Hortonworks Data Platform: Apache Ambari Operations
Copyright © 2012-2017 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 4.0 License.
http://creativecommons.org/licenses/by-sa/4.0/legalcode
# Table of Contents

1. Ambari Operations: Overview ................................................................. 1
   1.1. Ambari Architecture ....................................................................... 1
   1.2. Accessing Ambari Web .................................................................... 2
2. Understanding the Cluster Dashboard ................................................... 5
   2.1. Viewing the Cluster Dashboard ......................................................... 5
      2.1.1. Scanning Operating Status ......................................................... 6
      2.1.2. Viewing Details from a Metrics Widget ..................................... 7
      2.1.3. Linking to Service UIs .............................................................. 7
      2.1.4. Viewing Cluster-Wide Metrics .................................................. 8
   2.2. Modifying the Cluster Dashboard .................................................... 9
      2.2.1. Replace a Removed Widget to the Dashboard .............................. 10
      2.2.2. Reset the Dashboard ............................................................... 10
      2.2.3. Customizing Metrics Display .................................................... 11
   2.3. Viewing Cluster Heatmaps ............................................................. 11
3. Managing Hosts ........................................................................................ 13
   3.1. Understanding Host Status ............................................................. 13
   3.2. Searching the Hosts Page ............................................................... 14
   3.3. Performing Host-Level Actions ....................................................... 17
   3.4. Managing Components on a Host .................................................. 18
   3.5. Decommissioning a Master or Slave ............................................... 19
      3.5.1. Decommission a Component .................................................... 20
   3.6. Delete a Component ....................................................................... 20
   3.7. Deleting a Host from a Cluster ....................................................... 21
   3.8. Setting Maintenance Mode ............................................................. 21
      3.8.1. Set Maintenance Mode for a Service ........................................ 22
      3.8.2. Set Maintenance Mode for a Host ............................................ 22
      3.8.3. When to Set Maintenance Mode ............................................. 23
   3.9. Add Hosts to a Cluster .................................................................... 24
   3.10. Establishing Rack Awareness ......................................................... 25
      3.10.1. Set the Rack ID for Ambari ..................................................... 26
      3.10.2. Set the Rack ID Using a Custom Topology Script ....................... 27
4. Managing Services ................................................................................... 28
   4.1. Starting and Stopping All Services .................................................. 29
   4.2. Displaying Service Operating Summary .......................................... 29
      4.2.1. Alerts and Health Checks ....................................................... 30
      4.2.2. Modifying the Service Dashboard ............................................ 30
   4.3. Adding a Service ............................................................................ 32
   4.4. Performing Service Actions ........................................................... 36
   4.5. Rolling Restarts ............................................................................. 36
      4.5.1. Setting Rolling Restart Parameters .......................................... 37
      4.5.2. Aborting a Rolling Restart ....................................................... 38
   4.6. Monitoring Background Operations ................................................ 38
   4.7. Removing A Service ....................................................................... 40
   4.8. Operations Audit ........................................................................... 40
   4.9. Using Quick Links .......................................................................... 40
   4.10. Refreshing YARN Capacity Scheduler ........................................... 41
   4.11. Managing HDFS .......................................................................... 41
      4.11.1. Rebalancing HDFS ............................................................... 42
8.3. Modifying Alert Check Counts ................................................................. 100
8.4. Disabling and Re-enabling Alerts ............................................................. 101
8.5. Tables of Predefined Alerts ..................................................................... 101
    8.5.1. HDFS Service Alerts ......................................................................... 102
    8.5.2. HDFS HA Alerts ............................................................................... 105
    8.5.3. NameNode HA Alerts ....................................................................... 106
    8.5.4. YARN Alerts ..................................................................................... 107
    8.5.5. MapReduce2 Alerts ......................................................................... 108
    8.5.6. HBase Service Alerts ........................................................................ 108
    8.5.7. Hive Alerts ....................................................................................... 109
    8.5.8. Oozie Alerts ..................................................................................... 110
    8.5.9. ZooKeeper Alerts ............................................................................ 110
    8.5.10. Ambari Alerts .................................................................................. 110
    8.5.11. Ambari Metrics Alerts ..................................................................... 111
    8.5.12. SmartSense Alerts .......................................................................... 112
8.6. Managing Notifications ............................................................................. 112
8.7. Creating and Editing Notifications ............................................................. 112
8.8. Creating or Editing Alert Groups ................................................................. 114
8.9. Dispatching Notifications ......................................................................... 115
8.10. Viewing the Alert Status Log ................................................................... 115
    8.10.1. Customizing Notification Templates ................................................. 116
9. Using Ambari Core Services ......................................................................... 119
    9.1. Understanding Ambari Metrics ............................................................... 119
        9.1.1. AMS Architecture ........................................................................ 119
        9.1.2. Using Grafana ............................................................................... 120
        9.1.3. Grafana Dashboards Reference ...................................................... 125
        9.1.4. AMS Performance Tuning .............................................................. 163
        9.1.5. AMS High Availability (Technical Preview) .................................... 168
        9.1.6. AMS Security ............................................................................... 170
    9.2. Ambari Log Search (Technical Preview) .................................................. 175
        9.2.1. Log Search Architecture ................................................................. 175
        9.2.2. Installing Log Search ..................................................................... 176
        9.2.3. Using Log Search .......................................................................... 176
    9.3. Ambari Infra .......................................................................................... 179
1. Ambari Operations: Overview

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 not simple. To help you manage the complexity, Apache Ambari collects a wide range of information from the cluster’s nodes and services and presents it to you in an easy-to-use, centralized interface: Ambari Web.

Ambari Web displays information such as service-specific summaries, graphs, and alerts. You use Ambari Web to create and manage your HDP cluster and to perform basic operational tasks, such as starting and stopping services, adding hosts to your cluster, and updating service configurations. You also can use Ambari Web to perform administrative tasks for your cluster, such as enabling Kerberos security and performing Stack upgrades. Any user can view Ambari Web features. Users with administrator-level roles can access more options that operator-level or view-only users can. For example, an Ambari administrator can manage cluster security, an operator user can monitor the cluster, but a view-only user can only access features to which an administrator grants required permissions.

More Information

Hortonworks Data Platform Apache Ambari Administration

Hortonworks Data Platform Apache Ambari Upgrade

1.1. Ambari Architecture

The Ambari Server collects data from across your cluster. Each host has a copy of the Ambari Agent, which allows the Ambari Server to control each host.

The following graphic is a simplified representation of Ambari architecture:
Ambari Web is a client-side JavaScript application that 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. Communication between the browser and server occurs asynchronously using the REST API.

The Ambari Web UI periodically accesses the Ambari REST API, which resets the session timeout. Therefore, by default, Ambari Web sessions do not timeout automatically. You can configure Ambari to timeout after a period of inactivity.

More Information

Ambari Web Inactivity Timeout

1.2. Accessing Ambari Web

To access Ambari Web:

Steps

1. Open a supported browser.

2. Enter the Ambari Web URL:

   http://<your.ambari.server>:8080

   The Ambari Web login page displays in your browser.
3. Enter your user name and password.

   If you are an Ambari administrator accessing the Ambari Web UI for the first time, use the default Ambari administrator account

   admin/admin

4. Click Sign In.

   If Ambari Server is stopped, you can restart it using a command line editor on the Ambari Server host machine:

   ambari-server start

   Typically, you start the Ambari Server and Ambari Web as part of the installation process.

   Ambari administrators access the Ambari Admin page from the Manage Ambari option in Ambari Web:
More Information

Ambari Administration Overview

Hortonworks Data Platform Apache Ambari Installation
2. Understanding the Cluster Dashboard

You monitor your Hadoop cluster using the Ambari Web Cluster dashboard. You access the Cluster dashboard by clicking Dashboard at the top of the Ambari Web UI main window:

More Information

- Viewing the Cluster Dashboard [5]
- Modifying the Cluster Dashboard [9]
- Viewing Cluster Heatmaps [11]

2.1. Viewing the Cluster Dashboard

Ambari Web UI displays the Dashboard page as the home page. Use Dashboard to view the operating status of your cluster.

The left side of Ambari Web displays the list of Hadoop services currently running in your cluster. Dashboard includes Metrics, Heatmaps, and Config History tabs; by default, the Metrics tab is displayed. On the Metrics page, multiple widgets, represent operating status information of services in your HDP cluster. Most widgets display a single metric: for example, HDFS Disk Usage represented by a load chart and a percentage figure:

Metrics Widgets and Descriptions
HDFS metrics

HDFS Disk Usage: The percentage of distributed file system (DFS) used, which is a combination of DFS and non-DFS used.

Data Nodes Live: The number of DataNodes operating, as reported from the NameNode.

NameNode Heap: The percentage of NameNode Java Virtual Machine (JVM) heap memory 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 metrics (HDP 2.1 or later stacks)

ResourceManager Heap: The percentage of ResourceManager JVM heap memory used.

ResourceManager Uptime: The ResourceManager uptime calculation.

NodeManagers Live: The number of DataNodes operating, as reported from the ResourceManager.

YARN Memory: The percentage of available YARN memory (used versus total available).

HBase metrics

HBase Master Heap: The percentage of NameNode JVM heap memory 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 metrics (HDP 2.1 or later stacks)

Supervisors Live: The number of supervisors operating as reported by the Nimbus server.

More Information

Modifying the Service Dashboard [30]

Scanning Operating Status [6]

2.1.1. Scanning Operating Status

The service summary list on the left side of Ambari Web lists all of the Apache component services that are currently monitored. The icon shape, color, and action to the left of each item indicates the operating status of that item:
Status Indicators

<table>
<thead>
<tr>
<th>Color</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>solid green</td>
<td>All masters are running.</td>
</tr>
<tr>
<td>blinking green</td>
<td>Starting up</td>
</tr>
<tr>
<td>solid red</td>
<td>At least one master is down.</td>
</tr>
<tr>
<td>blinking red</td>
<td>Stopping</td>
</tr>
</tbody>
</table>

Click a service name to open the Services page, on which you can see more detailed information about that service.

2.1.2. Viewing Details from a Metrics Widget

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

- To remove a widget, click the white X
- To edit the display of information in a widget, click the edit (pencil) icon.

More Information

Customizing Metrics Display [11]

2.1.3. Linking to Service UIs

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 by 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 additional links to metrics for the following services:

**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 JMX: Links to the HBase Master JMX servlet
- Debug Dump: Links to debug information
- Thread Stacks: Links to the HBase Master thread stack traces

### 2.1.4. Viewing Cluster-Wide Metrics

From the Metrics tab, you can also view the following cluster-wide metrics:

![Metrics](image)

These metrics widgets show the following information:

- **Memory usage**: Cluster-wide memory used, including memory that is cached, swapped, used, and 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

You can customize this display as follows:
• To remove a widget
  Click the white X.

• To magnify the chart or itemize the widget display
  Hover your cursor over the widget.

• To remove or add metrics
  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 separate window:

![Cluster Load](image)

You can use the larger view in the same ways that you use the dashboard.

To close the larger view, click OK.

2.2. Modifying the Cluster Dashboard

You can modify the content of the Ambari Cluster dashboard in the following ways:

• Replace a Removed Widget to the Dashboard [10]
• Reset the Dashboard [10]
• Customizing Metrics Display [11]

2.2.1. Replace a Removed Widget to the Dashboard

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

Steps

1. Select Metric Actions:

2. Click Add.

3. Select a metric, such as Region in Transition.

4. Click Apply.

2.2.2. Reset the Dashboard

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

Steps

1. Click Metric Actions:

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

2.2.3. Customizing Metrics Display

Although not all widgets can be edited, you can customize the way that some of them display metrics by using the Edit (pencil) icon, if one is displayed.

Steps

1. Hover your cursor over a widget.
2. Click Edit.

The Customize Widget window appears:

3. Follow the instructions in Customize Widget 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, click Apply.
5. To close the editor without saving any changes, choose Cancel.

2.3. Viewing Cluster Heatmaps

As described earlier, the Ambari web interface home page is divided into a status summary panel on the left, and Metrics, Heatmaps, and Config History tabs at the top, with the Metrics page displayed by default. When you want to view a graphical representation of your overall cluster utilization, clicking Heatmaps provides you with that information, using simple color coding known as a heatmap:

A colored block represents each host in your cluster. You can see more information about a specific host by hovering over its block, which causes a separate window to display 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 usage is not available, the block displays **Invalid data**. You can solve this issue by changing the default maximum values for the heatmap, using the **Select Metric** menu:

Heatmaps supports the following metrics:

- **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 Memory Used**
  - `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`
3. Managing Hosts

As a Cluster administrator or Cluster operator, you need to know the operating status of each hosts. Also, you need to know which hosts have issues that require action. You can use the Ambari Web Hosts page to manage multiple Hortonworks Data Platform (HDP) components, such as DataNodes, NameNodes, NodeManagers, 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 enables you to filter your selection of host components to manage, based on operating status, host health, and defined host groupings.

The Hosts tab enables you to perform the following tasks:

• Understanding Host Status [13]

• Searching the Hosts Page [14]

• Performing Host-Level Actions [17]

• Managing Components on a Host [18]

• Decommissioning a Master or Slave [19]

• Delete a Component [20]

• Deleting a Host from a Cluster [21]

• Setting Maintenance Mode [21]

• Add Hosts to a Cluster [24]

• Establishing Rack Awareness [25]

3.1. Understanding Host Status

You can view the individual hosts in your cluster on the Ambari Web Hosts page. The hosts are listed by fully qualified domain name (FDQN) and accompanied by a colored icon that indicates the host's operating status:

Red Triangle At least one master component on that host is down. You can hover your cursor over the host name to see a tooltip that lists affected components.

Orange 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.
Maintenace Mode  Black "medical bag" icon indicates a host in maintenance mode.

Alert  Red square with white number indicates the number of alerts generated on a host.

A red icon overrides an orange icon, which overrides a yellow icon. In other words, a host that has a master component down is accompanied by a red icon, even though it might have slave component or connection issues as well. Hosts in maintenance mode or are experiencing alerts, are accompanied by an icon to the right of the host name.

The following example Hosts page shows three hosts, one having a master component down, one having a slave component down, one running normally, and two with alerts:

More Information

Maintenance Mode

Alerts

3.2. Searching the Hosts Page

You can search the full list of hosts, filtering your search by host name, component attribute, and component operating status. You can also search by keyword, simply by typing a word in the search box.

The Hosts search tool appears above the list of hosts:

Steps

1. Click the search box.

Available search types appear, including:

Search by Host Attribute  Search by host name, IP, host status, and other attributes, including:
Search by Service Find hosts that are hosting a component from a given service.

Search by Component Find hosts that are hosting a components in a given state, such as started, stopped, maintenance mode, and so on.

Search by keyword Type any word that describes what you are looking for in the search box. This becomes a text filter.

2. Click a Search type.

A list of available options appears, depending on your selection in step 1.

For example, if you click Service, current services appear:
3. Click an option, (in this example, the YARN service).

The list of hosts that match your current search criteria display on the Hosts page.

4. Click option(s) to further refine your search.

Examples of searches that you can perform, based on specific criteria, and which interface controls to use:

Find all hosts with a DataNode

Find all the hosts with a DataNode that are stopped
Find all the hosts with an HDFS component

Find all the hosts with an HDFS or HBase component

3.3. Performing Host-Level Actions

Use the Actions UI control to act on hosts in your cluster. Actions that you perform that comprise more than one operation, possibly on multiple hosts, are also known as bulk operations.

The Actions control comprises a workflow that uses a sequence of three menus to refine your search: a hosts menu, a menu of objects based on your host choice, and a menu of actions based on your object choice.

For example, if you want to restart the RegionServers on any host in your cluster on which a RegionServer exists:

Steps

1. In the Hosts page, select or search for hosts running a RegionServer:

2. Using the Actions control, click Filtered Hosts > RegionServers > Restart:

3. Click OK to start the selected operation.

4. Optionally, monitor background operations to follow, diagnose, or troubleshoot the restart operation.
3.4. Managing Components on a Host

To manage components running on a specific host, click one of the FQDNs listed on the Hosts page. For example, if you click c6403.ambari.apache.org, that host's page appears. Clicking the Summary tab displays a Components pane that lists all components installed on that host:

To manage all of the components on a single host, you can use the Host Actions control at the top right of the display to start, stop, restart, delete, or turn on maintenance mode for all components installed on the selected host.

Alternatively, you can manage components individually, by using the drop-down menu shown next to an individual component in the Components pane. Each component's menu is labeled with the component's current operating status. Opening the menu displays your available management options, based on that status: for example, you can decommission, restart, or stop the DataNode component for HDFS, as shown here:
3.5. Decommissioning a Master or Slave

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

- DataNodes
- NodeManagers
- RegionServers
Decommissioning executes the following tasks:

For DataNodes  Safely replicates the HDFS data to other DataNodes in the cluster
For NodeManagers  Stops accepting new job requests from the masters and stops the component
For RegionServers  Turns on drain mode and stops the component

### 3.5.1. Decommission a Component

To decommission a component (a DataNode, in the following example):

**Steps**

1. Using Ambari Web, browse the **Hosts** page.
2. Find and click the FQDN of the host on which the component resides.
3. Using the **Actions** control, click **Selected Hosts** > **DataNodes** > **Decommission**:

The UI shows Decommissioning status while in process:

![Decommissioning UI](image)

When this DataNode decommissioning process is finished, the status display changes to Decommissioned (shown here for NodeManager).

### 3.6. Delete a Component

To delete a component:

**Steps**

1. Using Ambari Web, browse the **Hosts** page.
2. Find and click the FQDN of the host on which the component resides.
3. In **Components**, find a decommissioned component.
4. If the component status is Started, stop it.
A decommissioned slave component may restart in the decommissioned state.

5. Click **Delete** from the component drop-down menu.

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.

6. To enable Ambari to recognize and monitor only the remaining components, restart services.

**More Information**

Review and Confirm Configuration Changes [76]

### 3.7. Deleting a Host from a Cluster

Deleting a host removes the host from the cluster.

**Prerequisites**

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 host Maintenance Mode, if it is on.

To delete a component:

**Steps**

1. Using Ambari Web, browse the hosts page to find and click the FQDN of the host that you want to delete.

2. On the **Host-Details** page, click **Host Actions**.

3. Click Delete.

**More Information**

Review and Confirm Configuration Changes [76]

### 3.8. Setting Maintenance Mode

Setting Maintenance Mode enables you to suppress alerts and omit bulk operations for specific services, components, and hosts in an Ambari-managed cluster when you want to
focus on performing hardware or software maintenance, changing configuration settings, troubleshooting, decommissioning, or removing cluster nodes.

Explicitly setting Maintenance Mode for a service implicitly sets Maintenance Mode for components and hosts that run the service. While Maintenance Mode prevents bulk operations being performed on the service, component, or host, you may explicitly start and stop a service, component, or host while in Maintenance Mode.

The following sections provide examples of how to use Maintenance Mode in a three-node, Ambari-managed cluster installed using default options and having one data node, on host c6403. They describe 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.

More Information

Set Maintenance Mode for a Service [22]
Set Maintenance Mode for a Host [22]
When to Set Maintenance Mode [23]

3.8.1. Set 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.

3.8.2. Set Maintenance Mode for a Host

To set Maintanence Mode for a host by using the Host Actions control:

Steps
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.

To set Maintanence Mode for a host, using the Actions control:

Steps
2. In Actions > Selected Hosts > Hosts, choose Turn On Maintenance Mode.
3. Choose OK.
Your list of hosts shows that Maintenance Mode is set for hosts c6401 and c6403:

If you hover your cursor over each Maintenance Mode icon appearing in the hosts list, you see the following information:

- Hosts c6401 and c6403 are in Maintenance Mode.
- On host c6401, HBaseMaster, HDFS client, NameNode, and ZooKeeper Server are also in Maintenance Mode.
- On host c6403, 15 components are in Maintenance Mode.
- On host c6402, HDFS client and Secondary NameNode are in Maintenance Mode, even though the host is not.

Notice also how the DataNode is affected by setting Maintenance Mode on this host:

- Alerts are suppressed for the DataNode.
- DataNode is omitted from HDFS Start/Stop/Restart All, Rolling Restart.
- DataNode is omitted from all Bulk Operations except Turn Maintenance Mode ON/OFF.
- DataNode is omitted from Start All and / Stop All components.
- DataNode is omitted from a host-level restart/restart all/stop all/start.

3.8.3. When to Set Maintenance Mode

Four common instances in which you might want to set Maintenance Mode are to perform maintenance, to test a configuration change, to delete a service completely, and to address alerts:

You want to perform hardware, firmware, or OS maintenance on a host. While performing maintenance, you want to be able to do the following:

- 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.
You want to test a service configuration change. You will stop, start, and restart the service using a "rolling" restart to test whether restarting activates the change.

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

To text configuration changes, you want to ensure the following conditions:

- No alerts are generated by any components in this service.
- No host-level or service-level bulk operations start, stop, or restart components in this service.

You want to stop a service.

To stop a service completely, you want to ensure the following conditions:

- No warnings are generated by the service.
- No components start, stop, or restart due to host-level actions or bulk operations.

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

You want to stop a host component from generating alerts.

To stop a host component from generating alerts, you must be able to do the following:

- 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, explicitly set Maintenance Mode 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.

3.9. Add Hosts to a Cluster

To add new hosts to your cluster:

Steps
1. Browse to the Hosts page and select **Actions > +Add New Hosts**.

The **Add Host** wizard provides a sequence of prompts similar to those in the Ambari Cluster Install wizard.

2. Follow the prompts, providing information similar to that provided to define the first set of hosts in your cluster:

   ![Add Host Wizard](image)

   **Next Steps**

   Review and confirm all recommended configuration changes.

   **More Information**

   Review and Confirm Configuration Changes [76]

**Install Options**

3.10. **Establishing Rack Awareness**

You can establish rack awareness either through Ambari or through your own topology script.

**More Information**

Set the Rack ID for Ambari [26]
3.10.1. Set the Rack ID for Ambari

By setting the Rack ID, you can enable Ambari to manage rack information for hosts, including displaying the hosts in heatmaps by Rack ID, enabling users to filter and find hosts on the Hosts page by using that Rack ID.

If HDFS is installed in your cluster, Ambari passes this Rack ID information to HDFS by using a topology script. Ambari generates a topology script at /etc/hadoop/conf/topology.py and sets the net.topology.script.file.name property in core-site automatically. This topology script reads a mappings file /etc/hadoop/conf/topology_mappings.data that Ambari automatically generates. When you make changes to Rack ID assignment in Ambari, this mappings file will be updated when you push out the HDFS configuration. HDFS uses this topology script to obtain Rack information about the DataNode hosts.

There are two ways using Ambari Web to set the Rack ID: for multiple hosts, using Actions, or for individual hosts, using Host Actions.

To set the Rack ID for multiple hosts:

Steps

1. Using Actions, click selected, filtered, or all hosts.
2. Click Hosts.
3. Click Set Rack.

To set the Rack ID on an individual host:

Steps

1. Browse to the Host page.
2. Click Host Actions.
3. Click Set Rack.
3.10.2. Set the Rack ID Using a Custom Topology Script

If you do not want to have Ambari manage the rack information for hosts, you can use a custom topology script. To do this, you must create your own topology script and manage distributing the script to all hosts. Note also that because Ambari will have no access to host rack information, heatmaps will not display by rack in Ambari Web.

To manage rack information by custom script:

Steps

1. Browse to Services > HDFS > Configs.

2. Modify net.topology.script.file.name to your own custom topology script.

   For example: /etc/hadoop/conf/topology.sh:

3. Distribute that topology script to your hosts.

You can now manage the rack mapping information for your script outside of Ambari.
4. Managing Services

You use the Services tab of the Ambari Web UI home page to monitor and manage selected services running in your Hadoop cluster.

All services installed in your cluster are listed in the leftmost panel:

The Services tab enables you to perform the following tasks:

- Starting and Stopping All Services [29]
- Displaying Service Operating Summary [29]
- Adding a Service [32]
- Changing Configuration Settings [74]
- Performing Service Actions [36]
- Rolling Restarts [36]
- Monitoring Background Operations [38]
- Removing A Service [40]
- Operations Audit [40]
4.1. Starting and Stopping All Services

To start or stop all listed services simultaneously, click Actions and then click Start All or Stop All:

4.2. Displaying Service Operating Summary

Clicking the name of a service from the list displays a Summary tab containing basic information about the operational status of that service, including any alerts To refresh the monitoring panels and display information about a different service, you can click a different name from the list.

Notice the colored icons next to each service name, indicating service operating status and any alerts generated for the service.

You can click one of the View Host links, as shown in the following example, to view components and the host on which the selected service is running:
4.2.1. Alerts and Health Checks

In the **Summary** tab, you can click **Alerts** to see a list of all health checks and their status for the selected service. Critical alerts are shown first. To see alert definitions, you can click the text title of each alert message in the list to see the alert definition. The following example shows the results when you click **HBase > Services > Alerts > HBase Master Process**: 

![Alerts for HBase](image)

4.2.2. Modifying the Service Dashboard

Depending on the service, the **Summary** tab includes a Metrics dashboard that is by default populated with important service metrics to monitor:

![Metrics dashboard](image)

If you have the Ambari Metrics service installed and are using Apache HDFS, Apache Hive, Apache HBase, or Apache YARN, you can customize the Metrics dashboard. You can add and remove widgets from the Metrics dashboard, and you can create new widgets and delete widgets. Widgets can be *private* to you and your dashboard, or they can be shared in a Widget Browser library.

You must have the Ambari Metrics service installed to be able to view, create, and customize the Metrics dashboard.

4.2.2.1. Adding or Removing a Widget

To add or remove a widget in the HDFS, Hive, HBase, or YARN service Metrics dashboard:

1. Either click + to launch the Widget Browser, or click **Browse Widgets** from **Actions > Metrics**.

2. The Widget Browser displays the widgets available to add to your service dashboard, including widgets already included in your dashboard, shared widgets, and widgets
you have created. Widgets that are shared are identified by the icon highlighted in the following example:

3. If you want to display only the widgets you have created, select the “Show only my widgets” check box.

4. If you want to remove a widget shown as added to your dashboard, click to remove it.

5. If you want to add an available widget that is not already added, click Add.

4.2.2.2. Creating a Widget

1. Click + to launch the Widget Browser.

2. Either click the Create Widget button or Create Widget in the Actions menu Metrics header.

3. Select the type of widget to create.

4. Depending on the service and type of widget, you can select metrics and use operators to create an expression to be displayed in the widget.

   A preview of the widget is displayed as you build the expression.

5. Enter the widget name and description.

6. Optionally, choose to share the widget.

   Sharing the widget makes the widget available to all Ambari users for this cluster. After a widget is shared, other Ambari Admins or Cluster Operators can modify or delete the widget. This cannot be undone.

4.2.2.3. Deleting a Widget

1. Click on the “+” to launch the Widget Browser. Alternatively, you can choose the Actions menu in the Metrics header to Browse Widgets.

2. The Widget Browser displays the available widgets to add to your Service Dashboard. This is a combination of shared widgets and widgets you have created. Widgets that are shared are identified by the icon highlighted in the following example.
3. If a widget is already added to your dashboard, it is shown as **Added**. Click to remove.

4. For widgets that you created, you can select the **More...** option to delete.

5. For widgets that are shared, if you are an Ambari Admin or Cluster Operator, you will also have the option to delete.

Deleting a shared widget removes the widget from all users. This cannot be undone.

### 4.2.2.4. Export Widget Graph Data

You can export the metrics data from widget graphs using the Export capability.

1. Hover your cursor over the widget graph, or click the graph to zoom, to display the **Export** icon.

2. Click the icon and specify either CSV or JSON format.

### 4.2.2.5. Setting Display Timezone

You can set the timezone used for displaying metrics data in widget graphs.

1. In Ambari Web, click your user name and select **Settings**.

2. In the **Locale** section, select the **Timezone**.

3. Click **Save**.

The Ambari Web UI reloads and graphs are displayed using the timezone you have set.

### 4.3. Adding a Service

The Ambari installation wizard installs all available Hadoop services by default. You can choose to deploy only some services initially, and then add other services as you need them. For example, many customers deploy only core Hadoop services initially. The **Add Service** option of the **Actions** control enables you to deploy additional services without interrupting operations in your Hadoop cluster. When you have deployed all available services, the **Add Service** control display is dimmed, indicating that it is unavailable.

To add a service, follow the steps shown in this example of adding the Apache Falcon service to your Hadoop cluster:

1. Click **Actions > Add Service**.
The Add Service wizard opens.

2. Click Choose Services.

The Choose Services pane displays, showing a table of those services that are already active in a green background and with their checkboxes checked.

3. In the Choose Services pane, select the empty check box next to the service that you want to add, and then click Next.

Notice that you can also select all services listed by selecting the checkbox next to the Service table column heading.
4. In **Assign Masters**, confirm the default host assignment.

   The Add Services Wizard indicates hosts on which the master components for a chosen service will be installed. A service chosen for addition shows a grey check mark.

   Alternatively, use the drop-down menu to choose a different host machine to which master components for your selected service will be added.

5. If you are adding a service that requires slaves and clients, in the **Assign Slaves and Clients** control, accept the default assignment of slave and client components to hosts by clicking **Next**.

   Alternatively, select hosts on which you want to install slave and client components (at least one host for the slave of each service being added), and click **Next**.

**Host Roles Required for Added Services**

<table>
<thead>
<tr>
<th>Service Added</th>
<th>Host Role Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>YARN</td>
<td>NodeManager</td>
</tr>
<tr>
<td>HBase</td>
<td>RegionServer</td>
</tr>
</tbody>
</table>

6. In **Customize Services**, accept the default configuration properties.

   Alternatively, edit the default values for configuration properties, if necessary. Choose **Override** to create a configuration group for this service. Then, choose **Next**:
7. In Review, verify that the configuration settings match your intentions, and then, click Deploy:

8. Monitor the progress of installing, starting, and testing the service, and when that finishes successfully, click Next:

9. When you see the summary display of installation results, click Complete:

10. Review and confirm recommended configuration changes.

11. Restart any other components that have stale configurations as a result of adding services.

More Information
Review and Confirm Configuration Changes [76]

Choose Services

Apache Spark Component Guide
Apache Storm Component Guide
Apache Ambari Apache Storm Kerberos Configuration
Apache Kafka Component Guide
Apache Ambari Apache Kafka Kerberos Configuration
Installing and Configuring Apache Atlas
Installing Ranger Using Ambari
Installing Hue
Apache Solr Search Installation
Installing Ambari Log Search (Technical Preview)
Installing Druid (Technical Preview)

4.4. Performing Service Actions

Manage a selected service on your cluster by performing service actions. In the Services tab, click Service Actions and click an option. Available options depend on the service you have selected; for example, HDFS service action options include:

Clicking Turn On 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.

More Information

Setting Maintenance Mode [21]
Enable Service Auto-Start [95]

4.5. Rolling Restarts

When you restart multiple services, components, or hosts, use rolling restarts to distribute the task. A rolling restart stops and then starts multiple running slave components, such as DataNodes, NodeManagers, RegionServers, or Supervisors, using a batch sequence.
Important

Rolling restarts of DataNodes should be performed only during cluster maintenance.

You set rolling restart parameter values to control the number, time between, tolerance for failures, and limits for restarts of many components across large clusters.

To run a rolling restart, follow these steps:

1. From the service summary pane on the left of the Service display, click a service name.
2. On the service Summary page, click a link, such as DataNodes or RegionServers, of any components that you want to restart.
   The Hosts page lists any host names in your cluster on which that component resides.
3. Using the host-level Actions menu, click the name of a slave component option, and then click Restart.
4. Review and set values for Rolling Restart Parameters.
5. Optionally, reset the flag to restart only components with changed configurations.
6. Click Trigger Restart.

After triggering the restart, you should monitor the progress of the background operations.

More Information

Setting Rolling Restart Parameters [37]
Monitoring Background Operations [38]
Performing Host-Level Actions [17]
Aborting a Rolling Restart [38]

4.5.1. Setting Rolling Restart Parameters

When you choose to restart slave components, you should 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 three-node cluster restarts one component at a time, waits two minutes between restarts, proceeds if only one failure occurs, and restarts all existing components that run this service. Enter integer, non-zero values for all parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch Size</td>
<td>Number of components to include in each restart batch.</td>
</tr>
<tr>
<td>Wait Time</td>
<td>Time (in seconds) to wait between queuing each batch of components.</td>
</tr>
<tr>
<td>Tolerate up to x failures</td>
<td>Total number of restart failures to tolerate, across all batches, before halting the restarts and not queuing batches.</td>
</tr>
</tbody>
</table>
If you trigger a rolling restart of components, the default value of **Restart components with stale configs** is “true.” If you trigger a rolling restart of services, this value is “false.”

### 4.5.2. Aborting a Rolling Restart

To abort future restart operations in the batch, click **Abort Rolling Restart**:

More Information

Rolling Restarts [36]

### 4.6. Monitoring Background Operations

You can use the Background Operations window to monitor progress and completion of a task that comprises multiple operations, such as a rolling restart of components. The Background Operations window opens by default when you run such a task. For example, to monitor the progress of a rolling restart, click elements in the Background Operations window:

1. Click the right-arrow for each operation to show restart operation progress on each host:
2. After restart operations are complete, you can click either the right-arrow or host name to view log files and any error messages generated on the selected host:

3. Optionally, you can use the **Copy**, **Open**, or **Host Logs** icons located at the upper-right of the Background Operations window to copy, open, or view logs for the rolling restart.

For example, choose **Host Logs** to view error and output logs information for host c6403.ambari.apache.org:

As shown here, you can also select the check box at the bottom of the Background Operations window to hide the window when performing tasks in the future.
4.7. Removing A Service

**Important**
Removing a service is not reversible and all configuration history will be lost.

To remove a service:

1. Click the name of the service from the left panes of the Services tab.
2. Click Service Actions > Delete.
3. As prompted, remove any dependent services.
4. As prompted, stop all components for the service.
5. Confirm the removal.

    After the service is stopped, you must confirm the removal to proceed.

More Information
Review and Confirm Configuration Changes [76]

4.8. Operations Audit

When you perform an operation using Ambari, such as user login or logout, stopping or starting a service, and adding or removing a service, Ambari creates an entry in an audit log. By reading the audit log, you can determine who performed the operation, when the operation occurred, and other, operation-specific information. You can find the Ambari audit log on your Ambari server host, at:

```
/var/log/ambari-server/ambari-audit.log
```

When you change configuration for a service, Ambari creates an entry in the audit log, and creates a specific log file, at:

```
ambari-config-changes.log
```

By reading the configuration change log, you can find out even more information about each change. For example:

```
service_name="HDFS" config_group='default' config_group_id='-1' version='2'
```

More Information
Changing Configuration Settings

4.9. 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 following:
Quick Links are not available for every service.

4.10. Refreshing YARN Capacity Scheduler

This topic describes how to “refresh” the Capacity Scheduler from Ambari when you add or modify existing queues. After you modify the Capacity Scheduler configuration, YARN enables you to refresh the queues without restarting your ResourceManager, if you have made no destructive changes (such as completely removing a queue) to your configuration. The Refresh operation will fail with the following message: Failed to re-init queues if you attempt to refresh queues in a case where you performed a destructive change, such as removing a queue. In cases where you have made destructive changes, you must perform a ResourceManager restart for the capacity scheduler change to take effect.

To refresh the Capacity Scheduler, follow these steps:

1. In Ambari Web, browse to Services > YARN > Summary.
2. Click Service Actions, and then click Refresh YARN Capacity Scheduler.
3. Confirm that you want to perform this operation.

The refresh operation is submitted to the YARN ResourceManager.

More Information

ResourceManager High Availability [64]

4.11. Managing HDFS

This section contains information specific to rebalancing and tuning garbage collection in Hadoop Distributed File System (HDFS).

More Information

Rebalancing HDFS [42]

Tuning Garbage Collection [42]
4.11.1. Rebalancing HDFS

HDFS provides a “balancer” utility to help balance the blocks across DataNodes in the cluster. To initiate a balancing process, follow these steps:

1. In Ambari Web, browse to Services > HDFS > Summary.
2. Click Service Actions, and then click Rebalance HDFS.
3. Enter the Balance Threshold value as a percentage of disk capacity.
4. Click Start.

You can monitor or cancel a rebalance process by opening the Background Operations window in Ambari.

More Information
Monitoring Background Operations [38]
Tuning Garbage Collection [42]

4.11.2. Tuning Garbage Collection

The Concurrent Mark Sweep (CMS) garbage collection (GC) process includes a set of heuristic rules used to trigger garbage collection. This makes garbage collection less predictable and tends to delay collection until capacity is reached, creating a Full GC error (which might pause all processes).

Ambari sets default parameter values for many properties during cluster deployment. Within the export HADOOP_NameNode_Opts= clause of the hadoop-env template, two parameters that affect the CMS GC process have the following default settings:

- `-XX:+UseCMSInitiatingOccupancyOnly`
  prevents the use of GC heuristics.

- `-XX:CMSInitiatingOccupancyFraction=<percent>`
  tells the Java VM when the CMS collector should be triggered.

If this percent is set too low, the CMS collector runs too often; if it is set too high, the CMS collector is triggered too late, and concurrent mode failure might occur. The default setting for `-XX:CMSInitiatingOccupancyFraction` is 70, which means that the application should utilize less than 70% of capacity.

To tune garbage collection by modifying the NameNode CMS GC parameters, follow these steps:

1. In Ambari Web, browse to Services > HDFS.
2. Open the **Configs** tab and browse to **Advanced > Advanced hadoop-env**.

3. **Edit the hadoop-env template**.

4. Save your configurations and restart, as prompted.

**More Information**

Rebalancing HDFS [42]

### 4.11.3. Customizing the HDFS Home Directory

By default, the HDFS home directory is set to `/user/<user_name>`. You can use the `dfs.user.home.base.dir` property to customize the HDFS home directory.

1. In Ambari Web, browse to **Services > HDFS > Configs > Advanced**.

2. Click **Custom hdfs-site**, then click **Add Property**.

3. On the Add Property pop-up, add the following property:

   ```
   dfs.user.home.base.dir=<home_directory>
   ```

Where `<home_directory>` is the path to the new home directory.

4. Click **Add**, then save the new configuration and restart, as prompted.

### 4.12. Managing Atlas in a Storm Environment

When you update the Apache Atlas configuration settings in Ambari, Ambari marks the services that require a restart. To restart these services, follow these steps:

1. In Ambari Web, click the **Actions** control.

2. Click **Restart All Required**.

**Important**

Apache Oozie requires a restart after an Atlas configuration update, but might not be marked as requiring restart in Ambari. If Oozie is not included, follow these steps to restart Oozie:

1. In Ambari Web, click **Oozie** in the services summary pane on the left of the display.

2. Click **Service Actions > Restart All**.

**More Information**

Installing and Configuring Atlas Using Ambari

Storm Guide
5. Managing Service High Availability

Ambari web provides a wizard-driven user experience that enables you to configure high availability of the components in many Hortonworks Data Platform (HDP) stack services. High availability is assured through establishing primary and secondary components. In the event that the primary component fails or becomes unavailable, the secondary component is available. After configuring high availability for a service, Ambari enables you to manage and disable (roll back) high availability of components in that service.

- NameNode High Availability [44]
- ResourceManager High Availability [64]
- HBase High Availability [67]
- Hive High Availability [68]
- Oozie High Availability [70]
- Apache Atlas High Availability [71]
- Enabling Ranger Admin High Availability [73]

5.1. NameNode High Availability

To ensure that another NameNode in your cluster is always available if the primary NameNode host fails, you should enable and configure NameNode high availability on your cluster using Ambari Web.

More Information

- Configuring NameNode High Availability [44]
- Rolling Back NameNode HA [49]
- Managing Journal Nodes [59]

5.1.1. Configuring NameNode High Availability

Prerequisites

- Verify that you have at least three hosts in your cluster and are running at least three Apache ZooKeeper servers.
- Verify that the Hadoop Distributed File System (HDFS) and ZooKeeper services are not in Maintenance Mode.

HDFS and ZooKeeper must stop and start when enabling NameNode HA. Maintenance Mode will prevent those start and stop operations from occurring. If the HDFS or
ZooKeeper services are in Maintenance Mode the NameNode HA wizard will not complete successfully.

**Steps**

1. In Ambari Web, select **Services > HDFS > Summary**.

2. Click **Service Actions**, then click **Enable NameNode HA**.

3. The Enable HA wizard launches. This wizard describes the set of automated and manual steps you must take to set up NameNode high availability.

4. On the **Get Started** page, type in a Nameservice ID and click **Next**.

You use this Nameservice ID instead of the NameNode FQDN after HA is set up.

5. On the **Select Hosts** page, select a host for the additional NameNode and the JournalNodes, and then click **Next**: 

6. On the Review page, confirm your host selections and click Next:
7. Follow the directions on the **Manual Steps Required: Create Checkpoint on NameNode** page, and then click **Next**:

**Manual Steps Required: Create Checkpoint on NameNode**

1. Log in to the NameNode from a client connected to the NameNode.
2. Put the NameNode into Safe mode (one-to-one mode).
3. Go to the NameNode page and click **Next**.
4. You will be able to proceed once Ambari detects that the NameNode is in Safe mode and the checkpoint has been created successfully.

You must log in to your current NameNode host and run the commands to put your NameNode into safe mode and create a checkpoint.

8. When Ambari detects success and the message on the bottom of the window changes to **Checkpoint created**, click **Next**.

9. On the **Configure Components** page, monitor the configuration progress bars, then click **Next**:

**Configure Components**

- Stop all services
- Restart Additional components
- Stop JournalNodes
- Restart HDFS
- Start JournalNodes
- Go to the NameNode page and click **Next**.

You must log in to your current NameNode host to run the command to initialize the JournalNodes.

10. Follow the instructions on the **Manual Steps Required: Initialize JournalNodes** page and then click **Next**:

**Manual Steps Required: Initialize JournalNodes**

1. Log in to the NameNode host from a client connected to the NameNode.
2. Run the command to initialize the JournalNodes.
3. You will be able to proceed once Ambari detects that the JournalNodes have been initialized successfully.

You must log in to your current NameNode host to run the command to initialize the JournalNodes.

11. When Ambari detects success and the message on the bottom of the window changes to **JournalNodes initialized**, click **Next**.

12. On the **Start Components** page, monitor the progress bars as the ZooKeeper servers and NameNode start; then click **Next**:

**Start Components**

- Stop ZooKeeper servers
- Start JournalNodes
- Go to the NameNode page and click **Next**.
Note

In a cluster with Ranger enabled, and with Hive configured to use MySQL, Ranger will fail to start if MySQL is stopped. To work around this issue, start the Hive MySQL database and then retry starting components.

13. On the Manual Steps Required: Initialize NameNode HA Metadata page: Complete each step, using the instructions on the page, and then click Next.

14. On the Finalize HA Setup page, monitor the progress bars as the wizard completes HA setup, then click Done to finish the wizard.

15. Restart any components using Ambari Web, if necessary.

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

Steps

a. Find the current FS root on the Hive host:

```
hive --config /etc/hive/conf/conf.server --service metatool -listFSRoot
```
The output should look similar to: Listing FS Roots... hdfs://<namenode-host>/apps/hive/warehouse.

b. Change the FS root:

```
$ hive --config /etc/hive/conf/conf.server --service metatool -updateLocation <new-location><old-location>
```

For example, if your Nameservice ID is mycluster, you input:

```
```

The output looks similar to:

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

**Important**

The Hive configuration path for a default HDP 2.3.x or later stack is /etc/hive/conf/conf.server

The Hive configuration path for a default HDP 2.2.x or earlier stack is /etc/hive/conf

17 Adjust the ZooKeeper Failover Controller retries setting for your environment:

a. Browse to Services > HDFS > Configs > Advanced core-site.

b. Set `ha.failover-controller.active-standby-elector.zk.op.retries=120`.

**Next Steps**

Review and confirm all recommended configuration changes.

**More Information**

Review and Confirm Configuration Changes [76]

**5.1.2. Rolling Back NameNode HA**

To disable (roll back) NameNode high availability, perform these tasks (depending on your installation):

1. Stop HBase [50]

2. Checkpoint the Active NameNode [50]

3. Stop All Services [51]
4. Prepare the Ambari Server Host for Rollback [51]
5. Restore the HBase Configuration [52]
6. Delete ZooKeeper Failover Controllers [53]
7. Modify HDFS Configurations [53]
8. Re-create the Secondary NameNode [55]
9. Re-enable the Secondary NameNode [56]
10. Delete All JournalNodes [57]
11. Delete the Additional NameNode [58]
12. Verify the HDFS Components [58]
13. Start HDFS [59]

More Information

Configuring NameNode High Availability [44]

5.1.2.1. Stop HBase

1. In the Ambari Web cluster dashboard, click the HBase service.
2. Click Service Actions > Stop.
3. Wait until HBase has stopped completely before continuing.

5.1.2.2. Checkpoint the Active NameNode

If HDFS is used after you enable NameNode HA, but you want to revert to a non-HA state, you must checkpoint the HDFS state before proceeding with the rollback.

If the Enable NameNode HA wizard failed and you need to revert, you can omit this step and proceed to stop all services.

Checkpointing the HDFS state requires different syntax, depending on whether Kerberos security is enabled on the cluster or not:

- If Kerberos security has not been enabled on the cluster, use the following command on the active NameNode host and as the HDFS service user, to save the namespace:

  sudo su -l <HDFS_USER> -c 'hdfs dfsadmin –safemode enter' sudo su -l <HDFS_USER> -c 'hdfs dfsadmin –saveNamespace'

- If Kerberos security has been enabled on the cluster, use the following commands to save the namespace:

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

In this example, <HDFS_USER> is the HDFS service user (for example, hdfs),
<HOSTNAME> is the Active NameNode hostname, and <REALM> is your Kerberos realm.

More Information

Stop All Services [51]

5.1.2.3. Stop All Services

After stopping HBase and, if necessary, checkpointing the Active NameNode, stop all services:

1. In Ambari Web, click the Services tab.
2. Click Stop All.
3. Wait for all services to stop completely before continuing.

5.1.2.4. Prepare the Ambari Server Host for Rollback

To prepare for the rollback procedure:

Steps

1. Log in to the Ambari server host.
2. Set the following environment variables

```bash
export AMBARI_USER=AMBARI_USERNAME
Substitute the value of the administrative user for Ambari Web. The default value is admin.

export AMBARI_PW=AMBARI_PASSWORD
Substitute the value of the administrative password for Ambari Web. The default value is admin.

export AMBARI_PORT=AMBARI_PORT
Substitute the Ambari Web port. The default value is 8080.

export AMBARI_PROTO=AMBARI_PROTOCOL
Substitute the value of the protocol for connecting to Ambari Web. Options are http or https. The default value is http.

export CLUSTER_NAME=CLUSTER_NAME
Substitute the name of your cluster, which you set during installation: for example, mycluster.

export NAMENODE_HOSTNAME=NN_HOSTNAME
Substitute the FQDN of the host for the non-HA NameNode: for example, nn01.mycompany.com.

export ADDITIONAL_NAMENODE_HOSTNAME=ANN_HOSTNAME
Substitute the FQDN of the host for the additional NameNode in your HA setup.
```
export SECONDARY_NAMENODE_HOSTNAME=SNN_HOSTNAME
Substitute the FQDN of the host for the secondary NameNode for the non-HA setup.

export JOURNALNODE1_HOSTNAME=JOUR1_HOSTNAME
Substitute the FQDN of the host for the first Journal Node.

export JOURNALNODE2_HOSTNAME=JOUR2_HOSTNAME
Substitute the FQDN of the host for the second Journal Node.

export JOURNALNODE3_HOSTNAME=JOUR3_HOSTNAME
Substitute the FQDN of the host for the third Journal Node.

3. Double check that these environment variables are set correctly.

5.1.2.5. Restore the HBase Configuration

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

1. From the Ambari server host, determine whether your current HBase configuration must be restored:

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

Use the environment variables that you set up when preparing the Ambari server host for rollback for the variable names.

If hbase.rootdir is set to the NameService ID you set up using the Enable NameNode HA wizard, you must revert hbase-site to non-HA values. For example, in "hbase.rootdir":"hdfs://<name-service-id>:8020/apps/hbase/data", the hbase.rootdir property points to the NameService ID and the value must be rolled back.

If hbase.rootdir points instead to a specific NameNode host, it does not need to be rolled back. For example, in "hbase.rootdir":"hdfs://<nn01.mycompany.com>:8020/apps/hbase/data", the hbase.rootdir property points to a specific NameNode host and not a NameService ID. This does not need to be rolled back; you can proceed to delete ZooKeeper failover controllers.

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

```
```

Use the environment variables that you set up when preparing the Ambari server host for rollback for the variable names.

3. On the Ambari server host, verify that the hbase.rootdir property has been restored properly:
/var/lib/ambari-server/resources/scripts/configs.sh -u
<AMBARI_USER> -p <AMBARI_PW> -port <AMBARI_PORT> get localhost
<CLUSTER_NAME> hbase-site

The hbase.rootdir property should now be the same as the NameNode hostname,
not the NameService ID.

More Information
Prepare the Ambari Server Host for Rollback [51]
Delete ZooKeeper Failover Controllers [53]

5.1.2.6. Delete ZooKeeper Failover Controllers

Prerequisites

If the following command on the Ambari server host returns an empty items array then
you must delete ZooKeeper (ZK) Failover Controllers:

curl -u <AMBARI_USER>:<AMBARI_PW> -H "X-Requested-By: ambari"
-i <AMBARI_PROTO>://localhost:<AMBARI_PORT>/api/v1/clusters/<CLUSTER_NAME>/host_components?HostRoles/component_name=ZKFC

To delete the failover controllers:

Steps

1. On the Ambari Server host, issue the following DELETE commands:

curl -u <AMBARI_USER>:<AMBARI_PW> -H "X-Requested-By: ambari"
-i -X DELETE <AMBARI_PROTO>://localhost:<AMBARI_PORT>/
api/v1/clusters/<CLUSTER_NAME>/hosts/<NAMENODE_HOSTNAME>/
host_components/ZKFC curl -u <AMBARI_USER>:<AMBARI_PW> -H "X-Requested-By: ambari" -i -X DELETE <AMBARI_PROTO>://
localhost:<AMBARI_PORT>/api/v1/clusters/<CLUSTER_NAME>/hosts/
<ADDITIONAL_NAMENODE_HOSTNAME>/host_components/ZKFC

2. Verify that the controllers are gone:

curl -u <AMBARI_USER>:<AMBARI_PW> -H "X-Requested-By: ambari"
-i <AMBARI_PROTO>://localhost:<AMBARI_PORT>/api/v1/clusters/<CLUSTER_NAME>/host_components?HostRoles/component_name=ZKFC

This command should return an empty items array.

5.1.2.7. Modify HDFS Configurations

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

Prerequisites
Check whether you need to modify your `hdfs-site` configuration, by executing the following command on the Ambari Server host:

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

If you see any of the following properties, you must delete them from your configuration.

- `dfs.nameservices`
- `dfs.client.failover.proxy.provider.<NAMESERVICE_ID>`
- `dfs.ha.namenodes.<NAMESERVICE_ID>`
- `dfs.ha.fencing.methods`
- `dfs.ha.automatic-failover.enabled`
- `dfs.namenode.http-address.<NAMESERVICE_ID>.nn1`
- `dfs.namenode.http-address.<NAMESERVICE_ID>.nn2`
- `dfs.namenode.rpc-address.<NAMESERVICE_ID>.nn1`
- `dfs.namenode.rpc-address.<NAMESERVICE_ID>.nn2`
- `dfs.namenode.shared.edits.dir`
- `dfs.journalnode.edits.dir`
- `dfs.journalnode.http-address`
- `dfs.journalnode.kerberos.internal.spnego.principal`
- `dfs.journalnode.kerberos.principal`
- `dfs.journalnode.keytab.file`

Where `<NAMESERVICE_ID>` is the NameService ID you created when you ran the Enable NameNode HA wizard.

To modify your `hdfs-site` configuration:

**Steps**

1. On the Ambari Server host, execute the following for each property you found:

   ```
   /var/lib/ambari-server/resources/scripts/configs.sh -u <AMBARI_USER> -p <AMBARI_PW> -port <AMBARI_PORT> delete localhost <CLUSTER_NAME> hdfs-site property_name
   
   Replace property_name with the name of each of the properties to be deleted.
   ```

2. Verify that all of the properties have been deleted:
None of the properties listed above should be present.

3. Determine whether you must modify your core-site configuration:

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

4. If you see the property `ha.zookeeper.quorum`, delete it:

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

5. If the property `fs.defaultFS` is set to the NameService ID, revert it to its non-HA value:

```
"fs.defaultFS":"hdfs://<name-service-id>" The property fs.defaultFS needs to be modified as it points to a NameService ID "fs.defaultFS":"hdfs://<nn01.mycompany.com>"
```

You need not change the property `fs.defaultFS`, because it points to a specific NameNode, not to a NameService ID.

6. Revert the property `fs.defaultFS` to the NameNode host value:

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

7. Verify that the core-site properties are now properly set:

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

The property `fs.defaultFS` should be the NameNode host and the property `ha.zookeeper.quorum` should not appear.

5.1.2.8. Re-create the Secondary NameNode

You may need to recreate your secondary NameNode.

Prerequisites

Check whether you need to recreate the secondary NameNode, on the Ambari Server host:

```
api/v1/clusters/<CLUSTER_NAME>/host_components?HostRoles/component_name=SECONDARY_NAMENODE

If this returns an empty items array, you must recreate your secondary NameNode. Otherwise you can proceed to re-enable your secondary NameNode.

To recreate your secondary NameNode:

**Steps**

1. On the Ambari Server host, run the following command:


2. Verify that the secondary NameNode now exists. On the Ambari Server host, run the following command:


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

More Information

Re-enable the Secondary NameNode [56]

**5.1.2.9. Re-enable the Secondary NameNode**

To re-enable the secondary NameNode:

**Steps**

1. Run the following commands on the Ambari Server host:

   curl -u <AMBARI_USER>:<AMBARI_PW> -H "X-Requested-By: ambari" -i -X PUT -d '{"RequestInfo": {"context":"Enable Secondary NameNode"},"Body": {"HostRoles":{"state":"INSTALLED"}}}' <AMBARI_PROTO>://localhost:<AMBARI_PORT>/api/v1/clusters/<CLUSTER_NAME>/hosts/<SECONDARY_NAMENODE_HOSTNAME>/host_components/SECONDARY_NAMENODE

2. Analyze the output:

   • If this returns 200, proceed to delete all JournalNodes.

   • If this input returns the value 202, wait a few minutes and then run the following command:
More Information

5.1.2.10. Delete All JournalNodes

You may need to delete any JournalNodes.

Prerequisites

Check to see if you need to delete JournalNodes, on the Ambari Server host:

```bash
```

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

To delete the JournalNodes:

Steps

1. On the Ambari Server host, run the following command:

```bash
```

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

```bash
```
This should return an empty items array.

More Information

Delete the Additional NameNode [58]
Delete All JournalNodes [57]

5.1.2.11. Delete the Additional NameNode

You may need to delete your Additional NameNode.

Prerequisites

Check to see if you need to delete your Additional NameNode, on the Ambari Server host:

```
```

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

To delete the Additional NameNode that was set up for HA:

Steps

1. On the Ambari Server host, run the following command:

```
```

2. Verify that the Additional NameNode has been deleted:

```
```

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

5.1.2.12. Verify the HDFS Components

Before starting HDFS, verify that you have the correct components:

1. Go to Ambari Web UI > Services; then select HDFS.

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

   - NameNode
   - SNameNode
   - DataNodes
5.1.2.13. Start HDFS

1. In the Ambari Web UI, click Service Actions, then click Start.

2. If the progress bar does not show that the service has completely started and has passed the service checks, repeat Step 1.

3. To start all of the other services, click Actions > Start All in the Services navigation panel.

5.1.3. Managing Journal Nodes

After you enable NameNode high availability in your cluster, you must maintain at least three, active JournalNodes in your cluster. You can use the Manage JournalNode wizard to assign, add, or remove JournalNodes on hosts in your cluster. The Manage JournalNode wizard enables you to assign JournalNodes, review and confirm required configuration changes, and will restart all components in the cluster to take advantage of the changes made to JournalNode placement and configuration.

Please note that this wizard will restart all cluster services.

Prerequisites

• NameNode high availability must be enabled in your cluster

To manage JournalNodes in your cluster:

Steps

1. In Ambari Web, select Services > HDFS > Summary.

2. Click Service Actions, then click Manage JournalNodes.

3. On the Assign JournalNodes page, make assignments by clicking the + and - icons and selecting host names in the drop-down menus. The Assign JournalNodes page
enables you to maintain three, current JournalNodes by updating each time you make an assignment.

When you complete your assignments, click **Next**.

4. On the **Review** page, verify the summary of your JournalNode host assignments and the related configuration changes. When you are satisfied that all assignments match your intentions, click **Next**:

5. Using a remote shell, complete the steps on the **Save Namespace** page. When you have successfully created a checkpoint, click **Next**:
6. On the Add/Remove JournalNodes page, monitor the progress bars, then click Next:

7. Follow the instructions on the Manual Steps Required: Format JournalNodes page and then click Next:
8. In the remote shell, confirm that you want to initialize JournalNodes, by entering `Y`, at the following prompt:

   Re-format filesystem in QJM to [host.ip.address.1, host.ip.address.2, host.ip.address.3] ? (Y or N) **Y**

9. On the **Start Active NameNodes** page, monitor the progress bars as services re-start; then click **Next**:

10. On the **Manual Steps Required: Bootstrap Standby NameNode** page: Complete each step, using the instructions on the page, and then click **Next**.
11. In the remote shell, confirm that you want to bootstrap the standby NameNode, by entering **Y**, at the following prompt:

   RE-format filesystem in Storage Directory /grid/0/hadoop/hdfs/namenode ? (Y or N) **Y**

12. On the **Start All Services** page, monitor the progress bars as the wizard starts all services, then click **Done** to finish the wizard.

   After the Ambari Web UI reloads, you may see some alert notifications. Wait a few minutes until all the services restart and alerts clear.

13. Restart any components using Ambari Web, if necessary.

**Next Steps**

Review and confirm all recommended configuration changes.

**More Information**
5.2. ResourceManager High Availability

If you are working in an HDP 2.2 or later environment, you can configure high availability for ResourceManager by using the Enable ResourceManager HA wizard.

Prerequisites

You must have at least three:

- hosts in your cluster
- Apache ZooKeeper servers running

More Information

Configure ResourceManager High Availability [64]
Disable ResourceManager High Availability [65]

5.2.1. Configure ResourceManager High Availability

To access the wizard and configure ResourceManager high availability:

Steps

1. In Ambari Web, browse to Services > YARN > Summary.

2. Select Service Actions and choose Enable ResourceManager HA.

   The Enable ResourceManager HA wizard launches, describing a set of automated and manual steps that you must take to set up ResourceManager high availability.

3. On the Get Started page, read the overview of enabling ResourceManager HA; then click Next to proceed:

4. On the Select Host page, accept the default selection, or choose an available host, then click Next to proceed.
5. On the **Review Selections** page, expand YARN, if necessary, to review all the configuration changes proposed for YARN. Click **Next** to approve the changes and start automatically configuring ResourceManager HA.

6. On the **Configure Components** page, click **Complete** when all the progress bars finish tracking:

---

**More Information**

**Disable ResourceManager High Availability**

5.2.2. **Disable ResourceManager High Availability**

To disable ResourceManager high availability, you must delete one ResourceManager and keep one ResourceManager. This requires using the Ambari API to modify the cluster configuration to delete the ResourceManager and using the ZooKeeper client to update the znode permissions.

**Prerequisites**

Because these steps involve using the Ambari REST API, you should test and verify them in a test environment prior to executing against a production environment.

To disable ResourceManager high availability:
Steps

1. In **Ambari Web**, stop YARN and ZooKeeper services.

2. On the Ambari Server host, use the Ambari API to retrieve the YARN configurations into a JSON file:

   ```bash
   /var/lib/ambari-server/resources/scripts/configs.sh get <ambari.server> <cluster.name> yarn-site yarn-site.json
   ```

   In this example, `ambari.server` is the hostname of your Ambari Server and `cluster.name` is the name of your cluster.

3. In the `yarn-site.json` file, change `yarn.resourcemanager.ha.enabled` to false and delete the following properties:

   - `yarn.resourcemanager.ha.rm-ids`
   - `yarn.resourcemanager.hostname.rm1`
   - `yarn.resourcemanager.hostname.rm2`
   - `yarn.resourcemanager.webapp.address.rm1`
   - `yarn.resourcemanager.webapp.address.rm2`
   - `yarn.resourcemanager.webapp.https.address.rm1`
   - `yarn.resourcemanager.webapp.https.address.rm2`
   - `yarn.resourcemanager.cluster-id`
   - `yarn.resourcemanager.ha.automatic-failover.zk-base-path`

4. Verify that the following properties in the `yarn-site.json` file are set to the ResourceManager hostname you are keeping:

   - `yarn.resourcemanager.hostname`
   - `yarn.resourcemanager.admin.address`
   - `yarn.resourcemanager.webapp.address`
   - `yarn.resourcemanager.resource-tracker.address`
   - `yarn.resourcemanager.scheduler.address`
   - `yarn.resourcemanager.webapp.https.address`
   - `yarn.timeline-service.webapp.address`
   - `yarn.timeline-service.webapp.https.address`
   - `yarn.timeline-service.address`
   - `yarn.log.server.url`
5. Search the yarn-site.json file and remove any references to the ResourceManager hostname that you are removing.

6. Search the yarn-site.json file and remove any properties that might still be set for ResourceManager IDs: for example, rm1 and rm2.

7. Save the yarn-site.json file and set that configuration against the Ambari Server:

```
/var/lib/ambari-server/resources/scripts/configs.sh set ambari.server cluster.name yarn-site yarn-site.json
```

8. Using the Ambari API, delete the ResourceManager host component for the host that you are deleting:

```
curl --user admin:admin -i -H "X-Requested-By: ambari" -X DELETE http://ambari.server:8080/api/v1/clusters/cluster.name/hosts/hostname/host_components/RESOURCEMANAGER
```

9. In Ambari Web, start the ZooKeeper service.

10. On a host that has the ZooKeeper client installed, use the ZooKeeper client to change znode permissions:

```
/usr/hdp/current/zookeeper-client/bin/zkCli.sh
getAcl /rmstore/ZKRMStateRoot
setAcl /rmstore/ZKRMStateRoot world:anyone:rwcda
```

11. In Ambari Web, restart ZooKeeper service and start YARN service.

**Next Steps**

Review and confirm all recommended configuration changes.

**More Information**

Review and Confirm Configuration Changes [76]

### 5.3. HBase High Availability

If you are working in an HDP 2.2 or later environment, you can configure high availability for HBase by adding an HBase master component. During the Apache HBase service installation and depending on your component assignment, Ambari installs and configures one HBase Master component and multiple RegionServer components. To configure high availability for the HBase service, you can run two or more HBase Master components. HBase uses ZooKeeper for coordination of the active Master in a cluster running two or more HBase Masters.

**More Information**

Adding an HBase Master Component [67]

#### 5.3.1. Adding an HBase Master Component

1. In Ambari Web, browse to Services > HBase.
2. In **Service Actions**, click the **+ Add HBase Master** option.

3. Choose the host on which to install the additional HBase Master; then click **Confirm Add**. Ambari installs the new HBase Master and reconfigures HBase to manage multiple Master instances.

**Next Steps**

Review and confirm all recommended configuration changes.

**More Information**

*Review and Confirm Configuration Changes [76]*

### 5.4. Hive High Availability

The Apache Hive service has multiple, associated components. The primary Hive components are Hive Metastore and HiveServer2. You can configure high availability for the Hive service in HDP 2.2 or later by running two or more of each of those components. The relational database that backs the Hive Metastore itself should also be made highly available using best practices defined for the database system in use and should be done after consultation with your in-house DBA.

**More Information**

*Adding a Hive Metastore Component [68]*

#### 5.4.1. Adding a Hive Metastore Component

**Prerequisites**

If you have ACID enabled in Hive, ensure that the **Run Compactor** setting is enabled (set to True) on only one Hive metastore host.

**Steps**

1. In **Ambari Web**, browse to **Services > Hive**.

2. In **Service Actions**, click the **+ Add Hive Metastore** option.

3. Choose the host to install the additional Hive Metastore; then click **Confirm Add**.

4. Ambari installs the component and reconfigures Hive to handle multiple Hive Metastore instances.

**Next Steps**

Review and confirm all recommended configuration changes.

**More Information**

*Review and Confirm Configuration Changes [76]*

*Using Host Config Groups*
5.4.2. Adding a HiveServer2 Component

Steps

1. In Ambari Web, browse to the host on which you want to install another HiveServer2 component.
2. On the Host page, click +Add.
3. Click HiveServer2 from the list.

Ambari installs the additional HiveServer2.

Next Steps

Review and confirm all recommended configuration changes.

More Information

Review and Confirm Configuration Changes [76]

5.4.3. Adding a WebHCat Server

Steps

1. In Ambari Web, browse to the host on which you want to install another WebHCat Server.
2. On the Host page, click +Add.
3. Click WebHCat from the list.

Ambari installs the new server and reconfigures Hive to manage multiple Metastore instances.

Next Steps

Review and confirm all recommended configuration changes.

More Information

Review and Confirm Configuration Changes [76]

5.5. Storm High Availability

In HDP 2.3 or later, you can configure high availability for the Apache Storm Nimbus server by adding a Nimbus component from Ambari.

5.5.1. Adding a Nimbus Component

Steps

1. In Ambari Web, browse to Services > Storm.
2. In **Service Actions**, click the **+ Add Nimbus** option.

3. Click the host on which to install the additional Nimbus; then click **Confirm Add**.

Ambari installs the component and reconfigures Storm to handle multiple Nimbus instances.

**Next Steps**

Review and confirm all recommended configuration changes.

**More Information**

*Review and Confirm Configuration Changes [76]*

## 5.6. Oozie High Availability

To set up high availability for the Apache Oozie service in HDP 2.2 or later, you can run two or more instances of the Oozie Server component.

**Prerequisites**

- The relational database that backs the Oozie Server should also be made highly available using best practices defined for the database system in use and should be done after consultation with your in-house DBA. Using the default installed Derby database instance is not supported with multiple Oozie Server instances; therefore, you must use an existing relational database. When using Apache Derby for the Oozie Server, you do not have the option to add Oozie Server components to your cluster.

- High availability for Oozie requires the use of an external virtual IP address or load balancer to direct traffic to the Oozie servers.

**More Information**

*Adding an Oozie Server Component [70]*

### 5.6.1. Adding an Oozie Server Component

**Steps**

1. In **Ambari Web**, browse to the host on which you want to install another Oozie Server.

2. On the **Host** page, click the **+Add** button.

3. Click **Oozie Server** from the list.

   Ambari installs the new Oozie Server.

4. Configure your external load balancer and then update the Oozie configuration.

5. Browse to **Services > Oozie > Configs**.

6. In oozie-site, add the following property values:
oozie.zookeeper.connection.string List of ZooKeeper hosts with ports: for example,
  c6401.ambari.apache.org:2181,
c6402.ambari.apache.org:2181,
c6403.ambari.apache.org:2181

oozie.services.ext org.apache.oozie.service.ZKLocksService,
  org.apache.oozie.service.ZKXLogStreamingService,
  org.apache.oozie.service.ZKJobsConcurrencyService

oozie.base.url http://<loadbalancer.hostname>:11000/oozie

7. In oozie-env, uncomment the oozie_base_url property and change its value to point to
   the load balancer:

   export oozie_base_url="http://<loadbalance.hostname>:11000/
   oozie"

8. Restart Oozie.

9. Update the HDFS configuration properties for the Oozie proxy user:
   
   a. Browse to Services > HDFS > Configs.
   
   b. In core-site, update the hadoop.proxyuser.oozie.hosts property to include the
      newly added Oozie Server host.

      Use commas to separate multiple host names.

10. Restart services.

Next Steps

Review and confirm all recommended configuration changes.

More Information

Review and Confirm Configuration Changes [76]

5.7. Apache Atlas High Availability

Prerequisites

In Ambari 2.4.0.0, adding or removing Atlas Metadata Servers requires manually editing
the atlas.rest.address property.

Steps

1. Click Hosts on the Ambari dashboard; then select the host on which to install the
   standby Atlas Metadata Server.

   Ambari adds the new Atlas Metadata Server in a Stopped state.

3. Click **Atlas > Configs > Advanced**.

4. Click **Advanced application-properties** and append the atlas.rest.address property with a comma and the value for the new Atlas Metadata Server: ,http(s):<host_name>:<port_number>.

   The default protocol is "http". If the `atlas.enableTLS` property is set to true, use "https". Also, the default HTTP port is 21000 and the default HTTPS port is 21443. These values can be overridden using the `atlas.server.http.port` and `atlas.server.https.port` properties, respectively.

5. Stop all of the Atlas Metadata Servers that are currently running.

   **Important**
   
   You must use the **Stop** command to stop the Atlas Metadata Servers. Do not use a **Restart** command: this attempts to first stop the newly added Atlas Server, which at this point does not contain any configurations in `/etc/atlas/conf`.

6. On the Ambari dashboard, click **Atlas > Service Actions > Start**.

   Ambari automatically configures the following Atlas properties in the `/etc/atlas/conf/atlas-application.properties` file:

   - `atlas.server.ids`
   - `atlas.server.address.$id`
   - `atlas.server.ha.enabled`

7. To refresh the configuration files, restart the following services that contain Atlas hooks:

   - Hive
   - Storm
   - Falcon
   - Sqoop
   - Oozie

8. Click **Actions > Restart All Required** to restart all services that require a restart.

   When you update the Atlas configuration settings in Ambari, Ambari marks the services that require restart.

9. Click **Oozie > Service Actions > Restart All** to restart Oozie along with the other services.
Apache Oozie requires a restart after an Atlas configuration update, but may not be included in the services marked as requiring restart in Ambari.

**Next Steps**

Review and confirm all recommended configuration changes.

**More Information**

Review and Confirm Configuration Changes [76]

### 5.8. Enabling Ranger Admin High Availability

You can configure Ranger Admin high availability (HA) with or without SSL on an Ambari-managed cluster. Please note that the configuration settings used in this section are sample values. You should adjust these settings to reflect your environment (folder locations, passwords, file names, and so on).

**Prerequisites**

**Steps**

- **HTTPD setup for HTTP** - Enable Ranger Admin HA with Ambari, begins at step 16.
- **HTTPD setup for HTTPS** - Enable Ranger Admin HA with Ambari, begins at step 14.
6. Managing Configurations

You can optimize performance of Hadoop components in your cluster by adjusting configuration settings and property values. You can also use Ambari Web to set up and manage groups and versions of configuration settings in the following ways:

- Changing Configuration Settings [74]
- Manage Host Config Groups [78]
- Configuring Log Settings [81]
- Set Service Configuration Versions [83]
- Download Client Configuration Files [88]

More Information

- Adjust Smart Config Settings [75]
- Edit Specific Properties [76]
- Review and Confirm Configuration Changes [76]
- Restart Components [78]

6.1. Changing Configuration Settings

You can optimize service performance using the **Configs** page for each service. The **Configs** page includes several tabs you can use to manage configuration versions, groups, settings, properties and values. You can adjust settings, called "Smart Configs" that control at a macro-level, memory allocation for each service. Adjusting Smart Configs requires related configuration settings to change throughout your cluster. Ambari prompts you to review and confirm all recommended changes and restart affected services.

Steps

1. In Ambari Web, click a service name in the service summary list on the left.

2. From the the service **Summary** page, click the **Configs** tab, then use one of the following tabs to manage configuration settings.

   - Use the **Configs** tab to manage configuration versions and groups.
   - Use the **Settings** tab to manage "Smart Configs" by adjusting the green, slider buttons.
   - Use the **Advanced** tab to edit specific configuration properties and values.

3. Click **Save**.
Next Steps

Enter a description for this version that includes your current changes, review and confirm each recommended change, and then restart all affected services.

More Information

Adjust Smart Config Settings [75]
Edit Specific Properties [76]
Review and Confirm Configuration Changes [76]
Restart Components [78]

6.1.1. Adjust Smart Config Settings

Use the Settings tab to manage "Smart Configs" by adjusting the green, slider buttons.

Steps

1. On the Settings tab, click and drag a green-colored slider button to the desired value.
2. Edit values for any properties that display the Override option.
   Edited values, also called stale configs, show an Undo option.

3. Click Save.

Next Steps

Enter a description for this version that includes your current changes, review and confirm each recommended change, and then restart all affected services.

More Information

Edit Specific Properties [76]
Review and Confirm Configuration Changes [76]
Restart Components [78]

6.1.2. Edit Specific Properties

Use the Advanced tab of the Configs page for each service to access groups of individual properties that affect performance of that service.

Steps

1. On a service Configs page, click Advanced.

2. On a service Configs Advanced page, expand a category.

3. Edit the value for any property.
   Edited values, also called stale configs, show an Undo option.

4. Click Save.

Next Steps

Enter a description for this version that includes your current changes, review and confirm each recommended change, and then restart all affected services.

More Information

Review and Confirm Configuration Changes [76]
Restart Components [78]

6.1.3. Review and Confirm Configuration Changes

When you change a configuration property value, the Ambari Stack Advisor captures and recommends changes to all related configuration properties affected by your original change. Changing a single property, a "Smart Configuration", and other actions, such as
adding or deleting a service, host or ZooKeeper Server, moving a master, or enabling high availability for a component, all require that you review and confirm related configuration changes. For example, if you increase the Minimum Container Size (Memory) setting for YARN, **Dependent Configurations** lists all recommended changes that you must review and (optionally) accept.

Types of changes are highlighted in the following colors:

- Value Changes: Yellow
- Added Properties: Green
- Deleted properties: Red

To review and confirm changes to configuration properties:

**Steps**

1. In **Dependent Configurations**, for each listed property review the summary information.
2. If the change is acceptable, proceed to review the next property in the list.
3. If the change is not acceptable, click the check mark in the blue box to the right of the listed property change.

   **Clicking the check mark clears the box. Changes for which you clear the box are not confirmed and will not occur.**

4. After reviewing all listed changes, click **OK** to confirm that all marked changes occur.

**Next Steps**

You must restart any components marked for restart to utilize the changes you confirmed.
More Information
Restart Components [78]

6.1.4. Restart Components

After editing and saving configuration changes, a Restart indicator appears next to components that require restarting to use the updated configuration values.

Steps

1. Click the indicated Components or Hosts links to view details about the requested restart.

2. Click Restart and then click the appropriate action.

For example, options to restart YARN components include the following:

More Information
Review and Confirm Configuration Changes [76]

6.2. Manage Host Config 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.

To manage Configuration Groups:

Steps

1. Click a service name, then click Configs.

2. In Configs, click Manage Config Groups.

To create new groups, reassign hosts, and override default settings for host components, you can use the Manage Configuration Groups control:
To create a new configuration group:

**Steps**

1. In *Manage Config Groups*, click *Create New Configuration Group*.

2. Name and describe the group; then choose **OK**.

To add hosts to the new configuration group:

**Steps**

1. In *Manage Config Groups*, click a configuration group name.

2. Click *Add Hosts to selected Configuration Group*.
3. Using **Select Configuration Group Hosts**, click Components, then click a component name from the list.

Choosing a component filters the list of hosts to only those on which that component exists for the selected service. To further filter the list of available host names, use the Filter drop-down list. The host list is filtered by IP address, by default.

4. After filtering the list of hosts, click the check box next to each host that you want to include in the configuration group.

5. Choose **OK**.

6. In Manage Configuration Groups, choose **Save**.

To edit settings for a configuration group:

**Steps**

1. In **Configs**, click a group name.

2. Click a Config Group; then expand components to expose settings that allow **Override**.

3. Provide a non-default value; then click **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. *Either* click the name of an existing configuration group (to which the property value override provided in Step 3 applies),

b. *Or* create a new configuration group (which includes default properties, plus the property override provided in Step 3).

c. Click **OK**.

5. In **Configs**, choose **Save**.

### 6.3. Configuring Log Settings

Ambari uses sets of properties called *Log4j properties* to control logging activities for each service running in your Hadoop cluster. Initial, default values for each property reside in a `<service_name>-log4j template file`. Log4j properties and values that limit the size and number of backup log files for each service appear above the log4j template file. To access the default Log4j settings for a service; in Ambari Web, browse to `<Service_name> > **Configs** > Advanced `<service_name>-log4j`. For example, the Advanced yarn-log4j property group for the YARN service looks like:
To change the limits for the size and number of backup log files for a service:

**Steps**

1. Edit the values for the `<service_name>` backup file size and `<service_name>` # of backup files properties.

2. Click **Save**.

To customize Log4j settings for a service:

**Steps**

1. Edit values of any properties in the `<service_name>` log4j template.

2. Copy the content of the log4j template file.

3. Browse to the custom `<service_name>log4j` properties group.

4. Paste the copied content into the custom `<service_name>log4j properties`, overwriting, the default content.

5. Click **Save**.

6. Review and confirm any recommended configuration changes, as prompted.

7. Restart affected services, as prompted.
Restarting components in the service pushes the configuration properties displayed in \textit{Custom log4j.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 see refresh indicators next to each service name after upgrading to Ambari 1.5.0 or higher. Ensure that logging properties displayed in \textit{Custom logj4.properties} include any customization.

Optionally, you can create configuration groups that include custom logging properties.

\textbf{More Information}

Review and Confirm Configuration Changes [76]

Restart Components [78]

Adjust Smart Config Settings [75]

Manage Host Config Groups [78]

\section*{6.4. Set Service Configuration Versions}

Ambari enables you to manage configurations associated with a service. You can make changes to configurations, see a history of changes, compare and revert changes, and push configuration changes to the cluster hosts.

\begin{itemize}
  \item \textit{Basic Concepts} [83]
  \item \textit{Terminology} [84]
  \item \textit{Saving a Change} [84]
  \item \textit{Viewing History} [85]
  \item \textit{Comparing Versions} [86]
  \item \textit{Reverting a Change} [87]
  \item \textit{Host Config Groups} [87]
\end{itemize}

\subsection*{6.4.1. Basic Concepts}

It is important to understand how service configurations are organized and stored in Ambari. Properties are grouped into configuration types. A set of \textit{config types} composes the set of configurations for a service.
For example, the Hadoop Distributed File System (HDFS) service includes the hdfs-site, core-site, hdfs-log4j, hadoop-env, and hadoop-policy config types. If you browse to `Services > HDFS > Configs`, you can edit the configuration properties for these config types.

Ambari performs configuration versioning at the service level. Therefore, when you modify a configuration property in a service, Ambari creates a service config version. The following figure shows V1 and V2 of a service config version with a change to a property in Config Type A. After changing a property value in Config Type A in V1, V2 is created.

![Service Config Version Diagram](image)

### 6.4.2. Terminology

The following table lists configuration versioning terms and concepts that you should know.

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>configuration property</td>
<td>Configuration property managed by Ambari, such as NameNode heap size or replication factor</td>
</tr>
<tr>
<td>configuration type (config type)</td>
<td>Group of configuration properties: for example, hdfs-site</td>
</tr>
<tr>
<td>service configurations</td>
<td>Set of configuration types for a particular service: for example, hdfs-site and core-site as part of the HDFS service configuration</td>
</tr>
<tr>
<td>change notes</td>
<td>Optional notes to save with a service configuration change</td>
</tr>
<tr>
<td>service config version (SCV)</td>
<td>A particular version of a configuration for a specific service</td>
</tr>
<tr>
<td>host config group (HCG)</td>
<td>A set of configuration properties to apply to a specific set of hosts</td>
</tr>
</tbody>
</table>

### 6.4.3. Saving a Change

1. In `Configs`, change the value of a configuration property.

2. Choose `Save`. 
3. Optionally, enter notes that describe the change:

![Save Configuration](image)

4. Click **Cancel** to continue editing, **Discard** to leave the control without making any changes, or **Save** to confirm your change.

### 6.4.4. Viewing History

You can view your configuration change history in two places in Ambari Web: on the **Dashboard page**, **Config History** tab, and on each service page's **Configs** tab.

The **Dashboard > Config History** tab shows a table of all versions across all services, with each version number and the date and time the version was created. You can also see which user authored the change, and any notes about the change. Using this table, you can filter, sort, and search across versions:

![Dashboard Config History Tab](image)

The **Service > Configs** tab shows you only the most recent configuration change, although you can use the version scrollbar to see earlier versions. Using this tab enables you to quickly access the most recent changes to a service configuration:

![Service Configs Tab](image)

Using this view, you can click any version in the scrollbar to view it, and hover your cursor over it to display an option menu that enables you to compare versions and perform a revert operation, which makes any config version that you select the current version.
6.4.5. Comparing Versions

When browsing the version scroll area on the Services > Configs tab, you can hover your cursor over a version to display options to view, compare, or revert (make current):

To compare two service configuration versions:

Steps

1. Navigate to a specific configuration version: for example, V6.

2. Using the version scrollbar, find the version you want to compare to V6.

   For example, if you want to compare V6 to V2, find V2 in the scrollbar.

3. Hover your cursor over V2 to display the option menu, and click Compare.

   Ambari displays a comparison of V6 to V2, with an option to revert to V2 (Make V2 Current). Ambari also filters the display by only Changed properties, under the Filter control:
6.4.6. Reverting a Change

You can revert to an older service configuration version by using the Make Current feature. Make Current creates a new service configuration version with the configuration properties from the version you are reverting: effectively, a clone.

After initiating the Make Current operation, you are prompted, on the Make Current Confirmation control, to enter notes for the clone and save it (Make Current). The notes text includes text about the version being cloned:

There are multiple methods to revert to a previous configuration version:

- View a specific version and click Make V* Current:

- Use the version navigation menu and click Make Current:

- Hover your cursor over a version in the version scrollbar and click Make Current:

- Perform a comparison and click Make V* Current:

6.4.7. Host Config Groups

Service configuration versions are scoped to a host config group. For example, changes made in the default group can be compared and reverted in that config group. The same applies to custom config groups.

The following workflow shows multiple host config groups and creates service configuration versions in each config group:
6.5. Download Client Configuration Files

Client configuration files include; .xml files, env-sh scripts, and log4j properties used to configure Hadoop services. For services that include client components (most services except SmartSense and Ambari Metrics Service), you can download the client configuration files associated with that service. You can also download the client configuration files for your entire cluster as a single archive.

To download client configuration files for a single service:

Steps

1. In Ambari Web, browse to the service for which you want the configurations.

2. Click Service Actions.

3. Click Download Client Configs.

Your browser downloads a "tarball" archive containing only the client configuration files for that service to your default, local downloads directory.

4. If prompted to save or open the client configs bundle:
5. Click **Save File**, then click **OK**.

To download all client configuration files for your entire cluster:

**Steps**

1. In Ambari Web, click **Actions** at the bottom of the service summary list.

2. Click **Download Client Configs**.

   Your browser downloads a "tarball" archive containing all client configuration files for your cluster to your default, local downloads directory.
7. Administering the Cluster

Using the Ambari Web Admin options:

any user can view information about the stack and versions of each service added to it

Cluster administrators can

• enable Kerberos security
• regenerate required key tabs
• view service user names and values
• enable auto-start for services

Ambari administrators can

• add new services to the stack
• upgrade the stack to a new version, by using the link to the Ambari administration interface

Related Topics

Hortonworks Data Platform Apache Ambari Administration
Using Stack and Versions Information [90]
Viewing Service Accounts [92]
Enabling Kerberos and Regenerating Keytabs [93]
Enable Service Auto-Start
Managing Versions

7.1. Using Stack and Versions Information

The Stack tab includes information about the services installed and available in the cluster stack. Any user can browse the list of services. As an Ambari administrator you can also click Add Service to start the wizard to install each service into your cluster.
The **Versions** tab includes information about which version of software is currently installed and running in the cluster. As an Ambari administrator, you can initiate an automated cluster upgrade from this page.
7.2. Viewing Service Accounts

As a Cluster administrator, you can view the list of Service Users and Group accounts used by the cluster services.

Steps

In Ambari Web UI > Admin, click Service Accounts.
7.3. Enabling Kerberos and Regenerating Keytabs

As a Cluster administrator, you can enable and manage Kerberos security in your cluster.

Prerequisites

Before enabling Kerberos in your cluster, you must prepare the cluster, as described in Configuring Ambari and Hadoop for Kerberos.

Steps

In the Ambari web UI > Admin menu, click Enable Kerberos to launch the Kerberos wizard.

After Kerberos is enabled, you can regenerate key tabs and disable Kerberos from the Ambari web UI > Admin menu.

More Information

Regenerate Key tabs [94]
### 7.3.1. Regenerate Key tabs

As a Cluster administrator, you can regenerate the key tabs required to maintain Kerberos security in your cluster.

**Prerequisites**

Before regenerating key tabs in your cluster:

- Your cluster must be Kerberos-enabled
- You must have KDC Admin credentials

**Steps**

1. Browse to Admin > Kerberos.
2. Click Regenerate Kerberos.
3. Confirm your selection to proceed.
4. Ambari connects to the Kerberos Key Distribution Center (KDC) and regenerates the key tabs for the service and Ambari principals in the cluster. Optionally, you can regenerate key tabs for only those hosts that are missing key tabs: for example, hosts that were not online or available from Ambari when enabling Kerberos.
5. Restart all services.

**More Information**

- [Disable Kerberos](#)
- Configuring Ambari and Hadoop for Kerberos
- Managing KDC Admin Credentials

### 7.3.2. Disable Kerberos

As a Cluster administrator, you can disable Kerberos security in your cluster.

**Prerequisites**

Before disabling Kerberos security in your cluster, your cluster must be Kerberos-enabled.

**Steps**

1. Browse to Admin > Kerberos.
2. Click **Disable Kerberos**.

3. Confirm your selection.

   Cluster services are stopped and the Ambari Kerberos security settings are reset.

4. To re-enable Kerberos, click **Enable Kerberos** and follow the wizard

**More Information**

*Configuring Ambari and Hadoop for Kerberos*

### 7.4. Enable Service Auto-Start

As a Cluster Administrator or Cluster Operator, you can enable each service in your stack to re-start automatically. Enabling auto-start for a service causes the ambari-agent to attempt re-starting service components in a stopped state without manual effort by a user. Auto-Start Services is enabled by default, but only the Ambari Metrics Collector component is set to auto-start by default.

As a first step, you should enable auto-start for the worker nodes in the core Hadoop services, the DataNode and NameNode components in YARN and HDFS, for example. You should also enable auto-start for all components in the SmartSense service. After enabling auto-start, monitor the operating status of your services on the Ambari Web dashboard. Auto-start attempts do not display as background operations. To diagnose issues with service components that fail to start, check the ambari agent logs, located at: `/var/log/ambari-agent.log` on the component host.

To manage the auto-start status for components in a service:

**Steps**

1. In **Auto-Start Services**, click a service name.

2. Click the grey area in the **Auto-Start Services** control of at least one component, to change its status to **Enabled**.
The green icon to the right of the service name indicates the percentage of components with auto-start enabled for the service.

3. To enable auto-start for all components in the service, click **Enable All**.

   ![Status](image)

   - Status
   - Disabled
   - Disabled
   - Disabled

   **Enable All** | **Disable All**

   The green icon fills to indicate all components have auto-start enabled for the service.

4. To disable auto-start for all components in the service, click **Disable All**.

   ![Status](image)

   - Enabled
   - Enabled
   - Enabled

   **Enable All** | **Disable All**

   The green icon clears to indicate that all components have auto-start disabled for the service.

5. To clear all pending status changes before saving them, click **Discard**.

6. When you finish changing your auto-start status settings, click **Save**.

---

To disable Auto-Start Services:

**Steps**

1. In Ambari Web, click **Admin > Service Auto-Start**.
2. In **Service Auto Start Configuration**, click the grey area in the **Auto-Start Services** control to change its status from **Enabled** to **Disabled**.

3. Click **Save**.

**More Information**

*Monitoring Background Operations [38]*
8. Managing Alerts and Notifications

Ambari uses a predefined set of seven types of alerts (web, port, metric, aggregate, script, server, and recovery) for each cluster component and host. You can use these alerts to monitor cluster health and to alert other users to help you identify and troubleshoot problems. You can modify alert names, descriptions, and check intervals, and you can disable and re-enable alerts.

You can also create groups of alerts and setup notification targets for each group so that you can notify different parties interested in certain sets of alerts by using different methods.

This section provides you with the following information:

- Understanding Alerts [98]
- Modifying Alerts [100]
- Modifying Alert Check Counts [100]
- Disabling and Re-enabling Alerts [101]
- Tables of Predefined Alerts [101]
- Managing Notifications [112]
- Creating and Editing Notifications [112]
- Creating or Editing Alert Groups [114]
- Dispatching Notifications [115]
- Viewing the Alert Status Log [115]

8.1. Understanding Alerts

Ambari predefines a set of alerts that monitor the cluster components and hosts. Each alert is defined by an alert definition, which specifies the alert type check interval and thresholds. When a cluster is created or modified, Ambari reads the alert definitions and creates alert instances for the specific items to monitor in the cluster. For example, if your cluster includes Hadoop Distributed File System (HDFS), there is an alert definition to monitor "DataNode Process". An instance of that alert definition is created for each DataNode in the cluster.

Using Ambari Web, you can browse the list of alerts defined for your cluster by clicking the Alerts tab. You can search and filter alert definitions by current status, by last status change, and by the service the alert definition is associated with (among other things). You can click alert definition name to view details about that alert, to modify the alert properties (such as check interval and thresholds), and to see the list of alert instances associated with that alert definition.

Each alert instance reports an alert status, defined by severity. The most common severity levels are OK, WARNING, and CRITICAL, but there are also severities for UNKNOWN and
NONE. Alert notifications are sent when alert status changes (for example, status changes from OK to CRITICAL).

More Information

Managing Notifications [112]

Tables of Predefined Alerts [101]

8.1.1. Alert Types

Alert thresholds and the threshold units depend on the type of the alert. The following table lists the types of alerts, their possible status, and to what units thresholds can be configured if the thresholds are configurable:

<table>
<thead>
<tr>
<th>Alert Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| WEB Alert Type    | WEB alerts watch a web URL on a given component; the alert status is determined based on the HTTP response code. Therefore, you cannot change which HTTP response codes determine the thresholds for WEB alerts. You can customize the response text for each threshold and the overall web connection timeout. A connection timeout is considered a CRITICAL alert. Threshold units are based on seconds. The response code and corresponding status for WEB alerts is as follows:  
  • OK status if the web URL responds with a code under 400.  
  • WARNING status if the web URL responds with code 400 and above.  
  • CRITICAL status if Ambari cannot connect to the web URL. |
| PORT Alert Type   | PORT alerts check the response time to connect to a given port; the threshold units are based on seconds.                                                                                                      |
| METRIC Alert Type | METRIC alerts check the value of a single or multiple metrics (if a calculation is performed). The metric is accessed from a URL endpoint available on a given component. A connection timeout is considered a CRITICAL alert. The thresholds are adjustable and the units for each threshold depend on the metric. For example, in the case of CPU utilization alerts, the unit is percentage; in the case of RPC latency alerts, the unit is milliseconds. |
| AGGREGATE Alert Type | AGGREGATE alerts aggregate the alert status as a percentage of the alert instances affected. For example, the Percent DataNode Process alert aggregates the DataNode Process alert. |
| SCRIPT Alert Type | SCRIPT alerts execute a script that determines status such as OK, WARNING, or CRITICAL. You can customize the response                                                                                   |
text and values for the properties and thresholds for the
SCRIPT alert.

SERVER Alert Type
SERVER alerts execute a server-side runnable class that
determines the alert status such as OK, WARNING, or
CRITICAL.

RECOVERY Alert Type
RECOVERY alerts are handled by the Ambari Agents that are
monitoring for process restarts. Alert status OK, WARNING,
and CRITICAL are based on the number of times a process
is restarted automatically. This is useful to know when
processes are terminating and Ambari is automatically
restarting.

8.2. Modifying Alerts

General properties for an alert include name, description, check interval, and thresholds.

The check interval defines the frequency with which Ambari checks alert status. For
example, "1 minute" value means that Ambari checks the alert status every minute.

The configuration options for thresholds depend on the alert type.

To modify the general properties of alerts:

Steps
1. Browse to the Alerts section in Ambari Web.
2. Find the alert definition and click to view the definition details.
3. Click Edit to modify the name, description, check interval, and thresholds (as applicable).
4. Click Save.
5. Changes take effect on all alert instances at the next check interval.

More Information
Alert Types [99]

8.3. Modifying Alert Check Counts

Ambari enables you to set the number of alert checks to perform before dispatching
a notification. If the alert state changes during a check, Ambari attempts to check the
condition a specified number of times (the check count) before dispatching a notification.

Alert check counts are not applicable to AGGREGATE alert types. A state change for an
AGGREGATE alert results in a notification dispatch.

If your environment experiences transient issues resulting in false alerts, you can increase
the check count. In this case, the alert state change is still recorded, but as a SOFT state
change. If the alert condition is still triggered after the specified number of checks, the
state change is then considered HARD, and notifications are sent.
You generally want to set the check count value globally for all alerts, but you can also override that value for individual alerts if a specific alert or alerts is experiencing transient issues.

To modify the global alert check count:

Steps
1. Browse to the Alerts section in Ambari Web.
2. In the Ambari Web, Actions menu, click Manage Alert Settings.
3. Update the Check Count value.
4. Click Save.

Changes made to the global alert check count might require a few seconds to appear in the Ambari UI for individual alerts.

To override the global alert check count for individual alerts:

Steps
1. Browse to the Alerts section in Ambari Web.
2. Select the alert for which you want to set a specific Check Count.
3. On the right, click the Edit icon next to the Check Count property.
4. Update the Check Count value.
5. Click Save.

More Information
Managing Notifications [112]

8.4. Disabling and Re-enabling Alerts

You can optionally disable alerts. When an alert is disabled, no alert instances are in effect and Ambari will no longer perform the checks for the alert. Therefore, no alert status changes will be recorded and no notifications (i.e. no emails or SNMP traps) will dispatched.

1. Browse to the Alerts section in Ambari Web.
2. Find the alert definition. Click the Enabled or Disabled text to enable/disable the alert.
3. Alternatively, you can click on the alert to view the definition details and click Enabled or Disabled to enable/disable the alert.
4. You will be prompted to confirm enable/disable.

8.5. Tables of Predefined Alerts

- HDFS Service Alerts [102]
- HDFS HA Alerts [105]
8.5.1. HDFS Service Alerts

<table>
<thead>
<tr>
<th>Alert</th>
<th>Alert Type</th>
<th>Description</th>
<th>Potential Causes</th>
<th>Possible Remedies</th>
</tr>
</thead>
<tbody>
<tr>
<td>NameNode Blocks Health</td>
<td>METRIC</td>
<td>This service-level alert is triggered if the number of corrupt or missing blocks exceeds the configured critical threshold.</td>
<td>Some DataNodes are down and the replicas that are missing blocks are only on those DataNodes. The corrupt or 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.</td>
<td>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 one exists.</td>
</tr>
<tr>
<td>NFS Gateway Process</td>
<td>PORT</td>
<td>This host-level alert is triggered if the NFS Gateway process cannot be confirmed as active.</td>
<td>NFS Gateway is down.</td>
<td>Check for a non-operating NFS Gateway in Ambari Web.</td>
</tr>
<tr>
<td>DataNode Storage</td>
<td>METRIC</td>
<td>This host-level alert is triggered if storage capacity is full on the DataNode (90% critical). It checks the DataNode JMX Servlet for the Capacity and Remaining properties.</td>
<td>Cluster storage is full. If cluster storage is not full, DataNode is full.</td>
<td>If the cluster still has storage, use the load 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 the load balancer.</td>
</tr>
<tr>
<td>DataNode Process</td>
<td>PORT</td>
<td>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, in seconds.</td>
<td>DataNode process is down or not responding. DataNode are not down but is not listening to the correct network port/address.</td>
<td>Check for non-operating DataNodes in Ambari Web. Check for any errors in the DataNode logs (/var/log/hadoop/hdfs) and restart the DataNode, if necessary.</td>
</tr>
<tr>
<td>Alert</td>
<td>Alert Type</td>
<td>Description</td>
<td>Potential Causes</td>
<td>Possible Remedies</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------</td>
<td>------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DataNode Web UI</td>
<td>WEB</td>
<td>This host-level alert is triggered if the DataNode web UI is unreachable.</td>
<td>The DataNode process is not running.</td>
<td>Check whether the DataNode process is running.</td>
</tr>
<tr>
<td>NameNode Host CPU Utilization</td>
<td>METRIC</td>
<td>This host-level alert is triggered if CPU utilization of the NameNode exceeds certain thresholds (200% warning, 250% critical). It checks the NameNode JMX Servlet for the SystemCPULoad property. This information is available only if you are running JDK 1.7.</td>
<td>Unusually high CPU utilization might be caused by a very unusual job or query workload, but this is generally the sign of an issue in the daemon.</td>
<td>Use the <code>top</code> command to determine which processes are consuming excess CPU. Reset the offending process.</td>
</tr>
<tr>
<td>NameNode Web UI</td>
<td>WEB</td>
<td>This host-level alert is triggered if the NameNode web UI is unreachable.</td>
<td>The NameNode process is not running.</td>
<td>Check whether the NameNode process is running.</td>
</tr>
<tr>
<td>Percent DataNodes with Available Space</td>
<td>AGGREGATE</td>
<td>This service-level alert is triggered if the storage is full on a certain percentage of DataNodes (10% warn, 30% critical).</td>
<td>Cluster storage is full. If cluster storage is not full, DataNode is full.</td>
<td>If the cluster still has storage, use the load balancer to distribute the data to relatively less-used DataNodes. If the cluster is full, delete unnecessary data or increase storage by adding either more DataNodes or more or larger disks to the DataNodes. After adding more storage, run the load balancer.</td>
</tr>
<tr>
<td>Percent DataNodes Available</td>
<td>AGGREGATE</td>
<td>This alert is triggered if the number of non-operating DataNodes in the cluster is greater than the configured critical threshold. This aggregates the DataNode process alert.</td>
<td>DataNodes are down. DataNodes are not down but are not listening to the correct network port/address.</td>
<td>Check for non-operating DataNodes in Ambari Web. Check for any errors in the DataNode logs (/var/log/hadoop/hdfs) and restart the DataNode hosts/processes. Run the <code>netstat-tuplpn</code> command to check if the DataNode process is bound to the correct network port.</td>
</tr>
<tr>
<td>NameNode RPC Latency</td>
<td>METRIC</td>
<td>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.</td>
<td>A job or an application is performing too many NameNode operations.</td>
<td>Review the job or the application for potential bugs causing it to perform too many NameNode operations.</td>
</tr>
<tr>
<td>NameNode Last Checkpoint</td>
<td>SCRIPT</td>
<td>This alert will trigger if the last time that the NameNode performed a checkpoint was too long ago or if the number</td>
<td>Too much time elapsed since last NameNode checkpoint.</td>
<td>Set NameNode checkpoint. Review threshold for uncommitted transactions.</td>
</tr>
<tr>
<td>Alert</td>
<td>Alert Type</td>
<td>Description</td>
<td>Potential Causes</td>
<td>Possible Remedies</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Secondary NameNode Process</td>
<td>WEB</td>
<td>If the Secondary NameNode process cannot be confirmed to be up and listening on the network. This alert is not applicable when NameNode HA is configured.</td>
<td>The Secondary NameNode is not running.</td>
<td>Check that the Secondary DataNode process is running.</td>
</tr>
<tr>
<td>NameNode Directory Status</td>
<td>METRIC</td>
<td>This alert checks if the NameNode NameDirStatus metric reports a failed directory.</td>
<td>One or more of the directories are reporting as not healthy.</td>
<td>Check the NameNode UI for information about unhealthy directories.</td>
</tr>
<tr>
<td>HDFS Capacity Utilization</td>
<td>METRIC</td>
<td>This service-level alert is triggered if the HDFS capacity utilization exceeds the configured critical threshold (80% warn, 90% critical). It checks the NameNode JMX Servlet for the CapacityUsed and CapacityRemaining properties.</td>
<td>Cluster storage is full.</td>
<td>Delete unnecessary data.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Archive unused data.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Add more DataNodes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Add more or larger disks to the DataNodes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>After adding more storage, run the load balancer.</td>
</tr>
<tr>
<td>DataNode Health Summary</td>
<td>METRIC</td>
<td>This service-level alert is triggered if there are unhealthy DataNodes.</td>
<td>A DataNode is in an unhealthy state.</td>
<td>Check the NameNode UI for the list of non-operating DataNodes.</td>
</tr>
<tr>
<td>HDFS Pending Deletion Blocks</td>
<td>METRIC</td>
<td>This service-level alert is triggered if the number of blocks pending deletion in HDFS exceeds the configured warning and critical thresholds. It checks the NameNode JMX Servlet for the PendingDeletionBlock property.</td>
<td>Large number of blocks are pending deletion.</td>
<td></td>
</tr>
<tr>
<td>HDFS Upgrade Finalized State</td>
<td>SCRIPT</td>
<td>This service-level alert is triggered if HDFS is not in the finalized state.</td>
<td>The HDFS upgrade is not finalized.</td>
<td>Finalize any upgrade you have in process.</td>
</tr>
<tr>
<td>DataNode Unmounted Data Dir</td>
<td>SCRIPT</td>
<td>This host-level alert is triggered if one of the data directories on a host was previously on a mount point and became unmounted.</td>
<td>If the mount history file does not exist, then report an error if a host has one or more mounted data directories as well as one or more unmounted data directories on the root partition. This may indicate that a data directory is writing to the root partition, which is undesirable.</td>
<td>Check the data directories to confirm they are mounted as expected.</td>
</tr>
<tr>
<td>DataNode Heap Usage</td>
<td>METRIC</td>
<td>This host-level alert is triggered if heap usage goes past thresholds on the DataNode. It checks the DataNode JMXServlet for the MemHeapUsedM and MemHeapMaxM properties. The threshold values are percentages.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NameNode Client RPC Queue Queue Latency</td>
<td>SCRIPT</td>
<td>This service-level alert is triggered if the deviation of RPC queue latency on client port has grown beyond the</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alert</td>
<td>Alert Type</td>
<td>Description</td>
<td>Potential Causes</td>
<td>Possible Remedies</td>
</tr>
<tr>
<td>-------</td>
<td>------------</td>
<td>-------------</td>
<td>------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>NameNode Client RPC Processing Latency</td>
<td>SCRIPT</td>
<td>This service-level alert is triggered if the deviation of RPC latency on client port has grown beyond the specified threshold within a given period. This alert will monitor Hourly and Daily periods.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NameNode Service RPC Queue Latency</td>
<td>SCRIPT</td>
<td>This service-level alert is triggered if the deviation of RPC latency on the DataNode port has grown beyond the specified threshold within a given period. This alert will monitor Hourly and Daily periods.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NameNode Service RPC Processing Latency</td>
<td>SCRIPT</td>
<td>This service-level alert is triggered if the deviation of RPC latency on the DataNode port has grown beyond the specified threshold within a given period. This alert will monitor Hourly and Daily periods.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HDFS Storage Capacity Usage</td>
<td>SCRIPT</td>
<td>This service-level alert is triggered if the increase in storage capacity usage deviation has grown beyond the specified threshold within a given period. This alert will monitor Daily and Weekly periods.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NameNode Heap Usage</td>
<td>SCRIPT</td>
<td>This service-level alert is triggered if the NameNode heap usage deviation has grown beyond the specified threshold within a given period. This alert will monitor Daily and Weekly periods.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 8.5.2. HDFS HA Alerts

<table>
<thead>
<tr>
<th>Alert</th>
<th>Alert Type</th>
<th>Description</th>
<th>Potential Causes</th>
<th>Possible Remedies</th>
</tr>
</thead>
<tbody>
<tr>
<td>JournalNode Web UI</td>
<td>WEB</td>
<td>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.</td>
<td>The JournalNode process is down or not responding. The JournalNode is not down but is not listening to the correct network port/address.</td>
<td>Check if the JournalNode process is running.</td>
</tr>
<tr>
<td>NameNode High Availability Health</td>
<td>SCRIPT</td>
<td>This service-level alert is triggered if either the Active NameNode or Standby NameNode are not running.</td>
<td>The Active, Standby or both NameNode processes are down.</td>
<td>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.</td>
</tr>
</tbody>
</table>
## 8.5.3. NameNode HA Alerts

<table>
<thead>
<tr>
<th>Alert</th>
<th>Alert Type</th>
<th>Description</th>
<th>Potential Causes</th>
<th>Possible Remedies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alert</td>
<td>Alert Type</td>
<td>Description</td>
<td>Potential Causes</td>
<td>Possible Remedies</td>
</tr>
<tr>
<td>JournalNode Process</td>
<td>WEB</td>
<td>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.</td>
<td>The JournalNode process is down or not responding. The JournalNode is not down but is not listening to the correct network port/address.</td>
<td>Check if the JournalNode process is running.</td>
</tr>
<tr>
<td>NameNode High Availability Health</td>
<td>SCRIPT</td>
<td>This service-level alert is triggered if either the Active NameNode or Standby NameNode are not running.</td>
<td>The Active, Standby or both NameNode processes are down.</td>
<td>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 <code>netstat-tulpn</code> command to check if the NameNode process is bound to the correct network port.</td>
</tr>
<tr>
<td>Percent JournalNodes Available</td>
<td>AGGREGATE</td>
<td>This service-level alert is triggered if the number of down JournalNodes in the cluster is greater than the configured critical threshold (33% warn, 50% crit ). It aggregates the results of JournalNode process checks.</td>
<td>JournalNodes are down. JournalNodes are not down but are not listening to the correct network port/address.</td>
<td>Check for non-operating JournalNodes in Ambari Web.</td>
</tr>
<tr>
<td>ZooKeeper Failover Controller Process</td>
<td>PORT</td>
<td>This alert is triggered if the ZooKeeper Failover Controller process cannot be confirmed to be up and listening on the network.</td>
<td>The ZKFC process is down or not responding.</td>
<td>Check if the ZKFC process is running.</td>
</tr>
</tbody>
</table>
## 8.5.4. YARN Alerts

<table>
<thead>
<tr>
<th>Alert</th>
<th>Alert Type</th>
<th>Description</th>
<th>Potential Causes</th>
<th>Possible Remedies</th>
</tr>
</thead>
</table>
| App Timeline Web UI    | WEB        | This host-level alert is triggered if the App Timeline Server Web UI is unreachable. | The App Timeline Server is down.  
App Timeline Service is not down but is not listening to the correct network port/address. | Check for non-operating App Timeline Server in Ambari Web.                        |
| Percent NodeManagers Available | AGGREGATE | This alert is triggered if the number of down NodeManagers in the cluster is greater than the configured critical threshold. It aggregates the results of DataNode process alert checks. | NodeManagers are down.  
NodeManagers are not down but are not listening to the correct network port/address. | Check for non-operating 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. |
| ResourceManager Web UI | WEB        | This host-level alert is triggered if the ResourceManager Web UI is unreachable. | The ResourceManager process is not running. | Check if the ResourceManager process is running.                                   |
| ResourceManager RPC Latency | METRIC   | 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. | A job or an application is performing too many ResourceManager operations. | Review the job or the application for potential bugs causing it to perform too many ResourceManager operations. |
| ResourceManager CPU Utilization | METRIC | This host-level alert is triggered if CPU utilization of the ResourceManager exceeds certain thresholds (200% warning, 250% critical). It checks the ResourceManager JMX Servlet for the SystemCPULoad property. This information is only available if you are running JDK 1.7. | 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. | Use the `top` command to determine which processes are consuming excess CPU.  
Reset the offending process. |
| NodeManager Web UI      | WEB        | 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. | NodeManager process is down or not responding.  
NodeManager is not down but is not listening to the correct network port/address. | Check if the NodeManager is running.  
Check for any errors in the NodeManager logs (/var/log/hadoop/yarn) and restart the NodeManager, if necessary. |
| NodeManager Health Summary | SCRIPT   | This host-level alert checks the node health property available from the NodeManager component. | NodeManager Health Check script reports issues or is not configured. | Check in the NodeManager logs (/var/log/hadoop/yarn) for health check errors and... |
### 8.5.5. MapReduce2 Alerts

<table>
<thead>
<tr>
<th>Alert</th>
<th>Alert Type</th>
<th>Description</th>
<th>Potential Causes</th>
<th>Possible Remedies</th>
</tr>
</thead>
<tbody>
<tr>
<td>History Server</td>
<td>WEB</td>
<td>This host-level alert is triggered if the HistoryServer Web UI is unreachable.</td>
<td>The HistoryServer process is not running.</td>
<td>Check if the HistoryServer process is running.</td>
</tr>
<tr>
<td>Web UI</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>History Server</td>
<td>METRIC</td>
<td>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.</td>
<td>A job or an application is performing too many HistoryServer operations.</td>
<td>Review the job or the application for potential bugs causing it to perform too many HistoryServer operations.</td>
</tr>
<tr>
<td>RPC latency</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>History Server</td>
<td>METRIC</td>
<td>This host-level alert is triggered if the percent of CPU utilization on the HistoryServer exceeds the configured critical threshold.</td>
<td>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.</td>
<td>Use the <code>top</code> command to determine which processes are consuming excess CPU. Reset the offending process.</td>
</tr>
<tr>
<td>Server CPU</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Utilization</td>
<td>PORT</td>
<td>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.</td>
<td>HistoryServer process is down or not responding. HistoryServer is not down but is not listening to the correct network port/address.</td>
<td>Check the HistoryServer is running. Check for any errors in the HistoryServer logs (/var/log/hadoop/mapred) and restart the HistoryServer, if necessary.</td>
</tr>
<tr>
<td>Process</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 8.5.6. HBase Service Alerts

<table>
<thead>
<tr>
<th>Alert</th>
<th>Description</th>
<th>Potential Causes</th>
<th>Possible Remedies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percent RegionServers</td>
<td>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 aggregates the results of RegionServer process down checks.</td>
<td>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 down because there were problems.</td>
<td>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. If the failure was associated with a particular workload, try to understand the workload better.</td>
</tr>
<tr>
<td>Available</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alert</td>
<td>Description</td>
<td>Potential Causes</td>
<td>Possible Remedies</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>------------------------------------------------------</td>
</tr>
<tr>
<td>Hortonworks Data Platform</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>June 2, 2016</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>8.5.7. Hive Alerts</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>HBase Master Process</strong></td>
<td>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.</td>
<td>The HBase master process is down. The HBase master has shut itself down because there were problems in the dependent services, ZooKeeper or HDFS.</td>
<td>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). Restart the master.</td>
</tr>
<tr>
<td><strong>HBase Master CPU Utilization</strong></td>
<td>This host-level alert is triggered if CPU utilization of the HBase Master exceeds certain thresholds (200% warning, 250% critical). It checks the HBase Master JMX Servlet for the SystemCPULoad property. This information is only available if you are running JDK 1.7.</td>
<td>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.</td>
<td>Use the <code>top</code> command to determine which processes are consuming excess CPU. Reset the offending process.</td>
</tr>
<tr>
<td><strong>RegionServers Health Summary</strong></td>
<td>This service-level alert is triggered if there are unhealthy RegionServers.</td>
<td>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).</td>
<td>Check for dead RegionServer in Ambari Web.</td>
</tr>
<tr>
<td><strong>HBase RegionServer Process</strong></td>
<td>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.</td>
<td>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).</td>
<td>Check for any errors in the logs (/var/log/hbase/) and restart the RegionServer process using Ambari Web. Run the <code>netstat-tuplpn</code> command to check if the RegionServer process is bound to the correct network port.</td>
</tr>
</tbody>
</table>

### 8.5.7. Hive Alerts

<table>
<thead>
<tr>
<th>Alert</th>
<th>Description</th>
<th>Potential Causes</th>
<th>Possible Remedies</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HiveServer2 Process</strong></td>
<td>This host-level alert is triggered if the HiveServer cannot be determined to be up and responding to client requests.</td>
<td>HiveServer2 process is not running. HiveServer2 process is not responding.</td>
<td>Using Ambari Web, check status of HiveServer2 component. Stop and then restart.</td>
</tr>
<tr>
<td><strong>HiveMetastore Process</strong></td>
<td>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.</td>
<td>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.</td>
<td>Using Ambari Web, stop the Hive service and then restart it.</td>
</tr>
<tr>
<td><strong>WebHCat Server Status</strong></td>
<td>This host-level alert is triggered if the WebHCat server cannot be determined to be up and responding to client requests.</td>
<td>The WebHCat server is down. The WebHCat server is hung and not responding.</td>
<td>Restart the WebHCat server using Ambari Web.</td>
</tr>
</tbody>
</table>
8.5.8. Oozie Alerts

<table>
<thead>
<tr>
<th>Alert</th>
<th>Description</th>
<th>Potential Causes</th>
<th>Possible Remedies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oozie Server Web UI</td>
<td>This host-level alert is triggered if the Oozie server Web UI is unreachable.</td>
<td>The Oozie server is down. Oozie Server is not down but is not listening to the correct network port/address.</td>
<td>Check for dead Oozie Server in Ambari Web.</td>
</tr>
<tr>
<td>Oozie Server Status</td>
<td>This host-level alert is triggered if the Oozie server cannot be determined to be up and responding to client requests.</td>
<td>The Oozie server is down. The Oozie server is hung and not responding. The Oozie server is not reachable over the network.</td>
<td>Restart the Oozie service using Ambari Web.</td>
</tr>
</tbody>
</table>

8.5.9. ZooKeeper Alerts

<table>
<thead>
<tr>
<th>Alert</th>
<th>Alert Type</th>
<th>Description</th>
<th>Potential Causes</th>
<th>Possible Remedies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percent ZooKeeper Servers Available</td>
<td>AGGREGATE</td>
<td>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 aggregates the results of ZooKeeper process checks.</td>
<td>The majority of your ZooKeeper servers are down and not responding.</td>
<td>Check the dependent services to make sure they are operating correctly. Check the ZooKeeper logs (/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.</td>
</tr>
<tr>
<td>ZooKeeper Server Process</td>
<td>PORT</td>
<td>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.</td>
<td>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).</td>
<td>Check for any errors in the ZooKeeper logs (/var/log/hbase/) and restart the ZooKeeper process using Ambari Web. Run the <code>netstat-tulpn</code> command to check if the ZooKeeper server process is bound to the correct network port.</td>
</tr>
</tbody>
</table>

8.5.10. Ambari Alerts

<table>
<thead>
<tr>
<th>Alert</th>
<th>Alert Type</th>
<th>Description</th>
<th>Potential Causes</th>
<th>Possible Remedies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host Disk Usage</td>
<td>SCRIPT</td>
<td>This host-level alert is triggered if the amount of free disk space left is low.</td>
<td>The amount of free disk space left is low.</td>
<td>Check host for disk space to free or add more storage.</td>
</tr>
</tbody>
</table>
### Alert

<table>
<thead>
<tr>
<th>Alert Type</th>
<th>Description</th>
<th>Potential Causes</th>
<th>Possible Remedies</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Ambari Agent Heartbeat</strong></td>
<td>This alert is triggered if the server has lost contact with an agent.</td>
<td>Ambari Server host is unreachable from Agent host, Ambari Agent is not running</td>
<td>Check connection from Agent host to Ambari Server, Check Agent is running</td>
</tr>
<tr>
<td><strong>Ambari Server Alerts</strong></td>
<td>This alert is triggered if the server detects that there are alerts which have not run in a timely manner</td>
<td>Agents are not reporting alert status, Agents are not running</td>
<td>Check that all Agents are running and heartbeating</td>
</tr>
<tr>
<td><strong>Ambari Server Performance</strong></td>
<td>This alert is triggered if the Ambari Server detects that there is a potential performance problem with Ambari.</td>
<td>This type of issue can arise for many reasons, but is typically attributed to slow database queries and host resource exhaustion.</td>
<td>Check your Ambari Server database connection and database activity, Check your Ambari Server host for resource exhaustion such as memory.</td>
</tr>
</tbody>
</table>

### 8.5.11. Ambari Metrics Alerts

<table>
<thead>
<tr>
<th>Alert</th>
<th>Description</th>
<th>Potential Causes</th>
<th>Possible Remedies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metrics Collector Process</td>
<td>This alert is triggered if the Metrics Collector cannot be confirmed to be up and listening on the configured port for number of seconds equal to threshold.</td>
<td>The Metrics Collector process is not running.</td>
<td>Check the Metrics Collector is running.</td>
</tr>
<tr>
<td>Metrics Collector – ZooKeeper Server Process</td>
<td>This host-level alert is triggered if the Metrics Collector ZooKeeper Server Process cannot be determined to be up and listening on the network.</td>
<td>The Metrics Collector process is not running.</td>
<td>Check the Metrics Collector is running.</td>
</tr>
<tr>
<td>Metrics Collector – HBase Master Process</td>
<td>This alert is triggered if the Metrics Collector HBase Master Processes cannot be confirmed to be up and listening on the network for the configured critical threshold, given in seconds.</td>
<td>The Metrics Collector process is not running.</td>
<td>Check the Metrics Collector is running.</td>
</tr>
<tr>
<td>Metrics Collector – HBase Master CPU Utilization</td>
<td>This host-level alert is triggered if CPU utilization of the Metrics Collector exceeds certain thresholds.</td>
<td>Unusually high CPU utilization generally the sign of an issue in the daemon configuration.</td>
<td>Tune the Ambari Metrics Collector.</td>
</tr>
<tr>
<td>Metrics Monitor Status</td>
<td>This host-level alert is triggered if the Metrics Monitor process cannot be confirmed to be up and running on the network.</td>
<td>The Metrics Monitor is down.</td>
<td>Check whether the Metrics Monitor is running on the given host.</td>
</tr>
<tr>
<td>Percent Metrics Monitors Available</td>
<td>This is an AGGREGATE alert of the Metrics Monitor Status.</td>
<td>Metrics Monitors are down.</td>
<td>Check the Metrics Monitors are running.</td>
</tr>
<tr>
<td>Metrics Collector - Auto-Restart Status</td>
<td>This alert is triggered if the Metrics Collector has been auto-started for number of times equal to start threshold in a 1 hour timeframe. By default if restarted 2 times in an hour, you will receive a Warning alert. If restarted 4 or more times in an hour, you will receive a Critical alert.</td>
<td>The Metrics Collector is running but is unstable and causing restarts. This could be due to improper tuning.</td>
<td>Tune the Ambari Metrics Collector.</td>
</tr>
</tbody>
</table>
### 8.5.12. SmartSense Alerts

<table>
<thead>
<tr>
<th>Alert</th>
<th>Description</th>
<th>Potential Causes</th>
<th>Possible Remedies</th>
</tr>
</thead>
<tbody>
<tr>
<td>SmartSense Server Process</td>
<td>This alert is triggered if the HST server process cannot be confirmed to be up and listening on the network for the configured critical threshold, given in seconds.</td>
<td>HST server is not running.</td>
<td>Start HST server process. If startup fails, check the hst-server.log.</td>
</tr>
<tr>
<td>SmartSense Bundle Capture Failure</td>
<td>This alert is triggered if the last triggered SmartSense bundle is failed or timed out.</td>
<td>Some nodes are timed out during capture or fail during data capture. It could also be because upload to Hortonworks fails.</td>
<td>From the &quot;Bundles&quot; page check the status of bundle. Next, check which agents have failed or timed out, and review their logs. You can also initiate a new capture.</td>
</tr>
<tr>
<td>SmartSense Long Running Bundle</td>
<td>This alert is triggered if the SmartSense in-progress bundle has possibility of not completing successfully on time.</td>
<td>Service components that are getting collected may not be running. Or some agents may be timing out during data collection/upload.</td>
<td>Restart the services that are not running. Force-complete the bundle and start a new capture.</td>
</tr>
<tr>
<td>SmartSense Gateway Status</td>
<td>This alert is triggered if the SmartSense Gateway server process is enabled but is unable to reach.</td>
<td>SmartSense Gateway is not running.</td>
<td>Start the gateway. If gateway start fails, review hst-gateway.log</td>
</tr>
</tbody>
</table>

### 8.6. Managing Notifications

Using alert groups and notifications enables you to create groups of alerts and set up notification targets for each group in such a way that you can notify different parties interested in certain sets of alerts by using different methods. For example, you might want your Hadoop Operations team to receive all alerts by email, regardless of status, while at the same time you want your System Administration team to receive only RPC and CPU-related alerts that are in Critical state, and only by simple network management protocol (SNMP).

To achieve these different results, you can have one alert notification that manages email for all alert groups for all severity levels, and a different alert notification group that manages SNMP on critical-severity alerts for an alert group that contains the RPC and CPU alerts.

### 8.7. Creating and Editing Notifications

To create or edit alert notifications:

**Steps**
1. In Ambari Web, click **Alerts**.

2. On the **Alerts** page, click the **Actions** menu, then click **Manage Notifications**.

3. In **Manage Alert Notifications**, click + to create a new alert notification.

   In **Create Alert Notification**,
   - In **Name**, enter a name for the notification
   - In **Groups**, click All or Custom to assign the notification to every or set of groups that you specify
   - In **Description**, type a phrase that describes the notification
   - In **Method**, click EMAIL, SNMP (for MIB-based) or Custom SNMP as the method by which Ambari server handles delivery of this notification.

4. Complete the fields for the notification method you selected.

   - For email notification, provide information about your SMTP infrastructure, such as SMTP server, port, to and from addresses, and whether authentication is required to relay messages through the server.
     
     You can add custom properties to the SMTP configuration based on Javamail SMTP options.

     **Email To** A comma-separated list of one or more email addresses to which to send the alert email
     
     **SMTP Server** The FQDN or IP address of the SMTP server to use to relay the alert email
     
     **SMTP Port** The SMTP port on the SMTP server
     
     **Email From** A single email address to be the originator of the alert email
     
     **Use Authentication** Determine whether your SMTP server requires authentication before it can relay messages. Be sure to also provide the username and password credentials

   - For MIB-based SNMP notification, provide the version, community, host, and port to which the SNMP trap should be sent.:

     **Version** SNMPv1 or SNMPv2c, depending on the network environment
     
     **Hosts** A comma-separated list of one or more host FQDNs to which to send the trap
     
     **Port** The port on which a process is listening for SNMP traps

For SNMP notifications, Ambari uses a "MIB", a text file manifest of alert definitions, to transfer alert information from cluster operations to the alerting infrastructure. A MIB summarizes how object IDs map to objects or attributes.
For example, MIB file content looks like this:

```
objectType OBJECT-TYPE  
SYNTAX AlertEntry  
MAX-ACCESS not-accessible  
STATUS current  
DESCRIPTION "Each Alert Event"  
INDEX { alertDefinitionId }  
 ::= { apacheAmbraiAlertTable 1 }  

alertDefinitionId OBJECT-TYPE  
SYNTAX Integer32 (-2147483648..2147483647)  
MAX-ACCESS not-accessible  
STATUS current  
DESCRIPTION "ID of the Alert"  
 ::= { apacheAmbraiAlertEntry 1 }  

alertDefinitionName OBJECT-TYPE  
SYNTAX OCTET STRING  
MAX-ACCESS accessible-for-notify  
STATUS current  
DESCRIPTION "Alert Definition Name"  
 ::= { apacheAmbraiAlertEntry 2 }  
```

You can find the MIB file for your cluster on the Ambari Server host, at:

```
/var/lib/ambari-server/resources/APACHE-AMBARI-MIB.txt
```

- For Custom SNMP notification, provide the version, community, host, and port to which the SNMP trap should be sent.

Also, the OID parameter must be configured properly for SNMP trap context. If no custom, enterprise-specific OID is used, you should use the following:

- **Version**: SNMPv1 or SNMPv2c, depending on the network environment
- **OID**: 1.3.6.1.4.1.18060.16.1.1
- **Hosts**: A comma-separated list of one or more host FQDNs to which to send the trap
- **Port**: The port on which a process is listening for SNMP traps

5. Click **Save**.

**More Information**

- **Managing Notifications [112]**
- **Javamail SMTP options**

### 8.8. Creating or Editing Alert Groups

To create or edit alert groups:
Steps

1. In Ambari Web, click Alerts.

2. On the Alerts page, click the Actions menu, then click Manage Alert Groups.

3. In Manage Alert Groups, click + to create a new alert notification.

4. In, Create Alert Group, enter a group name and click Save.

5. By clicking on the custom group in the list, you can add or delete alert definitions from this group, and change the notification targets for the group.

6. When you finish your assignments, click Save.

8.9. Dispatching Notifications

When an alert is enabled and the alert status changes (for example, from OK to CRITICAL or CRITICAL to OK), Ambari sends either an email or SNMP notification, depending on how notifications are configured.

For email notifications, Ambari sends an email digest that includes all alert status changes. For example, if two alerts become critical, Ambari sends one email message that Alert A is CRITICAL and Ambari B alert is CRITICAL. Ambari does not send another email notification until status changes again.

For SNMP notifications, Ambari sends one SNMP trap per alert status change. For example, if two alerts become critical, Ambari sends two SNMP traps, one for each alert, and then sends two more when the two alerts change.

8.10. Viewing the Alert Status Log

Whether or not Ambari is configured to send alert notifications, it writes alert status changes to a log on the Ambari Server host. To view this log:

Steps

1. On the Ambari server host, browse to the log directory:

   cd /var/log/ambari-server/

2. View the ambari-alerts.log file.

3. Log entries include the time of the status change, the alert status, the alert definition name, and the response text:

   2015-08-10 22:47:37,120 [OK] [HARD] [STORM] (Storm Server Process) TCP OK - 0.000s response on port 8744
   2015-08-11 11:06:18,479 [CRITICAL] [HARD] [AMBARI] [ambari_server_agent_heartbeat] (Ambari Agent Heartbeat) c6401.ambari.apache.org is not sending heartbeats
   2015-08-11 11:08:18,481 [OK] [HARD] [AMBARI] [ambari_server_agent_heartbeat] (Ambari Agent Heartbeat) c6401.ambari.apache.org is healthy
8.10.1. Customizing Notification Templates

The notification template content produced by Ambari is tightly coupled to a notification type. Email and SNMP notifications both have customizable templates that you can use to generate content. This section describes the steps necessary to change the template used by Ambari when creating alert notifications.

Alert Templates XML Location

By default, an alert-templates.xml ships with Ambari. This file contains all of the templates for every known type of notification (for example, EMAIL and SNMP). This file is bundled in the Ambari server .jar file so that the template is not exposed on the disk; however, that file is used in the following text, as a reference example.

When you customize the alert template, you are effectively overriding the default alert template's XML, as follows:

1. On the Ambari server host, browse to /etc/ambari-server/conf directory.
2. Edit the ambari.properties file.
3. Add an entry for the location of your new template:

   alerts.template.file=/foo/var/alert-templates-custom.xml

4. Save the file and restart Ambari Server.

After you restart Ambari, any notification types defined in the new template override those bundled with Ambari. If you choose to provide your own template file, you only need to define notification templates for the types that you wish to override. If a notification template type is not found in the customized template, Ambari will default to the templates that ship with the JAR.

Alert Templates XML Structure

The structure of the template file is defined as follows. Each <alert-template> element declares what type of alert notification it should be used for.

```xml
<alert-templates>
  <alert-template type="EMAIL">
    <subject>
      Subject Content
    </subject>
    <body>
      Body Content
    </body>
  </alert-template>
  <alert-template type="SNMP">
    <subject>
      Subject Content
    </subject>
    <body>
      Body Content
    </body>
  </alert-template>
</alert-templates>
```
Template Variables

The template uses Apache Velocity to render all tokenized content. The following variables are available for use in your template:

- `$alert.getAlertDefinition()` The definition of which the alert is an instance.
- `$alert.getAlertText()` The specific alert text.
- `$alert.getAlertName()` The name of the alert.
- `$alert.getAlertState()` The alert state (OK, WARNING, CRITICAL, or UNKNOWN).
- `$alert.getServiceName()` The name of the service that the alert is defined for.
- `$alert.hasComponentName()` True if the alert is for a specific service component.
- `$alert.getComponentName()` The component, if any, that the alert is defined for.
- `$alert.hasHostName()` True if the alert was triggered for a specific host.
- `$alert.getHostName()` The hostname, if any, that the alert was triggered for.
- `$ambari.getServerUrl()` The Ambari Server URL.
- `$ambari.getServerVersion()` The Ambari Server version.
- `$ambari.getServerHostName()` The Ambari Server hostname.
- `$dispatch.getTargetName()` The notification target name.
- `$dispatch.getTargetDescription()` The notification target description.
- `$summary.getAlerts(service,alertState)` A list of all alerts for a given service or alert state (OK | WARNING | CRITICAL | UNKNOWN).
- `$summary.getServicesByAlertState(alertState)` A list of all services for a given alert state (OK | WARNING | CRITICAL | UNKNOWN).
- `$summary.getServices()` A list of all services that are reporting an alert in the notification.
- `$summary.getCriticalCount()` The CRITICAL alert count.
- `$summary.getOkCount()` The OK alert count.
- `$summary.getTotalCount()` The total alert count.
- `$summary.getUnknownCount()` The UNKNOWN alert count.
- `$summary.getWarningCount()` The WARNING alert count.
- `$summary.getAlerts()` A list of all of the alerts in the notification.

Example: Modify Alert EMAIL Subject
The following example illustrates how to change the subject line of all outbound email notifications to include a hard-coded identifier:

1. Download the alert-templates.xml code as your starting point.

2. On the Ambari Server, save the template to a location such as /var/lib/ambari-server/resources/alert-templates-custom.xml.

3. Edit the alert-templates-custom.xml file and modify the subject link for the <alert-template type="EMAIL"> template:

   ```xml
   <subject>
     <![CDATA[Petstore Ambari has $summary.getTotalCount() alerts!]]>
   </subject>
   ```

4. Save the file.

5. Browse to /etc/ambari-server/conf directory.

6. Edit the ambari.properties file.

7. Add an entry for the location of your new template file.

   ```
   alerts.template.file=/var/lib/ambari-server/resources/alert-templates-custom.xml
   ```

8. Save the file and restart Ambari Server.
9. Using Ambari Core Services

The Ambari core services enable you to monitor, analyze, and search the operating status of hosts in your cluster. This chapter describes how to use and configure the following Ambari Core Services:

- Understanding Ambari Metrics [119]
- Ambari Log Search (Technical Preview) [175]
- Ambari Infra [179]

9.1. Understanding Ambari Metrics

Ambari Metrics System (AMS) collects, aggregates, and serves Hadoop and system metrics in Ambari-managed clusters.

- AMS Architecture [119]
- Using Grafana [120]
- Grafana Dashboards Reference [125]
- AMS Performance Tuning [163]
- AMS High Availability (Technical Preview) [168]

9.1.1. AMS Architecture

AMS has four components: Metrics Monitors, Hadoop Sinks, Metrics Collector, and Grafana.

- Metrics Monitors on each host in the cluster collect system-level metrics and publish to the Metrics Collector.

- Hadoop Sinks plug in to Hadoop components to publish Hadoop metrics to the Metrics Collector.

- The Metrics Collector is a daemon that runs on a specific host in the cluster and receives data from the registered publishers, the Monitors, and the Sinks.

- Grafana is a daemon that runs on a specific host in the cluster and serves pre-built dashboards for visualizing metrics collected in the Metrics Collector.

The following high-level illustration shows how the components of AMS work together to collect metrics and make those metrics available to Ambari.
9.1.2. Using Grafana

Ambari Metrics System includes Grafana with pre-built dashboards for advanced visualization of cluster metrics.

- Accessing Grafana [120]
- Viewing Grafana Dashboards [121]
- Viewing Selected Metrics on Grafana Dashboards [123]
- Viewing Metrics for Selected Hosts [124]

More Information

http://grafana.org/

9.1.2.1. Accessing Grafana

To access the Grafana UI:

Steps

1. In Ambari Web, browse to Services > Ambari Metrics > Summary.
2. Select Quick Links and then choose Grafana.

A read-only version of the Grafana interface opens in a new tab in your browser:
9.1.2.2. Viewing Grafana Dashboards

On the Grafana home page, **Dashboards** provides a short list of links to AMS, Ambari server, Druid and HBase metrics.

To view specific metrics included in the list:

**Steps**

1. In Grafana, browse to **Dashboards**.
2. Click a dashboard name.
3. To see more available dashboards, click the **Home** list.
4. Scroll down to view the whole list.

1. Click a dashboard name, for example **System - Servers**.

   The **System - Servers** dashboard opens:
9.1.2.3. Viewing Selected Metrics on Grafana Dashboards

On a dashboard, expand one or more rows to view detailed metrics, continuing the previous example using the **System - Servers** dashboard:

1. In the **System - Servers** dashboard, click a row name. For example, click **System Load Average - 1 Minute**.

   The row expands to display a chart that shows metrics information: in this example, the System Load Average - 1 Minute and the System Load Average - 15 Minute rows:
9.1.2.4. Viewing Metrics for Selected Hosts

By default, Grafana shows metrics for all hosts in your cluster. You can limit the displayed
metrics to one or more hosts by selecting them from the Hosts menu:

1. Expand Hosts.

2. Select one or more host names.

A check mark appears next to selected host names:
**Note**

Selections in the **Hosts** menu apply to all metrics in the current dashboard. Grafana refreshes the current dashboards when you select a new set of hosts.

### 9.1.3. Grafana Dashboards Reference

**Ambari Metrics System** includes Grafana with pre-built dashboards for advanced visualization of cluster metrics.

- AMS HBase Dashboards [125]
- Ambari Dashboards [133]
- HDFS Dashboards [135]
- YARN Dashboards [139]
- Hive Dashboards [142]
- Hive LLAP Dashboards [144]
- HBase Dashboards [148]
- Kafka Dashboards [157]
- Storm Dashboards [159]
- System Dashboards [160]
- NiFi Dashboard [162]

#### 9.1.3.1. AMS HBase Dashboards

AMS HBase refers to the HBase instance managed by Ambari Metrics Service independently. It does not have any connection with the cluster HBase service. AMS HBase Grafana dashboards track the same metrics as the regular HBase dashboard, but for the AMS-owned instance.

The following Grafana dashboards are available for AMS HBase:

- **AMS HBase - Home** [126]
- **AMS HBase - RegionServers** [127]
• AMS HBase - Misc [132]

9.1.3.1.1. AMS HBase - Home

The AMS HBase - Home dashboards display basic statistics about an HBase cluster. These dashboards provide insight to the overall status for the HBase cluster.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>REGIONSERVERS / REGIONS</td>
<td>Num RegionServers</td>
<td>Total number of RegionServers in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Num Dead RegionServers</td>
<td>Total number of RegionServers that are dead in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Num Regions</td>
<td>Total number of regions in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Avg Num Regions per RegionServer</td>
<td>Average number of regions per RegionServer.</td>
</tr>
<tr>
<td>NUM REGIONS/STORES</td>
<td>Num Regions / Stores - Total</td>
<td>Total number of regions and stores (column families) in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Store File Size / Count - Total</td>
<td>Total data file size and number of store files.</td>
</tr>
<tr>
<td>NUM REQUESTS</td>
<td>Num Requests - Total</td>
<td>Total number of requests (read, write and RPCs) in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Num Request - Breakdown - Total</td>
<td>Total number of get, put, mutate, etc requests in the cluster.</td>
</tr>
<tr>
<td>REGIONSERVER MEMORY</td>
<td>RegionServer Memory - Average</td>
<td>Average used, max or committed on-heap and offheap memory for RegionServers.</td>
</tr>
<tr>
<td></td>
<td>RegionServer Offheap Memory - Average</td>
<td>Average used, free or committed on-heap and offheap memory for RegionServers.</td>
</tr>
<tr>
<td>MEMORY - MEMSTORE BLOCKCACHE</td>
<td>Memstore - BlockCache - Average</td>
<td>Average blockcache and memstore sizes for RegionServers.</td>
</tr>
<tr>
<td></td>
<td>Num Blocks in BlockCache - Total</td>
<td>Total number of (hfile) blocks in the blockcaches across all RegionServers.</td>
</tr>
<tr>
<td>BLOCKCACHE</td>
<td>BlockCache Hit/ Miss/s Total</td>
<td>Total number of blockcache hits misses and evictions across all RegionServers.</td>
</tr>
<tr>
<td></td>
<td>BlockCache Hit Percent - Average</td>
<td>Average blockcache hit percentage across all RegionServers.</td>
</tr>
<tr>
<td>OPERATION LATENCIES - GET/MUTATE</td>
<td>Get Latencies - Average</td>
<td>Average min, median, max, 75th, 95th, 99th percentile latencies for Get operation across all RegionServers.</td>
</tr>
<tr>
<td></td>
<td>Mutate Latencies - Average</td>
<td>Average min, median, max, 75th, 95th, 99th percentile latencies for Mutate operation across all RegionServers.</td>
</tr>
<tr>
<td>OPERATION LATENCIES - DELETE/INCREMENT</td>
<td>Delete Latencies - Average</td>
<td>Average min, median, max, 75th, 95th, 99th percentile latencies for Delete operation across all RegionServers.</td>
</tr>
<tr>
<td></td>
<td>Increment Latencies - Average</td>
<td>Average min, median, max, 75th, 95th, 99th percentile latencies for Increment operation across all RegionServers.</td>
</tr>
<tr>
<td>OPERATION LATENCIES - APPEND/REPLAY</td>
<td>Append Latencies - Average</td>
<td>Average min, median, max, 75th, 95th, 99th percentile latencies for Append operation across all RegionServers.</td>
</tr>
<tr>
<td></td>
<td>Replay Latencies - Average</td>
<td>Average min, median, max, 75th, 95th, 99th percentile latencies for Replay operation across all RegionServers.</td>
</tr>
<tr>
<td>REGIONSERVER RPC</td>
<td>RegionServer RPC - Average</td>
<td>Average number of RPCs, active handler threads and open connections across all RegionServers.</td>
</tr>
<tr>
<td></td>
<td>RegionServer RPC Queues - Average</td>
<td>Average number of calls in different RPC scheduling queues and the size of all requests in the RPC queue across all RegionServers.</td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
<td>Description</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>REGIONSERVER RPC</td>
<td>RegionServer RPC Throughput - Average</td>
<td>Average sent and received bytes from the RPC across all RegionServers.</td>
</tr>
</tbody>
</table>

### 9.1.3.1.2. AMS HBase - RegionServers

The AMS HBase - RegionServers dashboards display metrics for RegionServers in the monitored HBase cluster, including some performance-related data. These dashboards help you view basic I/O data and compare load among RegionServers.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUM REGIONS</td>
<td>Num Regions</td>
<td>Number of regions in the RegionServer.</td>
</tr>
<tr>
<td>STORE FILES</td>
<td>Store File Size</td>
<td>Total size of the store files (data files) in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Store File Count</td>
<td>Total number of store files in the RegionServer.</td>
</tr>
<tr>
<td>NUM REQUESTS</td>
<td>Num Total Requests /s</td>
<td>Total number of requests (both read and write) per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Write Requests /s</td>
<td>Total number of write requests per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Read Requests /s</td>
<td>Total number of read requests per second in the RegionServer.</td>
</tr>
<tr>
<td>NUM REQUESTS - GET / SCAN</td>
<td>Num Get Requests /s</td>
<td>Total number of Get requests per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Scan Next Requests /s</td>
<td>Total number of Scan requests per second in the RegionServer.</td>
</tr>
<tr>
<td>NUM REQUESTS - MUTATE / DELETE</td>
<td>Num Mutate Requests /s</td>
<td>Total number of Mutate requests per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Delete Requests /s</td>
<td>Total number of Delete requests per second in the RegionServer.</td>
</tr>
<tr>
<td>NUM REQUESTS - APPEND / INCREMENT</td>
<td>Num Append Requests /s</td>
<td>Total number of Append requests per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Increment Requests /s</td>
<td>Total number of Increment requests per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Replay Requests /s</td>
<td>Total number of Replay requests per second in the RegionServer.</td>
</tr>
<tr>
<td>MEMORY</td>
<td>RegionServer Memory Used</td>
<td>Heap Memory used by the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Offheap Memory Used</td>
<td>Offheap Memory used by the RegionServer.</td>
</tr>
<tr>
<td>MEMSTORE</td>
<td>Memstore Size</td>
<td>Total Memstore memory size of the RegionServer.</td>
</tr>
<tr>
<td>BLOCKCACHE - OVERVIEW</td>
<td>BlockCache - Size</td>
<td>Total BlockCache size of the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>BlockCache - Free Size</td>
<td>Total free space in the BlockCache of the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Blocks in Cache</td>
<td>Total number of hfile blocks in the BlockCache of the RegionServer.</td>
</tr>
<tr>
<td>BLOCKCACHE - HITS/MISSES</td>
<td>Num BlockCache Hits /s</td>
<td>Number of BlockCache hits per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num BlockCache Misses /s</td>
<td>Number of BlockCache misses per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num BlockCache Evictions /s</td>
<td>Number of BlockCache evictions per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>BlockCache Caching Hit Percent</td>
<td>Percentage of BlockCache hits per second for requests that requested cache blocks in the RegionServer.</td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
<td>Description</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>BlockCache Hit Percent</td>
<td>Percentage of BlockCache hits per second in the RegionServer.</td>
</tr>
<tr>
<td>OPERATION LATENCIES - GET</td>
<td>Get Latencies - Mean</td>
<td>Mean latency for Get operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Get Latencies - Median</td>
<td>Median latency for Get operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Get Latencies - 75th Percentile</td>
<td>75th percentile latency for Get operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Get Latencies - 95th Percentile</td>
<td>95th percentile latency for Get operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Get Latencies - 99th Percentile</td>
<td>99th percentile latency for Get operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Get Latencies - Max</td>
<td>Max latency for Get operation in the RegionServer.</td>
</tr>
<tr>
<td>OPERATION LATENCIES - SCAN NEXT</td>
<td>Scan Next Latencies - Mean</td>
<td>Mean latency for Scan operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Scan Next Latencies - Median</td>
<td>Median latency for Scan operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Scan Next Latencies - 75th Percentile</td>
<td>75th percentile latency for Scan operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Scan Next Latencies - 95th Percentile</td>
<td>95th percentile latency for Scan operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Scan Next Latencies - 99th Percentile</td>
<td>99th percentile latency for Scan operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Scan Next Latencies - Max</td>
<td>Max latency for Scan operation in the RegionServer.</td>
</tr>
<tr>
<td>OPERATION LATENCIES - MUTATE</td>
<td>Mutate Latencies - Mean</td>
<td>Mean latency for Mutate operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Mutate Latencies - Median</td>
<td>Median latency for Mutate operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Mutate Latencies - 75th Percentile</td>
<td>75th percentile latency for Mutate operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Mutate Latencies - 95th Percentile</td>
<td>95th percentile latency for Mutate operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Mutate Latencies - 99th Percentile</td>
<td>99th percentile latency for Mutate operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Mutate Latencies - Max</td>
<td>Max latency for Mutate operation in the RegionServer.</td>
</tr>
<tr>
<td>OPERATION LATENCIES - DELETE</td>
<td>Delete Latencies - Mean</td>
<td>Mean latency for Delete operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Delete Latencies - Median</td>
<td>Median latency for Delete operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Delete Latencies - 75th Percentile</td>
<td>75th percentile latency for Delete operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Delete Latencies - 95th Percentile</td>
<td>95th percentile latency for Delete operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Delete Latencies - 99th Percentile</td>
<td>99th percentile latency for Delete operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Delete Latencies - Max</td>
<td>Max latency for Delete operation in the RegionServer.</td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
<td>Description</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>Increment Latencies - Mean</td>
<td>Mean latency for Increment operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Increment Latencies - Median</td>
<td>Median latency for Increment operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Increment Latencies - 75th Percentile</td>
<td>75th percentile latency for Increment operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Increment Latencies - 95th Percentile</td>
<td>95th percentile latency for Increment operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Increment Latencies - 99th Percentile</td>
<td>99th percentile latency for Increment operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Increment Latencies - Max</td>
<td>Max latency for Increment operation in the RegionServer.</td>
</tr>
<tr>
<td>OPERATION LATENCIES - INCREMENT</td>
<td>Append Latencies - Mean</td>
<td>Mean latency for Append operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Append Latencies - Median</td>
<td>Median latency for Append operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Append Latencies - 75th Percentile</td>
<td>75th percentile latency for Append operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Append Latencies - 95th Percentile</td>
<td>95th percentile latency for Append operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Append Latencies - 99th Percentile</td>
<td>99th percentile latency for Append operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Append Latencies - Max</td>
<td>Max latency for Append operation in the RegionServer.</td>
</tr>
<tr>
<td>OPERATION LATENCIES - APPEND</td>
<td>Replay Latencies - Mean</td>
<td>Mean latency for Replay operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Replay Latencies - Median</td>
<td>Median latency for Replay operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Replay Latencies - 75th Percentile</td>
<td>75th percentile latency for Replay operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Replay Latencies - 95th Percentile</td>
<td>95th percentile latency for Replay operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Replay Latencies - 99th Percentile</td>
<td>99th percentile latency for Replay operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Replay Latencies - Max</td>
<td>Max latency for Replay operation in the RegionServer.</td>
</tr>
<tr>
<td>OPERATION LATENCIES - REPLAY</td>
<td>Num RPC /s</td>
<td>Number of RPCs per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Active Handler Threads</td>
<td>Number of active RPC handler threads (to process requests) in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Connections</td>
<td>Number of connections to the RegionServer.</td>
</tr>
<tr>
<td>RPC - OVERVIEW</td>
<td>Num RPC Calls in General Queue</td>
<td>Number of RPC calls in the general processing queue in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num RPC Calls in Priority Queue</td>
<td>Number of RPC calls in the high priority (for system tables) processing queue in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num RPC Calls in Replication Queue</td>
<td>Number of RPC calls in the replication processing queue in the RegionServer.</td>
</tr>
<tr>
<td>RPC - QUEUES</td>
<td>RPC - Total Call Queue Size</td>
<td>Total data size of all RPC calls in the RPC queues in the RegionServer.</td>
</tr>
<tr>
<td>RPC - CALL QUEUED TIMES</td>
<td>RPC - Call Queued Time - Mean</td>
<td>Mean latency for RPC calls to stay in the RPC queue in the RegionServer.</td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------</td>
<td>---------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>RPC - Call Queued Time - Median</td>
<td>Median latency for RPC calls to stay in the RPC queue in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RPC - Call Queued Time - 75th Percentile</td>
<td>75th percentile latency for RPC calls to stay in the RPC queue in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RPC - Call Queued Time - 95th Percentile</td>
<td>95th percentile latency for RPC calls to stay in the RPC queue in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RPC - Call Queued Time - 99th Percentile</td>
<td>99th percentile latency for RPC calls to stay in the RPC queue in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RPC - Call Queued Time - Max</td>
<td>Max latency for RPC calls to stay in the RPC queue in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RPC - Call Process Time - Mean</td>
<td>Mean latency for RPC calls to be processed in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RPC - Call Process Time - Median</td>
<td>Median latency for RPC calls to be processed in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RPC - Call Process Time - 75th Percentile</td>
<td>75th percentile latency for RPC calls to be processed in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RPC - Call Process Time - 95th Percentile</td>
<td>95th percentile latency for RPC calls to be processed in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RPC - Call Process Time - 99th Percentile</td>
<td>99th percentile latency for RPC calls to be processed in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RPC - Call Process Time - Max</td>
<td>Max latency for RPC calls to be processed in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RPC - Received bytes /s</td>
<td>Received bytes from the RPC in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RPC - Sent bytes /s</td>
<td>Sent bytes from the RPC in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num WAL - Files</td>
<td>Number of Write-Ahead-Log files in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Total WAL File Size</td>
<td>Total files sized of Write-Ahead-Logs in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Num Appends /s</td>
<td>Number of append operations per second to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Num Sync /s</td>
<td>Number of sync operations per second to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Sync Latencies - Mean</td>
<td>Mean latency for Write-Ahead-Log sync operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Sync Latencies - Median</td>
<td>Median latency for Write-Ahead-Log sync operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Sync Latencies - 75th Percentile</td>
<td>75th percentile latency for Write-Ahead-Log sync operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Sync Latencies - 95th Percentile</td>
<td>95th percentile latency for Write-Ahead-Log sync operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Sync Latencies - 99th Percentile</td>
<td>99th percentile latency for Write-Ahead-Log sync operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Sync Latencies - Max</td>
<td>Max latency for Write-Ahead-Log sync operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Latencies - Mean</td>
<td>Mean latency for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Latencies - Median</td>
<td>Median latency for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Latencies - 75th Percentile</td>
<td>95th percentile latency for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Latencies - 95th Percentile</td>
<td>99th percentile latency for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Latencies - 99th Percentile</td>
<td>99th percentile latency for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Latencies - Max</td>
<td>Max latency for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Sizes - Mean</td>
<td>Mean data size for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Sizes - Median</td>
<td>Median data size for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Sizes - 75th Percentile</td>
<td>75th percentile data size for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Sizes - 95th Percentile</td>
<td>95th percentile data size for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Sizes - 99th Percentile</td>
<td>99th percentile data size for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Sizes - Max</td>
<td>Max data size for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL Num Slow Append /s</td>
<td>Number of append operations per second to the filesystem that took more than 1 second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Slow Gets /s</td>
<td>Number of Get requests per second that took more than 1 second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Slow Puts /s</td>
<td>Number of Put requests per second that took more than 1 second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Slow Deletes /s</td>
<td>Number of Delete requests per second that took more than 1 second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Flush Queue Length</td>
<td>Number of Flush operations waiting to be processed in the RegionServer. A higher number indicates flush operations being slow.</td>
</tr>
<tr>
<td></td>
<td>Compaction Queue Length</td>
<td>Number of Compaction operations waiting to be processed in the RegionServer. A higher number indicates compaction operations being slow.</td>
</tr>
<tr>
<td></td>
<td>Split Queue Length</td>
<td>Number of Region Split operations waiting to be processed in the RegionServer. A higher number indicates split operations being slow.</td>
</tr>
<tr>
<td></td>
<td>GC Count /s</td>
<td>Number of Java Garbage Collections per second.</td>
</tr>
<tr>
<td></td>
<td>GC Count ParNew /s</td>
<td>Number of Java ParNew (YoungGen) Garbage Collections per second.</td>
</tr>
<tr>
<td></td>
<td>GC Count CMS /s</td>
<td>Number of Java CMS Garbage Collections per second.</td>
</tr>
<tr>
<td></td>
<td>GC Times /s</td>
<td>Total time spend in Java Garbage Collections per second.</td>
</tr>
<tr>
<td></td>
<td>GC Times ParNew /s</td>
<td>Total time spend in Java ParNew(YoungGen) Garbage Collections per second.</td>
</tr>
<tr>
<td></td>
<td>GC Times CMS /s</td>
<td>Total time spend in Java CMS Garbage Collections per second.</td>
</tr>
</tbody>
</table>
### LOCALITY

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percent Files Local</td>
<td>Percentage of files served from the local DataNode for the RegionServer.</td>
</tr>
</tbody>
</table>

### 9.1.3.1.3. AMS HBase - Misc

The AMS HBase - Misc dashboards display miscellaneous metrics related to the HBase cluster. You can use these metrics for tasks like debugging authentication and authorization issues and exceptions raised by RegionServers.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>REGIONS IN TRANSITION</td>
<td>Master - Regions in Transition</td>
<td>Number of regions in transition in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Master - Regions in Transition Longer Than Threshold Time</td>
<td>Number of regions in transition that are in transition state for longer than 1 minute in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Regions in Transition Oldest Age</td>
<td>Maximum time that a region stayed in transition state.</td>
</tr>
<tr>
<td>NUM THREADS - RUNNABLE</td>
<td>Master Num Threads - Runnable</td>
<td>Number of threads in the Master.</td>
</tr>
<tr>
<td></td>
<td>RegionServer Num Threads - Runnable</td>
<td>Number of threads in the RegionServer.</td>
</tr>
<tr>
<td>NUM THREADS - BLOCKED</td>
<td>Master Num Threads - Blocked</td>
<td>Number of threads in the Blocked State in the Master.</td>
</tr>
<tr>
<td></td>
<td>RegionServer Num Threads - Blocked</td>
<td>Number of threads in the Blocked State in the RegionServer.</td>
</tr>
<tr>
<td>NUM THREADS - WAITING</td>
<td>Master Num Threads - Waiting</td>
<td>Number of threads in the Waiting State in the Master.</td>
</tr>
<tr>
<td></td>
<td>RegionServer Num Threads - Waiting</td>
<td>Number of threads in the Waiting State in the RegionServer.</td>
</tr>
<tr>
<td>NUM THREADS - TIMED WAITING</td>
<td>Master Num Threads - Timed Waiting</td>
<td>Number of threads in the Timed-Waiting State in the Master.</td>
</tr>
<tr>
<td></td>
<td>RegionServer Num Threads - Timed Waiting</td>
<td>Number of threads in the Timed-Waiting State in the RegionServer.</td>
</tr>
<tr>
<td>NUM THREADS - NEW</td>
<td>Master Num Threads - New</td>
<td>Number of threads in the New State in the Master.</td>
</tr>
<tr>
<td></td>
<td>RegionServer Num Threads - New</td>
<td>Number of threads in the New State in the RegionServer.</td>
</tr>
<tr>
<td>NUM THREADS - TERMINATED</td>
<td>Master Num Threads - Terminated</td>
<td>Number of threads in the Terminated State in the Master.</td>
</tr>
<tr>
<td></td>
<td>RegionServer Num Threads - Terminated</td>
<td>Number of threads in the Terminated State in the RegionServer.</td>
</tr>
<tr>
<td>RPC AUTHENTICATION</td>
<td>RegionServer RPC Authentication Successes /s</td>
<td>Number of RPC successful authentications per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RegionServer RPC Authentication Failures /s</td>
<td>Number of RPC failed authentications per second in the RegionServer.</td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RPC Authorization</td>
<td>RegionServer RPC Authorization Successes /s</td>
<td>Number of RPC successful authorizations per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RegionServer RPC Authorization Failures /s</td>
<td>Number of RPC failed authorizations per second in the RegionServer.</td>
</tr>
<tr>
<td>EXCEPTIONS</td>
<td>Master Exceptions /s</td>
<td>Number of exceptions in the Master.</td>
</tr>
<tr>
<td></td>
<td>RegionServer Exceptions /s</td>
<td>Number of exceptions in the RegionServer.</td>
</tr>
</tbody>
</table>

### 9.1.3.2. Ambari Dashboards

The following Grafana dashboards are available for Ambari:

- Ambari Server Database [133]
- Ambari Server JVM [133]
- Ambari Server Top N [134]

#### 9.1.3.2.1. Ambari Server Database

Metrics that show operating status for the Ambari server database.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOTAL READ ALL QUERY</td>
<td>Total Read All Query Counter (Rate)</td>
<td>Total ReadAllQuery operations performed.</td>
</tr>
<tr>
<td></td>
<td>Total Read All Query Timer (Rate)</td>
<td>Total time spent on ReadAllQuery.</td>
</tr>
<tr>
<td>TOTAL CACHE HITS &amp; MISSES</td>
<td>Total Cache Hits (Rate)</td>
<td>Total cache hits on Ambari Server with respect to EclipseLink cache.</td>
</tr>
<tr>
<td></td>
<td>Total Cache Misses (Rate)</td>
<td>Total cache misses on Ambari Server with respect to EclipseLink cache.</td>
</tr>
<tr>
<td>QUERY</td>
<td>Query Stages Timings</td>
<td>Average time spent on every query sub stage by Ambari Server</td>
</tr>
<tr>
<td></td>
<td>Query Types Avg. Timings</td>
<td>Average time spent on every query type by Ambari Server.</td>
</tr>
<tr>
<td>HOST ROLE COMMAND ENTITY</td>
<td>Counter.ReadAllQuery.HostRoleCommandEntity (Rate)</td>
<td>Rate (num operations per second) in which ReadAllQuery operation on HostRoleCommandEntity is performed.</td>
</tr>
<tr>
<td></td>
<td>Timer.ReadAllQuery.HostRoleCommandEntity (Rate)</td>
<td>Rate in which ReadAllQuery operation on HostRoleCommandEntity is performed.</td>
</tr>
<tr>
<td></td>
<td>ReadAllQuery.HostRoleCommandEntity</td>
<td>Average time taken for a ReadAllQuery operation on HostRoleCommandEntity (Timer / Counter).</td>
</tr>
</tbody>
</table>

#### 9.1.3.2.2. Ambari Server JVM

Metrics to see status for the Ambari Server Java virtual machine.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JVM - MEMORY PRESSURE</td>
<td>Heap Usage</td>
<td>Used, max or committed on-heap memory for Ambari Server.</td>
</tr>
</tbody>
</table>
### 9.1.3.2.3. Ambari Server Top N

Metrics to see top performing users and operations for Ambari.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>READ ALL QUERY</td>
<td>Top ReadAllQuery Counters</td>
<td>Top N Ambari Server entities by number of ReadAllQuery operations performed.</td>
</tr>
<tr>
<td></td>
<td>Top ReadAllQuery Timers</td>
<td>Top N Ambari Server entities by time spent on ReadAllQuery operations.</td>
</tr>
<tr>
<td>Cache Misses</td>
<td>Cache Misses</td>
<td>Top N Ambari Server entities by number of Cache Misses.</td>
</tr>
</tbody>
</table>

### 9.1.3.3. Druid Dashboards

The following Grafana dashboards are available for Druid:

- **Druid - Home [134]**
- **Druid - Ingestion [135]**
- **Druid - Query [135]**

#### 9.1.3.3.1. Druid - Home

Metrics that show operating status for Druid.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DRUID BROKER</td>
<td>JVM Heap</td>
<td>JVM Heap used by the Druid Broker Node</td>
</tr>
<tr>
<td></td>
<td>JVM GCM Time</td>
<td>Time spent by the Druid Broker Node in JVM Garbage collection</td>
</tr>
<tr>
<td>DRUID HISTORICAL</td>
<td>JVM Heap</td>
<td>JVM Heap used by the Druid Historical Node</td>
</tr>
<tr>
<td></td>
<td>JVM GCM Time</td>
<td>Time spent by the Druid Historical Node in JVM Garbage collection</td>
</tr>
<tr>
<td>DRUID COORDINATER</td>
<td>JVM Heap</td>
<td>JVM Heap used by the Druid Coordinator Node</td>
</tr>
<tr>
<td></td>
<td>JVM GCM Time</td>
<td>Time spent by the Druid Coordinator Node in JVM Garbage collection</td>
</tr>
<tr>
<td>DRUID OVERLORD</td>
<td>JVM Heap</td>
<td>JVM Heap used by the Druid Overlord Node</td>
</tr>
<tr>
<td></td>
<td>JVM GCM Time</td>
<td>Time spent by the Druid Overlord Node in JVM Garbage collection</td>
</tr>
<tr>
<td>DRUID MIDDLEMANAGER</td>
<td>JVM Heap</td>
<td>JVM Heap used by the Druid Middlemanager Node</td>
</tr>
</tbody>
</table>
9.1.3.3.2. Druid - Ingestion

Metrics to see status for Druid data ingestion rates.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>JVM GCM Time</td>
<td>Time spent by the Druid Middlemanager Node in JVM Garbage collection</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INGESTION METRICS</td>
<td>Ingested Events</td>
<td>Number of events ingested on real time nodes</td>
</tr>
<tr>
<td></td>
<td>Events Thrown Away</td>
<td>Number of events rejected because they are outside the windowPeriod.</td>
</tr>
<tr>
<td></td>
<td>Unparseable Events</td>
<td>Number of events rejected because they did not parse</td>
</tr>
<tr>
<td>INTERMEDIATE PERSISTS METRICS</td>
<td>Persisted Rows</td>
<td>Number of Druid rows persisted on disk</td>
</tr>
<tr>
<td></td>
<td>Average Persist Time</td>
<td>Average time taken to persist intermediate segments to disk</td>
</tr>
<tr>
<td></td>
<td>Intermediate Persist Count</td>
<td>Number of times that intermediate segments were persisted</td>
</tr>
<tr>
<td>SEGMENT SIZE METRICS</td>
<td>Ave Segment Size</td>
<td>Average size of added Druid segments</td>
</tr>
<tr>
<td></td>
<td>Total Segment Size</td>
<td>Total size of added Druid segments</td>
</tr>
</tbody>
</table>

9.1.3.3.3. Druid - Query

Metrics to see status of Druid queries.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query Time Metrics</td>
<td>Broker Query Time</td>
<td>Average Time taken by Druid Broker node to process queries</td>
</tr>
<tr>
<td></td>
<td>Historical Query Time</td>
<td>Average time taken by Druid historical nodes to process queries</td>
</tr>
<tr>
<td></td>
<td>Realtime Query Time</td>
<td>Average time taken by Druid real time nodes to process queries</td>
</tr>
<tr>
<td>SEGMENT SCAN METRICS</td>
<td>Historical Segment Scan Time</td>
<td>Average time taken by Druid historical nodes to scan individual segments</td>
</tr>
<tr>
<td></td>
<td>Realtime Segment Scan Time</td>
<td>Average time taken by Druid real time nodes to scan individual segments</td>
</tr>
<tr>
<td></td>
<td>Historical Query Wait Time</td>
<td>Average time spent waiting for a segment to be scanned on historical node</td>
</tr>
<tr>
<td></td>
<td>Realtime Query Wait Time</td>
<td>Average time spent waiting for a segment to be scanned on real time node</td>
</tr>
<tr>
<td></td>
<td>Pending Historical Segment Scans</td>
<td>Average Number of pending segment scans on historical nodes</td>
</tr>
<tr>
<td></td>
<td>Pending Realtime Segment Scans</td>
<td>Average Number of pending segment scans on real time nodes</td>
</tr>
</tbody>
</table>

9.1.3.4. HDFS Dashboards

The following Grafana dashboards are available for Hadoop Distributed File System (HDFS) components:

- HDFS - Home [136]
- HDFS - NameNodes [136]
9.1.3.4.1. HDFS - Home

The HDFS - Home dashboard displays metrics that show operating status for HDFS.

**Note**

In a NameNode HA setup, metrics are collected from and displayed for both the active and the standby NameNode.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUMBER OF FILES UNDER CONSTRUCTION &amp; RPC CLIENT CONNECTIONS</td>
<td>Number of Files Under Construction</td>
<td>Number of HDFS files that are still being written.</td>
</tr>
<tr>
<td></td>
<td>RPC Client Connections</td>
<td>Number of open RPC connections from clients on NameNode(s).</td>
</tr>
<tr>
<td>TOTAL FILE OPERATIONS &amp; CAPACITY USED</td>
<td>Total File Operations</td>
<td>Total number of operations on HDFS files, including file creation/deletion/rename/truncation, directory/file/block information retrieval, and snapshot related operations.</td>
</tr>
<tr>
<td></td>
<td>Capacity Used</td>
<td>&quot;CapacityTotalGB&quot; shows total HDFS storage capacity, in GB. &quot;CapacityUsedGB&quot; indicates total used HDFS storage capacity, in GB.</td>
</tr>
<tr>
<td>RPC CLIENT PORT SLOW CALLS &amp; HDFS TOTAL LOAD</td>
<td>RPC Client Port Slow Calls</td>
<td>Number of slow RPC requests on NameNode. A &quot;slow&quot; RPC request is one that takes more time to complete than 99.7% of other requests.</td>
</tr>
<tr>
<td></td>
<td>HDFS Total Load</td>
<td>Total number of connections on all the DataNodes sending/receiving data.</td>
</tr>
<tr>
<td>ADD BLOCK STATUS</td>
<td>Add Block Time</td>
<td>The average time (in ms) serving addBlock RPC request on NameNode(s).</td>
</tr>
<tr>
<td></td>
<td>Add Block Num Ops</td>
<td>The rate of addBlock RPC requests on NameNode(s).</td>
</tr>
</tbody>
</table>

9.1.3.4.2. HDFS - NameNodes

Metrics to see status for the NameNodes.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPC CLIENT QUEUE TIME</td>
<td>RPC Client Port Queue Time</td>
<td>Average time that a RPC request (on the RPC port facing to the HDFS clients) waits in the queue.</td>
</tr>
<tr>
<td></td>
<td>RPC Client Port Queue Num Ops</td>
<td>Total number of RPC requests in the client port queue.</td>
</tr>
<tr>
<td>RPC CLIENT PORT PROCESSING TIME</td>
<td>RPC Client Port Processing Time</td>
<td>Average RPC request processing time in milliseconds, on the client port.</td>
</tr>
<tr>
<td></td>
<td>RPC Client Port Processing Num Ops</td>
<td>Total number of RPC active requests through the client port.</td>
</tr>
<tr>
<td>GC COUNT &amp; GC TIME</td>
<td>GC Count</td>
<td>Shows the JVM garbage collection rate on the NameNode.</td>
</tr>
<tr>
<td></td>
<td>GC Time</td>
<td>Shows the garbage collection time in milliseconds.</td>
</tr>
<tr>
<td>GC PAR NEW</td>
<td>GC Count Par New</td>
<td>The number of times young generation garbage collection happened.</td>
</tr>
<tr>
<td></td>
<td>GC Time Par New</td>
<td>Indicates the duration of young generation garbage collection.</td>
</tr>
</tbody>
</table>
### GC EXTRA SLEEP & WARNING THRESHOLD EXCEEDED

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GC Extra Sleep Time</td>
<td>Indicates total garbage collection extra sleep time.</td>
</tr>
<tr>
<td>GC Warning Threshold Exceeded</td>
<td>Indicates number of times that the garbage collection warning threshold is exceeded</td>
</tr>
</tbody>
</table>

### RPC CLIENT PORT QUEUE & BACKOFF

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPC Client Port Queue Length</td>
<td>Indicates the current length of the RPC call queue.</td>
</tr>
<tr>
<td>RPC Client Port Backoff</td>
<td>Indicates number of client backoff requests.</td>
</tr>
</tbody>
</table>

### RPC SERVICE PORT QUEUE & NUM OPS

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPC Service Port Queue Time</td>
<td>Average time a RPC request waiting in the queue, in milliseconds. These requests are on the RPC port facing to the HDFS services, including DataNodes and the other NameNode.</td>
</tr>
<tr>
<td>RPC Service Port Queue Num Ops</td>
<td>Total number of RPC requests waiting in the queue. These requests are on the RPC port facing to the HDFS services, including DataNodes and the other NameNode.</td>
</tr>
</tbody>
</table>

### RPC SERVICE PORT PROCESSING TIME & NUM OPS

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPC Service Port Processing Time</td>
<td>Average RPC request processing time in milliseconds, for the service port.</td>
</tr>
<tr>
<td>RPC Service Port Processing Num Ops</td>
<td>Number of RPC requests processed for the service port.</td>
</tr>
</tbody>
</table>

### RPC SERVICE PORT CALL QUEUE LENGTH & SLOW CALLS

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPC Service Port Call Queue Length</td>
<td>The current length of the RPC call queue.</td>
</tr>
<tr>
<td>RPC Service Port Slow Calls</td>
<td>The number of slow RPC requests, for the service port.</td>
</tr>
</tbody>
</table>

### TRANSACTIONS SINCE LAST EDIT & CHECKPOINT

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transactions Since Last Edit Roll</td>
<td>Total number of transactions since the last editlog segment.</td>
</tr>
<tr>
<td>Transactions Since Last Checkpoint</td>
<td>Total number of transactions since the last editlog segment checkpoint.</td>
</tr>
</tbody>
</table>

### LOCK QUEUE LENGTH & EXPIRED HEARTBEATS

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lock Queue Length</td>
<td>Shows the length of the wait Queue for the FSNamespaceLock.</td>
</tr>
<tr>
<td>Expired Heartbeats</td>
<td>Indicates the number of times expired heartbeats are detected on NameNode.</td>
</tr>
</tbody>
</table>

### THREADS BLOCKED / WAITING

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Threads Blocked</td>
<td>Indicates the number of threads in a BLOCKED state, which means they are waiting for a lock.</td>
</tr>
<tr>
<td>Threads Waiting</td>
<td>Indicates the number of threads in a WAITING state, which means they are waiting for another thread to perform an action.</td>
</tr>
</tbody>
</table>

### 9.1.3.4.3. HDFS - DataNodes

Metrics to see status for the DataNodes.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blocks Written</td>
<td>The rate or number of blocks written to a DataNode.</td>
</tr>
<tr>
<td>Blocks Read</td>
<td>The rate or number of blocks read from a DataNode.</td>
</tr>
<tr>
<td>Fsynch Time</td>
<td>Average fsync time.</td>
</tr>
<tr>
<td>Fsynch Num Ops</td>
<td>Total number of fsync operations.</td>
</tr>
<tr>
<td>Data Packet Blocked Time</td>
<td>Indicates the average waiting time of transferring a data packet on a DataNode.</td>
</tr>
<tr>
<td>Data Packet Blocked Num Ops</td>
<td>Indicates the number of data packets transferred on a DataNode.</td>
</tr>
<tr>
<td>Packet Transfer Time</td>
<td>Average transfer time of sending data packets on a DataNode.</td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
</tr>
<tr>
<td>---------------------------------------------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
</tr>
<tr>
<td>NETWORK ERRORS / GC COUNT</td>
<td>Network Errors</td>
</tr>
<tr>
<td>GC TIME / GC TIME PARNEW</td>
<td>GC Count</td>
</tr>
<tr>
<td></td>
<td>GC Time</td>
</tr>
<tr>
<td></td>
<td>GC Time ParNew</td>
</tr>
</tbody>
</table>

**9.1.3.4.4. HDFS - Top-N**

**Metrics that show**

- Which users perform most HDFS operations on the cluster
- Which HDFS operations run most often on the cluster.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TOP N - Operations Count</strong></td>
<td>Top N Total Operations Count</td>
<td>Represents the metrics that show the total operation count per operation for all users</td>
</tr>
<tr>
<td></td>
<td>1 min sliding window</td>
<td>1 minute interval</td>
</tr>
<tr>
<td></td>
<td>Top N Total Operations Count</td>
<td>Represents the metrics that show the total operation count per operation for all users</td>
</tr>
<tr>
<td></td>
<td>5 min sliding window</td>
<td>5 minute interval</td>
</tr>
<tr>
<td></td>
<td>Top N Total Operations Count</td>
<td>Represents the metrics that show the total operation count per operation for all users</td>
</tr>
<tr>
<td></td>
<td>25 min sliding window</td>
<td>25 minute interval</td>
</tr>
<tr>
<td><strong>TOP N - Total Operations Count By User</strong></td>
<td>Top N Total Operations Count by User</td>
<td>Represents the metrics that show the total operation count per user shown for 1-minute intervals</td>
</tr>
<tr>
<td></td>
<td>1 min sliding window</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Top N Total Operations Count by User</td>
<td>Represents the metrics that show the total operation count per user shown for 5-minute intervals</td>
</tr>
<tr>
<td></td>
<td>5 min sliding window</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Top N Total Operations Count by User</td>
<td>Represents the metrics that show the total operation count per user shown for 25-minute intervals</td>
</tr>
<tr>
<td></td>
<td>25 min sliding window</td>
<td></td>
</tr>
<tr>
<td><strong>TOP N - Operations by User</strong></td>
<td>TOP N - Operations by User</td>
<td>Represents the drilled down User x Op metrics against the TotalCount shown for 1-minute intervals</td>
</tr>
<tr>
<td></td>
<td>1 min sliding window</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TOP N - Operations by User</td>
<td>Represents the drilled down User x Op metrics against the TotalCount shown for 5-minute intervals</td>
</tr>
<tr>
<td></td>
<td>1 min sliding window</td>
<td></td>
</tr>
</tbody>
</table>
Row | Metrics                     | Description                                                                 |
---|------------------------------|-----------------------------------------------------------------------------|
|    | 5 min sliding window         |                                                                             |
|    | TOP N - Operations by User   | Represents the drilled down User x Op metrics against the TotalCount        |
|    | 25 min sliding window        | shown for 25-minute intervals                                              |

9.1.3.4.5. HDFS - Users

Metrics to see status for Users.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Namenode Rpc Caller Volume</td>
<td>Namenode Rpc Caller Volume</td>
<td>Number of RPC calls made by top(10) users.</td>
</tr>
<tr>
<td>Namenode Rpc Caller Priority</td>
<td>Namenode Rpc Caller Priority</td>
<td>Priority assignment for incoming calls from top(10) users.</td>
</tr>
</tbody>
</table>

9.1.3.5. YARN Dashboards

The following Grafana dashboards are available for YARN:

- YARN - Home [139]
- YARN - Applications [139]
- YARN - MR JobHistory Server [140]
- YARN - MR JobHistory Server [140]
- YARN - NodeManagers [140]
- YARN - Queues [141]
- YARN - ResourceManager [141]

9.1.3.5.1. YARN - Home

Metrics to see the overall status for the YARN cluster.

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nodes</td>
<td>The number of (active, unhealthy, lost) nodes in the cluster.</td>
</tr>
<tr>
<td>Apps</td>
<td>The number of (running, pending, completed, failed) apps in the cluster.</td>
</tr>
<tr>
<td>Cluster Memory Available</td>
<td>Total available memory in the cluster.</td>
</tr>
</tbody>
</table>

9.1.3.5.2. YARN - Applications

Metrics to see status of Applications on the YARN Cluster.

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Applications By Running Time</td>
<td>Number of apps by running time in 4 categories by default ( &lt; 1 hour, 1 ~ 5 hours, 5 ~ 24 hours, &gt; 24 hours).</td>
</tr>
</tbody>
</table>
## Metrics

| Apps Running vs Pending | The number of running apps vs the number of pending apps in the cluster. |
| Apps Submitted vs Completed | The number of submitted apps vs the number of completed apps in the cluster. |
| Avg AM Launch Delay | The average time taken from allocating an AM container to launching an AM container. |
| Avg AM Register Delay | The average time taken from RM launches an AM container to AM registers back with RM. |

### 9.1.3.5.3. YARN - MR JobHistory Server

Metrics to see status of the Job History Server.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JVM METRICS</td>
<td>GC Count</td>
<td>Accumulated GC count over time.</td>
</tr>
<tr>
<td></td>
<td>GC Time</td>
<td>Accumulated GC time over time.</td>
</tr>
<tr>
<td></td>
<td>Heap Mem Usage</td>
<td>Current heap memory usage.</td>
</tr>
<tr>
<td></td>
<td>NonHeap Mem Usage</td>
<td>Current non-heap memory usage.</td>
</tr>
</tbody>
</table>

### 9.1.3.5.4. YARN - NodeManagers

Metrics to see status of YARN NodeManagers on the YARN cluster.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUM CONTAINERS</td>
<td>Containers Running</td>
<td>Current number of running containers.</td>
</tr>
<tr>
<td></td>
<td>Containers Failed</td>
<td>Accumulated number of failed containers.</td>
</tr>
<tr>
<td></td>
<td>Containers Killed</td>
<td>Accumulated number of killed containers.</td>
</tr>
<tr>
<td></td>
<td>Containers Completed</td>
<td>Accumulated number of completed containers.</td>
</tr>
<tr>
<td>MEMORY UTILIZATION</td>
<td>Memory Available</td>
<td>Available memory for allocating containers on this node.</td>
</tr>
<tr>
<td></td>
<td>Used Memory</td>
<td>Used memory by containers on this node.</td>
</tr>
<tr>
<td>DISK UTILIZATION</td>
<td>Disk Utilization for Good Log Dirs</td>
<td>Disk utilization percentage across all good log directories.</td>
</tr>
<tr>
<td></td>
<td>Disk Utilization for Good Local Dirs</td>
<td>Disk utilization percentage across all good local directories.</td>
</tr>
<tr>
<td></td>
<td>Bad Log Dirs</td>
<td>Number of bad log directories.</td>
</tr>
<tr>
<td></td>
<td>Bad Local Dirs</td>
<td>Number of bad local directories.</td>
</tr>
<tr>
<td>AVE CONTAINER LAUNCH DELAY</td>
<td>Ave Container Launch Delay</td>
<td>Average time taken for a NM to launch a container.</td>
</tr>
<tr>
<td>RPC METRICS</td>
<td>RPC Avg Processing Time</td>
<td>Average time for processing a RPC call.</td>
</tr>
<tr>
<td></td>
<td>RPC Avg Queue Time</td>
<td>Average time for queuing a RPC call.</td>
</tr>
<tr>
<td></td>
<td>RPC Call Queue Length</td>
<td>The length of the RPC call queue.</td>
</tr>
<tr>
<td></td>
<td>RPC Slow Calls</td>
<td>Number of slow RPC calls.</td>
</tr>
<tr>
<td>JVM METRICS</td>
<td>Heap Mem Usage</td>
<td>Current heap memory usage.</td>
</tr>
<tr>
<td></td>
<td>NonHeap Mem Usage</td>
<td>Current non-heap memory usage.</td>
</tr>
<tr>
<td></td>
<td>GC Count</td>
<td>Accumulated GC count over time.</td>
</tr>
<tr>
<td></td>
<td>GC Time</td>
<td>Accumulated GC time over time.</td>
</tr>
</tbody>
</table>
### LOG4J METRICS

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOG ERROR</td>
<td>Number of ERROR logs.</td>
</tr>
<tr>
<td>LOG FATAL</td>
<td>Number of FATAL logs.</td>
</tr>
</tbody>
</table>

### 9.1.3.5.5. YARN - Queues

Metrics to see status of Queues on the YARN cluster.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUM APPS</td>
<td>Apps Running</td>
<td>Current number of running applications.</td>
</tr>
<tr>
<td></td>
<td>Apps Pending</td>
<td>Current number of pending applications.</td>
</tr>
<tr>
<td></td>
<td>Apps Completed</td>
<td>Accumulated number of completed applications over time.</td>
</tr>
<tr>
<td></td>
<td>Apps Failed</td>
<td>Accumulated number of failed applications over time.</td>
</tr>
<tr>
<td></td>
<td>Apps Killed</td>
<td>Accumulated number of killed applications over time.</td>
</tr>
<tr>
<td></td>
<td>Apps Submitted</td>
<td>Accumulated number of submitted applications over time.</td>
</tr>
</tbody>
</table>

### NUM CONTAINERS

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Containers Running</td>
<td>Current number of running containers.</td>
</tr>
<tr>
<td>Containers Pending</td>
<td>Current number of pending containers.</td>
</tr>
<tr>
<td>Containers Reserved</td>
<td>Current number of Reserved containers.</td>
</tr>
<tr>
<td>Total Containers Allocated</td>
<td>Accumulated number of containers allocated over time.</td>
</tr>
<tr>
<td>Total Node Local Containers Allocated</td>
<td>Accumulated number of node-local containers allocated over time.</td>
</tr>
<tr>
<td>Total Rack Local Containers Allocated</td>
<td>Accumulated number of rack-local containers allocated over time.</td>
</tr>
<tr>
<td>Total OffSwitch Containers Allocated</td>
<td>Accumulated number of off-switch containers allocated over time.</td>
</tr>
</tbody>
</table>

### MEMORY UTILIZATION

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allocated Memory</td>
<td>Current amount of memory allocated for containers.</td>
</tr>
<tr>
<td>Pending Memory</td>
<td>Current amount of memory asked by applications for allocating containers.</td>
</tr>
<tr>
<td>Available Memory</td>
<td>Current amount of memory available for allocating containers.</td>
</tr>
<tr>
<td>Reserved Memory</td>
<td>Current amount of memory reserved for containers.</td>
</tr>
<tr>
<td>Memory Used by AM</td>
<td>Current amount of memory used by AM containers.</td>
</tr>
</tbody>
</table>

### CONTAINER ALLOCATION DELAY

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ave AM Container Allocation Delay</td>
<td>Average time taken to allocate an AM container since the AM container is requested.</td>
</tr>
</tbody>
</table>

### 9.1.3.5.6. YARN - ResourceManager

Metrics to see status of ResourceManagers on the YARN cluster.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPC STATS</td>
<td>RPC Avg Processing / Queue Time</td>
<td>Average time for processing/queuing a RPC call.</td>
</tr>
<tr>
<td></td>
<td>RPC Call Queue Length</td>
<td>The length of the RPC call queue.</td>
</tr>
<tr>
<td></td>
<td>RPC Slow calls</td>
<td>Number of slow RPC calls.</td>
</tr>
</tbody>
</table>
9.1.3.5.7. YARN - TimelineServer

Metrics to see the overall status for TimelineServer.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MEMORY USAGE</td>
<td>Heap Mem Usage</td>
<td>Current heap memory usage.</td>
</tr>
<tr>
<td></td>
<td>NonHeap Mem Usage</td>
<td>Current non-heap memory usage.</td>
</tr>
<tr>
<td>GC STATS</td>
<td>GC count</td>
<td>Accumulated GC count over time.</td>
</tr>
<tr>
<td></td>
<td>GcTime</td>
<td>Accumulated GC time over time.</td>
</tr>
<tr>
<td>LOG ERRORS</td>
<td>Log Error / Fatal</td>
<td>Number of ERROR/FATAL logs.</td>
</tr>
<tr>
<td>AUTHORIZATION &amp; AUTHENTICATION FAILURES</td>
<td>RPC Authorization Failures</td>
<td>Number of authorization failures.</td>
</tr>
<tr>
<td></td>
<td>RPC Authentication Failures</td>
<td>Number of authentication failures.</td>
</tr>
</tbody>
</table>

9.1.3.6. Hive Dashboards

The following Grafana dashboards are available for Hive:

- Hive - Home [142]
- Hive - HiveMetaStore [143]
- Hive - HiveServer2 [143]

9.1.3.6.1. Hive - Home

Metrics that show the overall status for Hive service.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WAREHOUSE SIZE - AT A GLANCE</td>
<td>DB count at startup</td>
<td>Number of databases present at the last warehouse service startup time.</td>
</tr>
<tr>
<td></td>
<td>Table count at startup</td>
<td>Number of tables present at the last warehouse service startup time.</td>
</tr>
<tr>
<td></td>
<td>Partition count at startup</td>
<td>Number of partitions present at the last warehouse service startup time.</td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
<td>Description</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>WAREHOUSE SIZE - REALTIME GROWTH</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>#tables created (ongoing)</td>
<td>Number of tables created since the last warehouse service startup.</td>
</tr>
<tr>
<td></td>
<td>#partitions created (ongoing)</td>
<td>Number of partitions created since the last warehouse service startup.</td>
</tr>
<tr>
<td><strong>MEMORY PRESSURE</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HiveMetaStore Memory - Max</td>
<td>Heap memory usage by Hive MetaStores. If applicable, indicates max usage across multiple instances.</td>
<td></td>
</tr>
<tr>
<td>HiveServer2 Memory - Max</td>
<td>Heap memory usage by HiveServer2. If applicable, indicates max usage across multiple instances.</td>
<td></td>
</tr>
<tr>
<td>HiveMetaStore Offheap Memory - Max</td>
<td>Non-heap memory usage by Hive MetaStores. If applicable, indicates max usage across multiple instances.</td>
<td></td>
</tr>
<tr>
<td>HiveServer2 Offheap Memory - Max</td>
<td>Non-heap memory usage by HiveServer2. If applicable, indicates max across multiple instances.</td>
<td></td>
</tr>
<tr>
<td>HiveMetaStore app stop times (due to GC stops)</td>
<td>Total time spent in application pauses caused by garbage collection across Hive MetaStores.</td>
<td></td>
</tr>
<tr>
<td>HiveServer2 app stop times (due to GC stops)</td>
<td>Total time spent in application pauses caused by garbage collection across HiveServer2.</td>
<td></td>
</tr>
<tr>
<td><strong>METASTORE - CALL TIMES</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>API call times - Health Check roundtrip (get_all_databases)</td>
<td>Time taken to process a low-cost call made by health checks to all metastores.</td>
<td></td>
</tr>
<tr>
<td>API call times - Moderate size call (get_partitions_by_names)</td>
<td>Time taken to process a moderate-cost call made by queries, exports/etc to all metastores. Data for this metric may not be available in a less active warehouse.</td>
<td></td>
</tr>
</tbody>
</table>

### 9.1.3.6.2. Hive - HiveMetaStore

Metrics that show operating status for HiveMetaStore hosts. Select a HiveMetaStore and a host to view relevant metrics.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>API TIMES</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>API call times - Health Check roundtrip (get_all_databases)</td>
<td>Time taken to process a low-cost call made by health checks to this metastore.</td>
<td></td>
</tr>
</tbody>
</table>

| **MEMORY PRESSURE** | | |
| App Stop times (due to GC) | Time spent in application pauses caused by garbage collection. |
| Heap Usage | Current heap memory usage. |
| Off-Heap Usage | Current non-heap memory usage. |

### 9.1.3.6.3. Hive - HiveServer2

Metrics that show operating status for HiveServer2 hosts. Select a HiveServer2 and a host to view relevant metrics.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>API TIMES</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>API call times - Health Check</td>
<td>Time taken to process a low-cost call made by health checks to the metastore embedded in this HiveServer2. Data for this</td>
<td></td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td><strong>roundtrip (get_all_databases)</strong></td>
<td>metric may not be available if HiverServer2 is not running in an embedded-metastore mode.</td>
</tr>
<tr>
<td></td>
<td><strong>API call times - Moderate size call (get_partitions_by_names)</strong></td>
<td>Time taken to process a moderate-cost call made by queries/exports/etc to the metastore embedded in this HiveServer2. This metric may not be available in a less active warehouse, or if HiveServer2 is not running in an embedded-metastore mode.</td>
</tr>
<tr>
<td>MEMORY PRESSURE</td>
<td><strong>App Stop times (due to GC)</strong></td>
<td>Time spent in application pauses caused by garbage collection.</td>
</tr>
<tr>
<td></td>
<td><strong>Heap Usage</strong></td>
<td>Current heap memory usage.</td>
</tr>
<tr>
<td></td>
<td><strong>Off-Heap Usage</strong></td>
<td>Current non-heap memory usage.</td>
</tr>
<tr>
<td>THREAD STATES</td>
<td><strong>Active operation count</strong></td>
<td>Current number of active operations in HiveServer2 and their running states.</td>
</tr>
<tr>
<td></td>
<td><strong>Completed operation states</strong></td>
<td>Number of completed operations on HiveServer2 since the last restart. Indicates whether they completed as expected or encountered errors.</td>
</tr>
</tbody>
</table>

### 9.1.3.7. Hive LLAP Dashboards

The following Grafana dashboards are available for Apache Hive LLAP. The LLAP Heat map dashboard and the LLAP Overview dashboard enable you to quickly see the hotspots among the LLAP daemons. If you find an issue and want to navigate to more specific information for a specific system, use the LLAP Daemon dashboard.

Note that all Hive LLAP dashboards show the state of the cluster and are useful for looking at cluster information from the previous hour or day. The dashboards do not show real-time results.

- [Hive LLAP - Heatmap](#)
- [Hive LLAP - Overview](#)
- [Hive LLAP - Daemon](#)

#### 9.1.3.7.1. Hive LLAP - Heatmap

The heat map dashboard shows all the nodes that are running LLAP daemons and includes a percentage summary for available executors and cache. This dashboard enables you to identify the hotspots in the cluster in terms of executors and cache.

The values in the table are color coded based on threshold: if the threshold is more than 50%, the color is green; between 20% and 50%, the color is yellow; and less than 20%, the color is red.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heat maps</td>
<td><strong>Remaining Cache Capacity</strong></td>
<td>Shows the percentage of cache capacity remaining across the nodes. For example, if the grid is green, the cache is being under utilized. If the grid is red, there is high utilization of cache.</td>
</tr>
<tr>
<td></td>
<td><strong>Remaining Cache Capacity</strong></td>
<td>Same as above (Remaining Cache Capacity), but shows the cache hit ratio.</td>
</tr>
<tr>
<td></td>
<td><strong>Executor Free Slots</strong></td>
<td>Shows the percentage of executor free slots that are available on each nodes.</td>
</tr>
</tbody>
</table>
### 9.1.3.7.2. Hive LLAP - Overview

The overview dashboard shows the aggregated information across all of the clusters: for example, the total cache memory from all the nodes. This dashboard enables you to see that your cluster is configured and running correctly. For example, you might have configured 10 nodes but you see only 8 nodes running.

If you find an issue by viewing this dashboard, you can open the LLAP Daemon dashboard to see which node is having the problem.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Overview</strong></td>
<td>Total Executor Threads</td>
<td>Shows the total number of executors across all nodes.</td>
</tr>
<tr>
<td></td>
<td>Total Executor Memory</td>
<td>Shows the total amount of memory for executors across all nodes.</td>
</tr>
<tr>
<td></td>
<td>Total Cache Memory</td>
<td>Shows the total amount of memory for cache across all nodes.</td>
</tr>
<tr>
<td></td>
<td>Total JVM Memory</td>
<td>Shows the total amount of max Java Virtual Machine (JVM) memory across all nodes.</td>
</tr>
<tr>
<td><strong>Cache Metrics Across all nodes</strong></td>
<td>Total Cache Usage</td>
<td>Shows the total amount of cache usage (Total, Remaining, and Used) across all nodes.</td>
</tr>
<tr>
<td></td>
<td>Average Cache Hit Rate</td>
<td>As the data is released from the cache, the curve should increase. For example, the first query should run at 0, the second at 80-90 seconds, and then the third 10% faster. If, instead, it decreases, there might be a problem in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Average Cache Read Requests</td>
<td>Shows how many requests are being made for the cache and how many queries you are able to run that make use of the cache. If it says 0, for example, your cache might not be working properly and this grid might reveal a configuration issue.</td>
</tr>
<tr>
<td><strong>Cache Metrics Across all nodes</strong></td>
<td>Total Cache Usage</td>
<td>Shows the total amount of cache usage (Total, Remaining, and Used) across all nodes.</td>
</tr>
<tr>
<td></td>
<td>Average Cache Hit Rate</td>
<td>As the data is released from the cache, the curve should increase. For example, the first query should run at 0, the second at 80-90 seconds, and then the third 10% faster. If, instead, it decreases, there might be a problem in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Average Cache Read Requests</td>
<td>Shows how many requests are being made for the cache and how many queries you are able to run that make use of the cache. If it says 0, for example, your cache might not be working properly and this grid might reveal a configuration issue.</td>
</tr>
<tr>
<td><strong>Executor Metrics Across All nodes</strong></td>
<td>Total Executor Requests</td>
<td>Shows the total number of task requests that were handled, succeeded, failed, killed, evicted and rejected across all nodes.</td>
</tr>
<tr>
<td></td>
<td>Succeeded: Total requests that were processed. For example, if you have 8 core machines, the number of total executor requests would be 8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Failed: Did not complete successfully because, for example, you ran out of memory</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Rejected: If all task priorities are the same, but there are still not enough slots to fulfill the request, the system will reject some tasks</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Evicted: Lower priority requests are evicted if the slots are filled by higher priority requests</td>
<td></td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
<td>Description</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>Total Execution Slots</td>
<td>Shows the total execution slots, the number of free or available slots, and number of slots occupied in the wait queue across all nodes.</td>
<td>Ideally, the threads available (blue) result should be the same as the threads that are occupied in the queue result.</td>
</tr>
<tr>
<td>Time to Kill Pre-empted Task (300s interval)</td>
<td>Shows the time that it took to kill a query due to pre-emption in percentile (50th, 90th, 99th) latencies in 300 second intervals.</td>
<td></td>
</tr>
<tr>
<td>Max Time To Kill Task (due to preemption)</td>
<td>Shows the maximum time taken to kill a task due to pre-emption. This grid and the one above show you if you are wasting a lot of time killing queries. Time lost while a task is waiting to be killed is time lost in the cluster. If your max time to kill is high, you might want to disable this feature.</td>
<td></td>
</tr>
<tr>
<td>Pre-emption Time Lost (300s interval)</td>
<td>Shows the time lost due to pre-emption in percentile (50th, 90th, 99th) latencies in 300 second intervals.</td>
<td></td>
</tr>
<tr>
<td>Max Time Lost In Cluster (due to pre-emption)</td>
<td>Shows the maximum time lost due to pre-emption. If your max time to kill is high, you might want to disable this feature.</td>
<td></td>
</tr>
<tr>
<td>IO Elevator Metrics Across All Nodes</td>
<td>Column Decoding Time (30s interval)</td>
<td>Shows the percentile (50th, 90th, 99th) latencies for time it takes to decode the column chunk (convert encoded column chunk to column vector batches for processing) in 30 second intervals.</td>
</tr>
<tr>
<td>Max Column Decoding Time</td>
<td>Shows the maximum time taken to decode column chunk (convert encoded column chunk to column vector batches for processing).</td>
<td></td>
</tr>
<tr>
<td>JVM Metrics across all nodes</td>
<td>Average JVM Heap Usage</td>
<td>Shows the average amount of Java Virtual Machine (JVM) heap memory used across all nodes.</td>
</tr>
<tr>
<td>Average JVM Non-Heap Usage</td>
<td>Shows the average amount of JVM non-heap memory used across all nodes.</td>
<td></td>
</tr>
<tr>
<td>Max GcTotalExtraSleepTimeMillis</td>
<td>Shows the maximum garbage collection extra sleep time in milliseconds across all nodes. Garbage collection extra sleep time measures when the garbage collection monitoring is delayed (for example, the thread does not wake up after 500 milliseconds).</td>
<td></td>
</tr>
<tr>
<td>Max GcTimeMillis</td>
<td>Shows the total maximum GC time in milliseconds across all nodes.</td>
<td></td>
</tr>
<tr>
<td>Total JVM Threads</td>
<td>Shows the total number of JVM threads that are in a NEW, RUNNABLE, WAITING, TIMED_WAITING, and TERMINATED state across all nodes.</td>
<td></td>
</tr>
<tr>
<td>JVM Metrics</td>
<td>Total JVM Heap Used</td>
<td>Shows the total amount of Java Virtual Machine (JVM) heap memory used in the daemon.</td>
</tr>
<tr>
<td>Total JVM Non-Heap Used</td>
<td>Shows the total amount of JVM non-heap memory used in the LLAP daemon.</td>
<td></td>
</tr>
</tbody>
</table>
If the non-heap memory is over-allocated, you might run out of memory and the task failure count would also increase.

Max \text{GcTotalExtraSleepTimeMillis} shows the maximum garbage collection extra sleep time in milliseconds in the LLAP daemon. Garbage collection extra sleep time measures when the garbage collection monitoring is delayed (for example, the thread does not wake up after 500 milliseconds).

Max \text{GcTimeMillis} shows the total maximum GC time in milliseconds in the LLAP daemon.

Max JVM Threads \text{Runnable} shows the maximum number of Java Virtual Machine (JVM) threads that are in \text{RUNNABLE} state.

Max JVM Threads \text{Blocked} shows the maximum number of JVM threads that are in \text{BLOCKED} state. If you are seeing spikes in the threads blocked, you might have a problem with your LLAP daemon.

Max JVM Threads \text{Waiting} shows the maximum number of JVM threads that are in \text{WAITING} state.

Max JVM Threads \text{Timed Waiting} shows the maximum number of JVM threads that are in \text{TIMED_WAITING} state.

9.1.3.7.3. Hive LLAP - Daemon

Metrics that show operating status for Hive LLAP Daemons.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Requests Submitted</td>
<td>Shows the total number of task requests handled by the daemon.</td>
<td></td>
</tr>
<tr>
<td>Total Requests Succeeded</td>
<td>Shows the total number of successful task requests handled by the daemon.</td>
<td></td>
</tr>
<tr>
<td>Total Requests Failed</td>
<td>Shows the total number of failed task requests handled by the daemon.</td>
<td></td>
</tr>
<tr>
<td>Total Requests Killed</td>
<td>Shows the total number of killed task requests handled by the daemon.</td>
<td></td>
</tr>
<tr>
<td>Total Requests Evicted From Wait Queue</td>
<td>Shows the total number of task requests handled by the daemon that were evicted from the wait queue. Tasks are evicted if all of the executor threads are in use by higher priority tasks.</td>
<td></td>
</tr>
<tr>
<td>Total Requests Rejected</td>
<td>Shows the total number of task requests handled by the daemon that were rejected by the task executor service. Task are rejected if all of the executor threads are in use and the wait queue is full of tasks that are not eligible for eviction.</td>
<td></td>
</tr>
<tr>
<td>Available Execution Slots</td>
<td>Shows the total number of free slots that are available for execution including free executor threads and free slots in the wait queue.</td>
<td></td>
</tr>
<tr>
<td>95th Percentile Pre-emption Time Lost (300s interval)</td>
<td>Shows the 95th percentile latencies for time lost due to pre-emption in 300 second intervals.</td>
<td></td>
</tr>
<tr>
<td>Max Pre-emption Time Lost</td>
<td>Shows the maximum time lost due to pre-emption.</td>
<td></td>
</tr>
<tr>
<td>95th Percentile Time to Kill Pre-empted Task (300s interval)</td>
<td>Shows the 95th percentile latencies for time taken to kill tasks due to pre-emption in 300 second intervals.</td>
<td></td>
</tr>
<tr>
<td>Max Time To Kill Task Pre-empted Task</td>
<td>Shows the maximum time taken to kill a task due to pre-emption.</td>
<td></td>
</tr>
</tbody>
</table>
### Cache Metrics

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total Cache Used</td>
<td>Shows the total amount of cache usage (Total, Remaining, and Used) in LLAP daemon cache.</td>
</tr>
<tr>
<td></td>
<td>Heap Usage</td>
<td>Shows the amount of memory remaining in LLAP daemon cache.</td>
</tr>
<tr>
<td></td>
<td>Average Cache Hit Rate</td>
<td>As the data is released from the cache, the curve should increase. For example, the first query should run at 0, the second at 80-90 seconds, and then the third 10% faster. If, instead, it decreases, there might be a problem in the LLAP daemon.</td>
</tr>
<tr>
<td></td>
<td>Total Cache Read Requests</td>
<td>Shows the total number of read requests received by LLAP daemon cache.</td>
</tr>
</tbody>
</table>

### THREAD STATES

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>95th Percentile Column Decoding Time (30s interval)</td>
<td>Shows the 95th percentile latencies for time it takes to decode the column chunk (convert encoded column chunk to column vector batches for processing) in 30 second intervals. The cache comes from IO Elevator. It loads data from HDFS to the cache, and then from the cache to the executor. This metric shows how well the threads are performing and is useful to see that the threads are running.</td>
</tr>
<tr>
<td></td>
<td>Max Column Decoding Time</td>
<td>Shows the maximum time taken to decode column chunk (convert encoded column chunk to column vector batches for processing).</td>
</tr>
</tbody>
</table>

### 9.1.3.8. HBase Dashboards

Monitoring an HBase cluster is essential for maintaining a high-performance and stable system. The following Grafana dashboards are available for HBase:

- HBase - Home [148]
- HBase - RegionServers [149]
- HBase - Misc [154]
- HBase - Tables [155]
- HBase - Users [157]

**Important**

Ambari disables per-region, per-table, and per-user metrics for HBase by default. See [Enabling Individual Region, Table, and User Metrics for HBase](#) if you want the Ambari Metrics System to display the more granular metrics of HBase system performance on the individual region, table, or user level.

#### 9.1.3.8.1. HBase - Home

The HBase - Home dashboards display basic statistics about an HBase cluster. These dashboards provide insight to the overall status for the HBase cluster.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Num RegionServers</td>
<td>Total number of RegionServers in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Num Dead RegionServers</td>
<td>Total number of RegionServers that are dead in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Num Regions</td>
<td>Total number of regions in the cluster.</td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NUM REGIONS/STORES</td>
<td>Avg Num Regions per RegionServer</td>
<td>Average number of regions per RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Regions / Stores - Total</td>
<td>Total number of regions and stores (column families) in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Store File Size / Count - Total</td>
<td>Total data file size and number of store files.</td>
</tr>
<tr>
<td>NUM REQUESTS</td>
<td>Num Requests - Total</td>
<td>Total number of requests (read, write and RPCs) in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Num Request - Breakdown - Total</td>
<td>Total number of get, put, mutate, etc requests in the cluster.</td>
</tr>
<tr>
<td>REGIONSERVER MEMORY</td>
<td>RegionServer Memory - Average</td>
<td>Average used, max or committed on-heap and offheap memory for RegionServers.</td>
</tr>
<tr>
<td></td>
<td>RegionServer Offheap Memory - Average</td>
<td>Average used, free or committed on-heap and offheap memory for RegionServers.</td>
</tr>
<tr>
<td>MEMORY - MEMSTORE BLOCKCACHE</td>
<td>Memstore - BlockCache - Average</td>
<td>Average blockcache and memstore sizes for RegionServers.</td>
</tr>
<tr>
<td></td>
<td>Num Blocks in BlockCache - Total</td>
<td>Total number of (hfile) blocks in the blockcaches across all RegionServers.</td>
</tr>
<tr>
<td>BLOCKCACHE</td>
<td>BlockCache Hit/ Miss/s Total</td>
<td>Total number of blockcache hits misses and evictions across all RegionServers.</td>
</tr>
<tr>
<td></td>
<td>BlockCache Hit Percent - Average</td>
<td>Average blockcache hit percentage across all RegionServers.</td>
</tr>
<tr>
<td>OPERATION LATENCIES - GET/MUTATE</td>
<td>Get Latencies - Average</td>
<td>Average min, median, max, 75th, 95th, 99th percentile latencies for Get operation across all RegionServers.</td>
</tr>
<tr>
<td></td>
<td>Mutate Latencies - Average</td>
<td>Average min, median, max, 75th, 95th, 99th percentile latencies for Mutate operation across all RegionServers.</td>
</tr>
<tr>
<td>OPERATION LATENCIES - DELETE/INCREMENT</td>
<td>Delete Latencies - Average</td>
<td>Average min, median, max, 75th, 95th, 99th percentile latencies for Delete operation across all RegionServers.</td>
</tr>
<tr>
<td></td>
<td>Increment Latencies - Average</td>
<td>Average min, median, max, 75th, 95th, 99th percentile latencies for Increment operation across all RegionServers.</td>
</tr>
<tr>
<td>OPERATION LATENCIES - APPEND/REPLAY</td>
<td>Append Latencies - Average</td>
<td>Average min, median, max, 75th, 95th, 99th percentile latencies for Append operation across all RegionServers.</td>
</tr>
<tr>
<td></td>
<td>Replay Latencies - Average</td>
<td>Average min, median, max, 75th, 95th, 99th percentile latencies for Replay operation across all RegionServers.</td>
</tr>
<tr>
<td>REGIONSERVER RPC</td>
<td>RegionServer RPC - Average</td>
<td>Average number of RPCs, active handler threads and open connections across all RegionServers.</td>
</tr>
<tr>
<td>REGIONSERVER RPC</td>
<td>RegionServer RPC Queues - Average</td>
<td>Average number of calls in different RPC scheduling queues and the size of all requests in the RPC queue across all RegionServers.</td>
</tr>
<tr>
<td></td>
<td>RegionServer RPC Throughput - Average</td>
<td>Average sent and received bytes from the RPC across all RegionServers.</td>
</tr>
</tbody>
</table>

### 9.1.3.8.2. HBase - RegionServers

The HBase - RegionServers dashboards display metrics for RegionServers in the monitored HBase cluster, including some performance-related data. These dashboards help you view basic I/O data and compare load among RegionServers.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUM REGIONS</td>
<td>Num Regions</td>
<td>Number of regions in the RegionServer.</td>
</tr>
<tr>
<td>STORE FILES</td>
<td>Store File Size</td>
<td>Total size of the store files (data files) in the RegionServer.</td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
<td>Description</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>Store File Count</td>
<td>Total number of store files in the RegionServer.</td>
</tr>
<tr>
<td>NUM REQUESTS</td>
<td>Num Total Requests /s</td>
<td>Total number of requests (both read and write) per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Write Requests /s</td>
<td>Total number of write requests per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Read Requests /s</td>
<td>Total number of read requests per second in the RegionServer.</td>
</tr>
<tr>
<td>NUM REQUESTS - GET / SCAN</td>
<td>Num Get Requests /s</td>
<td>Total number of Get requests per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Scan Next Requests /s</td>
<td>Total number of Scan requests per second in the RegionServer.</td>
</tr>
<tr>
<td>NUM REQUESTS - MUTATE / DELETE</td>
<td>Num Mutate Requests /s</td>
<td>Total number of Mutate requests per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Delete Requests /s</td>
<td>Total number of Delete requests per second in the RegionServer.</td>
</tr>
<tr>
<td>NUM REQUESTS - APPEND / INCREMENT</td>
<td>Num Append Requests /s</td>
<td>Total number of Append requests per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Increment Requests /s</td>
<td>Total number of Increment requests per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Replay Requests /s</td>
<td>Total number of Replay requests per second in the RegionServer.</td>
</tr>
<tr>
<td>MEMORY</td>
<td>RegionServer Memory Used</td>
<td>Heap Memory used by the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RegionServer Offheap Memory Used</td>
<td>Offheap Memory used by the RegionServer.</td>
</tr>
<tr>
<td>MEMSTORE</td>
<td>Memstore Size</td>
<td>Total Memstore memory size of the RegionServer.</td>
</tr>
<tr>
<td>BLOCKCACHE - OVERVIEW</td>
<td>BlockCache - Size</td>
<td>Total BlockCache size of the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>BlockCache - Free Size</td>
<td>Total free space in the BlockCache of the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Blocks in Cache</td>
<td>Total number of hfile blocks in the BlockCache of the RegionServer.</td>
</tr>
<tr>
<td>BLOCKCACHE - HITS/MISSES</td>
<td>Num BlockCache Hits /s</td>
<td>Number of BlockCache hits per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num BlockCache Misses /s</td>
<td>Number of BlockCache misses per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num BlockCache Evictions /s</td>
<td>Number of BlockCache evictions per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>BlockCache Caching Hit Percent</td>
<td>Percentage of BlockCache hits per second for requests that requested cache blocks in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>BlockCache Hit Percent</td>
<td>Percentage of BlockCache hits per second in the RegionServer.</td>
</tr>
<tr>
<td>OPERATION LATENCIES - GET</td>
<td>Get Latencies - Mean</td>
<td>Mean latency for Get operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Get Latencies - Median</td>
<td>Median latency for Get operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Get Latencies - 75th Percentile</td>
<td>75th percentile latency for Get operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Get Latencies - 95th Percentile</td>
<td>95th percentile latency for Get operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Get Latencies - 99th Percentile</td>
<td>99th percentile latency for Get operation in the RegionServer.</td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
<td>Description</td>
</tr>
<tr>
<td>-----</td>
<td>-----------------------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Get Latencies - Max</td>
<td>Max latency for Get operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Scan Next Latencies - Mean</td>
<td>Mean latency for Scan operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Scan Next Latencies - Median</td>
<td>Median latency for Scan operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Scan Next Latencies - 75th Percentile</td>
<td>75th percentile latency for Scan operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Scan Next Latencies - 95th Percentile</td>
<td>95th percentile latency for Scan operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Scan Next Latencies - 99th Percentile</td>
<td>99th percentile latency for Scan operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Scan Next Latencies - Max</td>
<td>Max latency for Scan operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Mutate Latencies - Mean</td>
<td>Mean latency for Mutate operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Mutate Latencies - Median</td>
<td>Median latency for Mutate operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Mutate Latencies - 75th Percentile</td>
<td>75th percentile latency for Mutate operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Mutate Latencies - 95th Percentile</td>
<td>95th percentile latency for Mutate operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Mutate Latencies - 99th Percentile</td>
<td>99th percentile latency for Mutate operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Mutate Latencies - Max</td>
<td>Max latency for Mutate operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Delete Latencies - Mean</td>
<td>Mean latency for Delete operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Delete Latencies - Median</td>
<td>Median latency for Delete operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Delete Latencies - 75th Percentile</td>
<td>75th percentile latency for Delete operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Delete Latencies - 95th Percentile</td>
<td>95th percentile latency for Delete operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Delete Latencies - 99th Percentile</td>
<td>99th percentile latency for Delete operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Delete Latencies - Max</td>
<td>Max latency for Delete operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Increment Latencies - Mean</td>
<td>Mean latency for Increment operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Increment Latencies - Median</td>
<td>Median latency for Increment operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Increment Latencies - 75th Percentile</td>
<td>75th percentile latency for Increment operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Increment Latencies - 95th Percentile</td>
<td>95th percentile latency for Increment operation in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Increment Latencies - 99th Percentile</td>
<td>99th percentile latency for Increment operation in the RegionServer.</td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------------------------------</td>
<td>----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Increment Latencies - Max</strong></td>
<td></td>
<td>Max latency for Increment operation in the RegionServer.</td>
</tr>
<tr>
<td><strong>Append Latencies - Mean</strong></td>
<td></td>
<td>Mean latency for Append operation in the RegionServer.</td>
</tr>
<tr>
<td><strong>Append Latencies - Median</strong></td>
<td></td>
<td>Median latency for Append operation in the RegionServer.</td>
</tr>
<tr>
<td><strong>Append Latencies - 75th Percentile</strong></td>
<td></td>
<td>75th percentile latency for Append operation in the RegionServer.</td>
</tr>
<tr>
<td><strong>Append Latencies - 95th Percentile</strong></td>
<td></td>
<td>95th percentile latency for Append operation in the RegionServer.</td>
</tr>
<tr>
<td><strong>Append Latencies - 99th Percentile</strong></td>
<td></td>
<td>99th percentile latency for Append operation in the RegionServer.</td>
</tr>
<tr>
<td><strong>Append Latencies - Max</strong></td>
<td></td>
<td>Max latency for Append operation in the RegionServer.</td>
</tr>
<tr>
<td><strong>Replay Latencies - Mean</strong></td>
<td></td>
<td>Mean latency for Replay operation in the RegionServer.</td>
</tr>
<tr>
<td><strong>Replay Latencies - Median</strong></td>
<td></td>
<td>Median latency for Replay operation in the RegionServer.</td>
</tr>
<tr>
<td><strong>Replay Latencies - 75th Percentile</strong></td>
<td></td>
<td>75th percentile latency for Replay operation in the RegionServer.</td>
</tr>
<tr>
<td><strong>Replay Latencies - 95th Percentile</strong></td>
<td></td>
<td>95th percentile latency for Replay operation in the RegionServer.</td>
</tr>
<tr>
<td><strong>Replay Latencies - 99th Percentile</strong></td>
<td></td>
<td>99th percentile latency for Replay operation in the RegionServer.</td>
</tr>
<tr>
<td><strong>Replay Latencies - Max</strong></td>
<td></td>
<td>Max latency for Replay operation in the RegionServer.</td>
</tr>
<tr>
<td><strong>Num Connections</strong></td>
<td>Number of connections to the RegionServer.</td>
<td></td>
</tr>
<tr>
<td><strong>Num Active Handler Threads</strong></td>
<td>Number of active RPC handler threads (to process requests) in the RegionServer.</td>
<td></td>
</tr>
<tr>
<td><strong>RPC - Total Call Queue Size</strong></td>
<td>Total data size of all RPC calls in the RPC queues in the RegionServer.</td>
<td></td>
</tr>
<tr>
<td><strong>RPC - Call Queued Time - Mean</strong></td>
<td>Mean latency for RPC calls to stay in the RPC queue in the RegionServer.</td>
<td></td>
</tr>
<tr>
<td><strong>RPC - Call Queued Time - Median</strong></td>
<td>Median latency for RPC calls to stay in the RPC queue in the RegionServer.</td>
<td></td>
</tr>
<tr>
<td><strong>RPC - Call Queued Time - 75th Percentile</strong></td>
<td>75th percentile latency for RPC calls to stay in the RPC queue in the RegionServer.</td>
<td></td>
</tr>
<tr>
<td><strong>RPC - Call Queued Time - 95th Percentile</strong></td>
<td>95th percentile latency for RPC calls to stay in the RPC queue in the RegionServer.</td>
<td></td>
</tr>
<tr>
<td><strong>RPC - Call Queued Time - 99th Percentile</strong></td>
<td>99th percentile latency for RPC calls to stay in the RPC queue in the RegionServer.</td>
<td></td>
</tr>
<tr>
<td><strong>RPC - Call Queued Time - Max</strong></td>
<td>Max latency for RPC calls to stay in the RPC queue in the RegionServer.</td>
<td></td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
<td>Description</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>RPC - CALL PROCESS TIMES</td>
<td>RPC - Call Process Time - Mean</td>
<td>Mean latency for RPC calls to be processed in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RPC - Call Process Time - Median</td>
<td>Median latency for RPC calls to be processed in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RPC - Call Process Time - 75th Percentile</td>
<td>75th percentile latency for RPC calls to be processed in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RPC - Call Process Time - 95th Percentile</td>
<td>95th percentile latency for RPC calls to be processed in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RPC - Call Process Time - 99th Percentile</td>
<td>99th percentile latency for RPC calls to be processed in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RPC - Call Process Time - Max</td>
<td>Max latency for RPC calls to be processed in the RegionServer.</td>
</tr>
<tr>
<td>RPC - THROUGHPUT</td>
<td>RPC - Received bytes /s</td>
<td>Received bytes from the RPC in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RPC - Sent bytes /s</td>
<td>Sent bytes from the RPC in the RegionServer.</td>
</tr>
<tr>
<td>WAL - FILES</td>
<td>Num WAL - Files</td>
<td>Number of Write-Ahead-Log files in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Total WAL File Size</td>
<td>Total files sized of Write-Ahead-Logs in the RegionServer.</td>
</tr>
<tr>
<td>WAL - THROUGHPUT</td>
<td>WAL - Num Apps /s</td>
<td>Number of append operations per second to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Num Sync /s</td>
<td>Number of sync operations per second to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td>WAL - SYNC LATENCIES</td>
<td>WAL - Sync Latencies - Mean</td>
<td>Mean latency for Write-Ahead-Log sync operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Sync Latencies - Median</td>
<td>Median latency for Write-Ahead-Log sync operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Sync Latencies - 75th Percentile</td>
<td>75th percentile latency for Write-Ahead-Log sync operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Sync Latencies - 95th Percentile</td>
<td>95th percentile latency for Write-Ahead-Log sync operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Sync Latencies - 99th Percentile</td>
<td>99th percentile latency for Write-Ahead-Log sync operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Sync Latencies - Max</td>
<td>Max latency for Write-Ahead-Log sync operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td>WAL - APPEND LATENCIES</td>
<td>WAL - Append Latencies - Mean</td>
<td>Mean latency for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Latencies - Median</td>
<td>Median latency for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Latencies - 75th Percentile</td>
<td>75th percentile latency for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Latencies - 95th Percentile</td>
<td>95th percentile latency for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Latencies - 99th Percentile</td>
<td>99th percentile latency for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Latencies - Max</td>
<td>Max latency for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>WAL - APPEND SIZES</strong></td>
<td>WAL - Append Sizes - Mean</td>
<td>Mean data size for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Sizes - Median</td>
<td>Median data size for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Sizes - 75th Percentile</td>
<td>75th percentile data size for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Sizes - 95th Percentile</td>
<td>95th percentile data size for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Sizes - 99th Percentile</td>
<td>99th percentile data size for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>WAL - Append Sizes - Max</td>
<td>Max data size for Write-Ahead-Log append operation to the filesystem in the RegionServer.</td>
</tr>
<tr>
<td><strong>SLOW OPERATIONS</strong></td>
<td>WAL Num Slow Append /s</td>
<td>Number of append operations per second to the filesystem that took more than 1 second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Slow Gets /s</td>
<td>Number of Get requests per second that took more than 1 second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Slow Puts /s</td>
<td>Number of Put requests per second that took more than 1 second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Slow Deletes /s</td>
<td>Number of Delete requests per second that took more than 1 second in the RegionServer.</td>
</tr>
<tr>
<td><strong>FLUSH/COMPACTION QUEUES</strong></td>
<td>Flush Queue Length</td>
<td>Number of Flush operations waiting to be processed in the RegionServer. A higher number indicates flush operations being slow.</td>
</tr>
<tr>
<td></td>
<td>Compaction Queue Length</td>
<td>Number of Compaction operations waiting to be processed in the RegionServer. A higher number indicates compaction operations being slow.</td>
</tr>
<tr>
<td></td>
<td>Split Queue Length</td>
<td>Number of Region Split operations waiting to be processed in the RegionServer. A higher number indicates split operations being slow.</td>
</tr>
<tr>
<td><strong>JVM - GC COUNTS</strong></td>
<td>GC Count /s</td>
<td>Number of Java Garbage Collections per second.</td>
</tr>
<tr>
<td></td>
<td>GC Count ParNew /s</td>
<td>Number of Java ParNew (YoungGen) Garbage Collections per second.</td>
</tr>
<tr>
<td></td>
<td>GC Count CMS /s</td>
<td>Number of Java CMS Garbage Collections per second.</td>
</tr>
<tr>
<td><strong>JVM - GC TIMES</strong></td>
<td>GC Times /s</td>
<td>Total time spend in Java Garbage Collections per second.</td>
</tr>
<tr>
<td></td>
<td>GC Times ParNew /s</td>
<td>Total time spend in Java ParNew(YoungGen) Garbage Collections per second.</td>
</tr>
<tr>
<td></td>
<td>GC Times CMS /s</td>
<td>Total time spend in Java CMS Garbage Collections per second.</td>
</tr>
<tr>
<td><strong>LOCALITY</strong></td>
<td>Percent Files Local</td>
<td>Percentage of files served from the local DataNode for the RegionServer.</td>
</tr>
</tbody>
</table>

**9.1.3.8.3. HBase - Misc**

The HBase - Misc dashboards display miscellaneous metrics related to the HBase cluster. You can use these metrics for tasks like debugging authentication and authorization issues and exceptions raised by RegionServers.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>REGIONS IN TRANSITION</strong></td>
<td>Master - Regions in Transition</td>
<td>Number of regions in transition in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Master - Regions in Transition Longer</td>
<td>Number of regions in transition that are in transition state for longer than 1 minute in the cluster.</td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
<td>Description</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>Than Threshold Time</td>
<td>Maximum time that a region stayed in transition state.</td>
</tr>
<tr>
<td></td>
<td>Regions in Transition Oldest Age</td>
<td></td>
</tr>
<tr>
<td>NUM THREADS - RUNNABLE</td>
<td>Master Num Threads - Runnable</td>
<td>Number of threads in the Master.</td>
</tr>
<tr>
<td></td>
<td>RegionServer Num Threads - Runnable</td>
<td>Number of threads in the RegionServer.</td>
</tr>
<tr>
<td>NUM THREADS - BLOCKED</td>
<td>Master Num Threads - Blocked</td>
<td>Number of threads in the Blocked State in the Master.</td>
</tr>
<tr>
<td></td>
<td>RegionServer Num Threads - Blocked</td>
<td>Number of threads in the Blocked State in the RegionServer.</td>
</tr>
<tr>
<td>NUM THREADS - WAITING</td>
<td>Master Num Threads - Waiting</td>
<td>Number of threads in the Waiting State in the Master.</td>
</tr>
<tr>
<td></td>
<td>RegionServer Num Threads - Waiting</td>
<td>Number of threads in the Waiting State in the RegionServer.</td>
</tr>
<tr>
<td>NUM THREADS - TIMED WAITING</td>
<td>Master Num Threads - Timed Waiting</td>
<td>Number of threads in the Timed-Waiting State in the Master.</td>
</tr>
<tr>
<td></td>
<td>RegionServer Num Threads - Timed Waiting</td>
<td>Number of threads in the Timed-Waiting State in the RegionServer.</td>
</tr>
<tr>
<td>NUM THREADS - NEW</td>
<td>Master Num Threads - New</td>
<td>Number of threads in the New State in the Master.</td>
</tr>
<tr>
<td></td>
<td>RegionServer Num Threads - New</td>
<td>Number of threads in the New State in the RegionServer.</td>
</tr>
<tr>
<td>NUM THREADS - TERMINATED</td>
<td>Master Num Threads - Terminated</td>
<td>Number of threads in the Terminated State in the Master.</td>
</tr>
<tr>
<td></td>
<td>RegionServer Num Threads - Terminated</td>
<td>Number of threads in the Terminated State in the RegionServer.</td>
</tr>
<tr>
<td>RPC AUTHENTICATION</td>
<td>RegionServer RPC Authentication Successes /s</td>
<td>Number of RPC successful authentications per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RegionServer RPC Authentication Failures /s</td>
<td>Number of RPC failed authentications per second in the RegionServer.</td>
</tr>
<tr>
<td>RPC Authorization</td>
<td>RegionServer RPC Authorization Successes /s</td>
<td>Number of RPC successful authorizations per second in the RegionServer.</td>
</tr>
<tr>
<td></td>
<td>RegionServer RPC Authorization Failures /s</td>
<td>Number of RPC failed authorizations per second in the RegionServer.</td>
</tr>
<tr>
<td>EXCEPTIONS</td>
<td>Master Exceptions /s</td>
<td>Number of exceptions in the Master.</td>
</tr>
<tr>
<td></td>
<td>RegionServer Exceptions /s</td>
<td>Number of exceptions in the RegionServer.</td>
</tr>
</tbody>
</table>

**9.1.3.8.4. HBase - Tables**

HBase - Tables metrics reflect data on the table level. The dashboards and data help you compare load distribution and resource use among tables in a cluster at different times.
<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUM REGIONS/STORES</td>
<td>Num Regions</td>
<td>Number of regions for the table(s).</td>
</tr>
<tr>
<td></td>
<td>Num Stores</td>
<td>Number of stores for the table(s).</td>
</tr>
<tr>
<td>TABLE SIZE</td>
<td>Table Size</td>
<td>Total size of the data (store files and MemStore) for the table(s).</td>
</tr>
<tr>
<td></td>
<td>Average Region Size</td>
<td>Average size of the region for the table(s). Average Region Size is calculated from average of average region sizes reported by each RegionServer (may not be the true average).</td>
</tr>
<tr>
<td>MEMSTORE SIZE</td>
<td>MemStore Size</td>
<td>Total MemStore size of the table(s).</td>
</tr>
<tr>
<td>STORE FILES</td>
<td>Store File Size</td>
<td>Total size of the store files (data files) for the table(s).</td>
</tr>
<tr>
<td></td>
<td>Num Store Files</td>
<td>Total number of store files for the table(s).</td>
</tr>
<tr>
<td>STORE FILE AGE</td>
<td>Max Store File Age</td>
<td>Maximum age of store files for the table(s). As compactions rewrite data, store files are also rewritten. Max Store File Age is calculated from the maximum of all maximum store file ages reported by each RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Min Store File Age</td>
<td>Minimum age of store files for the table(s). As compactions rewrite data, store files are also rewritten. Min Store File Age is calculated from the minimum of all minimum store file ages reported by each RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Average Store File Age</td>
<td>Average age of store files for the table(s). As compactions rewrite data, store files are also rewritten. Average Store File Age is calculated from the average of average store file ages reported by each RegionServer.</td>
</tr>
<tr>
<td></td>
<td>Num Reference Files - Total on All</td>
<td>Total number of reference files for the table(s).</td>
</tr>
<tr>
<td>NUM TOTAL REQUESTS</td>
<td>Num Total Requests /s on Tables</td>
<td>Total number of requests (both read and write) per second for the table(s).</td>
</tr>
<tr>
<td>NUM READ REQUESTS</td>
<td>Num Read Requests /s</td>
<td>Total number of read requests per second for the table(s).</td>
</tr>
<tr>
<td>NUM WRITE REQUESTS</td>
<td>Num Write Requests /s</td>
<td>Total number of write requests per second for the table(s).</td>
</tr>
<tr>
<td>NUM FLUSHES</td>
<td>Num Flushes /s</td>
<td>Total number of flushes per second for the table(s).</td>
</tr>
<tr>
<td>FLUSHED BYTES</td>
<td>Flushed MemStore Bytes</td>
<td>Total number of flushed MemStore bytes for the table(s).</td>
</tr>
<tr>
<td></td>
<td>Flushed Output Bytes</td>
<td>Total number of flushed output bytes for the table(s).</td>
</tr>
<tr>
<td>FLUSH TIME HISTOGRAM</td>
<td>Flush Time Mean</td>
<td>Mean latency for Flush operation for the table(s).</td>
</tr>
<tr>
<td></td>
<td>Flush Time Median</td>
<td>Median latency for Flush operation for the table(s).</td>
</tr>
<tr>
<td></td>
<td>Flush Time 95th Percentile</td>
<td>95th percentile latency for Flush operation for the table(s).</td>
</tr>
<tr>
<td></td>
<td>Flush Time Max</td>
<td>Maximum latency for Flush operation for the table(s).</td>
</tr>
<tr>
<td>FLUSH MEMSTORE SIZE HISTOGRAM</td>
<td>Flush MemStore Size Mean</td>
<td>Mean size of the MemStore for Flush operation for the table(s).</td>
</tr>
<tr>
<td></td>
<td>Flush MemStore Size Median</td>
<td>Median size of the MemStore for Flush operation for the table(s).</td>
</tr>
<tr>
<td></td>
<td>Flush Output Size 95th Percentile</td>
<td>95th percentile size of the MemStore for Flush operation for the table(s).</td>
</tr>
<tr>
<td></td>
<td>Flush MemStore Size Max</td>
<td>Max size of the MemStore for Flush operation for the table(s).</td>
</tr>
<tr>
<td>FLUSH OUTPUT SIZE HISTOGRAM</td>
<td>Flush Output Size Mean</td>
<td>Mean size of the output file for Flush operation for the table(s).</td>
</tr>
<tr>
<td></td>
<td>Flush Output Size Median</td>
<td>Median size of the output file for Flush operation for the table(s).</td>
</tr>
<tr>
<td>Row</td>
<td>Metrics</td>
<td>Description</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>Flush Output Size 95th Percentile</td>
<td>95th percentile size of the output file for Flush operation for the table(s).</td>
</tr>
<tr>
<td></td>
<td>Flush Output Size Max</td>
<td>Max size of the output file for Flush operation for the table(s).</td>
</tr>
</tbody>
</table>

### 9.1.3.8.5. HBase - Users

The HBase - Users dashboards display metrics and detailed data on a per-user basis across the cluster. You can click the second drop-down arrow in the upper-left corner to select a single user, a group of users, or all users, and you can change your user selection at any time.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUM REQUESTS - GET/SCAN</td>
<td>Num Get Requests /s</td>
<td>Total number of Get requests per second for the user(s).</td>
</tr>
<tr>
<td></td>
<td>Num Scan Next Requests /s</td>
<td>Total number of Scan requests per second for the user(s).</td>
</tr>
<tr>
<td>NUM REQUESTS - MUTATE/DELETE</td>
<td>Num Mutate Requests /s</td>
<td>Total number of Mutate requests per second for the user(s).</td>
</tr>
<tr>
<td></td>
<td>Num Delete Requests /s</td>
<td>Total number of Delete requests per second for the user(s).</td>
</tr>
<tr>
<td>NUM REQUESTS - APPEND/INCREMENT</td>
<td>Num Append Requests /s</td>
<td>Total number of Append requests per second for the user(s).</td>
</tr>
<tr>
<td></td>
<td>Num Increment Requests /s</td>
<td>Total number of Increment requests per second for the user(s).</td>
</tr>
</tbody>
</table>

### 9.1.3.9. Kafka Dashboards

The following Grafana dashboards are available for Kafka:

- Kafka - Home [157]
- Kafka - Hosts [158]
- Kafka - Topics [158]

#### 9.1.3.9.1. Kafka - Home

Metrics that show overall status for the Kafka cluster.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BYTES IN &amp; OUT / MESSAGES IN</td>
<td>Bytes In &amp; Bytes Out /sec</td>
<td>Rate at which bytes are produced into the Kafka cluster and the rate at which bytes are being consumed from the Kafka cluster.</td>
</tr>
<tr>
<td></td>
<td>Messages In /sec</td>
<td>Number of messages produced into the Kafka cluster.</td>
</tr>
<tr>
<td>CONTROLLER/LEADER COUNT &amp; REPLICA MAXLAG</td>
<td>Active Controller Count</td>
<td>Number of active controllers in the Kafka cluster. This should always equal one.</td>
</tr>
<tr>
<td></td>
<td>Replica MaxLag</td>
<td>Shows the lag of each replica from the leader.</td>
</tr>
<tr>
<td></td>
<td>Leader Count</td>
<td>Number of partitions for which a particular host is the leader.</td>
</tr>
<tr>
<td>UNDER REPLICATED PATRITIONS &amp; OFFLINE PARTITONS COUNT</td>
<td>Under Replicated Partitions</td>
<td>Indicates if any partitions in the cluster are under-replicated.</td>
</tr>
<tr>
<td></td>
<td>Offline Partitions Count</td>
<td>Indicates if any partitions are offline (which means that no leaders or replicas are available for producing or consuming).</td>
</tr>
<tr>
<td>PRODUCER &amp; CONSUMER REQUESTS</td>
<td>Producer Req /sec</td>
<td>Rate at which producer requests are made to the Kafka cluster.</td>
</tr>
</tbody>
</table>
### 9.1.3.9.2. Kafka - Hosts

Metrics that show operating status for Kafka cluster on a per broker level.

Use the drop-down menus to customize your results:

- Kafka broker
- Host
- Whether to view the largest (top) or the smallest (bottom) values
- Number of values that you want to view
- Aggregator to use: average, max value, or the sum of values

### 9.1.3.9.3. Kafka - Topics

Metrics related to Kafka cluster on a per topic level. Select a topic (by default, all topics are selected) to view the metrics for that topic.
### 9.1.3.10. Storm Dashboards

The following Grafana dashboards are available for Storm:

- Storm - Home [159]
- Storm - Topology [159]
- Storm - Components [160]

#### 9.1.3.10.1. Storm - Home

Metrics that show the operating status for Storm.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Topologies</td>
<td>Number of topologies in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Supervisors</td>
<td>Number of supervisors in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Total Executors</td>
<td>Total number of executors running for all topologies in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Total Tasks</td>
<td>Total number of tasks for all topologies in the cluster.</td>
</tr>
<tr>
<td>Unnamed</td>
<td>Free Slots</td>
<td>Number of free slots for all supervisors in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Used Slots</td>
<td>Number of used slots for all supervisors in the cluster.</td>
</tr>
<tr>
<td></td>
<td>Total Slots</td>
<td>Total number of slots for all supervisors in the cluster. Should be more than 0.</td>
</tr>
</tbody>
</table>

#### 9.1.3.10.2. Storm - Topology

Metrics that show the overall operating status for Storm topologies. Select a topology (by default, all topologies are selected) to view metrics for that topology.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>All Tasks Input/ Output</td>
<td><strong>Input Records</strong> is the number of input messages executed on all tasks, and <strong>Output Records</strong> is the number of messages emitted on all tasks.</td>
</tr>
<tr>
<td></td>
<td>All Tasks Aced Tuples</td>
<td>Number of messages ached (completed) on all tasks.</td>
</tr>
<tr>
<td></td>
<td>All Tasks Failed Tuples</td>
<td>Number of messages failed on all tasks.</td>
</tr>
<tr>
<td></td>
<td>All Spouts Latency</td>
<td>Average latency on all spout tasks.</td>
</tr>
<tr>
<td></td>
<td>All Tasks Queue</td>
<td><strong>Receive Queue Population</strong> is the total number of tuples waiting in the receive queue, and <strong>Send Queue Population</strong> is the total number of tuples waiting in the send queue.</td>
</tr>
<tr>
<td></td>
<td>All workers memory usage on heap</td>
<td>Used bytes on heap for all workers in topology.</td>
</tr>
</tbody>
</table>

---

159
9.1.3.10.3. Storm - Components

Metrics that show operating status for Storm topologies on a per component level. Select a topology and a component to view related metrics.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>All workers memory usage on non-heap</td>
<td>Used bytes on non-heap for all workers in topology.</td>
</tr>
<tr>
<td></td>
<td>All workers GC count</td>
<td>PSScavenge count is the number of occurrences for parallel scavenge collector and PSMarkSweep count is the number of occurrences for parallel scavenge mark and sweep collector.</td>
</tr>
<tr>
<td></td>
<td>All workers GC time</td>
<td>PSScavenge timeMs is the sum of the time parallel scavenge collector takes (in milliseconds), and PSMarkSweep timeMs is the sum of the time parallel scavenge mark and sweep collector takes (in milliseconds). Note that GC metrics are provided based on worker GC setting, so these metrics are only available for default GC option for worker.childopts. If you use another GC option for worker, you need to copy the dashboard and update the metric name manually.</td>
</tr>
</tbody>
</table>

9.1.3.11. System Dashboards

The following Grafana dashboards are available for System:

- System - Home [160]
- System - Servers [161]

9.1.3.11.1. System - Home

Metrics to see the overall status of the cluster.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OVERVIEW AVERAGES</td>
<td>Logical CPU Count Per Server</td>
<td>Average number of CPUs (including hyperthreading) aggregated for selected hosts.</td>
</tr>
<tr>
<td></td>
<td>Total Memory Per Server</td>
<td>Total system memory available per server aggregated for selected hosts.</td>
</tr>
<tr>
<td></td>
<td>Total Disk Space Per Server</td>
<td>Total disk space per server aggregated for selected hosts.</td>
</tr>
<tr>
<td>OVERVIEW - TOTALS</td>
<td>Logical CPU Count Total</td>
<td>Total Number of CPUs (including hyperthreading) aggregated for selected hosts.</td>
</tr>
</tbody>
</table>
### Row

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Total Memory</strong></td>
<td>Total system memory available per server aggregated for selected hosts.</td>
</tr>
<tr>
<td><strong>Total Disk Space</strong></td>
<td>Total disk space per server aggregated for selected hosts.</td>
</tr>
</tbody>
</table>

### CPU

**CPU Utilization - Average**

CPU utilization aggregated for selected hosts.

### SYSTEM LOAD

**System Load - Average**

Load average (1 min, 5 min and 15 min) aggregated for selected hosts.

### MEMORY

**Memory - Average**

Average system memory utilization aggregated for selected hosts.

**Memory - Total**

Total system memory available aggregated for selected hosts.

### DISK UTILIZATION

**Disk Utilization - Average**

Average disk usage aggregated for selected hosts.

**Disk Utilization - Total**

Total disk available for selected hosts.

### DISK IO

**Disk IO - Average**

Disk read/write counts (iops) co-related with bytes aggregated for selected hosts.

**Disk IO - Total**

Sum of read/write bytes/sec aggregated for selected hosts.

### NETWORK IO

**Network IO - Average**

Average Network statistics as MBps aggregated for selected hosts.

### NETWORK PACKETS

**Network IO - Total**

Sum of Network packets as MBps aggregated for selected hosts.

**Network Packets - Average**

Average of Network packets as KBps aggregated for selected hosts.

### SWAP/NUM PROCESSES

**Swap Space - Average**

Average swap space statistics aggregated for selected hosts.

**Num Processes - Average**

Average number of processes aggregated for selected hosts.

#### Note

- **Average** implies sum/count for values reported by all hosts in the cluster. Example: In a 30 second window, if 98 out of 100 hosts reported 1 or more value, it is the SUM(Avg value from each host + Interpolated value for 2 missing hosts)/100.

- **Sum/Total** implies the sum of all values in a timeslice (30 seconds) from all hosts in the cluster. The same interpolation rule applies.

### 9.1.3.11.2. System - Servers

Metrics to see the system status per host on the server.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CPU - USER/SYSTEM</strong></td>
<td>CPU Utilization - User</td>
<td>CPU utilization per user for selected hosts.</td>
</tr>
<tr>
<td></td>
<td>CPU Utilization - System</td>
<td>CPU utilization per system for selected hosts.</td>
</tr>
<tr>
<td><strong>CPU - NICE/IDLE</strong></td>
<td>CPU Utilization - Nice</td>
<td>CPU nice (Unix) time spent for selected hosts.</td>
</tr>
</tbody>
</table>
## Row | Metrics | Description
---|---|---
| CPU Utilization - Idle | CPU idle time spent for selected hosts. |
| CPU Utilization - iowait | CPU IO wait time for selected hosts. |
| CPU Utilization - Hardware Interrupt | CPU IO interrupt execute time for selected hosts. |
| CPU Utilization - Software Interrupt | CPU time spent processing soft irqs for selected hosts. |
| CPU Utilization - Steal (VM) | CPU time spent processing steal time (virtual cpu wait) for selected hosts. |
| System Load Average - 1 Minute | 1 minute load average for selected hosts. |
| System Load Average - 5 Minute | 5 minute load average for selected hosts. |
| System Load Average - 15 Minute | 15 minute load average for selected hosts. |
| Memory - Total | Total memory in GB for selected hosts. |
| Memory - Used | Used memory in GB for selected hosts. |
| Memory - Free | Total free memory in GB for selected hosts. |
| Memory - Cached | Total cached memory in GB for selected hosts. |
| Memory - Buffer | Total buffered memory in GB for selected hosts. |
| Memory - Shared | Total shared memory in GB for selected hosts. |
| Disk Used | Disk space used in GB for selected hosts. |
| Disk Free | Disk space available in GB for selected hosts. |
| Read Bytes | IOPS as read MBps for selected hosts. |
| Write Bytes | IOPS as write MBps for selected hosts. |
| Read Count | IOPS as read count for selected hosts. |
| Write Count | IOPS as write count for selected hosts. |
| Network Bytes Received | Network utilization as byte/sec received for selected hosts. |
| Network Bytes Sent | Network utilization as byte/sec sent for selected hosts. |
| Network Packets Received | Network utilization as packets received for selected hosts. |
| Network Packets Sent | Network utilization as packets sent for selected hosts. |
| Swap Space - Total | Total swap space available for selected hosts. |
| Swap Space - Free | Total free swap space for selected hosts. |
| Num Processes - Total | Count of processes and total running processes for selected hosts. |
| Num Processes - Runnable | Count of processes and total running processes for selected hosts. |

### 9.1.3.12. NiFi Dashboard

The following Grafana dashboard is available for NiFi:

- NiFi-Home [163]
9.1.3.12.1. NiFi-Home

You can use the following metrics to assess the general health of your NiFi cluster.

For all metrics available in the NiFi-Home dashboard, the single value you see is the average of the information submitted by each node in your NiFi cluster.

<table>
<thead>
<tr>
<th>Row</th>
<th>Metrics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JVM Info</td>
<td>JVM Heap Usage</td>
<td>Displays the amount of memory being used by the JVM process. For NiFi, the default configuration is 512 MB.</td>
</tr>
<tr>
<td></td>
<td>JVM File Descriptor Usage</td>
<td>Shows the number of connections to the operating system. You can monitor this metric to ensure that your JVM file descriptors, or connections, are opening and closing as tasks complete.</td>
</tr>
<tr>
<td></td>
<td>JVM Uptime</td>
<td>Displays how long a Java process has been running. You can use this metric to monitor Java process longevity, and any unexpected restarts.</td>
</tr>
<tr>
<td>Thread Info</td>
<td>Active Threads</td>
<td>NiFi has two user configurable thread pools:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Maximum timer driven thread count (default 10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Maximum event driven thread count (default 5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This metrics displays the number of active threads from these two pools.</td>
</tr>
<tr>
<td></td>
<td>Thread Count</td>
<td>Displays the total number of threads for the JVM process that is running NiFi. This value is larger than the two pools above, because NiFi uses more than just the timer and event driven threads.</td>
</tr>
<tr>
<td></td>
<td>Daemon Thread Count</td>
<td>Displays the number of daemon threads that are running. A daemon thread is a thread that does not prevent the JVM from exiting when the program finishes, even if the thread is still running.</td>
</tr>
<tr>
<td>FlowFile Info</td>
<td>FlowFiles Received</td>
<td>Displays the number of FlowFiles received into NiFi from an external system in the last 5 minutes.</td>
</tr>
<tr>
<td></td>
<td>FlowFiles Sent</td>
<td>Displays the number of FlowFiles sent from NiFi to an external system in the last 5 minutes.</td>
</tr>
<tr>
<td></td>
<td>FlowFiles Queued</td>
<td>Displays the number of FlowFiles queued in a NiFi processor connection.</td>
</tr>
<tr>
<td>Byte Info</td>
<td>Bytes Received</td>
<td>Displays the number of bytes of FlowFile data received into NiFi from an external system, in the last 5 minutes.</td>
</tr>
<tr>
<td></td>
<td>Bytes Sent</td>
<td>Displays the number of bytes of FlowFile data sent from NiFi to an external system, in the last 5 minutes.</td>
</tr>
<tr>
<td></td>
<td>Bytes Queued</td>
<td>Displays the number of bytes of FlowFile data queued in a NiFi processor connection.</td>
</tr>
</tbody>
</table>

9.1.4. AMS Performance Tuning

To set up Ambari Metrics System in your environment, review and customize the following Metrics Collector configuration options.

- Customizing the Metrics Collector Mode [164]
- Customizing TTL Settings [165]
- Customizing Memory Settings [166]
9.1.4.1. Customizing the Metrics Collector Mode

**Metrics Collector** is built using Hadoop technologies such as Apache HBase, Apache Phoenix, and Apache Traffic Server (ATS). The Collector can store metrics data on the local file system, referred to as *embedded mode*, or use an external HDFS, referred to as *distributed mode*. By default, the Collector runs in embedded mode. In embedded mode, the Collector captures and writes metrics to the local file system on the host where the Collector is running.

**Important**

When running in embedded mode, you should confirm that `hbase.rootdir` and `hbase.tmp.dir` have adequately sized and lightly used partitions. Directory configurations in *Ambari Metrics > Configs > Advanced > ams-hbase-site* are using a sufficiently sized and not heavily utilized partition, such as:

```
file:///grid/0/var/lib/ambari-metrics-collector/hbase.
```

You should also confirm that the TTL settings are appropriate.

When the Collector is configured for distributed mode, it writes metrics to HDFS, and the components run in distributed processes, which helps to manage CPU and memory consumption.

To switch the Metrics Collector from embedded mode to distributed mode,

**Steps**

1. In *Ambari Web*, browse to *Services > Ambari Metrics > Configs*.

2. Change the values of listed properties to the values shown in the following table:

<table>
<thead>
<tr>
<th>Configuration Section</th>
<th>Property</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
<td>Metrics Service operation mode (timeline.metrics.service.operation.mode)</td>
<td>Designates whether to run in distributed or embedded mode.</td>
<td>distributed</td>
</tr>
<tr>
<td>Advanced ams-hbase-site</td>
<td>hbase.cluster.distributed</td>
<td>Indicates AMS will run in distributed mode.</td>
<td>true</td>
</tr>
<tr>
<td>Advanced ams-hbase-site</td>
<td>hbase.rootdir</td>
<td>The HDFS directory location where metrics will be stored.</td>
<td>hdfs://$NAMENODE_FQDN:8020/apps/ams/metrics</td>
</tr>
</tbody>
</table>

3. Using *Ambari Web > Hosts > Components* restart the Metrics Collector.

If your cluster if configured for a highly available NameNode, set the `hbase.rootdir` value to use the HDFS name service instead of the NameNode host name:

```
hdfs://hdfsnameservice/apps/ams/metrics
```
Optionally, you can migrate existing data from the local store to HDFS prior to switching to distributed mode:

### Steps

1. Create an HDFS directory for the ams user:

   ```
   su - hdfs -c 'hdfs dfs -mkdir -p /apps/ams/metrics'
   ```

2. Stop Metrics Collector.

3. Copy the metric data from the AMS local directory to an HDFS directory. This is the value of hbase.rootdir in Advanced ams-hbase-site used when running in embedded mode. For example:

   ```
   su - hdfs -c 'hdfs dfs -copyFromLocal /var/lib/ambari-metrics-collector/hbase/* /apps/ams/metrics'
   su - hdfs -c 'hdfs dfs -chown -R ams:hadoop /apps/ams/metrics'
   ```

4. Switch to distributed mode.

5. Restart the Metrics Collector.

If you are working with Apache HBase cluster metrics and want to display the more granular metrics of HBase cluster performance on the individual region, table, or user level, see .

### More Information

- Customizing Cluster-Environment-Specific Settings [166]
- Customizing TTL Settings [165]
- Enabling Individual Region, Table, and User Metrics for HBase

#### 9.1.4.2. Customizing TTL Settings

AMS enables you to configure Time To Live (TTL) for aggregated metrics by navigating to Ambari Metrics > Configs > Advanced ams-site. Each property name is self explanatory and controls the amount of time to keep metrics (in seconds) before they are purged. The values for these TTL’s are set in seconds.

For example, assume that you are running a single-node sandbox and want to ensure that no values are stored for more than seven days, to reduce local disk space consumption. In this case, you can set to 604800 (seven days, in seconds) any property ending in .ttl that has a value greater than 604800.

You likely want to do this for properties such as timeline.metrics.cluster.aggregator.daily.ttl, which controls the daily aggregation TTL and is set by default to two years. Two other properties that consume a lot of disk space are

- timeline.metrics.cluster.aggregator.minute.ttl, which controls minute-level aggregated metrics TTL, and
- timeline.metrics.host.aggregator.ttl, which controls host-based precision metrics TTL.
If you are working in an environment prior to Apache Ambari 2.1.2, you should make these settings during installation; otherwise, you must use the HBase shell by running the following command from the Collector host:

```
/usr/lib/ams-hbase/bin/hbase --config /etc/ams-hbase/conf shell
```

After you are connected, you must update each of the following tables with the TTL value:

```
hbase(main):000:0> alter 'METRIC_RECORD_DAILY', { NAME => '0', TTL => 604800}:
```

<table>
<thead>
<tr>
<th>Map This TTL Property...</th>
<th>To This HBase Table...</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeline.metrics.cluster.aggregator.daily.ttl</td>
<td>METRIC_AGGREGATE_DAILY</td>
</tr>
<tr>
<td>timeline.metrics.cluster.aggregator.hourly.ttl</td>
<td>METRIC_AGGREGATE_HOURLY</td>
</tr>
<tr>
<td>timeline.metrics.cluster.aggregator.minute.ttl</td>
<td>METRIC_AGGREGATE</td>
</tr>
<tr>
<td>timeline.metrics.host.aggregator.daily.ttl</td>
<td>METRIC_RECORD_DAILY</td>
</tr>
<tr>
<td>timeline.metrics.host.aggregator.hourly.ttl</td>
<td>METRIC_RECORD_HOURLY</td>
</tr>
<tr>
<td>timeline.metrics.host.aggregator.minute.ttl</td>
<td>METRIC_RECORD_MINUTE</td>
</tr>
<tr>
<td>timeline.metrics.host.aggregator.ttl</td>
<td>METRIC_RECORD</td>
</tr>
</tbody>
</table>

### 9.1.4.3. Customizing Memory Settings

Because AMS uses multiple components (such as Apache HBase and Apache Phoenix) for metrics storage and query, multiple tunable properties are available to you for tuning memory use:

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advanced ams-env</td>
<td>metrics_collector_heapsize</td>
<td>Heap size configuration for the Collector.</td>
</tr>
<tr>
<td>Advanced ams-hbase-env</td>
<td>hbase_regionserver_heapsize</td>
<td>Heap size configuration for the single AMS HBase Region Server.</td>
</tr>
<tr>
<td>Advanced ams-hbase-env</td>
<td>hbase_master_heapsize</td>
<td>Heap size configuration for the single AMS HBase Master.</td>
</tr>
<tr>
<td>Advanced ams-hbase-env</td>
<td>regionserver_xmn_size</td>
<td>Maximum value for the young generation heap size for the single AMS HBase RegionServer.</td>
</tr>
<tr>
<td>Advanced ams-hbase-env</td>
<td>hbase_master_xmn_size</td>
<td>Maximum value for the young generation heap size for the single AMS HBase Master.</td>
</tr>
</tbody>
</table>

### 9.1.4.4. Customizing Cluster-Environment-Specific Settings

The Metrics Collector mode, TTL settings, memory settings, and disk space requirements for AMS depend on the number of nodes in the cluster. The following table lists specific recommendations and tuning guidelines for each.

<table>
<thead>
<tr>
<th>Cluster Environment</th>
<th>Host Count</th>
<th>Disk Space</th>
<th>Collector Mode</th>
<th>TTL</th>
<th>Memory Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-Node Sandbox</td>
<td>1</td>
<td>2GB</td>
<td>embedded</td>
<td>Reduce TTLs to 7 Days</td>
<td>metrics_collector_heap_size=1024</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>hbase_regionserver_heapsize=512</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>hbase_master_heapsize=512</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>hbase_master_xmn_size=128</td>
</tr>
<tr>
<td>PoC</td>
<td>1-5</td>
<td>5GB</td>
<td>embedded</td>
<td>Reduce TTLs to 30 Days</td>
<td>metrics_collector_heap_size=1024</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>hbase_regionserver_heapsize=512</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>hbase_master_heapsize=512</td>
</tr>
</tbody>
</table>
### 9.1.4.5. Moving the Metrics Collector

Use this procedure to move the Ambari Metrics Collector to a new host:

1. In Ambari Web, stop the Ambari Metrics service.

2. Execute the following API call to delete the current Metric Collector component:

   ```bash
   ```

3. Execute the following API call to add Metrics Collector to a new host:

<table>
<thead>
<tr>
<th>Cluster Environment</th>
<th>Host Count</th>
<th>Disk Space</th>
<th>Collector Mode</th>
<th>TTL</th>
<th>Memory Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-Production</td>
<td>5-20</td>
<td>20GB</td>
<td>embedded</td>
<td>Reduce TTLs to 3 Months</td>
<td>metrics_collector_heap_size=1024</td>
</tr>
<tr>
<td>Production</td>
<td>20-50</td>
<td>50GB</td>
<td>embedded</td>
<td>n.a.</td>
<td>metrics_collector_heap_size=1024</td>
</tr>
<tr>
<td>Production</td>
<td>50-200</td>
<td>100GB</td>
<td>embedded</td>
<td>n.a.</td>
<td>metrics_collector_heap_size=2048</td>
</tr>
<tr>
<td>Production</td>
<td>200-400</td>
<td>200GB</td>
<td>embedded</td>
<td>n.a.</td>
<td>metrics_collector_heap_size=2048</td>
</tr>
<tr>
<td>Production</td>
<td>400-800</td>
<td>200GB</td>
<td>distributed</td>
<td>n.a.</td>
<td>metrics_collector_heap_size=8192</td>
</tr>
<tr>
<td>Production</td>
<td>800+</td>
<td>500GB</td>
<td>distributed</td>
<td>n.a.</td>
<td>metrics_collector_heap_size=12288</td>
</tr>
</tbody>
</table>
curl -u admin:admin -H "X-Requested-By:ambari" -i -X
POST http://ambari.server:8080/api/v1/clusters/cluster.name/
hosts/metrics.collector.hostname/host_components/METRICS_COLLECTOR

4. In Ambari Web, go the page of the host on which you installed the new Metrics Collector and click Install the Metrics Collector.

5. In Ambari Web, start the Ambari Metrics service.

Note

Restarting all services is not required after moving the Ambari Metrics Collector, using Ambari 2.5 and later.

9.1.4.6. (Optional) Enabling Individual Region, Table, and User Metrics for HBase

Other than HBase RegionServer metrics, Ambari disables per region, per table, and per user metrics by default, because these metrics can be numerous and therefore cause performance issues.

If you want Ambari to collect these metrics, you can re-enable them; however, you should first test this option and confirm that your AMS performance is acceptable.

1. On the Ambari Server, browse to the following location:

   /var/lib/ambari-server/resources/common-services/
   HBASE/0.96.0.2.0/package/templates

2. Edit the following template files:

   hadoop-metrics2-hbase.properties-GANGLIA-MASTER.j2
   hadoop-metrics2-hbase.properties-GANGLIA-RS.j2

3. Either comment out or remove the following lines:

   *.source.filter.class=org.apache.hadoop.metrics2.filter.RegexFilter
   hbase.*.source.filter.exclude=.*(Regions|Users|Tables).*

4. Save the template files and restart Ambari Server for the changes to take effect.

Important

If you upgrade Ambari to a newer version, you must re-apply this change to the template file.

9.1.5. AMS High Availability (Technical Preview)

Ambari installs the Ambari Metrics System (AMS), into the cluster with a single Metrics Collector component by default. The Collector is a daemon that runs on a specific host in the cluster and receives data from the registered publishers, the Monitors and Sinks. Depending on your needs, you might require AMS to have two Collectors to cover a High
Availability scenario. This section describes the steps to enable AMS High Availability. The AMS high availability capability isTechnical Preview.

Prerequisite

You must deploy AMS in distributed (not embedded) mode.

To provide AMS High Availability:

Steps

1. In Ambari Web, browse to the host where you would like to install another collector.

2. On the Host page, choose +Add.

3. Select Metrics Collector from the list.

Ambari installs the new Metrics Collector and configures Ambari Metrics for HA.

The new Collector will be installed in a “stopped” state.

4. In Ambari Web, will have to start the new Collector component from Ambari Web.
Note

If you attempt to add a second Collector to the cluster without first switching AMS to distributed mode, the collector will install but will not be able to be started.

```
Traceback (most recent call last):
  File "/var/lib/ambari-agent/cache/common-services/AMBARI_METRICS/0.1.0/package/scripts/metrics_collector.py", line 150, in <module>
    AmsCollector().execute()
  File "/usr/lib/python2.6/site-packages/resource_management/libraries/script/script.py", line 313, in execute
    method(env)
  File "/var/lib/ambari-agent/cache/common-services/AMBARI_METRICS/0.1.0/package/scripts/metrics_collector.py", line 48, in start
    self.configure(env, action = 'start') # for security
  File "/usr/lib/python2.6/site-packages/resource_management/libraries/script/script.py", line 116, in locking_configure
    original_configure(obj, *args, **kw)
  File "/var/lib/ambari-agent/cache/common-services/AMBARI_METRICS/0.1.0/package/scripts/metrics_collector.py", line 42, in configure
    raise Fail("AMS in embedded mode cannot have more than 1 instance. Delete all but 1 instances or switch to Distributed mode ")
resource_management.core.exceptions.Fail: AMS in embedded mode cannot have more than 1 instance. Delete all but 1 instances or switch to Distributed mode
```

Workaround: Delete the newly added Collector, enable distributed mode, then re-add the Collector.

More Information

AMS Architecture [119]

Customizing the Metrics Collector Mode [164]

9.1.6. AMS Security

The following sections describe tasks to be performed when setting up security for the Ambari Metrics System.

- Changing the Grafana Admin Password [170]
- Set Up HTTPS for AMS [171]
- Set Up HTTPS for Grafana [174]

9.1.6.1. Changing the Grafana Admin Password

If you need to change the Grafana Admin password after you initially install Ambari, you have to change the password directly in Grafana, and then make the same change in the Ambari Metrics configuration:
Steps

1. In Ambari Web, browse to Services > Ambari Metrics select Quick Links, and then choose Grafana.

   The Grafana UI opens in read-only mode.

2. Click Sign In, in the left column.

3. Log in as admin, using the unchanged password.

4. Click the admin label in the left column to view the admin profile, and then click Change password.

5. Enter the unchanged password, enter and confirm the new password, and click Change Password.

6. Return to Ambari Web > Services > Ambari Metrics and browse to the Configs tab.

7. In the General section, update and confirm the Grafana Admin Password with the new password.

8. Save the configuration and restart the services, as prompted.

9.1.6.2. Set Up HTTPS for AMS

   If you want to limit access to AMS to HTTPS connections, you must provide a certificate. While it is possible to use a self-signed certificate for initial trials, it is not suitable for production environments. After you get your certificate, you must run a special setup command.

Steps

1. Create your own CA certificate.

   openssl req -new -x509 -keyout ca.key -out ca.crt -days 365

2. Import CA certificate into the truststore.

   # keytool -keystore /<path>/truststore.jks -alias CARoot -import -file ca.crt -storepass bigdata

3. Check truststore.

   # keytool -keystore /<path>/truststore.jks -list
   Enter keystore password:

   Keystore type: JKS
   Keystore provider: SUN

   Your keystore contains 2 entries
   caroot, Feb 22, 2016, trustedCertEntry,
   Certificate fingerprint (SHA1):
You should see trustedCertEntry for CA.

4. Generate certificate for AMS Collector and store private key in keystore.

```bash
# keytool -genkey -alias c6401.ambari.apache.org -keyalg RSA -keysize 1024
-dname "CN=c6401.ambari.apache.org,OU=IT,O=Apache,L=US,ST=US,C=US" -keypass bigdata -keystore /<path>/keystore.jks -storepass bigdata
```

**Note**

If you use an alias different than the default hostname (c6401.ambari.apache.org), then, in step 12, set the ssl.client.truststore.alias config to use that alias.

5. Create certificate request for AMS collector certificate.

```bash
keytool -keystore /<path>/keystore.jks -alias c6401.ambari.apache.org -certreq -file c6401.ambari.apache.org.csr -storepass bigdata
```

6. Sign the certificate request with the CA certificate.

```bash
openssl x509 -req -CA ca.crt -CAkey ca.key -in c6401.ambari.apache.org.csr -out c6401.ambari.apache.org_signed.crt -days 365 -CAcreateserial -passin pass:bigdata
```

7. Import CA certificate into the keystore.

```bash
keytool -keystore /<path>/keystore.jks -alias CARoot -import -file ca.crt -storepass bigdata
```

8. Import signed certificate into the keystore.

```bash
keytool -keystore /<path>/keystore.jks -alias c6401.ambari.apache.org -import -file c6401.ambari.apache.org_signed.crt -storepass bigdata
```

9. Check keystore.

```bash
caroot2, Feb 22, 2016, trustedCertEntry,
Certificate fingerprint (SHA1):
[root@c6401 tmp]# keytool -keystore /tmp/keystore.jks -list
Enter keystore password:
Keystore type: JKS
Keystore provider: SUN

Your keystore contains 2 entries

caroot, Feb 22, 2016, trustedCertEntry,
Certificate fingerprint (SHA1):
c6401.ambari.apache.org, Feb 22, 2016, PrivateKeyEntry,
Certificate fingerprint (SHA1):
```

You should see PrivateKeyEntry for the ams collector hostname entry and trustedCertEntry for CA.
10. Copy /<path>/truststore.jks to all nodes to /<path>/truststore.jks and set appropriate access permissions.

11. Copy /<path>/keystore.jks to AMS collector node ONLY to /<path>/keystore.jks and set appropriate access permissions. Recommended: set owner to ams user and access permissions to 400.


   • ams-ssl-server/ssl.server.keystore.keypassword=bigdata
   • ams-ssl-server/ssl.server.keystore.location=/<path>/keystore.jks
   • ams-ssl-server/ssl.server.keystore.password=bigdata
   • ams-ssl-server/ssl.server.keystore.type=jks
   • ams-ssl-server/ssl.server.truststore.location=/<path>/truststore.jks
   • ams-ssl-server/ssl.server.truststore.password=bigdata
   • ams-ssl-server/ssl.server.truststore.reload.interval=10000
   • ams-ssl-server/ssl.server.truststore.type=jks
   • ams-ssl-client/ssl.client.truststore.location=/<path>/truststore.jks
   • ams-ssl-client/ssl.client.truststore.password=bigdata
   • ams-ssl-client/ssl.client.truststore.type=jks
   • ssl.client.truststore.alias=<Alias used to create certificate for AMS. (Default is hostname)>

13. Restart services with stale configs.

14. Configure Ambari server to use truststore.

```
# ambari-server setup-security
Using python /usr/bin/python
Security setup options...

Choose one of the following options:
  [5] Import certificate to truststore.

Enter choice, (1-5): 4
Do you want to configure a truststore [y/n] (y)?
TrustStore type [jks/jceks/pkcs12] (jks):jks
Path to TrustStore file :/<path>/keystore.jks
Password for TrustStore:
Re-enter password:
Ambari Server 'setup-security' completed successfully.
```
15. Configure ambari server to use https instead of http in requests to AMS Collector by adding "server.timeline.metrics.https.enabled=true" to ambari.properties file.

```
# echo "server.timeline.metrics.https.enabled=true" >> /etc/ambari-server/conf/ambari.properties
```

16. Restart ambari server.

### 9.1.6.3. Set Up HTTPS for Grafana

If you want to limit access to the Grafana to HTTPS connections, you must provide a certificate. While it is possible to use a self-signed certificate for initial trials, it is not suitable for production environments. After you get your certificate, you must run a special setup command.

**Steps**

1. Log on to the host with Grafana.

2. Browse to the Grafana configuration directory:

   ```
   cd /etc/ambari-metrics-grafana/conf/
   ```

3. Locate your certificate.

   If you want to create a temporary self-signed certificate, you can use this as an example:

   ```
   openssl genrsa -out ams-grafana.key 2048
   openssl req -new -key ams-grafana.key -out ams-grafana.csr
   openssl x509 -req -days 365 -in ams-grafana.csr -signkey ams-grafana.key -out ams-grafana.crt
   ```

4. Set the certificate and key file ownership and permissions so that they are accessible to Grafana:

   ```
   chown ams:hadoop ams-grafana.crt
   chown ams:hadoop ams-grafana.key
   chmod 400 ams-grafana.crt
   chmod 400 ams-grafana.key
   ```

   For a non-root Ambari user, use

   ```
   chmod 444 ams-grafana.crt
   ```

   to enable the agent user to read the file.

5. In Ambari Web, browse to > Services > Ambari Metrics > Configs.

6. Update the following properties in the Advanced ams-grafana-ini section:

   ```
   protocol https
   cert_file /etc/ambari-metrics-grafana/conf/ams-grafana.crt
   cert-Key /etc/ambari-metrics-grafana/conf/ams-grafana.key
   ```

7. Save the configuration and restart the services as prompted.
9.2. Ambari Log Search (Technical Preview)

The following sections describe the Technical Preview release of Ambari Log Search, which you should use only in non-production clusters with fewer than 150 nodes.

- Log Search Architecture [175]
- Installing Log Search [176]
- Using Log Search [176]

9.2.1. Log Search Architecture

Ambari Log Search enables you to search for logs generated by Ambari-managed HDP components. Ambari Log Search relies on the Ambari Infra service to provide Apache Solr indexing services. Two components compose the Log Search solution:

- Log Feeder [175]
- Log Search Server [176]

9.2.1.1. Log Feeder

The Log Feeder component parses component logs. A Log Feeder is deployed to every node in the cluster and interacts with all component logs on that host. When started, the Log Feeder begins to parse all known component logs and sends them to the Apache Solr instances (managed by the Ambari Infra service) to be indexed.

By default, only FATAL, ERROR, and WARN logs are captured by the Log Feeder. You can temporarily or permanently add other log levels using the Log Search UI filter settings (for temporary log level capture) or through the Log Search configuration control in Ambari.
9.2.1.2. Log Search Server

The Log Search Server hosts the Log Search UI web application, providing the API that is used by Ambari and the Log Search UI to access the indexed component logs. After logging in as a local or LDAP user, you can use the Log Search UI to visualize, explore, and search indexed component logs.

9.2.2. Installing Log Search

Log Search is a built-in service in Ambari 2.4 and later. You can add it during a new installation by using the +Add Service menu. The Log Feeders are automatically installed on all nodes in the cluster; you manually place the Log Search Server, optionally on the same server as the Ambari Server.

9.2.3. Using Log Search

Using Ambari Log Search includes the following activities:

- Accessing Log Search [176]
- Using Log Search to Troubleshoot [178]
- Viewing Service Logs [178]
- Viewing Access Logs [179]

9.2.3.1. Accessing Log Search

After Log Search is installed, you can use any of three ways to search the indexed logs:

- Ambari Background Ops Log Search Link [176]
- Host Detail Logs Tab [177]
- Log Search UI [177]

Note

Single Sign On (SSO) between Ambari and Log Search is not currently available.

9.2.3.1.1. Ambari Background Ops Log Search Link

When you perform lifecycle operations such as starting or stopping services, it is critical that you have access to logs that can help you recover from potential failures. These logs are now available in Background Ops. Background Ops also links to the Host Detail Logs tab, which lists all the log files that have been indexed and can be viewed for a specific host:
More Information

Background Ops

9.2.3.1.2. Host Detail Logs Tab

A Logs tab is added to each host detail page, containing a list of indexed, viewable log files, organized by service, component, and type. You can open and search each of these files by using a link to the Log Search UI:

9.2.3.1.3. Log Search UI

The Log Search UI is a purpose-built web application used to search HDP component logs. The UI is focussed on helping operators quickly access and search logs from a single location. Logs can be filtered by log level, time, component, and can be searched by keyword. Helpful tools such as histograms to show number of logs by level for a time period are available, as well as controls to help rewind and fast forward search sessions, contextual click to include/exclude terms in log viewing, and multi-tab displays for troubleshooting multi-component and host issues.

The Log Search UI is available from the Quick Links menu of the Log Search Service within Ambari Web.
To see a guided tour of Log Search UI features, choose Take a Tour from the Log Search UI main menu. Click Next to view each topic in the guided tour series.

9.2.3.2. Using Log Search to Troubleshoot

To find logs related to a specific problem, use the Troubleshooting tab in the UI to select the service, components, and time frame related to the problem you are troubleshooting. For example, if you select HDFS, the UI automatically searches for HDFS-related components. You can select a time frame of yesterday or last week, or you can specify a custom value. Each of these specifications filters the results to match your interests. When you are ready to view the matching logs, you can click Go to Logs:

9.2.3.3. Viewing Service Logs

The Service Logs tab enables you to search across all component logs for specific keywords and to filter for specific log levels, components, and time ranges. The UI is organized so that you can quickly see how many logs were captured for each log level across the entire cluster, search for keywords, include and exclude components, and match logs to your search query:
9.2.3.4. Viewing Access Logs

When troubleshooting HDFS-related issues, you might find it helpful to search for and spot trends in HDFS access by users. The Access Logs tab enables you to view HDFS Audit log entries for a specific time frame, to see aggregated usage data showing the top ten HDFS users by file system resources accessed, as well as the top ten file system resources accessed across all users. This can help you find anomalies or hot and cold data sets.

9.3. Ambari Infra

Many services in HDP depend on core services to index data. For example, Apache Atlas uses indexing services for tagging lineage-free text search, and Apache Ranger uses indexing for audit data. The role of Ambari Infra is to provide these common shared services for stack components.

Currently, the Ambari Infra Service has only one component: the Infra Solr Instance. The Infra Solr Instance is a fully managed Apache Solr installation. By default, a single-node SolrCloud installation is deployed when the Ambari Infra Service is chosen for installation; however, you should install multiple Infra Solr Instances so that you have distributed indexing and search for Atlas, Ranger, and LogSearch (Technical Preview).

To install multiple Infra Solr Instances, you simply add them to existing cluster hosts through Ambari’s +Add Service capability. The number of Infra Solr Instances you deploy depends on the number of nodes in the cluster and the services deployed.
Because one Ambari Infra Solr Instance is used by multiple HDP components, you should be careful when restarting the service, to avoid disrupting those dependent services. In HDP 2.5 and later, Atlas, Ranger, and Log Search (Technical Preview) dependent on Ambari Infra Service.

Note

Infra Solr Instance is intended for use only by HDP components; use by third-party components or applications is not supported.