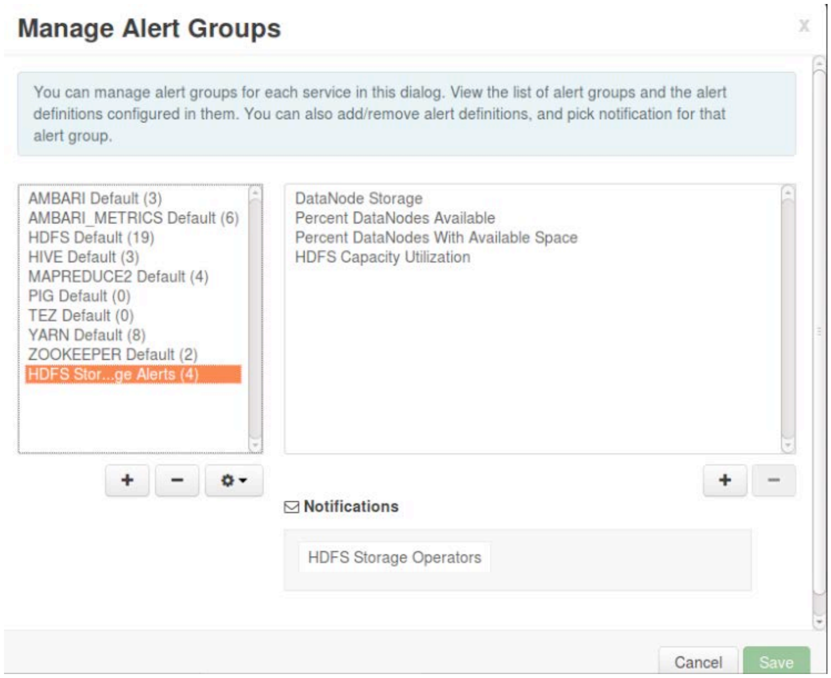
14 . Click the **Actions** menu button again and select **Manage Alert Groups**.



15 . Select the **HDFS Storage Alerts** group.

Notice that the **HDFS Storage Operators** notification has been assigned to the alert group.

There is nothing to change so click **Cancel**.



Result

You have now:
Listed alert definitions; disabled, enabled, and edited an alert definition; listed alert instances and triggered an alert; and created a new alert group and notification.

Lab 26: Deploy a HDP Cluster using Ambari Blueprints

About This Lab

Objective:	Provision a cluster using Ambari Blueprints
Successful outcome:	Ambari Blueprints will be used to install and scale out a multi-node cluster
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Ambari Blueprints***

Lab Steps

Perform the following steps:

1. Use SSH to login to the your Ambari Server.

2. Download the Blueprint and Cluster Template files on to your Ambari Server.

```
wget
https://raw.githubusercontent.com/HortonworksUniversity/Ops_Labs/master/labfile
s/Admin_Core/MultinodeHDP2.5.blueprint
wget
https://raw.githubusercontent.com/HortonworksUniversity/Ops_Labs/master/labfile
s/Admin_Core/ThreeNodeCluster.install
wget
https://raw.githubusercontent.com/HortonworksUniversity/Ops_Labs/master/labfile
s/Admin_Core/AddOneHosts.install
```

3. List the directory to confirm files.

```
[centos@ip-172-30-0-34 ~]$ ls -l
total 264
-rw-rw-r--. 1 centos centos    152 Aug 14 19:24 AddOneHosts.install
-rw-rw-r--. 1 centos centos 258882 Aug 14 19:23 MultinodeHDP2.5.blueprint
-rw-rw-r--. 1 centos centos    450 Aug 14 19:24 ThreeNodeCluster.install
[centos@ip-172-30-0-34 ~]$
```

Register blueprints with Ambari Server.

a. These are the blueprint files we will upload and register via API calls to install and scale HDP.

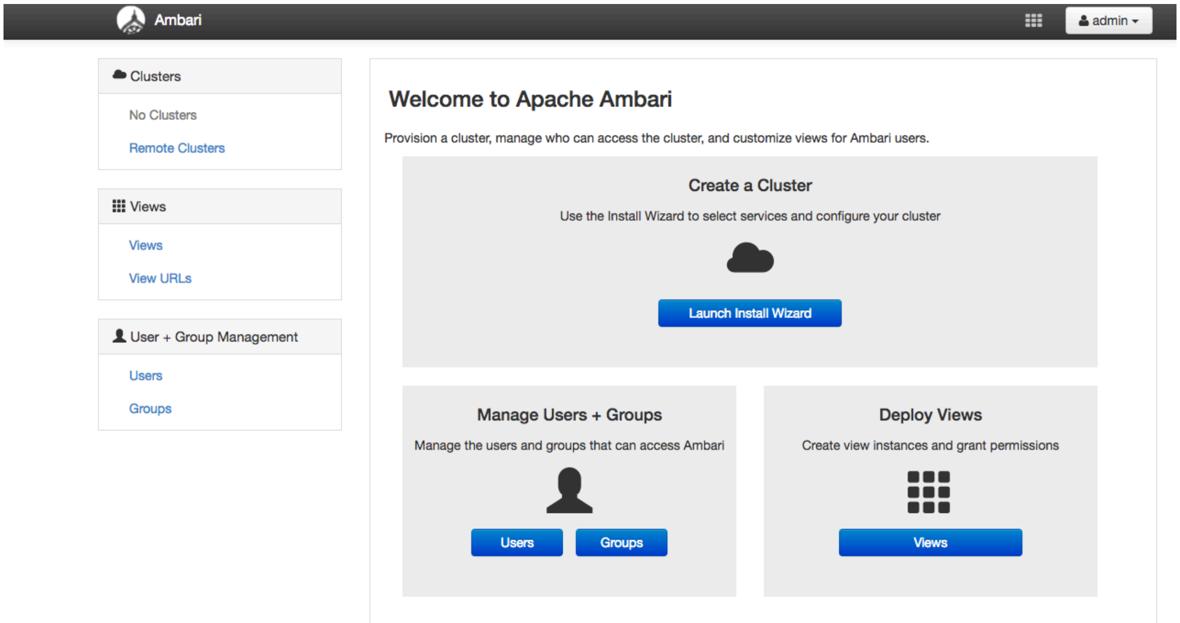
b. Use cURL to issue an API call to register the Multinode.blueprint file with your newly-installed Ambari Server instance.

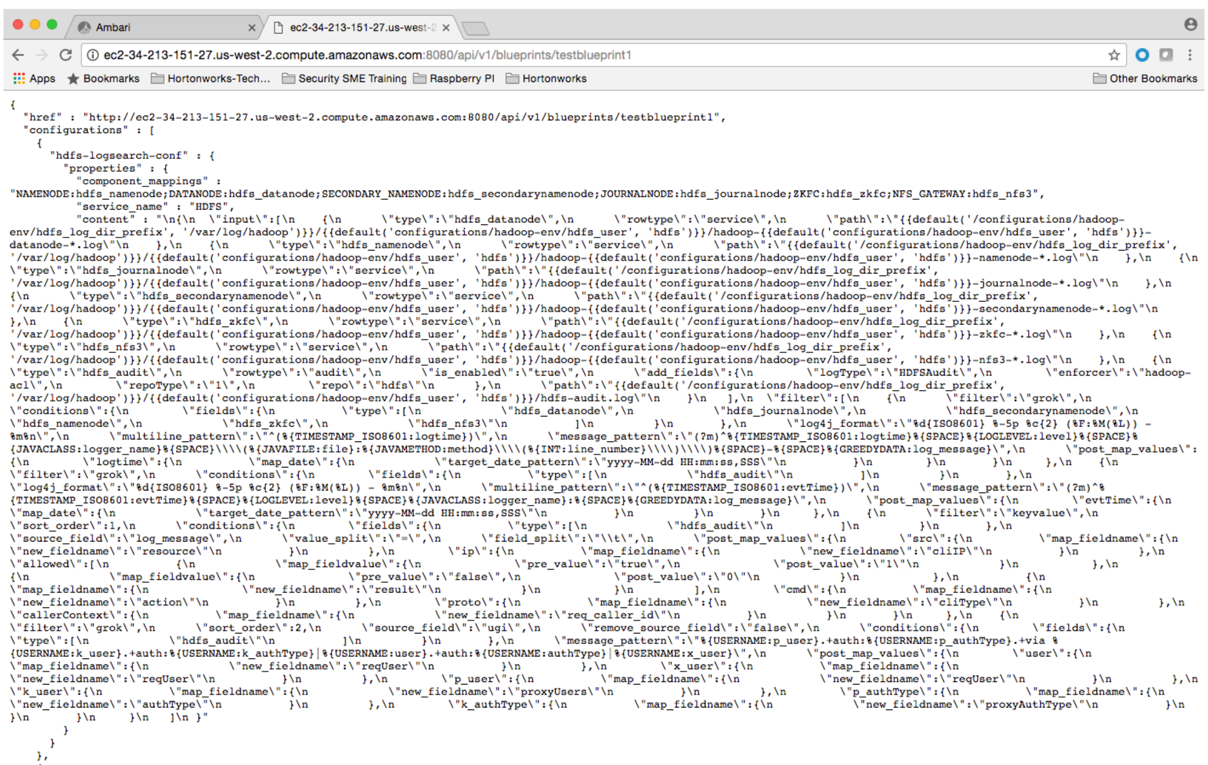
```
curl -u admin:BadPass#1 -i -H "X-Requested-By: root" -X POST -d
@MultinodeHDP2.5.blueprint
http://localhost:8080/api/v1/blueprints/testblueprint1
```

```
[centos@ip-172-30-0-110 ~]$ curl -u admin:BadPass#1 -i -H "X-Requested-By: root" -X POST -d @MultinodeHDP2
.4.blueprint http://node1:8080/api/v1/blueprints/testblueprint1
HTTP/1.1 100 Continue

HTTP/1.1 201 Created
X-Frame-Options: DENY
X-XSS-Protection: 1; mode=block
Set-Cookie: AMBARISESSIONID=1cdmhxg2vkesg1k417kajcvu8x;Path=/;HttpOnly
Expires: Thu, 01 Jan 1970 00:00:00 GMT
User: admin
Content-Type: text/plain
Content-Length: 0
Server: Jetty(8.1.19.v20160209)
```

c. Verify that the blueprint is registered by opening up a web browser and logging in to Ambari at http://<Your Ambari Node IP Address>:8080 with the username admin and password of BadPass#1in.
Then open a new tab, and go to

```
http://<Your Ambari Node IP Address>:8080/api/v1/blueprints/testblueprint1
```





This blueprint can now be used to install HDP.

4. **Install and scale an HDP cluster using Ambari Blueprints.**

   a. The `ThreeNodeCluster.install` file contains instructions to install an HDP cluster using a blueprint named testblueprint1 on a server with an `FQDN.` Edit this file and replace node1, node2 and Node3 with your AWS Internal Hostname provided by your instructor. WARNING: DO NOT USE THE AWS EXTERNAL HOSTNAMES.

```
{
  "blueprint" : "testblueprint1",
  "default_password" : "BadPass#1",
  "configurations" : [
    { "ams-grafana-env" : {
      "metrics_grafana_password" : "BadPass@1"
      }
    }
  ],
  "host_groups" :[
    { "name" : "host_group_1",
      "hosts" :[ { "fqdn" : "node1"} ]
      },
    { "name" : "host_group_2",
      "hosts" :[ { "fqdn" : "node2" } ]
      },
    { "name" : "host_group_3",
      "hosts" :[ { "fqdn" : "node3" } ]
    }
  ]
}
~
~
~
~
~
~
~
~
~
~
~
~
"ThreeNodeCluster.install" 21L, 450C
```

The hostname names "node1, node2, node3" have been replaced with the AWS Internal Hostnames provided by your instructor. See next screen shot.

```
{
  "blueprint" : "testblueprint1",
  "default_password" : "BadPass#1",
  "configurations" : [
    { "ams-grafana-env" : {
      "metrics_grafana_password" : "BadPass@1"
      }
    }
  ],
  "host_groups" :[
    { "name" : "host_group_1",
      "hosts" :[ { "fqdn" : "ip-172-30-0-34.us-west-2.compute.internal"} ]
      },
    { "name" : "host_group_2",
      "hosts" :[ { "fqdn" : "ip-172-30-0-180.us-west-2.compute.internal" } ]
      },
    { "name" : "host_group_3",
      "hosts" :[ { "fqdn" : "ip-172-30-0-96.us-west-2.compute.internal" } ]
    }
  ]
}
~
~
~
~
~
~
~
~
~
~
~
~
"ThreeNodeCluster.install" 21L, 559C written
```

When you are ready, use cURL to issue an API call to install the HDP cluster using the instructions in this file. You can name your cluster anything, the example below has named the cluster "Admin_Core_Training".

```
curl -u admin:BadPass#1 -i -H "X-Requested-By: root" -X POST -d
@ThreeNodeCluster.install
http://localhost:8080/api/v1/clusters/Admin_Core_Training
```

```
[centos@ip-172-30-0-34 ~]$ curl -u admin:BadPass#1 -i -H "X-Requested-By: root" -X POST -d @ThreeNode
Cluster.install http://localhost:8080/api/v1/clusters/Admin_Core_Training
HTTP/1.1 202 Accepted
X-Frame-Options: DENY
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
Cache-Control: no-store
Pragma: no-cache
Set-Cookie: AMBARISESSIONID=ypt41p6q04wr1tzpmlziyq62x;Path=/;HttpOnly
Expires: Thu, 01 Jan 1970 00:00:00 GMT
User: admin
Content-Type: text/plain
Vary: Accept-Encoding, User-Agent
Content-Length: 147

{
  "href" : "http://localhost:8080/api/v1/clusters/Admin_Core_Training/requests/1",
  "Requests" : {
    "id" : 1,
    "status" : "Accepted"
  }
}[centos@ip-172-30-0-34 ~]$
```

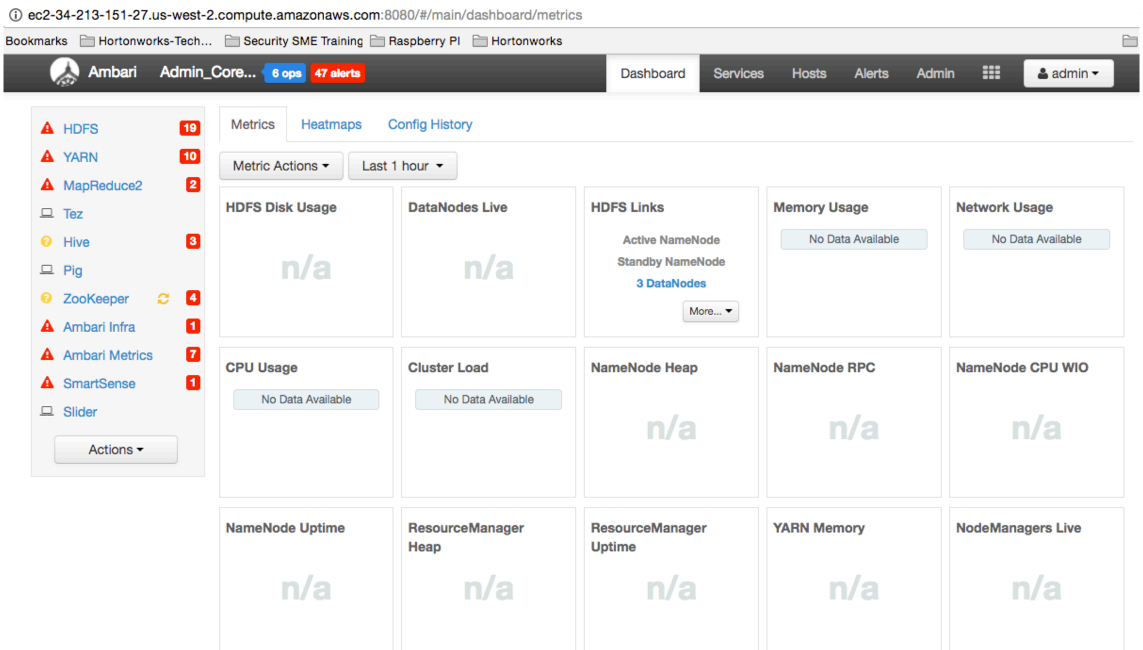
b. Verify that the cluster installation was accepted by opening up a browser tab to `http://<AWS External Ambari Hostname>:8080/api/v1/clusters/<YOUR CLUSTER NAME>`.

```
ec2-34-213-151-27.us-west-2.compute.amazonaws.com:8080/api/v1/clusters/Admin_Core_Training
Apps  Bookmarks  Hortonworks-Tech...  Security SME Training  Raspberry PI  Hortonworks
```

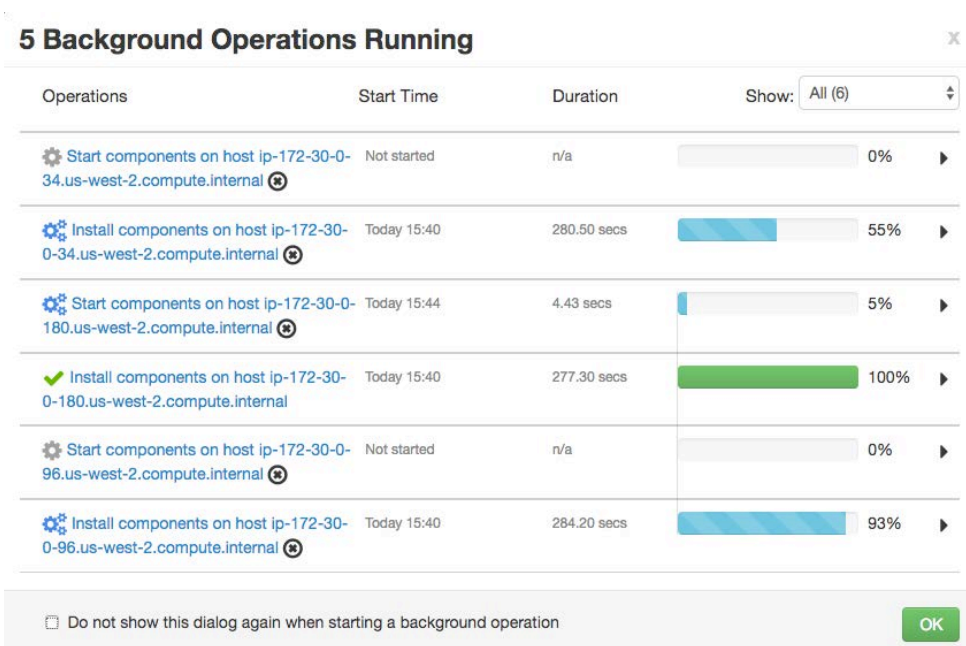

```
{
  "href" : "http://ec2-34-213-151-27.us-west-2.compute.amazonaws.com:8080/api/v1/clusters/Admin_Core_Training",
  "Clusters" : {
    "cluster_id" : 2,
    "cluster_name" : "Admin_Core_Training",
    "health_report" : {
      "Host/stale_config" : 2,
      "Host/maintenance_state" : 0,
      "Host/host_state/HEALTHY" : 3,
      "Host/host_state/UNHEALTHY" : 0,
      "Host/host_state/HEARTBEAT_LOST" : 0,
      "Host/host_state/INIT" : 0,
      "Host/host_status/HEALTHY" : 0,
      "Host/host_status/UNHEALTHY" : 3,
      "Host/host_status/UNKNOWN" : 0,
      "Host/host_status/ALERT" : 0
    },
    "provisioning_state" : "INSTALLED",
    "security_type" : "NONE",
    "total_hosts" : 3,
    "version" : "HDP-2.5",
    "credential_store_properties" : {
      "storage.persistent" : "false",
      "storage.temporary" : "true"
    },
    "desired_configs" : {
      "activity-conf" : {
        "tag" : "INITIAL",
        "user" : "admin",
        "version" : 1
      },
      "activity-env" : {
        "tag" : "INITIAL",
        "user" : "admin",
        "version" : 1
      },
      "activity-log4j" : {
        "tag" : "INITIAL",
        "user" : "admin",
        "version" : 1
      },
      "activity-zeppelin-env" : {
        "tag" : "INITIAL",
        "user" : "admin",
        "version" : 1
      },
```


c. Open a new browser tab and login to the Ambari Web UI again by going to `http://<Your Ambari Node IP Address>:8080` and providing the username `admin` and password of `BadPass#1`.

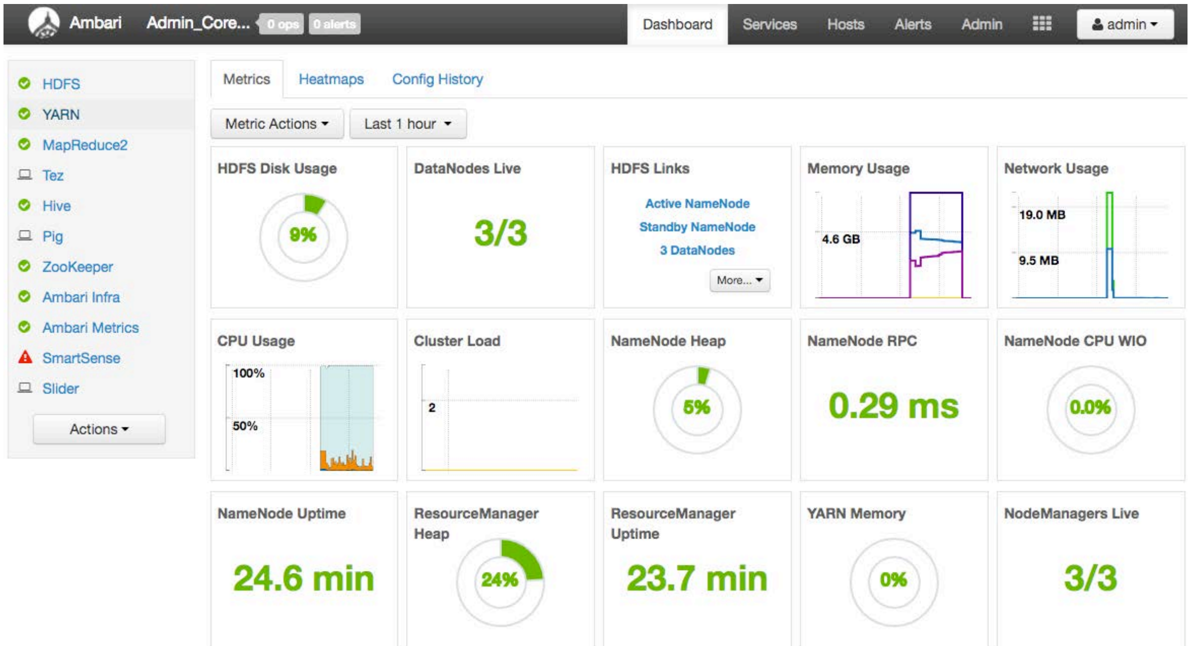
NOTE: That Ambari is probably still in the process of installing HDP according to the blueprint specifications.

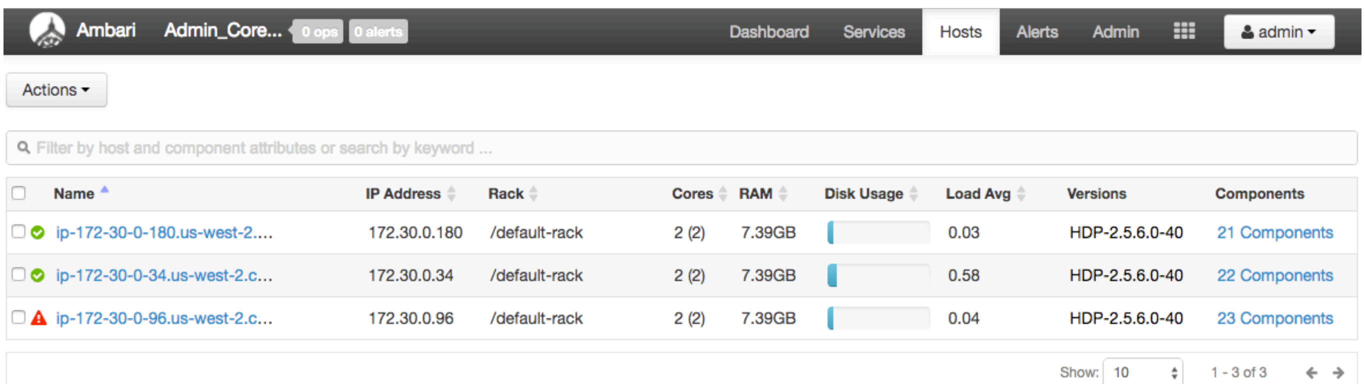


d. Click on the blue ops link at the top-left corner and watch as the installation is completed.



e. When the operations complete, click **OK**.
You should note that HDP has been installed and is running.
Click on the **hosts** tab and verify that three nodes are listed.





f. The `AddOneHosts.install` file contains instructions to install one HDP nodes using a blueprint named `testblueprint1`, but rather than look for specific FQDNs, it is configured to look for servers running Centos7.

```
cat AddOneHosts.install
```

The file should look like this.

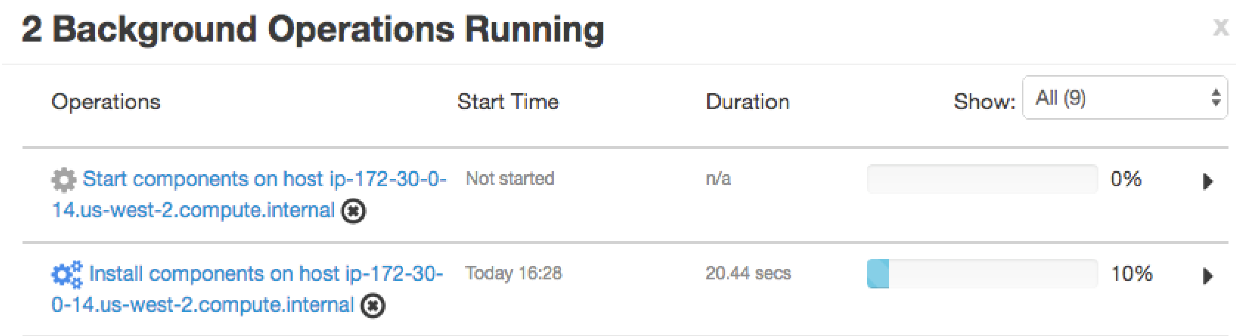
```
[centos@ip-172-30-0-34 ~]$ cat AddOneHosts.install
[
  {
    "blueprint" : "testblueprint1",
    "host_group" : "host_group_4",
    "host_count" : 1,
    "host_predicate" : "Hosts/os_type=centos7"
  }
]
[centos@ip-172-30-0-34 ~]$
```

g. Next, use cURL to issue an API call to install the new node on the existing HDP cluster.

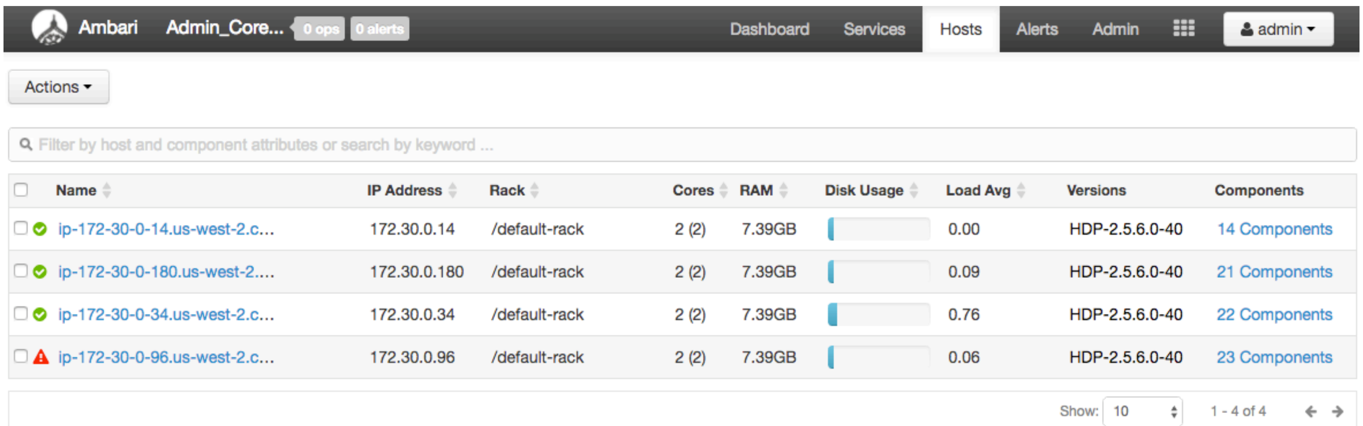
```
curl -u admin:BadPass#1 -i -H "X-Requested-By: root" -X POST -d
@AddOneHosts.install
http://localhost:8080/api/v1/clusters/Admin_Core_Training/hosts
```

```
[centos@ip-172-30-0-34 ~]$ curl -u admin:BadPass#1 -i -H "X-Requested-By: root" -X POST -d @AddOneHos
ts.install http://localhost:8080/api/v1/clusters/Admin_Core_Training/hosts
HTTP/1.1 202 Accepted
X-Frame-Options: DENY
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
Cache-Control: no-store
Pragma: no-cache
Set-Cookie: AMBARISESSIONID=17kw9668z55pbzqu9ljb21zii;Path=/;HttpOnly
Expires: Thu, 01 Jan 1970 00:00:00 GMT
User: admin
Content-Type: text/plain
Vary: Accept-Encoding, User-Agent
Content-Length: 149

{
  "href" : "http://localhost:8080/api/v1/clusters/Admin_Core_Training/requests/11",
  "Requests" : {
    "id" : 11,
    "status" : "Accepted"
  }
}[centos@ip-172-30-0-34 ~]$
```

h. Go back to the web browser which is logged into the Ambari Web UI. The ops link should once again be flashing blue. Click on it to watch the new node be installed in real time.



i. When the installation completes, the Hosts view in the Ambari Web UI should now show four nodes.



Result

You have successfully installed, configured, and scaled an HDP cluster using Ambari Blueprints.

Lab 27: Performing a HDP Upgrade

About This Lab

Objective:	Use the Ambari Web UI to set up an HDP upgrade
File locations:	N/A
Successful outcome:	The HDP cluster nodes are prepared for upgrade from version 2.5 to 2.6.
Before you begin	Start and connect to your classroom lab environment
Related lesson:	***Performing an HDP Rolling Upgrade***

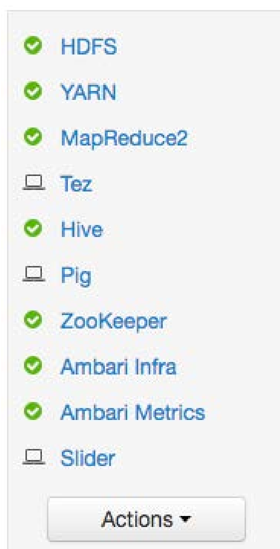
Lab Steps

Perform the following steps:

1. Open the browser and connect to the Ambari Server at URL. Login to the Ambari Web UI using the user name `admin` and the password `BadPass#1`.

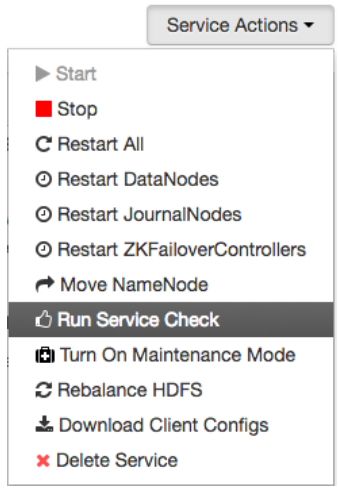
   http://<Ambari Server Hostname>:8080

2. PRE-LAB STEP: Verify that all cluster services are running and out of maintenance mode.

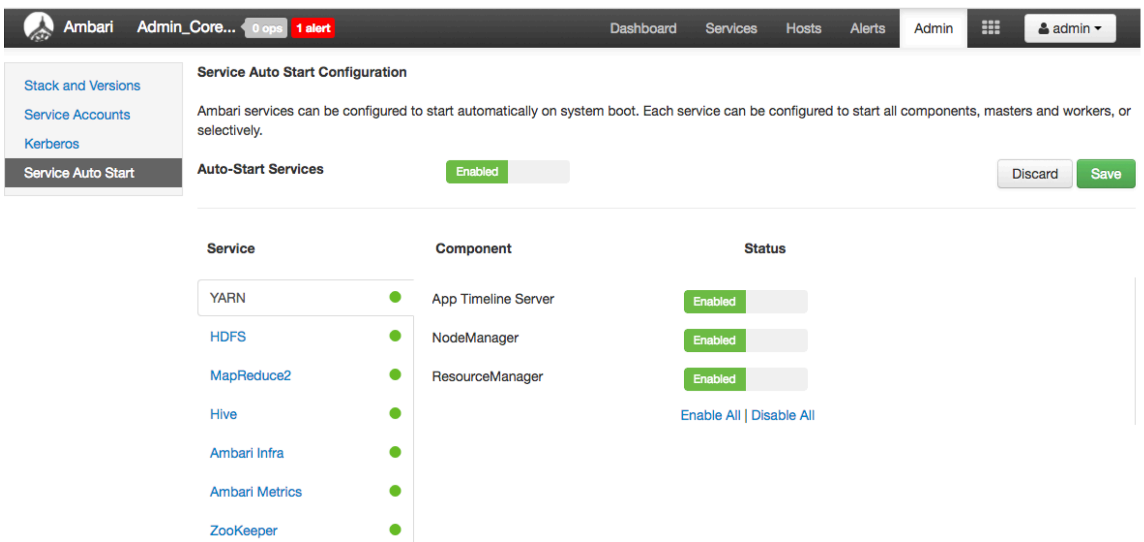
   

   Ensure that there are no outstanding alerts for any HDP services in your cluster. If any services are stopped and/or in maintenance mode, take them out of maintenance mode and start those services.

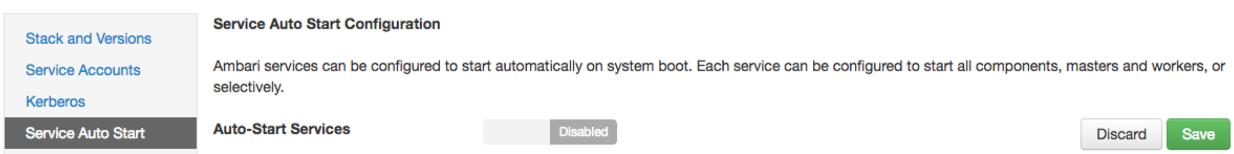
3. For all Services run the "Run Service Check" and make sure they pass.



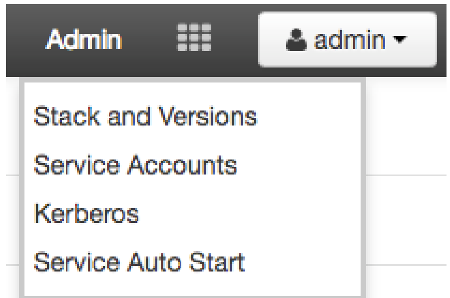
4. Disbale Service Auto Start Configuration for all components. Under Admin -> Service Auto Start -> Select Disable.



5. Click Save

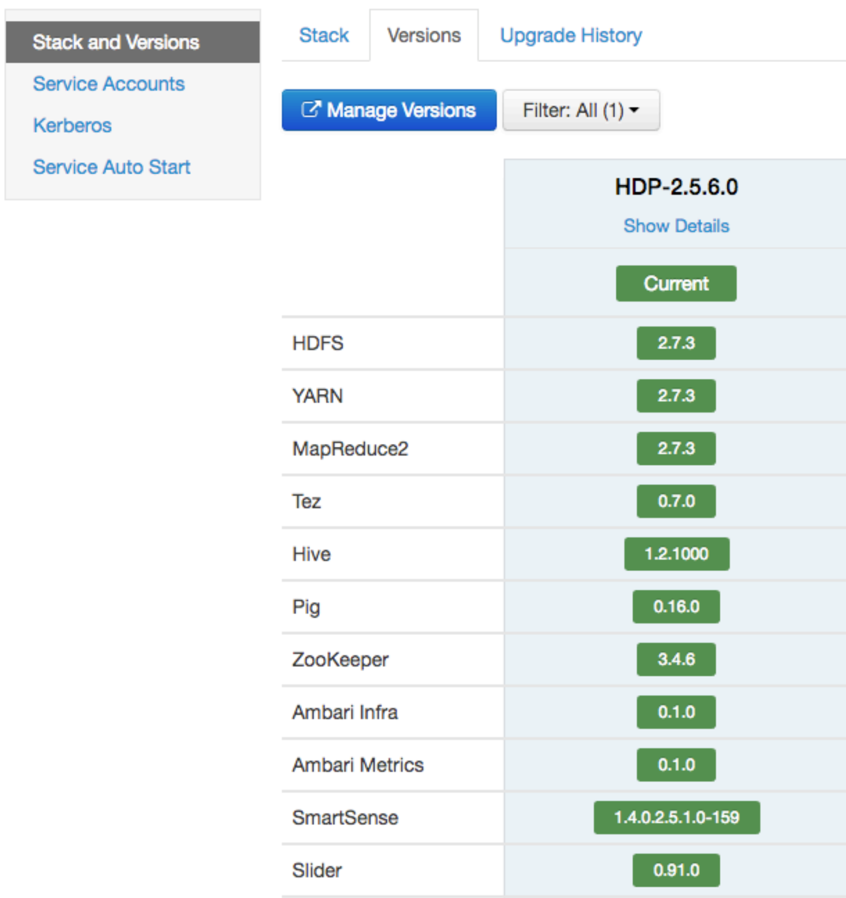


6. **Register a new HDP version using the Ambari Web UI.**
   a. Click **Admin**, and select **Stack and Versions**.



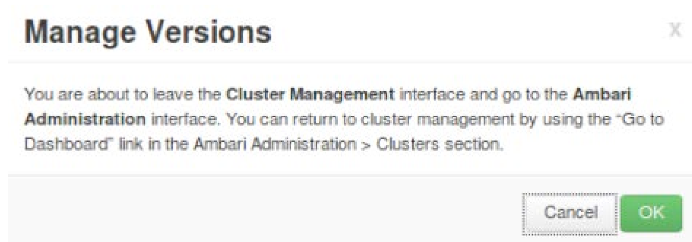
b. Click **Versions** to open the Versions pane.

The current HDP version is displayed.



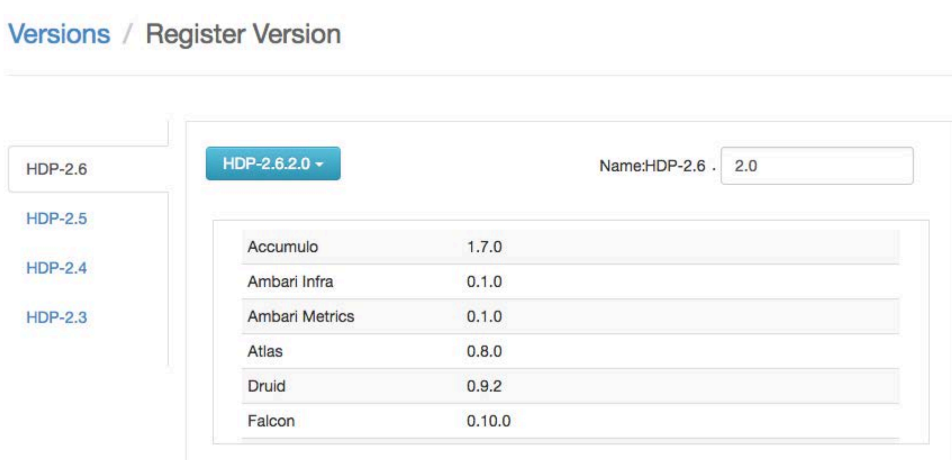
c. Click the **Manage Versions** button to register a new version.

d. In the Manage Versions window, read the message and click **OK**.



e. Click the **Register Version** button.

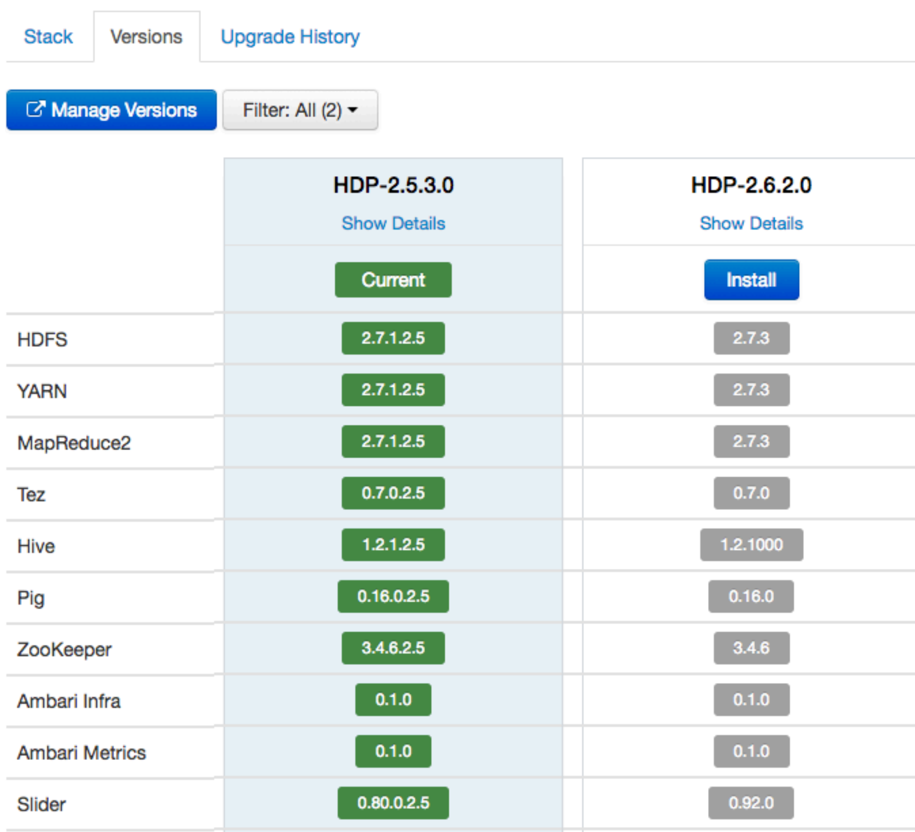
f. In the Register Version window, ensure that **HDP-2.6** appears in the drop-down. The text box to the right of **HDP-2.6** will have the minor version number filled in. Then scroll down and click **Save**.



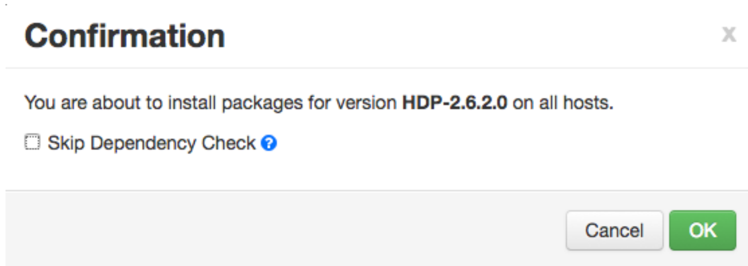
7. **Install a new HDP version using the Ambari Web UI.**

   a. In the Versions window next to the new HDP 2.6 entry, click **Install on** and select the cluster name.

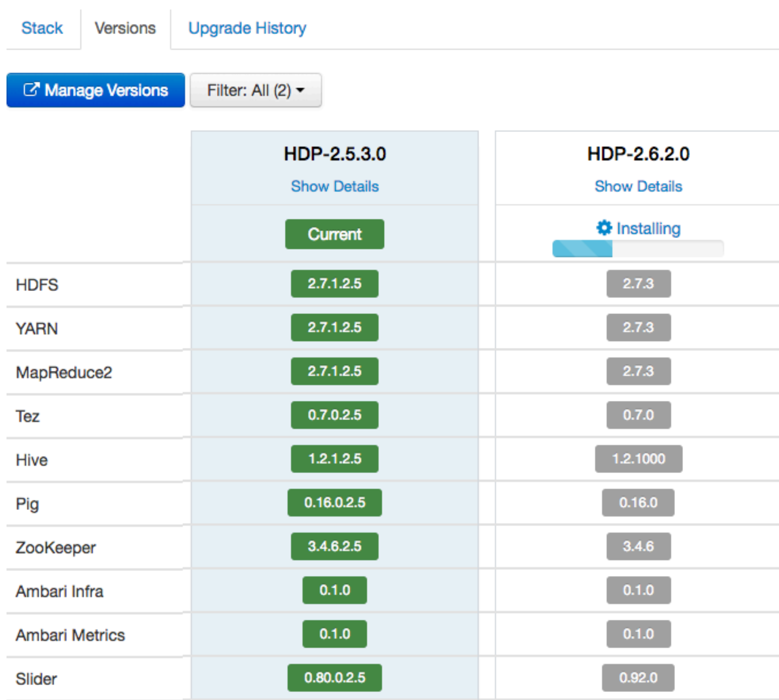
b. In the HDP-2.6.2.0 panel, click **Install**.



c. In the Confirmation window, click **OK**.

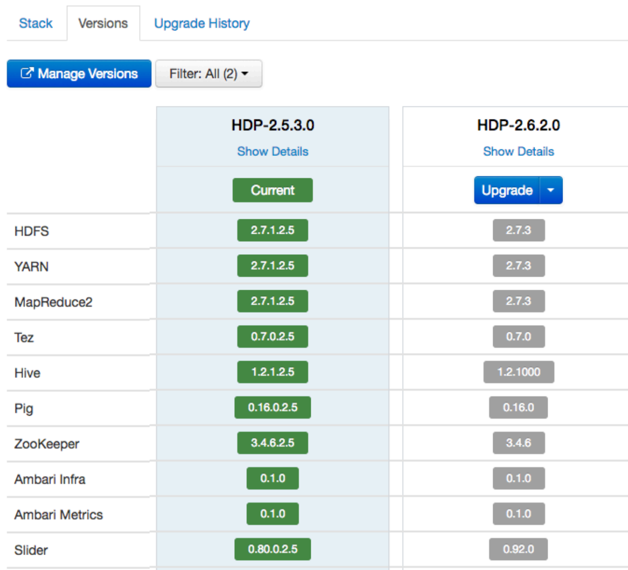


d. Monitor the installation progress bar until installation is complete.

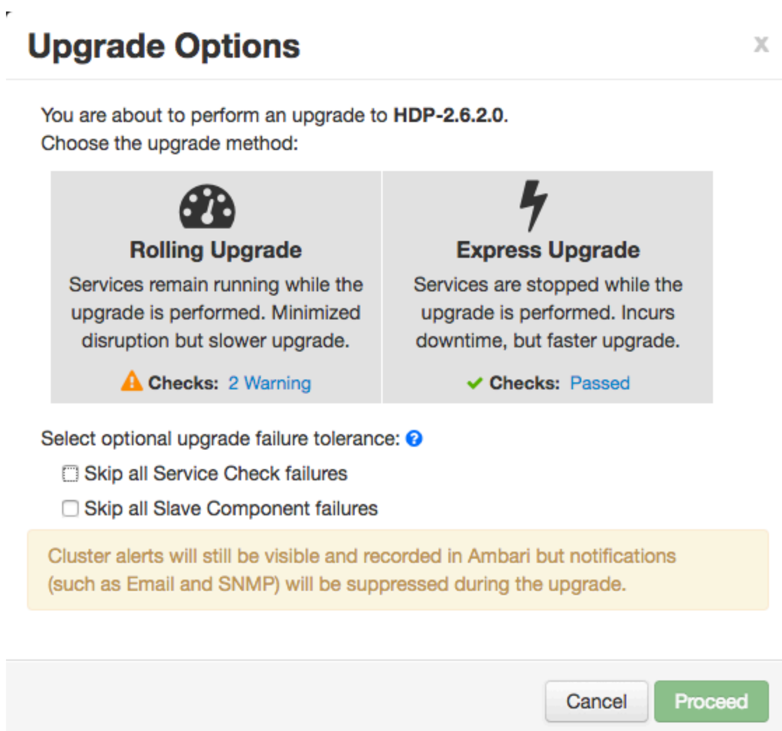


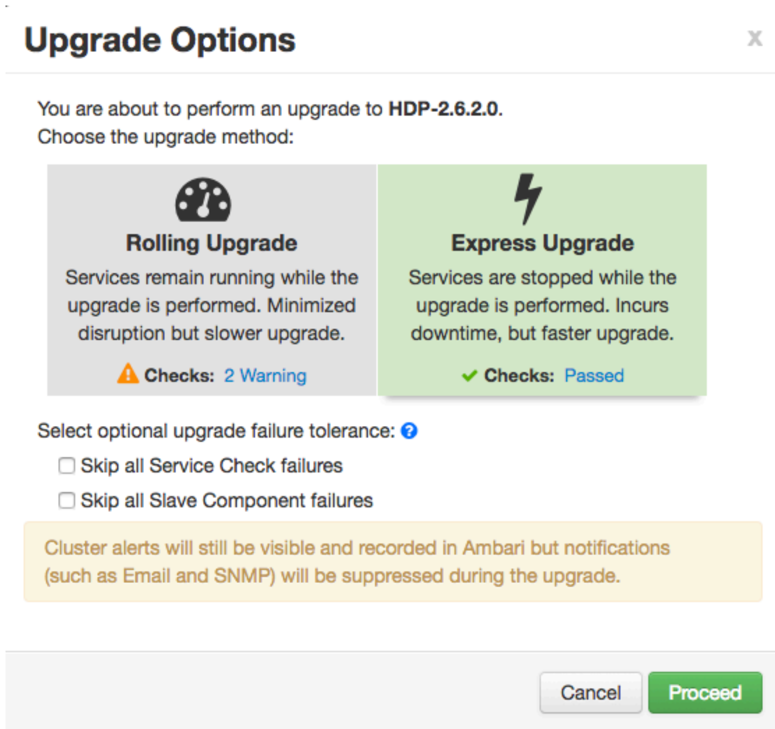
8. **Perform the upgrade using the Ambari Web UI.**

   a. To start the upgrade process, click **Upgrade**.

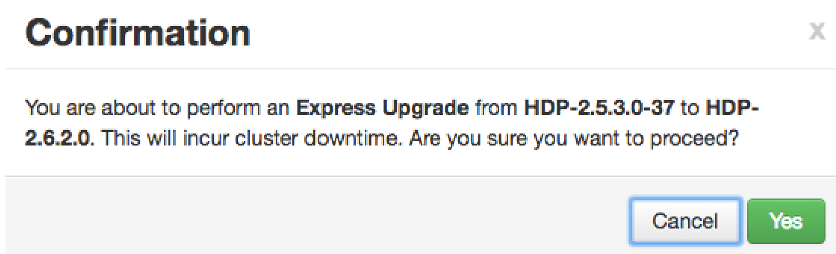


b. Choose Upgrade option "Rolling Upgrade" or "Express Upgrade" by clicking inside the appropriate box.

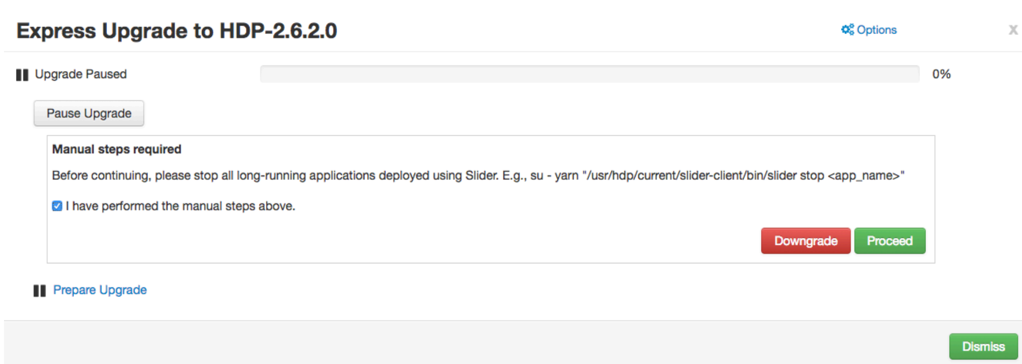




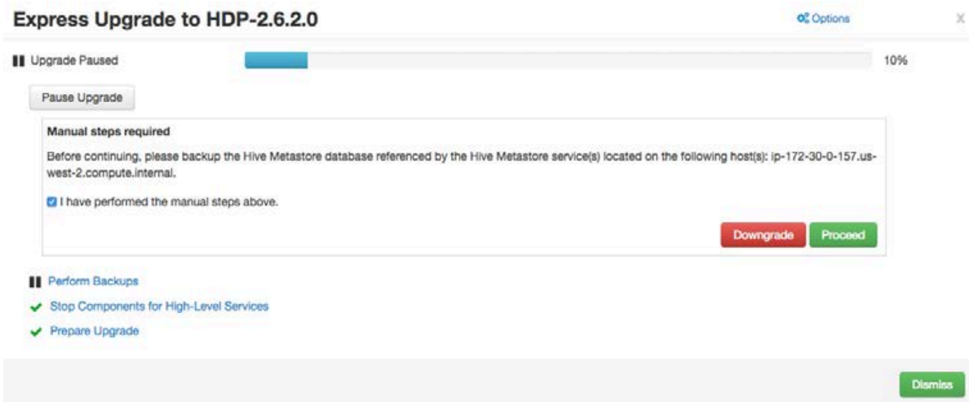
c. Click "Proceed" to start the Express Upgrade.



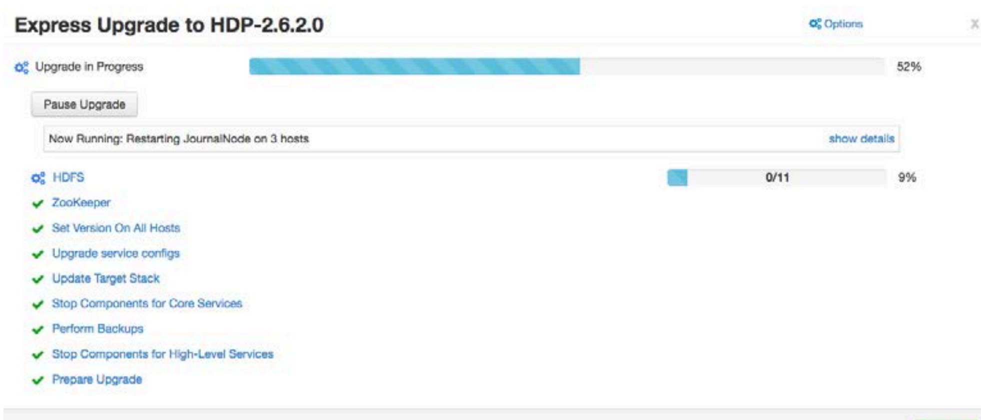
d. Click Yes to confirm the upgrade process.



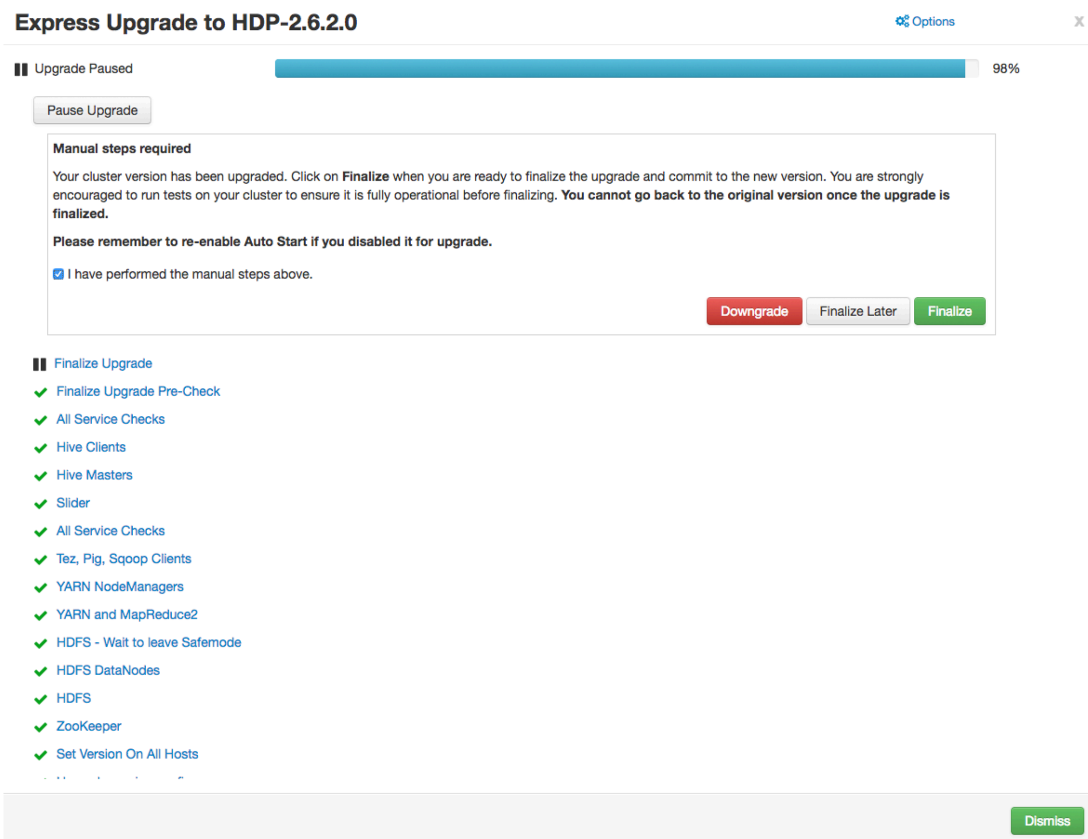
e. Click the **I have performed the manual steps above** check box, and then click **Proceed**.



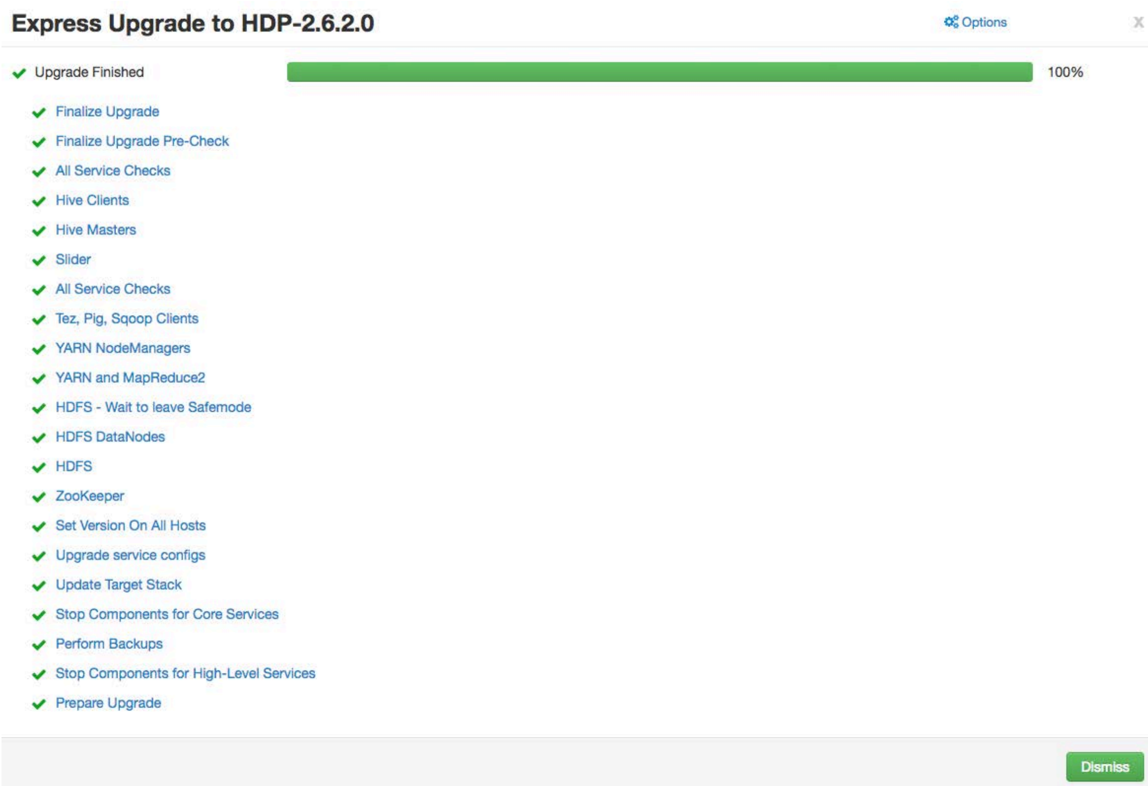
f. When again asked to confirm if the manual checks have been performed, click the **I have performed the manual steps above** check box, and then click Proceed



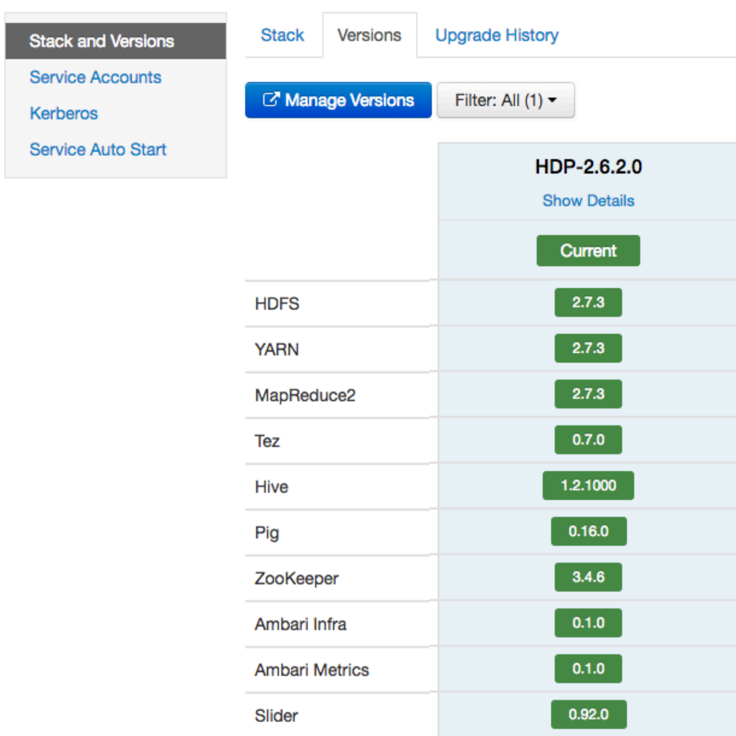
g. Upgrade will pause to allow testing. Click on Finalize when you are ready to finalize the upgrade and commit to the new version.



h. When finalization is complete, click **Dismiss**.



i. The new current version of the cluster software.



j. The express upgrade lab is complete.

Result

The HDP cluster has been upgraded using Ambari's Express Upgrade option.

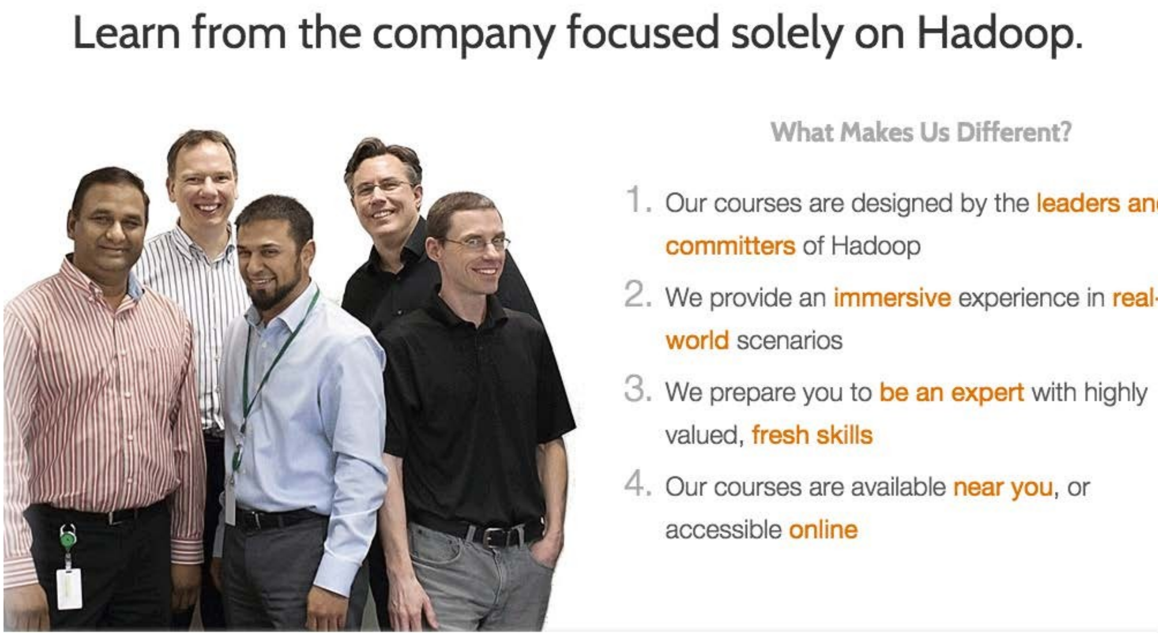
Classes Available Worldwide Through Our Partners

Study Options Worldwide

In combination with our partner providers, classes are often available in numerous locations across the world.

Private On-site Training

Hortonworks training in-house covers all of our basic coursework, and provides a more intimate setting for 6 or more students.
Contact us for more details



Hortonworks University courses are designed by the leaders and committers of Apache Hadoop. We provide immersive, real-world experience in scenario-based training. Courses offer unmatched depth and expertise available in both the classroom or online from anywhere in the world. We prepare you to be an expert with highly valued skills and for Certification.

