

administering ambari views 2

## Administering Ambari Views

**Date of Publish:** 2018-04-30



<http://docs.hortonworks.com>

# Contents

<b>Understanding Ambari Views.....</b>	<b>3</b>
<b>Ambari Views terminology.....</b>	<b>4</b>
<b>Increase memory available to Ambari Views server.....</b>	<b>5</b>
<b>Review the number of expected concurrent Ambari Views users.....</b>	<b>5</b>
<b>Configure a trust store for the Ambari Views server.....</b>	<b>5</b>
<b>Increase timeout value for Ambari Views server.....</b>	<b>6</b>
<b>Run a remote, standalone Ambari Views server.....</b>	<b>7</b>
<b>Comparing standalone and operational Ambari server set up.....</b>	<b>7</b>
<b>Running standalone Ambari Views servers behind a reverse proxy.....</b>	<b>8</b>
<b>Prepare to set up a remote, standalone Ambari Views server.....</b>	<b>8</b>
<b>Configuring Ambari View Instances.....</b>	<b>8</b>
<b>Create an Ambari View instance.....</b>	<b>9</b>
<b>Migrate Ambari View instance data.....</b>	<b>10</b>
<b>Create an Ambari View URL.....</b>	<b>10</b>
<b>Set Ambari View Permissions.....</b>	<b>11</b>
<b>Configure Ambari Views for Kerberos.....</b>	<b>11</b>

## Understanding Ambari Views

from a Development, Persona, Versions and Deployment perspective.

Apache Ambari includes the Ambari Views Framework, which enables developers to create UI components, or Views, that “plug into” the Ambari Web interface. Ambari automatically creates and presents to users some instances of Views, if the service used by that View is added to the cluster. For example, if Apache YARN service is added to the cluster, the YARN Queue Manager View displays to Ambari web users. In other cases, the Ambari Admin user must manually create a view instance.

Developing and using Views enables you to extend and customize the Ambari web to meet your specific needs.

Using Views also extends your Ambari implementation to allow third parties to plug in new resource types, along with APIs, providers, and UIs to support them. Views are deployed on the Ambari Server, which enables Ambari Admins to create View instances and set access privileges for users and groups.

The following sections describe the basics of Views and how to deploy and manage View instances in Ambari:

Views are basically web applications that can be “plugged in to” Ambari. Just like a typical web application, a View can include server-side resources and client-side assets. Server-side resources, which are written in Java, can integrate with external systems (such as cluster services) and expose REST end-points that are used by the view. Client-side assets, such as HTML, JavaScript, and CSS, provide the UI for the view that is rendered in the Ambari web interface.

Development:

Ambari Views Framework Ambari exposes the Views Framework as the basis for View development. The Framework provides the following:

- Method for describing and packaging a View
- Method for deploying a View
- Framework services for a View to integrate with Ambari
- Method for managing View versions, instances, and permissions

The Ambari Views framework is separate from Views themselves. The framework is a core feature of Ambari that you use to create, deploy, integrate, and manage your own, custom views.

You develop and deliver a view by performing the following tasks:

- Develop the View (similar to how you would build a web application)
- Package the View (similar to a WAR)
- Deploy the View to Ambari (using the Ambari Administration interface)
- Create and configure instances of the View (performed by Ambari Admins)

Persona:

Three user persona interact with Views:

### **View developer**

Person who builds the front end and back end of a View and uses the framework services available during development. The developer creates the View, resulting in a View package that is delivered to an Ambari Admin.

### **Ambari Admin**

Ambari user that has Ambari Admin privilege and uses the Views Management section of the Ambari Administration interface to create and managing instances of Views. Ambari Admin also deploys the View packages delivered by the View developer.

### **View user**

Ambari user that has access to one or more Views in the Ambari web (basically, the end user).

**Versions:**

Each View must have a unique name, although it can have one or more View versions. Each View name and version combination is a single *View package*. After a View package is deployed, Ambari Admins can create *View instances*, each of which is identified by a unique View instance name. The Ambari Admin can then set access permissions for each View instance.

**Deployment:**

Views can be deployed and managed in the *operational Ambari Server*, the Ambari Server operating your cluster. Alternatively, Views can be deployed and managed in one or more separate *standalone Ambari Servers*. Running standalone Ambari Server instances is useful when users who will access views will not have (and should not) have access to the operational Ambari Server. You can run one or more separate standalone Ambari Server instances to scale-out your solution for handling a large number of users.

The following Ambari views currently available to you:

<b>Yarn Queue Manager View</b>	Provides a visual way to configure YARN capacity scheduler queue capacity.
<b>Files View</b>	Allows you to browse the HDFS file system.
<b>SmartSense View</b>	Allows you to capture bundles, set bundle capture schedule, and view and download captured bundles.
<b>Workflow Manager View</b>	Allows you to easily create and schedule workflows and monitor workflow jobs.

Subsequent chapters in this guide describe tasks performed by an Ambari Administrator to make Views available to users in their Ambari-managed cluster. This guide does not describe View development and packaging.

**Related Information**

[Understanding cluster roles](#)

## Ambari Views terminology

The following are Views terms you should be familiar with:

<b>Views framework</b>	The core framework that is used to develop a View: similar to a Java web application.
<b>View definition</b>	The View resources and core View properties, such as name, version, and any necessary configuration properties. Ambari reads View definition during deployment.
<b>View package</b>	A bundle of View client and server assets (and dependencies) that is ready to deploy to Ambari.
<b>View deployment</b>	The process of instantiating a View instance in Ambari, which makes that View available to Ambari Admins for creating instances.
<b>View name</b>	The unique identifier for a View. A View can have one or more versions. The name is defined in the View Definition (created by the View Developer) and built into the View Package.

<b>View version</b>	The uniquely named version of a View. Multiple versions of a View (uniquely identified by View name) can be deployed to Ambari.
<b>View instance</b>	The instantiation of a specific View version. Instances are created and configured by Ambari Admins and must have a unique View instance name.
<b>View instance name</b>	The unique identifier of a specific instance of a View.
<b>framework services</b>	View context, instance data, configuration properties, and events

## Increase memory available to Ambari Views server

You must increase the amount of memory available to the Ambari server hosting views.

### About this task

This is particularly true if you intend to deploy and use multiple views concurrently.

### Before you begin

- Review the amount of memory available to the Ambari server that hosts views for your cluster.
- Review whether your Ambari server is configured for HTTPS.

### Procedure

1. On the Ambari Server host, edit the `ambari-env.sh` file.  
`vi /var/lib/ambari-server/ambari-env.sh`
2. For the `AMBARI_JVM_ARGS` variable, replace the default `-Xmx2048m` with the following value:  
`-Xmx4096m -XX:PermSize=128m -XX:MaxPermSize=128m`
3. Restart the server.  
`ambari-server restart`

## Review the number of expected concurrent Ambari Views users

Consider the following guidance when planning for Views user capacity. An Ambari Views server:

### Procedure

- On an 8-core box with 16GB of RAM and `client.threadpool.size.max = 100` can handle approximately 40 concurrent users
- On an 16-core box with 32GB of RAM and `client.threadpool.size.max = 100` can handle approximately 60 concurrent users

### What to do next

The recommended best practice is scaling multiple Ambari Views servers horizontally, behind a load balancer.

## Configure a trust store for the Ambari Views server

If your Ambari Server instance is configured for HTTPS, you must configure a trust store for the Ambari Views server.

**About this task**

Configure a trust store so that the deployed views accept the certificate used by the Ambari Server during API communications. To configure a trust store:

**Procedure**

1. On the Ambari Server, create a new keystore to contain the server's HTTPS certificate:  
keytool -import -file <path\_to\_the\_Ambari\_Server's\_SSL\_Certificate> -alias ambari-server -keystore ambari-server-truststore
2. When prompted, trust the certificate by typing yes.
3. Configure the server to use the new trust store.

```
ambari-server setup-security
Using python /usr/bin/python2.6
Security setup options...
=====
Choose one of the following options:
  [1] Enable HTTPS for Ambari server.
  [2] Encrypt passwords stored in ambari.properties file.
  [3] Setup Ambari kerberos JAAS configuration.
  [4] Setup truststore.
  [5] Import certificate to truststore.
=====
Enter choice, (1-5): 4
Do you want to configure a truststore [y/n] (y)? y
TrustStore type [jks/jceks/pkcs12] (jks): jks
Path to TrustStore file : <path_to_the_ambari-server-truststore keystore>
Password for TrustStore:
Re-enter password:
Ambari Server 'setup-security' completed successfully.
```

4. Restart the server.  
ambari-server restart

## Increase timeout value for Ambari Views server

If you experience timeouts or long wait times, you can increase the timeout values to lower response times.

**About this task**

The `views.request.read.timeout.millis` property in `/etc/ambari-server/conf/ambari.properties` sets the timeout value for requests made by the Ambari Views server to non-ambari services, such as webHcat, or Hive. By default, `views.request.read.timeout.millis` is set to 10 seconds.

The `views.ambari.request.read.timeout.millis` property in `/etc/ambari-server/conf/ambari.properties` sets the timeout values for requests made by Ambari views to Ambari services. By default, `views.ambari.request.read.timeout.millis` is set to 5 seconds.

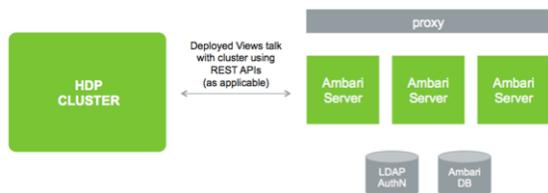
Usually no action is required. However, if you experience timeouts, or long wait times, you can increase the values for each of these properties to lower response times.

**Procedure**

1. On the Ambari server host, browse to `/etc/ambari-server/conf/ambari.properties`.
2. Adjust the value of `views.request.read.timeout.millis`.

## Run a remote, standalone Ambari Views server

A recommended strategy that limits user access to your operational Ambari Server while managing a large number of Views users is to set up one or more standalone Ambari Servers. You can configure your operational Ambari Server as a remote cluster, then use the Remote Cluster option when configuring each view instance. A diagram of this strategy follows:



### Related Information

[Register a remote cluster](#)

## Comparing standalone and operational Ambari server set up

Setting up a standalone Ambari Server instance is very similar to setting up an operational Ambari Server instance. Many of the steps are the same, with one key exception: you do not install a cluster using a standalone server instance. A standalone Ambari Server instance does not manage a cluster and does not deploy or communicate with Ambari Agents; instead, a standalone Ambari Server runs as web server instance, serving views for users.

The following table compares the high-level tasks required to set up an operational Ambari Server and a standalone Ambari server:

**Table 1: Operational and Standalone Ambari Servers, Compared**

	Operational Ambari Server	Standalone Ambari Server
1	Install ambari-server package	Install ambari-server package
2	Run ambari-server setup (DB, JDK)	Run ambari-server setup (DB, JDK)  <b>Important:</b> Do not share the DB with an Operational Ambari Server.
3	Configure external LDAP authentication	Configure external LDAP authentication
4	Install cluster	Do not install cluster
5	Deploy views	Deploy views
6	Create and configure view instances	Create and configure view instances
7		(Optional) Repeat for each Ambari Server instance
8		(Optional) Set up proxy for Ambari Server instances
9		(Optional) Set up SSL for Ambari

### Related Information

[Apache Ambari Installation](#)

[Configuring Ambari for LDAP or AD authentication](#)

[Set up SSL for Ambari](#)

## Running standalone Ambari Views servers behind a reverse proxy

If you require many users to access Ambari views, you should install and run multiple standalone Ambari Server instances behind a reverse proxy.

In this case, the reverse proxy must honor *session affinity*, meaning that after a session is established, the reverse proxy routes each subsequent request to the same Ambari server instance. Depending on the reverse proxy implementation, you can achieve session affinity in several different ways, including hashing client IP and using the JSESSIONID header.



### Important:

Using a reverse proxy is supported only for standalone Ambari Server instances.

Using multiple, operational Ambari Sever instances behind a reverse proxy is not supported.

## Prepare to set up a remote, standalone Ambari Views server

### About this task

When setting up multiple standalone Ambari Server instances, you must be aware of the following requirements:

### Procedure

- All Ambari Server instances should be the same version.
- The main Ambari Server should be on its own database instance.  
All other standalone instances should share the same one. Ensure that it is not the same database that is being used by the Operational Ambari Server that is managing the HDP cluster.
- The Ambari database should be scaled and highly available, independent of Ambari Server.
- For an external authentication source such as LDAP or Active Directory, Ambari Server authentication should be identical for all instances.
- If the cluster that Views users access is Kerberos-enabled, you must configure Ambari and the views for Kerberos.
- You must run each standalone Ambari Server instance behind a reverse proxy.

### Related Information

[Register a remote cluster](#)

## Configuring Ambari View Instances

When creating a View instance, you specify some basic configuration information about the view and configure the view to communicate with a cluster.

Based on the resources managed by your Ambari Server, choose one of three options when completing the Cluster Configuration section; Local Cluster, Remote Cluster, or Custom. Use the following descriptions to guide your choice.

### Local Cluster

If you are configuring a view instance in an Ambari Server that is also managing a cluster, you can select **Local Cluster**. When you select this option, Ambari

**Remote Cluster**

automatically determines the cluster configuration properties required.

If your Ambari Server is not managing a cluster, then you must select either **Remote Cluster** or **Custom**.

If you plan to configure a view to work with a cluster that is remote from an Ambari Server and that cluster is being managed by Ambari, you should select **Remote Cluster**.

Registering a Remote Cluster enables the Remote Cluster option. When you select the Remote Cluster option for a view instance, Ambari automatically determines the cluster configuration properties required for the view instance. Be sure the Remote Cluster includes all services required for the view you are configuring.

**Custom**

If your cluster is remote from and not being managed by the Ambari Server running the view, you must select **Custom** and then manually configure the view to work with the cluster.

You can use the following table to help determine which options are available for view configuration:

**Table 2: Locating a View Relative to a Cluster**

If you are working in this scenario...	Choose this option...
Your cluster is managed by a local Ambari Server that is also running the view	Local Cluster
Your cluster is managed by Ambari and your cluster is remote from the standalone Ambari Server running the view	Remote Cluster
Your cluster is remote from the standalone Ambari Server running the view and your cluster is not managed by Ambari.	Custom

## Create an Ambari View instance

To create a View instance:

### Procedure

1. On the **Ambari Admin** page, browse to a View and expand it.
2. Click **Create Instance**.
3. Provide the following information:

**Table 3: Required and Optional Elements for a View**

Item	Required?	Description
View version	Yes	Exact version to instantiate
Instance name	Yes	Name unique to the selected View
Display label	Yes	Readable display name of the View instance in Ambari Web
Description	Yes	Readable description of the View instance in Ambari Web

Item	Required?	Description
Visible	No	Whether the View is visible or not visible to the end-user in Ambari Web Use this property to temporarily hide a view from users.
Settings	Maybe	Depending on the View, a group of settings that can be customized If a setting is required, you are prompted to provide the required information.
Cluster configuration	Maybe	Depending on the View, you can choose a local or remote cluster, or manually configure a custom View.

### What to do next

If Ambari has a cluster configured that will work with the View instance, then the choice of **Local Cluster** will be available. If you have registered one or more Remote Clusters, then the choice of **Remote Cluster** will also be available. If neither local or remote clusters are available, you will have to enter the **Custom** configuration manually.

## Migrate Ambari View instance data

Migrating data from one view to another is useful when a new view version is released and you want to use data from the previous version in the newer version.

### About this task

If you have more than one instance of the same Ambari View, you can migrate view data such as entity data, instance data, and View use permissions from one instance to another.



**Important:** Migrating view data between instances is supported only for Hive, Pig and Tez views.

For example, consider a case to migrate from view INSTANCE A (the source view instance) to view INSTANCE B (the target view instance).

### Procedure

- In this case, you run the following command:  

```
curl -v -u admin:admin -X PUT -H "X-Requested-By:1" http://AMBARI_SERVER_HOST:8080/api/v1/views/VIEW_NAME/versions/TARGET_VIEW_VERSION/instances/INSTANCEB/migrate/SOURCE_VIEW_VERSION/INSTANCEA
```

The command values are as follows:

- AMBARI\_SERVER\_HOST is the Ambari Server host name or IP address.
- VIEW\_NAME is the name of the view.
- TARGET\_VIEW\_VERSION is the version of the target view.
- SOURCE\_VIEW\_VERSION is the version of the source view.

For example, if you are migrating from version 1.0.0 to 1.0.1, your SOURCE\_VIEW\_VERSION is 1.0.0 and TARGET\_VIEW\_VERSION is 1.0.1.

## Create an Ambari View URL

After creating a View instance, you should create a URL by which to access it, based on the view name, version, and instance name.

**About this task**

You can also create a short URL of your choosing. You can copy and embed View URLs to provide user access to specific view instances.

**Procedure**

1. In the **Ambari Admin** page, browse to the **View URLs** section.
2. Click **Create New URL**.
3. Enter a URL name, select the view, select the instance, and (optionally) type a short URL.  
Short URLs must include only lowercase, alphanumeric characters.
4. Click **Save**.

## Set Ambari View Permissions

An Ambari Admin must specify which users and groups can use the View or, on a local cluster, specify permissions based on cluster roles.

**About this task**

By default, a new View instance has no permissions set. An Admin can also set permissions other than those required to use a View.

**Procedure**

1. Browse to a view and expand it.
2. Click the name of the view instance you want to modify.
3. In the **Permissions** section, click the **Users or Groups** control.
4. Modify the user and group lists, as appropriate.
5. Click the check mark to save changes.

**What to do next**

The Views Framework provides a way for view developers to specify custom permissions, beyond just the default Use permission. If custom permissions are specified, they will show up in the Ambari Admin interface and the Ambari Admin can set users and groups on these permissions. View permissions can also be inherited from Cluster roles. If you are using a Local Cluster for view configuration, you can optionally choose to provide view Use permission based on cluster roles.

## Configure Ambari Views for Kerberos

**About this task**

If the cluster that your views communicate with is Kerberos-enabled, you must:

**Procedure**

- Configure all Ambari Server instances for Kerberos.
- Configure each view for Kerberos.
- Install the Kerberos client utilities on the Ambari Server so that Ambari can kinit.

<b>OS Family</b>	<b>Install Command</b>
<b>RHEL/CentOS/Oracle Linux</b>	<code>yum install krb5-workstation</code>

**OS Family****Install Command****SLES**`zypper install krb5-client`**Ubuntu/Debian**`apt-get install krb5-user krb5-config`

- If a view requires HDFS or WebHCat to be configured for a proxy user, you must use the primary Kerberos principal as that user, instead of the ambari-server daemon user.  
For example, if you configure Ambari Server for Kerberos principal `ambari-server@EXAMPLE.COM`, this value would be `ambari-server`.

**What to do next**

Follow specific instructions to configure each view for Kerberos, and the cluster for Kerberos access from the view.

**Related Information**

[Enabling Kerberos Authentication using Ambari](#)