

Hortonworks Data Platform

Administering Hadoop with Ambari Web

(Aug 19, 2013)

Hortonworks Data Platform : Administering Hadoop with Ambari Web

Copyright © 2012, 2013 Hortonworks, Inc. Some rights reserved.

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

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

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



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

Hortonworks Data Platform (HDP) and any of its components are not anticipated to be combined with any hardware, software or data, except as expressly recommended in this documentation.

Table of Contents

1. Introducing Ambari Web	1
1.1. Cluster Monitoring Sources	1
1.2. Architecture	1
1.2.1. Sessions	2
1.3. Starting and Accessing Ambari Web	3
2. Navigating Ambari Web	4
2.1. The Navigation Header	4
2.2. The Dashboard View	4
2.2.1. The Widget Version	4
2.2.2. The Classic Version	8
2.3. The Heatmaps View	10
2.4. The Services View	12
2.4.1. Services Navigation	13
2.4.2. Services Summary	13
2.4.3. Configs	13
2.4.4. Quick Links	14
2.4.5. Alerts and Health Checks	14
2.4.6. Management Header	14
2.4.7. Metrics	15
2.5. The Hosts View	15
2.5.1. Host Details	16
2.5.2. Add Hosts	17
2.6. The Jobs View	18
2.6.1. Browsing Jobs	18
2.6.2. Using Job Charts	18
2.7. Admin View	20
2.7.1. Managing Ambari Web Users	20
2.7.2. Enabling Kerberos Security	21
2.7.3. Checking Stack and Component Versions	23
2.7.4. Checking Users and Groups	23
3. Using Nagios With Hadoop	25
3.1. Basic Nagios Architecture	25
3.2. Installing Nagios	25
3.3. File Locations	26
3.4. Configuring Nagios Alerts For Hadoop Services	26
3.5. Nagios Alerts For Hadoop Services	27
3.5.1. Host Alerts	27
3.5.2. System Alerts for Master Daemon Nodes	27
3.5.3. HDFS Service Alerts	28
3.5.4. MapReduce Service Alerts	32
3.5.5. HBase Service Alerts	33
3.5.6. Hive Metastore Service Alerts	35
3.5.7. ZooKeeper Service Alerts	36
3.5.8. Oozie Service Alerts	36
3.5.9. WebHCat Service Alerts	37
3.5.10. Nagios and Ganglia Server Alerts	37
3.6. Configuring New Alerts For Hadoop	38

List of Figures

1.1. Architectural Overview 2

List of Tables

- 2.1. Service Status 5
- 2.2. Widget Interactions 6
- 2.3. Widget Interactions 2 6
- 2.4. Widget Interactions 3 7
- 2.5. Service Status 8
- 3.1. Define Command 39
- 3.2. Define Service 40

1. Introducing Ambari Web

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



Note

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

1.1. Cluster Monitoring Sources

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

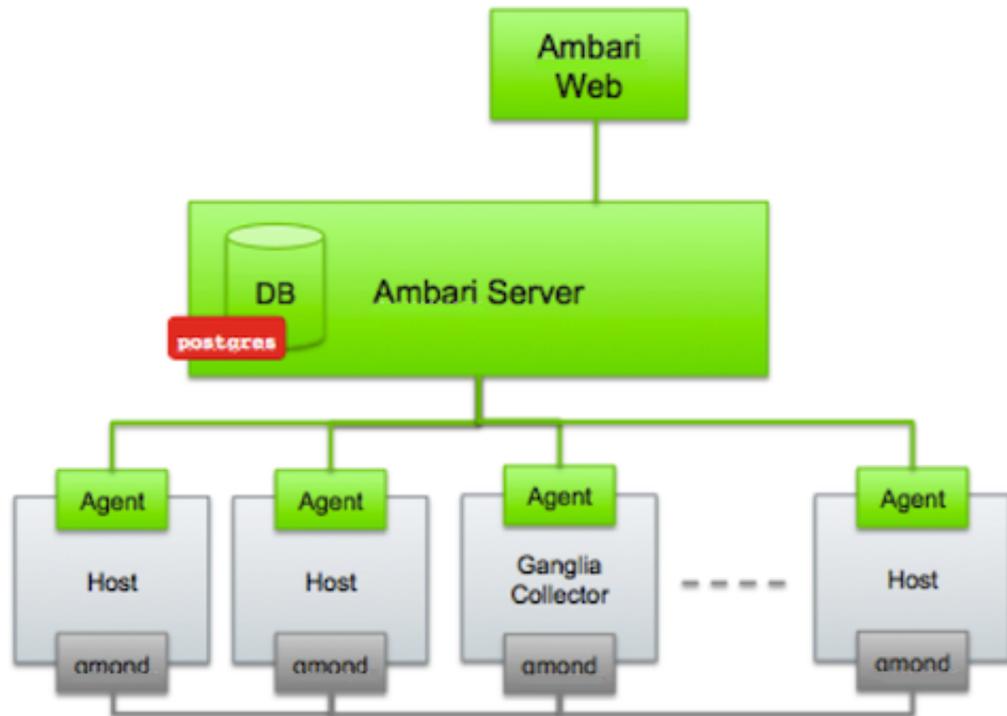
- **Host and System Information:** Ambari monitors basic host and system information such as CPU utilization, disk I/O bandwidth and operations per second, average memory and swap space utilization, and average network latency.
- **Service Information:** Ambari monitors the health and performance status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

1.2. Architecture

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

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

Figure 1.1. Architectural Overview



1.2.1. Sessions

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



Note

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



1.3. Starting and Accessing Ambari Web

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

```
ambari-server start
```

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

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

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

2. Navigating Ambari Web

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

2.1. The Navigation Header

At the top of every screen is the Navigation Header.



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

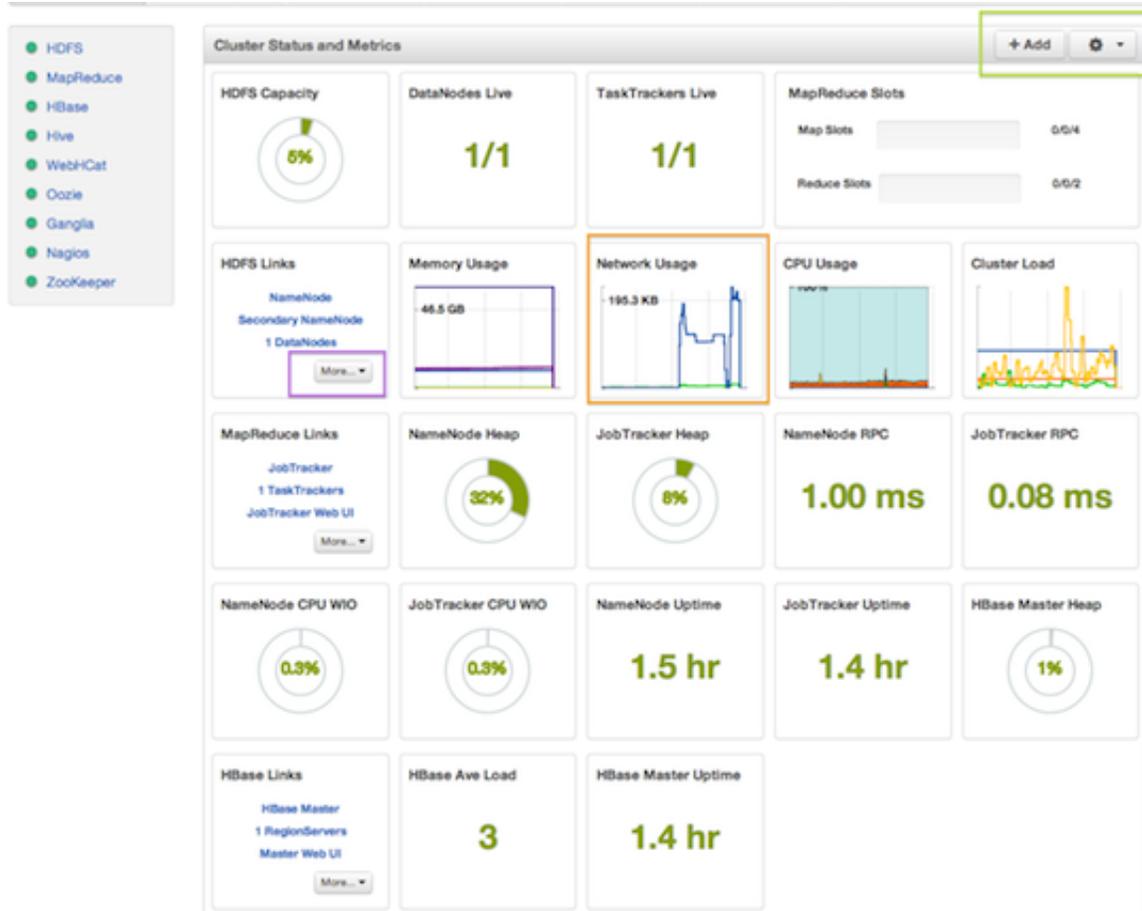
2.2. The Dashboard View

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

- The [Widget](#) version
- The [Classic](#) version

2.2.1. The Widget Version

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



The Widget version is divided into two main sections:

- [Services Summary](#)
- [Cluster Status and Metrics](#)

2.2.1.1. Services Summary

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

Table 2.1. Service Status

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

Color	Name	Status
	Blinking Red	Stopping

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

2.2.1.2. Cluster Status and Metrics

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

Table 2.2. Widget Interactions

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

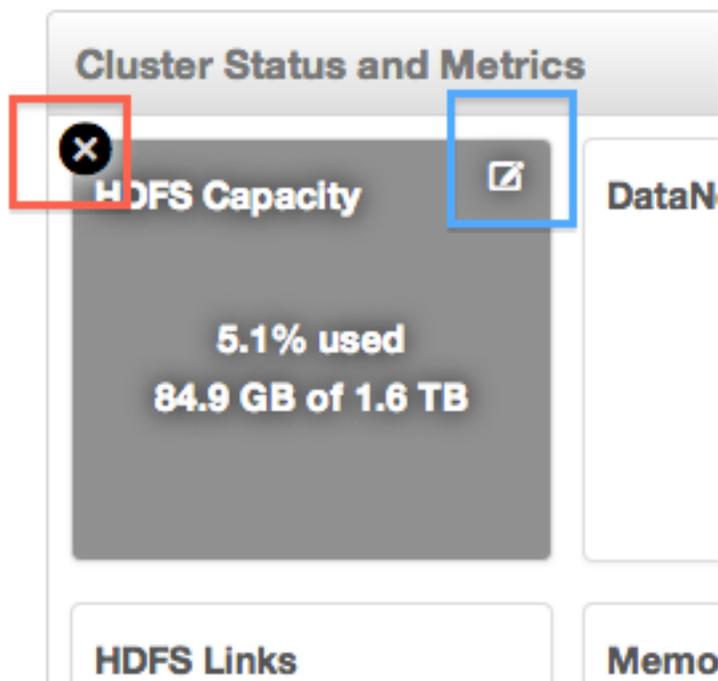
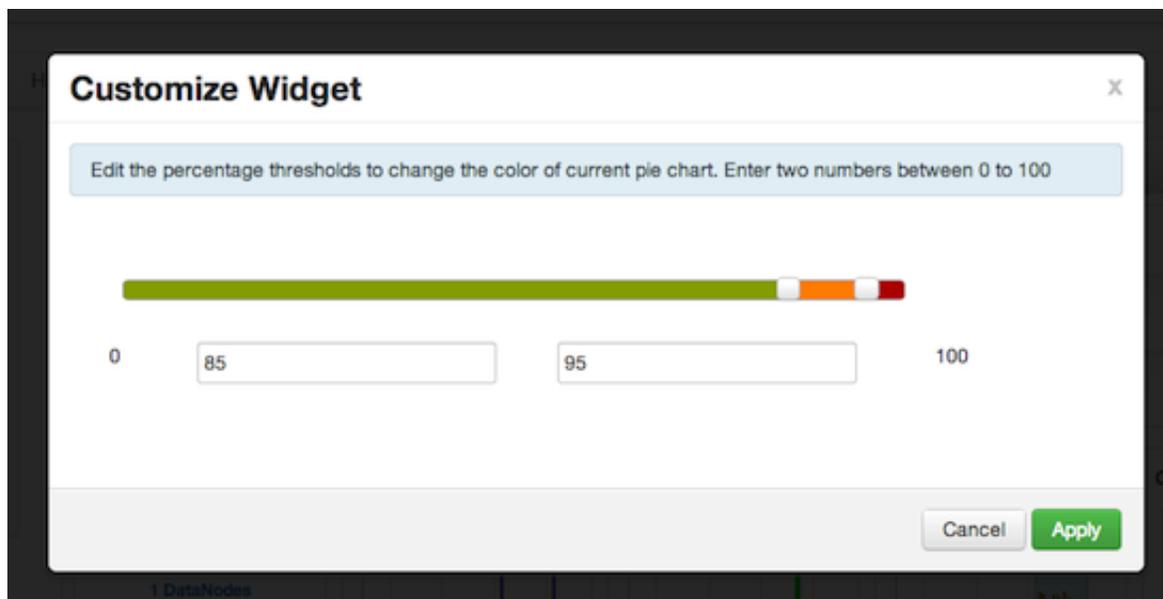


Table 2.3. Widget Interactions 2

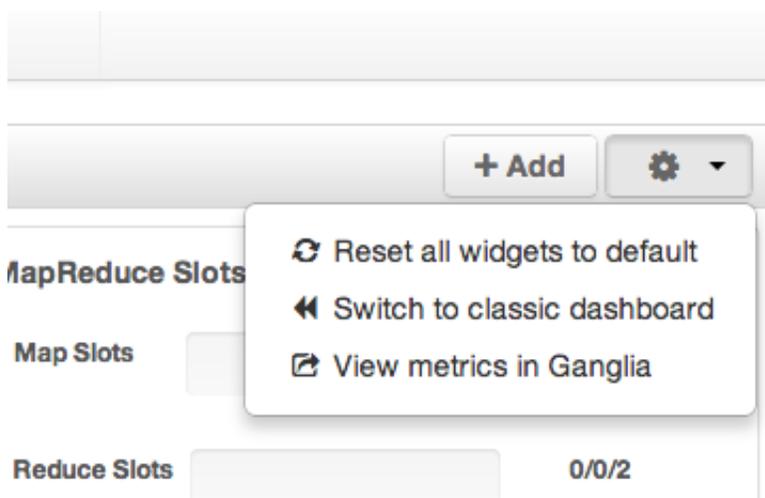
To:	Do:
Delete a widget	Click the X, marked in red
Edit a widget	Click the small edit icon, marked in blue. The Customize Widget popup appears.



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

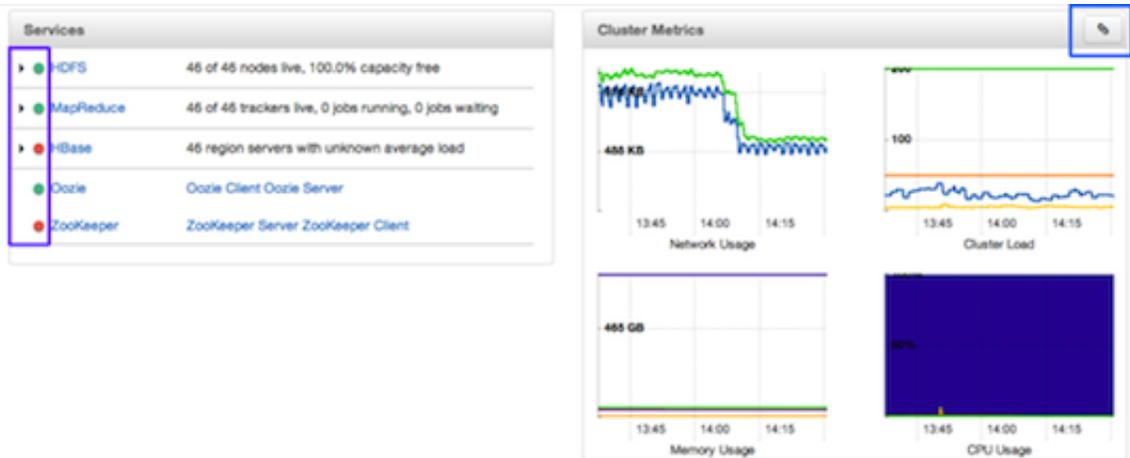
Table 2.4. Widget Interactions 3

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



2.2.2. The Classic Version

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



The Classic version is also divided into two main sections:

- [Services Summary](#)
- [Cluster Metrics](#)

2.2.2.1. Services Summary

On the left side of the screen is the **Services** summary. Use this section to get an overall view of the status of your services. In the image above [8], notice that **HDFS**, **MapReduce**, and **Oozie** have green dots next to them but that **HBase** and **ZooKeeper** have red dots. The dot colors give you a quick overview of the service's status.

Table 2.5. Service Status

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

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

Services

▼ ● **HDFS** 1 of 1 nodes live, 5.1% capacity used

NameNode	● Started	View Host
SNameNode	● Started	View Host
DataNodes	1/1 DataNodes Live	View Host

NameNode Uptime: 36.43 mins

NameNode Heap: 40.9 MB / 957.5 MB (4.3% used)

DataNodes Status: 1 live / 0 dead / 0 decommissioning

HDFS Disk Capacity: 84.9 GB / 1.6 TB (5.1% used)

Blocks (total): 209

Block Errors: 0 corrupt / 0 missing / 209 under replicated

Total Files + Directories: 335

Upgrade Status: No pending upgrade

Safe Mode Status: Not in safe mode



Capacity (Used/Total)

[Quick Links](#) ▼

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

Services

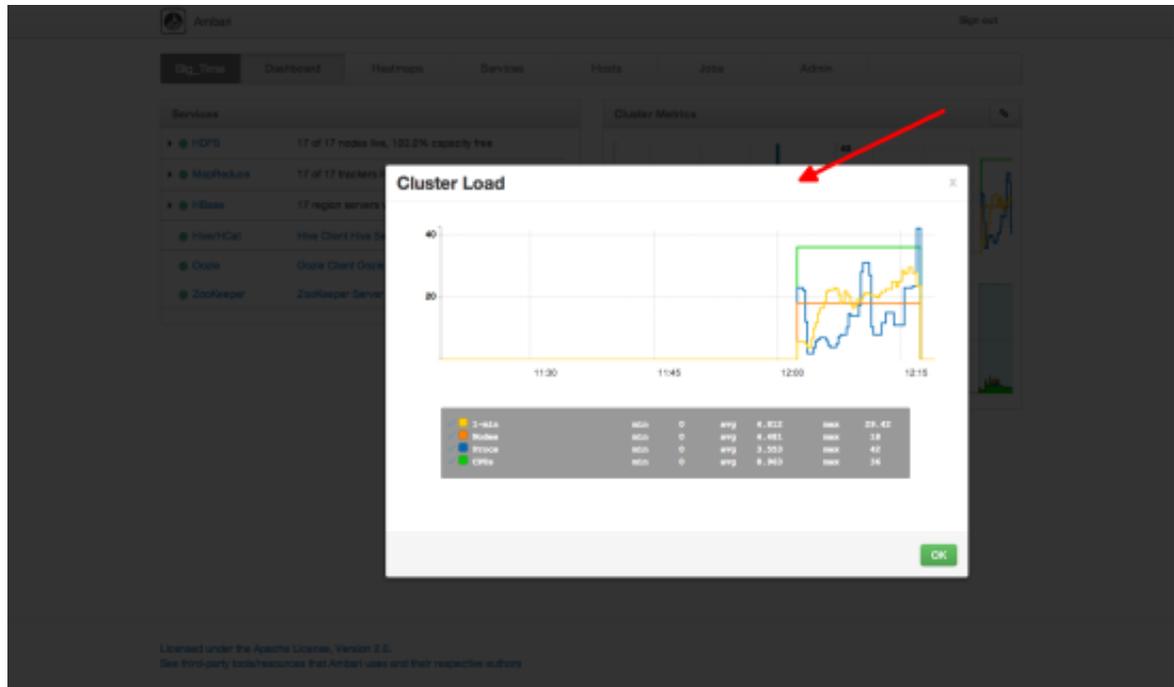
- 1
HDFS
1 of 2 nodes live, 100.0% capacity free
- MapReduce
2 of 2 trackers live, 0 jobs running, 0 jobs waiting
- 3
HBase
2 region servers with unknown average load
- Oozie
Oozie Client Oozie Server
- 1
ZooKeeper
ZooKeeper Server ZooKeeper Client

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

2.2.2.2. Cluster Metrics

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

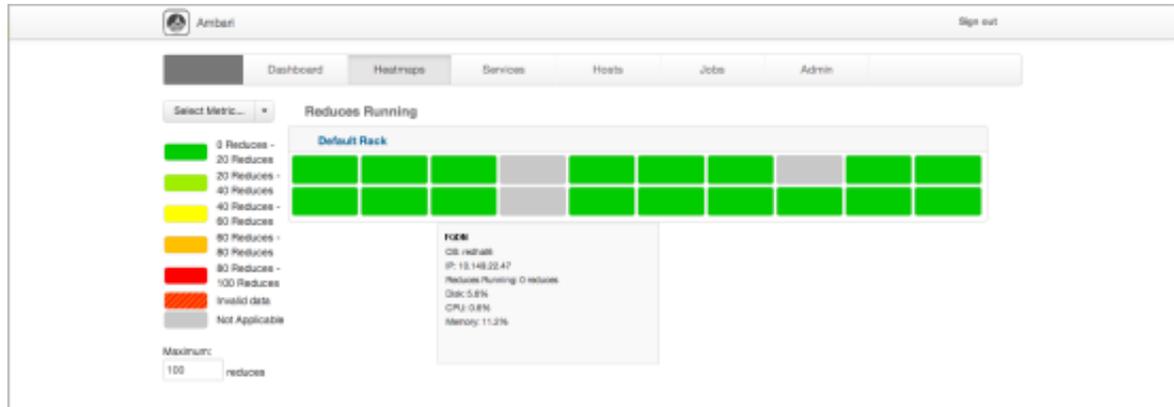
To see a larger view of the chart, click on it. The larger chart pops out.



Notice the link symbol on the upper right side of the Cluster Metric section, outlined in blue in the [overview screenshot \[8\]](#) above. This is a link to the GUI for the Ganglia Server itself, where you can find much more detailed information on your cluster. You can also use this to switch back to the Widget version.

2.3. The Heatmaps View

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



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

Select Metric... **CPU Wait I/O %**

Host

HDFS

MapReduce

HBase

RegionServer read request count

RegionServer write request count

RegionServer compaction queue size

RegionServer regions

RegionServer memstore sizes

80% - 100%

Invalid data

Not Applicable

Maximum: 100 %

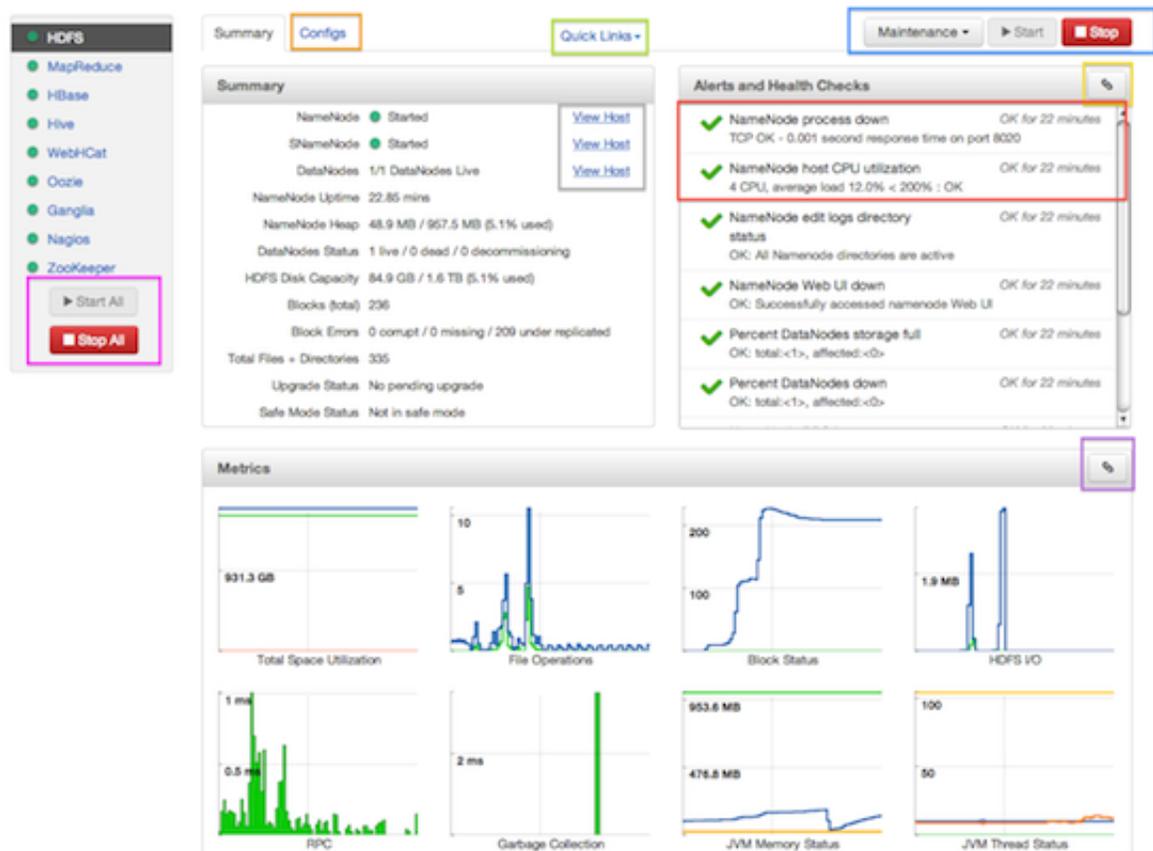
Currently the following metrics are supported:

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

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

2.4. The Services View

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



The Services view is divided into seven sections:

- [Services Navigation](#)
- [Services Summary](#)
- [Configuration Update](#)
- [Quick Links](#)
- [Alerts and Health Checks](#)
- [Management Header](#)
- [Metrics](#)

2.4.1. Services Navigation

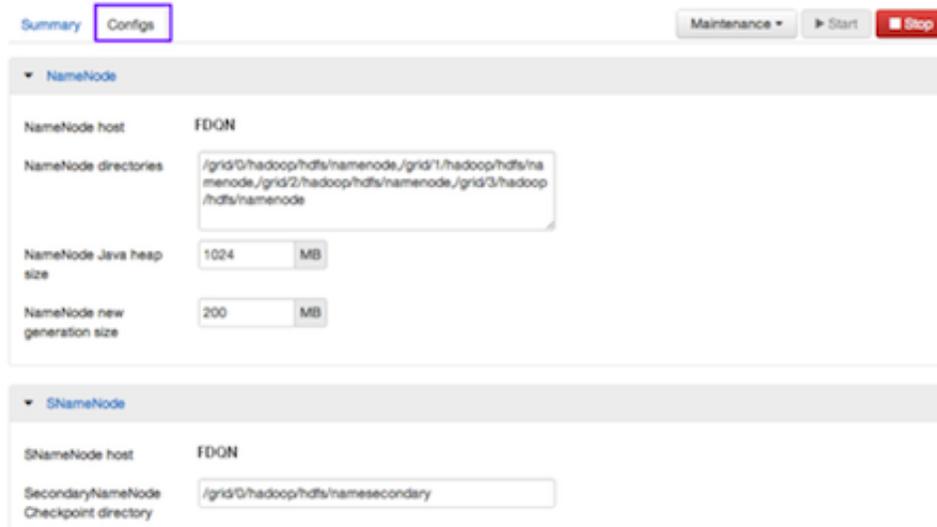
The Services navigation panel on the left gives you a quick look at the status of your services. The [color of the dot](#) next to the left of service name tells you the service running state and a small rectangle to the right lets you know if there are any alerts assigned to it. To move the larger screen display from showing information about one service to another, click the service name. To stop or start all of the services at once, use the **Start All** or **Stop All** buttons, outlined in [magenta \[12\]](#).

2.4.2. Services Summary

The Services **Summary** tab displays basic information about the selected service. For services that run as master/slave sets, this is the same panel that is displayed on the **Classic Dashboard** when you click the triangle next to the service name. Use the **View Host** links, marked in [gray \[12\]](#) above to move to the **Host Details** view of the host that is running the service. Clicking **Configs**, outlined in [orange \[12\]](#) above, opens a second tab, the Configuration Update tab.

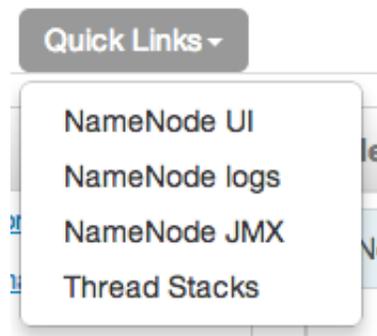
2.4.3. Configs

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



2.4.4. Quick Links

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

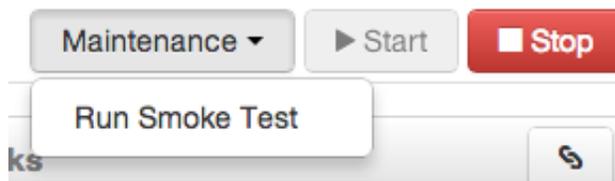


2.4.5. Alerts and Health Checks

The **Alerts and Health Checks** panel displays the results of the health checks performed on your cluster by Nagios. Alerts display a brief summary of the issue and its rating. They are sorted by descending severity, followed by descending time. To access more detailed information, click the link in the upper right corner of the panel, outlined in yellow [12] above. This opens the native Nagios GUI. Use the username and password you set up during installation to log in.

2.4.6. Management Header

The management header, outlined in blue [12], gives you a convenient way to stop and start an individual service. You can also use **Maintenance** to perform smoke tests on the service, for example, if you have updated the service's configuration.



2.4.7. Metrics

The **Metrics** panel displays a set of charts measuring common metrics for the service. Hover your mouse above a particular chart to see a legend and click a specific metric type to select or deselect it. Click the chart to see a larger version. To get more information, click the link in the upper right corner of the panel, outlined in purple [12]. This opens the native Ganglia GUI.

2.5. The Hosts View

The Hosts view presents your cluster in terms of the hosts on which the services are running. If your cluster is large, you can use the health filter, outlined in red, to locate only hosts in that state. The more general filter tool, shown outlined in blue, allows you to select specific types of hosts and to sort your result.

 A screenshot of the Hosts View interface. At the top, there is a health filter bar with a red outline containing: 'All (5) | Healthy (5) | Master Down (0) | Slave Down (0) | No Heartbeat (0) | Alerts (2)'. To the right is a '+ Add New Hosts' button with a green outline. Below is a table with columns: Name, IP Address, CPU, RAM, Disk Usage, Load Avg, and Components. The table has a blue outline around the filter area. The table contains five rows of host data. The second row has a red square icon next to its FQDN. The third row has a red square icon next to its FQDN. At the bottom right, there is a 'Show: 10' dropdown and '1 - 5 of 5' with navigation arrows.

Name	IP Address	CPU	RAM	Disk Usage	Load Avg	Components
FQDN	10.110.54.198	2	7.29GB		0.19	DN, GM and 2 more
FQDN	10.4.103.222	2	7.29GB		0.03	GM, ZKS
FQDN	10.108.18.182	2	7.29GB		1.86	GM, HDFSC and 11 more
FQDN	10.46.157.174	2	7.29GB		1.20	GM, GS and 9 more
FQDN	10.2.43.56	2	7.29GB		1.19	DN, GM and 11 more

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

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

- Red - At least one master component on that host is down. Hover to see a tooltip with a list of affected components.
- Orange - At least one slave component on that host is down. Hover to see a tooltip with a list of affected components.

- Yellow - Ambari Server has not received a heartbeat from that host for more than 3 minutes.
- Green - Normal running state

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

2.5.1. Host Details

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

The screenshot displays the Ambari interface for a host named 'fdqn'. The top navigation bar includes 'Dashboard', 'Heatmaps', 'Services', 'Hosts', 'Jobs', and 'Admin'. The 'Hosts' tab is active, showing a list of hosts with 'fdqn' selected. Below the navigation, there is a 'Summary' section with the following details:

- Hostname: fdqn
- IP Address: 10.143.136.78
- OS: redhat5 (x86_64)
- CPU: 1
- Disk: 5.78%
- Memory: 15.32GB
- Load Avg: 0.05
- Agent: less than a minute ago
- Heartbeat:

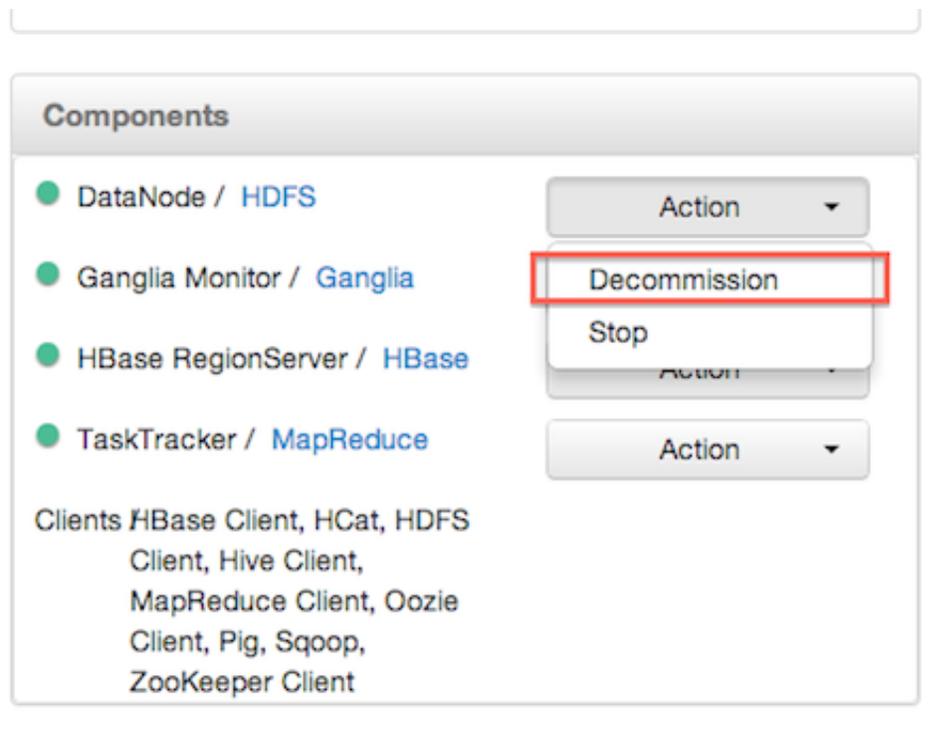
The 'Components' section lists several services, each with a green status indicator and an 'Action' dropdown menu. These components are:

- Hive Metastore / Hive/HCat
- WebHCat Server / WebHCat
- HBase Master / HBase
- ZooKeeper Server / ZooKeeper
- MySQL Server / Hive/HCat
- Oozie Server / Oozie
- Hive Server / Hive/HCat
- Ganglia Monitor / Ganglia

Below the components list, there are 'Clients' listed as 'MapReduce Client, ZooKeeper Client, HDFS Client'. The 'Host Metrics' section on the right contains six graphs: CPU Usage (90%), Disk Usage (372.9 GB, 166.2 GB), Load (2), Memory Usage (9.3 GB), Network Usage (1.9 MB, 976.5 KB), and Processes (200).

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

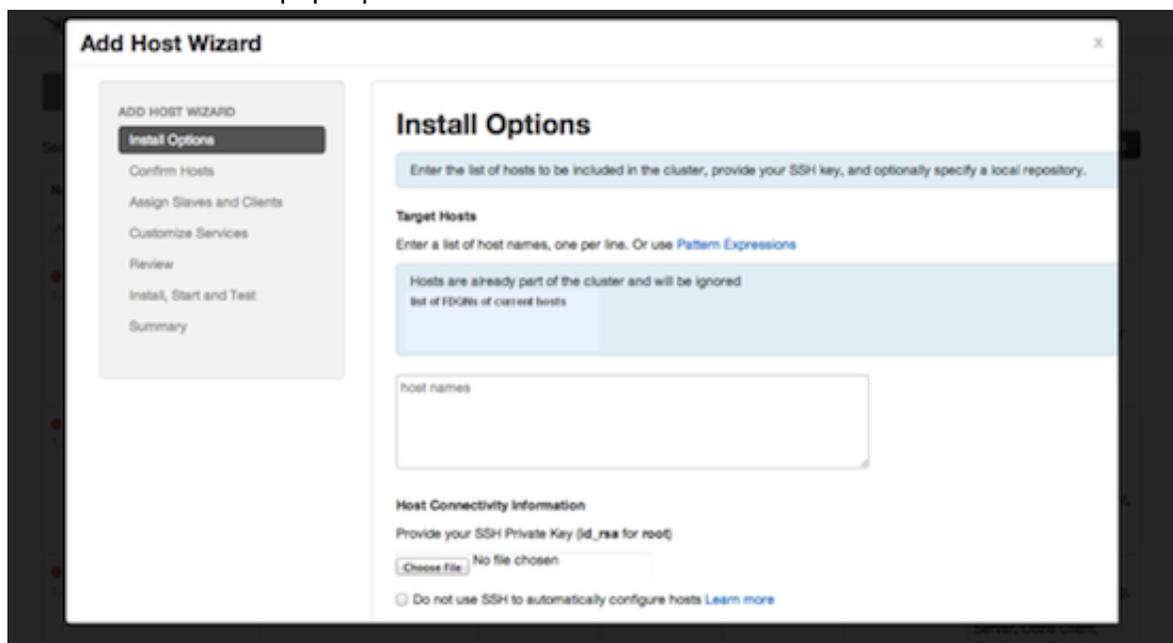
You can also use the Action dropdown on a DataNode host to decommission that node and safely move data from it to other DataNodes in your cluster.



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

2.5.2. Add Hosts

To add new hosts to your cluster, click **+Add New Hosts**, outlined in [green \[15\]](#) above. The **Add Host Wizard** pops up.



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

2.6. The Jobs View

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

2.6.1. Browsing Jobs

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

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

Jobs							Input	Output	Duration	Oldest	Most Recent
Avg		1.00	10.7KB	10.1KB	18.06 secs	Tue Jan 29 2013	Tue Jan 29 2013				
Min / Max		1 / 1	1.5KB / 36.6KB	<1KB / 36.6KB	11.29 secs / 25.90 secs						

Show: All (4) Filtered (4)		Clear filters		Search	
------------------------------	--	---------------	--	--------	--

App ID	Name	Type	User	Jobs	Input	Output	Duration	Run Date
mr_201301291547_0001	word count	MapReduce	ambari_qa	1	2.1KB	2.1KB	25.90 secs	Today 12:49:22 PM
mr_201301291547_0003	oozie:launcher:T=map-reduce:W=map-reduce-wf:A=mr-node:ID=0000000-130129154952030-oozie-oozi-W	MapReduce	ambari_qa	1	36.6KB	36.6KB	11.29 secs	Today 12:52:18 PM
mr_201301291547_0004	oozie:action:T=map-reduce:W=map-reduce-wf:A=mr-node:ID=0000000-130129154952030-oozie-oozi-W	MapReduce	ambari_qa	1	1.5KB	1.5KB	19.67 secs	Today 12:52:34 PM
pig_79ee67442-1b75-4e	/tmp/pigSmoke.sh	Pig	ambari_qa	1	2.4KB	<1KB	15.17 secs	Today 12:50:24 PM

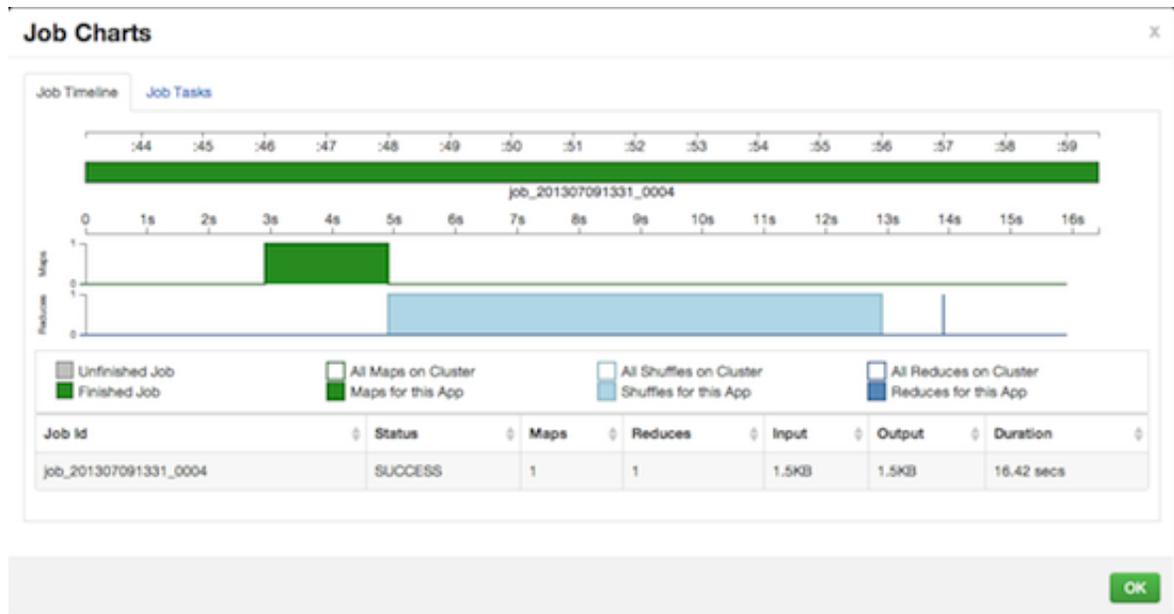
Show: 10	1 - 4 of 4
----------	------------

2.6.2. Using Job Charts

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

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

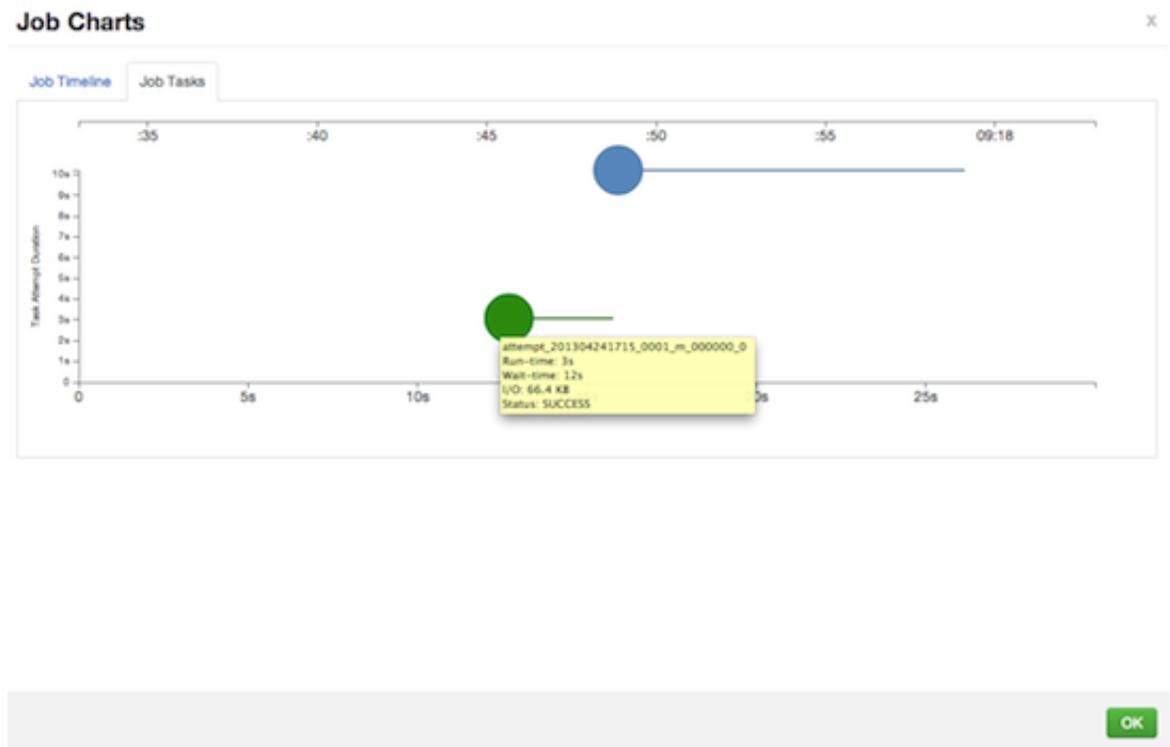
The **Job Charts** screen pops up in the **Job Timeline** tab.



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

- The X-axis displays the overall execution time. The example starts at 0 and increases to 25 seconds on the far right. Map tasks are shown in green, shuffle segments in light blue, and the actual reduce tasks in dark blue.
- The Y-axis shows the number of tasks running. The color-filled objects show the tasks for the currently displayed job and the white-filled objects show the number of tasks running in the system as a whole at that moment, so that you can understand the context in which the job is being executed.

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



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

2.7. Admin View

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

2.7.1. Managing Ambari Web Users

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

Username	Admin	Type	Action
admin	<input checked="" type="checkbox"/>	Local	edit delete

[+Add Local User](#)

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

Username

Password

Retype Password

Admin

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

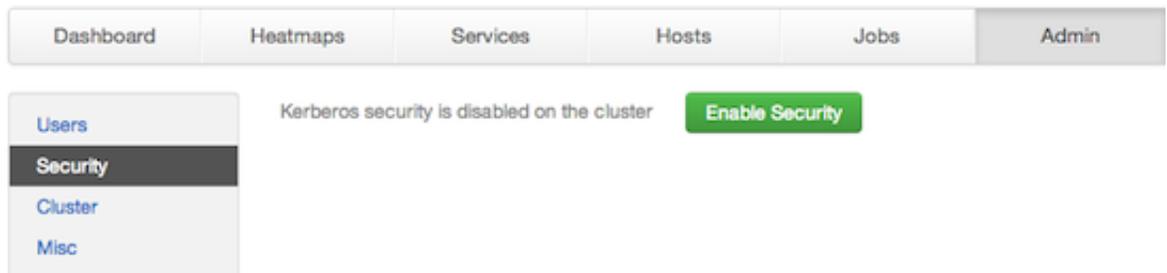
For an Ambari Web existing user, enter the old and new **Passwords**. Make your changes and click **Save**.

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

2.7.2. Enabling Kerberos Security

To turn on Kerberos-based security you must:

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



- a. **Get Started:** This step just reminds you that you need to set up Kerberos before you start. For more information, see [Setting Up Kerberos for Use with Ambari](#).
- b. **Configure Services:** This step asks you for your Kerberos information: principals and paths to keytabs, path to your Kerberos tools, realm names and so on. For

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

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



Note

If you have a large cluster, you may want to go to the **Create Principals and Keytabs** step first. Step through the wizard accepting the defaults to get to the 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 filename. Then use the **Back** button to return to the **Configure Services** step after you have gotten everything set up.

ADD SECURITY WIZARD

[Get Started](#)

[Configure Services](#)

Create Principals and Keytabs

[Save and Apply Configuration](#)

Create Principals and Keytabs

You need to create the following principals and keytabs on the hosts shown. You can download the list as a CSV file and use it to create a script to generate the principals and keytabs. Once the principals and keytabs have been created, click on Proceed to continue. If you need to make configuration changes, click Back.

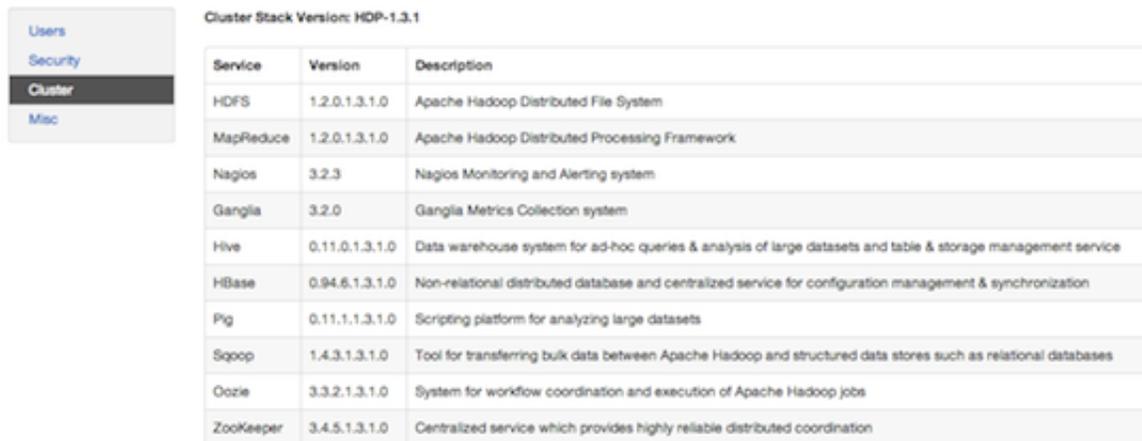
Host	Component	Principal	Keytab
FQDN	Ambari Smoke Test User	ambari-qa@EXAMPLE.COM	/etc/security/keytabs/smokeuser.headless.keytab
FQDN	HDFS User	hdfs@EXAMPLE.COM	/etc/security/keytabs/hdfs.headless.keytab
FQDN	HBase User	hbase@EXAMPLE.COM	/etc/security/keytabs/hbase.headless.keytab
FQDN	SPNEGO User	HTTP/p-10-191-37-233.ec2.internal@EXAMPLE.COM	/etc/security/keytabs/spnego.service.keytab
FQDN	DataNode	dn/p-10-191-37-233.ec2.internal@EXAMPLE.COM	/etc/security/keytabs/dn.service.keytab
FQDN	HBase Master	hbase/p-10-191-37-233.ec2.internal@EXAMPLE.COM	/etc/security/keytabs/hbase.service.keytab
FQDN	HiveServer2	hive/p-10-191-37-233.ec2.internal@EXAMPLE.COM	/etc/security/keytabs/hive.service.keytab
FQDN	JobTracker	jt/p-10-191-37-233.ec2.internal@EXAMPLE.COM	/etc/security/keytabs/jt.service.keytab

[← Back](#)
[Download CSV](#)
[Apply →](#)

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

2.7.3. Checking Stack and Component Versions

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



Cluster Stack Version: HDP-1.3.1

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

2.7.4. Checking Users and Groups

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

Users
Security
Cluster
Misc

Service Users and Groups

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

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

3.1. Basic Nagios Architecture

The basic Nagios set-up consists of a central server that polls all the Hadoop daemons - Hadoop master and slave nodes, the Templeton server, Zookeeper nodes, and HBase servers - in your installation.

Nagios checks these services for one of the following states:

- OK
- Warning
- Critical

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

Out of the box Ambari provides a set of Nagios plugins specially designed for monitoring important aspects of your Hadoop cluster.

3.2. Installing Nagios

The Ambari Installation Wizard gives you the option of installing and configuring Nagios, including the provided Hadoop-specific alerts. The Nagios server, Nagios plugins, and its default web-based user interface are installed on the Nagios host, as specified during the installation procedure.

Some of the Nagios plugins also require the Simple Network Management Protocol (SNMP) daemon to work, so the daemon is automatically installed on all cluster nodes if Nagios

is installed. For example, the SNMP daemon is used to collect information about system resources (like memory, storage, swap space, and CPU utilization).



Note

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

3.3. File Locations

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

Hadoop-specific plugins are stored in the Nagios plugins directory. The default location is `/usr/lib64/nagios/plugins/`.

For security reasons the Nagios server runs as separate user named "nagios".

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.

To change these default directive values, an administrator must:

- Modify the configuration file, `/etc/nagios/objects/hadoop-services.cfg`

See [Configuring New Alerts For Hadoop](#) for more information on the structure of this file.

- Restart the Nagios service using following command as `root` user:

```
service nagios restart
```

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 the **Dashboard** View or the individual **Services** Views.

3.5.1. Host Alerts

These alerts cover the status of the all the hosts in your installation.

3.5.1.1. Host down

This alert is configured for all nodes in the Hadoop cluster (Hadoop master and slave nodes) as well as the Nagios and Ganglia monitoring servers. By default, it uses the Nagios plugin `check_ping` to find the average round trip response (RTT) time and the packet loss percentage by pinging each cluster node.

This alert helps Ambari Web determine the number of cluster nodes that are up and down at a given time. A network outage may also result in a host down alert.



Note

The `hadoop-services.cfg` file does not define this alert explicitly. Instead, this alert is defined as a part of the generic host definition in the `templates.cfg` file using the `check-host-alive` plugin.

3.5.1.1.1. Possible causes

- The host is actually down
- There is a network outage and the Nagios server cannot access the host

3.5.1.1.2. Possible remedies

- Check the host and restart if necessary
- Check network connections

3.5.2. System Alerts for Master Daemon Nodes

The following alerts are configured for all nodes in the cluster running Hadoop master daemons, that is those running NameNode, JobTracker, and HBaseMaster:

3.5.2.1. CPU utilization alert

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

3.5.2.1.1. Possible causes

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

3.5.2.1.2. Potential remedies

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

3.5.3. HDFS Service Alerts

These alerts are used to monitor the HDFS service.

3.5.3.1. Percent capacity utilization alert

This alert is triggered if the HDFS capacity utilization exceeds the configured critical threshold. It uses the `check_hdfs_capacity` plug-in.

3.5.3.1.1. Potential causes

- Cluster storage is full

3.5.3.1.2. 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

3.5.3.2. Corrupt/missing blocks alert

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

3.5.3.2.1. 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

3.5.3.2.2. 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

3.5.3.3. NameNode edit logs directory status alert

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

3.5.3.3.1. 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

3.5.3.3.2. 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.

3.5.3.4. Percent DataNodes down alert

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

3.5.3.4.1. 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

3.5.3.4.2. Possible remedies

- Check for dead DataNodes in the **Services** list.
- 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.

3.5.3.5. DataNode process down alert

This alert is triggered if the various 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.

3.5.3.5.1. Potential causes

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

3.5.3.5.2. Possible remedies

- Check for dead DataNodes in the **Services** list.
- 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

3.5.3.6. Percent DataNodes storage full alert

This alert is triggered if the percentage of DataNodes in `storage full` condition exceeds the configured critical threshold. It uses the `check_aggregate` plugin, to aggregate the results of `DATANODE::Storage full` alert checks.

3.5.3.6.1. Potential causes

- Check for dead DataNodes in the **Services** list.
- Data distribution among the live nodes is not balanced

3.5.3.6.2. Possible remedies

- Run the Hadoop Balancer tool to distribute data to relatively less used DataNodes, freeing up space on the full nodes.

3.5.3.7. NameNode process down alert

This 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` ¹plugin.

¹The `check_tcp` plugin tests if a process is up and listening on a specified socket (host/port) address. Ambari uses this check to determine the run time status of various Hadoop services. With future Ambari releases this functionality will include more robust tests such as running some service operations to make sure the service is healthy.

3.5.3.7.1. 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.

3.5.3.7.2. 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

3.5.3.8. NameNode RPC latency alert

This 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.

3.5.3.8.1. Potential causes

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

3.5.3.8.2. Possible remedies

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

3.5.3.9. DataNode storage full alert

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

3.5.3.9.1. Potential causes

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

3.5.3.9.2. 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

3.5.4. MapReduce Service Alerts

These alerts are used to monitor the MapReduce service.

3.5.4.1. JobTracker process down alert

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

3.5.4.1.1. Potential causes

- The JobTracker daemon is down for reasons such as `OutOfMemory` errors or the misconfiguration of JobTracker, etc. Using CapacityScheduler should usually prevent this issue from occurring.
- There are hardware problems on the JobTracker host

3.5.4.1.2. Possible remedies

- Login to the JobTracker machine and verify that the JobTracker daemon is not running
- Check the logs for errors
- Check the status of the host itself
- Restart JobTracker

3.5.4.2. JobTracker RPC latency alert

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

3.5.4.2.1. Potential causes

- High load on the JobTracker in terms of the number of tasks being scheduled and completed. For example, a large number of very short-running tasks which cause extreme load on the JobTracker could cause this. Using CapacityScheduler should usually prevent this from occurring.

3.5.4.2.2. Possible remedies

- Check the running jobs using `bin/hadoop job -list` or on the JobTracker UI to find the offending job(s) running very large number of short-running tasks.
- If necessary, abort the offending job(s) via `bin/hadoop job -kill [jobId]`

3.5.4.3. Percent TaskTrackers down alert

This alert is triggered when the configured critical threshold of TaskTracker hosts become inaccessible in a short time-window. It uses the `check_aggregate` plugin to aggregate the results of individual `Tasktracker process down alert` checks.

3.5.4.3.1. Potential causes

- Connectivity issues such as general network problems, switch failures on the top-of-the-rack, etc.

3.5.4.3.2. Possible remedies

- Check the JobTracker UI for the list of TaskTrackers. If you see a lot of down TaskTrackers on a small set of racks, check for network connectivity issues between the racks
- Check for errors in the TaskTracker logs on the individual machines (see TaskTracker Process Down Alert section for more information)
- Fix the hardware/network issues and restart the TaskTrackers

3.5.4.4. TaskTracker process down alert

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

3.5.4.4.1. Potential causes

- Hardware failure
- Network connectivity issues
- Resource exhaustion (disk space, file handles, memory) for the TaskTracker process

3.5.4.4.2. Possible remedies

- Login to the individual TaskTracker host and check the TaskTracker logs
- Restart TaskTracker

3.5.5. HBase Service Alerts

These alerts are used to monitor the HBase service.

3.5.5.1. HBasemaster process down alert

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

3.5.5.1.1. 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

3.5.5.1.2. 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

3.5.5.2. RegionServer process down alert

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

3.5.5.2.1. Potential causes

- Misconfiguration or less-than-ideal configuration has caused the RegionServers to crash
- Cascading failures brought on by some workload has caused the RegionServers to crash
- The RegionServers have shut themselves down on their 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 Zookeeper

3.5.5.2.2. 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

3.5.5.3. HBase percent region servers down alert

This alert is triggered if the configured percentage of Region Server processes cannot be determined to be up and listening on the network for the configured critical threshold. The default setting is 10% to produce a WARN alert and 30% to produce a CRITICAL alert. It

uses the `check_aggregate` plugin to aggregate the results of `RegionServer` process down alert checks.

3.5.5.3.1. 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 Zookeeper

3.5.5.3.2. 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. Hive Metastore Service Alerts

These alerts are used to monitor the Hive Metastore service.

3.5.6.1. Hive-Metastore process alert

This 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.

3.5.6.1.1. Potential causes

- The Hive Metastore service is down
- The MySQL service is down
- The Hive Metastore host is not reachable over the network

3.5.6.1.2. Possible remedies

- Using the **Management Header** on the **Services** View, stop the Hive service and then restart it.
- Use `ping` to check the network connection between the Nagios server and the Hive Metastore process

3.5.7. ZooKeeper Service Alerts

These alerts are used to monitor the Zookeeper service.

3.5.7.1. Percent ZooKeeper servers down alert

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

3.5.7.2. Zookeeper process down alert

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

3.5.7.2.1. Potential causes

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

3.5.7.2.2. Possible remedies

- Check for dead DataNodes in the **Services** list.
- Check for any errors in the ZooKeeper logs (`/var/log/hadoop/zookeeper`) and restart the ZooKeeper hosts/processes
- Run the `netstat -tuplpn` command to check if the ZooKeeper process is bound to the correct network port.

3.5.8. Oozie Service Alerts

These alerts are used to monitor the Oozie service.

3.5.8.1. Oozie status alert

This alert is triggered if the Oozie service cannot be determined to be up and responding to client requests.

3.5.8.1.1. Potential causes

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

3.5.8.1.2. Possible remedies

- Restart the Oozie service using the management console.

3.5.9. WebHCat Service Alerts

These alerts are used to monitor the WebHCat service.

3.5.9.1. WebHCat status alert

This alert is triggered if the WebHCat service cannot be determined to be up and responding to client requests.

3.5.9.1.1. Potential causes

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

3.5.9.1.2. Possible remedies

- Restart the WebHCat service using Ambari Web.

3.5.10. Nagios and Ganglia Server Alerts

To see these alerts, use the **Nagios** and **Ganglia** buttons on the Dashboard.

3.5.10.1. Nagios status log staleness alert

This 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.

3.5.10.1.1. 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.

3.5.10.1.2. Possible remedies

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

3.5.10.2. Ganglia [gmetad] Process down alert

This alert determines if the Ganglia server (`gmetad`) is running and listening on the network port. It uses the Nagios `check_tcp` plugin.

3.5.10.2.1. Potential causes

- The `gmetad` process is down
- The `gmetad` process is hanging
- The network connection is down between the Nagios and Ganglia servers

3.5.10.2.2. Possible remedies

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

3.5.10.3. Ganglia collector [`gmond`] processes down alert for workers, NameNode, JobTracker, HBaseMaster

These alerts check if the Ganglia collector daemons (`gmond`) on the Ganglia server are running and listening on the network port. Ganglia uses collector daemons (`gmond`) on the Ganglia server: one for the Hadoop master daemon and one for aggregated metrics from the group of Hadoop slaves. This alert uses the Nagios `check_tcp` plugin.

3.5.10.3.1. Potential causes

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

3.5.10.3.2. Possible remedies

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

3.6. Configuring New Alerts For Hadoop

The out-of-the-box Nagios alerts displayed in Ambari Web cover a broad range of Hadoop behavior, but often you want to create additional alerts based on the needs of the individual installation. This section provides a high-level description of the process of adding those alerts so that they can be displayed in Ambari Web.

Step 1: Create a Nagios Plugin Script/Executable

You must begin by creating a Nagios plugin that can check for the particular conditions that you wish to monitor. There are many pre-written plugin scripts available at the Open Source Nagios Plugin project that can be customized for your specific purposes. You can also look at the OOTB plugin scripts that ship with Ambari. The default location for those

files on the Nagios server is `/usr/lib64/nagios/plugins/`. For more information on creating Nagios plugins see the Nagios Plugin project page at <http://nagiosplug.sourceforge.net/developer-guidelines.html>.

Step 2: Save Your Plugin to the Plugin Directory on the Nagios Server Machine

The default location is `/usr/lib64/nagios/plugins/`.

Step 3: Define the Command to Execute the New Plug-In

In `/etc/nagios/objects` find and open the `hadoop-commands.cfg` file with a text editor. Add the following information to the list:

```
define command{
    command_name my_command_name
    command_line $USER1$/my_command_name.sh
                $HOSTADDRESS$ $ARG1$ $ARG2$
```

where:

Table 3.1. Define Command

Variable Name	Variable Definition
<code>command_name</code>	The command name
<code>command_line</code>	The command with arguments used to launch the command

Notice that the `command_line` in the sample includes standard Nagios variables like `$ARG1$` and `$HOSTADDRESS$`. The variable `$USER1$` is the Nagios plugin directory path. Write the full command with arguments down for later use.

Step 4: Decide Which Hostgroup Your Plugin Should Check

In `/etc/nagios/objects` find and open the `hadoop-hostgroups.cfg` file. Write down the `hostgroup_name` that corresponds to the set of hosts your check should run against.

Step 5: Decide Which Servicegroup Your Plugin Belongs To

In `/etc/nagios/objects` find and open the `hadoop-servicegroups.cfg` file. Write down the `servicegroup_name` that is most applicable, creating your own if necessary. These service groups are helpful in enabling/disabling multiple alerts as a unit using the Nagios Web UI.

Step 6: Define the Alert Entry

In `/etc/nagios/objects` find and open the `hadoop-services.cfg` file. Create a service entry like the following and add it to the list:

```
define service {
    hostgroup_name    nagios-server
    use                hadoop-service
    service_description NAGIOS::Nagios status log staleness
    servicegroups     NAGIOS
    check_command      check_nagios!10!/var/
                    nagios/status.dat!/usr/bin/nagios
```

```

normal_check_interval 5
retry_check_interval 0.5
max_check_attempts 2
}

```

where:

Table 3.2. Define Service

Variable Name	Variable Definition
hostgroup	The name is the name you wrote down in Step 4
use	Indicates that this service inherits from <code>hadoop-service</code> . All services inherit from <code>hadoop-service</code> .
service_description	The name of the service/alert ^a
servicegroups	The group name you wrote down in Step 5
check_command	The command_line you entered in the <code>hadoop-commands.cfg</code> file in Step 3 ^b
normal_check_interval	The number of minutes between regularly scheduled checks on the host as long as the check does not change the state
retry_check_interval	The number of minutes between "retries" ^c
max_check_attempts	The maximum number of retry attempts ^d

^aFollow the convention of using one of the predefined Hadoop service names as a prefix, followed by double colon and then a short description of the new alert. The service name prefix is used to determine under which Service the alert appears. The list of predefined Hadoop services names includes NAMENODE, HDFS, JOBTRACKER, MAPREDUCE, HBASEMASTER, HBASE, ZOOKEEPER, HIVE-METASTORE, OOZIE, and TEMPLETON.

^bIn this format arguments are separated by the "!" character.

^cWhen 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.

^dUsually when the state of a service changes, this change is considered "soft" until multiple retries confirm it. After the state change is confirmed, it is considered "hard". This value indicates the number of attempts that must be made to confirm this state as "hard" and thus to display it.

Step 7: Restart the Server to See the New Alerts When you have finished making your edits, restart the Nagios service using following command as `root` user:

```
service nagios restart
```