Hortonworks Data Platform

Administering Hadoop 2 with Ambari Web

(Feb 10, 2014)

docs.hortonworks.com

Hortonworks Data Platform : Administering Hadoop 2 with Ambari Web

Copyright © 2012, 2014 Hortonworks, Inc. Some rights reserved.

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

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

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

Except where otherwise noted, this document is licensed under
Creative Commons Attribution ShareAlike 3.0 License.
http://creativecommons.org/licenses/by-sa/3.0/legalcode

Table of Contents

1. Introducing Ambari Web 1
1.1. Cluster Monitoring Sources 1
1.2. Architecture 1
1.2.1. Sessions 2
1.3. Starting and Accessing Ambari Web 3
2. Navigating Ambari Web for Hadoop 2.x 4
2.1. The Navigation Header 4
2.2. The Dashboard View 4
2.2.1. The Widget Version 4
2.2.2. The Classic Version 7
2.3. The Heatmaps View 11
2.4. The Services View 14
2.4.1. Services Navigation 14
2.4.2. Services Summary 15
2.4.3. Configs 15
2.4.4. Quick Links 15
2.4.5. Alerts and Health Checks 16
2.4.6. Management Header 16
2.4.7. Metrics 16
2.5. The Hosts View 16
2.5.1. Host Details 17
2.5.2. Add Hosts 19
2.6. Admin View 20
2.6.1. Managing Ambari Web Users 20
2.6.2. Setting Up NameNode High Availabilty 21
2.6.3. Enabling Kerberos Security 34
2.6.4. Checking Stack and Component Versions 36
2.6.5. Checking Service User Accounts and Groups 37
3. Navigating Ambari Web for Hadoop 1.x 38
3.1. The Navigation Header 38
3.2. The Dashboard View 38
3.2.1. The Widget Version 38
3.2.2. The Classic Version 42
3.3. The Heatmaps View 44
3.4. The Services View 46
3.4.1. Services Navigation 47
3.4.2. Services Summary 47
3.4.3. Configs 47
3.4.4. Quick Links 48
3.4.5. Alerts and Health Checks 48
3.4.6. Management Header 48
3.4.7. Metrics 48
3.5. The Hosts View 48
3.5.1. Host Details 49
3.5.2. Add Hosts 51
3.6. The Jobs View 52
3.6.1. Browsing Jobs 52
3.6.2. Using Job Charts 52

3.7. Admin View 54
3.7.1. Managing Ambari Web Users 54
3.7.2. Enabling Kerberos Security 55
3.7.3. Checking Stack and Component Versions 56
3.7.4. 57
4. Using Nagios With Hadoop 58
4.1. Basic Nagios Architecture 58
4.2. Installing Nagios 58
4.3. Configuration File Locations 59
4.4. Configuring Nagios Alerts For Hadoop Services 59
4.5. Nagios Alerts For Hadoop Services 60
4.5.1. HDFS Service Alerts 60
4.5.2. NameNode HA Alerts (Hadoop 2 only) 64
4.5.3. YARN Alerts (Hadoop 2 only) 65
4.5.4. MapReduce2 Alerts (Hadoop 2 only) 68
4.5.5. MapReduce Service Alerts (Hadoop 1 only) 70
4.5.6. HBase Service Alerts 72
4.5.7. Hive Alerts 74
4.5.8. WebHCat Alerts 75
4.5.9. Oozie Alerts 75
4.5.10. Ganglia Alerts 76
4.5.11. Nagios Alerts 77
4.5.12. ZooKeeper Alerts 77
4.5.13. Ambari Alerts 78

List of Figures

1.1. Architectural Overview 2

List of Tables

2.1. Service Status 5
2.2. Widget Interactions 6
2.3. Widget Interactions 2 6
2.4. Widget Interactions 3 7
2.5. Service Status 8
2.6. Set Environment Variables 28
3.1. Service Status 39
3.2. Widget Interactions 40
3.3. Widget Interactions 2 40
3.4. Widget Interactions 3 41
3.5. Service Status 42

1. Introducing Ambari Web

Hadoop is a large scale distributed data storage and processing infrastructure using clusters of commodity hosts networked together. Monitoring and managing such complex distributed systems is a non-trivial task. To help you deal with the complexity, Apache Ambari collects a wide range of information from the cluster's nodes and services and presents them to you in an easy to read and use centralized web interface, Ambari Web. Ambari Web displays information such as service-specific summaries, graphs, and alerts. It also allows you to perform basic management tasks such as starting and stopping services, adding hosts to your cluster, and updating service configurations.

Note

At this time, Ambari Web is supported only in deployments made using the Ambari Install Wizard.

1.1. Cluster Monitoring Sources

Using proven open-source monitoring systems including Ganglia and Nagios, Ambari gathers information on the status of both of the hosts and the services that run on them, including the status of any jobs running on those services.

- **Host and System Information**: Ambari monitors basic host and system information such as CPU utilization, disk I/O bandwidth and operations per second, average memory and swap space utilization, and average network latency.
- **Service Information**: Ambari monitors the health and performance status of each service by presenting information generated by that service. Because services that run in master/slave configurations (HDFS, MapReduce, and HBase) are fault tolerant in regard to service slaves, master information is presented individually, whereas slave information is presented largely in aggregate.
- **Job Information**: Ambari monitors job status by using information from MapReduce's JobTracker, which produces detailed data both on job status, current and historical, and on job scheduling.
- **Alert Information**: Using Nagios with Hadoop-specific plugins and configurations, Ambari Web can issue alerts based on service states defined on three basic levels:
  - OK
  - Warning
  - Critical

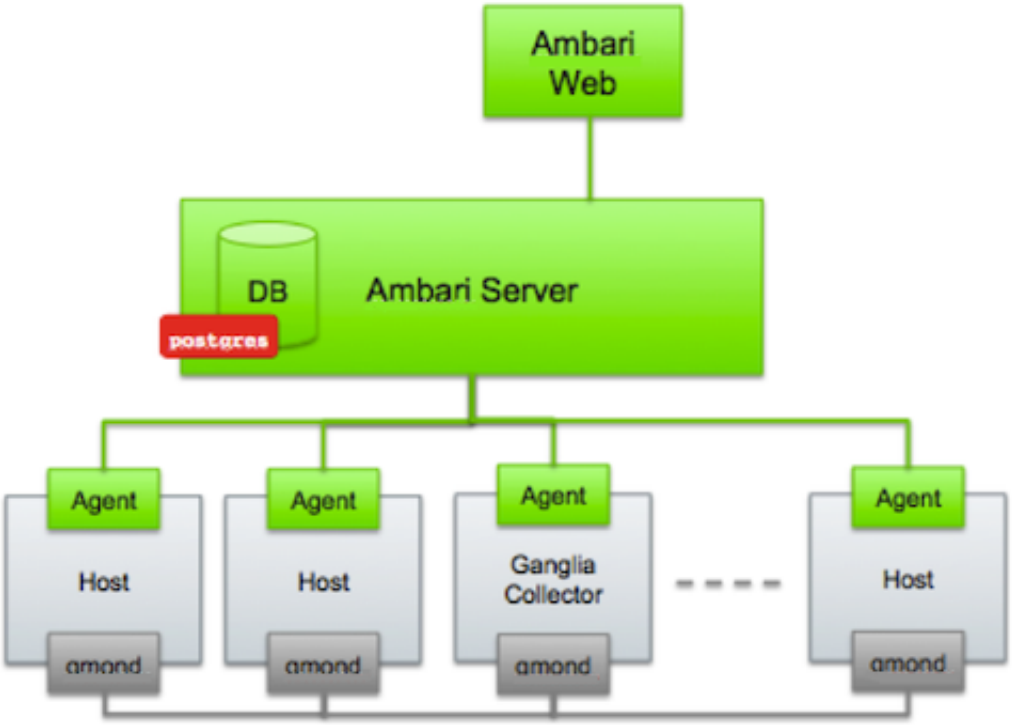
  The thresholds for these alerts can be tuned using configuration files, and new alerts can be added. See Using Nagios with Hadoop for more information.

1.2. Architecture

The Ambari Server serves as the collection point for data from across your cluster. Each host has a copy of the Ambari Agent - either installed automatically by the Install wizard or

manually - which allows the Ambari Server to control each host. In addition, each host has a copy of Ganglia Monitor (`gmond`), which collects metric information that is passed to the Ganglia Connector, and then on to the Ambari Server.

Figure 1.1. Architectural Overview



1.2.1. Sessions

Ambari Web is a client-side JavaScript application, which calls the Ambari REST API (accessible from the Ambari Server) to access cluster information and perform cluster operations. After authenticating to Ambari Web, the application authenticates to the Ambari Server and communication between the browser and server occur asynchronously via the REST API.

Note

Ambari Web sessions do not timeout since the application is constantly accessing the REST API, which resets the session timeout. As well, if there is a period of Ambari Web inactivity, the Ambari Web interface is automatically refreshed. Therefore you must **explicitly sign out** of the Ambari Web interface to destroy the Ambari session with the server.

1.3. Starting and Accessing Ambari Web

Generally the Ambari Server and Ambari Web are started as part of the installation process. If for some reason the server is not running, on the Ambari Server machine, type:

```
ambari-server start
```

To access Ambari Web, open a supported browser and enter the Ambari Web URL:

```
http://{your.ambari.server}:8080
```

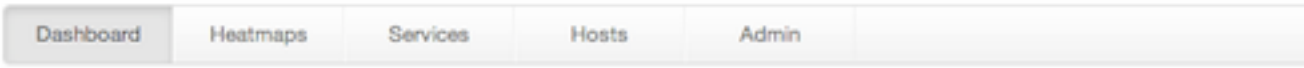
Enter your username and password. If this is the first time Ambari Web is accessed, use the default values, `admin/admin`. These values can be changed, and new users provisioned, using the **Admin** view in Ambari Web itself.

2. Navigating Ambari Web for Hadoop 2.x

This section gives you an overview of using the Ambari Web GUI for monitoring and managing your Hadoop 2 cluster.

2.1. The Navigation Header

At the top of every screen is the Navigation Header.



This header appears in all views in Ambari Web. Use this bar to move from view to view. The active view is indicated with a slightly darker gray.

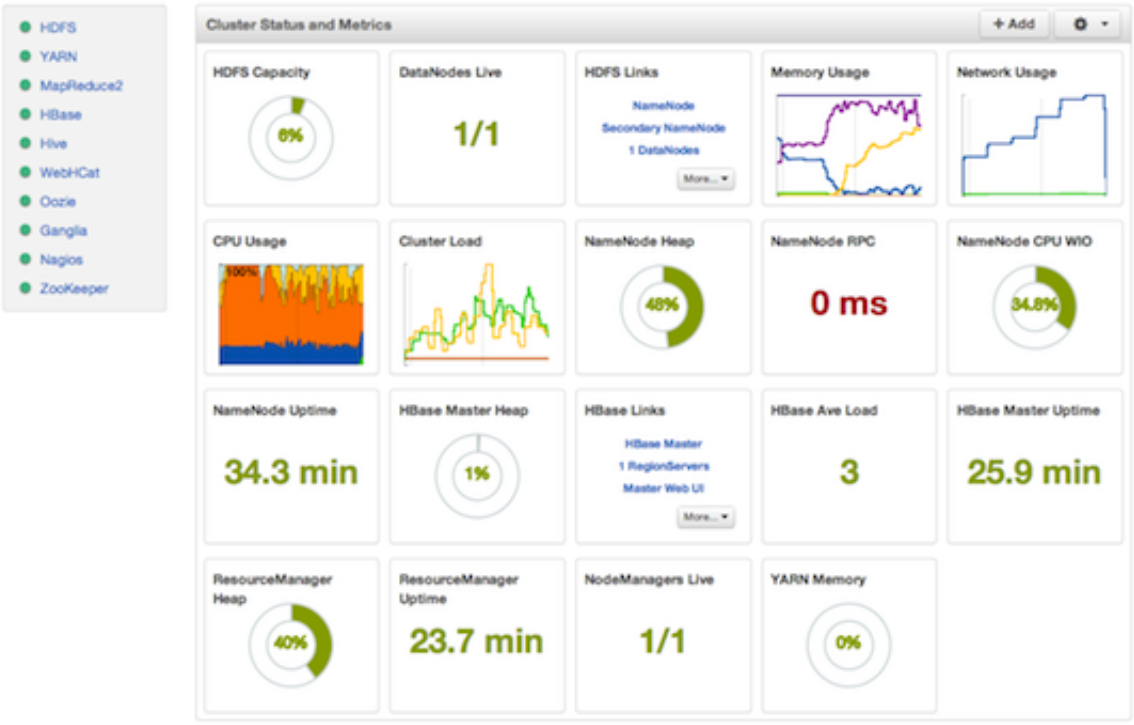
2.2. The Dashboard View

When you open Ambari Web, you are placed in the **Dashboard** view. This view is divided into two versions:

- The Widget version
- The Classic version

2.2.1. The Widget Version

The **Dashboard** Widget version opens first. This view gives you a customizable overview of the state of your cluster as a whole.



The Widget version is divided into two main sections:

- Services Summary
- Cluster Status and Metrics

2.2.1.1. Services Summary

On the left border of the screen is the **Services** summary. You can use this section to get an overall view of the status of your services. In the image above [4], notice that each service name has a green dot next to it. Use the dot colors give you a quick overview of the service's status.

Table 2.1. Service Status

Color	Name	Status
	Solid Green	All masters are running
	Blinking Green	Starting up
	Solid Red	At least one master is down
	Blinking Red	Stopping

Click the service name to open the **Services** screen, where you can see more detailed information on each service.

2.2.1.2. Cluster Status and Metrics

On the main section of the screen [4], a customizable set of widget tiles presents information on the status of your cluster. There are simple pie and bar charts, more complex usage and load charts, and sets of links to additional data sources, as well as status information like uptime and average RPC queue wait times.

Table 2.2. Widget Interactions

To:	Do:
Move widgets around on the screen	Drag and drop to desired position
See more detailed information	Hover over the widget box

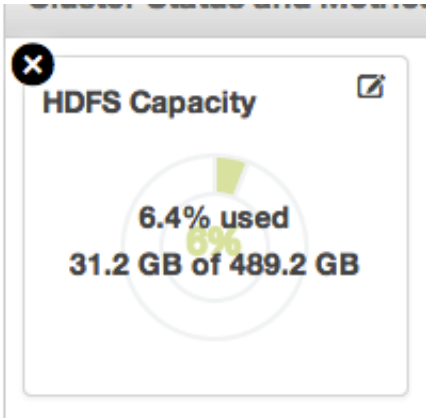
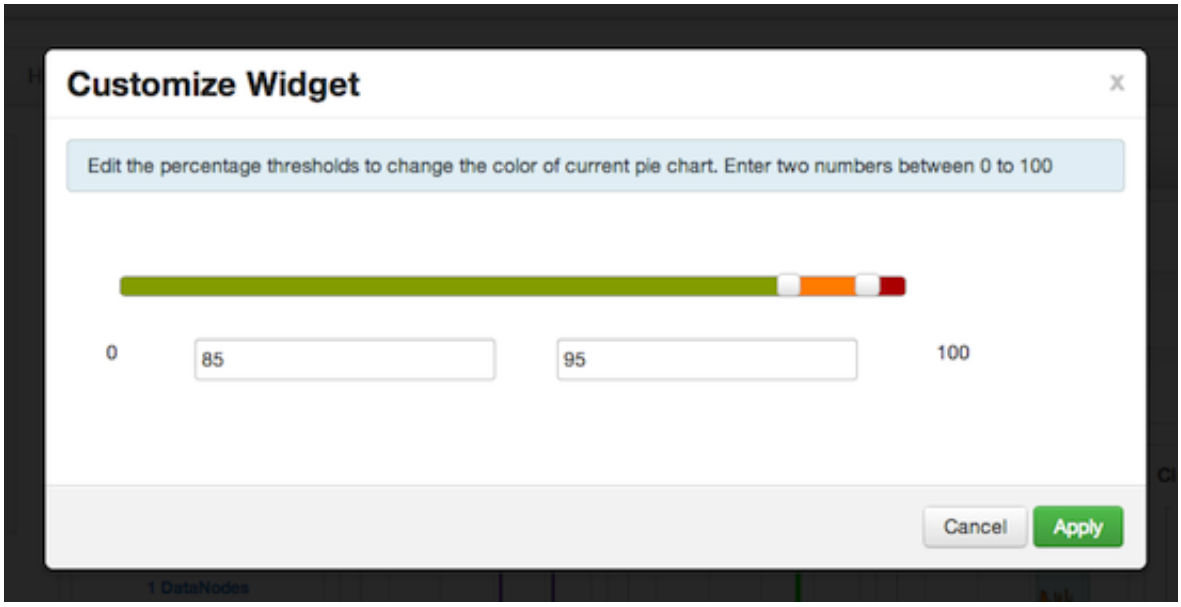


Table 2.3. Widget Interactions 2

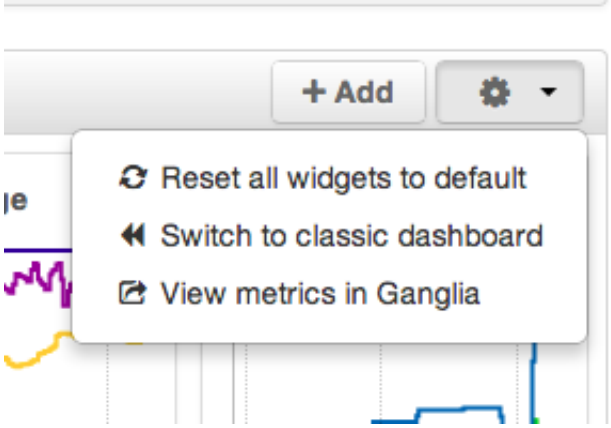
To:	Do:
Delete a widget	Click the X, marked in red
Edit a widget	Click the small edit icon, marked in blue. The **Customize Widget** popup appears.



Follow the instructions in the popup to customize the widget display. In this case, you can adjust the thresholds at which the **HDFS Capacity** bar chart changes color, from green to orange to red. Click **Apply** to save your changes. Not all widgets have an edit icon.

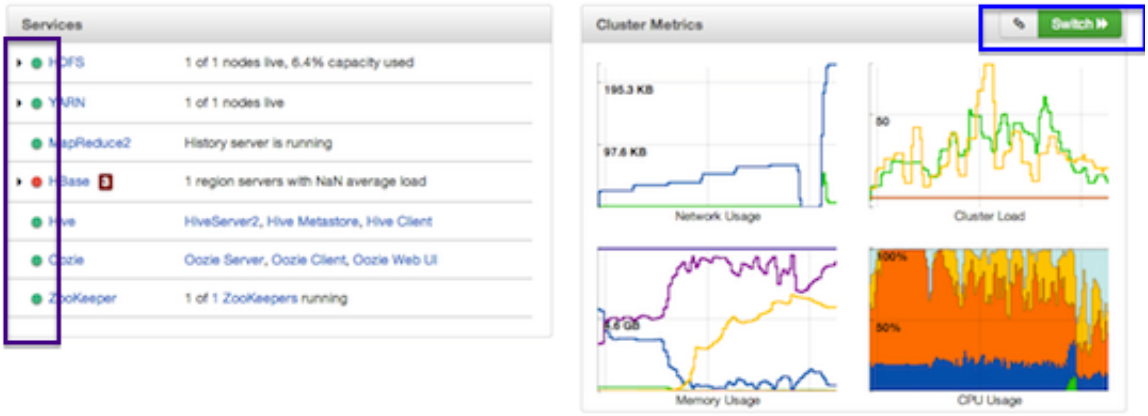
Table 2.4. Widget Interactions 3

To:	Do:
Add back a deleted widget	Click **+Add**, outlined in green [4], and use the checkbox to select the widget you want from the dropdown menu
Use quick links to other information, like thread stacks, logs and native component GUIs	Click **More**, outlined in purple [4], and select from the dropdown menu
Zoom into more complex charts, like the Network Usage chart, outlined in orange [4]	Click the chart. A larger version pops up. You can choose to add and remove information from the chart by selecting or deselecting the metric in the legend
Switch to the Classic version, return widgets to default setting, or view metrics in the native Ganglia Server interface	Use the gear icon



2.2.2. The Classic Version

The **Dashboard** Classic version provides a slightly different view of high-level cluster information.



The Classic version is also divided into two main sections:

- Services Summary
- Cluster Metrics

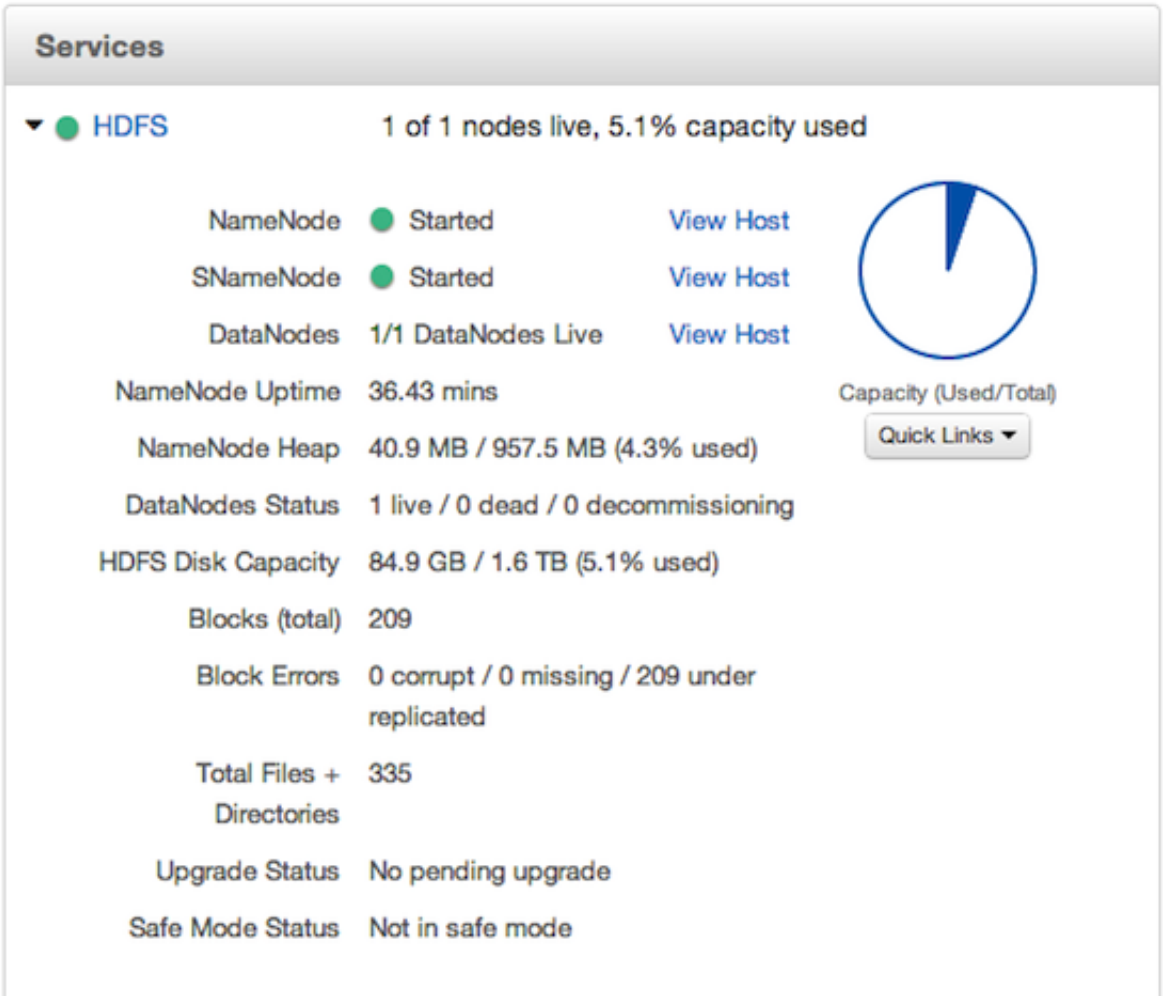
2.2.2.1. Services Summary

On the left side of the screen is the **Services** summary. Use this section to get an overall view of the status of your services. In the image above [7], notice that **HDFS**, **YARN**, **MapReduce2, Hive**, **Oozie**, and **Zookeeper** have green dots next to them but that **HBase** has a red dot. The dot colors give you a quick overview of the service's status.

Table 2.5. Service Status

Color	Name	Status
	Solid Green	All masters are running
	Blinking Green	Starting up
	Solid Red	At least one master is down
	Blinking Red	Stopping

Notice also that HDFS, YARN, and HBase all have small triangles next to them. These are the services that are deployed in master/slave sets. Click the triangle to open a more detailed picture of the service.



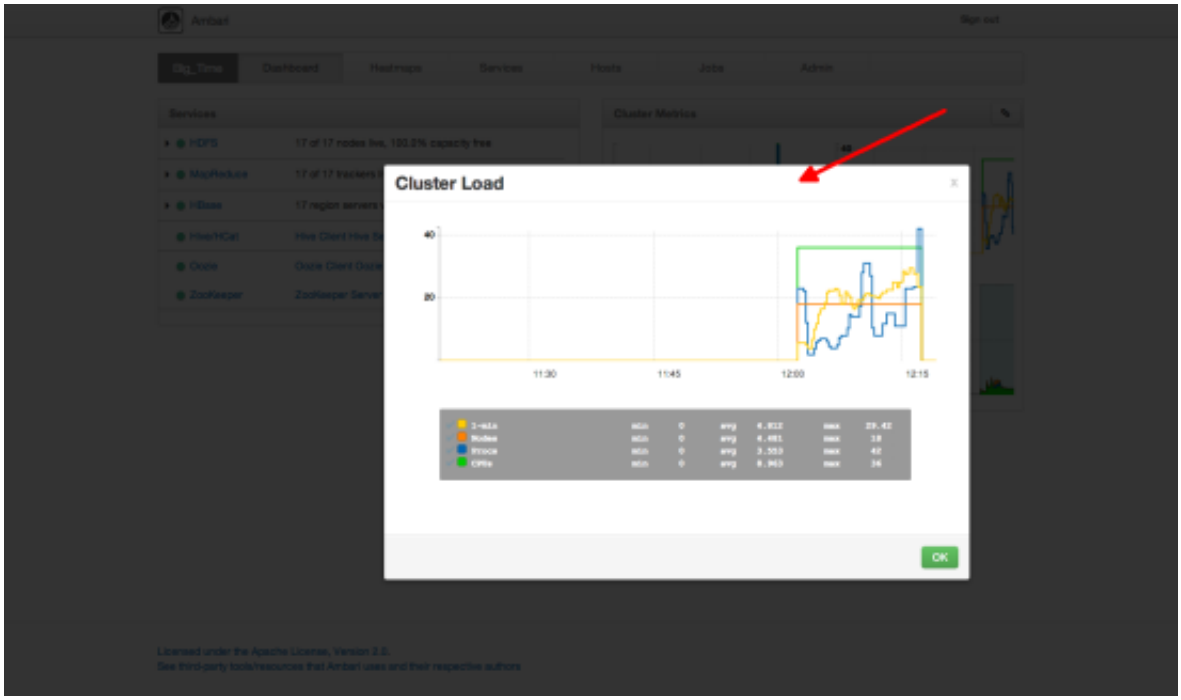
If an alert is raised against the service, a small rectangle showing the number of alerts raised, marked in blue below, appears next to the service name.

Click the service name to open the **Services** screen, where you can see more detailed information on the alert.

2.2.2.2. Cluster Metrics

On the right side of the screen is the **Cluster Metrics** section. This section gives you charts for a snapshot of the important clusterwide metrics: **Network Usage**, **Cluster Load**, **Memory Usage**, and **CPU Usage**. To see a legend for the chart, hover over it. To remove a metric from the chart, click on the legend in the metric to remove the checkmark and deselect it.

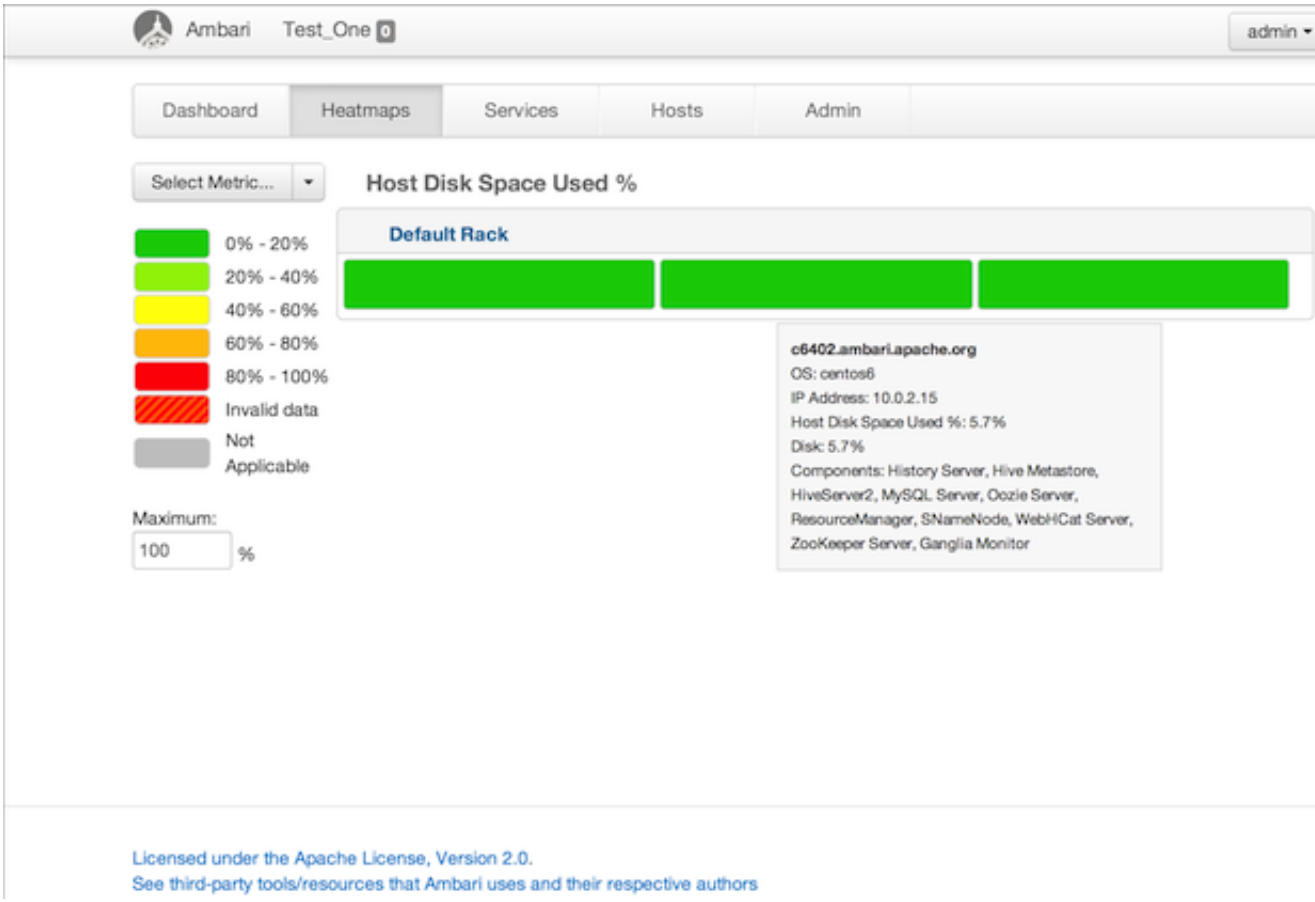
To see a larger view of the chart, click on it. The larger chart pops out.



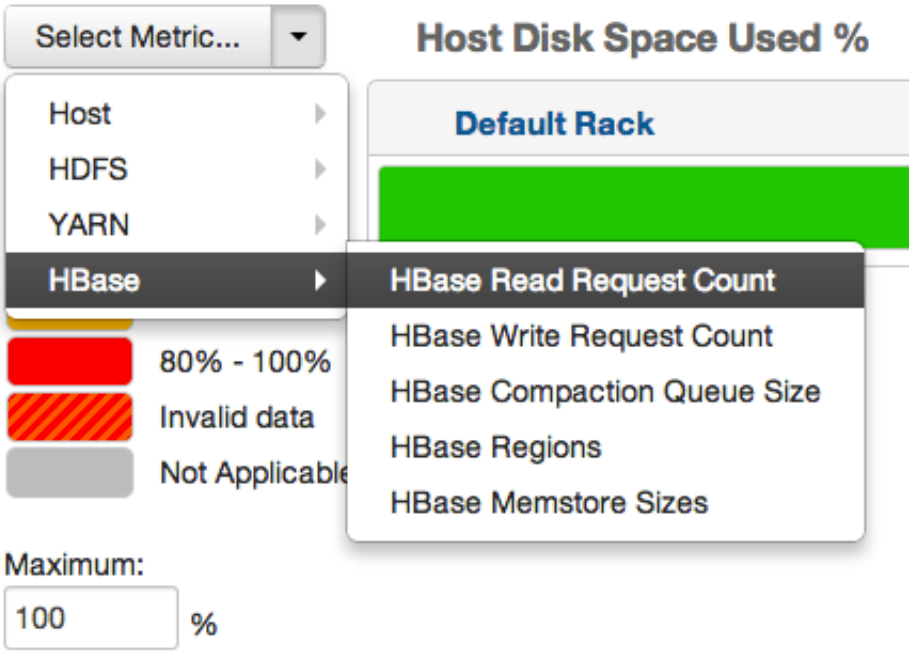
Notice the link symbol on the upper right side of the Cluster Metric section, outlined in blue in the overview screenshot [7] above. This is a link to the GUI for the Ganglia Server itself, where you can find much more detailed information on your cluster. You can also use this to switch back to the Widget version.

2.3. The Heatmaps View

The **Heatmaps** view gives you a graphic representation of the overall utilization of your cluster using simple color coding.



Each host in the cluster is represented by a block. To see more information on a specific host, hover over the block you are interested in, and a popup with key host data appears. The color of the blocks represents usage in an appropriate unit based on a selectable set of metrics. If the data necessary to determine state are not all available, the block is marked as having Invalid Data. Changing the default maximum values for the heatmap lets you fine tune the representation. Use the Select Metric dropdown to select the metric type.

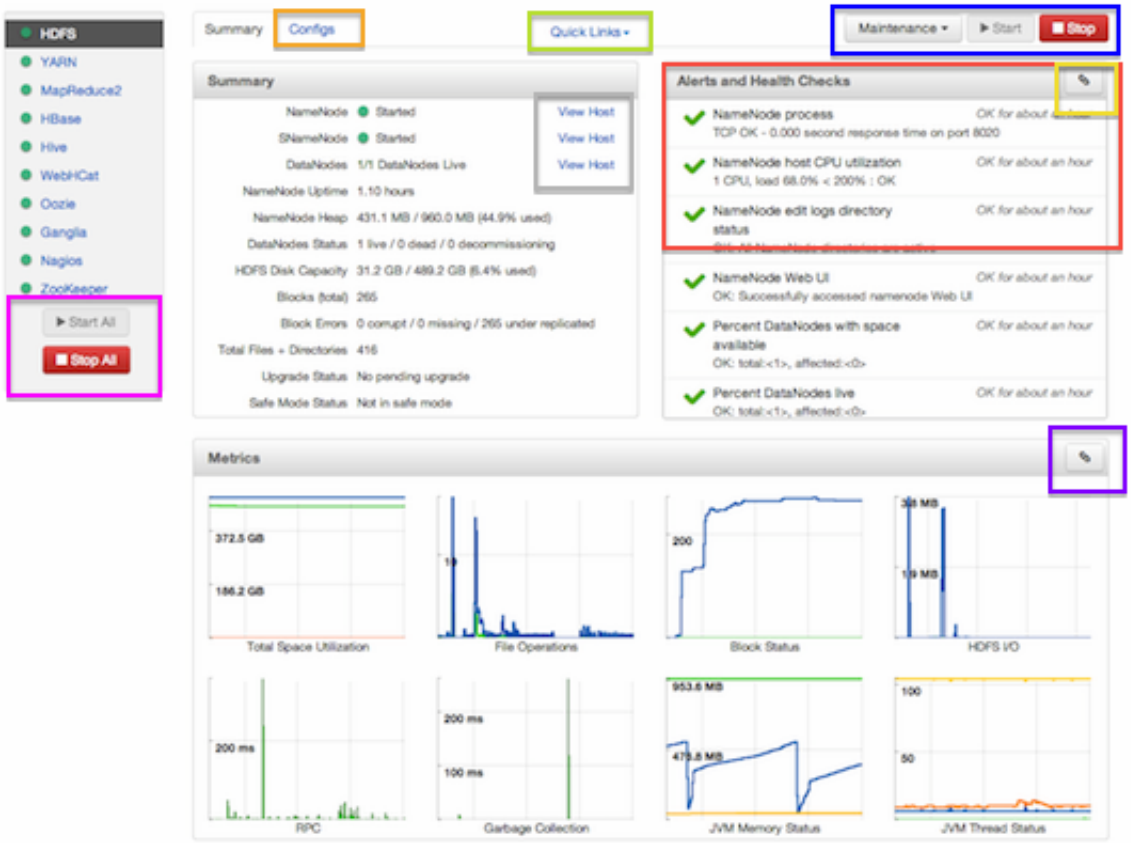


Currently the following metrics are supported:

- Host/Disk Space Used % Uses disk.disk_free and disk.disk_total
- Host/Memory Used %: Uses memory.mem_free and memory.mem_total
- Host/CPU Wait I/O %: Uses cpu.cpu_wio
- HDFS/Bytes Read: Uses dfs.datanode.bytes_read
- HDFS/Bytes Written: Uses dfs.datanode.bytes_written
- HDFS/Garbage Collection Time: Uses jvm.gcTimeMillis
- HDFS/JVM Heap MemoryUsed: Uses jvm.memHeapUsedM
- YARN/Garbage Collection Time: Uses jvm.gcTimeMillis
- YARN / JVM Heap Memory Used: Uses jvm.memHeapUsedM
- YARN / Memory used %: Uses UsedMemoryMB and AvailableMemoryMB
- HBase/RegionServer read request count: Uses hbase.regionserver.readRequestsCount
- HBase/RegionServer write request count: Uses hbase.regionserver.writeRequestsCount
- HBase/RegionServer compaction queue size: Uses hbase.regionserver.compactionQueueSize
- HBase/RegionServer regions: Uses hbase.regionserver.regions
- HBase/RegionServer memstore sizes: Uses hbase.regionserver.memstoreSizeMB

2.4. The Services View

The **Services** view gives you access to detailed information on each of the services running in your cluster. It also allows you to start and stop the service, run smoke tests, and change configuration details.



The Services view is divided into seven sections:

- Services Navigation
- Services Summary
- Configuration Update
- Quick Links
- Alerts and Health Checks
- Management Header
- Metrics

2.4.1. Services Navigation

The Services navigation panel on the left gives you a quick look at the status of your services. The color of the dot next to the left of service name tells you the service running

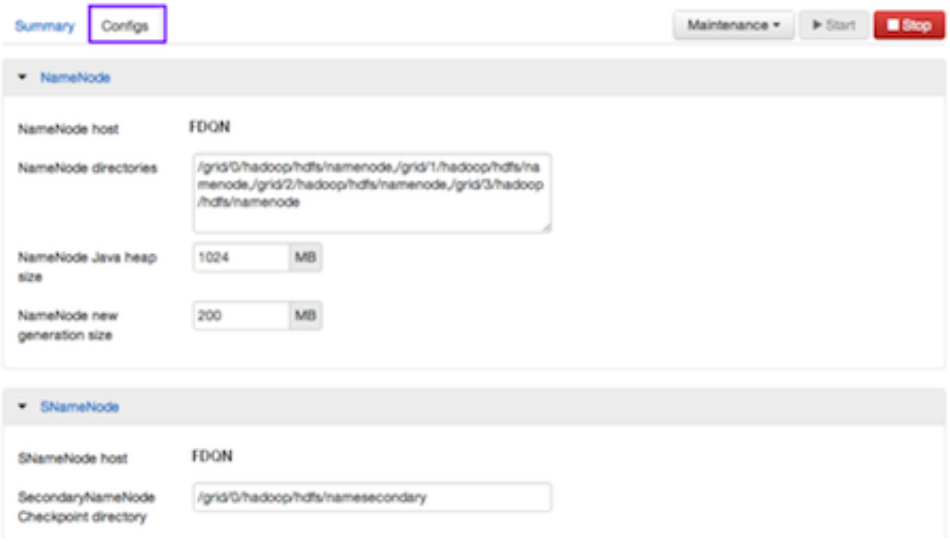
state and a small rectangle to the right lets you know if there are any alerts assigned to it. To move the larger screen display from showing information about one service to another, click the service name. To stop or start all of the services at once, use the **Start All** or **Stop All** buttons, outlined in magenta [14].

2.4.2. Services Summary

The Services **Summary** tab displays basic information about the selected service. Use the **View Host** links, marked in gray [14] above to move to the **Host Details** view of the host that is running the service. Clicking **Configs**, outlined in orange [14] above, opens a second tab, the Configuration Update tab.

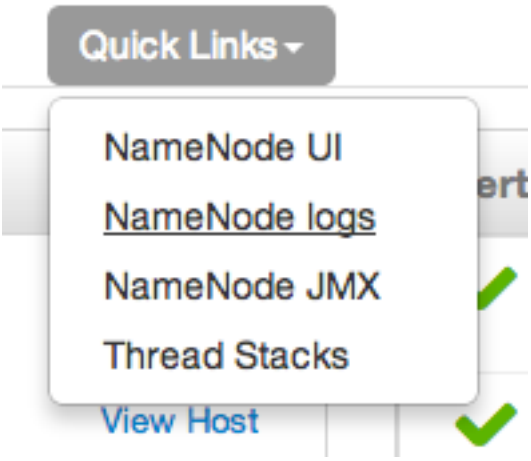
2.4.3. Configs

The Configuration Update tab allows you to update configurations for your service. The screen that appears is familiar from the Install wizard configuration page. Once you have made your changes, use the management header to stop and restart the service.



2.4.4. Quick Links

Quick Links, outlined in light green, takes you to additional sources of information about this service. In the case of HDFS, for example, it contains links to the native NameNode GUI, NameNode logs, the NameNode JMX output, and thread stacks for the service. Not all **Services** pages include Quick Links.

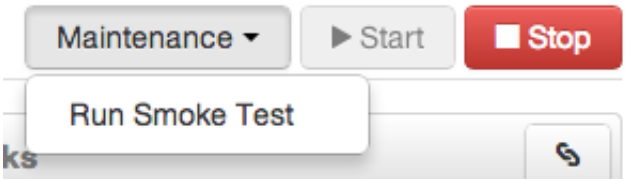


2.4.5. Alerts and Health Checks

The **Alerts and Health Checks** panel displays the results of the health checks performed on your cluster by Nagios. Alerts display a brief summary of the issue and its rating. They are sorted by descending severity, followed by descending time. To access more detailed information, click the link in the upper right corner of the panel, outlined in yellow [14] above. This opens the native Nagios GUI. Use the username and password you set up during installation to log in.

2.4.6. Management Header

The management header, outlined in blue [14], gives you a convenient way to stop and start an individual service. You can also use **Maintenance** to perform smoke tests on the service, for example, if you have updated the service's configuration.



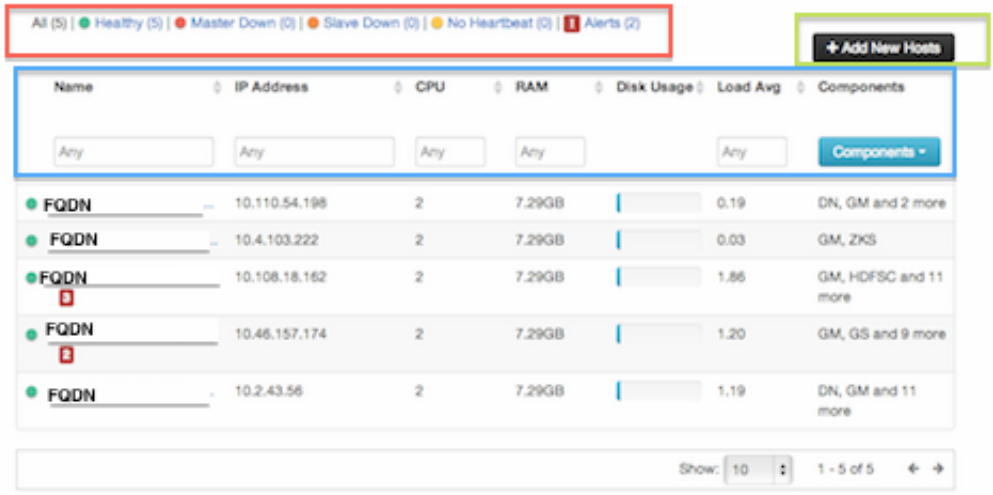
2.4.7. Metrics

The **Metrics** panel displays a set of charts measuring common metrics for the service. Hover above a particular chart to see a legend and click a specific metric type to select or deselect it. Click the chart to see a larger version. To get more information, click the link in the upper right corner of the panel, outlined in purple [14]. This opens the native Ganglia GUI.

2.5. The Hosts View

The Hosts view presents your cluster in terms of the hosts on which the services are running. If your cluster is large, you can use the health filter, outlined in red, to locate only

hosts in that state. The more general filter tool, shown outlined in blue, allows you to select specific types of hosts and to sort your result.



Use the dropdown list in the **Components** column to locate only the hosts that run the specific Master components in which you are interested. After you have made your choice, click the **Apply** button at the bottom of the list.

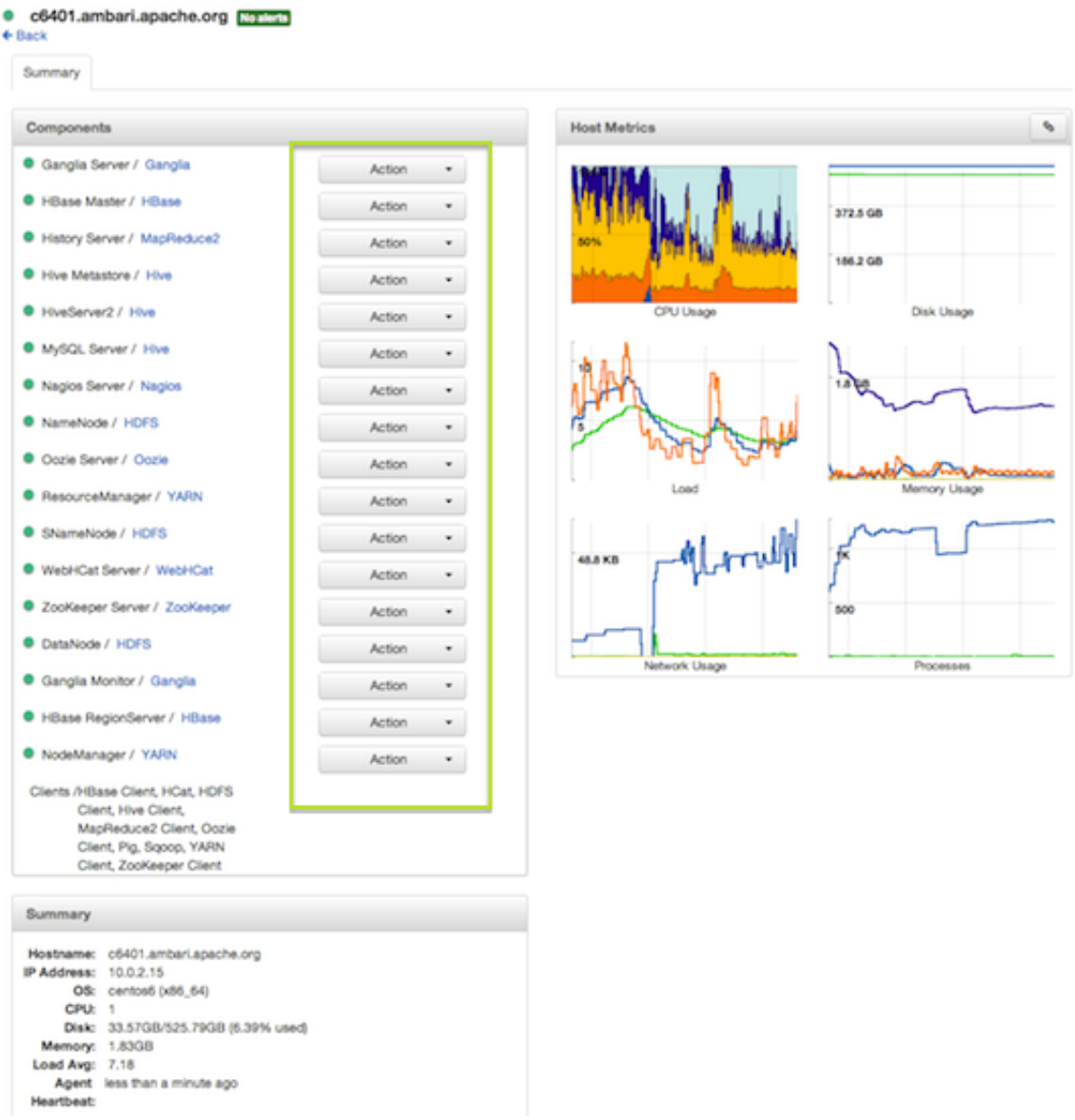
Note the colored dots next to the FQDN in the main Hosts view. These give you a quick overview of the host status:

- Red - At least one master component on that host is down. Hover to see a tooltip with a list of affected components.
- Orange - At least one slave component on that host is down. Hover to see a tooltip with a list of affected components.
- Yellow - Ambari Server has not received a heartbeat from that host for more than 3 minutes.
- Green - Normal running state

The trigger for a red condition overrides a trigger for an orange condition, and so on down the list.

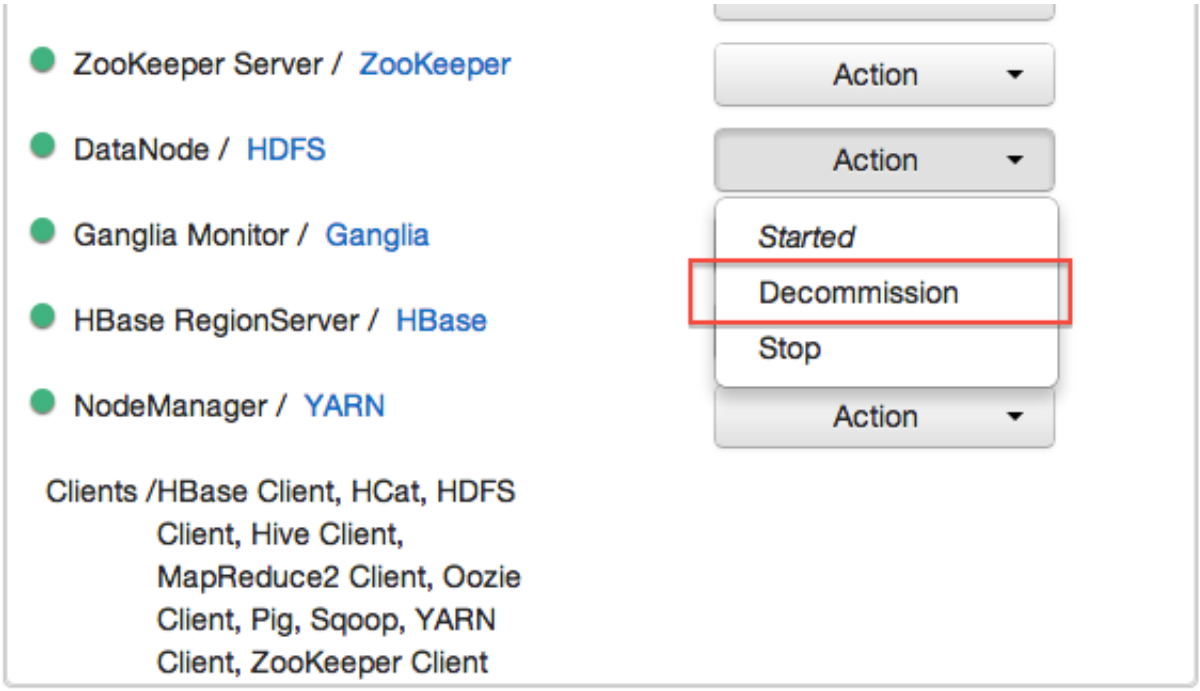
2.5.1. Host Details

To see more detailed information on a specific host, click the host FQDN. The Host Details screen opens.



Use the **Action** dropdowns, outlined in green above, to stop or start a service component running on that host.

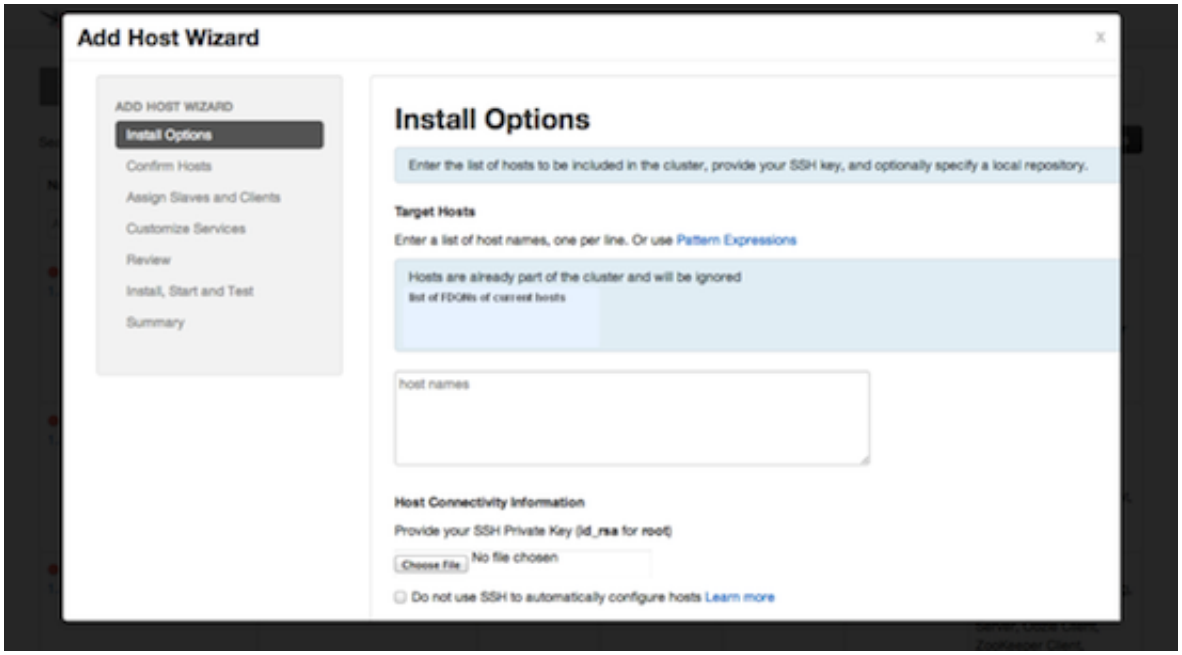
You can also use the Action dropdown on a DataNode host to decommision that node and safely move data from it to other DataNodes in your cluster.



Decommissioning a DataNode runs as a background operation. When the data has all been transferred and replication is complete, the DataNode shuts itself down. The health status colored dot turns red to let you know the decommission process is complete.

2.5.2. Add Hosts

To add new hosts to your cluster, click **+Add New Hosts**, outlined in green [16] above. The **Add Host Wizard** pops up.



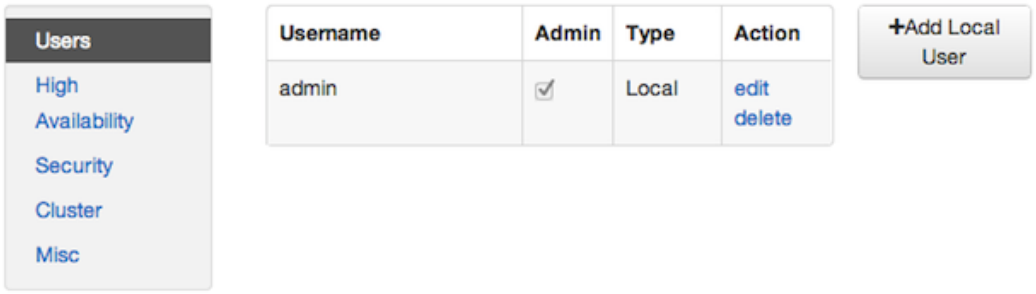
Follow the wizard through the sequence of steps - similar to those in the Install wizard - to add hosts. To review the Install wizard, see Installing, Configuring, and Deploying the Cluster in the Ambari Installation Guide.

2.6. Admin View

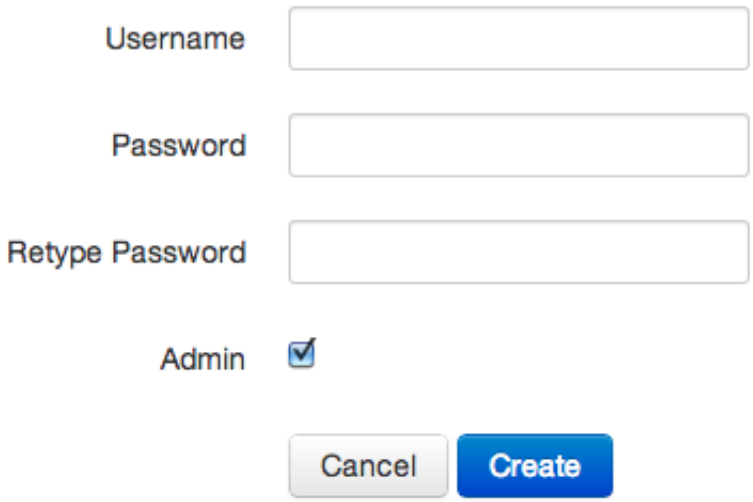
The **Admin** view allows you to manage Ambari Web users and check for general information about the cluster.

2.6.1. Managing Ambari Web Users

Select Users in the left nav bar to add Ambari Web users, make them administrators, delete them, or change their passwords. There are two user roles: **User** and **Admin**. **Users** can view metrics, view service status and configuration, and browse job information. **Admins** can do all User tasks and in addition can start or stop services, modify configurations, and run smoke tests.



To make a change in a current user's password, click **edit**, in red above. To add an additional Ambari Web user, click **+Add Local User**. In both cases, a variant of the username pop up appears.



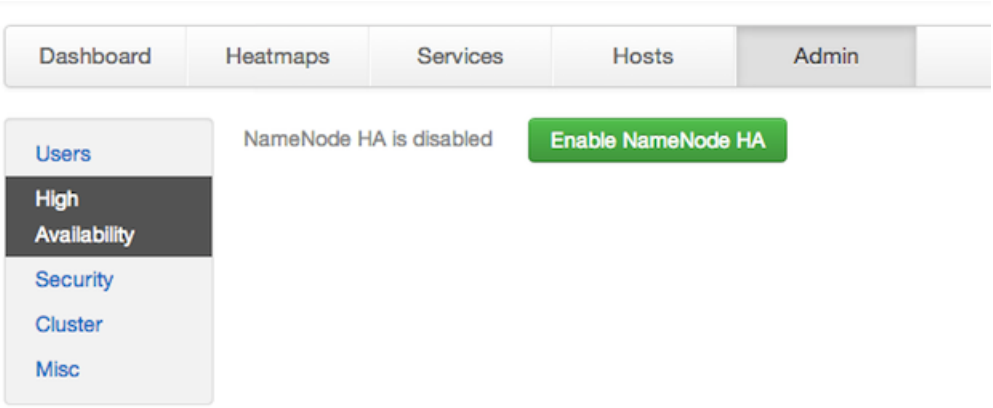
For a new Ambari Web user, enter the desired **Username** and **Password**. Check **Admin** to make this user an administrator. Click **Create** to make the user.

For an Ambari Web existing user, enter the old and new **Passwords**. Make your changes and click **Save**.

To delete an existing Ambari Web user, click **delete** and fill in the **Username**.

2.6.2. Setting Up NameNode High Availabilty

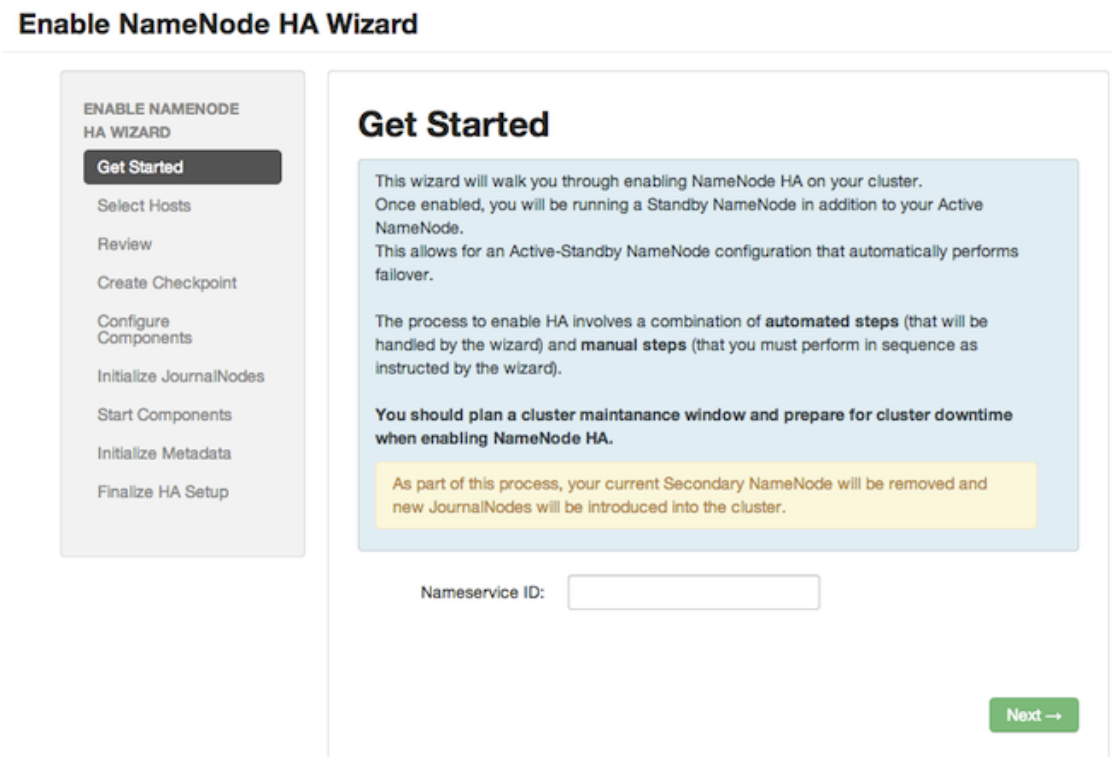
Select High Availabilty in the left nav bar to set up NameNode high availability.



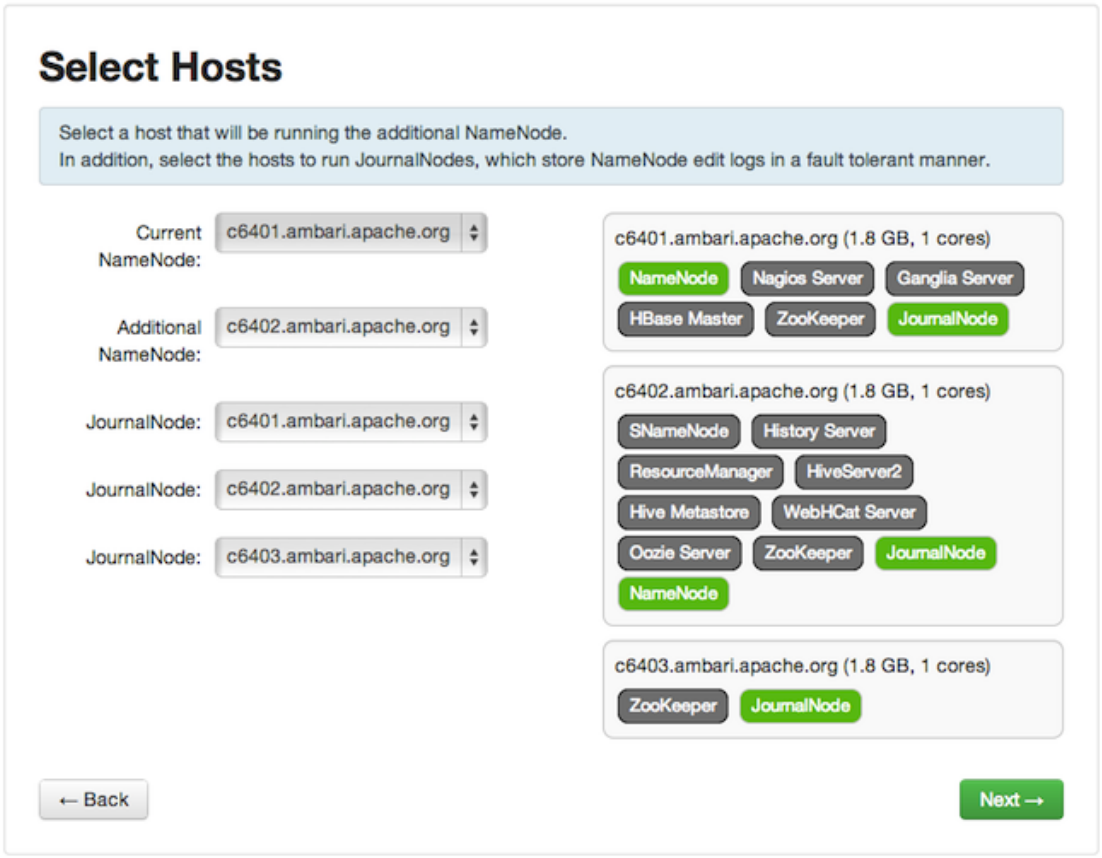
1. Check to make sure you have at least three hosts in your cluster and are running at least three ZooKeeper servers.

2. Click **Enable NameNode HA** and follow the **Enable NameNode HA Wizard**. The wizard describes a set of automated and manual steps you must take to set up NameNode high availability.

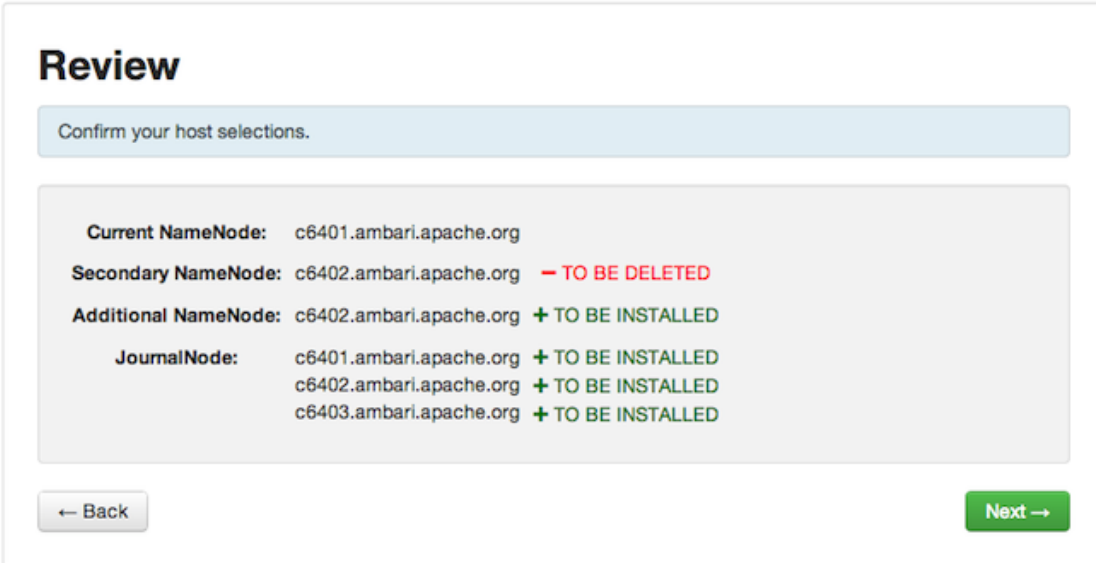
3. **Get Started**: This step gives you an overview of the process and allows you to select a Nameservice ID. You use this Nameservice ID instead of the NameNode FQDN once HA has been set up. Click **Next** to proceed.



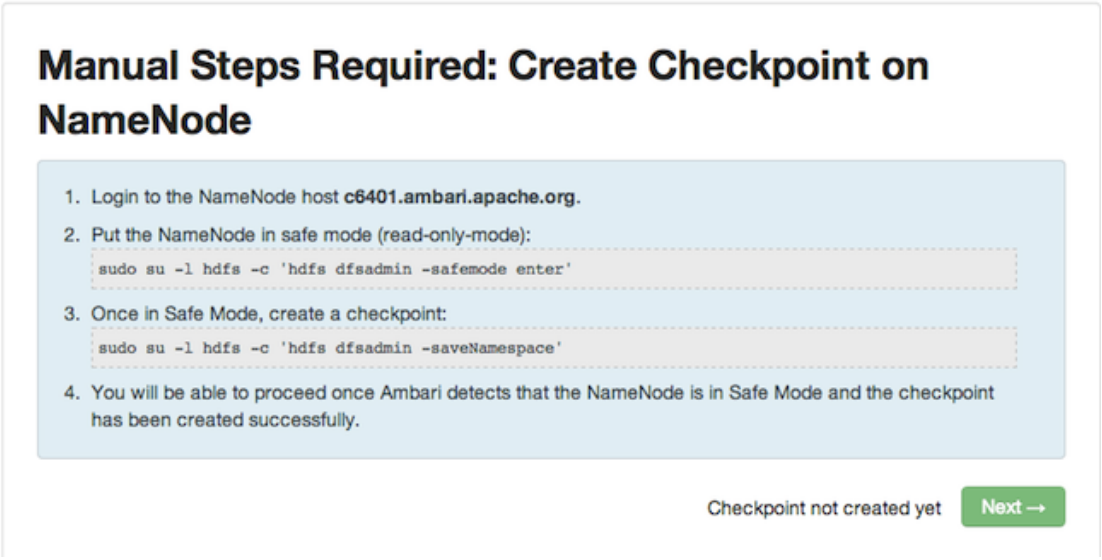
4. **Select Hosts**: Select a host for the additional NameNode and the JournalNodes. The wizard suggests options, but you can adjust using the dropdown lists. Click **Next** to proceed.



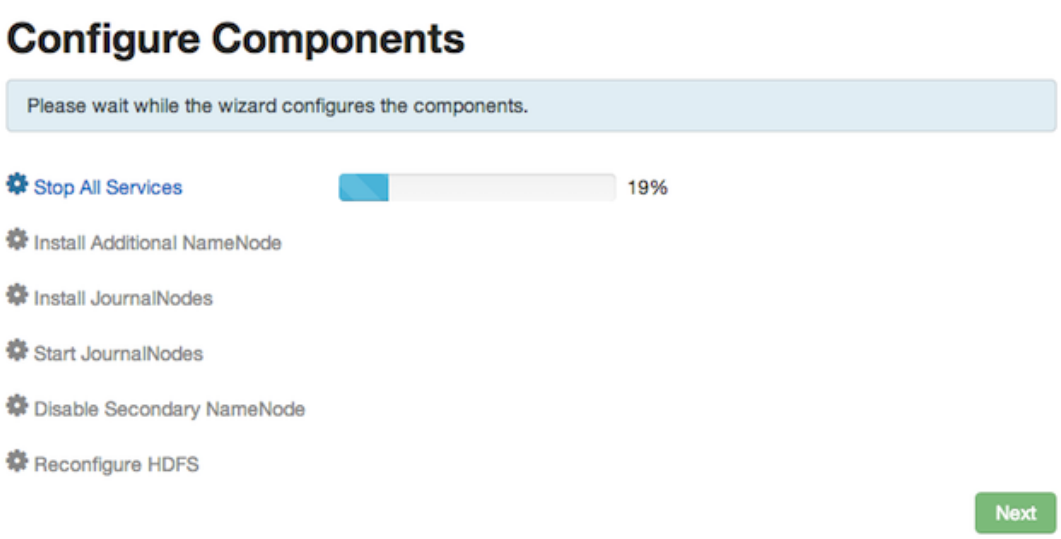
5. **Review**: Confirm your host selections and click **Next**.



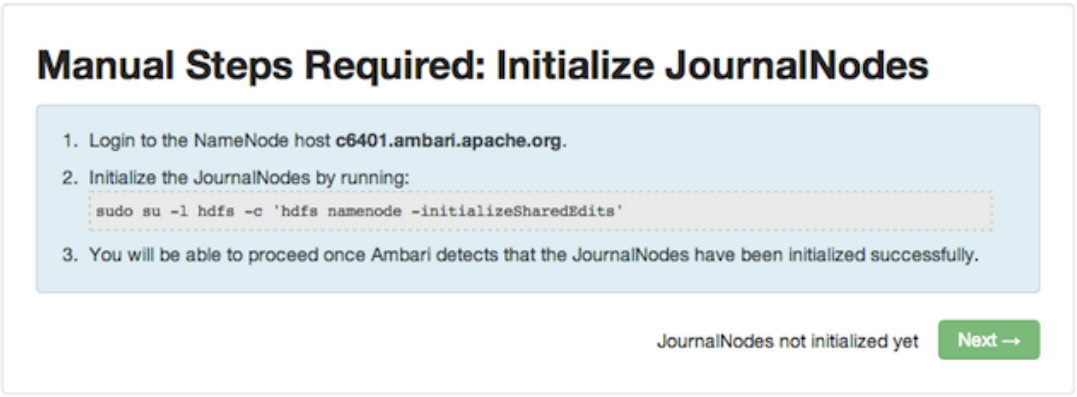
6. **Create Checkpoints**: Follow the instructions in the step. You need to login to your **current** NameNode host to run the commands to put your NameNode into safe mode and create a checkpoint. When Ambari detects success, the message on the bottom of the window changes. Click **Next**.



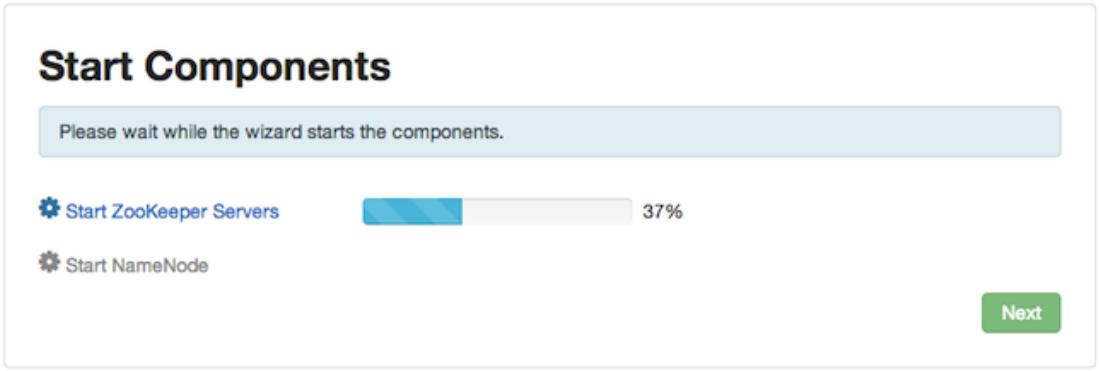
7. **Configure Components**: The wizard configures your components, displaying progress bars to let you track the steps. Click **Next** to continue.



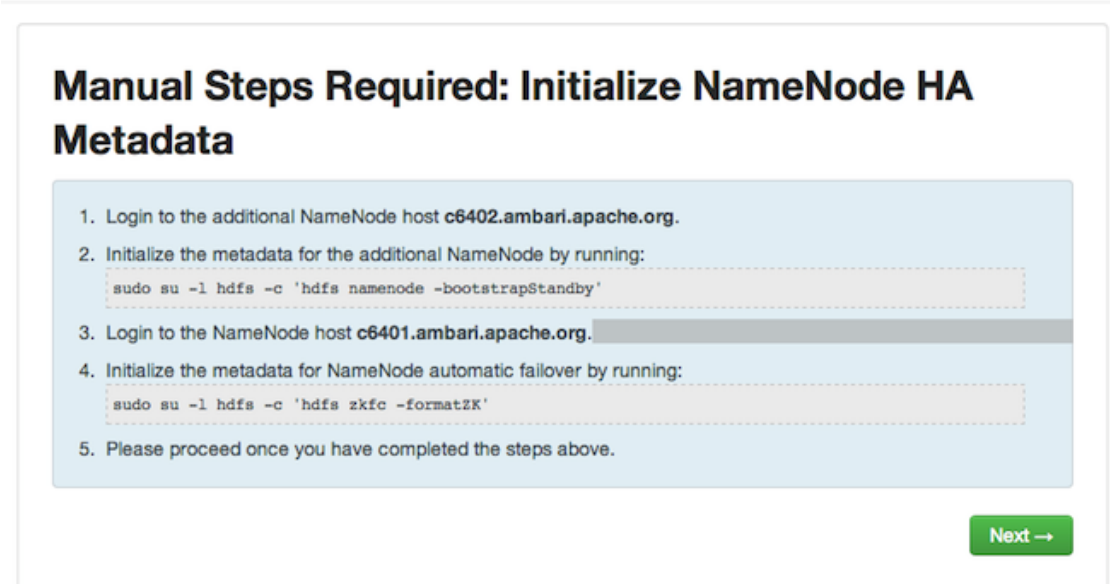
8. **Initialize JournalNodes**: Follow the instructions in the step. You need to login to your **current** NameNode host to run the command to initialize the JournalNodes. When Ambari detects success, the message on the bottom of the window changes. Click **Next**.



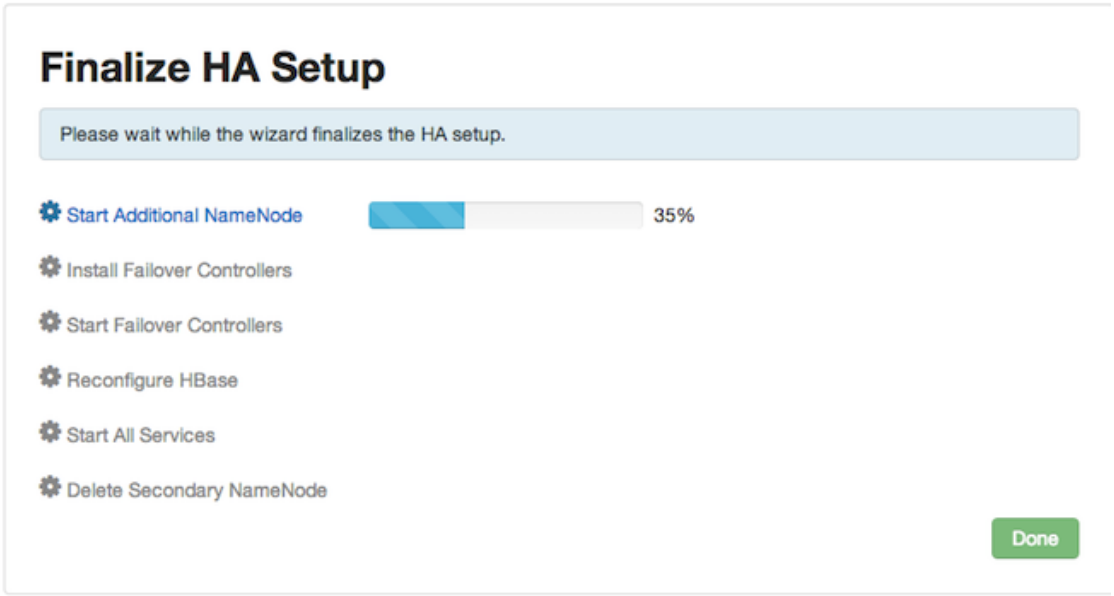
9. **Start Components**: The wizard starts the ZooKeeper servers and the NameNode, displaying progress bars to let you track the steps. Click **Next** to continue.



10. **Initialize Metadata**: Follow the instructions in the step. For this step you must login to both the **current** NameNode and the **additional** NameNode. Make sure you are logged into the correct host for each command. Click **Next** when you have completed the two commands. A **Confirmation** popup appears to remind you that you must do both steps. Click **OK** to confirm.



11. **Finalize HA Setup**: The wizard the setup, displaying progress bars to let you track the steps. Click **Done** to finish the wizard. After the Ambari Web GUI reloads, you may see some alert notifications. Wait a few minutes until the services come back up. If necessary, restart any components using Ambari Web.



12. If you are using Hive, you need to manually change the Hive Metastore FS root to point to the Nameservice URI instead of the NameNode URI. You created the Nameservice ID in the **Get Started** step.

    a. Check the current FS root. On the Hive host:

    ```
    /usr/lib/hive/bin/metatool -listFSRoot
    ```

    The output might look like this:

    ```
    Listing FS Roots..
    ```

```
hdfs://<namenode-host>:8020/apps/hive/warehouse
```

b. Use this command to change the FS root:

```
$ /usr/lib/hive/bin/metatool -updateLocation <new-location> <old-
location>

For example, where the Nameservice ID is mycluster:
 $ /usr/lib/hive/bin/metatool -updateLocation hdfs://mycluster:8020/apps/
hive/warehouse hdfs://c6401.ambari.apache.org:8020/apps/hive/warehouse
```

The output might look like this:

```
Successfully updated the following locations..
Updated X records in SDS table
```

13. If you are using Oozie, you need to use the Nameservice URI instead of the NameNode URI in your workflow files. For example, where the Nameservice ID is `mycluster`:

```
<workflow-app xmlns="uri:oozie:workflow:0.2" name="map-reduce-wf">
    <start to="mr-node"/>
    <action name="mr-node">
        <map-reduce>
            <job-tracker>${jobTracker}</job-tracker>
            <name-node>hdfs://mycluster</name-node>
```

2.6.2.1. Rolling Back NameNode HA

To roll back NameNode HA to the previous non-HA state use the following step-by-step manual process. Some of the steps are optional depending on your installation.

1. Stop HBase
2. Checkpoint the Active NameNode
3. Stop All Services
4. Prepare the Ambari Server Host for Rollback
5. Restore the HBase Configuration
6. Delete ZK Failover Controllers
7. Modify HDFS Configurations
8. Recreate the Secondary NameNode
9. Re-enable the Secondary NameNode
10. Delete All JournalNodes
11. Delete the Additional NameNode
12. Verify the HDFS Components
13. Start HDFS

2.6.2.1.1. Stop HBase

1. From Ambari Web, go to the Services view and select HBase.

2. Click **Stop** on the Management Header.
3. Wait until HBase has stopped completely before continuing.

2.6.2.1.2. Checkpoint the Active NameNode

If HDFS has been in use **after** you enabled NameNode HA, but you wish to revert back to a non-HA state, you must checkpoint HDFS state before proceeding with the rollback.

If the **Enable NameNode HA** wizard failed and you need to revert back, you can skip this step and move on to Stop All Services.

- If Kerberos security has **not** been enabled on the cluster:

  On the Active NameNode host, execute the following commands to save the namespace. You must be the HDFS service user (*$HDFS_USER*) to do this.

  ```
  sudo su -l $HDFS_USER -c 'hdfs dfsadmin -safemode enter'
  sudo su -l $HDFS_USER -c 'hdfs dfsadmin -saveNamespace'
  ```

- If Kerberos security **has** been enabled on the cluster:

  ```
  sudo su -l $HDFS_USER -c 'kinit -kt /etc/security/keytabs/nn.service.keytab
   nn/$HOSTNAME@$REALM;hdfs dfsadmin -safemode enter'
  sudo su -l $HDFS_USER -c 'kinit -kt /etc/security/keytabs/nn.service.keytab
   nn/$HOSTNAME@$REALM;hdfs  dfsadmin -saveNamespace'
  ```

  Where *$HDFS_USER* is the HDFS service user, *$HOSTNAME* is the Active NameNode hostname, and *$REALM* is your Kerberos realm.

2.6.2.1.3. Stop All Services

Use the Services view in Ambari Web and click **Stop All** in the Services navigation panel. You must wait until all the services are completely stopped.

2.6.2.1.4. Prepare the Ambari Server Host for Rollback

Log into the Ambari server host and set the following environment variables to prepare for the rollback procedure:

Table 2.6. Set Environment Variables

Variable	Value
export AMBARI_USER=*AMBARI_USERNAME*	Substitute the value of the administrative user for Ambari Web. The default value is `admin`.
export AMBARI_PW=*AMBARI_PASSWORD*	Substitute the value of the administrative password for Ambari Web. The default value is `admin`.
export AMBARI_PORT=*AMBARI_PORT*	Substitute the Ambari Web port. The default value is `8080`.
export AMBARI_PROTO=*AMBARI_PROTOCOL*	Substitute the value of the protocol for connecting to Ambari Web. Options are `http` or `https`. The default value is `http`.
export CLUSTER_NAME=*CLUSTER_NAME*	Substitute the name of your cluster, set during the Ambari Install Wizard process. For example: `mycluster`.
export NAMENODE_HOSTNAME=*NN_HOSTNAME*	Substitute the FQDN of the host for the non-HA NameNode. For example: `nn01.mycompany.com`.

Variable	Value
export ADDITIONAL_NAMENODE_HOSTNAME=*ANN_HOSTNAME*	Substitute the FQDN of the host for the additional NameNode in your HA setup.
export SECONDARY_NAMENODE_HOSTNAME=*SNN_HOSTNAME*	Substitue the FQDN of the host for the Secondary NameNode for the non-HA setup.
export JOURNALNODE1_HOSTNAME=*JOUR1_HOSTNAME*	Substitute the FQDN of the host for the first Journal Node.
export JOURNALNODE2_HOSTNAME=*JOUR2_HOSTNAME*	Substitute the FQDN of the host for the second Journal Node.
export JOURNALNODE3_HOSTNAME=*JOUR3_HOSTNAME*	Substitute the FQDN of the host for the third Journal Node.

Important

Double check that these environment variables are set correctly.

2.6.2.1.5. Restore the HBase Configuration

If you have installed HBase, you may need to restore a configuration to its pre-HA state.

1. To check if your current HBase configuration needs to be restored, on the Ambari Server host:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p
 $AMBARI_PW -port $AMBARI_PORT get localhost $CLUSTER_NAME hbase-site
```

   Where the environment variables you set up before substitute for the variable names.

   Look for the configuration property `hbase.rootdir`. If the value is set to the NameService ID you set up using the **Enable NameNode HA** wizard, you need to revert the `hbase-site` configuration set up back to non-HA values. If it points instead to a specific NameNode host, it does not need to be rolled back and you can go on to Delete ZK Failover Controllers.

   For example:

```
"hbase.rootdir":"hdfs://name-service-id:8020/apps/hbase/data"
The hbase.rootdir property points to the NameService ID and the value needs
 to be rolled back

"hbase.rootdir":"hdfs://nn01.mycompany.com:8020/apps/hbase/data"
The hbase.rootdir property points to a specific NameNode host and not a
 NameService ID. This does not need to be rolled back.
```

2. If you need to roll back the `hbase.rootdir` value, on the Ambari Server host, use the `config.sh` script to make the necessary change:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p
 $AMBARI_PW -port $AMBARI_PORT set localhost $CLUSTER_NAME hbase-site hbase.
rootdir hdfs://${NAMENODE_HOSTNAME}:8020/apps/hbase/data
```

   Where the environment variables you set up before substitute for the variable names.

3. Verify that the `hbase.rootdir` property has been restored properly. On the Ambari Server host:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p
 $AMBARI_PW -port $AMBARI_PORT get localhost $CLUSTER_NAME hbase-site
```

The `hbase.rootdir` property should now be set to the NameNode hostname, not the NameService ID.

2.6.2.1.6. Delete ZK Failover Controllers

You may need to delete ZK Failover Controllers.

1. To check if you need to delete ZK Failover Controllers, on the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i ${AMBARI_PROTO}://localhost:
${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/host_components?HostRoles/
component_name=ZKFC
```

   If this returns an empty `items` array, you can go on to Modify HDFS Configuration. Otherwise you must use the DELETE commands below.

2. To delete all ZK Failover Controllers, on the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X DELETE ${AMBARI_PROTO}:/
/localhost:${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/hosts/
${NAMENODE_HOSTNAME}/host_components/ZKFC
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X DELETE ${AMBARI_PROTO}:/
/localhost:${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/hosts/
${ADDITIONAL_NAMENODE_HOSTNAME}/host_components/ZKFC
```

3. Verify that the ZK Failover Controllers have been deleted. On the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i ${AMBARI_PROTO}://localhost:
${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/host_components?HostRoles/
component_name=ZKFC
```

   This command should return an empty `items` array.

2.6.2.1.7. Modify HDFS Configurations

You may need to modify your `hdfs-site` configuration and/or your `core-site` configuration.

1. To check if you need to modify your `hdfs-site` configuration, on the Ambari Server host:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p
 $AMBARI_PW -port $AMBARI_PORT get localhost $CLUSTER_NAME hdfs-site
```

   If you see **any** of the following properties, you must delete them from your `hdfs-site` configuration.

   - `dfs.nameservices`
   - `dfs.client.failover.proxy.provider.${NAMESERVICE_ID}`
   - `dfs.ha.namenodes.${NAMESERVICE_ID}`
   - `dfs.ha.fencing.methods`
   - `dfs.ha.automatic-failover.enabled`

- `dfs.namenode.http-address.${NAMESERVICE_ID}.nn1`
- `dfs.namenode.http-address.${NAMESERVICE_ID}.nn2`
- `dfs.namenode.rpc-address.${NAMESERVICE_ID}.nn1`
- `dfs.namenode.rpc-address.${NAMESERVICE_ID}.nn2`
- `dfs.namenode.shared.edits.dir`
- `dfs.journalnode.edits.dir`
- `dfs.journalnode.http-address`
- `dfs.journalnode.kerberos.internal.spnego.principal`
- `dfs.journalnode.kerberos.principal`
- `dfs.journalnode.keytab.file`

  Where `${NAMESERVICE_ID}` is the NameService ID you created when you ran the **Enable NameNode HA** wizard.

2. To delete these properties, execute the following **for each property** you found. On the Ambari Server host:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -
p $AMBARI_PW -port $AMBARI_PORT delete localhost $CLUSTER_NAME hdfs-
site property_name
```

   Where you replace *`property_name`* with the name of **each** of the properties to be deleted.

3. Verify that all of the properties have been deleted. On the Ambari Server host:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p
 $AMBARI_PW -port $AMBARI_PORT get localhost $CLUSTER_NAME hdfs-site
```

   None of the properties listed above should be present.

4. To check if you need to modify your `core-site` configuration, on the Ambari Server host:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p
 $AMBARI_PW -port $AMBARI_PORT get localhost $CLUSTER_NAME core-site
```

5. If you see the property `ha.zookeeper.quorum`, it must be deleted. On the Ambari Server host:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p
 $AMBARI_PW -port $AMBARI_PORT delete localhost $CLUSTER_NAME core-site ha.
zookeeper.quorum
```

6. If the property `fs.defaultFS` is set to the NameService ID, it must be reverted back to its non-HA value. For example:

```
“fs.defaultFS” : “hdfs://name-service-id”
```

```
The property fs.defaultFS needs to be modified as it points to a NameService
 ID

“fs.defaultFS” : “hdfs://nn01.mycompany.com”
The property fs.defaultFS does not need to be changed as it points to a
 specific NameNode and not a NameService ID
```

7. To revert the property `fs.defaultFS` to the NameNode host value, on the Ambari Server host:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p
 $AMBARI_PW -port $AMBARI_PORT set localhost $CLUSTER_NAME core-site fs.
defaultFS hdfs://${NAMENODE_HOSTNAME}
```

8. Verify that the `core-site` properties are now properly set. On the Ambari Server host:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p
 $AMBARI_PW -port $AMBARI_PORT get localhost $CLUSTER_NAME core-site
```

   The property `fs.defaultFS` should be set to point to the NameNode host and the property `ha.zookeeper.quorum` should not be there.

2.6.2.1.8. Recreate the Secondary NameNode

You may need to recreate your Secondary NameNode.

1. To check to see if you need to recreate the Secondary NameNode, on the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X GET ${AMBARI_PROTO}://localhost:
${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/host_components?HostRoles/
component_name=SECONDARY_NAMENODE
```

   If this returns an empty `items` array, you must recreate your Secondary NameNode. Otherwise you can go on to Re-enable Secondary NameNode.

2. Recreate your Secondary NameNode. On the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X POST -d '{"host_components" :
 [{"HostRoles":{"component_name":"SECONDARY_NAMENODE"}}] }'
 ${AMBARI_PROTO}://localhost:${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/
hosts?Hosts/host_name=${SECONDARY_NAMENODE_HOSTNAME}
```

3. Verify that the Secondary NameNode now exists. On the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X GET ${AMBARI_PROTO}://localhost:
${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/host_components?HostRoles/
component_name=SECONDARY_NAMENODE
```

   This should return a non-empty `items` array containing the Secondary NameNode.

2.6.2.1.9. Re-enable the Secondary NameNode

To re-enable the Secondary NameNode, on the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X PUT -d '{"RequestInfo":
{"context":"Enable Secondary NameNode"},"Body":{"HostRoles":
{"state":"INSTALLED"}}}' ${AMBARI_PROTO}://localhost:${AMBARI_PORT}/api/v1/
```

```
clusters/${CLUSTER_NAME}/hosts/${SECONDARY_NAMENODE_HOSTNAME}/host_components/
SECONDARY_NAMENODE
```

- If this returns 200, go to Delete All JournalNodes.
- If this returns 202, wait a few minutes and run the following on the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X GET "${AMBARI_PROTO}://localhost:
${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/host_components?HostRoles/
component_name=SECONDARY_NAMENODE&fields=HostRoles/state"
```

When `"state" : "INSTALLED"` is in the response, go on to the next step.

2.6.2.1.10. Delete All JournalNodes

You may need to delete any JournalNodes.

1. To check to see if you need to delete JournalNodes, on the Ambari Server host:

```
 curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X GET ${AMBARI_PROTO}://localhost:
${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/host_components?HostRoles/
component_name=JOURNALNODE
```

   If this returns an empty `items` array, you can go on to Delete Additional NameNode. Otherwise you must delete the JournalNodes.

2. To delete the JournalNodes, on the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X DELETE ${AMBARI_PROTO}:/
/localhost:${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/hosts/
${JOURNALNODE1_HOSTNAME}/host_components/JOURNALNODE

curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X DELETE ${AMBARI_PROTO}:/
/localhost:${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/hosts/
${JOURNALNODE2_HOSTNAME}/host_components/JOURNALNODE


curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X DELETE ${AMBARI_PROTO}:/
/localhost:${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/hosts/
${JOURNALNODE3_HOSTNAME}/host_components/JOURNALNODE
```

3. Verify that all the JournalNodes have been deleted. On the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X GET ${AMBARI_PROTO}://localhost:
${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/host_components?HostRoles/
component_name=JOURNALNODE
```

   This should return an empty `items` array.

2.6.2.1.11. Delete the Additional NameNode

You may need to delete your Additional NameNode.

1. To check to see if you need to delete your Additional NameNode, on the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X GET ${AMBARI_PROTO}://localhost:
${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/host_components?HostRoles/
component_name=NAMENODE
```

If the `items` array contains two NameNodes, the Additional NameNode must be deleted.

2. To delete the Additional NameNode that was set up for HA, on the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X DELETE ${AMBARI_PROTO}:/
/localhost:${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/hosts/
${ADDITIONAL_NAMENODE_HOSTNAME}/host_components/NAMENODE
```

3. Verify that the Additional NameNode has been deleted:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X GET ${AMBARI_PROTO}://localhost:
${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/host_components?HostRoles/
component_name=NAMENODE
```

This should return an `items` array that shows only one NameNode.

2.6.2.1.12. Verify your HDFS Components

Make sure you have the correct componets showing in HDFS.

1. In Ambari Web, go to the Services view and select HDFS from the Services navigation panel.

2. Check the Summary panel and make sure that the first three lines look like this:

   - NameNode
   - SNameNode
   - DataNodes

   You should **not** see any line for JournalNodes.

2.6.2.1.13. Start HDFS

1. On the HDFS page of the Services view, click **Start** in the Management Header to start up HDFS. Wait until the service is fully started and has passed the service check.

   If HDFS does not start, you may need to go through the previous steps again.

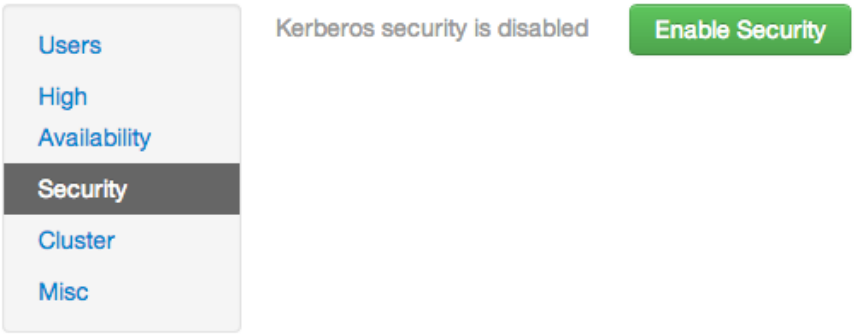
2. Start all the other services by using **Start All** in the Services navigation panel.

2.6.3. Enabling Kerberos Security

To turn on Kerberos-based security you must:

1. Have already set up Kerberos for your cluster. For more information on setting up Kerberos, see Setting Up Kerberos for Use with Ambari

2. Click **Enable Security** and follow the Enable Security Wizard.



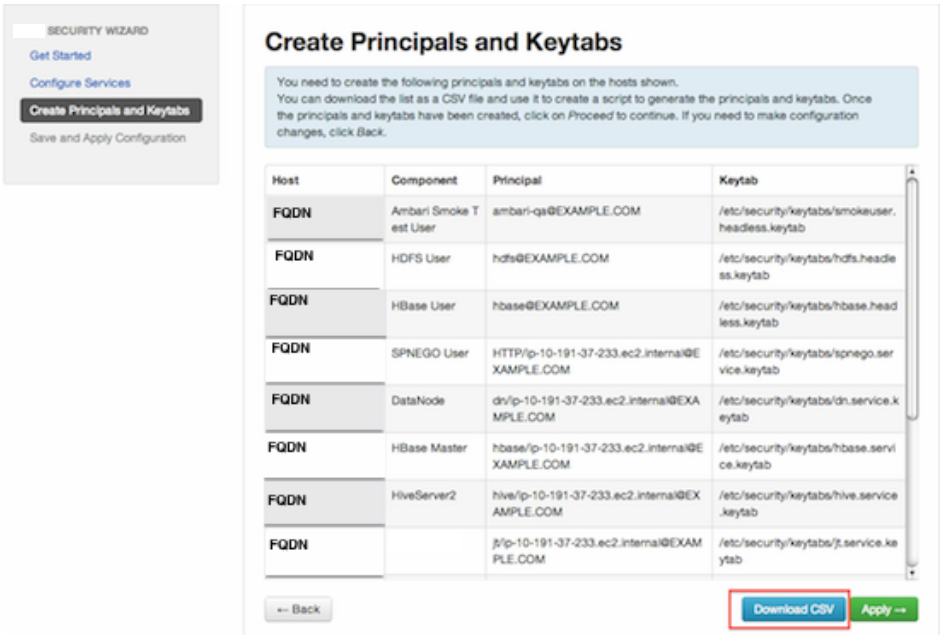
a. **Get Started**: This step just reminds you that you need to set up Kerberos before you start.

b. **Configure Services**: This step asks you for your Kerberos information: principals and paths to keytabs, path to your Kerberos tools, realm names and so on. For more information about a specific field, hover over it, and a popup with a definition appears.

c. **Create Principals and Keytabs**: Use this step to check that all your information is correct. Click **Back** to make any changes. Click **Apply** when you are satisfied with the assignments.

Note

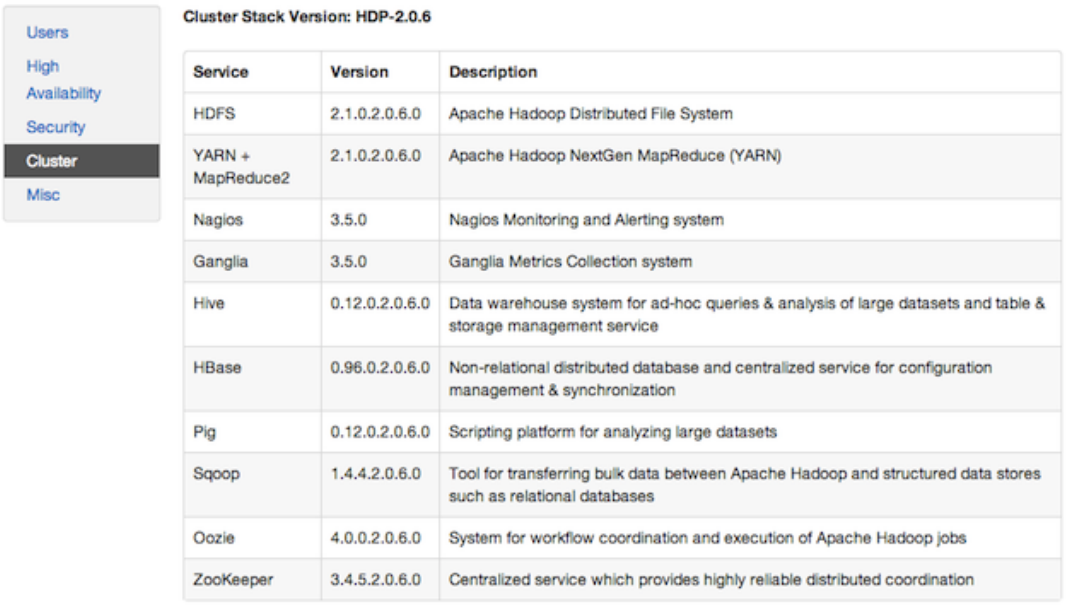
If you have a large cluster, you may want to go to the **Create Principals and Keytabs** step first. Step through the wizard accepting the defaults to get to the appropriage page. On the page, use the **Download CSV** button to get a list of all the necessary principals and keytabs in CSV form, which can be used to set up a script. The list includes hostname, principal description, principal name, keytab user, keytab group, keytab permissions, absolute keytab path, and keytab filename.



d. **Save and Apply Configuration**: This step displays a bar showing the progress of integrating the Kerberos information into your Ambari Server.

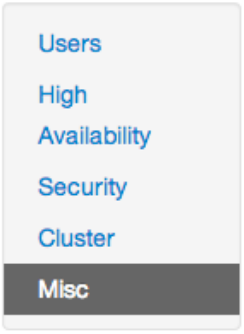
2.6.4. Checking Stack and Component Versions

To check which version of the Stack you are using and to see the component versions that are included in that Stack, click **Cluster**.



2.6.5. Checking Service User Accounts and Groups

To check the service user accounts and groups that have been assigned to various Hadoop services, click the **Misc** tab.



Service Users and Groups

Name	Value
Proxy group for Hive, WebHCat, and Oozie	users
HDFS User	hdfs
MapReduce User	mapred
YARN User	yarn
HBase User	hbase
Hive User	hive
HCat User	hcat
WebHCat User	hcat
Oozie User	oozie
ZooKeeper User	zookeeper
Ganglia User	nobody
Nagios User	nagios
Nagios Group	nagios
Smoke Test User	ambari-qa
Hadoop Group	hadoop

3. Navigating Ambari Web for Hadoop 1.x

This section gives you an overview of using the Ambari Web GUI for monitoring and managing your Hadoop 1 cluster.

3.1. The Navigation Header

At the top of every screen is the Navigation Header.

This header appears in all views in Ambari Web. Use this bar to move from view to view. The active view is indicated with a slightly darker gray.

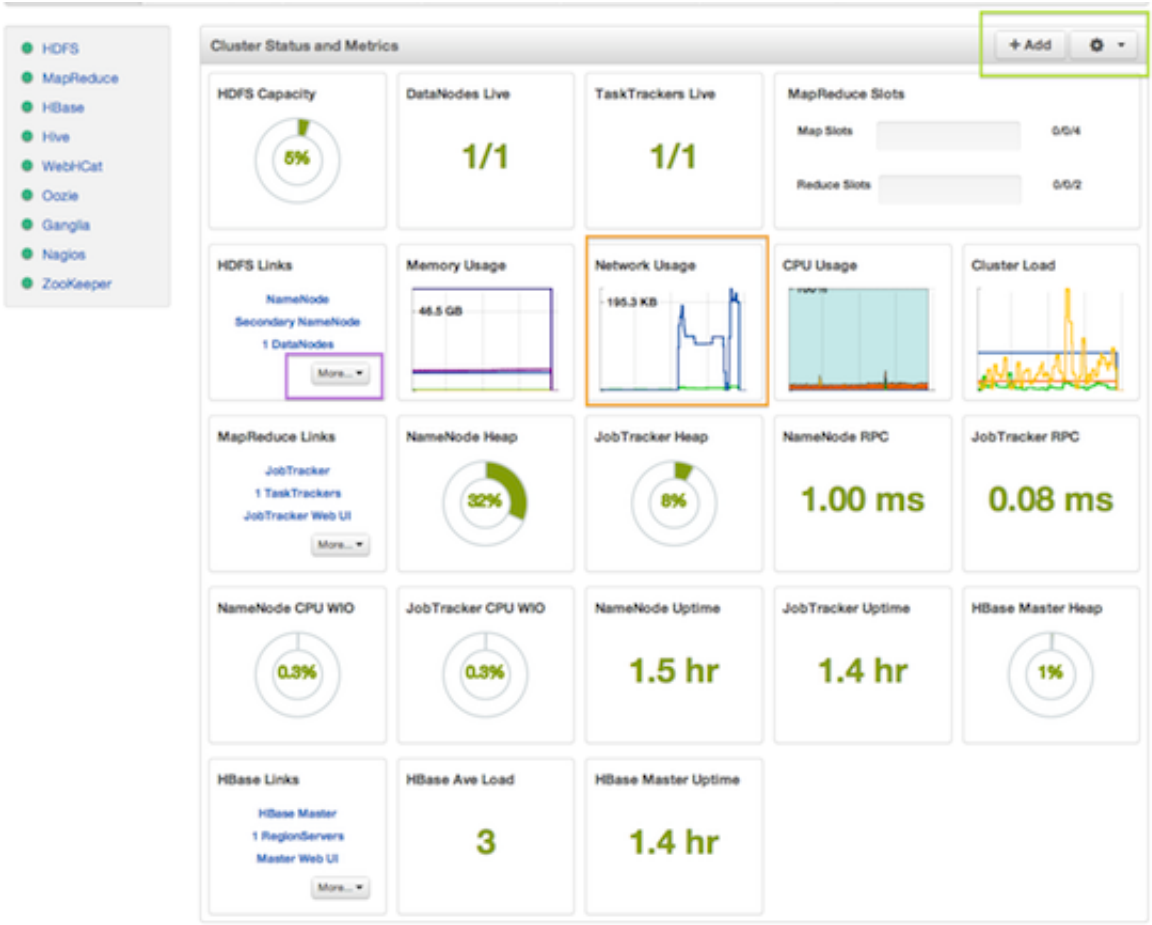
3.2. The Dashboard View

When you open Ambari Web, you are placed in the **Dashboard** view. This view is divided into two versions:

- The Widget version
- The Classic version

3.2.1. The Widget Version

The **Dashboard** Widget version opens first. This view gives you a customizable overview of the state of your cluster as a whole.



The Widget version is divided into two main sections:

- Services Summary
- Cluster Status and Metrics

3.2.1.1. Services Summary

On the left border of the screen is the **Services** summary. You can use this section to get an overall view of the status of your services. In the image above [38], notice that each service name has a green dot next to it. Use the dot colors give you a quick overview of the service's status.

Table 3.1. Service Status

Color	Name	Status
	Solid Green	All masters are running
	Blinking Green	Starting up
	Solid Red	At least one master is down

Color	Name	Status
	Blinking Red	Stopping

Click the service name to open the **Services** screen, where you can see more detailed information on each service.

3.2.1.2. Cluster Status and Metrics

On the main section of the screen [38], a customizable set of widget tiles presents information on the status of your cluster. There are simple pie and bar charts, more complex usage and load charts, and sets of links to additional data sources, as well as status information like uptime and average RPC queue wait times.

Table 3.2. Widget Interactions

To:	Do:
Move widgets around on the screen	Drag and drop to desired position
See more detailed information	Hover over the widget box

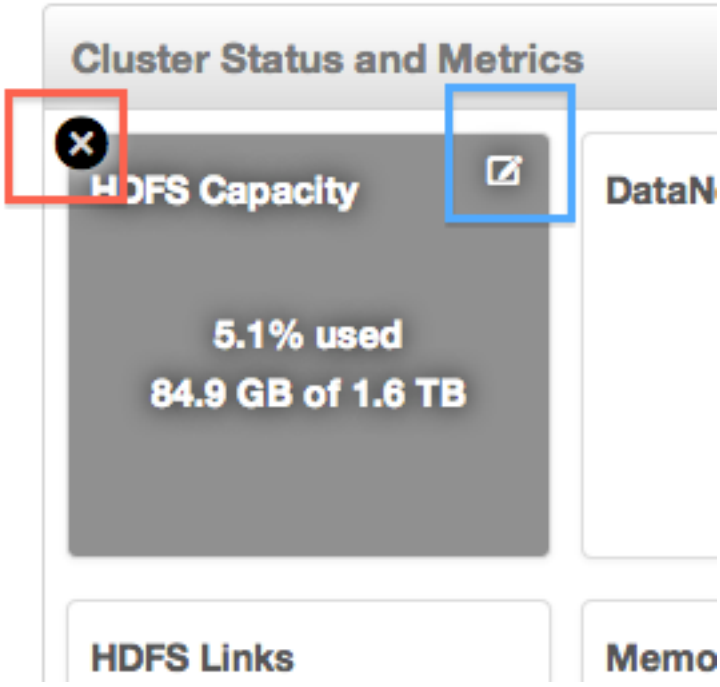
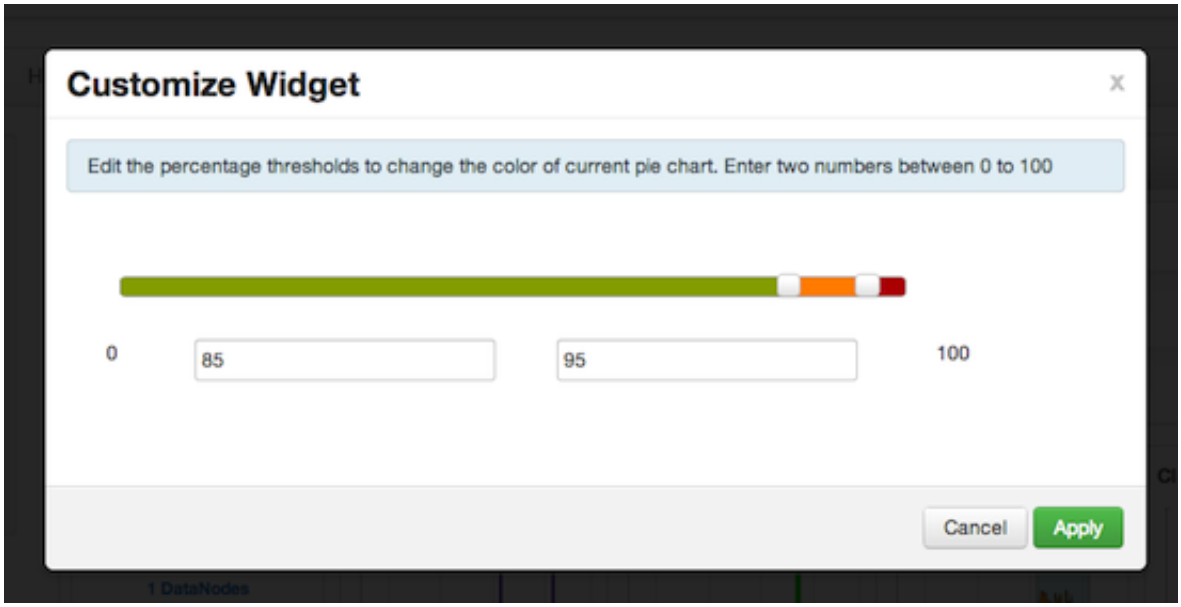


Table 3.3. Widget Interactions 2

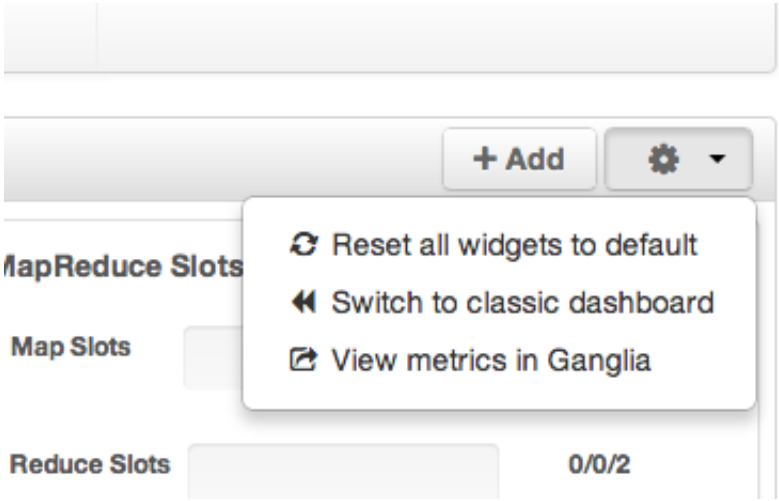
To:	Do:
Delete a widget	Click the X, marked in red
Edit a widget	Click the small edit icon, marked in blue. The **Customize Widget** popup appears.



Follow the instructions in the popup to customize the widget display. In this case, you can adjust the thresholds at which the **HDFS Capacity** bar chart changes color, from green to orange to red. Click **Apply** to save your changes. Not all widgets have an edit icon.

Table 3.4. Widget Interactions 3

To:	Do:
Add back a deleted widget	Click **+Add**, outlined in green [38], and use the checkbox to select the widget you want from the dropdown menu
Use quick links to other information, like thread stacks, logs and native component GUIs	Click **More**, outlined in purple [38], and select from the dropdown menu
Zoom into more complex charts, like the Network Usage chart, outlined in orange [38]	Click the chart. A larger version pops up. You can choose to add and remove information from the chart by selecting or deselecting the metric in the legend
Switch to the Classic version, return widgets to default setting, or view metrics in the native Ganglia Server interface	Use the gear icon



3.2.2. The Classic Version

The **Dashboard** Classic version provides a slightly different view of high-level cluster information.

The Classic version is also divided into two main sections:

- Services Summary
- Cluster Metrics

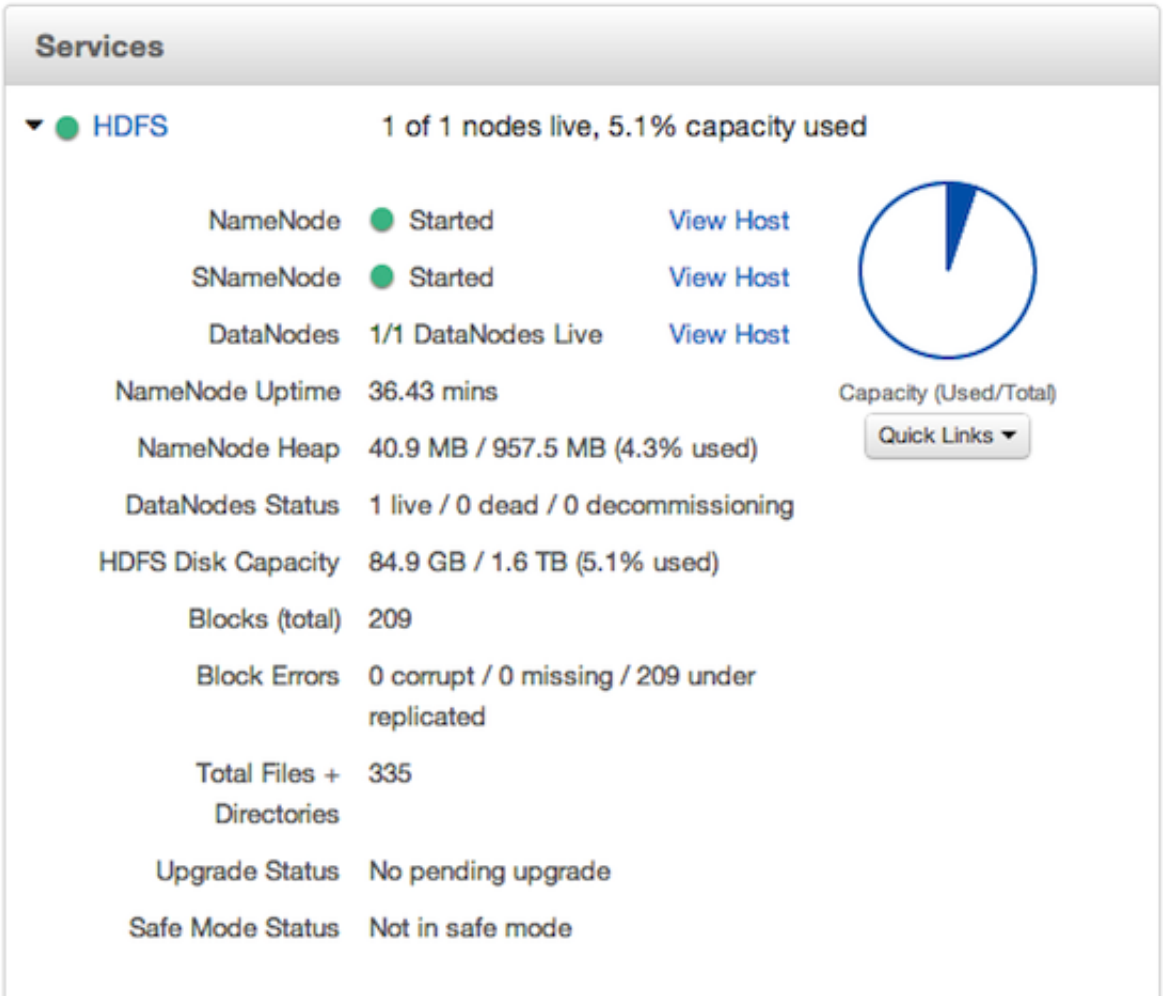
3.2.2.1. Services Summary

On the left side of the screen is the **Services** summary. Use this section to get an overall view of the status of your services. In the image above [42], notice that **HDFS**, **MapReduce**, and **Oozie** have green dots next to them but that **HBase** and **ZooKeeper** have red dots. The dot colors give you a quick overview of the service's status.

Table 3.5. Service Status

Color	Name	Status
	Solid Green	All masters are running
	Blinking Green	Starting up
	Solid Red	At least one master is down
	Blinking Red	Stopping

Notice also that HDFS, MapReduce, and HBase all have small triangles next to them. These are the services that are deployed in master/slave sets. Click the triangle to open a more detailed picture of the service.



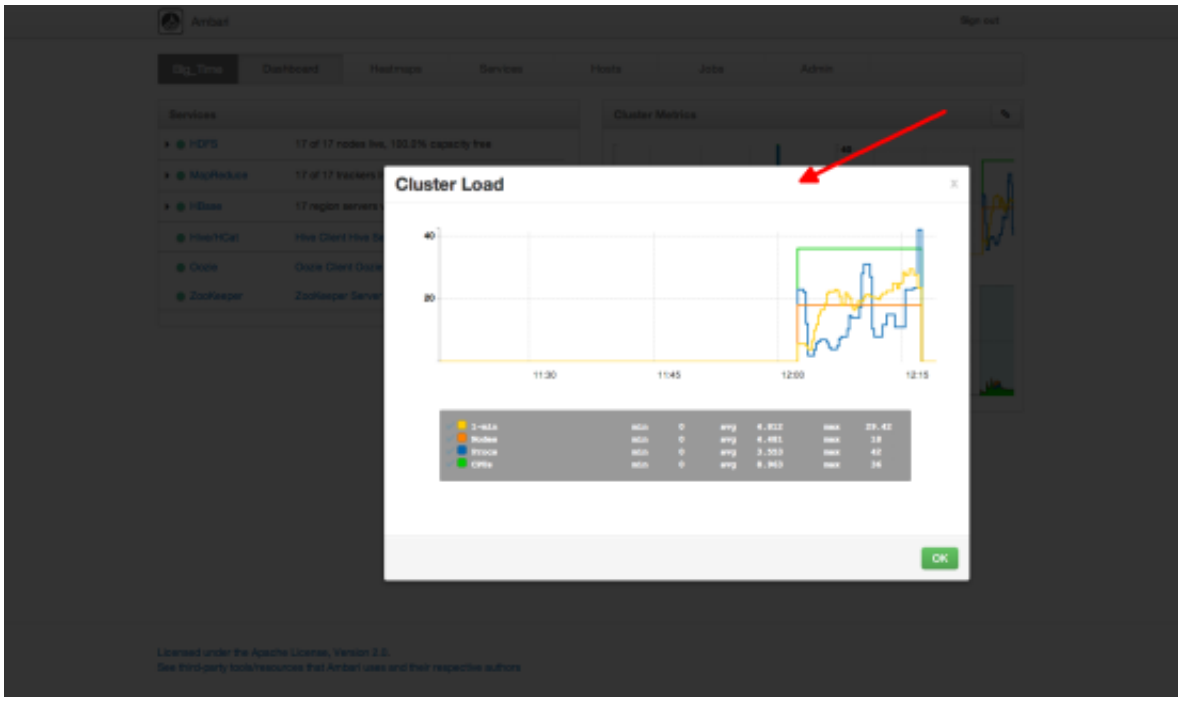
If an alert is raised against the service, a small rectangle showing the number of alerts raised, marked in blue below, appears next to the service name.

Click the service name to open the **Services** screen, where you can see more detailed information on the alert.

3.2.2.2. Cluster Metrics

On the right side of the screen is the **Cluster Metrics** section. This section gives you charts for a snapshot of the important clusterwide metrics: **Network Usage**, **Cluster Load**, **Memory Usage**, and **CPU Usage**. To see a legend for the chart, hover over it. To remove a metric from the chart, click on the legend in the metric to remove the checkmark and deselect it.

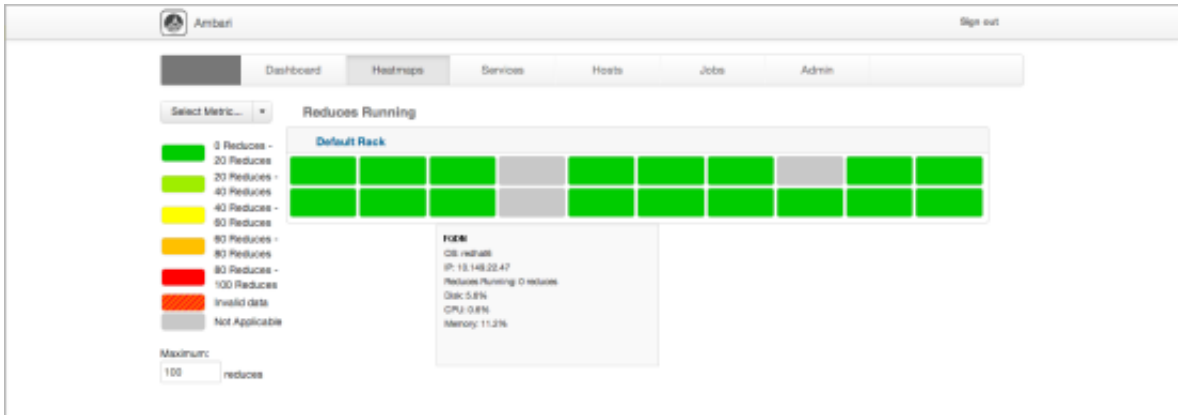
To see a larger view of the chart, click on it. The larger chart pops out.



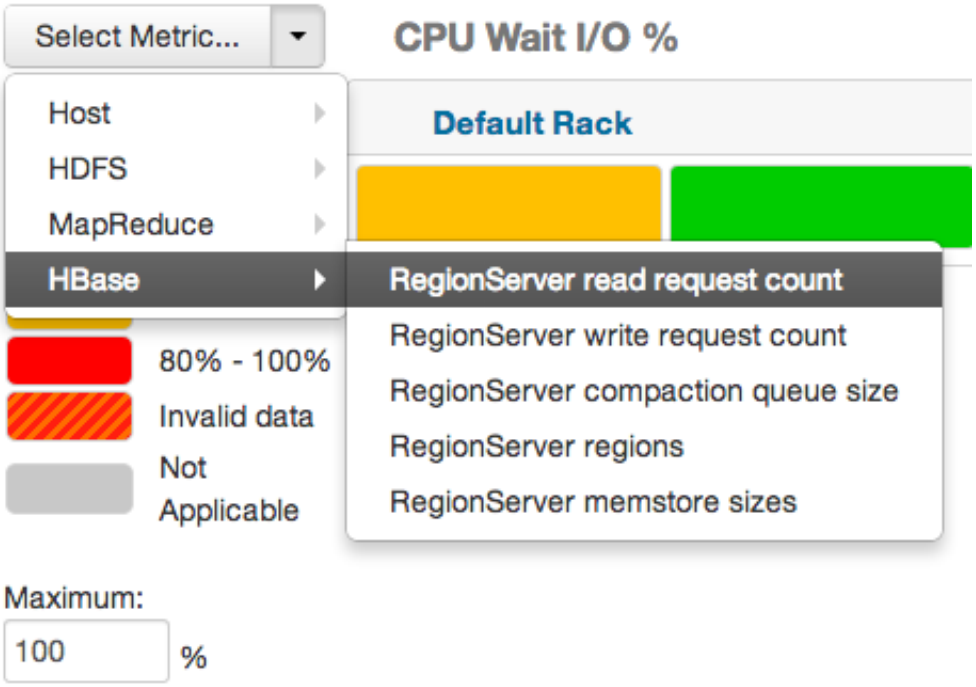
Notice the link symbol on the upper right side of the Cluster Metric section, outlined in blue in the overview screenshot [42] above. This is a link to the GUI for the Ganglia Server itself, where you can find much more detailed information on your cluster. You can also use this to switch back to the Widget version.

3.3. The Heatmaps View

The **Heatmaps** view gives you a graphic representation of the overall utilization of your cluster using simple color coding.



Each host in the cluster is represented by a block. To see more information on a specific host, hover over the block you are interested in, and a popup with key host data appears. The color of the blocks represents usage in an appropriate unit based on a selectable set of metrics. If the data necessary to determine state are not all available, the block is marked as having Invalid data. Changing the default maximum values for the heatmap lets you fine tune the representation. Use the Select Metric dropdown to select the metric type.



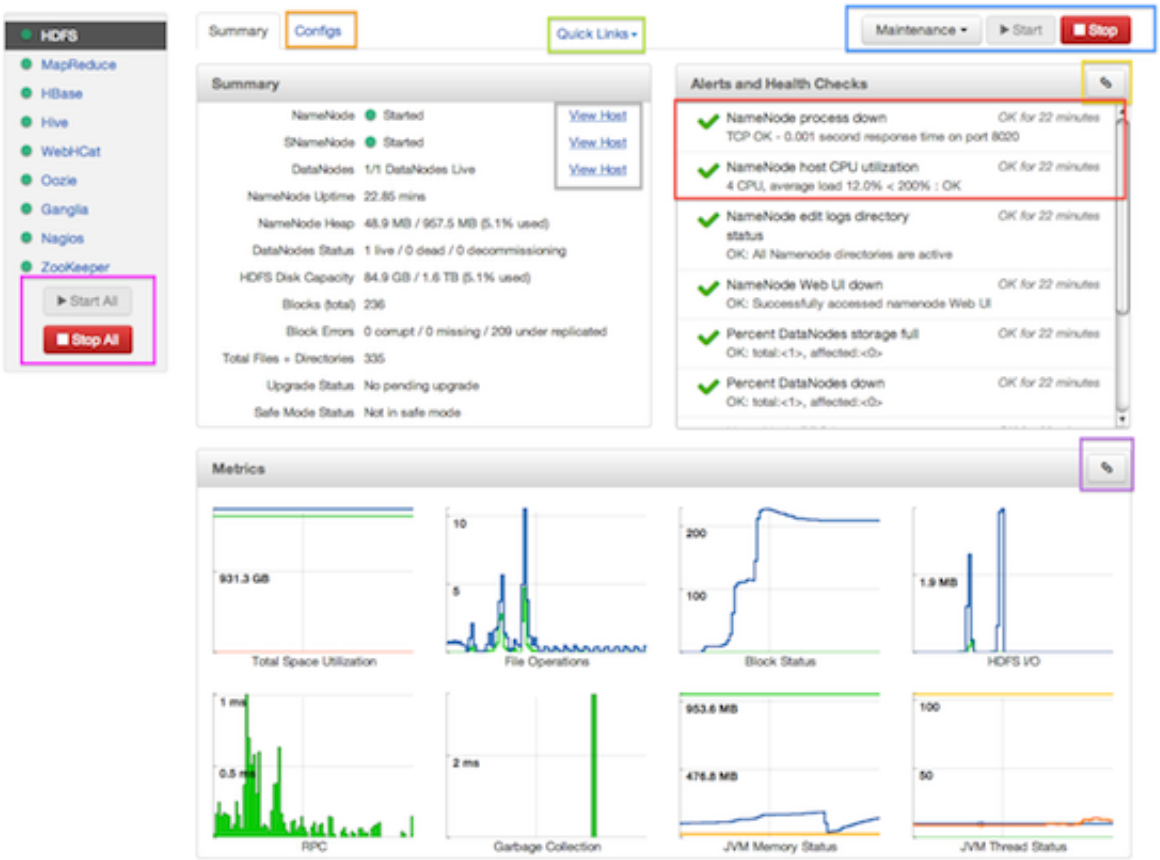
Currently the following metrics are supported:

- Host/Disk Space Used % Uses disk.disk_free and disk.disk_total
- Host/Memory Used %: Uses memory.mem_free and memory.mem_total
- HDFS/Bytes Read: Uses dfs.datanode.bytes_read
- HDFS/Bytes Written: Uses dfs.datanode.bytes_written
- HDFS/Garbage Collection Time: Uses jvm.gcTimeMillis

- HDFS/JVM Heap MemoryUsed: Uses jvm.memHeapUsedM
- MapReduce/Maps Running: Uses mapred.tasktracker.maps_running
- MapReduce/Reduces Running: Uses mapred.tasktracker.reduces_running
- MapReduce/Garbage Collection Time: Uses jvm.gcTimeMillis
- MapReduce/JVM Heap Memory Used: Uses jvm.memHeapUsedM
- HBase/RegionServer read request count: Uses hbase.regionserver.readRequestsCount
- HBase/RegionServer write request count: Uses hbase.regionserver.writeRequestsCount
- HBase/RegionServer compaction queue size: Uses hbase.regionserver.compactionQueueSize
- HBase/RegionServer regions: Uses hbase.regionserver.regions
- HBase/RegionServer memstore sizes: Uses hbase.regionserver.memstoreSizeMB

3.4. The Services View

The **Services** view gives you access to detailed information on each of the services running in your cluster. It also allows you to start and stop the service, run smoke tests, and change configuration details.



The Services view is divided into seven sections:

- Services Navigation
- Services Summary
- Configuration Update
- Quick Links
- Alerts and Health Checks
- Management Header
- Metrics

3.4.1. Services Navigation

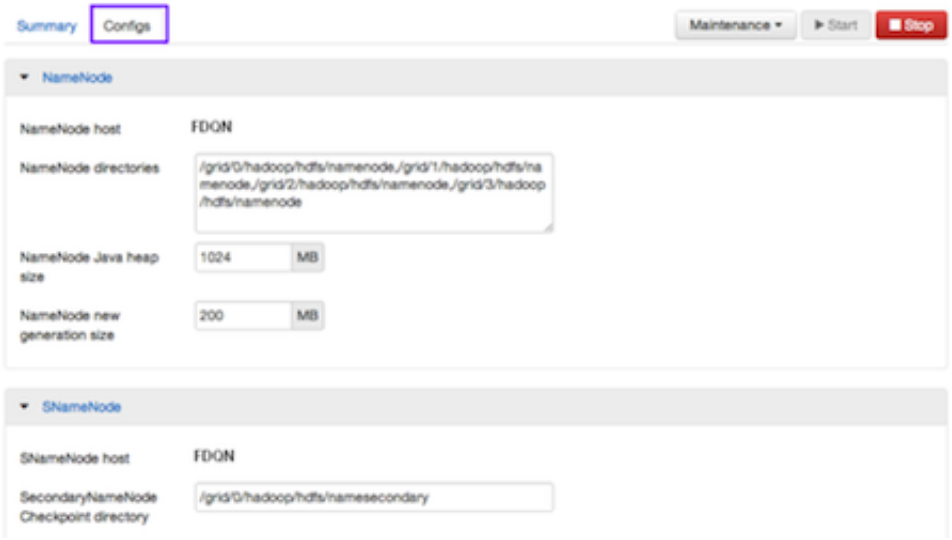
The Services navigation panel on the left gives you a quick look at the status of your services. The color of the dot next to the left of service name tells you the service running state and a small rectangle to the right lets you know if there are any alerts assigned to it. To move the larger screen display from showing information about one service to another, click the service name. To stop or start all of the services at once, use the **Start All** or **Stop All** buttons, outlined in magenta [46].

3.4.2. Services Summary

The Services **Summary** tab displays basic information about the selected service. Use the **View Host** links, marked in gray [46] above to move to the **Host Details** view of the host that is running the service. Clicking **Configs**, outlined in orange [46] above, opens a second tab, the Configuration Update tab.

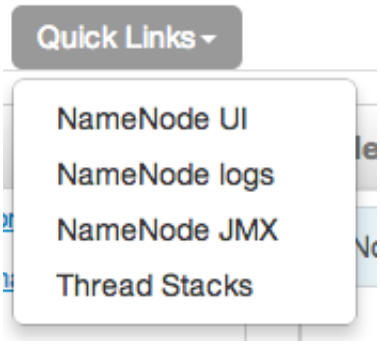
3.4.3. Configs

The Configuration Update tab allows you to update configurations for your service. The screen that appears is familiar from the Install wizard configuration page. Once you have made your changes, use the management header to stop and restart the service.



3.4.4. Quick Links

Quick Links, outlined in light green, takes you to additional sources of information about this service. In the case of HDFS, for example, it contains links to the native NameNode GUI, NameNode logs, the NameNode JMX output, and thread stacks for the service. Not all **Services** pages include Quick Links.

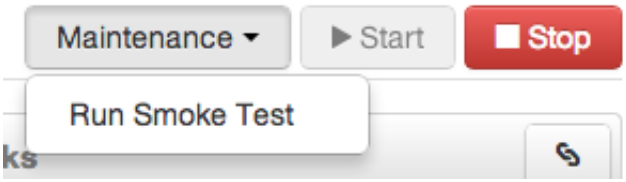


3.4.5. Alerts and Health Checks

The **Alerts and Health Checks** panel displays the results of the health checks performed on your cluster by Nagios. Alerts display a brief summary of the issue and its rating. They are sorted by descending severity, followed by descending time. To access more detailed information, click the link in the upper right corner of the panel, outlined in yellow [46] above. This opens the native Nagios GUI. Use the username and password you set up during installation to log in.

3.4.6. Management Header

The management header, outlined in blue [46], gives you a convenient way to stop and start an individual service. You can also use **Maintenance** to perform smoke tests on the service, for example, if you have updated the service's configuration.



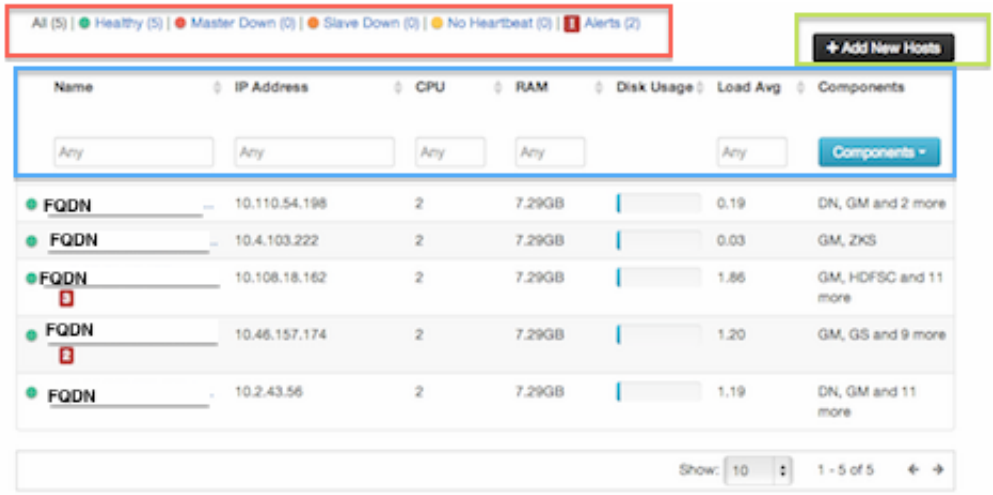
3.4.7. Metrics

The **Metrics** panel displays a set of charts measuring common metrics for the service. Hover above a particular chart to see a legend and click a specific metric type to select or deselect it. Click the chart to see a larger version. To get more information, click the link in the upper right corner of the panel, outlined in purple [46]. This opens the native Ganglia GUI.

3.5. The Hosts View

The Hosts view presents your cluster in terms of the hosts on which the services are running. If your cluster is large, you can use the health filter, outlined in red, to locate only

hosts in that state. The more general filter tool, shown outlined in blue, allows you to select specific types of hosts and to sort your result.



Use the dropdown list in the **Components** column to locate only the hosts that run the specific Master components in which you are interested. After you have made your choice, click the **Apply** button at the bottom of the list.

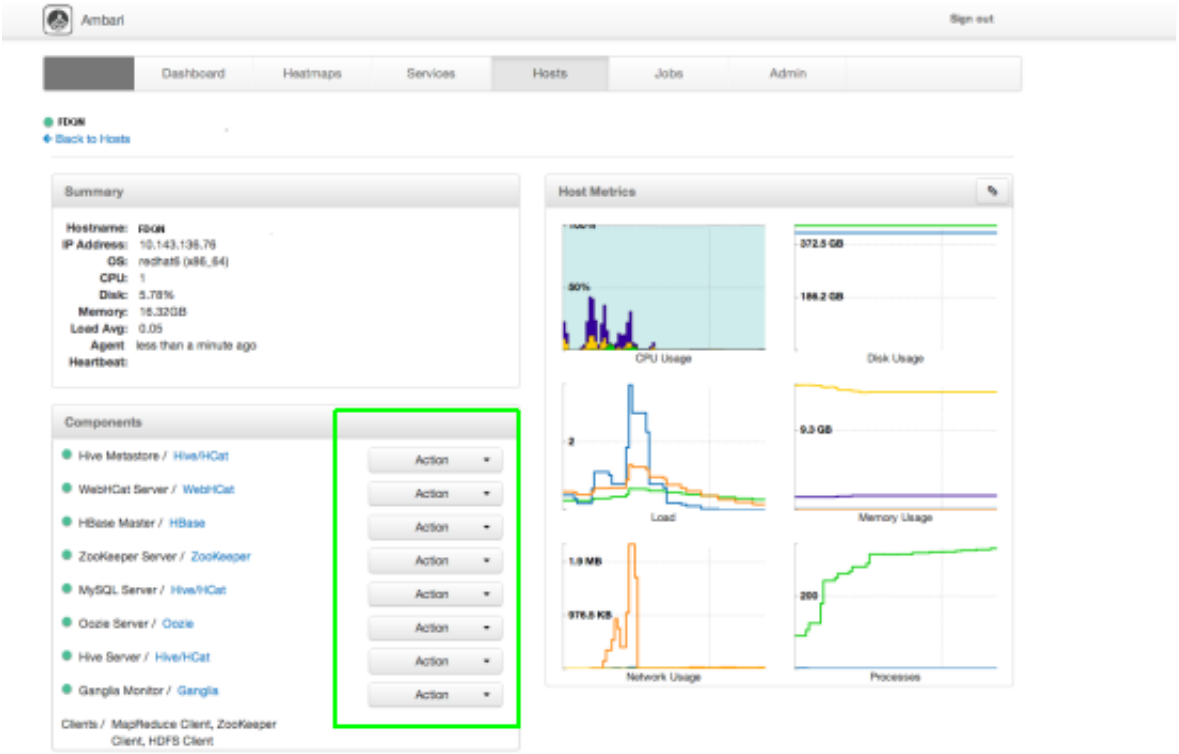
Note the colored dots next to the FQDN in the main Hosts view. These give you a quick overview of the host status:

- Red - At least one master component on that host is down. Hover to see a tooltip with a list of affected components.
- Orange - At least one slave component on that host is down. Hover to see a tooltip with a list of affected components.
- Yellow - Ambari Server has not received a heartbeat from that host for more than 3 minutes.
- Green - Normal running state

The trigger for a red condition overrides a trigger for an orange condition, and so on down the list.

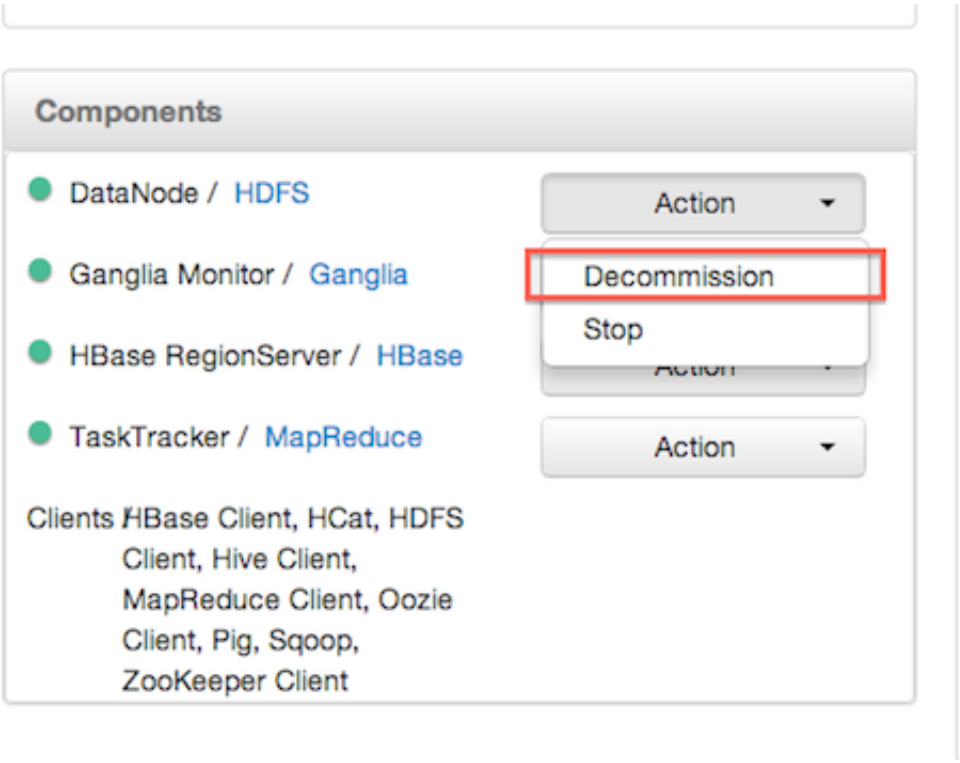
3.5.1. Host Details

To see more detailed information on a specific host, click the host FQDN. The Host Details screen opens.



Use the **Action** dropdowns, outlined in green above, to stop or start a service component running on that host.

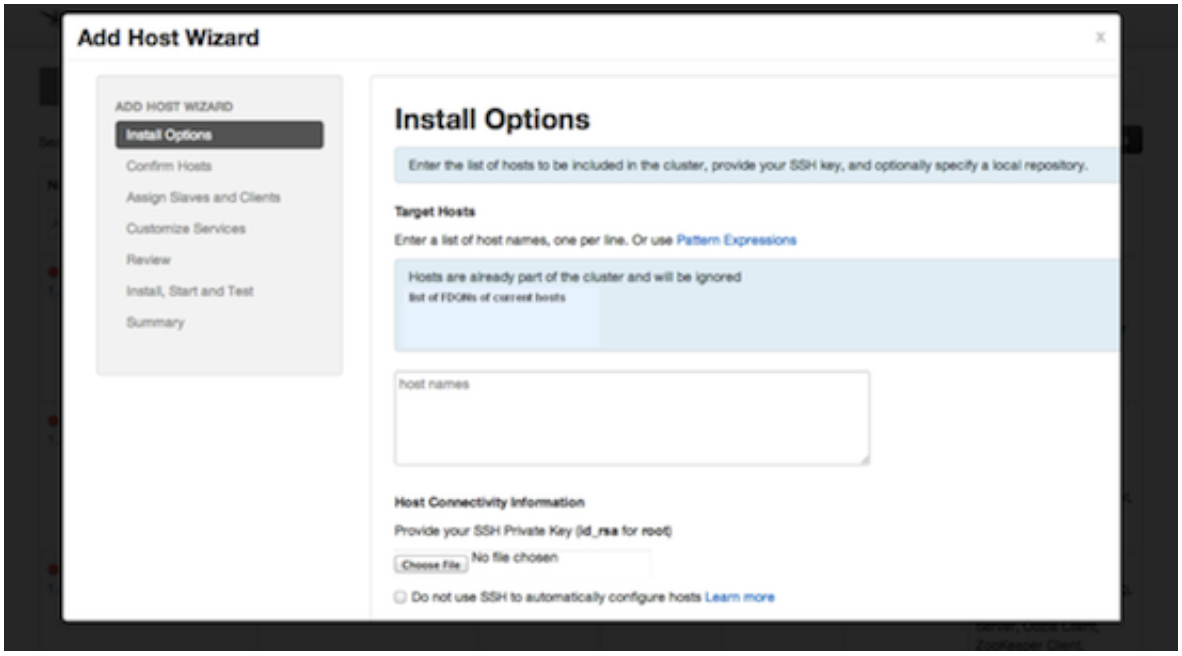
You can also use the Action dropdown on a DataNode host to decommision that node and safely move data from it to other DataNodes in your cluster.



Decommissioning a DataNode runs as a background operation. When the data has all been transferred and replication is complete, the DataNode shuts itself down. The health status colored dot turns red to let you know the decommission process is complete.

3.5.2. Add Hosts

To add new hosts to your cluster, click **+Add New Hosts**, outlined in green [48] above. The **Add Host Wizard** pops up.



Follow the wizard through the sequence of steps - similar to those in the Install wizard - to add hosts. To review the Install wizard, see Installing, Configuring, and Deploying the Cluster in the Ambari Installation Guide.

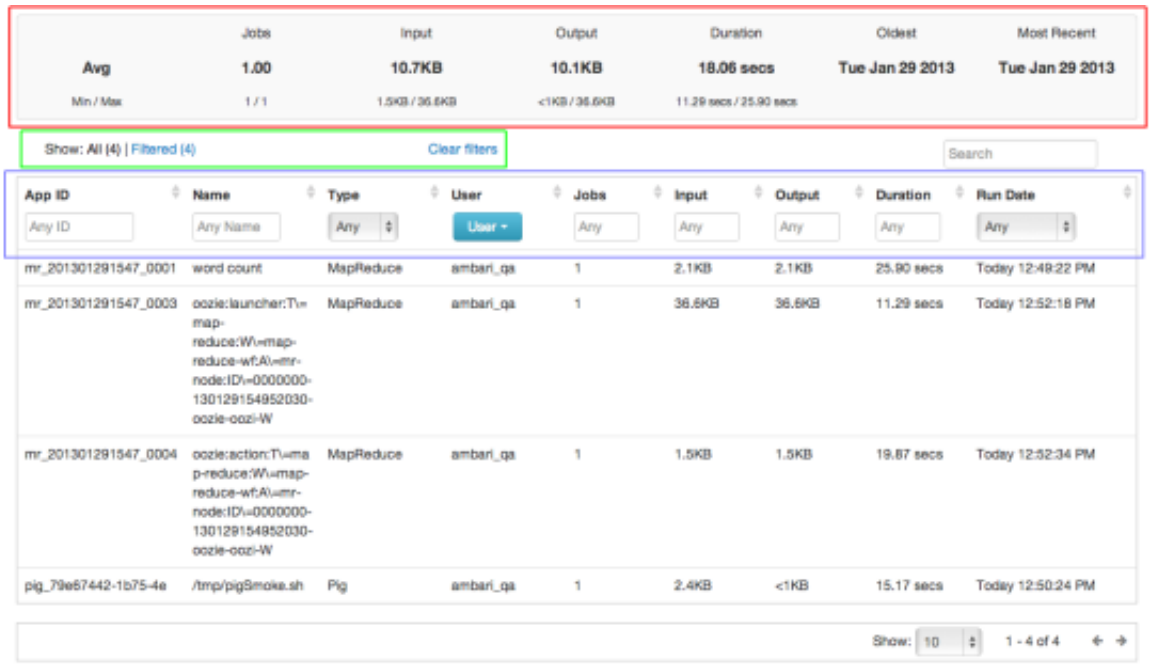
3.6. The Jobs View

The Jobs view displays detailed information about all the jobs running on your cluster. It is useful for diagnosing job execution performance. You can browse job details, including name, execution time, input and output. This view also includes visualizations to show the interdependent jobs (and tasks) that can make up a Hive or Pig job.

3.6.1. Browsing Jobs

The aggregate panel, outlined in red, gives you an overall view of sets of jobs, grouped by filters. Use the selector, outlined in green, to clear previous filters.

To create a filter, use the filter tool, outlined in purple. To search for a particular user, click the **User** dropdown list, select the user, and then click **Apply** when you have made your selection.

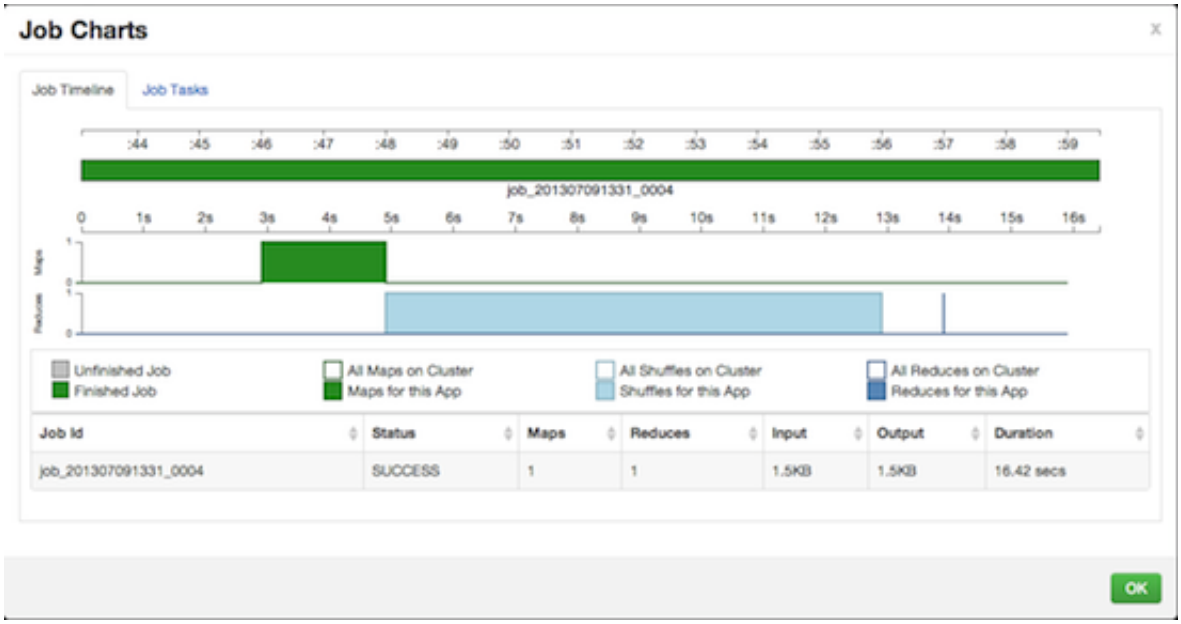


3.6.2. Using Job Charts

To track a specific job, particularly a Hive or Pig query that has been broken down into multiple interdependent jobs, use the **Job Charts** screen to see a more detailed picture. The **Job Timeline** tab displays a detailed graph and timeline of each map and reduce task and the **Job Tasks** tab displays task attempt information per task and job.

To open the **Job Charts** screen, click the appropriate Job id number.

The **Job Charts** screen pops up in the **Job Timeline** tab.

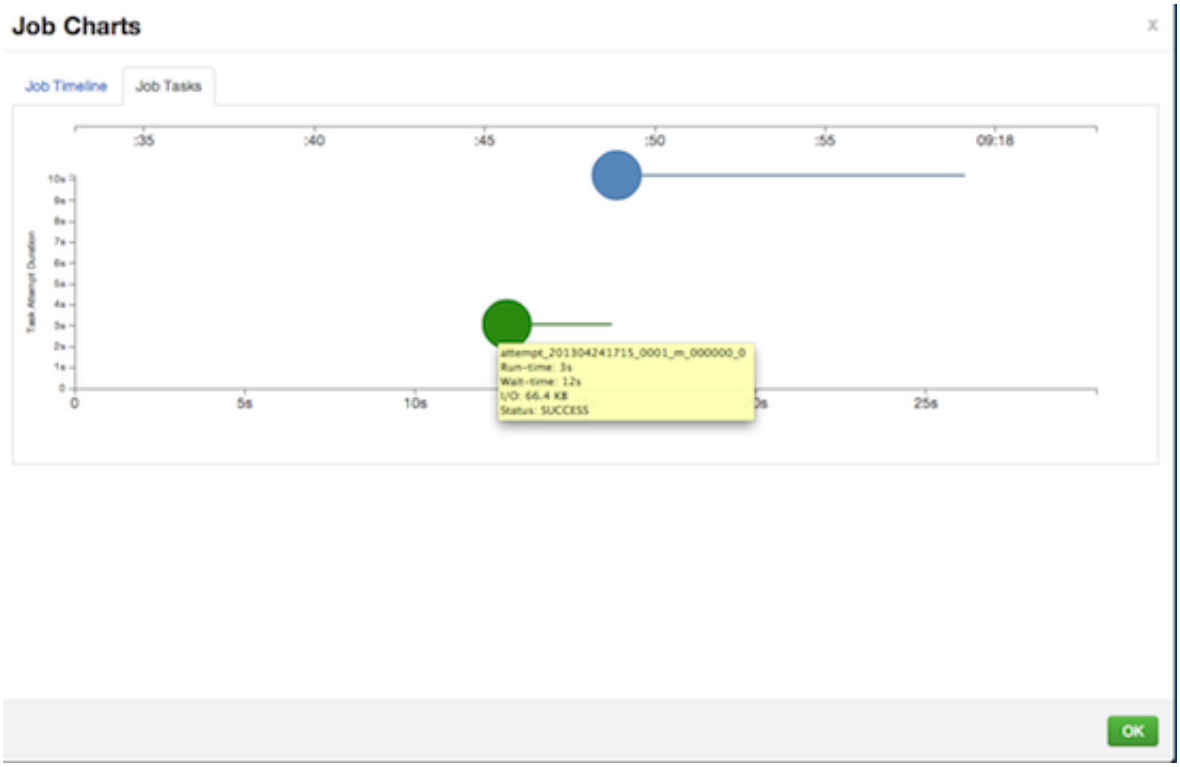


The graph displays the execution sequence of each job and each map and reduce segment against a timeline as well as other information, including the duration of execution, I/O, and number of map and reduce tasks. Jobs are shown at the top of the graph in sequence, while the tasks that make up the individual jobs are shown in a "swimlane" format.

- The X-axis displays the overall execution time. The example starts at 0 and increases to 25 seconds on the far right. Map tasks are shown in green, shuffle segments in light blue, and the actual reduce tasks in dark blue.

- The Y-axis shows the number of tasks running. The color-filled objects show the tasks for the currently displayed job and the white-filled objects show the number of tasks running in the system as a whole at that moment, so that you can understand the context in which the job is being executed.

Click on the **JobTasks** tab to view a scatterplot of job tasks.



The **Job Tasks** graph also shows the task execution time on the X-axis but on the Y-axis shows the Task Attempt Duration time (in seconds). Hover on each task circle and see more details, such as Wait-time and I/O. The size of the circle shown is based on the amount of I/O for the task.

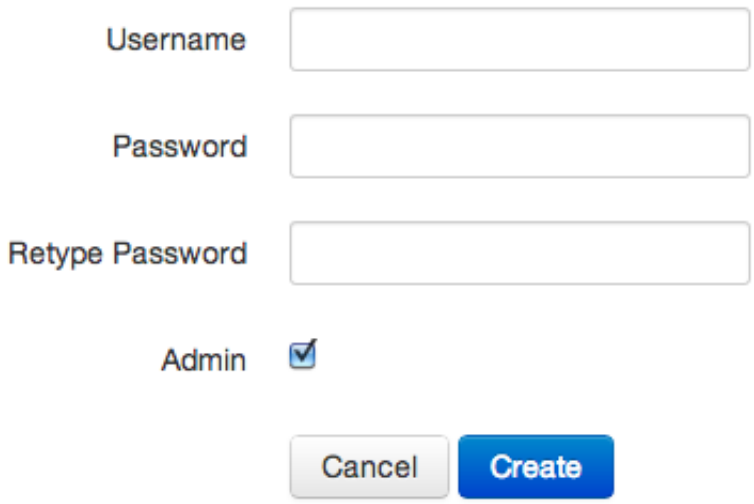
3.7. Admin View

The **Admin** view allows you to manage Ambari Web users, set up Kerberos security, and check for general information about the cluster.

3.7.1. Managing Ambari Web Users

Select Users in the left nav bar to add Ambari Web users, make them administrators, delete them, or change their passwords. There are two user roles: **User** and **Admin**. **Users** can view metrics, view service status and configuration, and browse job information. **Admins** can do all User tasks and in addition can start or stop services, modify configurations, and run smoke tests.

To make a change in a current user's password, click **edit**, in red above. To add an additional Ambari Web user, click **+Add Local User**. In both cases, a variant of the username pop up appears.



For a new Ambari Web user, enter the desired **Username** and **Password**. Check **Admin** to make this user an administrator. Click **Create** to make the user.

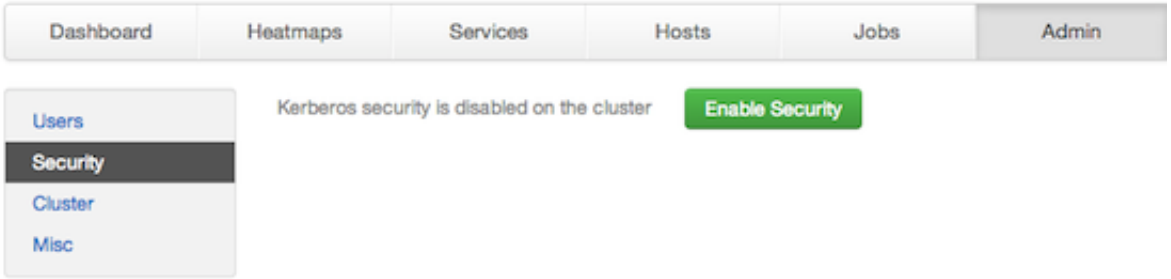
For an Ambari Web existing user, enter the old and new **Passwords**. Make your changes and click **Save**.

To delete an existing Ambari Web user, click **delete** and fill in the **Username**.

3.7.2. Enabling Kerberos Security

To turn on Kerberos-based security you must:

1. Have already set up Kerberos for your cluster.
2. Click **Enable Security** and follow the **Add security wizard**.



   a. **Get Started**: This step just reminds you that you need to set up Kerberos before you start. For more information, see Setting Up Kerberos for Use with Ambari.

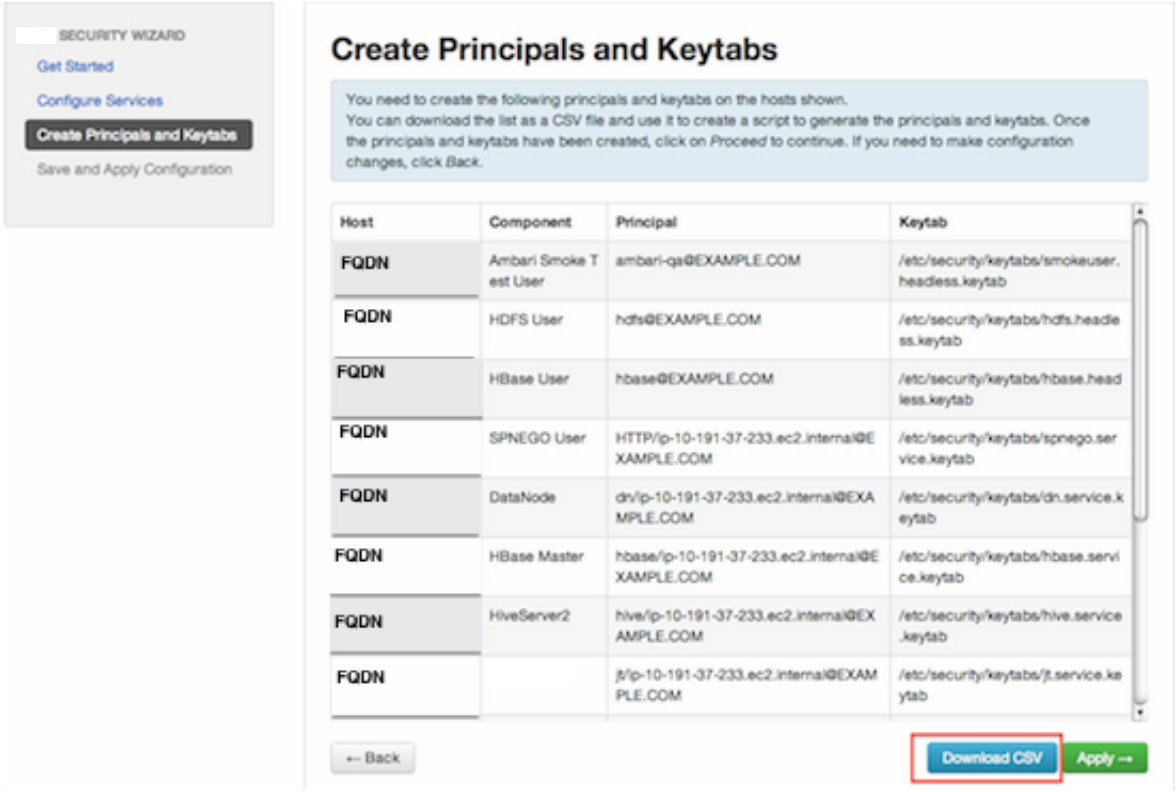
   b. **Configure Services**: This step asks you for your Kerberos information: principals and paths to keytabs, path to your Kerberos tools, realm names and so on. For

more information about a specific field, hover over it, and a popup with a definition appears.

c. **Create Principals and Keytabs**: Use this step to check that all your information is correct. Click **Back** to make any changes. Click **Apply** when you are satisfied with the assignments.

Note

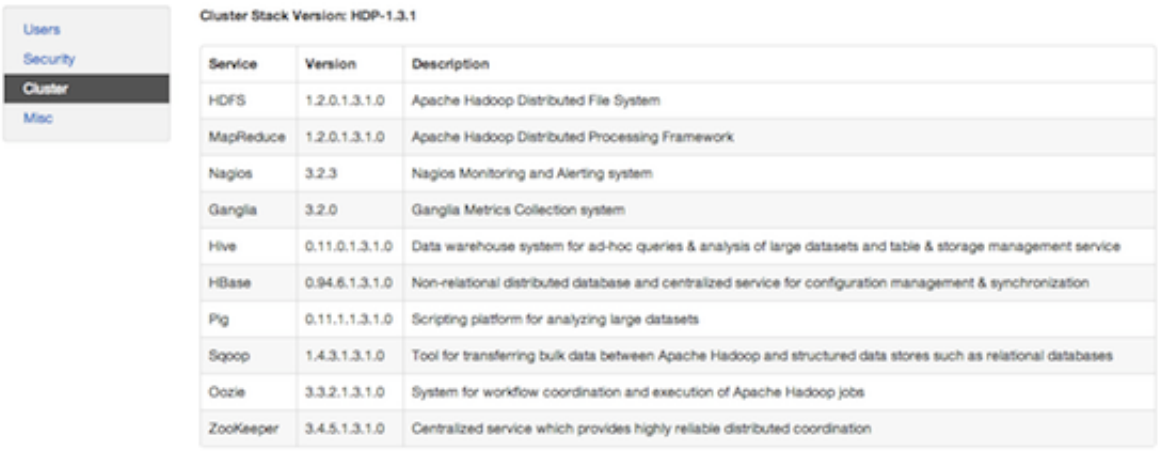
If you have a large cluster, you may want to go to the **Create Principals and Keytabs** step first. Step through the wizard accepting the defaults to get to the appropriage page. On the page, use the **Download CSV** button to get a list of all the necessary principals and keytabs in CSV form, which can be used to set up a script. The list includes hostname, principal description, principal name, keytab user, keytab group, keytab permissions, absolute keytab path, and keytab filename.



d. **Save and Apply Configuration**: This step displays a bar showing the progress of integrating the Kerberos information into your Ambari Server. If you decide you do not want to complete the process at this time, shut the window and click **Cancel** in the popup window.

3.7.3. Checking Stack and Component Versions

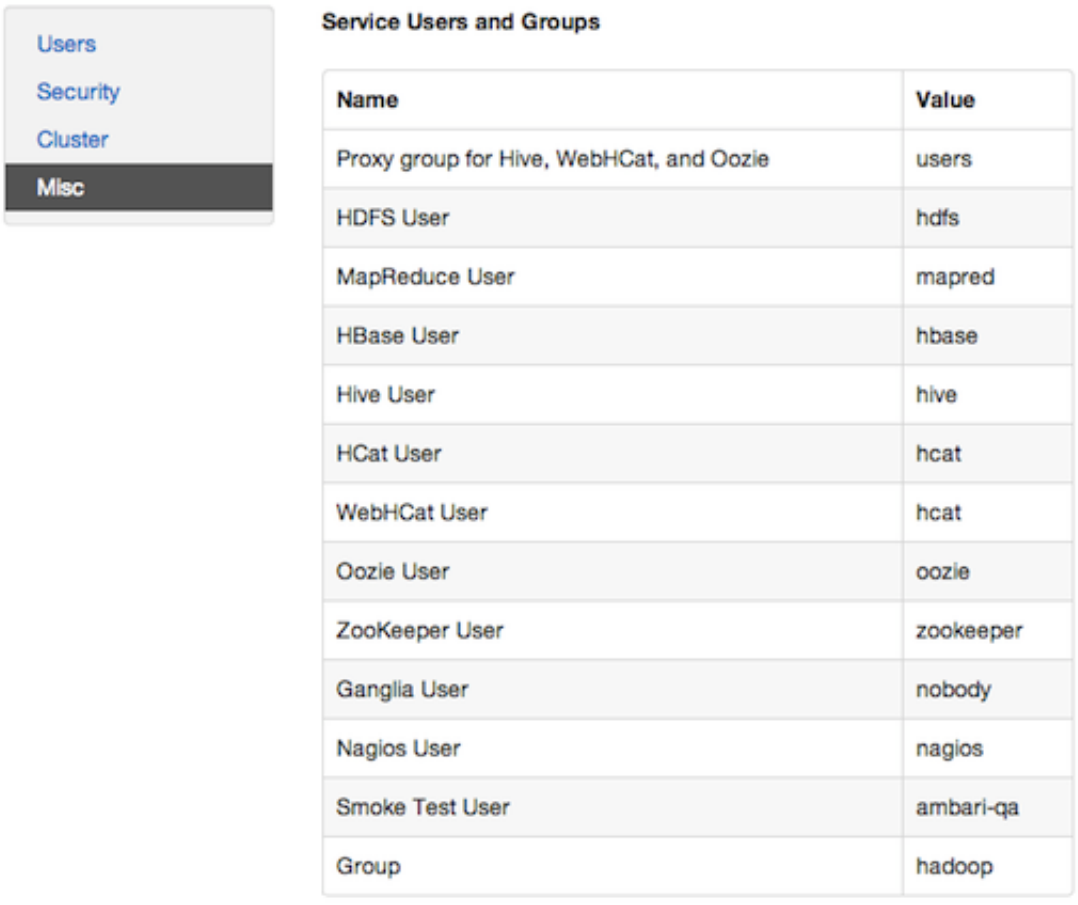
To check which version of the Stack you are using and to see the component versions that are included in that Stack, click **Cluster**.



Service	Version	Description
HDFS	1.2.0.1.3.1.0	Apache Hadoop Distributed File System
MapReduce	1.2.0.1.3.1.0	Apache Hadoop Distributed Processing Framework
Nagios	3.2.3	Nagios Monitoring and Alerting system
Ganglia	3.2.0	Ganglia Metrics Collection system
Hive	0.11.0.1.3.1.0	Data warehouse system for ad-hoc queries & analysis of large datasets and table & storage management service
HBase	0.94.6.1.3.1.0	Non-relational distributed database and centralized service for configuration management & synchronization
Pig	0.11.1.1.3.1.0	Scripting platform for analyzing large datasets
Sqoop	1.4.3.1.3.1.0	Tool for transferring bulk data between Apache Hadoop and structured data stores such as relational databases
Oozie	3.3.2.1.3.1.0	System for workflow coordination and execution of Apache Hadoop jobs
ZooKeeper	3.4.5.1.3.1.0	Centralized service which provides highly reliable distributed coordination

3.7.4.

To check the service user accounts and groups that have been assigned to various Hadoop services, click the **Misc** tab.



Name	Value
Proxy group for Hive, WebHCat, and Oozie	users
HDFS User	hdfs
MapReduce User	mapred
HBase User	hbase
Hive User	hive
HCat User	hcat
WebHCat User	hcat
Oozie User	oozie
ZooKeeper User	zookeeper
Ganglia User	nobody
Nagios User	nagios
Smoke Test User	ambari-qa
Group	hadoop

4. Using Nagios With Hadoop

Nagios is an open source network monitoring system designed to monitor all aspects of your Hadoop cluster (such as hosts, services, and so forth) over the network. It can monitor many facets of your installation, ranging from operating system attributes like CPU and memory usage to the status of applications, files, and more. Nagios provides a flexible, customizable framework for collecting data on the state of your Hadoop cluster.

Nagios is primarily used for the following kinds of tasks:

- Getting instant information about your organization's Hadoop infrastructure
- Detecting and repairing problems, and mitigating future issues, before they affect end-users and customers
- Leveraging Nagios' event monitoring capabilities to receive alerts for potential problem areas
- Analyzing specific trends; for example: what is the CPU usage for a particular Hadoop service weekdays between 2 p.m. and 5 p.m

For more information, see the Nagios website at http://www.nagios.org.

Note

Nagios is an optional component of Ambari. During cluster install you can choose to install and configure Nagios. When selected, out-of-the-box Ambari provides a set of Nagios plugins specially designed for monitoring important aspects of your Hadoop cluster, based on your Stack selection.

4.1. Basic Nagios Architecture

The basic Nagios set-up consists of a central server - the "Nagios Server" - that polls the Hadoop servcies and components in the cluster installation to check for one of the following states:

- OK
- Warning
- Critical

Nagios writes its output to a status log, `/var/nagios/status.dat`. The alert information from that file is periodically collected and displayed in Ambari Web. For more details on Nagios architecture, see the Nagios Overview at the `nagios.org` web site.

4.2. Installing Nagios

The Ambari Installation Wizard gives you the option of installing and configuring Nagios, including the out-of-the-box plugins for Hadoop-specific alerts. The Nagios server, Nagios

plugins, and the web-based user interface are installed on the Nagios server host, as specified during the installation procedure.

Some of the Nagios plugins also require the Simple Network Management Protocol (SNMP) daemon to work, so the daemon is automatically installed on all cluster nodes if Nagios is installed. For example, the SNMP daemon is used to collect information about system resources (like memory, storage, swap space, and CPU utilization).

Note

By default the Nagios server does not use any authentication or encryption scheme while communicating with SNMP daemons. However you can enable secure communication with SNMP daemons by setting the appropriate SNMP configuration.

4.3. Configuration File Locations

All Hadoop-specific configurations are added to Nagios through files prefixed with "hadoop-" located in the `/etc/nagios/objects` directory of the Nagios Server host. The default general Nagios configuration file, `nagios.cfg` (in `/etc/nagios`), is set up to pick up the new Hadoop specific configurations from this directory.

Hadoop-specific plugins are stored in the Nagios plugins directory,`/usr/lib64/nagios/plugins/`.

By default, the Nagios server runs as a user named "nagios" which is in a group also named "nagios". The user and group can be customized during the Ambari Cluster Install (Cluster Install Wizard > Customize Services > Misc). Once Nagios is installed, use Ambari Web to start and stop the Nagios server.

4.4. Configuring Nagios Alerts For Hadoop Services

For each alert, the out of the box Hadoop Nagios configuration file defines default values for the following Nagios directives:

Warning threshold	The value that produces a warning alert.
Critical threshold	The value that produces a critical alert.
Check interval	The number of minutes between regularly scheduled checks on the host as long as the check does not change the state.
Retry interval	The number of minutes between "retries" When a service changes state, Nagios can confirm that state change by retrying the check multiple times. This retry interval can be different than the original check interval.
Maximum number of check attempts	The max number of retry attempts. Usually when the state of a service changes, this change is considered

"soft" until multiple retries confirm it. Once the state change is confirmed, it is considered "hard". Ambari Web displays hard states for all the Nagios Hadoop specific checks.

4.5. Nagios Alerts For Hadoop Services

The following section provides more information on the various Hadoop alerts provided by Ambari. All these alerts are displayed in Ambari Web and in the native Nagios web interface.

There are two types of alerts configured out-of-the-box with Ambari:

- **Host-level Alerts**

  These alerts are related to a specific host and specific component running on that host. These alerts check a component and system-level metrics to determine health of the host.

- **Service-level Alerts**

  These alerts are related to a Hadoop Service and do not pertain to a specific host. These alerts check one or more components of a service as well as system-level metrics to determine overall health of a Hadoop Service.

4.5.1. HDFS Service Alerts

These alerts are used to monitor the HDFS service.

4.5.1.1. Blocks health

This service-level alert is triggered if the number of corrupt or missing blocks exceeds the configured critical threshold. This alert uses the `check_hdfs_blocks` plugin.

4.5.1.1.1. Potential causes

- Some DataNodes are down and the replicas that are missing blocks are only on those DataNodes
- The corrupt/missing blocks are from files with a replication factor of 1. New replicas cannot be created because the only replica of the block is missing

4.5.1.1.2. Possible remedies

- For critical data, use a replication factor of 3
- Bring up the failed DataNodes with missing or corrupt blocks.
- Identify the files associated with the missing or corrupt blocks by running the Hadoop `fsck` command
- Delete the corrupt files and recover them from backup, if it exists

4.5.1.2. NameNode process

This host-level alert is triggered if the NameNode process cannot be confirmed to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp`

4.5.1.2.1. Potential causes

- The NameNode process is down on the HDFS master host
- The NameNode process is up and running but not listening on the correct network port (default 8201)
- The Nagios server cannot connect to the HDFS master through the network.

4.5.1.2.2. Possible remedies

- Check for any errors in the logs (`/var/log/hadoop/hdfs/`)and restart the NameNode host/process using the **HMC Manage Services** tab.
- Run the `netstat-tuplpn` command to check if the NameNode process is bound to the correct network port
- Use `ping` to check the network connection between the Nagios server and the NameNode

4.5.1.3. DataNode space

This host-level alert is triggered if storage capacity is full on the DataNodes. It uses the `check_snmp_storage` plugin.

4.5.1.3.1. Potential causes

- Cluster storage is full
- If cluster storage is not full, DataNode is full

4.5.1.3.2. Possible remedies

- If cluster still has storage, use Balancer to distribute the data to relatively less used datanodes
- If the cluster is full, delete unnecessary data or add additional storage by adding either more DataNodes or more or larger disks to the DataNodes. After adding more storage run Balancer

4.5.1.4. DataNode process

This host-level alert is triggered if the individual DataNode processes cannot be established to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

4.5.1.4.1. Potential causes

- DataNode process is down or not responding

- DataNode are not down but is not listening to the correct network port/address
- Nagios server cannot connect to the DataNodes

4.5.1.4.2. Possible remedies

- Check for dead DataNodes in **Ambari Web**.
- Check for any errors in the DataNode logs (`/var/log/hadoop/hdfs`) and restart the DataNode, if necessary
- Run the `netstat-tuplpn` command to check if the DataNode process is bound to the correct network port
- Use `ping` to check the network connection between the Nagios server and the DataNode

4.5.1.5. NameNode host CPU utilization

This ost-level alert is triggered if the percent of CPU utilization on the master host exceeds the configured critical threshold. It uses the Nagios `check_snmp_load` plugin.

4.5.1.5.1. Potential causes

- Unusually high CPU utilization: Can be caused by a very unusual job/query workload, but this is generally the sign of an issue in the daemon.
- A down SNMP daemon on the master node, producing an unknown status

4.5.1.5.2. Possible remedies

- Use the `top` command to determine which processes are consuming excess CPU.
- Reset the offending process
- Check the status of the SNMP daemon

4.5.1.6. NameNode edit logs directory status

This host-level alert is triggered if the NameNode cannot write to one of its configured edit log directories.

4.5.1.6.1. Potential causes

- At least one of the multiple edit log directories is mounted over NFS and has become unreachable
- The permissions on at least one of the multiple edit log directories has become read-only

4.5.1.6.2. Possible remedies

- Check permissions on all edit log directories
- Use the `dfs.name.dir` parameter in the `hdfs-site.xml` file on the NameNode to identify the locations of all the edit log directories for the NameNode. Check whether the NameNode can reach all those locations.

4.5.1.7. NameNode Web UI

This host-level alert is triggered if the NameNode Web UI is unreachable.

4.5.1.7.1. Potential causes

- The NameNode Web UI is unreachable from the Nagios Server.
- The NameNode process is not running

4.5.1.7.2. Possible remedies

- Check whether the NameNode process is running
- Check whether the Nagios Server can ping the NameNode server.
- Using a browser, check whether the Nagios Server can reach the NameNode Web UI.

4.5.1.8. Percent DataNodes with space available

This service-level alert is triggered if storage capacity is full on the DataNodes. All the local data partitions storing HDFS data are checked against the total capacity across all the partitions. It uses the `check_snmp_storage` plugin.

4.5.1.8.1. Potential causes

- Cluster storage is full
- If cluster storage is not full, DataNode is full

4.5.1.8.2. Possible remedies

- If cluster still has storage, use Balancer to distribute the data to relatively less used DataNodes
- If the cluster is full, delete unnecessary data or add additional storage by adding either more DataNodes or more or larger disks to the DataNodes. After adding more storage run Balancer

4.5.1.9. Percent DataNodes live

This alert is triggered if the number of down DataNodes in the cluster is greater than the configured critical threshold. It uses the `check_aggregate` plugin to aggregate the results of `Data node process` checks.

4.5.1.9.1. Potential causes

- DataNodes are down
- DataNodes are not down but are not listening to the correct network port/address
- Nagios server cannot connect to one or more DataNodes

4.5.1.9.2. Possible remedies

- Check for dead DataNodes in **Ambari Web**.
- Check for any errors in the DataNode logs (`/var/log/hadoop/hdfs`) and restart the DataNode hosts/processes
- Run the `netstat-tuplpn` command to check if the DataNode process is bound to the correct network port.
- Use `ping` to check the network connection between the Nagios server and the DataNodes.

4.5.1.10. NameNode RPC latency

This host-level alert is triggered if the NameNode operations RPC latency exceeds the configured critical threshold. Typically an increase in the RPC processing time increases the RPC queue length, causing the average queue wait time to increase for NameNode operations. It uses the Nagios `check_rpcq_latency` plugin.

4.5.1.10.1. Potential causes

- A job or an application is performing too many NameNode operations.

4.5.1.10.2. Possible remedies

- Review the job or the application for potential bugs causing it to perform too many NameNode operations.

4.5.1.11. HDFS capacity utilization

This service-level alert is triggered if the HDFS capacity utilization exceeds the configured critical threshold. It uses the `check_hdfs_capacity` plugin.

4.5.1.11.1. Potential causes

- Cluster storage is full

4.5.1.11.2. Possible remedies

- Delete unnecessary data.
- Archive unused data.
- Add more DataNodes.
- Add more or larger disks to the DataNodes.
- After adding more storage, run Balancer.

4.5.2. NameNode HA Alerts (Hadoop 2 only)

These alerts are available only when you are using Hadoop 2.x and you have enabled NameNode HA.

4.5.2.1. JournalNode process

This host-level alert is triggered if the individual JournalNode process cannot be established to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

4.5.2.1.1. Potential causes

- The JournalNode process is down or not responding.
- The JournalNode is not down but is not listening to the correct network port/address.
- The Nagios server cannot connect to the JournalNode.

4.5.2.1.2. Possible remedies

- Check if the JournalNode process is dead.
- Use `ping` to check the network connection between the Nagios server and the JournalNode host.

4.5.2.2. NameNode HA Healthy process

This service-level alert is triggered if either the Active NameNode or Standby NameNode are not running.

4.5.2.2.1. Potential causes

- The Active, Standby or both NameNode processes are down.
- The Nagios Server cannot connect to one or both NameNode hosts.

4.5.2.2.2. Possible remedies

- On each host running NameNode, check for any errors in the logs (`/var/log/hadoop/hdfs/`) and restart the NameNode host/process using **Ambari Web**.
- On each host running NameNode, run the `netstat-tuplpn` command to check if the NameNode process is bound to the correct network port.
- Use `ping` to check the network connection between the Nagios server and the hosts running NameNode.

4.5.3. YARN Alerts (Hadoop 2 only)

These alerts are used to monitor YARN.

4.5.3.1. ResourceManager process

This host-level alert is triggered if the individual ResourceManager process cannot be established to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

4.5.3.1.1. Potential causes

- The ResourceManager process is down or not responding.
- The ResourceManager is not down but is not listening to the correct network port/address.
- Nagios Server cannot connect to the ResourceManager

4.5.3.1.2. Possible remedies

- Check for a dead ResourceManager.
- Check for any errors in the ResourceManager logs (`/var/log/hadoop/yarn`) and restart the ResourceManager, if necessary.
- Use `ping` to check the network connection between the Nagios Server and the ResourceManager host.

4.5.3.2. Percent NodeManagers live

This alert is triggered if the number of down NodeManagers in the cluster is greater than the configured critical threshold. It uses the `check_aggregate` plugin to aggregate the results of DataNode process alert checks.

4.5.3.2.1. Potential causes

- NodeManagers are down.
- NodeManagers are not down but are not listening to the correct network port/address .
- Nagios server cannot connect to one or more NodeManagers.

4.5.3.2.2. Possible remedies

- Check for dead NodeManagers.
- Check for any errors in the NodeManager logs (`/var/log/hadoop/yarn`) and restart the NodeManagers hosts/processes, as necessary.
- Run the `netstat-tuplpn` command to check if the NodeManager process is bound to the correct network port.
- Use `ping` to check the network connection between the Nagios Server and the NodeManagers host.

4.5.3.3. ResourceManager Web UI

This host-level alert is triggered if the ResourceManager Web UI is unreachable.

4.5.3.3.1. Potential causes

- The ResourceManager Web UI is unreachable from the Nagios Server.

- The ResourceManager process is not running.

4.5.3.3.2. Possible remedies

- Check if the ResourceManager process is running.
- Check whether the Nagios Server can ping the ResourceManager server.
- Using a browser, check whether the Nagios Server can reach the ResourceManager Web UI.

4.5.3.4. ResourceManager RPC latency

This host-level alert is triggered if the ResourceManager operations RPC latency exceeds the configured critical threshold. Typically an increase in the RPC processing time increases the RPC queue length, causing the average queue wait time to increase for ResourceManager operations. It uses the Nagios `check_rpcq_latency` plugin.

4.5.3.4.1. Potential causes

- A job or an application is performing too many ResourceManager operations.

4.5.3.4.2. Possible remedies

- Review the job or the application for potential bugs causing it to perform too many ResourceManager operations.

4.5.3.5. ResourceManager CPU utilization

This host-level alert is triggered if the percent of CPU utilization on the ResourceManager exceeds the configured critical threshold. This alert uses the Nagios `check_snmp_load` plugin.

4.5.3.5.1. Potential causes

- Unusually high CPU utilization: Can be caused by a very unusual job/query workload, but this is generally the sign of an issue in the daemon.
- A down SNMP daemon on the ResourceManager node, producing an unknown status.

4.5.3.5.2. Possible remedies

- Use the `top` command to determine which processes are consuming excess CPU.
- Reset the offending process.
- Check the status of the SNMP daemon.

4.5.3.6. NodeManager process

This host-level alert is triggered if the NodeManager process cannot be established to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

4.5.3.6.1. Potential causes

- NodeManager process is down or not responding.
- NodeManager is not down but is not listening to the correct network port/address.
- Nagios Server cannot connect to the NodeManager

4.5.3.6.2. Possible remedies

- Check if the NodeManager is running.
- Check for any errors in the NodeManager logs (`/var/log/hadoop/yarn`) and restart the NodeManager, if necessary.
- Use `ping` to check the network connection between the Nagios Server and the NodeManager host.

4.5.3.7. NodeManager health

This host-level alert checks the node health property available from the NodeManager component.

4.5.3.7.1. Potential causes

- Node Health Check script reports issues or is not configured.

4.5.3.7.2. Possible remedies

- Check in the NodeManager logs (`/var/log/hadoop/yarn`) for health check errors and restart the NodeManager, and restart if necessary.
- Check in the ResoruceManager UI logs (`/var/log/hadoop/yarn`) for health check errors.

4.5.4. MapReduce2 Alerts (Hadoop 2 only)

These alerts are used to monitor MR2.

4.5.4.1. HistoryServer Web UI

This host-level alert is triggered if the HistoryServer Web UI is unreachable.

4.5.4.1.1. Potential causes

- The HistoryServer Web UI is unreachable from the Nagios Server.
- The HistoryServer process is not running.

4.5.4.1.2. Possible remedies

- Check if the HistoryServer process is running.

- Check whether the Nagios Server can ping the HistoryServer server.
- Using a browser, check whether the Nagios Server can reach the HistoryServer Web UI.

4.5.4.2. HistoryServer RPC latency

This host-level alert is triggered if the HistoryServer operations RPC latency exceeds the configured critical threshold. Typically an increase in the RPC processing time increases the RPC queue length, causing the average queue wait time to increase for NameNode operations. It uses the Nagios `check_rpcq_latency` plugin.

4.5.4.2.1. Potential causes

- A job or an application is performing too many HistoryServer operations

4.5.4.2.2. Possible remedies

- Review the job or the application for potential bugs causing it to perform too many HistoryServer operations.

4.5.4.3. HistoryServer CPU utilization

This host-level alert is triggered if the percent of CPU utilization on the HistoryServer exceeds the configured critical threshold. This alert uses the Nagios `check_snmp_load` plug-in.

4.5.4.3.1. Potential causes

- Unusually high CPU utilization: Can be caused by a very unusual job/query workload, but this is generally the sign of an issue in the daemon.
- A down SNMP daemon on the HistoryServer node, producing an unknown status

4.5.4.3.2. Possible remedies

- Use the `top` command to determine which processes are consuming excess CPU.
- Reset the offending process.
- Check the status of the SNMP daemon.

4.5.4.4. HistoryServer process

This host-level alert is triggered if the HistoryServer process cannot be established to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

4.5.4.4.1. Potential causes

- HistoryServer process is down or not responding.
- HistoryServer is not down but is not listening to the correct network port/address.

- Nagios Server cannot connect to the HistoryServer,

4.5.4.4.2. Possible remedies

- Check the HistoryServer is running.
- Check for any errors in the HistoryServer logs (`/var/log/hadoop/mapred`) and restart the HistoryServer, if necessary
- Use `ping` to check the network connection between the Nagios Server and the HistoryServer host.

4.5.5. MapReduce Service Alerts (Hadoop 1 only)

These alerts are used to monitor the MapReduce service.

4.5.5.1. JobTracker RPC latency alert

This host-level alert is triggered if the JobTracker operations RPC latency exceeds the configured critical threshold. Typically an increase in the RPC processing time increases the RPC queue length, causing the average queue wait time to increase for JobTracker operations. This alert uses the Nagios `check_rpcq_latency` plugin.

4.5.5.1.1. Potential causes

- A job or an application is performing too many JobTracker operations.

4.5.5.1.2. Possible remedies

- Review the job or the application for potential bugs causing it to perform too many JobTracker operations

4.5.5.2. JobTracker process

This host-level alert is triggered if the individual JobTracker process cannot be confirmed to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

4.5.5.2.1. Potential causes

- JobTracker process is down or not responding.
- JobTracker is not down but is not listening to the correct network port/address.
- The Nagios server cannot connect to the JobTracker

4.5.5.2.2. Possible remedies

- Check if the JobTracker process is running.
- Check for any errors in the JobTracker logs (`/var/log/hadoop/mapred`) and restart the JobTracker, if necessary

- Use `ping` to check the network connection between the Nagios Server and the JobTracker host.

4.5.5.3. JobTracker Web UI

This Host-level alert is triggered if the JobTracker Web UI is unreachable.

4.5.5.3.1. Potential causes

- The JobTracker Web UI is unreachable from the Nagios Server.
- The JobTracker process is not running.

4.5.5.3.2. Possible remedies

- Check if the JobTracker process is running.
- Check whether the Nagios Server can ping the JobTracker server.
- Using a browser, check whether the Nagios Server can reach the JobTracker Web UI.

4.5.5.4. JobTracker CPU utilization

This host-level alert is triggered if the percent of CPU utilization on the JobTracker exceeds the configured critical threshold. This alert uses the Nagios `check_snmp_load` plug-in.

4.5.5.4.1. Potential causes

- Unusually high CPU utilization: Can be caused by a very unusual job/query workload, but this is generally the sign of an issue in the daemon.
- A down SNMP daemon on the JobTracker node, producing an unknown status.

4.5.5.4.2. Possible remedies

- Use the `top` command to determine which processes are consuming excess CPU
- Reset the offending processor.
- Check the status of the SNMP daemon.

4.5.5.5. HistoryServer Web UI

This host-level alert is triggered if the HistoryServer Web UI is unreachable.

4.5.5.5.1. Potential causes

- The HistoryServer Web UI is unreachable from the Nagios Server.
- The HistoryServer process is not running.
- Using a browser, check whether the Nagios Server can reach the HistoryServer Web UI.

4.5.5.5.2. Possible remedies

- Check the HistoryServer process is running.
- Check whether the Nagios Server can ping the HistoryServer server.
- Check the status of the SNMP daemon.

4.5.5.6. HistoryServer process

This host-level alert is triggered if the HistoryServer process cannot be established to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

4.5.5.6.1. Potential causes

- The HistoryServer process is down or not responding.
- The HistoryServer is not down but is not listening to the correct network port/address.
- The Nagios Server cannot connect to the HistoryServer.

4.5.5.6.2. Possible remedies

- Check for any errors in the HistoryServer logs (`/var/log/hadoop/mapred`) and restart the HistoryServer, if necessary.
- Use `ping` to check the network connection between the Nagios Server and the HistoryServer host.

4.5.6. HBase Service Alerts

These alerts are used to monitor the HBase service.

4.5.6.1. Percent RegionServers live

This service-level alert is triggered if the configured percentage of Region Server processes cannot be determined to be up and listening on the network for the configured critical threshold.The default setting is 10% to produce a WARN alert and 30% to produce a CRITICAL alert. It uses the `check_aggregate` plugin to aggregate the results of `RegionServer process down` checks.

4.5.6.1.1. Potential causes

- Misconfiguration or less-than-ideal configuration caused the RegionServers to crash
- Cascading failures brought on by some workload caused the RegionServers to crash
- The RegionServers shut themselves own because there were problems in the dependent services, ZooKeeper or HDFS
- GC paused the RegionServer for too long and the RegionServers lost contact with Zoo-keeper

4.5.6.1.2. Possible remedies

- Check the dependent services to make sure they are operating correctly.
- Look at the RegionServer log files (usually `/var/log/hbase/*.log`) for further information
- Look at the configuration files (`/etc/hbase/conf`)
- If the failure was associated with a particular workload, try to understand the workload better
- Restart the RegionServers

4.5.6.2. HBase Master process

This alert is triggered if the HBase master processes cannot be confirmed to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

4.5.6.2.1. Potential causes

- The HBase master process is down
- The HBase master has shut itself down because there were problems in the dependent services, ZooKeeper or HDFS
- The Nagios server cannot connect to the HBase master through the network

4.5.6.2.2. Possible remedies

- Check the dependent services.
- Look at the master log files (usually `/var/log/hbase/*.log`) for further information
- Look at the configuration files (`/etc/hbase/conf`)

  Use `ping` to check the network connection between the Nagios server and the HBase master
- Restart the master

4.5.6.3. HBase Master Web UI

This host-level alert is triggered if the HBase Master Web UI is unreachable.

4.5.6.3.1. Potential causes

- The HBase Master Web UI is unreachable from the Nagios Server.
- The HBase Master process is not running.

4.5.6.3.2. Possible remedies

- Check if the Master process is running.

- Check whether the Nagios Server can ping the HBase Master server.
- Using a browser, check whether the Nagios Server can reach the HBase Master Web UI.

4.5.6.4. HBase Master CPU utilization

This host-level alert is triggered if the percent of CPU utilization on the master host exceeds the configured critical threshold. This alert uses the Nagios `check_snmp_load` plugin.

4.5.6.4.1. Potential causes

- Unusually high CPU utilization: Can be caused by a very unusual job/query workload, but this is generally the sign of an issue in the daemon.
- A down SNMP daemon on the master node, producing an unknown status.

4.5.6.4.2. Possible remedies

- Use the `top` command to determine which processes are consuming excess CPU
- Reset the offending process.
- Check the status of the SNMP daemon.

4.5.6.5. RegionServer process

This host-level alert is triggered if the RegionServer processes cannot be confirmed to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

4.5.6.5.1. Potential causes

- The RegionServer process is down on the host
- The RegionServer process is up and running but not listening on the correct network port (default 60030).
- The Nagios server cannot connect to the RegionServer through the network.

4.5.6.5.2. Possible remedies

- Check for any errors in the logs (`/var/log/hbase/`) and restart the RegionServer process using **Ambari Web**.
- Run the `netstat-tuplpn` command to check if the RegionServer process is bound to the correct network port.
- Use `ping` to check the network connection between the Nagios Server and the RegionServer.

4.5.7. Hive Alerts

These alerts are used to monitor the Hive service.

4.5.7.1. Hive-Metastore status

This host-level alert is triggered if the Hive Metastore process cannot be determined to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

4.5.7.1.1. Potential causes

- The Hive Metastore service is down.
- The database used by the Hive Metastore is down.
- The Hive Metastore host is not reachable over the network.

4.5.7.1.2. Possible remedies

- Using **Ambari Web**, stop the Hive service and then restart it.
- Use `ping` to check the network connection between the Nagios server and the Hive Metastore server.

4.5.8. WebHCat Alerts

These alerts are used to monitor the WebHCat service.

4.5.8.1. WebHCat Server status

This host-level alert is triggered if the WebHCat server cannot be determined to be up and responding to client requests.

4.5.8.1.1. Potential causes

- The WebHCat server is down.
- The WebHCat server is hung and not responding.
- The WebHCat server is not reachable over the network.

4.5.8.1.2. Possible remedies

- Restart the WebHCat server using **Ambari Web**.

4.5.9. Oozie Alerts

These alerts are used to monitor the Oozie service.

4.5.9.1. Oozie status

This host-level alert is triggered if the Oozie server cannot be determined to be up and responding to client requests.

4.5.9.1.1. Potential causes

- The Oozie server is down.

- The Oozie server is hung and not responding.
- The Oozie server is not reachable over the network.

4.5.9.1.2. Possible remedies

- Restart the Oozie service using **Ambari Web**.

4.5.10. Ganglia Alerts

These alerts are used to monitor the Ganglia service.

4.5.10.1. Ganglia Server status

This host-level alert determines if the Ganglia server is running and listening on the network port. It uses the Nagios `check_tcp` plugin.

4.5.10.1.1. Potential causes

- The Ganglia server process is down.
- The Ganglia server process is hanging.
- The network connection is down between the Nagios and Ganglia servers

4.5.10.1.2. Possible remedies

- Check the Ganglia server (`gmetad`) related log in `/var/log/messages` for any errors.
- Restart the Ganglia server.
- Check if `ping` works between Nagios and Ganglia servers.

4.5.10.2. Ganglia Monitor process

These host-level alerts check if the Ganglia monitor daemons (`gmond`) on the Ganglia server are running and listening on the network port. This alert uses the Nagios `check_tcp` plugin.

Ganglia Monitoring daemons run for the following collections:

- Slaves
- NameNode
- HBase Master
- JobTracker (Hadoop 1 only)
- ResourceManager (Hadoop 2 only)
- HistoryServer (Hadoop 2 only)

4.5.10.2.1. Potential causes

- A `gmond` process is down.
- A `gmond` process is hanging.
- The network connection is down between the Nagois and Ganglia servers.

4.5.10.2.2. Possible remedies

- Check the `gmond` related log in `/var/log/messages` for any errors.
- Check if `ping` works between Nagios and Ganglia servers.

4.5.11. Nagios Alerts

These alerts are used to monitor the Nagios service.

4.5.11.1. Nagios status log freshness

This host-level alert determines if the Nagios server is updating its status log regularly. Ambari depends on the status log (`/var/nagios/status.dat`) to receive all the Nagios alerts.

4.5.11.1.1. Potential causes

- The Nagios server is hanging and thus not scheduling new alerts
- The file `/var/nagios/status.dat` does not have appropriate write permissions for the Nagios user.

4.5.11.1.2. Possible remedies

- Restart the Nagios server
- Check the permissions on `/var/nagios/status.dat`.
- Check `/var/log/messages` for any related errors.

4.5.12. ZooKeeper Alerts

These alerts are used to monitor the Zookeeper service.

4.5.12.1. Percent ZooKeeper servers live

This service-level alert is triggered if the configured percentage of ZooKeeper processes cannot be determined to be up and listening on the network for the configured critical threshold, given in seconds. It uses the `check_aggregate` plugin to aggregate the results of `Zookeeper process` checks.

4.5.12.1.1. Potential causes

- The majority of your ZooKeeper servers are down and not responding.

4.5.12.1.2. Possible remedies

- Check the dependent services to make sure they are operating correctly.
- Check at the ZooKeeper logs files (`/var/log/hadoop/zookeeper.log`) for further information.
- If the failure was associated with a particular workload, try to understand the workload better.
- Restart the ZooKeeper servers from the Ambari UI.

4.5.12.2. Zookeeper Server process

This host-level alert is triggered if the ZooKeeper server process cannot be determined to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

4.5.12.2.1. Potential causes

- The ZooKeeper server process is down on the host.
- The ZooKeeper server process is up and running but not listening on the correct network port (default 2181).
- The Nagios server cannot connect to the ZooKeeper server through the network.

4.5.12.2.2. Possible remedies

- Check for any errors in the ZooKeeper logs (`/var/log/hbase/`) and restart the ZooKeeper process using **Ambari Web**.
- Run the `netstat-tuplpn` command to check if the ZooKeeper server process is bound to the correct network port.
- Use `ping` to check the network connection between the Nagios server and the ZooKeeper server.

4.5.13. Ambari Alerts

This alert is used to monitor the Ambari Agent service.

4.5.13.1. Ambari Agent process.

This host-level alert is triggered if the Ambari Agent process cannot be confirmed to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

4.5.13.1.1. Potential causes

- The Ambari Agent process is down on the host.
- The Ambari Agent process is up and running but heartbeating to the Ambari Server.

- The Ambari Agent process is up and running but is unreachable through the network from the Nagios Server.
- The Ambari Agent cannot connect to the Ambari Server through the network.

4.5.13.1.2. Possible remedies

- Check for any errors in the logs (`/var/log/ambari-agent/ambari-agent.log`) and restart the Ambari Agent process.
- Use `ping` to check the network connection between the Ambari Agent host and the Ambari Servers.