Hortonworks Data Platform

Ambari User's Guide

(Jul 15, 2014)

docs.hortonworks.com

Hortonworks Data Platform : Ambari User's Guide

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

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

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

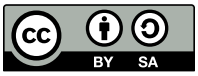


Table of Contents

List of Figures

List of Tables

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.

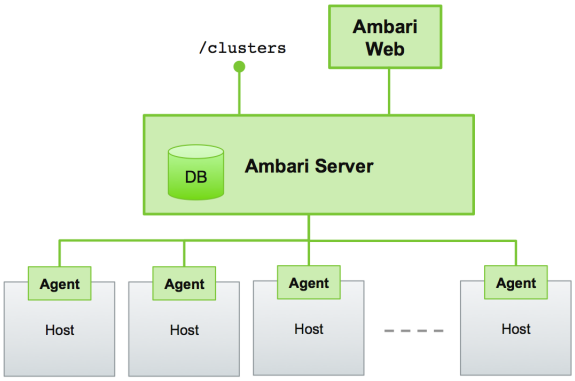
Note

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

1.1. 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



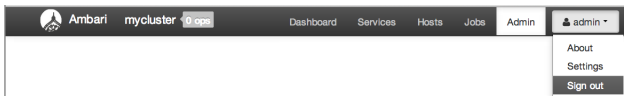
1.1.1. Sessions

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.



1.2. 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
```

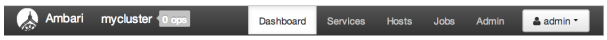
To access Ambari Web, open a supported browser and enter the Ambari Web URL:

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

Enter your user name 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. Monitoring and Managing HDP Clusters Using Ambari Web

This guide describes how to use Ambari Web features to monitor and manage your HDP cluster. To navigate, select one of the following feature tabs located at the top of the Ambari main window. A selected tab appears white.



- Viewing Metrics on the Dashboard
- Monitoring and Managing Services
- Managing Hosts
- Monitoring Jobs
- Administering Ambari

2.1. Viewing Metrics on the Dashboard

Ambari Web displays the **Dashboard** page as the home page. Use the Dashboard to view the operating status of your cluster in the following three ways:

- Scanning System Metrics
- Scanning Service Status
- Viewing Heatmaps

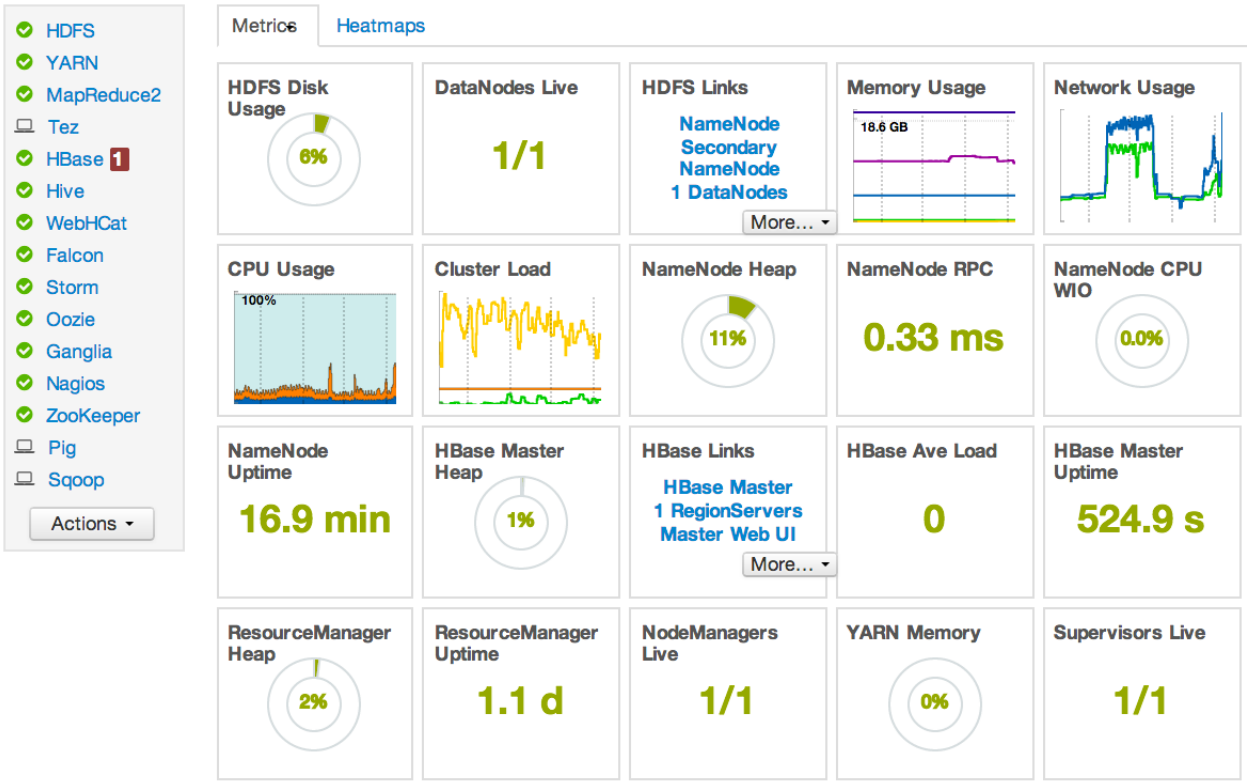
2.1.1. Scanning System Metrics

View **Metrics** that indicate the operating status of your cluster on a customizable mashup. Each metrics widget displays status information for a single service in your HDP cluster. The Ambari dashboard displays all metrics for the HDFS, YARN, HBase, and Storm services, and cluster-wide metrics by default.

Note

Metrics data for Storm is buffered and sent as a batch to Ambari every 5 minutes. After adding the Storm service, anticipate a five-minute delay for Storm metrics to appear.

You can add and remove individual widgets, and rearrange the mashup by dragging and dropping each widget to a new location in the mashup.



Status information appears as simple pie and bar charts, more complex charts showing usage and load, sets of links to additional data sources, and values for operating parameters such as uptime and average RPC queue wait times. Most widgets displays a single fact by default. For example, HDFS Disk Usage displays a load chart and a percentage figure. The Ambari Dashboard includes metrics for the following services:

Table 2.1. Ambari Service Metrics and Descriptions

Metric:	Description:
HDFS	
HDFS Disk Usage	The Percentage of DFS used, which is a combination of DFS and non-DFS used.
Data Nodes Live	The number of DataNodes live, as reported from the NameNode.
NameNode Heap	The percentage of NameNode JVM Heap used.
NameNode RPC	The average RPC queue latency.
NameNode CPU WIO	The percentage of CPU Wait I/O.
NameNode Uptime	The NameNode uptime calculation.
YARN[a]	
ResourceManager Heap	The percentage of ResourceManager JVM Heap used.
ResourceManager Uptime	The ResourceManager uptime calculation.
NodeManagers Live	The number of DataNodes live, as reported from the ResourceManager.
YARN Memory	The percentage of available YARN memory (used vs. total available).
HBase	

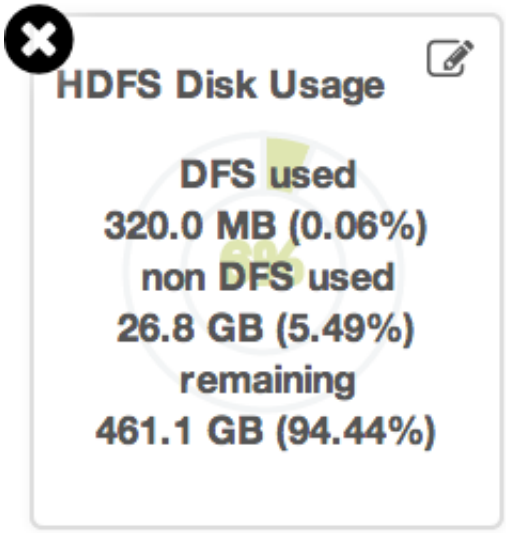
Metric:	Description:
HBase Master Heap	The percentage of NameNode JVM Heap used.
HBase Ave Load	The average load on the HBase server.
HBase Master Uptime	The HBase Master uptime calculation.
Region in Transition	The number of HBase regions in transition.
Storm[b]	
Supervisors Live	The number of Supervisors live, as reported from the Nimbus server.
MapReduce[c]	
JobTracker Heap	The percentage of JobTracker JVM Heap used.
TaskTrackers Live	The number of TaskTrackers live, as reported from the JobTracker.

[a]HDP 2.0 and 2.1 Stacks
[b]HDP 2.1 Stack
[c]HDP 1.3 Stack

2.1.1.1. Drilling Into Metrics for a Service

- To see more detailed information about a service, hover your cursor over a Metrics widget.

  More detailed information about the service displays, as shown in the following example:



- To remove a widget from the mashup, click the white X.
- To edit the display of information in a widget, click the pencil icon. For more information about editing a widget, see Customizing Metrics Display .

2.1.1.2. Viewing Cluster-Wide Metrics

Cluster Metrics display information that represents your whole cluster. The Ambari Dashboard includes widgets that display the following cluster-wide metrics:

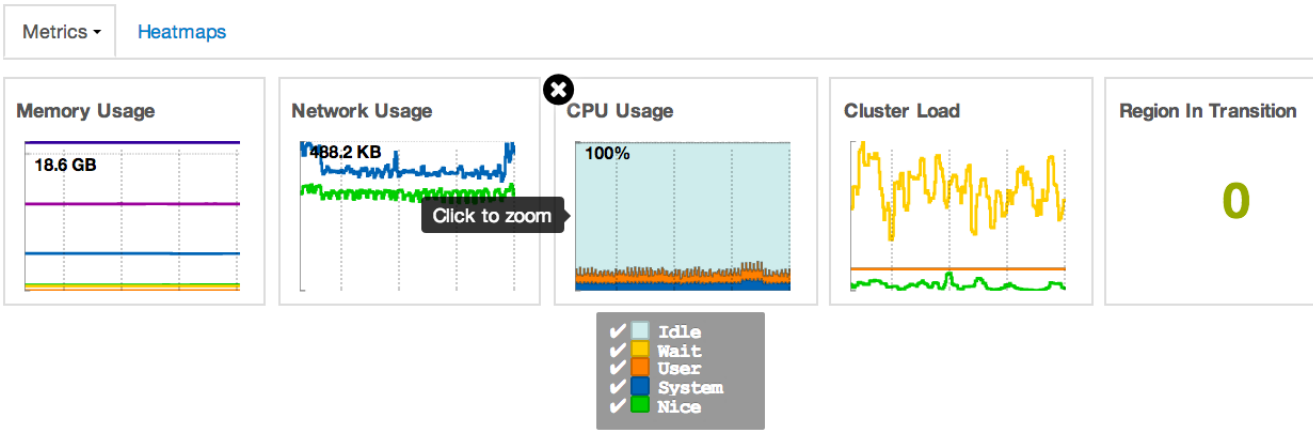
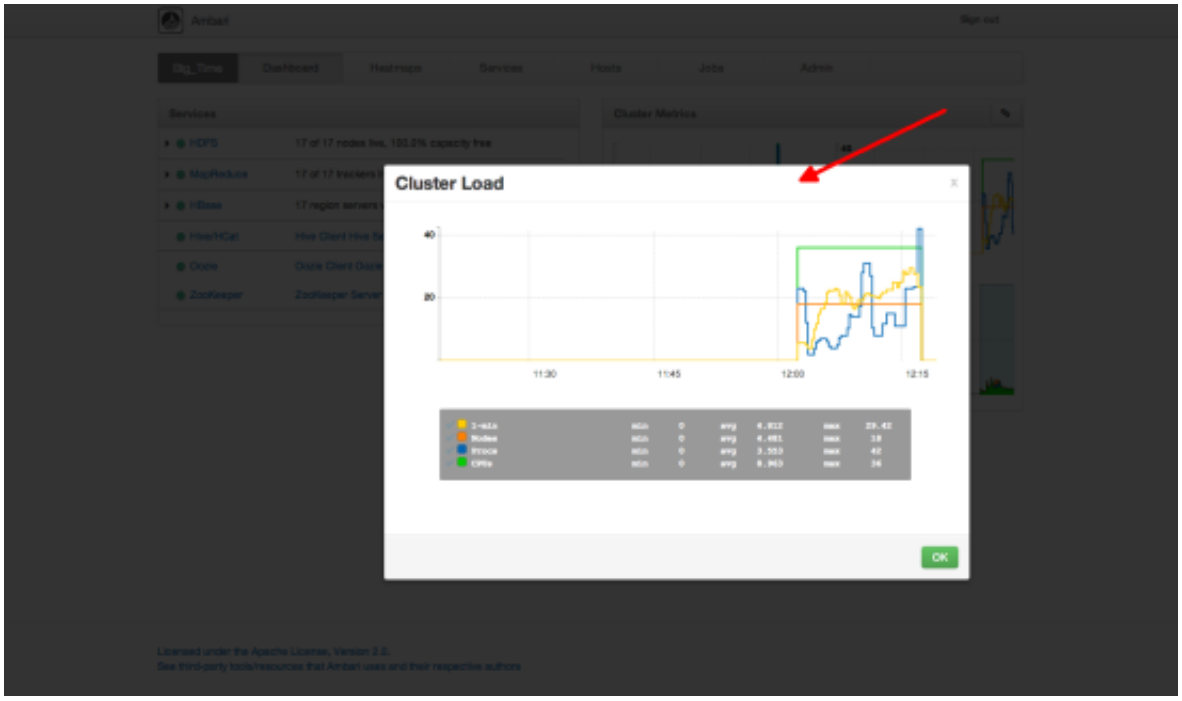


Table 2.2. Ambari Cluster-Wide Metrics and Descriptions

Metric:	Description:
Memory Usage	The cluster-wide memory utilization, including memory cached, swapped, used, shared.
Network Usage	The cluster-wide network utilization, including in-and-out.
CPU Usage	Cluster-wide CPU information, including system, user and wait IO.
Cluster Load	Cluster-wide Load information, including total number of nodes. total number of CPUs, number of running processes and 1-min Load.

- To remove a widget from the dashboard, click the white X.
- Hover your cursor over each cluster-wide metric to magnify the chart or itemize the widget display.
- To remove or add metric items from each cluster-wide metric widget, select the item on the widget legend.
- To see a larger view of the chart, select the magnifying glass icon.

  Ambari displays a larger version of the widget in a pop-out window, as shown in the following example:



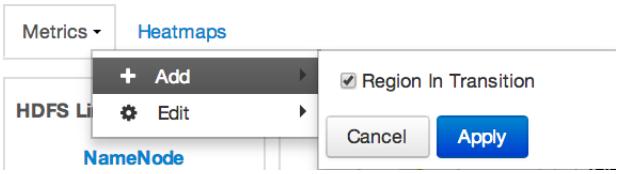
Use the pop-up window in the same ways that you use cluster-wide metric widgets on the dashboard.

To close the widget pop-up window, choose OK.

2.1.1.3. Adding a Widget to the Dashboard

To replace a widget that has been removed from the dashboard:

1. Select the Metrics drop-down, as shown in the following example:

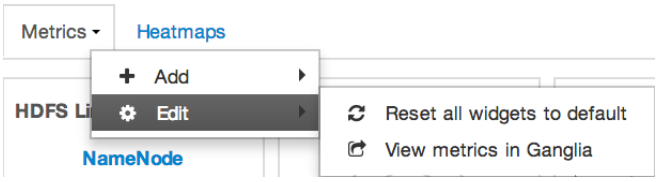


2. Choose Add.
3. Select a metric, such as Region in Transition.
4. Choose Apply.

2.1.1.4. Resetting the Dashboard

To reset all widgets on the dashboard to display default settings:

1. Select the Metrics drop-down, as shown in the following example:



2. Choose Edit.
3. Choose Reset all widgets to default.

2.1.1.5. Viewing Metrics in Ganglia

To view metrics for your cluster using the Ganglia UI:

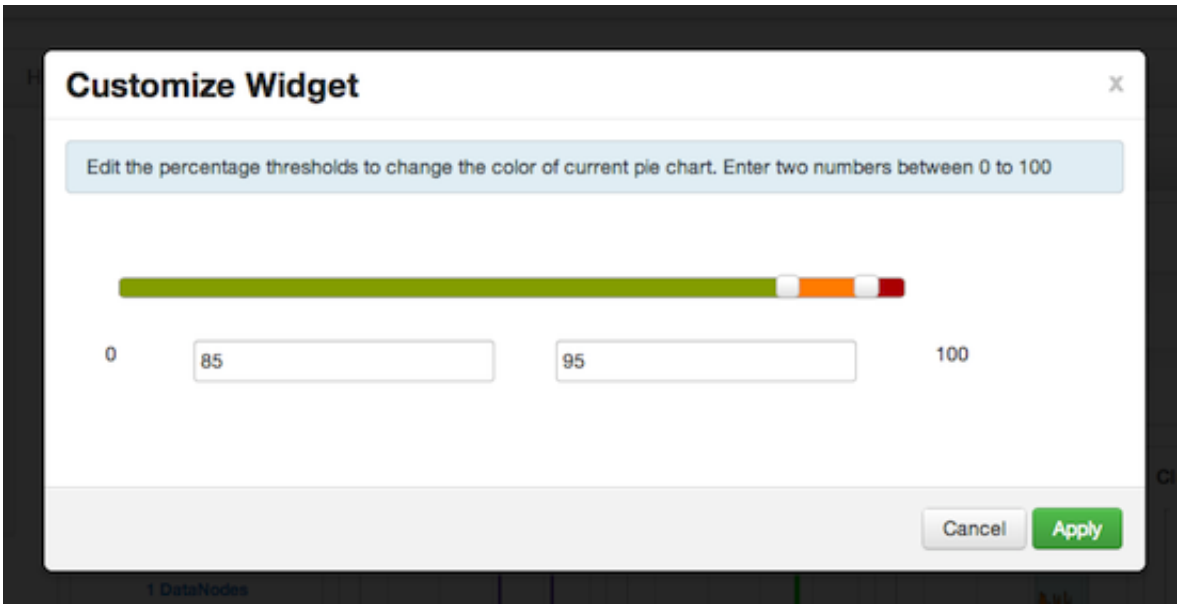
1. Select the Metrics drop-down:
2. Choose Edit.
3. Choose View Metrics in Ganglia.

2.1.1.6. Customizing Metrics Display

To customize the way a service widget displays metrics information:

1. Hover your cursor over a service widget.
2. Select the pencil-shaped, edit icon that appears in the upper-right corner.

   The Customize Widget pop-up window displays properties that you can edit, as shown in the following example.



3. Follow the instructions in the Customize Widget pop-up to customize widget appearance.

In this example, you can adjust the thresholds at which the HDFS Capacity bar chart changes color, from green to orange to red.

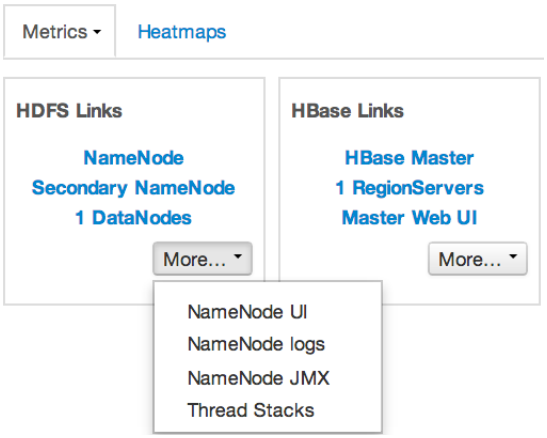
4. To save your changes and close the editor, choose **Apply**.
5. To close the editor without saving any changes, choose **Cancel**.

Note

Not all widgets support editing.

2.1.1.7. Viewing More Metrics for your HDP Stack

The HDFS Links and HBase Links widgets list HDP components for which links to more metrics information, such as thread stacks, logs and native component UIs are available. For example, you can link to NameNode, Secondary NameNode, and DataNode components for HDFS, using the links shown in the following example:



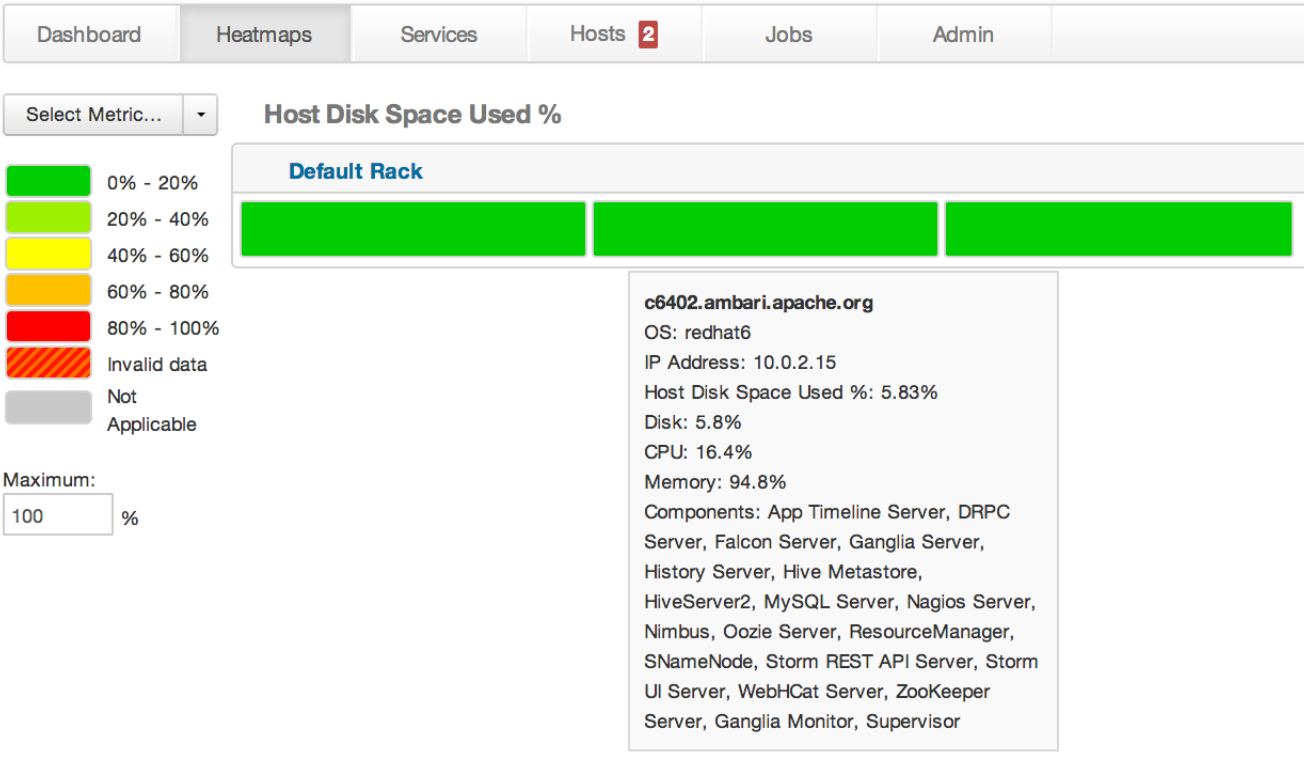
Choose the **More** drop-down to select from the list of links available for each service. The Ambari Dashboard includes **More** links to metrics for the following services:

Table 2.3. Links to More Metrics for HDP Services

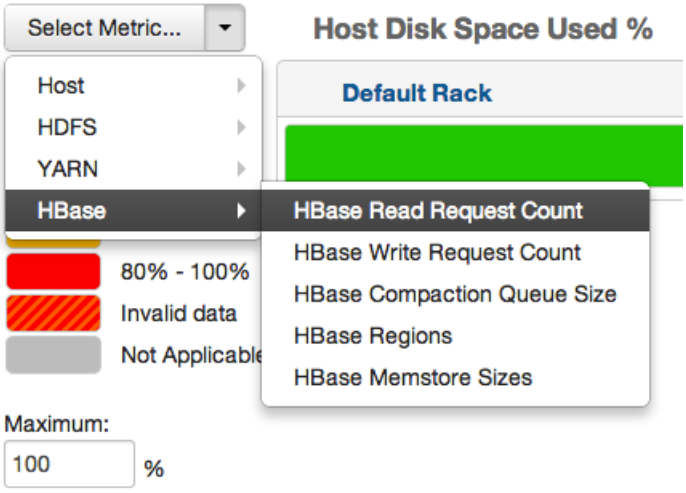
Service:	Metric:	Description:
HDFS		
	NameNode UI	Links to the NameNode UI.
	NameNode Logs	Links to the NameNode logs.
	NameNode JMX	Links to the NameNode JMX servlet.
	Thread Stacks	Links to the NameNode thread stack traces.
HBase		
	HBase Master UI	Links to the HBase Master UI.
	HBase Logs	Links to the HBase logs.
	ZooKeeper Info	Links to ZooKeeper information.
	HBase Master JPX	Links to the HBase Master JMX servlet.
	Debug Dump	Links to debug information.
	Thread Stacks	Links to the HBase Master thread stack traces.

2.1.2. Viewing Heatmaps

Heatmaps provides a graphical representation of your overall cluster utilization using simple color coding.



A colored block represents each host in your cluster. To see more information about a specific host, hover over the block representing the host in which you are interested. A pop-up window displays metrics about HDP components installed on that host. Colors displayed in the block represent usage in a unit appropriate for the selected set of metrics. If any data necessary to determine state is not available, the block displays "Invalid Data". Changing the default maximum values for the heatmap lets you fine tune the representation. Use the Select Metric drop-down to select the metric type.



Heatmaps supports the following metrics:

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

2.1.3. Scanning Services Status

Notice the color of the dot appearing next to each service name in the list of **Services**. The dot color and blinking action indicates operating status of each service. For example, in the Dashboard Widget [3] image, notice green dot next to each service name. The following colors and actions indicate service status:

Table 2.4. Service Status

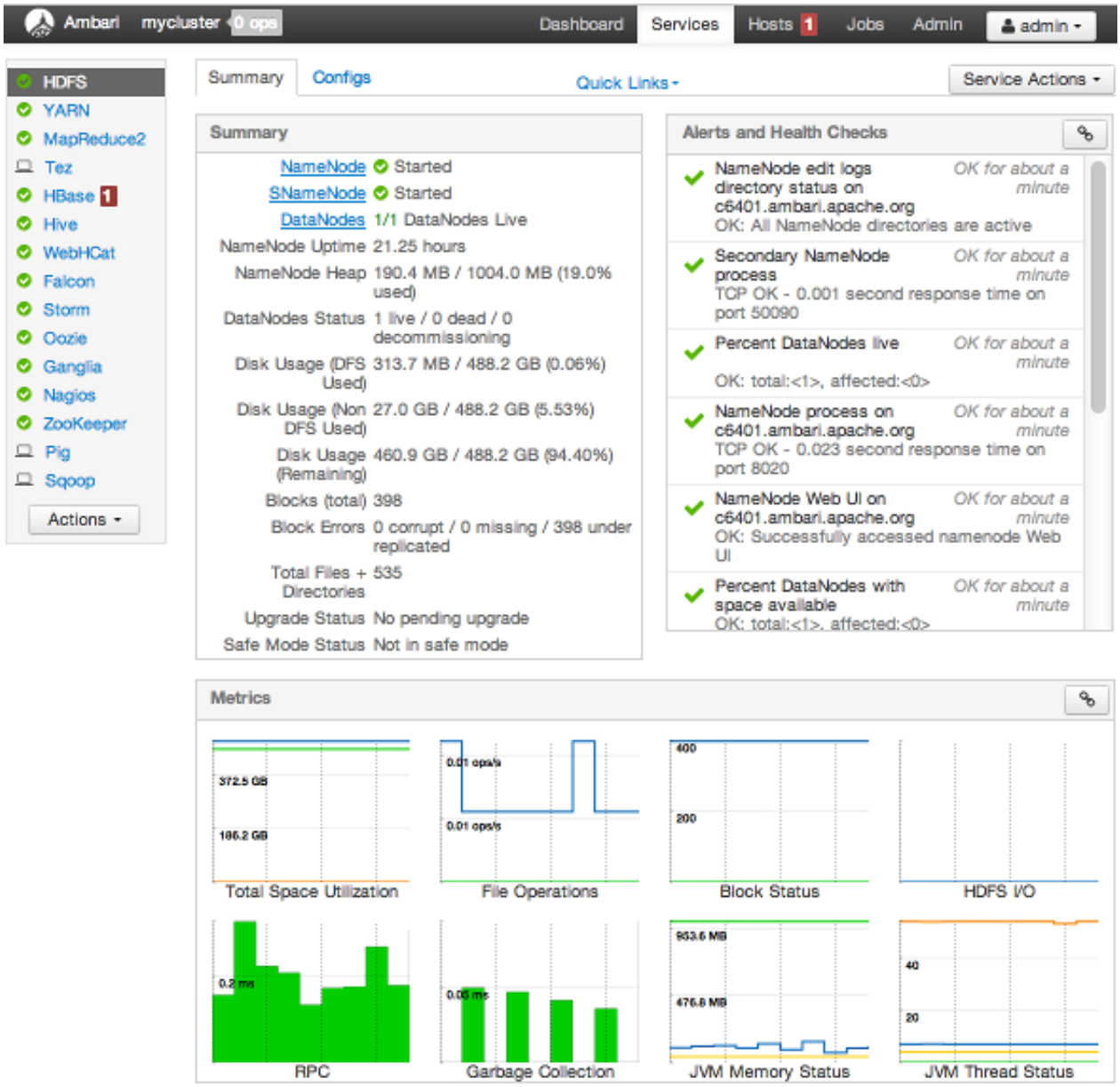
Color	Name	Status
	Solid Green	All masters are running
	Blinking Green	Starting up
	Solid Red	At least one master is down
	Blinking Red	Stopping

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

2.2. Monitoring and Managing Services

Use **Services** to monitor and manage selected services running in your Hadoop cluster.

All services installed in your cluster are listed in the leftmost **Services** panel.

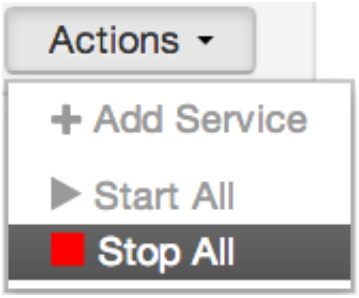


Services supports the following tasks:

- Starting and Stopping All Services
- Selecting a Service
- Viewing Summary, Alert, and Health Information
- Configuring Services
- Rolling Restarts
- Using Quick Links
- Analyzing Service Metrics

2.2.1. Starting and Stopping All Services

To start or stop all listed services at once, select **Actions**, then choose **Start All** or **Stop All**, as shown in the following example:



2.2.2. Selecting a Service

Selecting a service name from the list shows current summary, alert, and health information for the selected service. To refresh the monitoring panels and show information about a different service, select a different service name from the list.

Notice the colored dot next to each service name, indicating service operating status and a small, red, numbered rectangle indicating any alerts generated for the service.

2.2.2.1. Adding a Service

The Ambari install wizard installs all available Hadoop services by default. You may choose to deploy only some services initially, then add other services at later times. For example, many customers deploy only core Hadoop services initially. **Add Service** supports deploying additional services without interrupting operations in your Hadoop cluster. When you have deployed all available services, **Add Service** displays disabled.

For example, if you are using HDP 2.1 Stack and did not install Falcon or Storm, you can use the **Add Service** capability to add those services to your cluster.

Note

After installing Storm via Ambari, you should configure the Storm service to run as a supervised service.

For more information, see Configuring Storm for Supervision.

To add a service, select **Actions** -> **Add Service**, then complete the following procedure using the Add Service Wizard.

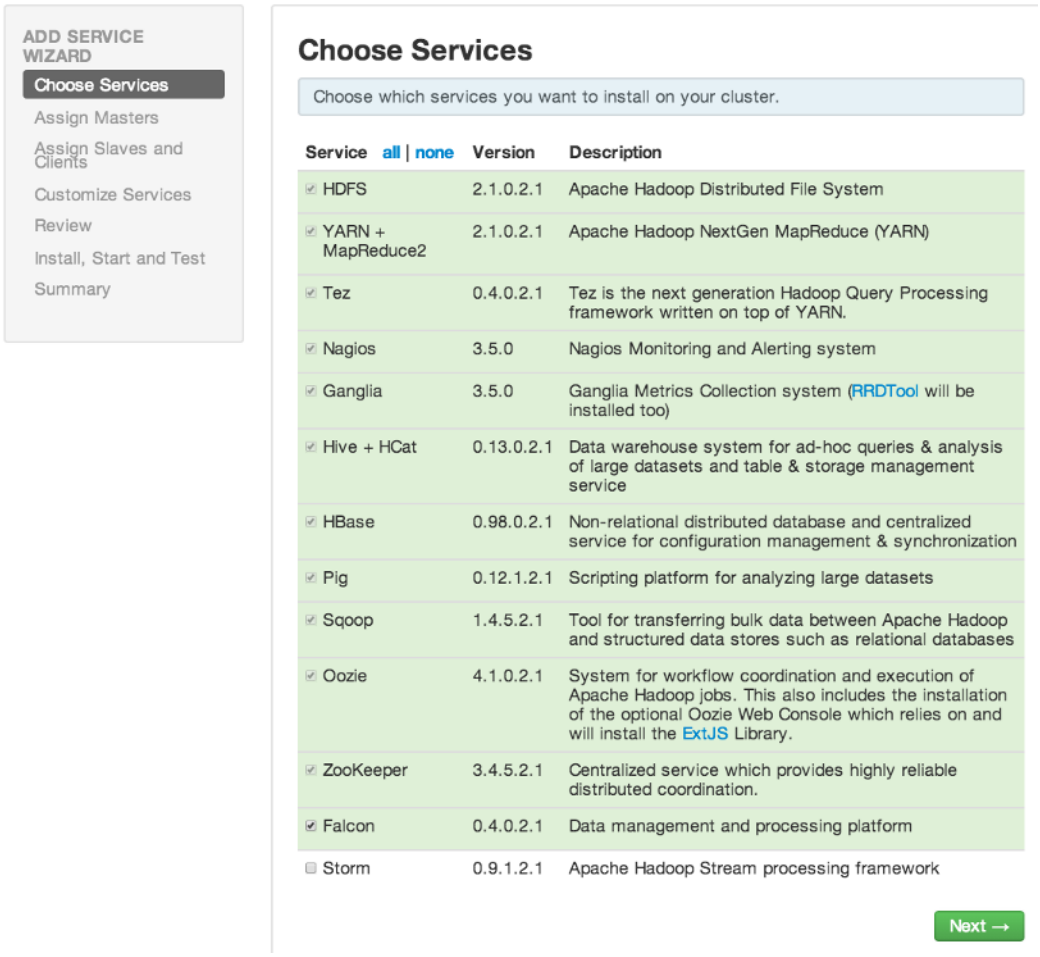
2.2.2.1.1. Adding a Service to your Hadoop cluster

This example shows the Falcon service selected for addition.

1. Choose Services.

   Choose an available service. Alternatively, choose all to add all available services to your cluster. Then, choose Next.

The Add Services wizard displays installed services highlighted green and marked unavailable for selection.



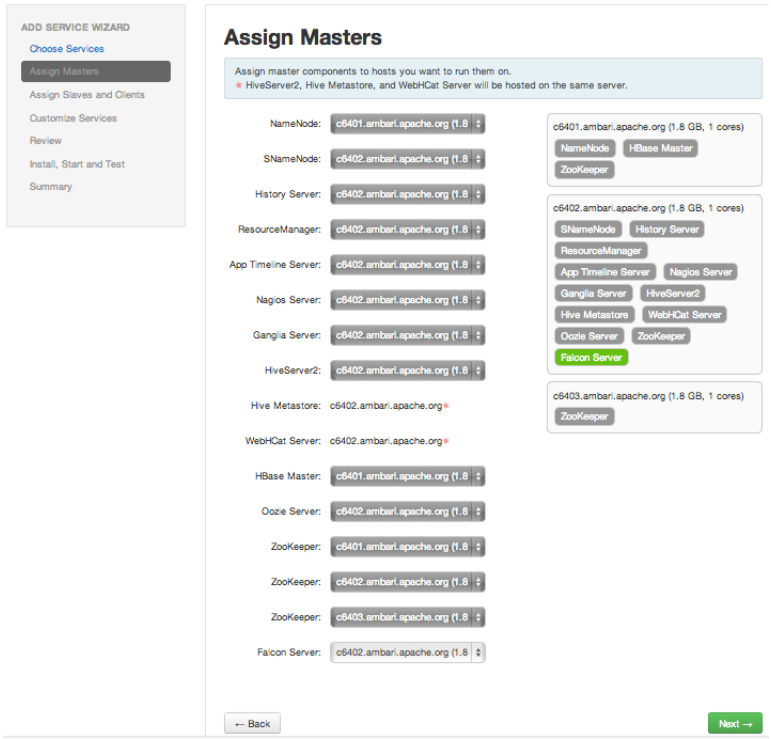
2. In **Assign Masters**, confirm the default host assignment. Alternatively, choose a different host machine to which master components for your selected service will be added. Then, choose Next.

   The Add Services Wizard indicates hosts on which the master components for a chosen service will be installed. A service chosen for addition displays as:

   - A green label located on the host to which its master components will be added, or

   - An active drop-down list on which available host names appear.

   Using the drop-down, choose an alternate host name, if necessary.



3. In Assign Slaves and Clients, accept the default assignment of slave and client components to hosts. Then, choose Next.

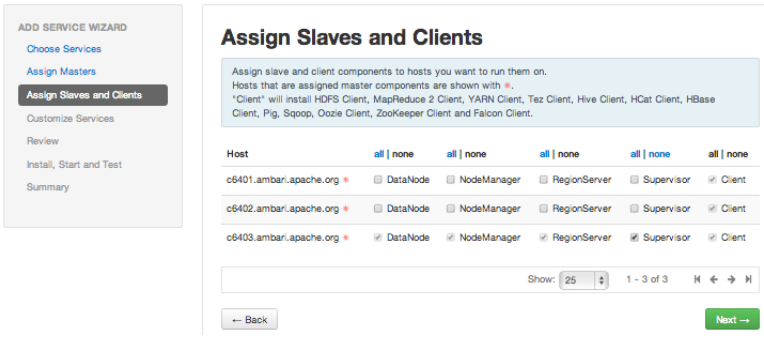
   Alternatively, select hosts to which you want to assign slave and client components.

   You must select at least one host for the slave of each service being added.

Table 2.5. Host Roles Required for Added Services

Service Added	Host Role Required
MapReduce	TaskTracker
YARN	NodeManager
HBase	RegionServer

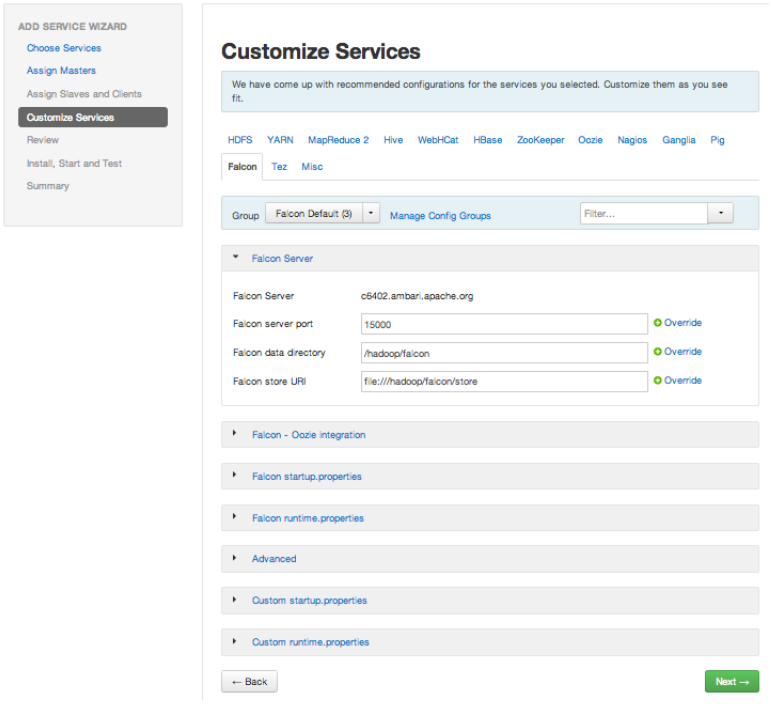
The Add Service Wizard skips and disables the Assign Slaves and Clients step for a service requiring no slave nor client assignment.



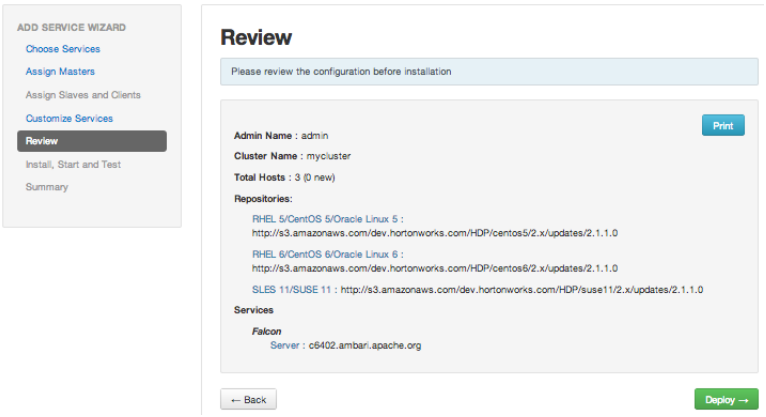
4. In Customize Services, accept the default configuration properties. Then, choose Next.

Alternatively, edit the default values for configuration properties, if necessary.

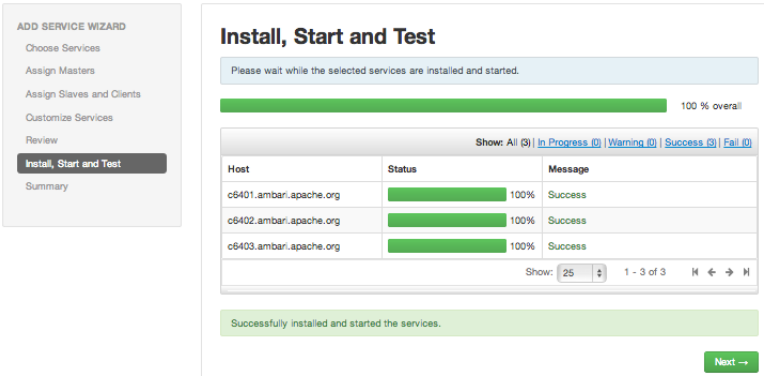
Choose Override to create a configuration group for this service.



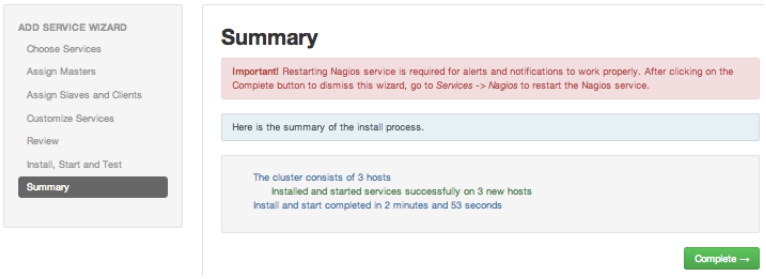
5. In Review, make sure the configuration settings match your intentions. Then, choose Deploy.



6. Monitor the progress of installing, starting, and testing the service. When the service installs and starts successfully, choose Next.



7. Summary displays the results of installing the service. Choose Complete.



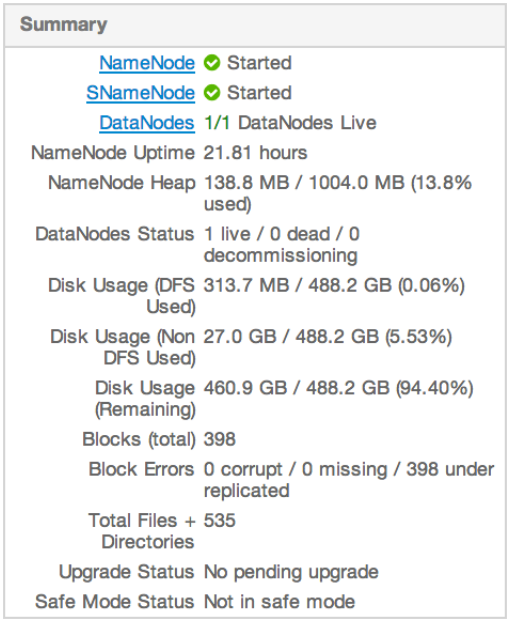
8. Restart the Nagios service and any other components having stale configurations.

Important

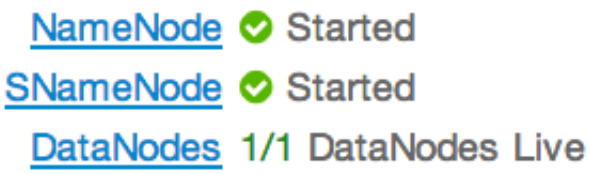
If you do not restart Nagios service after completing the Add Service Wizard, alerts and notifications may not work properly.

2.2.3. Viewing Summary, Alert, and Health Information

After you select a service, **Summary** tab displays basic information about the selected service.

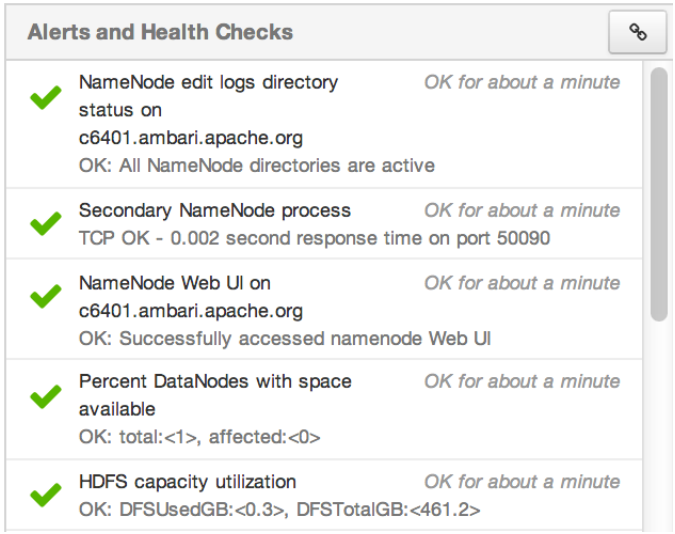


Select a **View Host** link, as shown in the following example, to view components and the host on which the selected service is running.



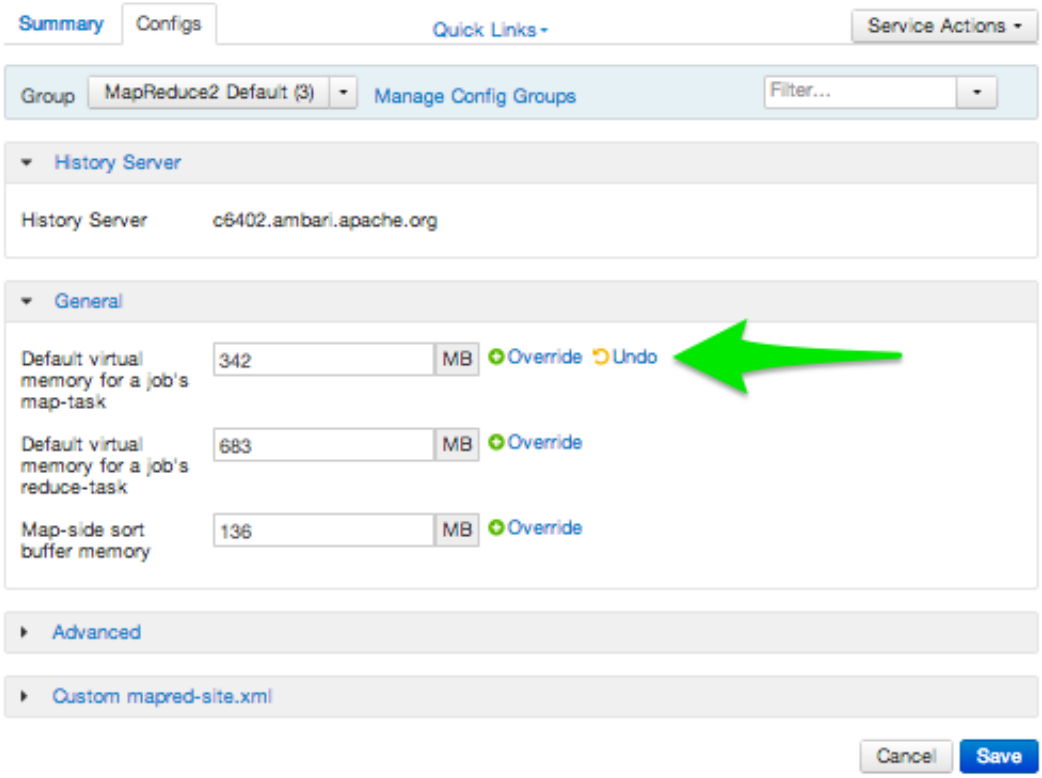
2.2.3.1. Alerts and Health Checks

View results of the health checks performed on your cluster by Nagios in **Alerts and Health Checks**. Alerts and Health Checks displays a list of each issue and its rating, sorted first by descending severity, then by descending time. To access more detailed information, select the native Nagios GUI link located at the upper right corner of the panel. Use the Nagios credentials you set up during installation to log in to Nagios.



2.2.4. Configuring Services

Select a service, then select **Configs** to view and update configuration properties for the selected service. For example, select MapReduce2, then select Configs. Expand a config category to view configurable service properties.



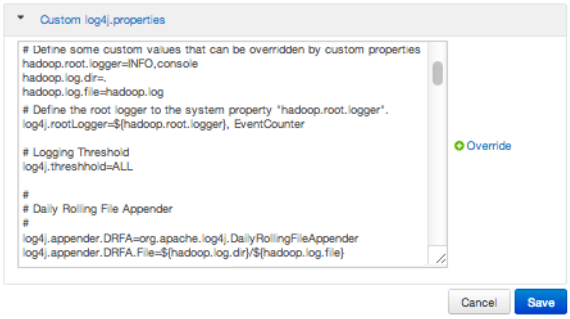
2.2.4.1. Updating Service Properties

1. Expand a configuration category.
2. Edit values for one or more properties that have the Override option.

   Edited values, also called stale configs, show an Undo option.
3. Choose Save.

2.2.4.2. Customizing Logging Properties

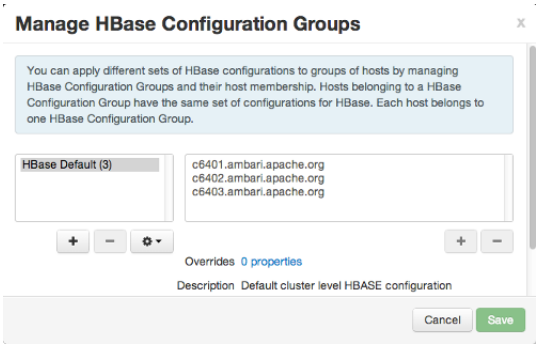
Ambari Web displays default logging properties in **Service Configs > Custom log 4j Properties**. Log 4j properties control logging activities for the selected service.



Restarting components in the service pushes the configuration properties displayed in Custom log 4j Properties to each host running components for that service. If you have customized logging properties that define how activities for each service are logged, you will see refresh indicators next to each service name after upgrading to Ambari 1.5.0 or higher. Make sure that logging properties displayed in Custom log 4j Properties include any customization. Optionally, you can create configuration groups that include custom logging properties. For more information about saving and overriding configuration settings, see Managing Configuration Groups.

2.2.4.3. Managing Configuration Groups

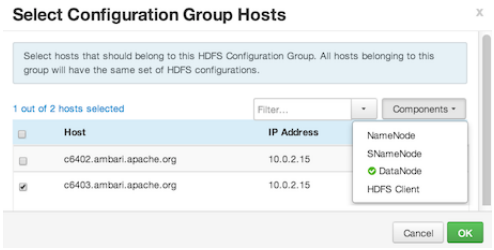
Ambari initially assigns all hosts in your cluster to one, default configuration group for each service you install. For example, after deploying a three-node cluster with default configuration settings, each host belongs to one configuration group that has default configuration settings for the HDFS service. In Configs, select **Manage Config Groups**, to create new groups, re-assign hosts, and override default settings for host components you assign to each group.



To create a Configuration Group:

1. Choose Add New Configuration Group.
2. Name and Describe the group, then choose Save.
3. Select a Config Group, then choose Add Hosts to Config Group.
4. Select Components and choose from available Hosts to add hosts to the new group.

   Select Configuration Group Hosts enforces host membership in each group, based on installed components for the selected service.



5. Choose OK.

6. In Manage Configuration Groups, choose Save.

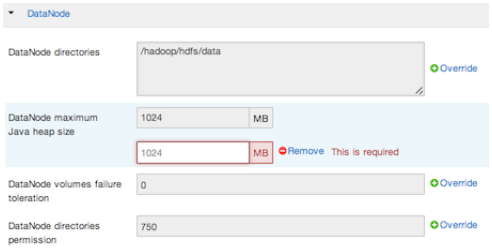
To edit settings for a configuration group:

1. In Configs, choose a Group.

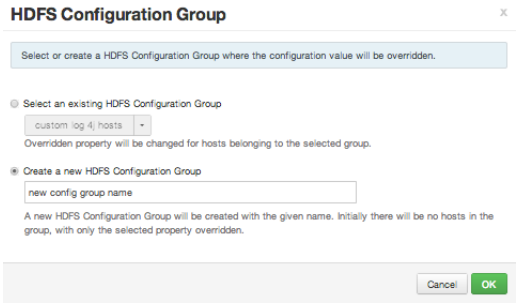
2. Select a Config Group, then expand components to expose settings that allow Override.

3. Provide a non-default value, then choose Override or Save.

   Configuration groups enforce configuration properties that allow override, based on installed components for the selected service and group.



4. Override prompts you to choose one of the following options:



   a. Select an existing configuration group (to which the property value override provided in step 3 will apply), or

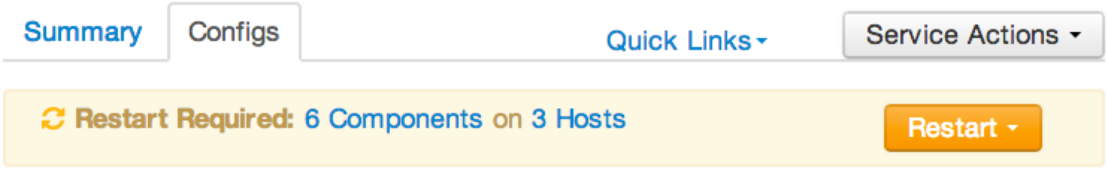
   b. Create a new configuration group (which will include default properties, plus the property override provided in step 3).

   c. Then, choose OK.

5. In Configs, choose Save.

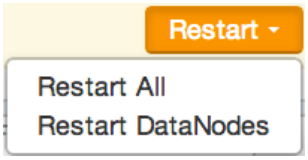
2.2.4.4. Restarting components

After editing and saving a service configuration, Restart indicates components that you must restart.



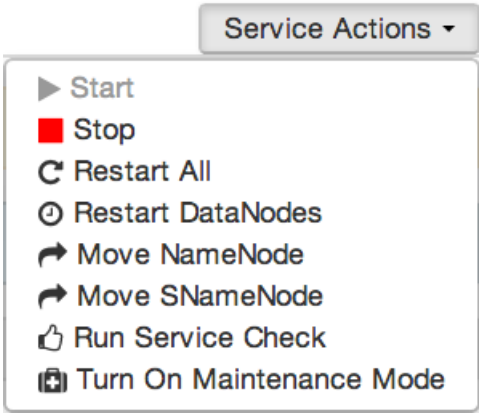
Select the Components or Hosts links to view details about components or hosts requiring a restart.

Then, choose an option appearing in Restart. For example, options to restart YARN components include:



2.2.4.5. Performing Service Actions

Manage a selected service on your cluster by performing service actions. In **Services**, select the **Service Actions** drop-down menu, then choose an option. Available options depend on the service you have selected. For example, HDFS service action options include:



Optionally, choose **Turn On Maintenance Mode** to suppress alerts about this service before performing a service action. Maintenance Mode suppresses alerts and status indicator changes generated by the service, while allowing you to start, stop, restart, move, or perform maintenance tasks on the service. For more information about how Maintenance Mode affects bulk operations for host components, see Maintenance Mode.

2.2.4.6. Rolling Restarts

When you restart multiple services, components, or hosts, use rolling restarts to distribute the task; minimizing cluster downtime and service disruption. A rolling restart stops,

then starts multiple, running slave components such as DataNodes, TaskTrackers, NodeManagers, RegionServers, or Supervisors, using a batch sequence. You set rolling restart parameter values to control the number of, time between, tolerance for failures, and limits for restarts of many components across large clusters.

To run a rolling restart:

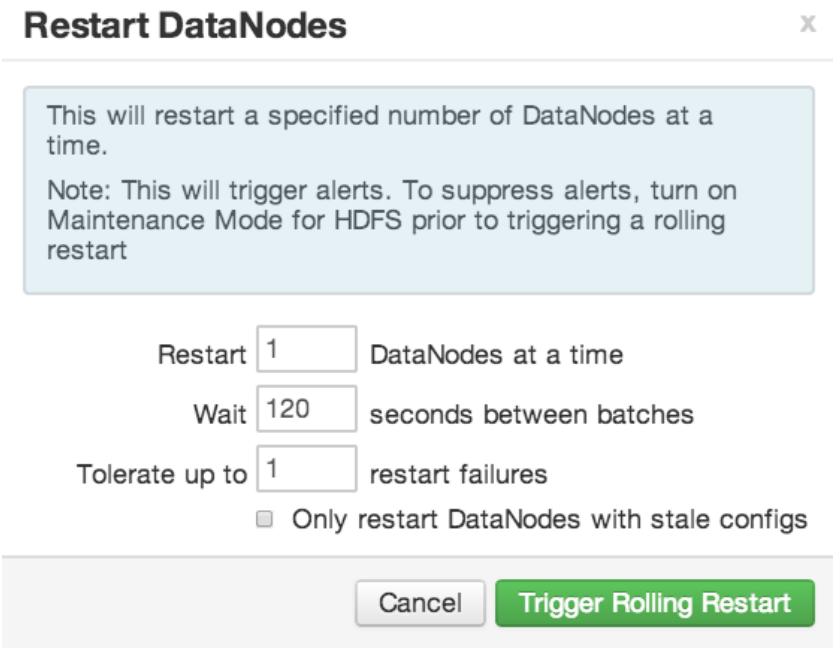
1. Select a Service, then link to a lists of specific components or hosts that Require Restart.
2. Select Restart, then choose a slave component option.
3. Review and set values for Rolling Restart Parameters.
4. Optionally, reset the flag to only restart components with changed configurations.
5. Choose Trigger Restart.

Use Background Operations to monitor progress of rolling restarts.

2.2.4.6.1. Setting Rolling Restart Parameters

When you choose to restart slave components, use parameters to control how restarts of components roll. Parameter values based on ten percent of the total number of components in your cluster are set as default values. For example, default settings for a rolling restart of components in a 3-node cluster restarts 1 component at a time, waits 2 minutes between restarts, will proceed if only one failure occurs, and restarts all existing components that run this service.

If you trigger a rolling restart of components, Restart components with stale configs defaults to true. If you trigger a rolling restart of services, Restart services with stale configs defaults to false.



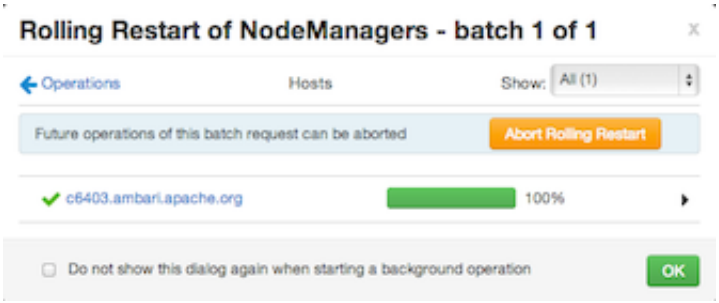
Rolling restart parameter values must satisfy the following criteria:

Table 2.6. Validation Rules for Rolling Restart Parameters

Parameter	Required	Value	Description
Batch Size	Yes	Must be an integer > 0	Number of components to include in each restart batch.
Wait Time	Yes	Must be an integer > = 0	Time (in seconds) to wait between queuing each batch of components.
Tolerate up to x failures	Yes	Must be an integer > = 0	Total number of restart failures to tolerate, across all batches, before halting the restarts and not queuing batches.

2.2.4.6.2. Aborting a Rolling Restart

To abort future restart operations in the batch, choose Abort Rolling Restart.

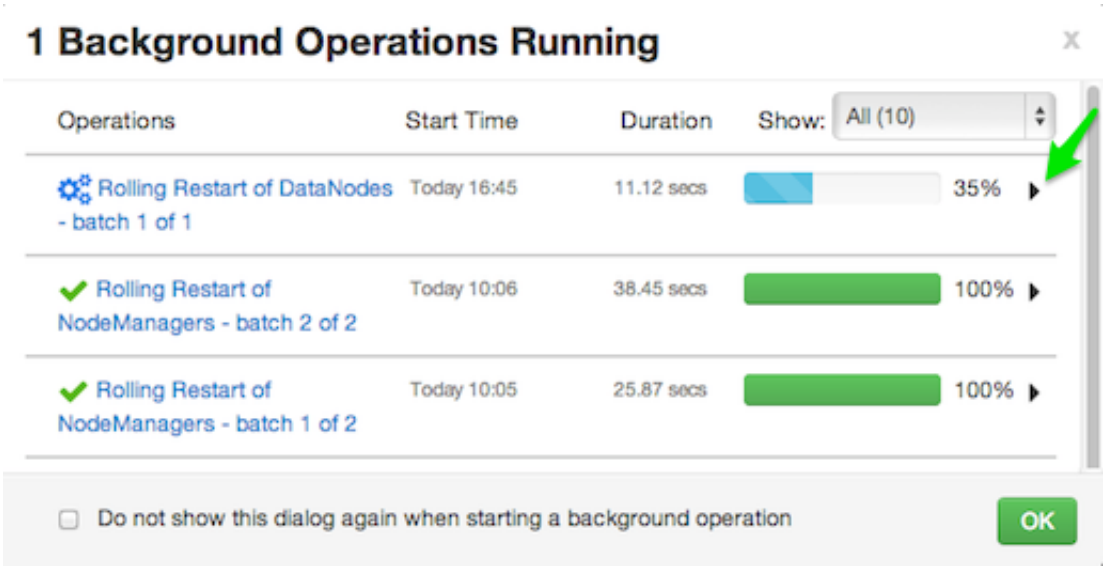


2.2.4.7. Monitoring Background Operations

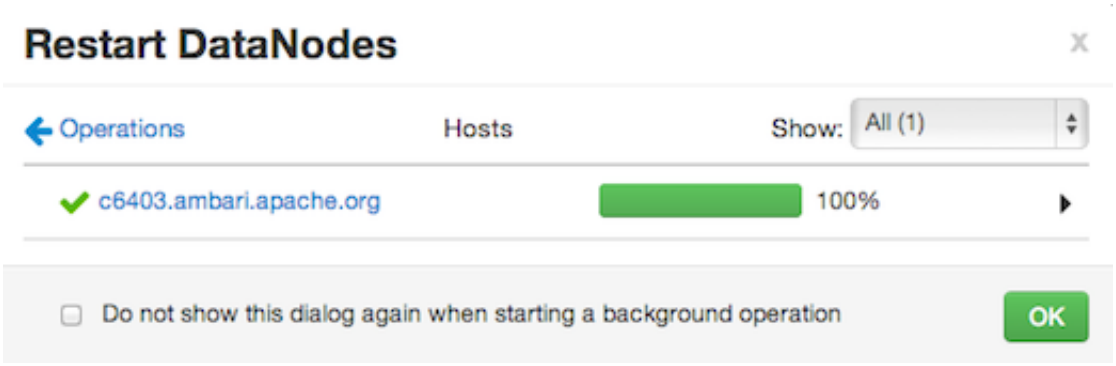
Optionally, use Background Operations to monitor progress and completion of bulk operations such as rolling restarts.

Background Operations opens by default when you run a job that executes bulk operations.

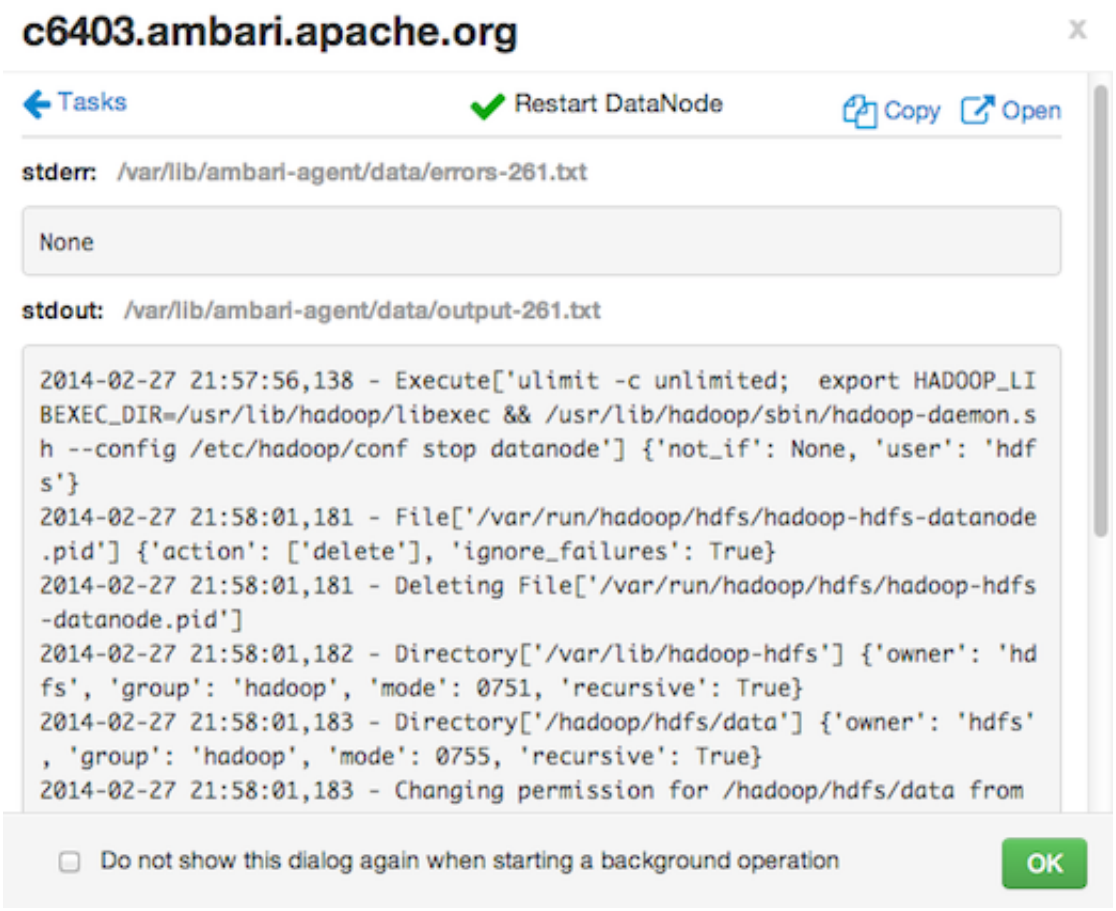
1. Select the right-arrow for each operation to show restart operation progress on each host.



2. After restarts complete, Select the right-arrow, or a host name, to view log files and any error messages generated on the selected host.



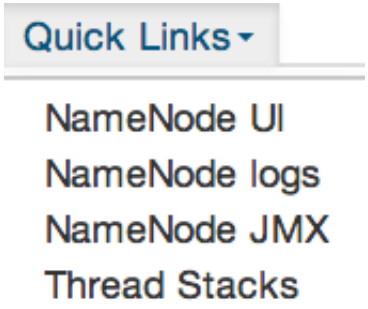
3. Select links at the upper-right to copy or open text files containing log and error information.



Optionally, select the option to not show the bulk operations dialog.

2.2.4.8. Using Quick Links

Select **Quick Links** options to access additional sources of information about a selected service. For example, HDFS Quick Links options include the native NameNode GUI, NameNode logs, the NameNode JMX output, and thread stacks for the HDFS service. Quick Links are not available for every service.



2.2.4.9. Analyzing Service Metrics

Review visualizations in**Metrics** that chart common metrics for a selected service. Services > Summary displays metrics widgets for HDFS, HBase, Storm services. For more

information about using metrics widgets, see Scanning System Metrics. To see more metrics information, select the link located at the upper right of the Metrics panel that opens he native Ganglia GUI.

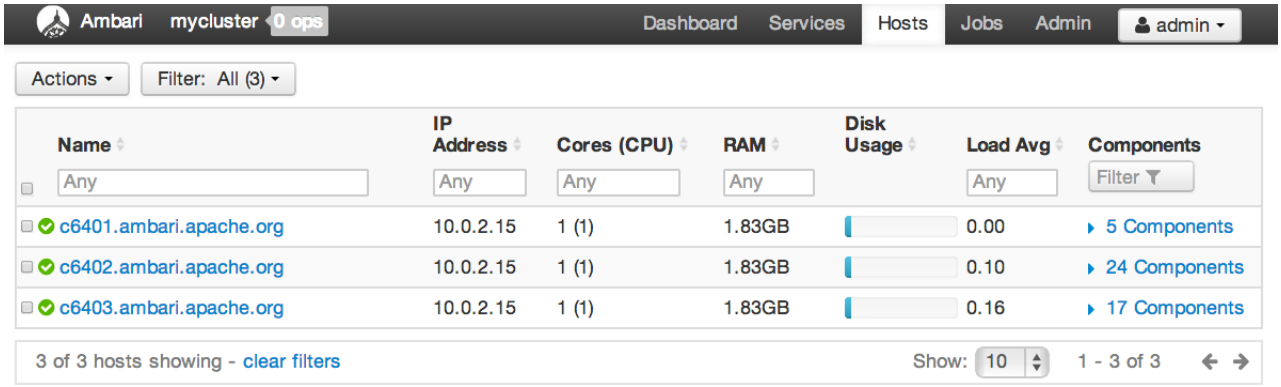
2.3. Managing Hosts

Use Ambari Hosts to manage multiple HDP components such as DataNodes, NameNodes, TaskTrackers and RegionServers, running on hosts throughout your cluster. For example, you can restart all DataNode components, optionally controlling that task with rolling restarts. Ambari Hosts supports filtering your selection of host components, based on operating status, host health, and defined host groupings.

2.3.1. Working with Hosts

Use Hosts to view hosts in your cluster on which Hadoop services run. Use options on **Actions** to perform actions on one or more hosts in your cluster.

View individual hosts, listed by fully-qualified domain name, on the Hosts landing page.



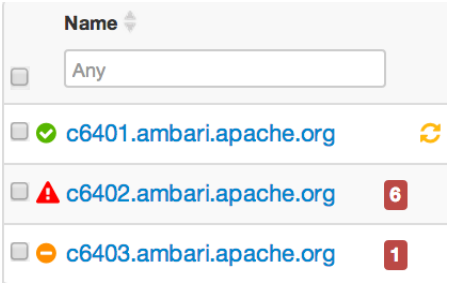
2.3.2. Determining Host Status

A colored dot beside each host name indicates operating status of each host, as follows:

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

A red condition flag overrides an orange condition flag, which overrides a yellow condition flag. In other words, a host having a master component down may also have other issues. The following example shows three hosts, one having a master component down, one

having a slave component down, and one healthy. Warning indicators appear next to hosts having a component down.



2.3.3. Filtering the Hosts List

Use Filters to limit listed hosts to only those having a specific operating status. The number of hosts in your cluster having a listed operating status appears after each status name, in parenthesis. For example, the following cluster has one host having healthy status and three hosts having Maintenance Mode turned on.

All (3)
Healthy (1)
Master Down (2)
Slave Down (0)
Lost Heartbeat (0)
Alerts (2)
Restart (2)
Maintenance Mode (0)

For example, to limit the list of hosts appearing on Hosts home to only those with Healthy status, select Filters, then choose the Healthy option. In this case, one host name appears on Hosts home. Alternatively, to limit the list of hosts appearing on Hosts home to only those having Maintenance Mode on, select Filters, then choose the Maintenance Mode option. In this case, three host names appear on Hosts home.

Use the general filter tool to apply specific search and sort criteria that limits the list of hosts appearing on the Hosts page.

2.3.4. Performing Host-Level Actions

Use Actions to act on one, or multiple hosts in your cluster. Actions performed on multiple hosts are also known as bulk operations.

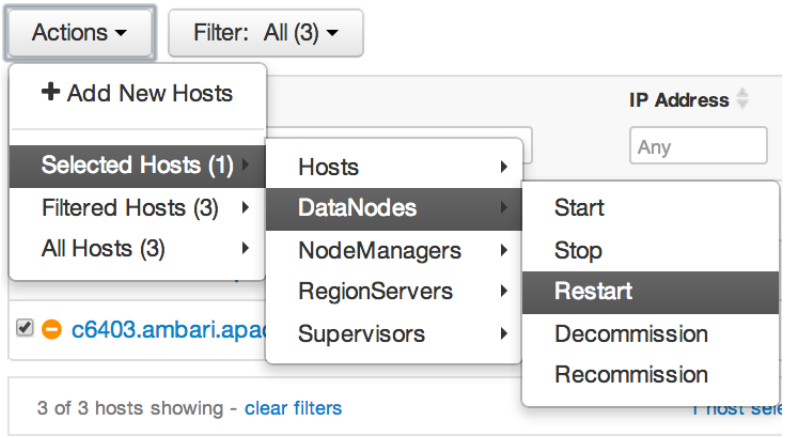
Actions comprises three menus that list the following options types:

- Hosts - lists selected, filtered or all hosts options, based on your selections made using Hosts home and Filters.
- Objects - lists component objects that match your host selection criteria.
- Operations - lists all operations available for the component objects you selected.

For example, to restart DataNodes on one host:

1. In Hosts, select a host running at least one DataNode.

2. In Actions, choose Selected Hosts > DataNodes > Restart, as shown in the following image.

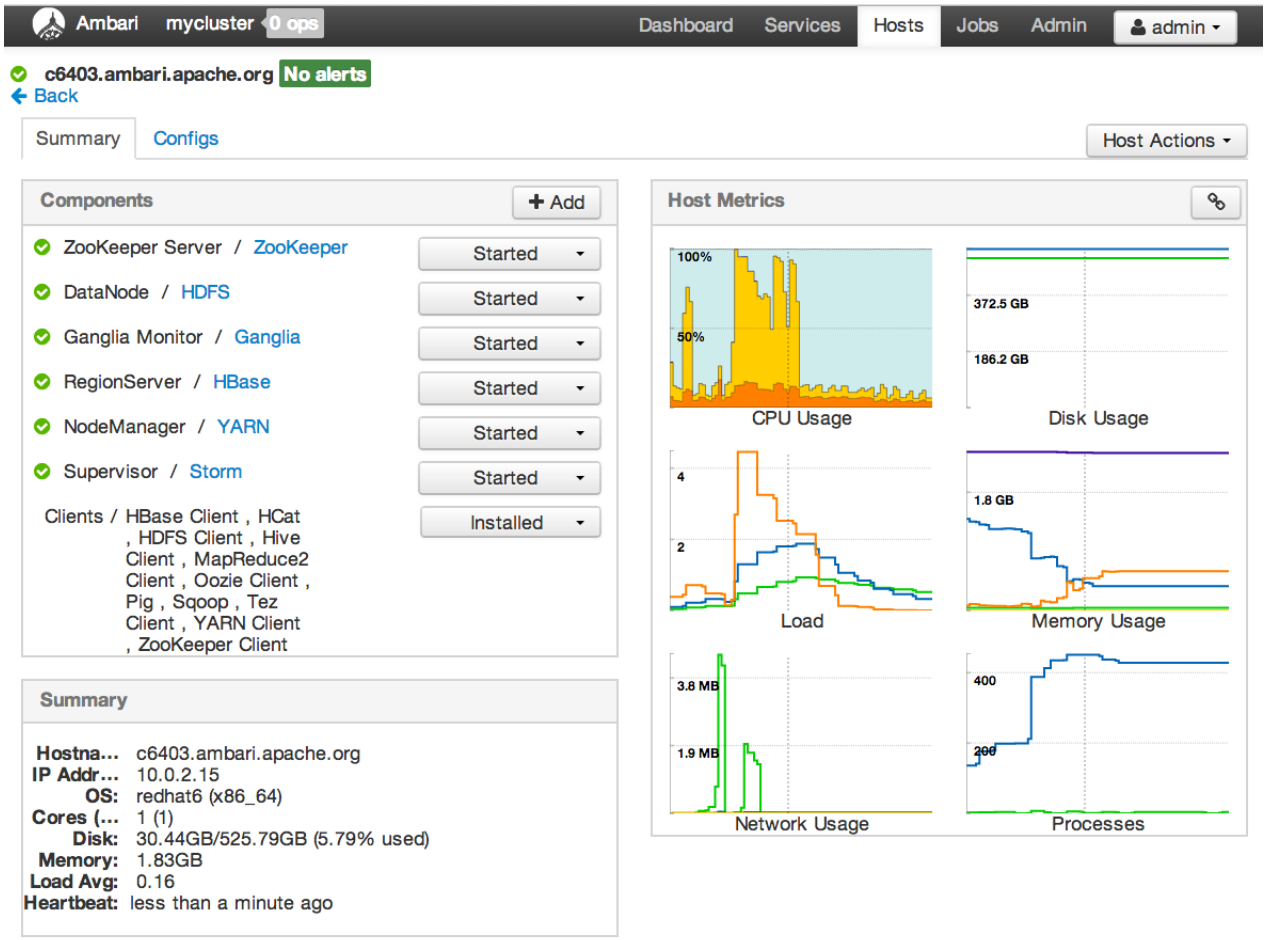


3. Choose OK to confirm starting the selected operation.

4. Optionally, use Background Operations to monitor progress of the restart operation.

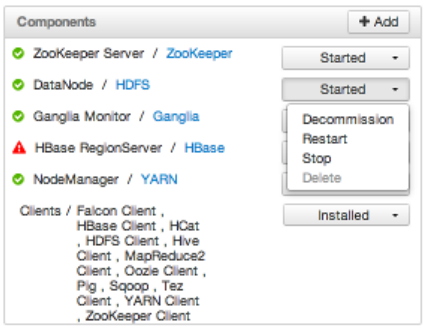
2.3.5. Viewing Components on a Host

To manage components running on a specific host, choose a FQDN on the Hosts page. For example, choose c6403.ambari.apache.org in the default example shown. Summary-Components lists all components installed on that host.



Choose options in **Host Actions**, to start, stop, restart, delete, or turn on maintenance mode for all components installed on the selected host.

Alternatively, choose action options from the drop-down menu next to an individual component on a host. The drop-down menu shows current operation status for each component, For example, you can decommission, restart, or stop the DataNode component (started) for HDFS, by selecting one of the options shown in the following example:



2.3.6. Decommissioning Masters and Slaves

Decommissioning is a process that supports removing a component from the cluster. You must decommission a master or slave running on a host before removing the component or host from service. Decommissioning helps prevent potential loss of data or service disruption. Decommissioning is available for the following component types:

- DataNodes
- NodeManagers
- TaskTrackers
- RegionServers

Decommissioning executes the following tasks:

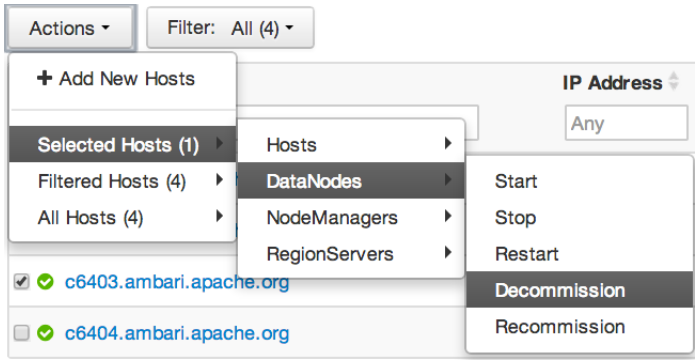
- For DataNodes, safely replicates the HDFS data to other DataNodes in the cluster.
- For NodeManagers and TaskTrackers, stops accepting new job requests from the masters and stops the component.
- For RegionServers, turns on drain mode and stops the component.

2.3.6.1. How to Decommission a Component

To decommission a component using Ambari Web, browse **Hosts** to find the host FQDN on which the component resides.

Using **Actions**, select **Hosts Component Type**, then choose **Decommission**.

For example:



The UI shows "Decommissioning" status while steps process, then "Decommissioned" when complete.

2.3.6.2. How to Delete a Component

To delete a component using Ambari Web, on **Hosts** choose the host FQDN on which the component resides.

1. In **Components**, find a decommissioned component.
2. Stop the component, if necessary.

Note

A decommissioned slave component may restart in the decommissioned state.

3. For a decommissioned component, choose **Delete** from the component drop-down menu.
4. Restart the Ganglia and Nagios services.

Note

Restarting services enables Ambari to recognize and monitor the correct number of components.

Deleting a slave component, such as a DataNode does not automatically inform a master component, such as a NameNode to remove the slave component from its exclusion list. Adding a deleted slave component back into the cluster presents the following issue; the added slave remains decommissioned from the master's perspective. Restart the master component, as a work-around.

2.3.7. Deleting a Host from a Cluster

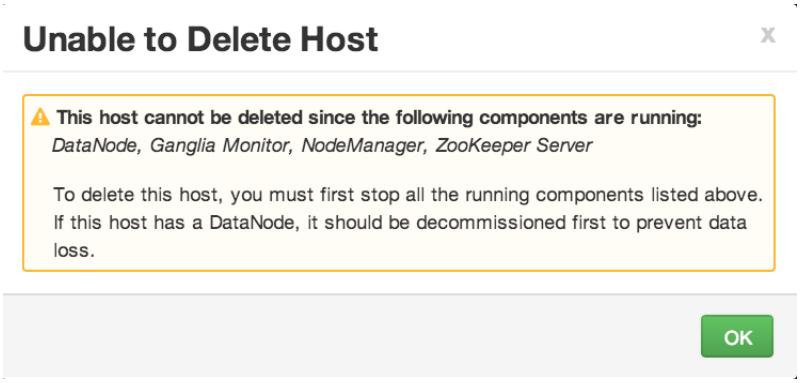
Deleting a host removes the host from the cluster. Before deleting a host, you must complete the following prerequisites:

- Stop all components running on the host.
- Decommission any DataNodes running on the host.
- Move from the host any master components, such as NameNode or ResourceManager, running on the host.
- Turn Off Maintenance Mode, if necessary, for the host.

2.3.7.1. How to Delete a Host from a Cluster

1. In Hosts, click on a host name.
2. On the Host-Details page, select Host Actions drop-down menu.
3. Choose Delete.

If you have not completed prerequisite steps, a warning message similar to the following one appears:



2.3.8. Setting Maintenance Mode

Maintenance Mode supports suppressing alerts and skipping bulk operations for specific services, components and hosts in an Ambari-managed cluster. You typically turn on Maintenance Mode when performing hardware or software maintenance, changing configuration settings, troubleshooting, decommissioning, or removing cluster nodes. You may place a service, component, or host object in Maintenance Mode before you perform necessary maintenance or troubleshooting tasks.

Maintenance Mode affects a service, component, or host object in the following two ways:

- Maintenance Mode suppresses alerts, warnings and status change indicators generated for the object
- Maintenance Mode exempts an object from host-level or service-level bulk operations

Explicitly turning on Maintenance Mode for a service implicitly turns on Maintenance Mode for components and hosts that run the service. While Maintenance Mode On prevents bulk operations being performed on the service, component, or host, you may explicitly start and stop a service, component, or host having Maintenance Mode On.

2.3.8.1. Setting Maintenance Mode for Services, Components, and Hosts

For example, examine using Maintenance Mode in a 3-node, Ambari-managed cluster installed using default options. This cluster has one data node, on host c6403. This example describes how to explicitly turn on Maintenance Mode for the HDFS service, alternative procedures for explicitly turning on Maintenance Mode for a host, and the implicit effects of turning on Maintenance Mode for a service, a component and a host.

2.3.8.1.1. How to Turn On Maintenance Mode for a Service

1. Using Services, select HDFS.
2. Select Service Actions, then choose Turn On Maintenance Mode.
3. Choose OK to confirm.

   Notice, on Services Summary that Maintenance Mode turns on for the NameNode and SNameNode components.

2.3.8.1.2. How to Turn On Maintenance Mode for a Host

1. Using Hosts, select c6401.ambari.apache.org.
2. Select Host Actions, then choose Turn On Maintenance Mode.
3. Choose OK to confirm.

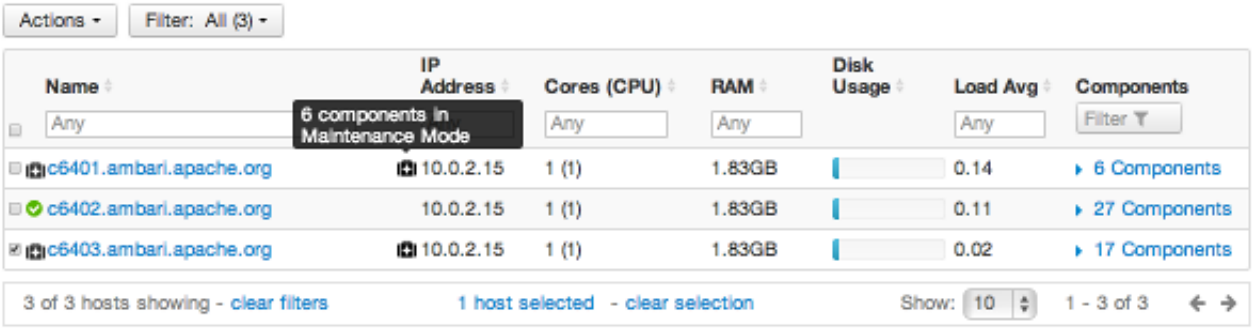
   Notice on Components, that Maintenance Mode turns on for all components.

2.3.8.1.3. How to Turn On Maintenance Mode for a Host (alternative using filtering for hosts)

1. Using Hosts, select c6403.ambari.apache.org.
2. In Actions -> Selected Hosts -> Hosts choose Turn On Maintenance Mode.
3. Choose OK to confirm.

   Notice that Maintenance Mode turns on for host c6403.ambari.apache.org.

Your list of Hosts now shows Maintenance Mode On for hosts c6401 and c6403.



- Hover your cursor over each Maintenance Mode icon appearing in the Hosts list.
  - Notice that hosts c6401 and c6403 have Maintenance Mode On.
  - Notice that on host c6401; Ganglia Monitor, HbaseMaster, HDFS client, NameNode, and Zookeeper Server have Maintenance Mode turned On.
  - Notice on host c6402, that HDFS client and Secondary NameNode have Maintenance Mode On.
  - Notice on host c6403, that 15 components have Maintenance Mode On.
- The following behavior also results:
  - Alerts are suppressed for the DataNode.
  - DataNode is skipped from HDFS Start/Stop/Restart All, Rolling Restart.
  - DataNode is skipped from all Bulk Operations except Turn Maintenance Mode ON/OFF.
  - DataNode is skipped from Start All and / Stop All components.

- DataNode is skipped from a host-level restart/restart all/stop all/start.

2.3.8.1.4. Maintenance Mode Use Cases

Four common Maintenance Mode Use Cases follow:

1. You want to perform hardware, firmware, or OS maintenance on a host.

   You want to:

   - Prevent alerts generated by all components on this host.
   - Be able to stop, start, and restart each component on the host.
   - Prevent host-level or service-level bulk operations from starting, stopping, or restarting components on this host.

   To achieve these goals, turn On Maintenance Mode explicitly for the host. Putting a host in Maintenance Mode implicitly puts all components on that host in Maintenance Mode.

2. You want to test a service configuration change. You will stop, start, and restart the service using a rolling restart to test whether restarting picks up the change.

   You want:

   - No alerts generated by any components in this service.
   - To prevent host-level or service-level bulk operations from starting, stopping, or restarting components in this service.

   To achieve these goals, turn on Maintenance Mode explicitly for the service. Putting a service in Maintenance Mode implicitly turns on Maintenance Mode for all components in the service.

3. You turn off a service completely.

   You want:

   - The service to generate no warnings.
   - To ensure that no components start, stop, or restart due to host-level actions or bulk operations.

   To achieve these goals, turn On Maintenance Mode explicitly for the service. Putting a service in Maintenance Mode implicitly turns on Maintenance Mode for all components in the service.

4. A host component is generating alerts.

   You want to:

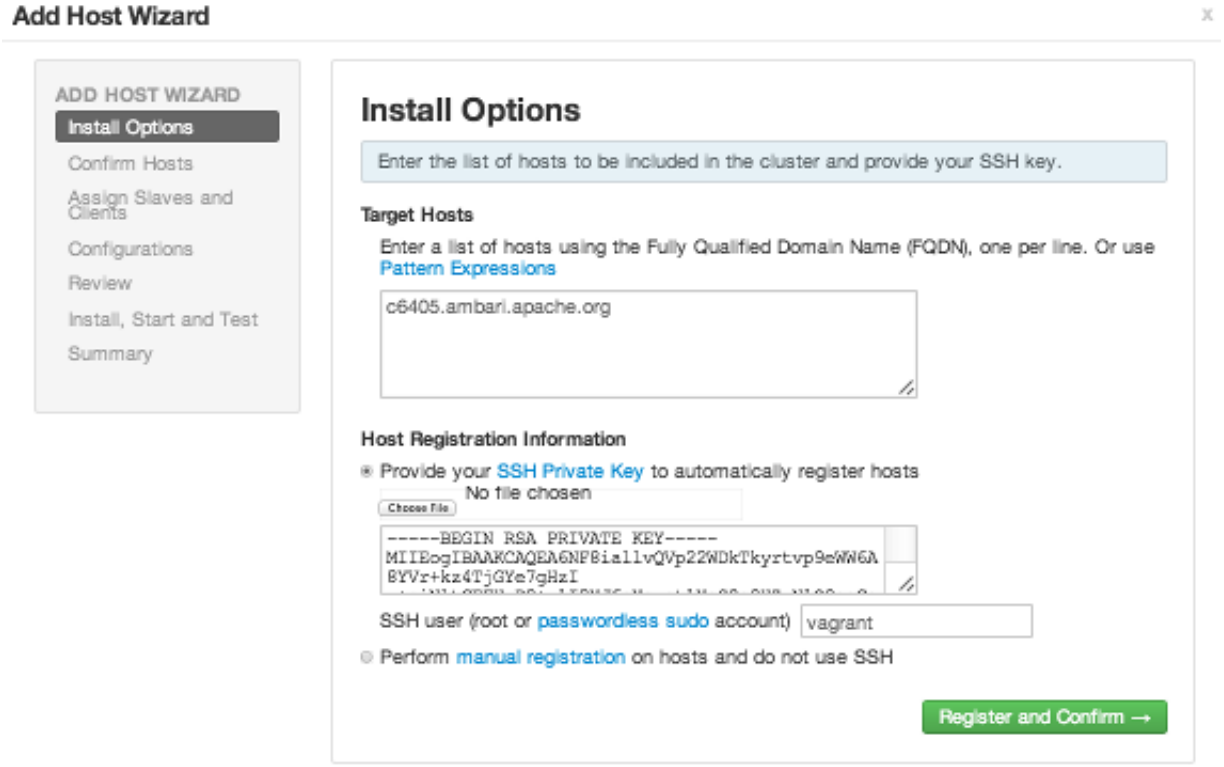
   - Check the component.
   - Assess warnings and alerts generated for the component.

- Prevent alerts generated by the component while you check its condition.

To achieve these goals, turn on Maintenance Mode explicitly for the host component. Putting a host component in Maintenance Mode prevents host-level and service-level bulk operations from starting or restarting the component. You can restart the component explicitly while Maintenance Mode is on.

2.3.9. Adding Hosts to a Cluster

To add new hosts to your cluster, browse to the Hosts page and select **+Add New Hosts**, from the Actions menu. The **Add Host Wizard** provides a sequence of prompts similar to those in the Ambari Install Wizard. Follow the prompts, providing information similar to that provided to define the first set of hosts in your cluster.



To review the Ambari Install Wizard procedure, see Installing, Configuring, and Deploying the Cluster in the Ambari Installation Guide.

2.4. Administering Ambari

Use **Admin** options to manage the following administrative tasks:

- Managing Ambari Web users
- Enabling High Availability of HDP Components
- Enabling Security Configurations
- Checking Stack and Component Versions

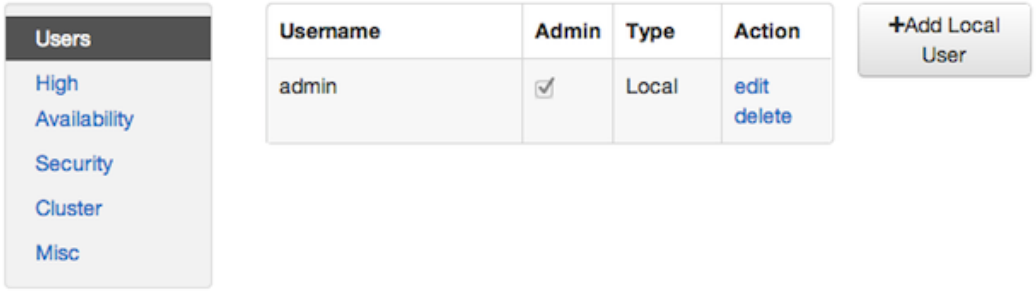
- Checking Service User and Group Accounts
- Accessing Jobs Monitoring Information

in an Ambari-administered, HDP 2.x cluster.

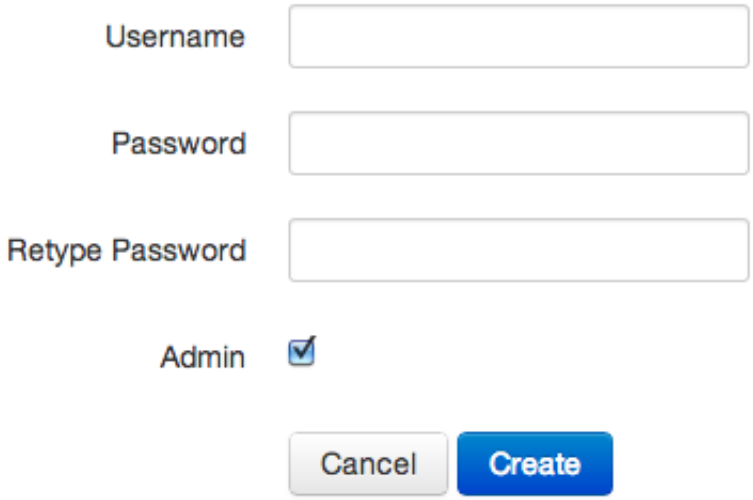
2.4.1. Managing Ambari Web Users

Ambari Web uses a local user store for authentication by default. This section describes managing users in the local user store. Optionally, you can configure Ambari to use LDAP or Active Directory for authentication. For more information, see Setting Up LDAP or Active Directory Authentication.

To manage local users in Ambari Web, select **Admin** > **Users** in the left nav bar. You can add Ambari Web users, grant a user administrator priveleges, delete users, or change the password for a user account. Ambari supports two user roles: User and Admin. A User can view metrics, view service status and configuration, and browse job information. An Admin can do all User tasks and the following ones: start and stop services, modify configurations, and run service checks.



To change the password for a listed user, choose **edit**. Or, to add an Ambari Web user, choose **+Add Local User**. In both cases, a variant of the user name pop-up appears.



To create a new Ambari Web user account, enter a unique **Username** and **Password**. Select the **Admin** checkbox to add administrator privleges to the user account. Choose **Create** to complete the user account creation.

To edit an existing Ambari Web user account, enter the old and new **Passwords**. Make your changes, then choose **Save**.

To delete an existing Ambari Web user account, choose **delete**, then choose Yes to confirm the delete action.

2.4.2. Enabling High Availability of HDP Components

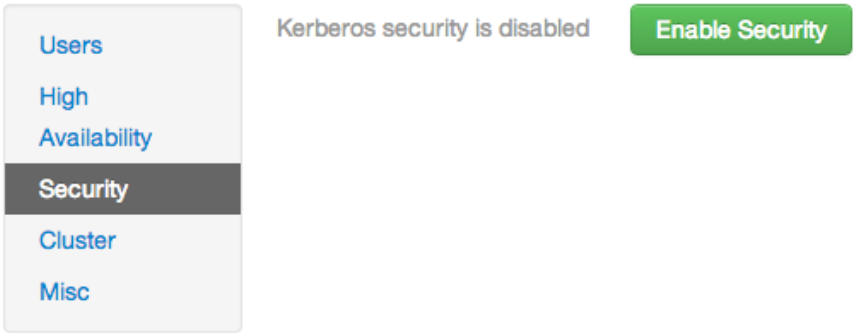
To ensure that a NameNode in your cluster is always available if the primary NameNode hosts fails, enable and set up NameNode High Availability on your cluster using Ambari Web. In **Admin**, select **High Availability**, then choose **Enable NameNode HA**. Follow the steps in the Enable NameNode HA Wizard. For more information about using The Enable NameNode HA Wizard to set up NameNode High Availability, see Setting Up NameNode High Availability.

2.4.3. Enabling Kerberos Security

Ambari supports the Kerberos protocol which allows nodes in your cluster to prove their identites, or authenticate in a secure manner. To enable Kerberos security you must:

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

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



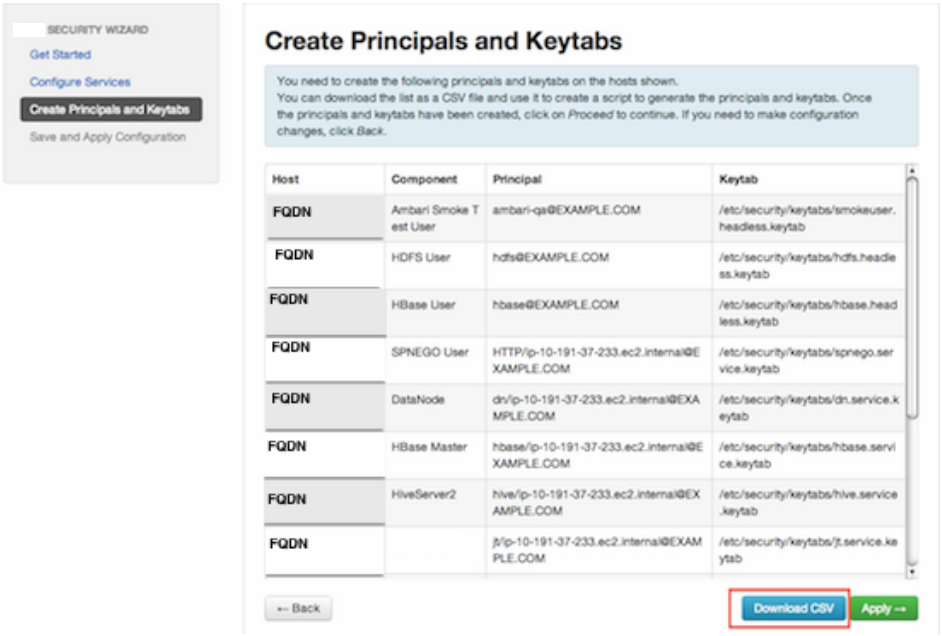
   a. **Get Started**: Read the overview of the procedure necessary to set up Kerberos, supported by the Enable Security Wizard.

   b. **Configure Services**: Prompts you to provide the information required to implement Kerberos for each Hadoop service in your HDP cluster, including principals and paths to keytabs, path to your Kerberos tools, realm names and so on. For more information about a specific field, hover your cursor over the field until a pop-up window that provides 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

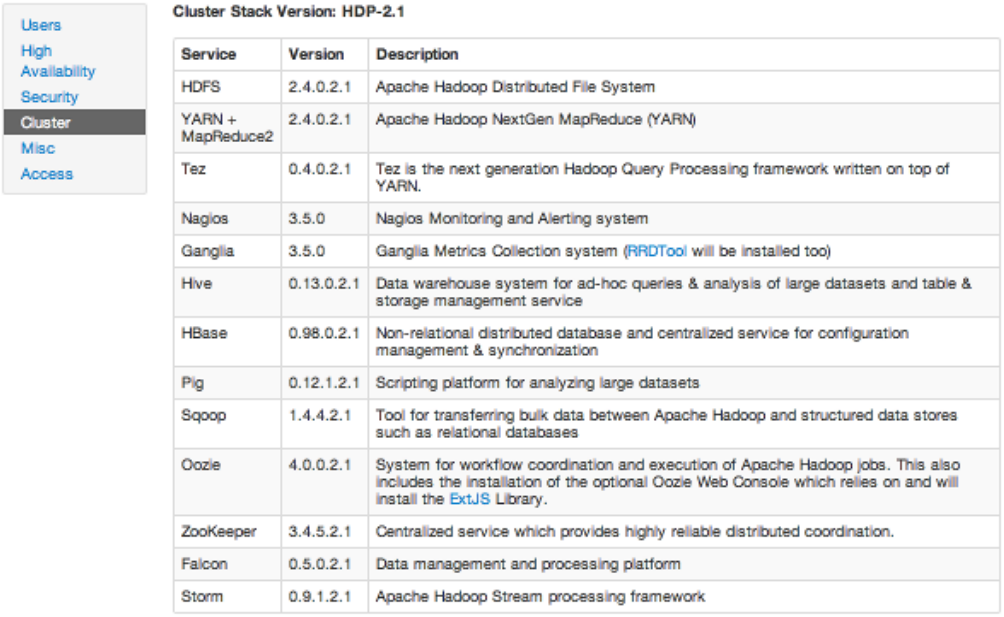
If you have a large cluster, you may want to go to the **Create Principals and Keytabs** step first. Step through the wizard accepting the defaults to get to the appropriate page. On the page, use the **Download CSV** button to get a list of all the necessary principals and keytabs in CSV form, which 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 file name.



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

2.4.4. 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, choose **Cluster**.



2.4.5. Managing Stack Repositories

Ambari supports managing the repositories from which you deploy the HDP Stack in your Hadoop cluster. You can view and edit the Base URL at which HDP software source bits are stored.

- Viewing a Stack Repository URL
- Modifying a Stack Repository URL

2.4.5.1. Viewing a Stack Repository URL

To view the locations for your HDP repositories using Ambari Web, select **Admin**, then choose **Cluster**. Repositories lists the Base URL for each repository, sorted by OS family, as shown in the following example:

2.1

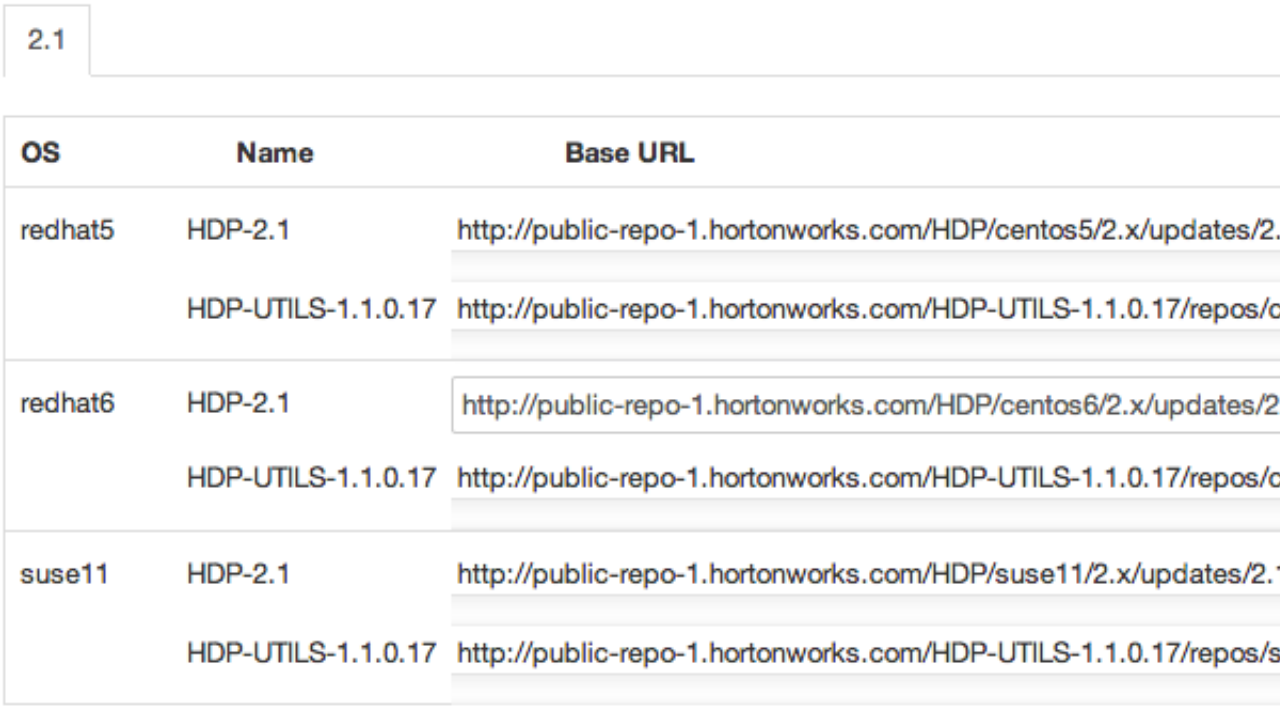
OS	Name	Base URL
redhat5	HDP-2.1	http://public-repo-1.hortonworks.com/HDP/centos5/2.x/updates/2.1.
	HDP-UTILS-1.1.0.17	http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.17/repos/cen
redhat6	HDP-2.1	http://public-repo-1.hortonworks.com/HDP/centos6/2.x/updates/2.1.
	HDP-UTILS-1.1.0.17	http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.17/repos/cen
suse11	HDP-2.1	http://public-repo-1.hortonworks.com/HDP/suse11/2.x/updates/2.1.3
	HDP-UTILS-1.1.0.17	http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.17/repos/sus

2.4.5.2. Modifying Stack Repository URL

To modify the Base URL for your HDP Repository using Ambari Web, select **Admin**, then choose **Cluster**.

1. For a specific URL in Repositories, choose Edit.

2. Make changes to the path shown in Base URL. For example, change the URL for redhat6 to http://public-repo-1.hortonworks.com/HDP/centos6/2.x/updates/2.1.2.0, as shown in the following example:



2.1

OS	Name	Base URL
redhat5	HDP-2.1	http://public-repo-1.hortonworks.com/HDP/centos5/2.x/updates/2.
	HDP-UTILS-1.1.0.17	http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.17/repos/c
redhat6	HDP-2.1	http://public-repo-1.hortonworks.com/HDP/centos6/2.x/updates/2
	HDP-UTILS-1.1.0.17	http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.17/repos/c
suse11	HDP-2.1	http://public-repo-1.hortonworks.com/HDP/suse11/2.x/updates/2.
	HDP-UTILS-1.1.0.17	http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.17/repos/s

3. Choose Save.

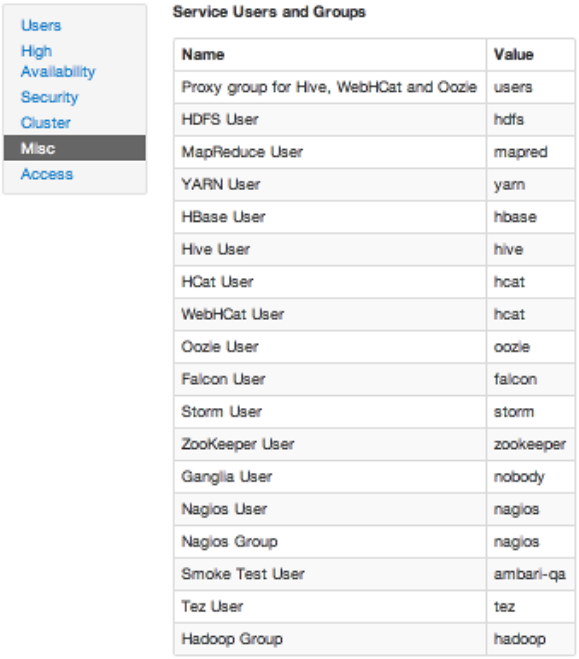
   Ambari performs a validation check for each Base URL. The check confirms that Ambari can reach the URL and the URL points to a valid software repository. A valid repository finds the metadata associated with the repository.

4. If the validation check succeeds, the Base URLs are saved. If validation fails, a validation error message displays. Choose one of the following options:

   - Revert to discard changes
   - OK to save a modified, valid Base URL
   - Save Anyway to save a modified URL that fails to validate

2.4.6. 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.

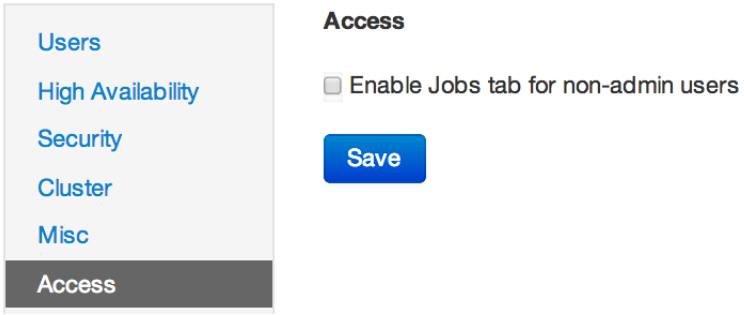


Name	Value
Proxy group for Hive, WebHCat and Oozie	users
HDFS User	hdfs
MapReduce User	mapred
YARN User	yarn
HBase User	hbase
Hive User	hive
HCat User	hcat
WebHCat User	hcat
Oozie User	oozie
Falcon User	falcon
Storm User	storm
ZooKeeper User	zookeeper
Ganglia User	nobody
Nagios User	nagios
Nagios Group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

2.4.7. Accessing Jobs Monitoring Information

Ambari enables the Jobs tab to display for non-administrator users by default. In HDP 2.1 and higher, Jobs displays status information about Hive query jobs throughout the cluster. This information provides useful metrics for administering Hive, but is not useful to most non-administrator users. In a production environment, you may disable access to Jobs for non-administrator users. You may also choose to prevent Hive and Tez from writing event metrics to the Application Timeline Server, by disabling the `yarn.timeline-service`.

To manage access that your non-administrator users have to Jobs information, choose **Access**.



2.4.7.1. Disabling the Jobs View

To prevent non-administrator users from viewing information about Hive queries, disable the Jobs view.

1. Select **Admin > Access** .
2. Clear the `Enable Jobs tab for non-Admin users` checkbox.

3. Choose Save.

2.4.7.2. Disabling the YARN Timeline Service

To prevent Hive and Tez from writing event metrics to the Application Timeline Server, disable the Yarn Timeline Service.

1. Select **Services > YARN > Configs** .
2. Expand `Application Timeline Server`.
3. Clear the `yarn.timeline-service.enabled` checkbox.
4. Restart the YARN service.

3. 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 plug-ins specially designed for monitoring important aspects of your Hadoop cluster, based on your Stack selection.

3.1. Basic Nagios Architecture

Using the open source monitoring system Nagios, Ambari gathers information on the status of both of the hosts and the services that run on them.

- **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.
- **Alert Information**: Using Nagios with Hadoop-specific plug-ins 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. For more details on Nagios architecture, see the Nagios Overview at the `nagios.org` web site.

3.2. Installing Nagios

The Ambari Installation Wizard gives you the option of installing and configuring Nagios, including the out-of-the-box plug-ins for Hadoop-specific alerts. The Nagios server, Nagios plug-ins, and the web-based user interface are installed on the Nagios server host, as specified during the installation procedure.

3.3. 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 plug-ins are stored in the Nagios plug-ins directory,`/usr/lib64/nagios/plug-ins/`.

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.

3.4. 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.

3.5. 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.

3.5.1. HDFS Service Alerts

These alerts are used to monitor the HDFS service.

3.5.1.1. 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` plug-in.

3.5.1.1.1. 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

3.5.1.1.2. 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

3.5.1.2. 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`

3.5.1.2.1. 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.

3.5.1.2.2. 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

3.5.1.3. DataNode space

This host-level alert is triggered if storage capacity if full on the DataNode (90% critical). It uses the `check_datanode_storage.php` plug-in which checks the DataNode JMX Servlet for the **Capacity** and **Remaining** properties.

3.5.1.3.1. Potential causes

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

3.5.1.3.2. 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

3.5.1.4. 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.

3.5.1.4.1. 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

3.5.1.4.2. 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

3.5.1.5. NameNode host CPU utilization

This host-level alert is triggered if CPU utilization of the NameNode exceeds certain thresholds (200% warning, 250% critical). It uses the `check_cpu.php` plug-in which checks the NameNode JMX Servlet for the **SystemCPULoad** property. This information is only available if you are running JDK 1.7.

3.5.1.5.1. 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.

3.5.1.5.2. Possible remedies

- Use the `top` command to determine which processes are consuming excess CPU.
- Reset the offending process

3.5.1.6. NameNode edit logs directory status

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

3.5.1.6.1. 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

3.5.1.6.2. 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.

3.5.1.7. NameNode Web UI

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

3.5.1.7.1. Potential causes

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

3.5.1.7.2. 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.

3.5.1.8. Percent DataNodes with space available

This service-level alert is triggered if the storage if full on a certain percentage of DataNodes (10% warn, 30% critical). It uses the `check_aggregate.php` plug-in which aggregates the result from the `check_datanode_storage.php` plug-in.

3.5.1.8.1. Potential causes

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

3.5.1.8.2. 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

3.5.1.9. 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` plug-in to aggregate the results of `Data node process` checks.

3.5.1.9.1. 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

3.5.1.9.2. 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.

3.5.1.10. 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` plug-in.

3.5.1.10.1. Potential causes

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

3.5.1.10.2. Possible remedies

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

3.5.1.11. HDFS capacity utilization

This service-level alert is triggered if the HDFS capacity utilization exceeds the configured critical threshold (80% warn, 90% critical). It uses the `check_hdfs_capacity.php` plug-in which checks the **NameNode JMX Servlet** for the **CapacityUsed** and **CapacityRemaining** properties.

3.5.1.11.1. Potential causes

- Cluster storage is full

3.5.1.11.2. 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.

3.5.2. NameNode HA Alerts (Hadoop 2 only)

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

3.5.2.1. 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` plug-in.

3.5.2.1.1. 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.

3.5.2.1.2. Possible remedies

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

3.5.2.2. NameNode HA Healthy process

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

3.5.2.2.1. Potential causes

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

3.5.2.2.2. 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.

3.5.3. YARN Alerts (Hadoop 2 only)

These alerts are used to monitor YARN.

3.5.3.1. 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` plug-in.

3.5.3.1.1. 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

3.5.3.1.2. 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.

3.5.3.2. 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` plug-in to aggregate the results of DataNode process alert checks.

3.5.3.2.1. 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.

3.5.3.2.2. 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.

3.5.3.3. ResourceManager Web UI

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

3.5.3.3.1. Potential causes

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

3.5.3.3.2. 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.

3.5.3.4. 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` plug-in.

3.5.3.4.1. Potential causes

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

3.5.3.4.2. Possible remedies

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

3.5.3.5. ResourceManager CPU utilization

This host-level alert is triggered if CPU utilization of the ResourceManager exceeds certain thresholds (200% warning, 250% critical). It uses the `check_cpu.php` plug-in which checks the ResourceManager JMX Servlet for the **SystemCPULoad** property. This information is only available if you are running JDK 1.7.

3.5.3.5.1. 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.

3.5.3.5.2. Possible remedies

- Use the `top` command to determine which processes are consuming excess CPU.
- Reset the offending process.

3.5.3.6. 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` plug-in.

3.5.3.6.1. 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

3.5.3.6.2. 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.

3.5.3.7. NodeManager health

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

3.5.3.7.1. Potential causes

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

3.5.3.7.2. 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 ResourceManager UI logs (`/var/log/hadoop/yarn`) for health check errors.

3.5.4. MapReduce2 Alerts (Hadoop 2 only)

These alerts are used to monitor MR2.

3.5.4.1. HistoryServer Web UI

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

3.5.4.1.1. Potential causes

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

3.5.4.1.2. 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.

3.5.4.2. 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` plug-in.

3.5.4.2.1. Potential causes

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

3.5.4.2.2. Possible remedies

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

3.5.4.3. 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.

3.5.4.3.1. 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

3.5.4.3.2. 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.

3.5.4.4. 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` plug-in.

3.5.4.4.1. 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,

3.5.4.4.2. 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.

3.5.5. MapReduce Service Alerts (Hadoop 1 only)

These alerts are used to monitor the MapReduce service.

3.5.5.1. 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` plug-in.

3.5.5.1.1. Potential causes

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

3.5.5.1.2. Possible remedies

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

3.5.5.2. 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` plug-in.

3.5.5.2.1. 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

3.5.5.2.2. 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.

3.5.5.3. JobTracker Web UI

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

3.5.5.3.1. Potential causes

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

3.5.5.3.2. 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.

3.5.5.4. JobTracker CPU utilization

This host-level alert is triggered if CPU utilization of the JobTracker exceeds certain thresholds (200% warning, 250% critical). It uses the `check_cpu.php` plug-in which checks the JobTracker JMX Servlet for the **SystemCPULoad** property. This information is only available if you are running JDK 1.7.

3.5.5.4.1. 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.

3.5.5.4.2. Possible remedies

- Use the `top` command to determine which processes are consuming excess CPU
- Reset the offending processor.

3.5.5.5. HistoryServer Web UI

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

3.5.5.5.1. 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.

3.5.5.5.2. 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.

3.5.5.6. 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` plug-in.

3.5.5.6.1. 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.

3.5.5.6.2. 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.

3.5.6. HBase Service Alerts

These alerts are used to monitor the HBase service.

3.5.6.1. 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` plug-in to aggregate the results of `RegionServer process down` checks.

3.5.6.1.1. 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

3.5.6.1.2. 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

3.5.6.2. 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` plug-in.

3.5.6.2.1. 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

3.5.6.2.2. 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

3.5.6.3. HBase Master Web UI

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

3.5.6.3.1. Potential causes

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

3.5.6.3.2. 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.

3.5.6.4. HBase Master CPU utilization

This host-level alert is triggered if CPU utilization of the HBase Master exceeds certain thresholds (200% warning, 250% critical). It uses the `check_cpu.php` plug-in which checks the HBase Master JMX Servlet for the **SystemCPULoad** property. This information is only available if you are running JDK 1.7.

3.5.6.4.1. 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.

3.5.6.4.2. Possible remedies

- Use the `top` command to determine which processes are consuming excess CPU
- Reset the offending process.

3.5.6.5. 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` plug-in.

3.5.6.5.1. 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.

3.5.6.5.2. 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.

3.5.7. Hive Alerts

These alerts are used to monitor the Hive service.

3.5.7.1. 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` plug-in.

3.5.7.1.1. 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.

3.5.7.1.2. 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.

3.5.8. WebHCat Alerts

These alerts are used to monitor the WebHCat service.

3.5.8.1. WebHCat Server status

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

3.5.8.1.1. Potential causes

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

3.5.8.1.2. Possible remedies

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

3.5.9. Oozie Alerts

These alerts are used to monitor the Oozie service.

3.5.9.1. Oozie status

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

3.5.9.1.1. Potential causes

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

3.5.9.1.2. Possible remedies

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

3.5.10. Ganglia Alerts

These alerts are used to monitor the Ganglia service.

3.5.10.1. 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` plug-in.

3.5.10.1.1. 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

3.5.10.1.2. 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.

3.5.10.2. 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` plug-in.

Ganglia Monitoring daemons run for the following collections:

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

3.5.10.2.1. Potential causes

- A `gmond` process is down.
- A `gmond` process is hanging.

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

3.5.10.2.2. Possible remedies

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

3.5.11. Nagios Alerts

These alerts are used to monitor the Nagios service.

3.5.11.1. 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.

3.5.11.1.1. 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.

3.5.11.1.2. Possible remedies

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

3.5.12. ZooKeeper Alerts

These alerts are used to monitor the Zookeeper service.

3.5.12.1. 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` plug-in to aggregate the results of `Zookeeper process` checks.

3.5.12.1.1. Potential causes

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

3.5.12.1.2. 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.

3.5.12.2. 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` plug-in.

3.5.12.2.1. 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.

3.5.12.2.2. 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.

3.5.13. Ambari Alerts

This alert is used to monitor the Ambari Agent service.

3.5.13.1. 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` plug-in.

3.5.13.1.1. 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.

3.5.13.1.2. 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.