

Providing Authorization with Apache Ranger

Date of Publish: 2018-07-15

Contents

Using Ranger to Provide Authorization in Hadoop.....	4
Ranger Policies Overview.....	4
Ranger Tag-Based Policies.....	4
Tags and Policy Evaluation.....	5
Apache Ranger Access Conditions.....	7
Using the Ranger Console.....	9
Opening and Closing the Ranger Console.....	10
Ranger Console Navigation.....	11
Resource-Based Services and Policies.....	14
Configuring Resource-Based Services.....	14
Configure a Resource-based Service: HBase.....	15
Configure a Resource-based Service: HDFS.....	17
Configure a Resource-based Service: Hive.....	19
Configure a Resource-based Service: Kafka.....	22
Configure a Resource-based Service: Knox.....	24
Configure a Resource-based Service: Solr.....	25
Configure a Resource-based Service: Storm.....	27
Configure a Resource-based Service: YARN.....	28
Configure a Resource-based Service: Atlas.....	30
Configuring Resource-Based Policies.....	31
Configure a Resource-based Policy: HBase.....	32
Configure a Resource-based Policy: HDFS.....	35
Configure a Resource-based Policy: Hive.....	37
Configure a Resource-based Policy: Kafka.....	41
Configure a Resource-based Policy: Knox.....	43
Configure a Resource-based Policy: Solr.....	45
Configure a Resource-based Policy: Storm.....	47
Configure a Resource-based Policy: YARN.....	50
Configure a Resource-based Policy: Atlas.....	52
Wildcards and Variables in Resource-based Policies.....	54
Importing and Exporting Resource-Based Policies.....	55
Import Resource-Based Policies for a Specific Service.....	57
Import Resource-Based Policies for All Services.....	59
Export Resource-Based Policies for a Specific Service.....	61
Export All Resource-Based Policies for All Services.....	62
Row-level Filtering and Column Masking in Hive.....	63
Row-level Filtering in Hive with Ranger Policies.....	63
Dynamic Resource-Based Column Masking in Hive with Ranger Policies.....	67
Dynamic Tag-Based Column Masking in Hive with Ranger Policies.....	71
Tag-Based Services and Policies.....	74
Adding a Tag-based Service.....	74

Adding Tag-Based Policies.....	76
Using Tag Attributes and Values in Ranger Tag-Based Policy Conditions.....	79
Adding a Tag-Based PII Policy.....	80
Tag Policy Default EXPIRES ON Policy.....	84
Importing and Exporting Tag-Based Policies.....	86
Import Tag Based Policies.....	88
Export Tag-Based Policies.....	92
Create a Time-bound Policy.....	93
Administering Ranger Users/Groups and Permissions.....	95
Add a User.....	96
Edit a User.....	97
Delete a User.....	100
Add a Group.....	101
Edit a Group.....	103
Delete a Group.....	105
Add/Edit Permissions.....	106
Administering Ranger Reports.....	107
View Ranger Reports.....	107
Search Ranger Reports.....	108
Export Reports.....	109
Editing Ranger Policies from the Reports Page.....	110
Adding a New Component to Apache Ranger.....	110
Configuring Advanced Authorization Settings.....	112
Developing a Custom Authorization Module.....	113
Special Requirements for High Availability Environments.....	114
Configure Advanced Usersync Settings.....	114
Configure User Sync LDAP SSL.....	117
Set Up Database Users Without Sharing DBA Credentials.....	118
Updating Ranger Admin Passwords.....	118
Ranger Password Requirements.....	119

Using Ranger to Provide Authorization in Hadoop

Ranger manages access control through a user interface that ensures consistent policy administration across Hadoop data access components. Security administrators can define security policies at the database, table, column, and file levels, and can administer permissions for specific LDAP-based groups or individual users. Rules based on dynamic conditions such as time or geolocation can also be added to an existing policy rule. The Ranger authorization model is pluggable and can be easily extended to any data source using a service-based definition.

Once a user has been authenticated, their access rights must be determined. Authorization defines user access rights to resources. For example, a user may be allowed to create a policy and view reports, but not allowed to edit users and groups. You can use Ranger to set up and manage access to Hadoop services.

Ranger enables you to create services for specific Hadoop resources (HDFS, HBase, Hive, etc.) and add access policies to those services. You can also create tag-based services and add access policies to those services. Using tag-based policies enables you to control access to resources across multiple Hadoop components without creating separate services and policies in each component. You can also use Ranger TagSync to synchronize the Ranger tag store with an external metadata service such as Apache Atlas.

For more information on Ranger authorization, see the “HDP Security Features” Authorization overview.

Related Information

[HDP Security Features](#)

Ranger Policies Overview

Ranger has two types of policies: resource-based and tag-based.

Resource-based policies

Ranger enables you to configure resource-based services (HDFS, HBase, Hive, etc.) and add access policies to those services.

Tag-based policies

Ranger enables you to create tag-based services and add access policies to those services.

Ranger Tag-Based Policies

Ranger enables you to create tag-based services and add access policies to those services.

Tag-Based Policies Overview

- An important feature of Ranger tag-based authorization is the separation of resource-classification from access-authorization. For example, resources (HDFS file/directory, Hive database/table/column etc.) containing sensitive data such as social security numbers, credit card numbers, or sensitive health care data can be tagged with PII/PCI/PHI – either as the resource enters the Hadoop ecosystem or at a later time. Once a resource is tagged, the authorization for the tag would be automatically enforced, thus eliminating the need to create or update policies for the resource.
- Using tag-based policies also enables you to control access to resources across multiple Hadoop components without creating separate services and policies in each component.
- Tag details are stored in a tag store. Ranger TagSync can be used to synchronize the tag store with an external metadata service such as Apache Atlas.

Tag Store

Details of tags associated with resources are stored in a tag store. Apache Ranger plugins retrieve the tag details from the tag store for use during policy evaluation. To minimize the performance impact during policy evaluation (in finding tags for resources), Apache Ranger plugins cache the tags and periodically poll the tag store for any changes. When a change is detected, the plugins update the cache. In addition, the plugins store the tag details in a local cache file – just as the policies are stored in a local cache file. On component restart, the plugins will use the tag data from the local cache file if the tag store is not reachable.

Apache Ranger plugins download the tag details from the store managed by Ranger Admin. Ranger Admin persists the tag details in its policy store and provides a REST interface for the plugins to download the tag details.

Tags

Ranger Tags can have attributes. Tag attribute values can be used in Ranger tag-based policies to influence the authorization decision.

For example, to deny access to a resource after a specific date:

1. Add the EXPIRES_ON tag to the resource.
2. Add an expiry_date tag attribute and set its value to the expiry date.
3. Create a Ranger policy for the EXPIRES_ON tag.
4. Add a condition in this policy to deny access when the date specified in the expiry_date tag attribute is later than the current date.

Note that the EXPIRES_ON tag policy is created as the default policy in tag service instances.

TagSync

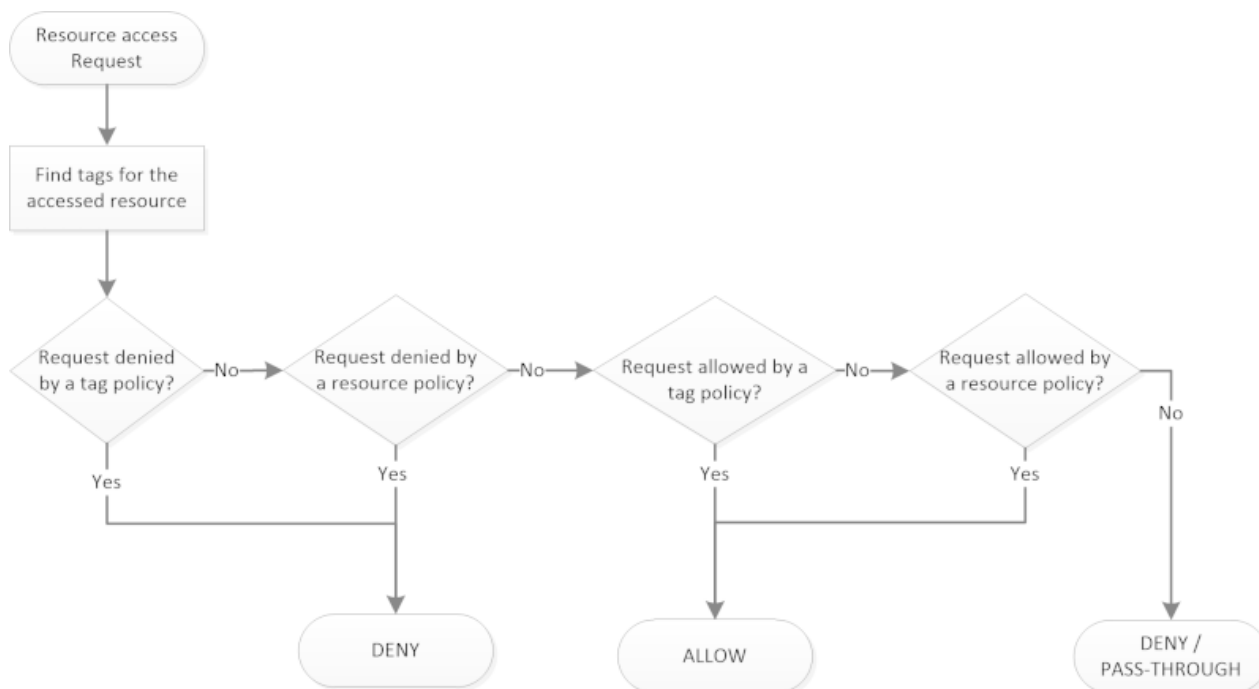
Ranger TagSync is used to synchronize the tag store with an external metadata service such as Apache Atlas. TagSync is a daemon process similar to the Ranger UserSync process.

Ranger TagSync receives tag details from Apache Atlas via change notifications. As tags are added to, updated, or deleted from resources in Apache Atlas, Ranger TagSync receives notifications and updates the tag store.

Tags and Policy Evaluation

When authorizing an access request, an Apache Ranger plugin evaluates applicable Ranger policies for the resource being accessed. The following diagram shows the details of the policy evaluation flow. More details on the steps in this workflow are provided in the subsequent sections.

Apache Ranger Policy Evaluation Flow with Tags



Apache Ranger Policy Evaluation Flow with Tags

Finding Tags

Apache Ranger supports a service to register context enrichers, which are used to update context data to the access request.

The Ranger Tag service, which is part of the tag-based policies feature, adds a context enricher named `RangerTagEnricher`. This context enricher is responsible for finding tags for the requested resource and adding the tag details to the request context. This context enricher keeps a cache of the available tags; while processing an access request, it finds the tags applicable for the requested resource and adds the tags to the request context. The context enricher keeps the cache updated by periodically polling Ranger Admin for changes.

Evaluating Tag-Based Policies

Once the list of tags for the requested resource is found, the Apache Ranger policy engine evaluates the tag-based policies applicable to the tags. If a policy for one of these tag results in a deny, access will be denied. If none of the tags are denied, and if a policy allows for one of the tags, access will be allowed. If there is no result for any tag, or if there are no tags for the resource, the policy engine will evaluate the resource-based policies to make the authorization decision.

Using Tags in Conditions

Apache Ranger allows the use of custom conditions while evaluating authorization policies. The Apache Ranger policy engine makes various request details – such as user, groups, resource, and context – available to the conditions. Tags in the request context, which are added by the enricher, are available to the conditions and can be used to influence the authorization decision.

The default policy in tag service instances, the `EXPIRES_ON` tag, uses such condition to check to see if the request date is later than the value specified in tag attribute `expiry_date`. This default policy does not work unless an `EXPIRES_ON` tag has been created in Atlas.

Related Information

[Apache Ranger Wiki> Context Enrichers](#)

Apache Ranger Access Conditions

The Apache Ranger access policy model consists of two major components: specification of the resources a policy is applied to, such as HDFS files and directories, Hive databases, tables, and columns, HBase tables, column-families, and columns, and so on; and the specification of access conditions for specific users and groups

Allow Deny and Exclude Conditions

Apache Ranger supports the following access conditions:

- Allow
- Exclude from Allow
- Deny
- Exclude from Deny

These access conditions enable you to set up fine-grained access control policies.

For example, you can allow access to a "finance" database to all users in the "finance" group, but deny access to all users in the "interns" group. Let's say that one of the members of the "interns" group, "scott", needs to work on an assignment that requires access to the "finance" database. In that case, you can add an Exclude from Deny condition that will allow user "scott" to access the "finance" database. The following image shows how this policy would be set up in Apache Ranger:

Policy Details :

Policy ID: 15

Policy Name: finance database enabled

Hive Database: finance Include

table: * Include **Resource**

Hive Column: * Include

Description: authorization for finance database

Audit Logging: YES

Allow Conditions

Allow Conditions:

Select Group	Select User	Permissions	Delegate Admin	
finance	Select User	All	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Exclude from Allow Conditions:

Deny Conditions

Deny Conditions:

Select Group	Select User	Permissions	Delegate Admin	
interns	Select User	All	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Exclude from Deny Conditions:

Deny Excludes

Exclude from Deny Conditions:

Select Group	Select User	Permissions	Delegate Admin	
Select Group	scot*	select	<input type="checkbox"/>	<input type="checkbox"/>

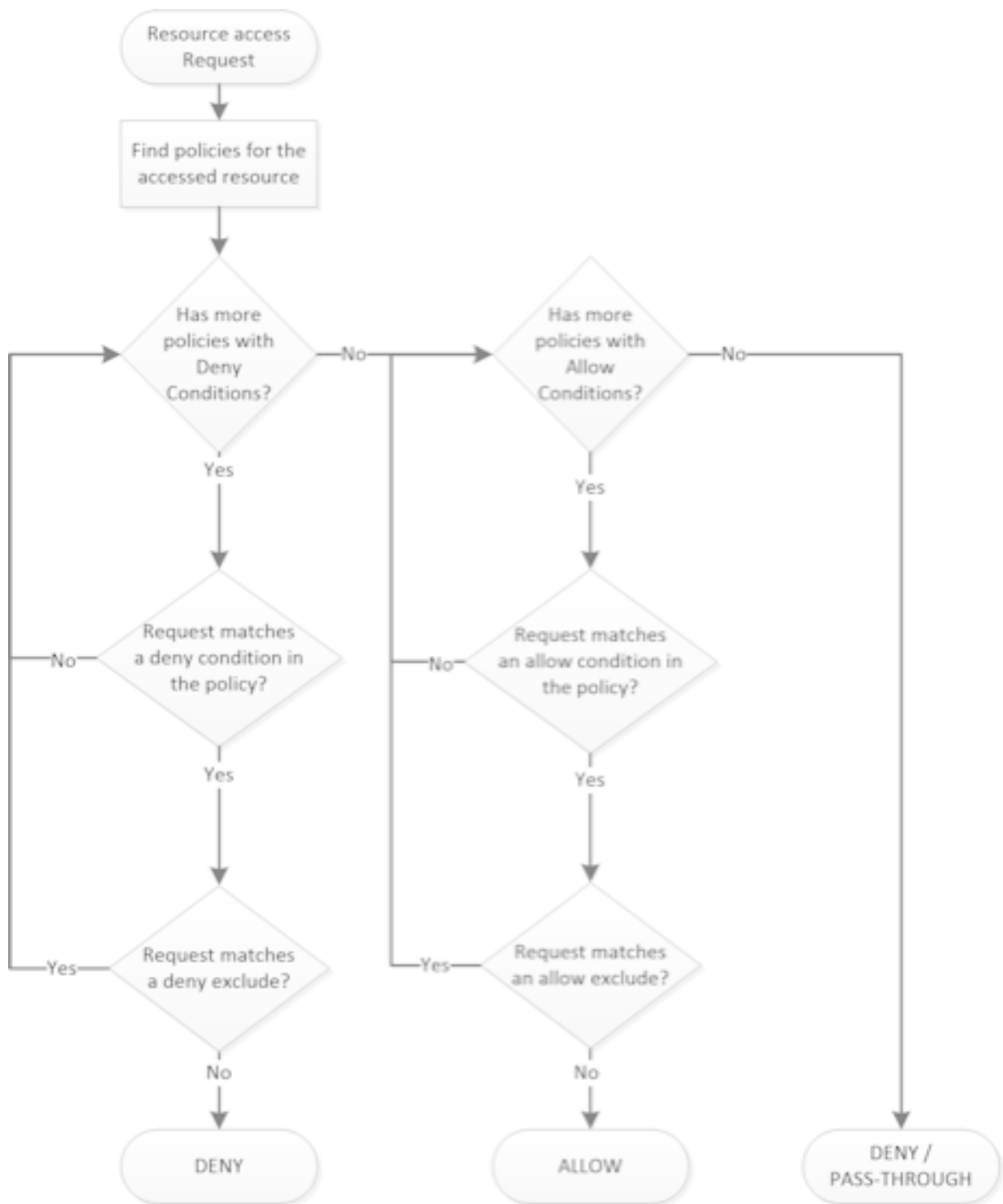
Enable Deny Conditions for Policies

The deny condition in policies is disabled by default and must be enabled for use.

1. From Ambari>Ranger>Configs>Advanced>Custom ranger-admin-site, add ranger.servicedef.enableDenyAndExceptionsInPolicies=true .
2. Restart Ranger.

Policy Evaluation of Access Conditions

Apache Ranger policies are evaluated in a specific order to ensure predictable results (if there is no access policy that allows access, the authorization request will typically be denied). The following diagram shows the policy evaluation work-flow:



Apache Ranger Policy Evaluation Flow

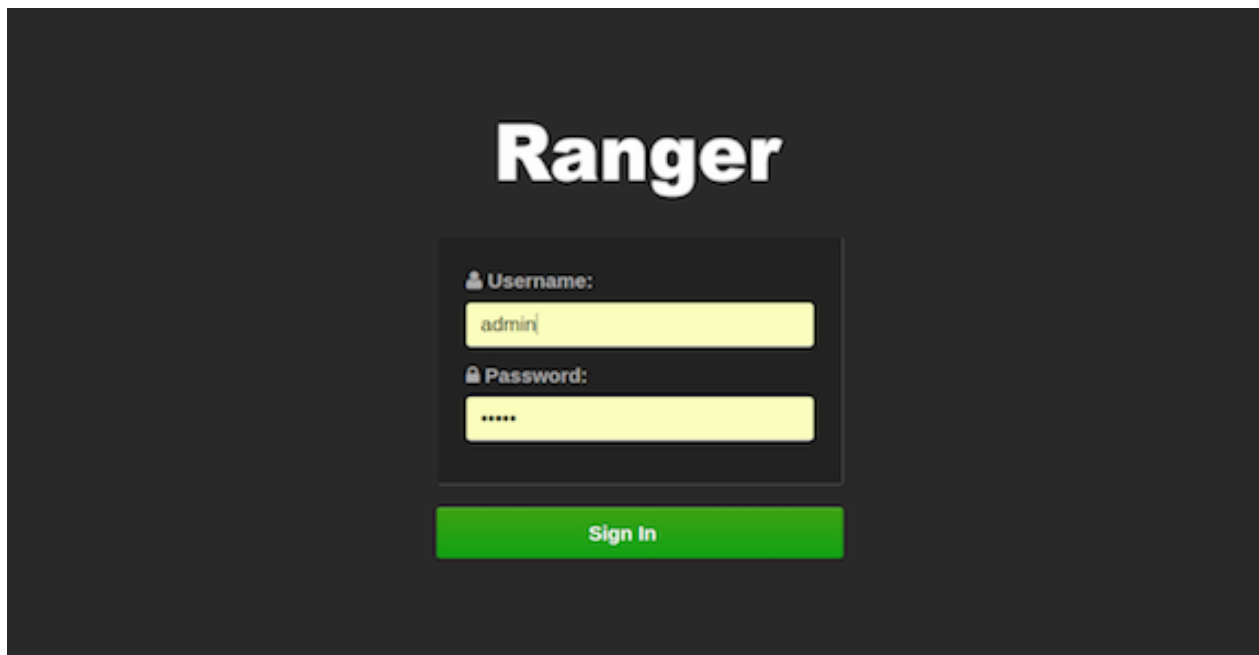
Using the Ranger Console

This chapter contains an overview of the Ranger console.

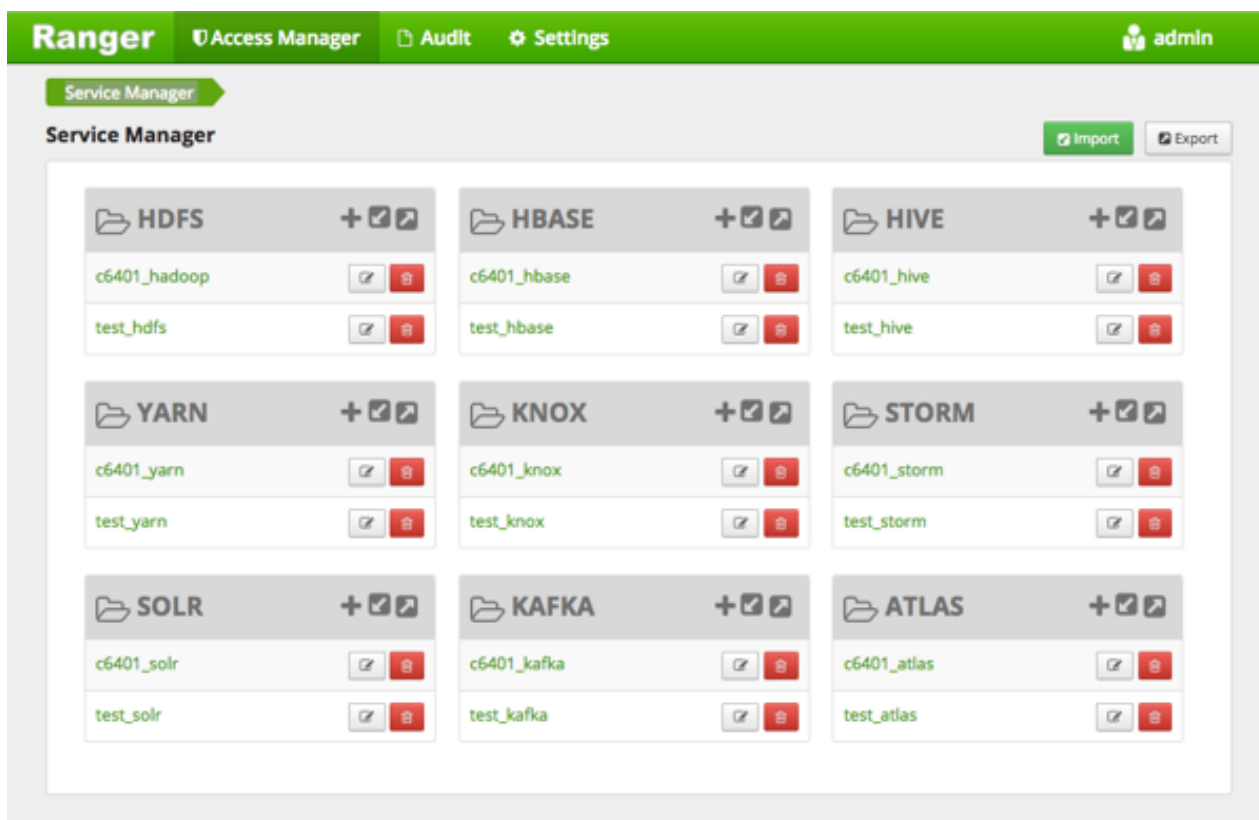
Opening and Closing the Ranger Console

Overview of how to open and close the Ranger console.

To open the Ranger Console, log in to the Ranger portal at `http://<your_ranger_server_address>:6080` (E.G., `http://dw-weekly.field.hortonworks.com:6080`). To log in, enter your user name and password, then click Sign In.

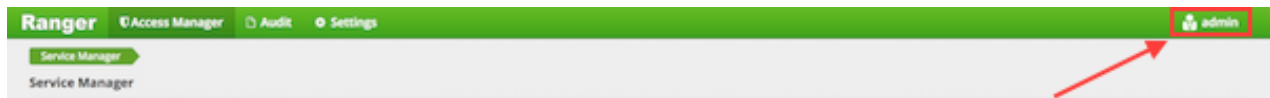


Ranger Console Home Page

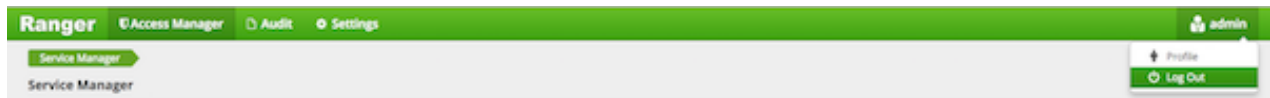


Ranger Login Console

After you log in, your user name is displayed at the top right of the Ranger Console.



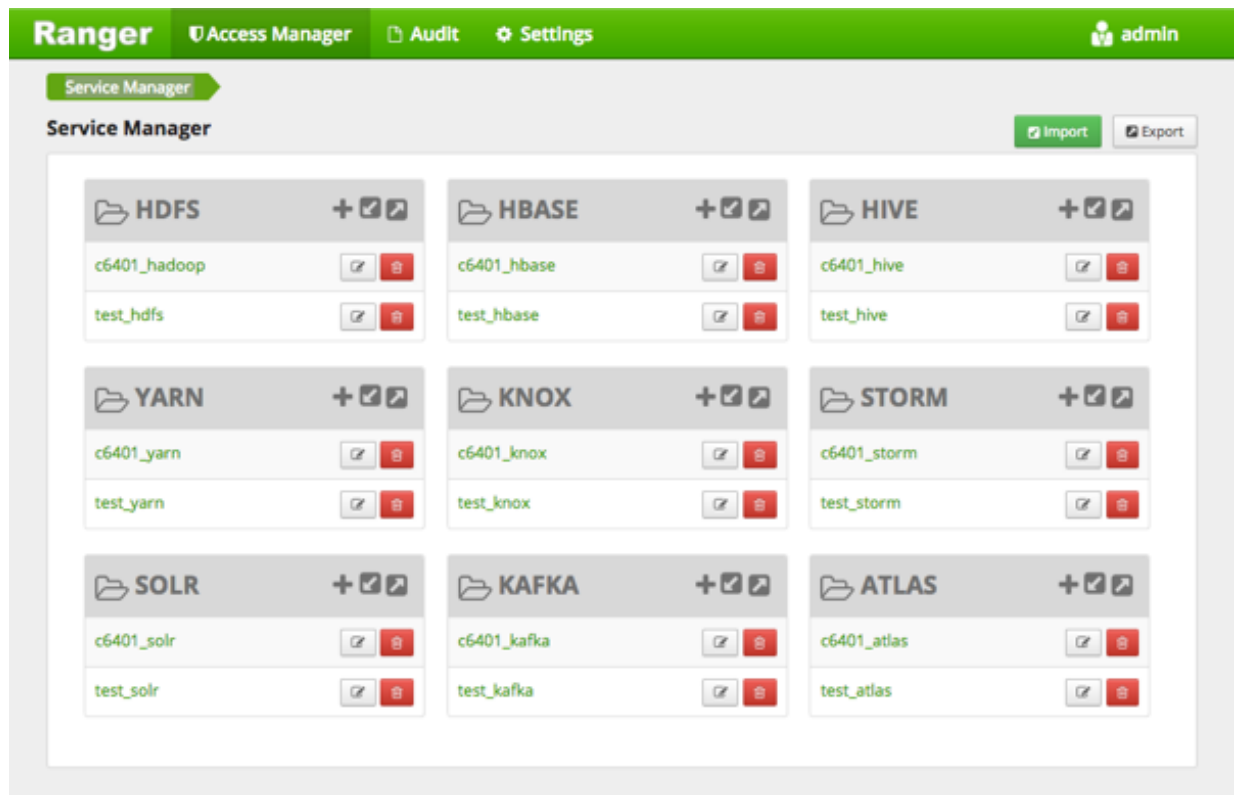
To log out of the Ranger Console, click your user name, then select Log Out.



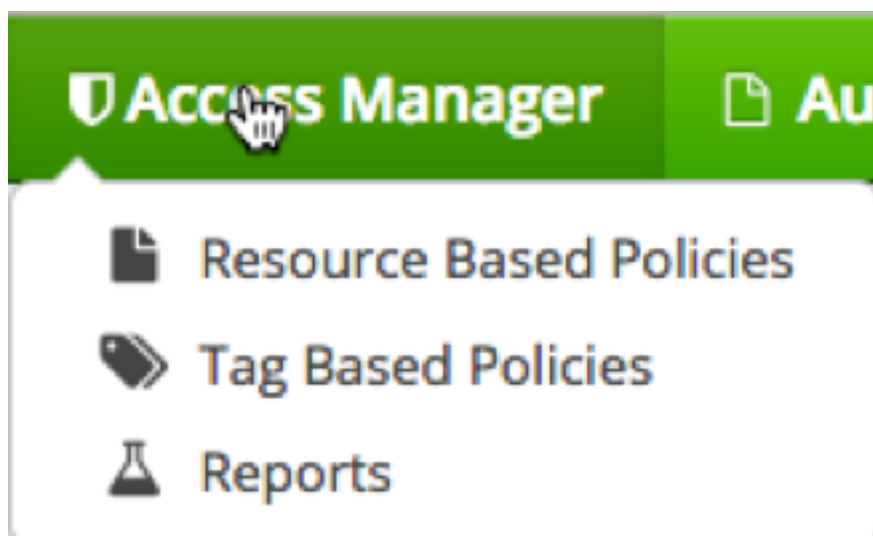
Ranger Console Navigation

Explains the basic Ranger console/GUI.

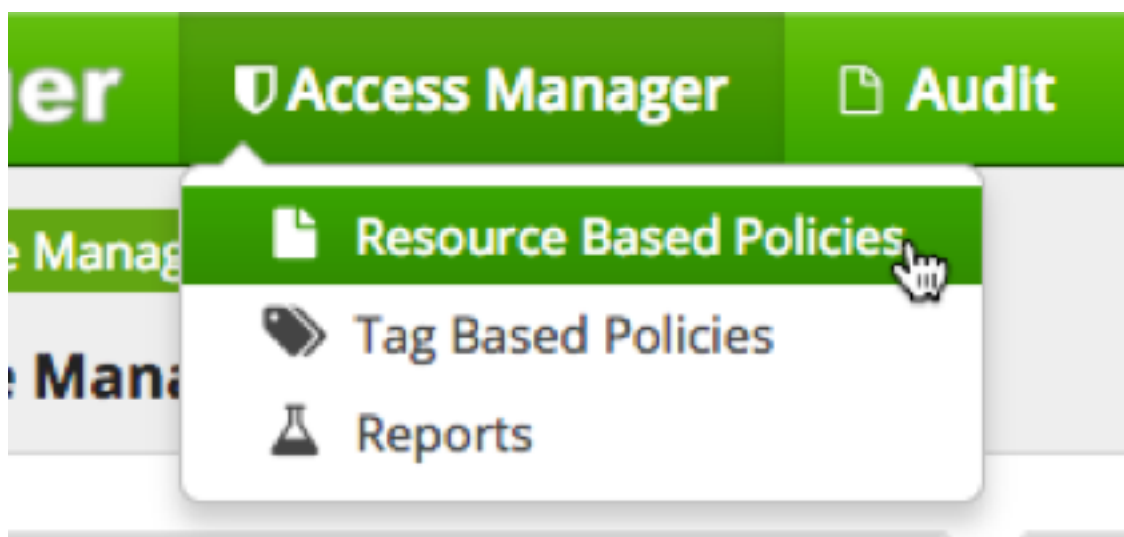
- The Service Manager for Resource Based Policies page is displayed when you log in to the Ranger Console. You can use this page to create services for Hadoop resources (HDFS, HBase, Hive, etc.) and add access policies to those resources.



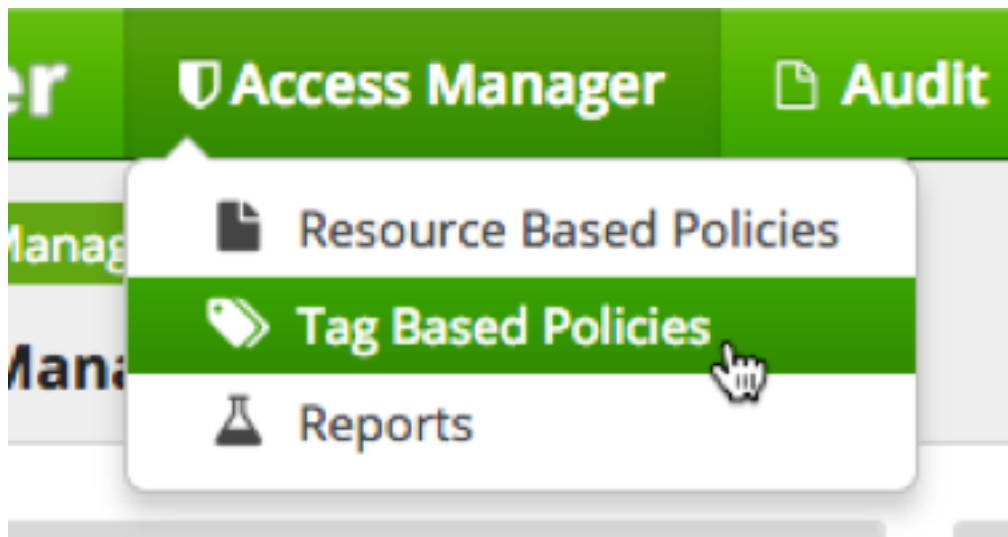
Clicking Access Manager in the top menu opens the Service Manager for Resource Based Policies page, and also displays a submenu with links to Resource Based Policies, Tag Based Policies, and Reports (this submenu is also displayed when you pass the mouse over the Access Manager link).



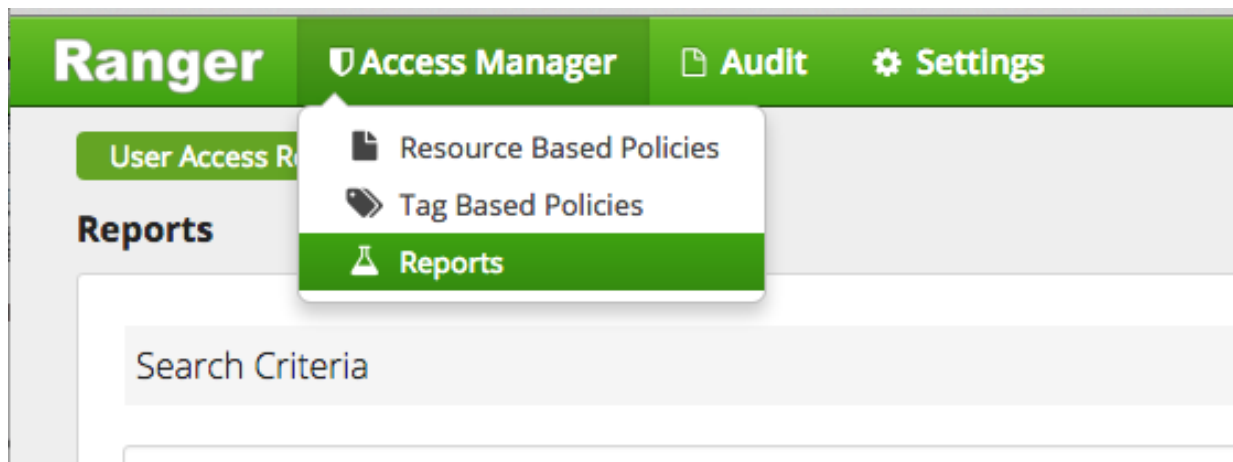
- Access Manager > Resource Based Policies -- Opens the Service Manager for Resource Based Policies page. You can use this page to create services for resources (HDFS, HBase, Hive, etc.) and add access policies to those services.



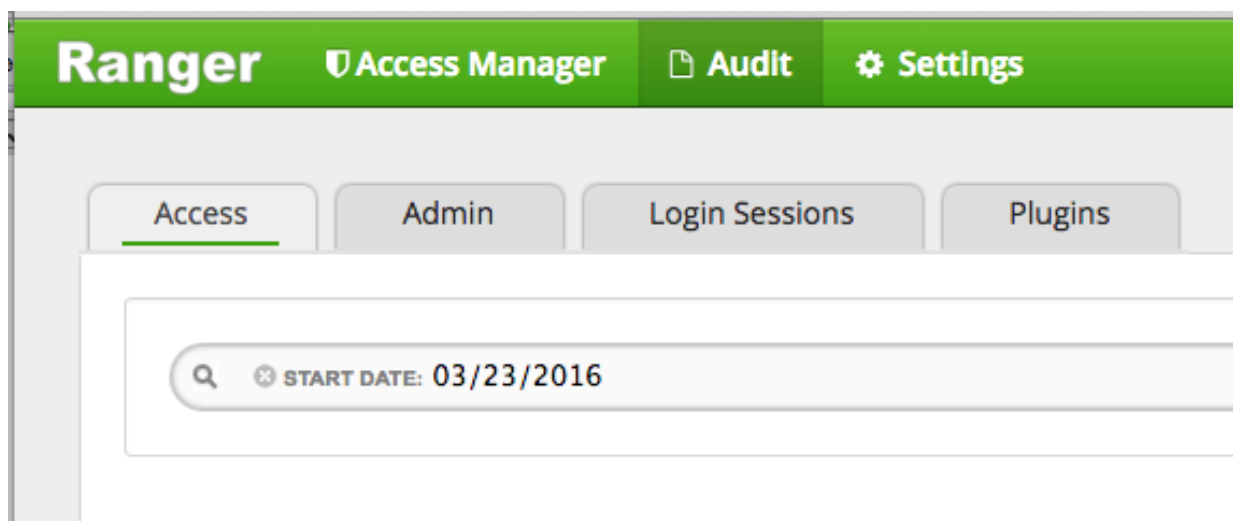
- Access Manager > Tag Based Policies -- Opens the Service Manager for Tag Based Policies page. You can use this page to create tag-based services and add access policies to those services. Using tag-based policies enables you to control access to resources across multiple Hadoop components without creating separate services and policies in each component.



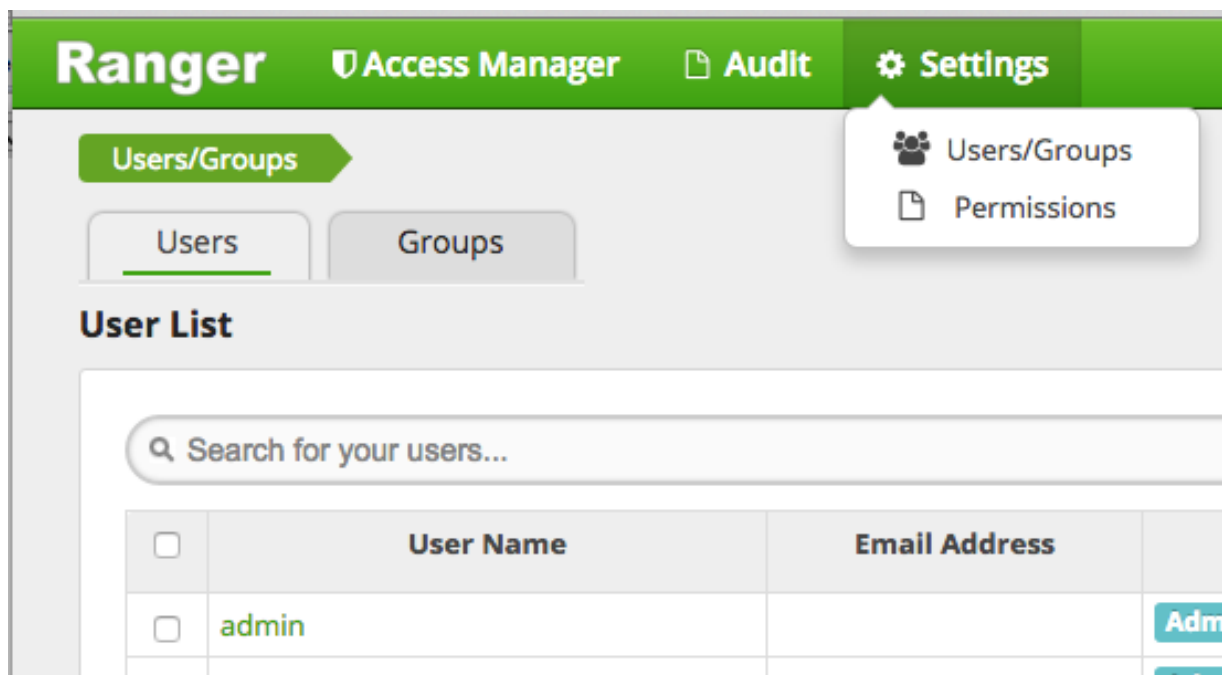
- Access Manager > Reports -- Opens the Reports page. You can use this page to generate user access reports for resource and tag-based policies based on policy name, resource, group, and user name.



- Audit -- You can use the Audit page to monitor user activity at the resource level, and also to set up conditional auditing based on users, groups, or time. The Audit page includes the Access, Admin, Login Sessions, and Plugins tabs.



- Settings -- Enables you to manage and assign policy permissions to users and groups. Clicking or passing the mouse over Settings displays a submenu with links to the Users/Groups and Permissions pages.



Resource-Based Services and Policies

Ranger enables you to configure resource-based services for Hadoop components (e.g. HBase, Kafka, Storm, etc.) and add access policies to those services.

Configuring Resource-Based Services

The Service Manager for Resource Based Policies page is displayed when you log in to the Ranger Console. You can also access this page by selecting Access Manager > Resource Based Policies. You can use this page to create services for Hadoop resources (HDFS, HBase, Hive, etc.) and add access policies to those resources.

- To add a new resource-based service, click the Add icon



()
in the applicable box on the Service Manager page. Enter the required configuration settings, then click **Add**.

- To edit a resource-based service, click the Edit icon

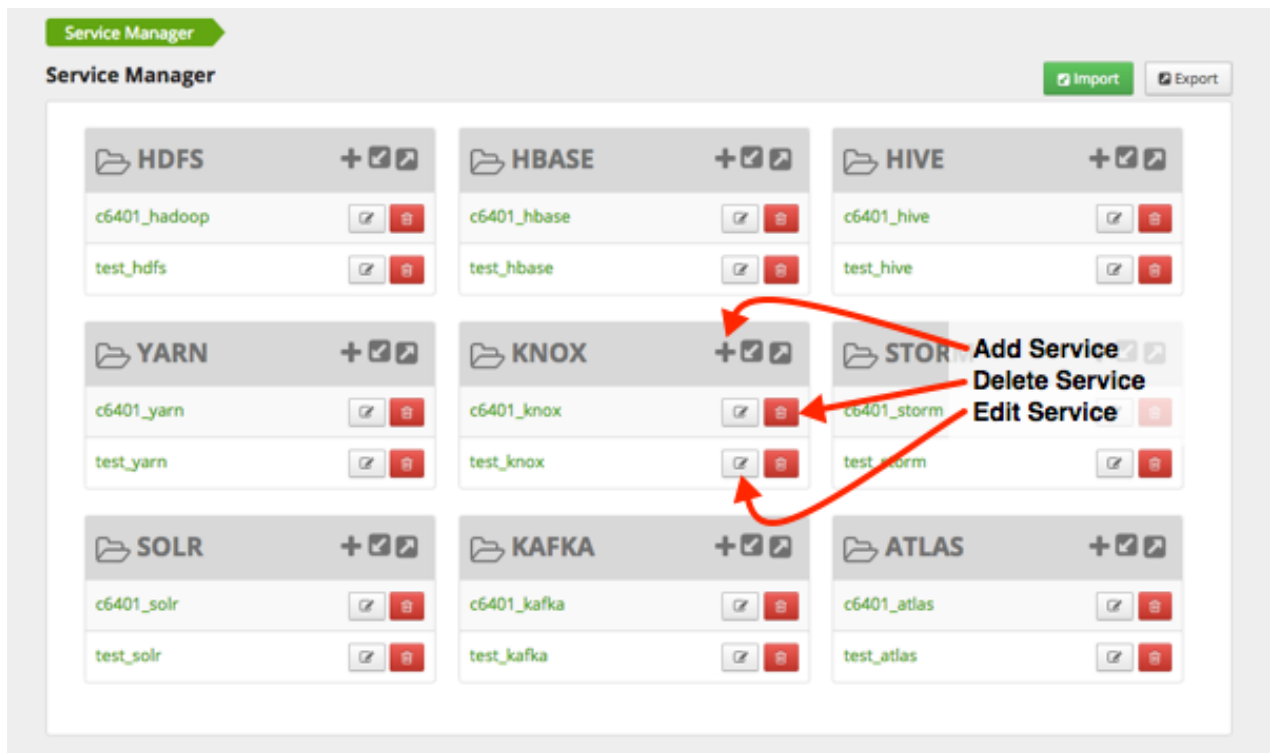


()
at the right of the service. Edit the service settings, then click **Save** to save your changes.

- To delete a resource-based service, click the Delete icon



()
at the right of the service. Deleting a service also deletes all of the policies for that service.



This section describes how to configure resource-based services for the following Hadoop components:

- HBase
- HDFS
- Hive
- Kafka
- Knox
- Solr
- Storm
- YARN
- Atlas

Configure a Resource-based Service: HBase

How to add a service to HBase.

Procedure

1. On the Service Manager page, click the Add icon



(next to HBase.)

The Create Service page appears.

2. Enter the following information on the Create Service page:

Table 1: Service Details

Field name	Description
Service Name	The name of the service; required when configuring agents.
Description	A description of the service.
Active Status	Enabled or Disabled.
Select Tag Service	Select a tag-based service to apply the service and its tag-based policies to HBase.

Table 2: Config Properties

Field name	Description
Username	The end system username that can be used for connection.
Password	The password for the username entered above.
hadoop.security.authorization	The complete connection URL, including port and database name. (Default port: 10000.) For example, on the sandbox, jdbc:hive2://sandbox:10000/.
hbase.master.kerberos.principal	The Kerberos principal for the HBase Master. (Required only if Kerberos authentication is enabled.)
hbase.security.authentication	As noted in the hadoop configuration file hbase-site.xml.
hbase.zookeeper.property.clientPort	As noted in the hadoop configuration file hbase-site.xml.
hbase.zookeeper.quorum	As noted in the hadoop configuration file hbase-site.xml.
zookeeper.znode.parent	As noted in the hadoop configuration file hbase-site.xml.
Common Name for Certificate	The name of the certificate. This field is interchangeably named Common Name For Certificate and Ranger Plugin SSL CName in Create Service pages.
Add New Configurations	Add any other new configuration(s).

3. Click **Test Connection**.

4. Click **Add**.

Configure a Resource-based Service: HDFS

How to add a service to HDFS.

Procedure

1. On the Service Manager page, click the Add icon



(next to HDFS.)

The Create Service page appears.

2. Enter the following information on the Create Service page:

Table 3: Service Details

Field name	Description
Service Name	The name of the service; required when configuring agents.
Description	A description of the service.
Active Status	Enabled or Disabled.
Select Tag Service	Select a tag-based service to apply the service and its tag-based policies to HDFS.

Table 4: Config Properties

Field name	Description
Username	The end system username that can be used for connection.
Password	The password for the username entered above.
NameNode URL	hdfs://NAMENODE_FQDN:8020 The location of the Hadoop HDFS service, as noted in the hadoop configuration file core-site.xml OR (if this is a HA environment) the path for the primary NameNode. This field was formerly named fs.defaultFS.
Authorization Enabled	Authorization involves restricting access to resources. If enabled, user need authorization credentials.
Authentication Type	The type of authorization in use, as noted in the hadoop configuration file core-site.xml; either simple or Kerberos. (Required only if authorization is enabled). This field was formerly named hadoop.security.authorization.
hadoop.security.auth_to_local	Maps the login credential to a username with Hadoop; use the value noted in the hadoop configuration file, core-site.xml.
dfs.datanode.kerberos.principal	The principal associated with the datanode where the service resides, as noted in the hadoop configuration file hdfs-site.xml. (Required only if Kerberos authentication is enabled).
dfs.namenode.kerberos.principal	The principal associated with the NameNode where the service resides, as noted in the hadoop configuration file hdfs-site.xml. (Required only if Kerberos authentication is enabled).
dfs.secondary.namenode.kerberos.principal	The principal associated with the secondary NameNode where the service resides, as noted in the hadoop configuration file hdfs-site.xml. (Required only if Kerberos authentication is enabled).
RPC Protection Type	Only authorised user can view, use, and contribute to a dataset. A list of protection values for secured SASL connections. Values: Authentication, Integrity, Privacy
Common Name For Certificate	The name of the certificate. This field is interchangeably named Common Name For Certificate and Ranger Plugin SSL CName in Create Service pages.
Add New Configurations	Add any other new configuration(s).

3. Click **Test Connection**.

4. Click **Add**.

Configure a Resource-based Service: Hive

How to add a service to Hive.

Procedure

1. On the Service Manager page, click the Add icon



(next to Hive.

The Create Service page appears.

2. Enter the following information on the Create Service page:

Table 5: Service Details

Field name	Description
Service Name	The name of the service; required when configuring agents.
Description	A description of the service.
Active Status	Enabled or Disabled.
Select Tag Service	Select a tag-based service to apply the service and its tag-based policies to Hive.

Table 6: Config Properties

Field name	Description
Username	The end system username that can be used for connection.
Password	The password for the username entered above.
jdbc.driver ClassName	The full classname of the driver used for Hive connections. Default: org.apache.hive.jdbc.HiveDriver

Field name	Description
jdbc.url	The complete connection URL, including port and database name. (Default port: 10000.) For example, on the sandbox, jdbc:hive2://sandbox:10000/.
Common Name For Certificate	The name of the certificate. This field is interchangeably named Common Name For Certificate and Ranger Plugin SSL CName in Create Service pages.
Add New Configurations	Add any other new configuration(s).

3. Click **Test Connection**.

4. Click **Add**.

What to do next

Usually, the Ranger Hive service definition uses the HiveServer2 (HS2) JDBC driver to fetch Hive database/table info for resource lookup and testing the connection. Alternatively, you can configure the service definition to use Hive metastore libraries connecting to the Hive metastore database directly. This is recommended when it is difficult to set up HiveServer2 on your cluster, such as when using HDCloud for AWS.

1. Under Ambari>Hive>Configs>Advanced, edit Hive properties:
2. Add the below properties to custom ranger-hive-plugin-properties:

```
ranger.service.config.param.enable.hive.metastore.lookup = true
```

```
ranger.service.config.param.hive.site.file.path = /etc/hive/conf/hive-site.xml
```



3. Save and restart required components.

4. To test the configuration is successful, create a new Hive service and specify the jdbc.url as "none", then run **Test**

Config Properties :

Username *

Password *

jdbc.driverClassName *

jdbc.url *

Common Name for Certificate

Add New Configurations

Name	Value	
enable.hive.metastore.lookup	<input type="text" value="true"/>	<input type="button" value="x"/>
hive.site.file.path	<input type="text" value="/etc/hive/conf/hive-site.xml"/>	<input type="button" value="x"/>
ambari.service.check.user	<input type="text" value="ambari-qa"/>	<input type="button" value="x"/>

Connection.

Configure a Resource-based Service: Kafka

How to add a service to Kafka.

Procedure

1. On the Service Manager page, click the Add icon



next to Kafka.

The Create Service page appears.

2. Enter the following information on the Create Service page:

Table 7: Service Details

Field name	Description
Service Name	The name of the service; required when configuring agents.
Description	A description of the service.
Active Status	Enabled or Disabled.
Select Tag Service	Select a tag-based service to apply the service and its tag-based policies to Kafka.

Table 8: Config Properties

Field name	Description
Username	The end system username that can be used for connection.
Password	The password for the username entered above.
ZooKeeper Connect String	Defaults to localhost:2181 (Provide FQDN of zookeeper host : 2181).

Field name	Description
Ranger Plugin SSL CName	Provide common.name.for.certificate which is registered with Ranger (in Wire Encryption environment). This field is interchangeably named Common Name For Certificate and Ranger Plugin SSL CName in Create Service pages.
Add New Configurations	Add any other new configuration(s).

3. Click **Test Connection**.
4. Click **Add**.

Configure a Resource-based Service: Knox

How to add a service to Knox.

Procedure

1. On the Service Manager page, click the Add icon



next to Knox.

The Create Service page appears.

The screenshot shows the 'Create Service' page in the Ranger interface. The page is divided into two main sections: 'Service Details' and 'Config Properties'.

Service Details:

- Service Name:
- Description:
- Active Status: Enabled Disabled
- Select Tag Service:

Config Properties:

- Username:
- Password:
- knox.url:
- Common Name for Certificate:

Add New Configurations:

Name	Value
<input type="text"/>	<input type="text"/>

Buttons: Test Connection, Add, Cancel

2. Enter the following information on the Create Service page:

Table 9: Service Details

Field name	Description
Service Name	The name of the service; required when configuring agents.
Description	A description of the service.
Active Status	Enabled or Disabled.
Select Tag Service	Select a tag-based service to apply the service and its tag-based policies to Knox.

Table 10: Config Properties

Field name	Description
Username	The end system username that can be used for connection.
Password	The password for the username entered above.
knox.url	The Gateway URL for Knox.
Common Name For Certificate	The name of the certificate. This field is interchangeably named Common Name For Certificate and Ranger Plugin SSL CName in Create Service pages.
Add New Configurations	Add any other new configuration(s).

3. Click **Test Connection**.
4. Click **Add**.

Configure a Resource-based Service: Solr

How to add a service to Solr.

Procedure

1. On the Service Manager page, click the Add icon



(next to Solr.)

The Create Service page appears.

2. Enter the following information on the Create Service page:

Table 11: Service Details

Field name	Description
Service Name	The name of the service; required when configuring agents.
Description	A description of the service.
Active Status	Enabled or Disabled.
Select Tag Service	Select a tag-based service to apply the service and its tag-based policies to Solr.

Table 12: Config Properties

Field name	Description
Username	The end system username that can be used for connection.
Password	The password for the username entered above.
Solr URL	For HDP Search's Solr Instance: http://Solr_host:8983 For Ambari Infra's Solr Instance: http://Solr_host:8886

Field name	Description
Ranger Plugin SSL CName	Provide common.name.for.certificate which is registered with Ranger (in Wire Encryption environment). This field is interchangeably named Common Name For Certificate and Ranger Plugin SSL CName in Create Service pages.
Add New Configurations	Add any other new configuration(s).

3. Click **Test Connection**.
4. Click **Add**.

Configure a Resource-based Service: Storm

How to add a service to Storm.

Procedure

1. On the Service Manager page, click the Add icon



next to Storm.

The Create Service page appears.

2. Enter the following information on the Create Service page:

Table 13: Service Details

Field name	Description
Service Name	The name of the service; required when configuring agents.
Description	A description of the service.
Active Status	Enabled or Disabled.
Select Tag Service	Select a tag-based service to apply the service and its tag-based policies to Storm.

Table 14: Config Properties

Field name	Description
Username	The end system username that can be used for connection.
Password	The password for the username entered above.
Nimbus URL	Host name of nimbus format, in the form: http://ipaddress:8080. This field was formerly named nimbus.url.
Common Name For Certificate	The name of the certificate. This field is interchangeably named Common Name For Certificate and Ranger Plugin SSL CName in Create Service pages.
Add New Configurations	Add any other new configuration(s).

3. Click **Test Connection**.
4. Click **Add**.

Configure a Resource-based Service: YARN

How to add a service to YARN.

Procedure

1. On the Service Manager page, click the Add icon



(next to YARN.)

The Create Service page appears.

2. Enter the following information on the Create Service page:

Table 15: Service Details

Field name	Description
Service Name	The name of the service; required when configuring agents.
Description	A description of the service.
Active Status	Enabled or Disabled.
Select Tag Service	Select a tag-based service to apply the service and its tag-based policies to YARN.

Table 16: Config Properties

Field name	Description
Username	The end system username that can be used for connection.
Password	The password for the username entered above.
YARN REST URL	Http or https://RESOURCEMANAGER_FQDN:8088.

Field name	Description
Authentication Type	The type of authorization in use, as noted in the hadoop configuration file core-site.xml; either simple or Kerberos. (Required only if authorization is enabled). This field was formerly named hadoop.security.authorization.
Common Name For Certificate	The name of the certificate. This field is interchangeably named Common Name For Certificate and Ranger Plugin SSL CName in Create Service pages.
Add New Configurations	Add any other new configuration(s).

3. Click **Test Connection**.
4. Click **Add**.

Configure a Resource-based Service: Atlas

How to add a service to Atlas.

Procedure

1. On the Service Manager page, click the Add icon

()
next to Storm.

The Create Service page appears.

2. Enter the following information on the Create Service page:

Table 17: Service Details

Field name	Description
Service Name	The name of the service; required when configuring agents.
Description	A description of the service.
Active Status	Enabled or Disabled.
Select Tag Service	Select a tag-based service to apply the service and its tag-based policies to Atlas.

Table 18: Config Properties

Field name	Description
Username	The end system username that can be used for connection.
Password	The password for the username entered above.
atlas.rest.address	Atlas host and port: : http://atlas_host_FQDN:21000.
Common Name For Certificate	The name of the certificate. This field is interchangeably named Common Name For Certificate and Ranger Plugin SSL CName in Create Service pages.
Add New Configurations	Add any other new configuration(s).

3. Click **Test Connection**.
4. Click **Add**.

Configuring Resource-Based Policies

To view the policies associated with a service, click the service name on the Resource Based Policies Service Manager page. The policies for that service will be displayed in a list, along with a search box.


- To add a new resource-based policy to the service, click **Add New Policy**.
- To edit a resource-based policy, click the Edit icon



()
at the right of the entry for that service. Edit the policy settings, then click Save to save your changes.

- To delete a resource-based policy, click the Delete icon



()
at the right of the entry for that service.

Policy ID	Policy Name	Policy Labels	Status	Audit Logging	Groups	Users	Action
3	all - table, column-family, column	--	Enabled	Enabled	--	hbase	[View] [Edit] [Delete]
4	Service Check User Policy for Hb...	service_check	Enabled	Enabled	--	ambari-qa	[View] [Edit] [Delete]
11	grant-1532451641294	--	Enabled	Enabled	--	atlas	[View] [Edit] [Delete]
12	grant-1532451641509	--	Enabled	Enabled	--	atlas	[View] [Edit] [Delete]

- Configure a Resource-based Service Policy:

- HBase
- HDFS
- Hive
- Kafka
- Knox
- Solr
- Storm
- YARN
- Atlas

Related Information

[Importing and Exporting Resource-Based Policies](#)

Configure a Resource-based Policy: HBase

How to add a new policy to an existing HBase service.

Procedure

1. On the Service Manager page, select an existing service under HBase.



The List of Policies page appears.

Ranger Access Manager Audit Settings admin

Service Manager > dwweekly_hbase Policies

List of Policies : dwweekly_hbase

Q Search for your policy... ⓘ Add New Policy

Policy ID	Policy Name	Policy Labels	Status	Audit Logging	Groups	Users	Action
3	all - table, column-family, column	--	Enabled	Enabled	--	hbase	
4	Service Check User Policy for Hb...	--	Enabled	Enabled	--	ambari-qa	
12	grant-1540414674220	--	Enabled	Enabled	--	atlas	
13	grant-1540414674586	--	Enabled	Enabled	--	atlas	

2. Click **Add New Policy**.

The Create Policy page appears.

Ranger Access Manager Audit Settings admin

Service Manager > HBase_service1 Policies > Create Policy

Create Policy

Policy Details :

Policy Type **Access**

Policy Name * enabled

HBase Table * include

HBase Column-family * include

HBase Column * include

Description

Audit Logging **YES**

Allow Conditions :

Select Group Select User

add/edit permissions

- Read
- Write
- Create
- Admin
- Select/Deselect All

3. Complete the Create Policy page as follows:

Table 19: Policy Details

Label	Description
Policy Name	Enter an appropriate policy name. This name cannot be duplicated across the system. This field is mandatory.
HBase Table	Select the appropriate database. Multiple databases can be selected for a particular policy. This field is mandatory.
HBase Column-family	For the selected table, specify the column families to which the policy applies.
HBase Column	For the selected table and column families, specify the columns to which the policy applies.
Description	(Optional) Describe the purpose of the policy.
Audit Logging	Specify whether this policy is audited. (De-select to disable auditing).
Policy Label	Specify a label for this policy. You can search reports and filter policies based on these labels.

Table 20: Allow Conditions

Label	Description
Select Group	Specify the group to which this policy applies. To designate the group as an Administrator for the chosen resource, specify Admin permissions. (Administrators can create child policies based on existing policies). The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify a particular user to which this policy applies (outside of an already-specified group) OR designate a particular user as Admin for this policy. (Administrators can create child policies based on existing policies).
Permissions	Add or edit permissions: Read, Write, Create, Admin, Select/ Deselect All.
Delegate Admin	When a policy is assigned to a user or a group of users those users become the delegated admin. The delegated admin can update, delete the policies. It can also create child policies based on the original policy (base policy).

4. You can use the Plus (+) symbol to add additional conditions. Conditions are evaluated in the order listed in the policy. The condition at the top of the list is applied first, then the second, then the third, and so on.

5. Click **Add**.

What to do next

Provide User Access to HBase Database Tables from the Command Line

HBase provides the means to manage user access to HBase database tables directly from the command line. The most commonly-used commands are:

- GRANT

Syntax:

```
grant '<user-or-group>', '<permissions>', '<table>
```

For example, to create a policy that grants user1 read/write permission on the table usertable, the command would be:

```
grant 'user1', 'RW', 'usertable'
```

The syntax is the same for granting CREATE and ADMIN rights.

- REVOKE

Syntax:

```
revoke '<user-or-group>', '<usertable>'
```

For example, to revoke the read/write access of user1 to the table usertable, the command would be:

```
revoke 'user1', 'usertable'
```

Note:

Unlike Hive, HBase has no specific revoke commands for each user privilege.

Related Information

[Wildcards and Variables in Resource-based Policies](#)

Configure a Resource-based Policy: HDFS

How to add a new policy to an existing HDFS service.

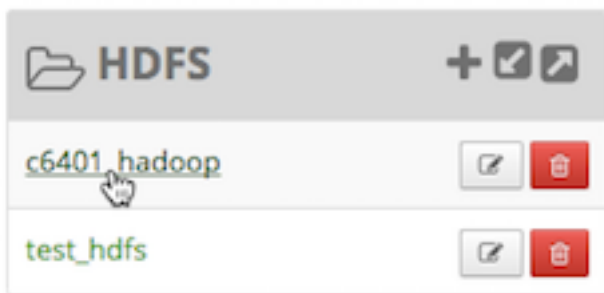
About this task

Through configuration, Apache Ranger enables both Ranger policies and HDFS permissions to be checked for a user request. When the NameNode receives a user request, the Ranger plugin checks for policies set through the Ranger Service Manager. If there are no policies, the Ranger plugin checks for permissions set in HDFS.

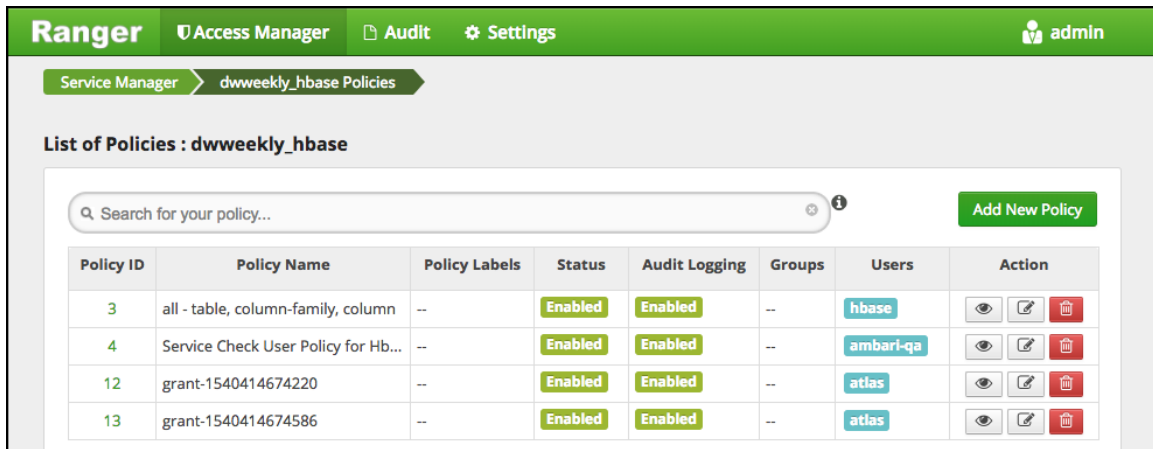
We recommend that permissions be created at the Ranger Service Manager, and to have restrictive permissions at the HDFS level.

Procedure

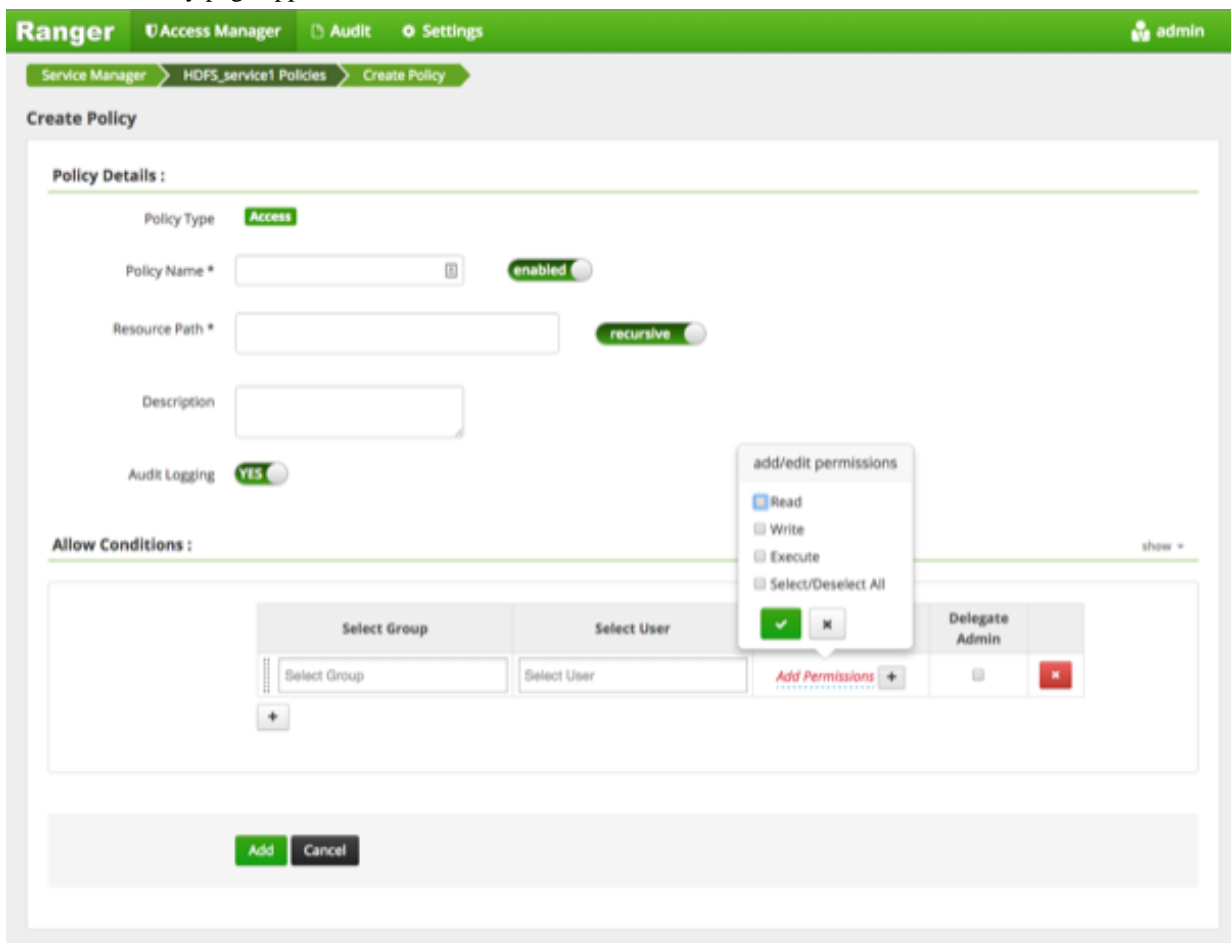
1. On the Service Manager page, select an existing service under HDFS.



The List of Policies page appears.



2. Click **Add New Policy**.
The Create Policy page appears.



3. Complete the Create Policy page as follows:

Table 21: Policy Details

Field	Description
Policy Name	Enter a unique name for this policy. The name cannot be duplicated anywhere in the system.

Field	Description
Resource Path	Define the resource path for the policy folder/file. To avoid the need to supply the full path OR to enable the policy for all subfolders or files, you can either complete this path using wildcards (for example, /home*) or specify that the policy should be recursive. (See below.)
Description	(Optional) Describe the purpose of the policy.
Audit Logging	Specify whether this policy is audited. (De-select to disable auditing).
Policy Label	Specify a label for this policy. You can search reports and filter policies based on these labels.

Table 22: Allow Conditions

Label	Description
Select Group	Specify the group to which this policy applies. To designate the group as an Administrator for the chosen resource, specify Admin permissions. (Administrators can create child policies based on existing policies). The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify a particular user to which this policy applies (outside of an already-specified group) OR designate a particular user as Admin for this policy. (Administrators can create child policies based on existing policies).
Permissions	Add or edit permissions: Read, Write, Create, Admin, Select/ Deselect All.
Delegate Admin	When a policy is assigned to a user or a group of users those users become the delegated admin. The delegated admin can update, delete the policies. It can also create child policies based on the original policy (base policy).

- You can use the Plus (+) symbol to add additional conditions. Conditions are evaluated in the order listed in the policy. The condition at the top of the list is applied first, then the second, then the third, and so on.
- Click **Add**.

Related Information

[Wildcards and Variables in Resource-based Policies](#)

Configure a Resource-based Policy: Hive

How to add a new policy to an existing Hive service.

Procedure

- On the Service Manager page, select an existing service under Hive.



The List of Policies page appears.

The screenshot shows the Ranger interface with the 'List of Policies' page for 'dwweekly_hbase'. The page includes a search bar, an 'Add New Policy' button, and a table of existing policies.

Policy ID	Policy Name	Policy Labels	Status	Audit Logging	Groups	Users	Action
3	all - table, column-family, column	--	Enabled	Enabled	--	hbase	[View] [Edit] [Delete]
4	Service Check User Policy for Hb...	--	Enabled	Enabled	--	ambari-qa	[View] [Edit] [Delete]
12	grant-1540414674220	--	Enabled	Enabled	--	atlas	[View] [Edit] [Delete]
13	grant-1540414674586	--	Enabled	Enabled	--	atlas	[View] [Edit] [Delete]

2. Click **Add New Policy**.

The Create Policy page appears.

The screenshot shows the 'Create Policy' page in Ranger. The 'Policy Details' section includes fields for Policy Type (Access), Policy Name (hive-superuser), database (hive), table (h), and Hive Column (c). There are 'enabled', 'include', and 'audit logging' toggle switches. A dropdown menu for 'add/edit permissions' is open, showing options like select, update, Create, Drop, Alter, Index, Lock, All, Read, Write, ReplAdmin, Service Admin, and Select/Deselect All. The 'Allow Conditions' section has 'Select Group' (hive-users) and 'Select User' (hive-superuser) dropdowns, along with an 'Add Permissions' button and a 'Delegate Admin' checkbox. 'Add' and 'Cancel' buttons are at the bottom.

3. Complete the Create Policy page as follows:

Table 23: Policy Details

Field	Description
Policy Name	Enter an appropriate policy name. This name cannot be duplicated across the system. This field is mandatory. The policy is enabled by default.
Database	Type in the applicable database name. The autocomplete feature displays available databases based on the entered text. Include is selected by default to allow access. Select Exclude to deny access..
Table	To continue adding a table-based policy, keep Table selected. Type in the applicable table name. The autocomplete feature displays available tables based on the entered text. Include is selected by default to allow access. Select Exclude to deny access.
UDF	To continue adding a UDF-based policy, select UDF. Type in the applicable UDF name. The autocomplete feature displays available tables based on the entered text. Include is selected by default to allow access. Select Exclude to deny access.
Column	Type in the applicable Hive column name. The autocomplete feature displays available columns based on the entered text. Include is selected by default to allow access. Select Exclude to deny access. If using the Ranger Hive plugin with HiveServer2 or HiveServer2-LLAP, where column or description permissions include all, you must set a parameter for Hive columns to display as expected: in Ambari>Hive, under ranger-hive-security.xml, enter: xasecure.hive.describetable.showcolumns.authorization.option=show-all. Failure to set this parameter will result in the error message HiveAccessControlException.
URL	Specify the cloud storage path (for example s3a://dev-admin/demo/campaigns.txt) where the end-user permission is needed to read/write the Hive data from/to a cloud storage path. Permissions: READ operation on the URL permits the user to perform HiveServer2 operations which use S3 as data source for Hive tables. WRITE operation on the URL permits the user to perform HiveServer2 operations which write data to the specified S3 location. This feature is a Technical Preview: it is not ready for production deployment.
URI	Hive INSERT OVERWRITE queries require a Ranger URI policy to allow write operations, even if the user has write privilege granted through HDFS policy. Failure to specify this field will result in the following error: Error while compiling statement: FAILED: HiveAccessControlException Permission denied: user [jdoe] does not have [WRITE] privilege on [/tmp/*] (state=42000,code=40000) Example value: /tmp/*

Field	Description
Description	(Optional) Describe the purpose of the policy. If using the Ranger Hive plugin with HiveServer2 or HiveServer2-LLAP, where column or description permissions include all, you must set a parameter for Hive columns to display as expected: in Ambari>Hive, under ranger-hive-security.xml, enter: xasecure.hive.describetable.showcolumns.authorization.option=show-all. Failure to set this parameter will result in the error message HiveAccessControlException.
Hive Service Name	hiveservice is used only in conjunction with Permissions=Service Admin. Enables a user who has Service Admin permission in Ranger to run the kill query API: kill query <queryID> . Supported value: *. (Required)
Audit Logging	Specify whether this policy is audited. (De-select to disable auditing).
Policy Label	Specify a label for this policy. You can search reports and filter policies based on these labels.

Table 24: Allow Conditions

Label	Description
Select Group	Specify a group to which this policy applies. To designate the group as an Administrator for the chosen resource, select the Delegate Admin check box. (Administrators can create child policies based on existing policies). The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify one or more users to which this policy applies. To designate the group as an Administrator for the chosen resource, select the Delegate Admin check box. (Administrators can create child policies based on existing policies).
Permissions	Add or edit permissions: Select, Update, Create, Drop, Alter, Index, Lock, All, ReplAdmin, Service Admin, Select/Deselect All. If using the Ranger Hive plugin with HiveServer2 or HiveServer2-LLAP, where column or description permissions include all, you must set a parameter for Hive columns to display as expected: in Ambari>Hive, under ranger-hive-security.xml, enter: xasecure.hive.describetable.showcolumns.authorization.option=show-all. Failure to set this parameter will result in the error message HiveAccessControlException. In order to execute repl dump, repl load, or repl status commands, you must set a parameter: in Ambari>Hive, under hive-site.xml, enter: hive.distcp.privileged.doAs=hive. Service Admin is used in conjunction with Hive Service Name and the kill query API: kill query <queryID> .
Delegate Admin	When Delegate Admin is selected, administrative privileges are assigned to the applicable users and groups. Delegated administrators can update and delete policies, and can also create child policies based on the original policy.

4. You can use the Plus (+) symbol to add additional conditions. Conditions are evaluated in the order listed in the policy. The condition at the top of the list is applied first, then the second, then the third, and so on.
5. Click **Add**.

What to do next

Provide User Access to Hive Database Tables from the Command Line

Hive provides the means to manage user access to Hive database tables directly from the command line. The most commonly-used commands are:

- GRANT

Syntax:

```
grant <permissions> on table <table> to user <user or group>;
```

For example, to create a policy that grants user1 SELECT permission on the table default-hivesmoke22074, the command would be:

```
grant select on table default.hivesmoke22074 to user user1;
```

The syntax is the same for granting UPDATE, CREATE, DROP, ALTER, INDEX, LOCK, ALL, and ADMIN rights.

- REVOKE

Syntax:

```
revoke <permissions> on table <table> from user <user or group>;
```

For example, to revoke the SELECT rights of user1 to the table default.hivesmoke22074, the command would be:

```
revoke select on table default.hivesmoke22074 from user user1;
```

The syntax is the same for revoking UPDATE, CREATE, DROP, ALTER, INDEX, LOCK, ALL, and ADMIN rights.

Related Information

[Wildcards and Variables in Resource-based Policies](#)

Configure a Resource-based Policy: Kafka

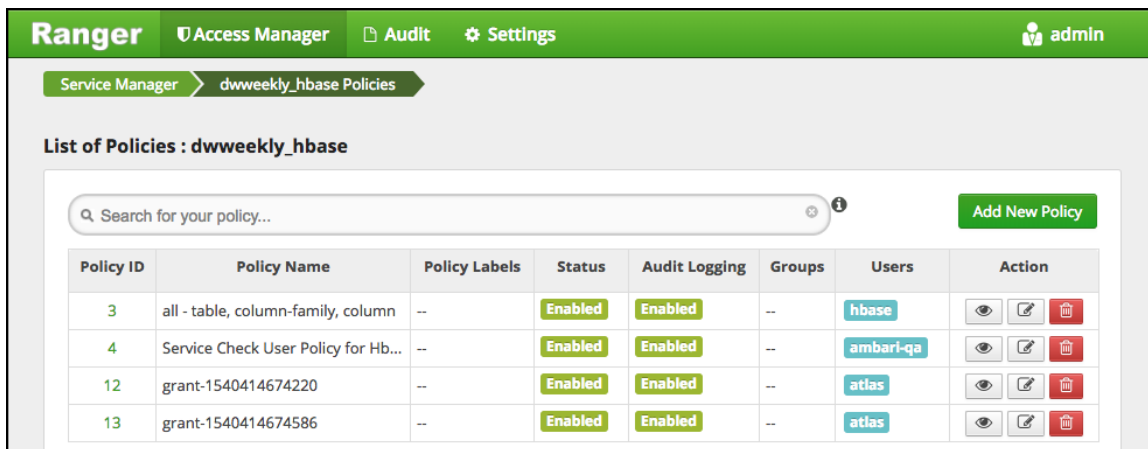
How to add a new policy to an existing Kafka service.

Procedure

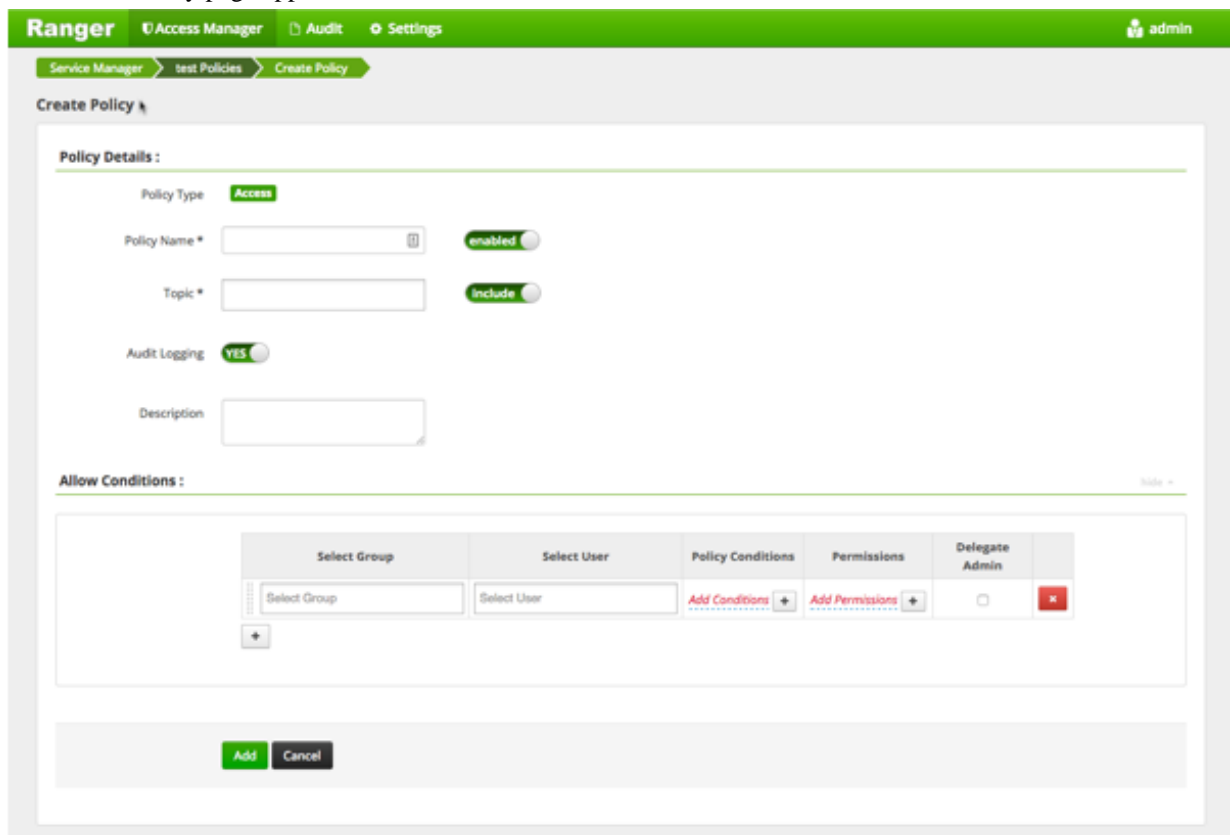
1. On the Service Manager page, select an existing service under Kafka.



The List of Policies page appears.



2. Click **Add New Policy**.
The Create Policy page appears.



3. Complete the Create Policy page as follows:

Table 25: Policy Details

Field	Description
Policy Name	Enter an appropriate policy name. This name cannot be duplicated across the system. This field is mandatory.
Topic	A topic is a category or feed name to which messages are published.
Description	(Optional) Describe the purpose of the policy.
Audit Logging	Specify whether this policy is audited. (De-select to disable auditing).

Field	Description
Policy Label	Specify a label for this policy. You can search reports and filter policies based on these labels.

Table 26: Allow Conditions

Label	Description
Select Group	Specify the group to which this policy applies. To designate the group as an Administrator for the chosen resource, specify Admin permissions. (Administrators can create child policies based on existing policies). The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify a particular user to which this policy applies (outside of an already-specified group) OR designate a particular user as Admin for this policy. (Administrators can create child policies based on existing policies).
Policy Conditions	Specify IP address range.
Permissions	Add or edit permissions: Read, Write, Create, Admin, Select/ Deselect All.
Delegate Admin	When a policy is assigned to a user or a group of users those users become the delegated admin. The delegated admin can update, delete the policies. It can also create child policies based on the original policy (base policy).

- You can use the Plus (+) symbol to add additional conditions. Conditions are evaluated in the order listed in the policy. The condition at the top of the list is applied first, then the second, then the third, and so on.
- Click **Add**.

Related Information

[Wildcards and Variables in Resource-based Policies](#)

Configure a Resource-based Policy: Knox

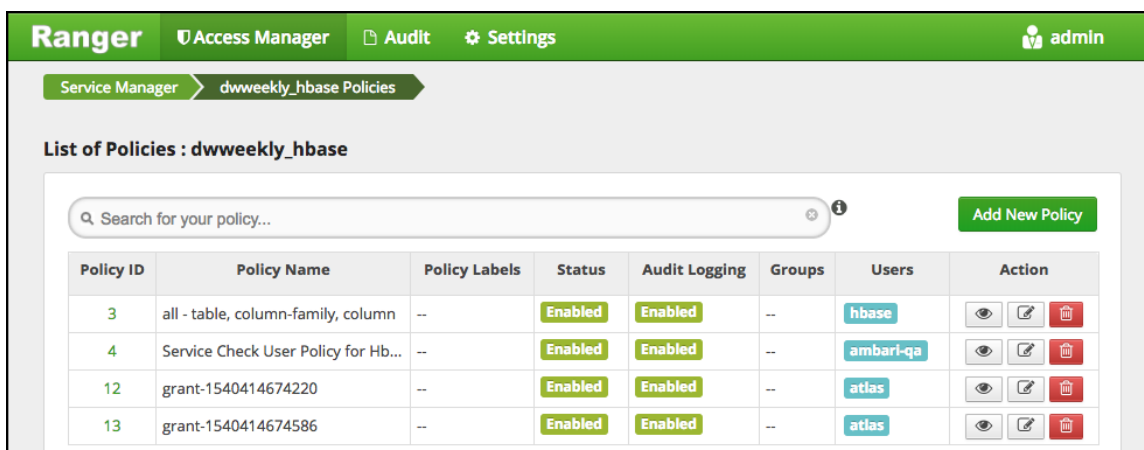
How to add a new policy to an existing Knox service.

Procedure

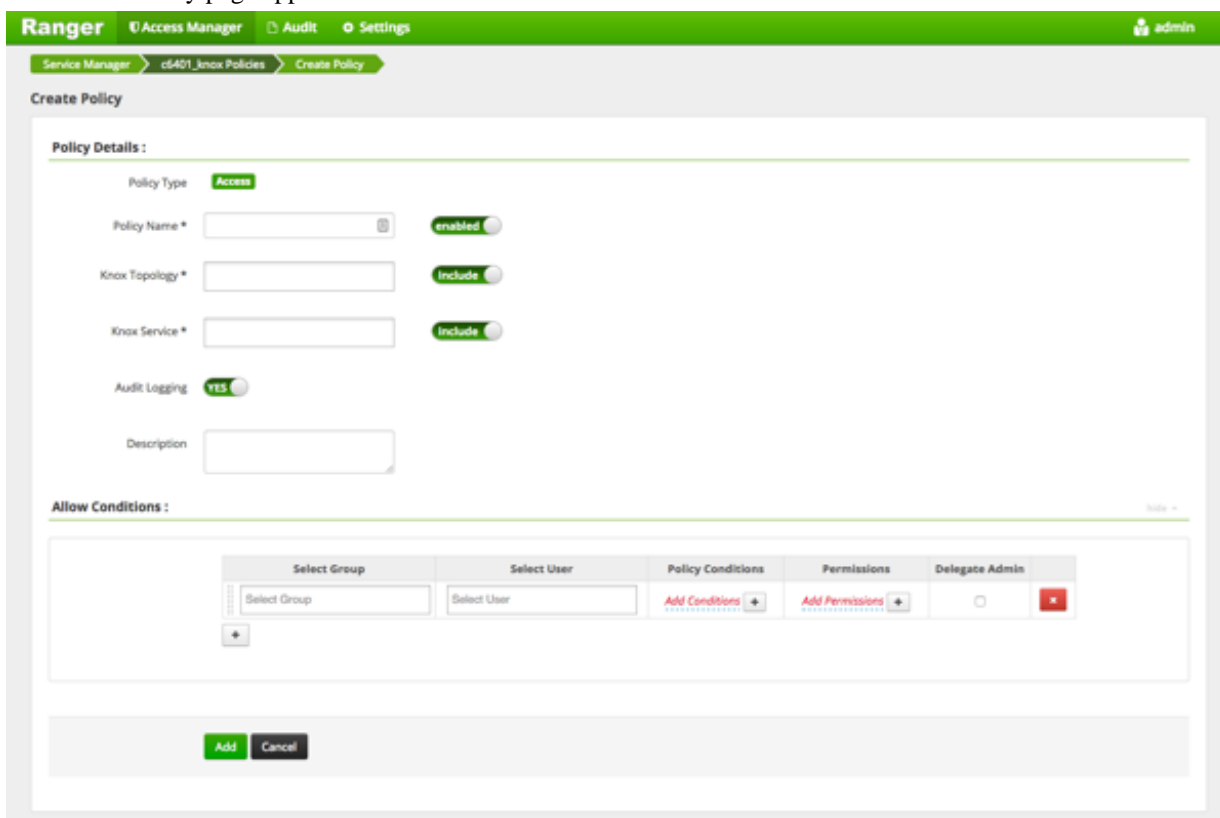
- On the Service Manager page, select an existing service under Knox.



The List of Policies page appears.



2. Click **Add New Policy**.
The Create Policy page appears.



3. Complete the Create Policy page as follows:

Table 27: Policy Details

Field	Description
Policy Name	Enter an appropriate policy name. This name cannot be duplicated across the system. This field is mandatory.
Knox Topology	Enter an appropriate Topology Name.
Knox Service	Enter an appropriate Service Name.
Description	(Optional) Describe the purpose of the policy.
Audit Logging	Specify whether this policy is audited. (De-select to disable auditing).

Field	Description
Policy Label	Specify a label for this policy. You can search reports and filter policies based on these labels.

Table 28: Allow Conditions

Label	Description
Select Group	Specify the group to which this policy applies. To designate the group as an Administrator for the chosen resource, specify Admin permissions. (Administrators can create child policies based on existing policies). The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify a particular user to which this policy applies (outside of an already-specified group) OR designate a particular user as Admin for this policy. (Administrators can create child policies based on existing policies).
Policy Conditions	Specify IP address range,
Permissions	Add or edit permissions: Read, Write, Create, Admin, Select/ Deselect All.
Delegate Admin	When a policy is assigned to a user or a group of users those users become the delegated admin. The delegated admin can update, delete the policies. It can also create child policies based on the original policy (base policy).

Since Knox does not provide a command line methodology for assigning privileges or roles to users, the User and Group Permissions portion of the Knox Create Policy form is especially important.

- You can use the Plus (+) symbol to add additional conditions. Conditions are evaluated in the order listed in the policy. The condition at the top of the list is applied first, then the second, then the third, and so on.
- Click **Add**.

Related Information

[Wildcards and Variables in Resource-based Policies](#)

Configure a Resource-based Policy: Solr

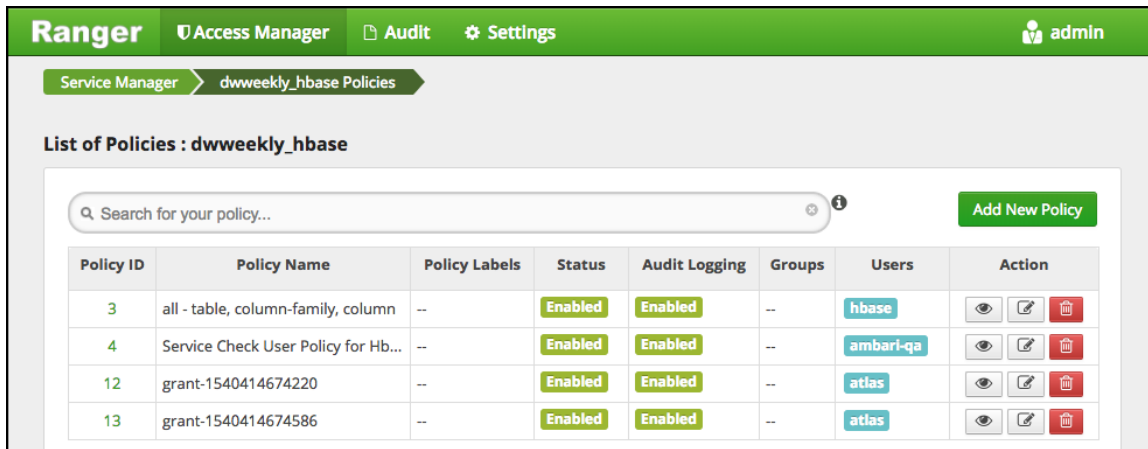
How to add a new policy to an existing Solr service.

Procedure

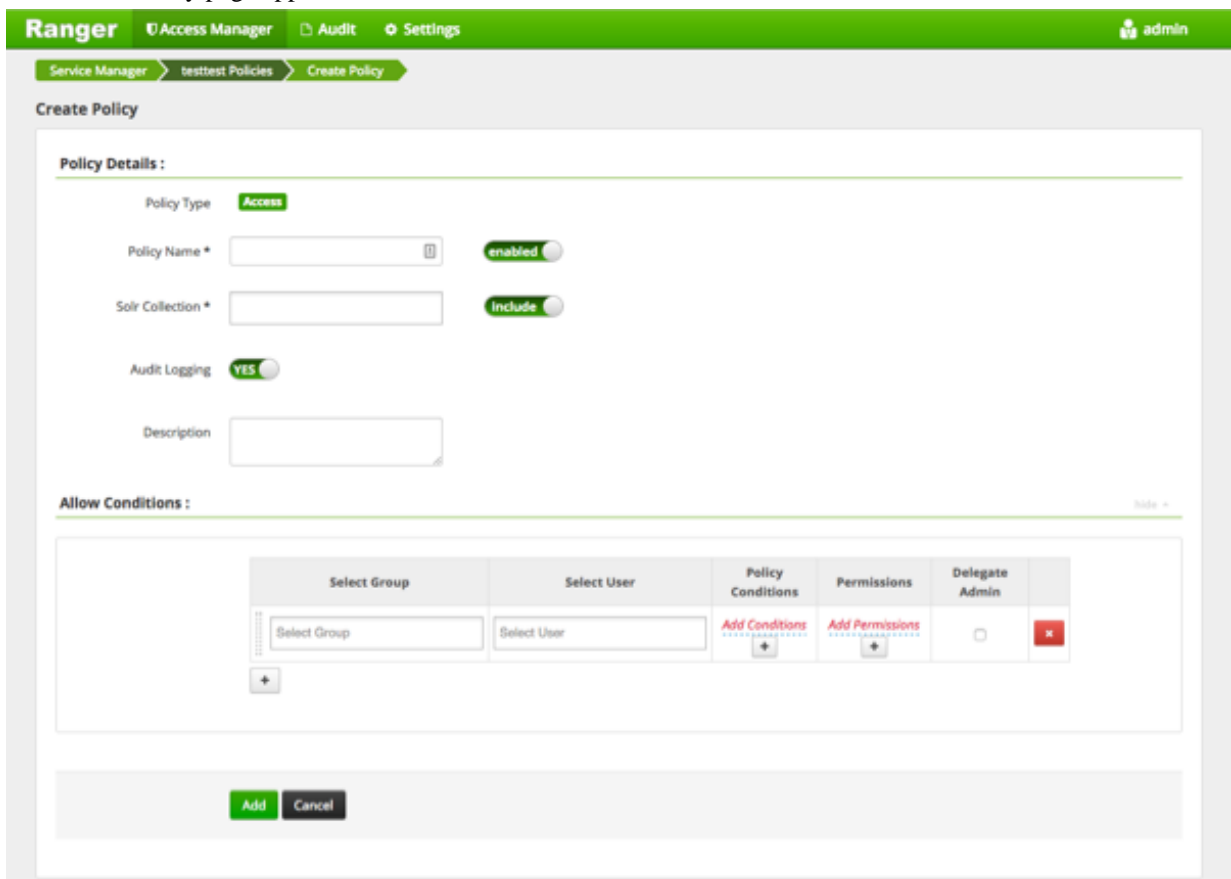
- On the Service Manager page, select an existing service under Solr.



The List of Policies page appears.



- Click **Add New Policy**.
The Create Policy page appears.



- Complete the Create Policy page as follows:

Table 29: Policy Details

Field	Description
Policy Name	Enter an appropriate policy name. This name cannot be duplicated across the system. This field is mandatory.
Solr Collection	For HDP Search's Solr Instance: http:host_ip:8983/solr For Ambari Infra's Solr Instance: http:host_ip:8886/solr
Description	(Optional) Describe the purpose of the policy.

Field	Description
Audit Logging	Specify whether this policy is audited. (De-select to disable auditing).
Policy Label	Specify a label for this policy. You can search reports and filter policies based on these labels.

Table 30: Allow Conditions

Label	Description
Select Group	Specify the group to which this policy applies. To designate the group as an Administrator for the chosen resource, specify Admin permissions. (Administrators can create child policies based on existing policies). The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify a particular user to which this policy applies (outside of an already-specified group) OR designate a particular user as Admin for this policy. (Administrators can create child policies based on existing policies).
Policy Conditions	Specify IP address range,
Permissions	Add or edit permissions: Read, Write, Create, Admin, Select/ Deselect All.
Delegate Admin	When a policy is assigned to a user or a group of users those users become the delegated admin. The delegated admin can update, delete the policies. It can also create child policies based on the original policy (base policy).

- You can use the Plus (+) symbol to add additional conditions. Conditions are evaluated in the order listed in the policy. The condition at the top of the list is applied first, then the second, then the third, and so on.
- Click **Add**.

Related Information

[Wildcards and Variables in Resource-based Policies](#)

Configure a Resource-based Policy: Storm

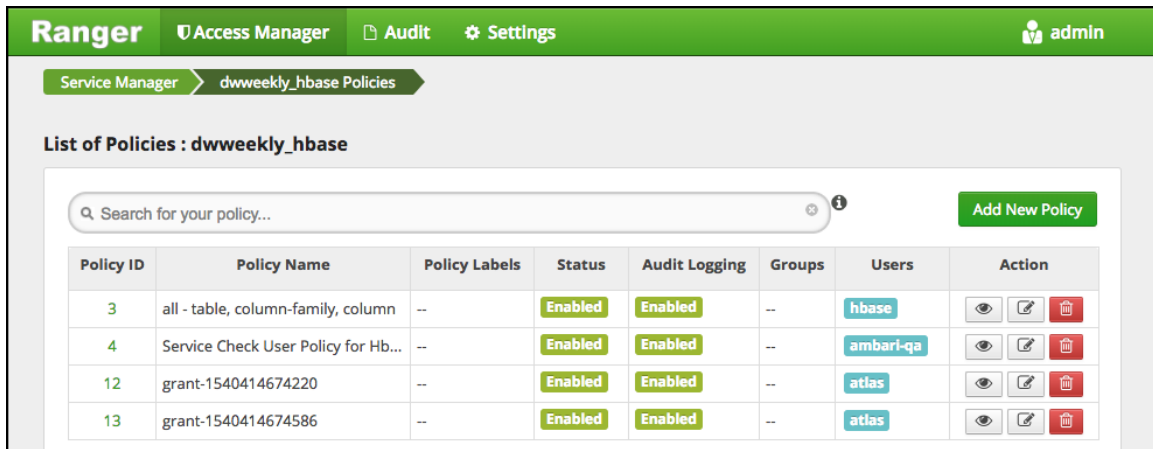
How to add a new policy to an existing Storm service.

Procedure

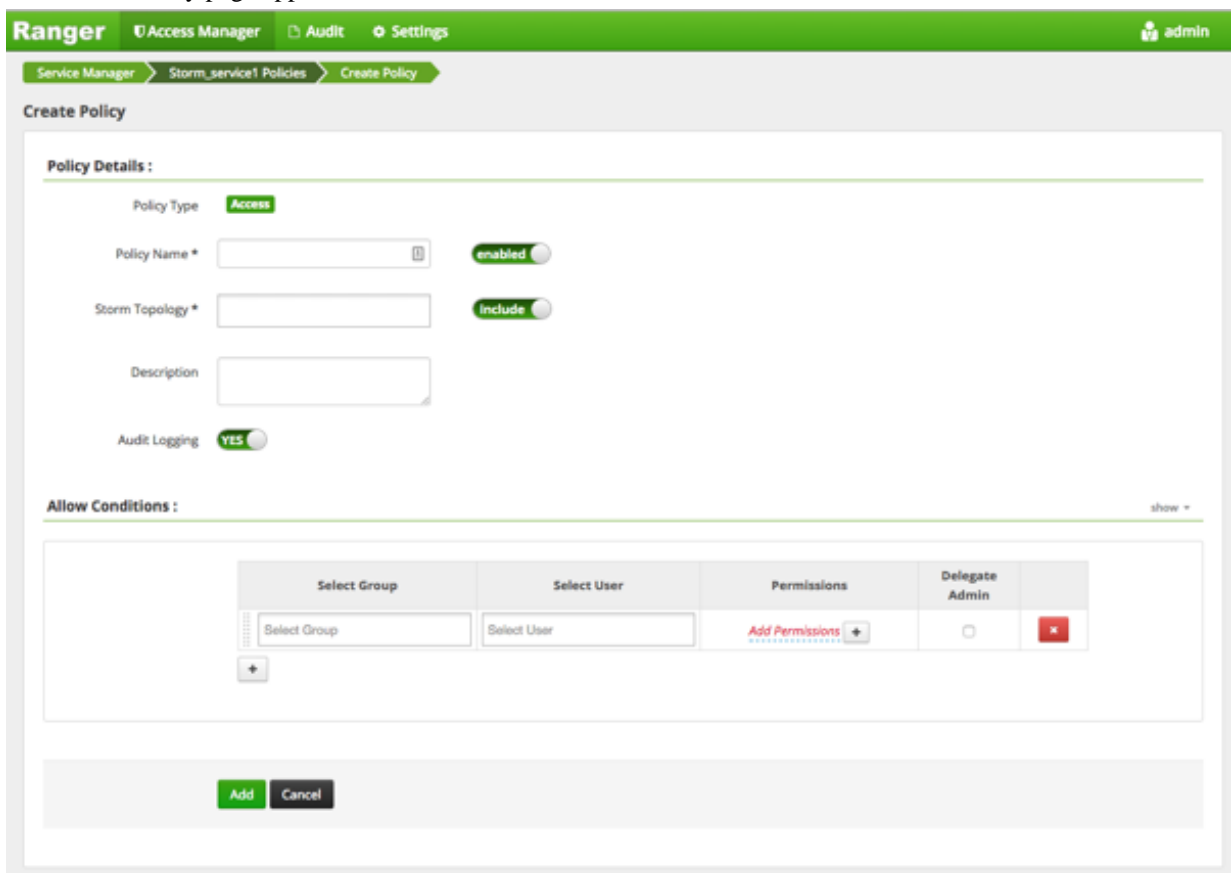
- On the Service Manager page, select an existing service under Storm.



The List of Policies page appears.



2. Click **Add New Policy**.
The Create Policy page appears.



3. Complete the Create Policy page as follows:

Table 31: Policy Details

Label	Description
Policy Name	Enter an appropriate policy name. This name is cannot be duplicated across the system. This field is mandatory.
Storm Topology	Enter an appropriate Topology Name.
Description	(Optional) Describe the purpose of the policy.

Label	Description
Audit Logging	Specify whether this policy is audited. (De-select to disable auditing).
Policy Label	Specify a label for this policy. You can search reports and filter policies based on these labels.

Table 32: Allow Conditions

Label	Description
Select Group	Specify the group to which this policy applies. To designate the group as an Administrator for the chosen resource, specify Admin permissions. (Administrators can create child policies based on existing policies). The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify a particular user to which this policy applies (outside of an already-specified group) OR designate a particular user as Admin for this policy. (Administrators can create child policies based on existing policies).
Storm User and Group Permissions*	Add or edit permissions: Read, Write, Create, Admin, Select/ Deselect All.
Delegate Admin	When a policy is assigned to a user or a group of users those users become the delegated admin. The delegated admin can update, delete the policies. It can also create child policies based on the original policy (base policy).

Since Storm does not provide a command line methodology for assigning privileges or roles to users, the User and Group Permissions portion of the Storm Create Policy form is especially important.

Table 33: * Storm User and Group Permissions

Actions	Description
File upload	Allows a user to upload files.
Get Nimbus Conf	Allows a user to access Nimbus configurations.
Get Cluster Info	Allows a user to get cluster information.
File Download	Allows a user to download files.
Kill Topology	Allows a user to kill the topology.
Rebalance	Allows a user to rebalance topologies.
Activate	Allows a user to activate a topology.
Deactivate	Allows a user to deactivate a topology.
Get Topology Conf	Allows a user to access a topology configuration.
Get Topology	Allows a user to access a topology.
Get User Topology	Allows a user to access a user topology.
Get Topology Info	Allows a user to access topology information.
Upload New Credential	Allows a user to upload a new credential.
Admin	Provides a user with delegated admin access.

- You can use the Plus (+) symbol to add additional conditions. Conditions are evaluated in the order listed in the policy. The condition at the top of the list is applied first, then the second, then the third, and so on.
- Click **Add**.

Related Information[Wildcards and Variables in Resource-based Policies](#)**Configure a Resource-based Policy: YARN**

How to add a new policy to an existing YARN service.

Procedure

1. On the Service Manager page, select an existing service under YARN.



The List of Policies page appears.

Policy ID	Policy Name	Policy Labels	Status	Audit Logging	Groups	Users	Action
3	all - table, column-family, column	--	Enabled	Enabled	--	hbase	
4	Service Check User Policy for Hb...	--	Enabled	Enabled	--	ambari-qa	
12	grant-1540414674220	--	Enabled	Enabled	--	atlas	
13	grant-1540414674586	--	Enabled	Enabled	--	atlas	

2. Click **Add New Policy**.
The Create Policy page appears.

3. Complete the Create Policy page as follows:

Table 34: Policy Details

Field	Description
Policy Name	Enter an appropriate policy name. This name cannot be duplicated across the system. This field is mandatory.
Queue	The fundamental unit of scheduling in yarn.
Recursive	You can indicate whether all files or folders within the existing folder comes under the policy. Can be used instead of wildcard characters.
Description	(Optional) Describe the purpose of the policy.
Audit Logging	Specify whether this policy is audited. (De-select to disable auditing).
Policy Label	Specify a label for this policy. You can search reports and filter policies based on these labels.

Table 35: Allow Conditions

Label	Description
Select Group	Specify the group to which this policy applies. To designate the group as an Administrator for the chosen resource, specify Admin permissions. (Administrators can create child policies based on existing policies). The public group contains all users, so granting access to the public group grants access to all users.

Label	Description
Select User	Specify a particular user to which this policy applies (outside of an already-specified group) OR designate a particular user as Admin for this policy. (Administrators can create child policies based on existing policies).
Permissions	Add or edit permissions: Read, Write, Create, Admin, Select/ Deselect All.
Delegate Admin	When a policy is assigned to a user or a group of users those users become the delegated admin. The delegated admin can update, delete the policies. It can also create child policies based on the original policy (base policy).

4. Click **Add**.

Related Information

[Wildcards and Variables in Resource-based Policies](#)

Configure a Resource-based Policy: Atlas

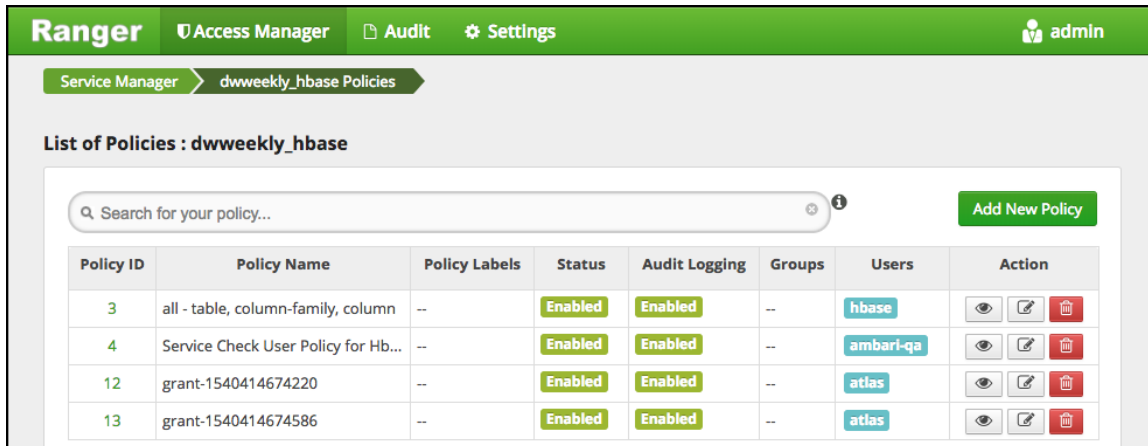
How to add a new policy to an existing Atlas service.

Procedure

1. On the Service Manager page, select an existing service under Atlas.

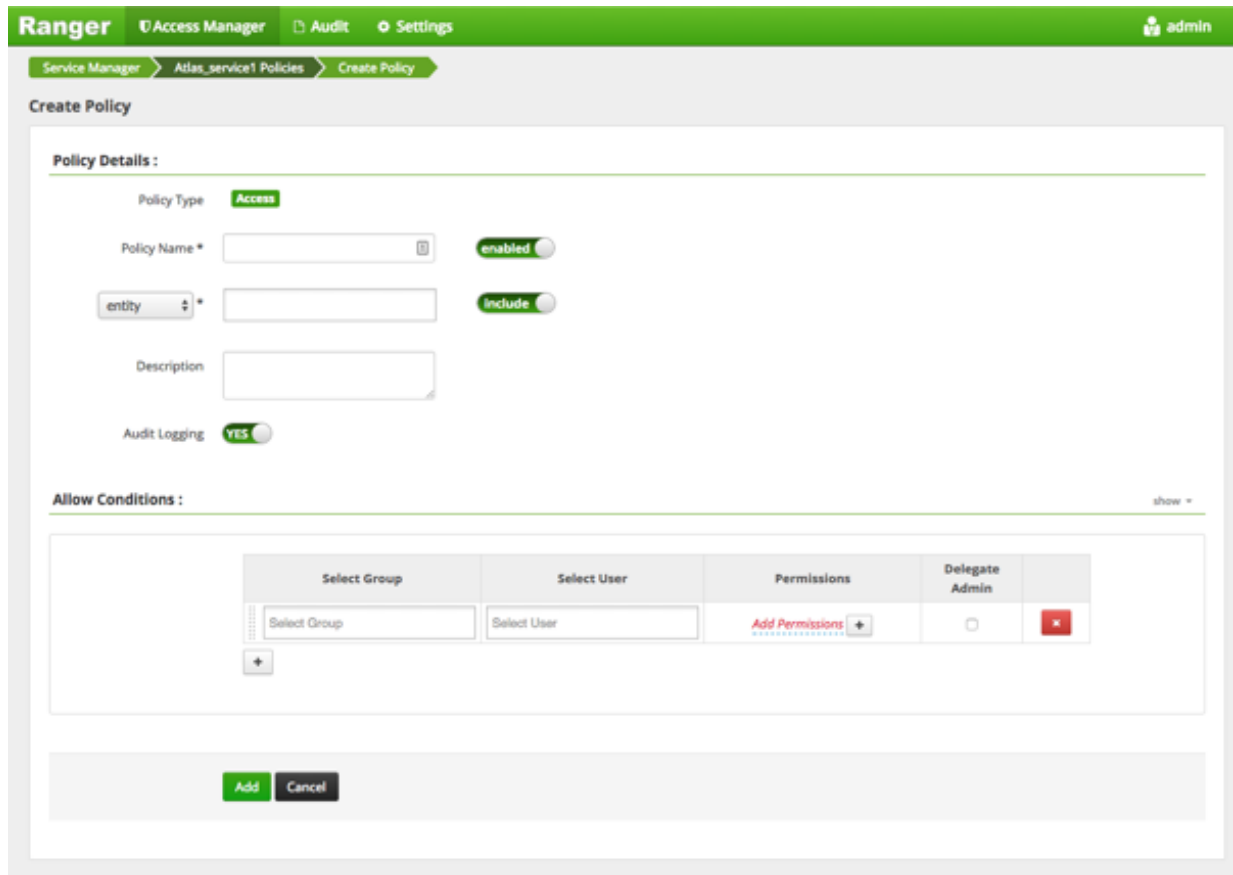


The List of Policies page appears.



2. Click **Add New Policy**.

The Create Policy page appears.



3. Complete the Create Policy page as follows:

Table 36: Policy Details

Field	Description
Policy Name	Enter an appropriate policy name. This name cannot be duplicated across the system. This field is mandatory.
entity	Select entity, type, operation, taxonomy, or term.
Description	(Optional) Describe the purpose of the policy.
Audit Logging	Specify whether this policy is audited. (De-select to disable auditing).
Policy Label	Specify a label for this policy. You can search reports and filter policies based on these labels.

Table 37: Allow Conditions

Label	Description
Select Group	Specify the group to which this policy applies. To designate the group as an Administrator for the chosen resource, specify Admin permissions. (Administrators can create child policies based on existing policies). The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify a particular user to which this policy applies (outside of an already-specified group) OR designate a particular user as Admin for this policy. (Administrators can create child policies based on existing policies).

Label	Description
Permissions	Add or edit permissions: Read, Write, Create, Admin, Select/Deselect All.
Delegate Admin	When a policy is assigned to a user or a group of users those users become the delegated admin. The delegated admin can update, delete the policies. It can also create child policies based on the original policy (base policy).

- You can use the Plus (+) symbol to add additional conditions. Conditions are evaluated in the order listed in the policy. The condition at the top of the list is applied first, then the second, then the third, and so on.
- Click **Add**.

Related Information

[Wildcards and Variables in Resource-based Policies](#)

Wildcards and Variables in Resource-based Policies

Reference for wildcards and variables in resource-based policies.

Ranger Authorization Resource Policy Wildcard Characters

Wildcard characters can be included in the resource path, the database name, the table name, or the column name:

- * indicates zero or more occurrences of characters
- ? indicates a single character

Ranger Authorization Resource Policy {USER} Variable

The variable {USER} can be used to autofill the accessing user, for example:

In **Select User**, choose {USER}.

In **Resource Path**, enter data_{USER}.

Ranger Authorization Resource Policy {USER} Variable Recommended Practices and Customizability

Ranger requires that string '{USER}' is used to represent accessing user as the user in the policy-item in a Ranger policy. However, Ranger provides flexible way of customizing the string that is used as shorthand to represent the accessing user's name in the policy resource specification. By default, Ranger policy resource specification expects characters '{' and '}' as delimiters for string 'USER', however, ranger supports customizable way of specifying delimiter characters, escaping those delimiters, and the string 'USER' itself by prefixing it with another, user-specified string on a per resource-level basis in the service definition of each component supported by Ranger.

For example, if for a certain HDFS installation, if the path names may contain '{' or '}' as valid characters, but not '%' character, then the service-definition for HDFS can be specified as:

```
"resources": [
{
  "itemId": 1,
  "name": "path",
  "type": "path",
  "level": 10,
  "parent": "",
  "mandatory": true,
  "lookupSupported": true,
  "recursiveSupported": true,
  "excludesSupported": false,
  "matcher":
"org.apache.ranger.plugin.resourcematcher.RangerPathResourceMatcher",
  "matcherOptions": {"wildcard": true, "ignoreCase": false},
  "replaceTokens":true, "tokenDelimiterStart": "%", "tokenDelimiterEnd": "%",
  "tokenDelimiterPrefix": "rangerToken:" }
  "validationRegEx": "",
  "validationMessage": "" ,
```

```
    "uiHint": "",
    "label": "Resource Path",
    "description": "HDFS file or directory
path"
}
]
```

Corresponding ranger policy for the use case for HDFS will be written as follow:

```
resource: path=/home/%rangerToken:USER%
user: {USER}
permissions: all, delegateAdmin=true
```

The following customizable matcherOptions are available for this feature:

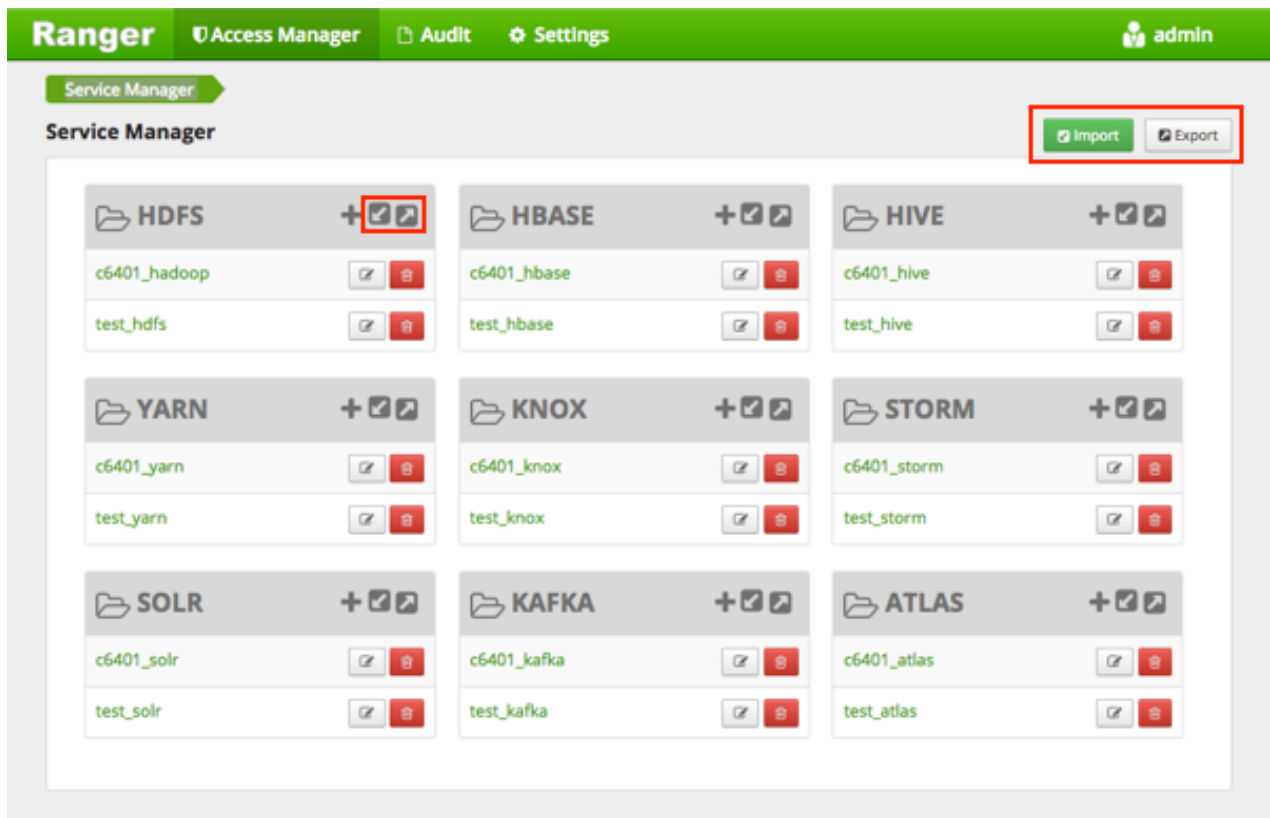
- `replaceTokens`: true if short-hand for user in resource-spec needs to be replaced at run-time with current-user's name; false if the resource-spec needs to be interpreted as it is. Default value: true.
- `tokenDelimiterStart`: Identifies start character of short-hand for current-user in resource specification. Default value: {.
- `tokenDelimiterEnd`: Identifies end character of short-hand for current-user in resource specification. Default value: }.
- `tokenDelimiterEscape`: Identifies escape character for escaping `tokenDelimiterStart` or `tokenDelimiterEnd` values in resource specification. Default value: \.
- `tokenDelimiterPrefix`: Identifies special prefix which together with string 'USER' makes up short-hand for current-user's name in the resource specification. Default value: .

Importing and Exporting Resource-Based Policies

You can export and import policies from the Ranger Admin UI for cluster resiliency (backups), during recovery operations, or when moving policies from test clusters to production clusters. You can export/import a specific subset of policies (such as those that pertain to specific resources or user/groups) or clone the entire repository (or multiple repositories) via Ranger Admin UI.

Interfaces

You can import and export policies from the Access Manager page:



You can also export policies from the Reports page:

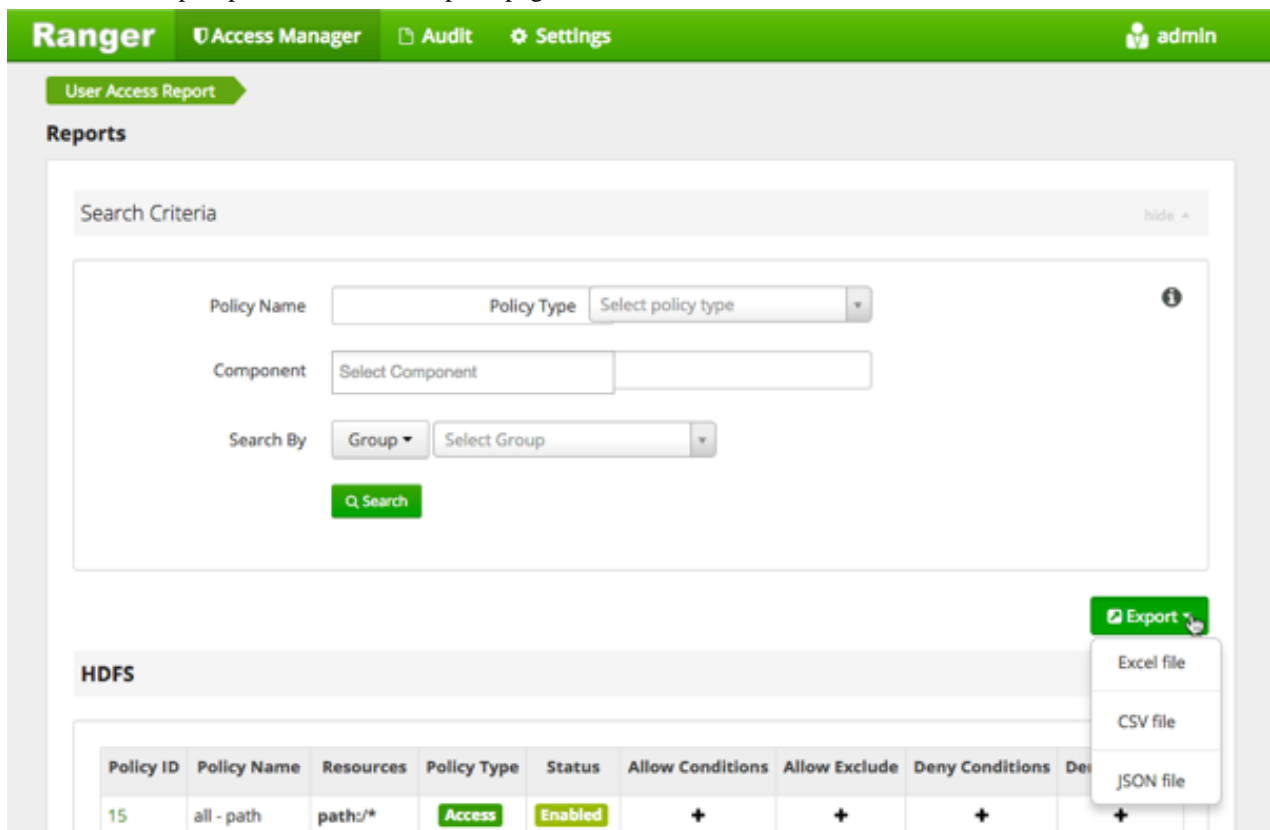


Table 38: Export Policy Options

	Access Manager Page	Reports Page
Formats	JSON	JSON Excel CSV
Filtering Supported	No	Yes
Specific Service Export	Yes	Via filtering

Filtering

When exporting from the Reports page, you can apply filters before saving the file.

Export Formats

You can export policies in the following formats:

- Excel
- JSON
- CSV

Note: CSV format is not supported for importing policies.

When you export policies from the Access Manager page, the policies are automatically downloaded in JSON format. If you wish to export in Excel or CSV format, export the policies from the Reports page dropdown menu.

Required User Roles

The Ranger admin user can import and export only Resource & Tag based policies. The credentials for this user are set in Ranger Configs > Advanced ranger-env in the fields labeled admin_username (default: admin/admin).

The Ranger KMS keyadmin user can import and export only KMS policies. The default credentials for this user are keyadmin/keyadmin.

Limitations

To successfully import policies, use the following database versions:

- MariaDB: 10.1.16+
- MySQL: 5.6.x+
- Oracle: 11gR2+
- PostgreSQL: 8.4+
- MS SQL: 2008 R2+

Partial import is not supported.

Related Information

[Configuring Resource-Based Policies](#)

[Importing and Exporting Tag-Based Policies](#)

Import Resource-Based Policies for a Specific Service

How to import the policies for a specific service (HBase, YARN, etc).

Procedure

1. From the Access Manager page, click the Import icon beside the service:



The Import Policy page opens.

Import Policy ×

Select File :

Select file

No file chosen

Override Policy :

Specify Service Mapping :

Source Destination

Enter service name To Select service name ×

2. Select the file to import.
You can only import policies in JSON format.
3. (Optional) Configure the import operation:
 - a) The Override Policy option deletes all policies of the destination repositories.
 - b) Service Mapping maps the downloaded file repository, i.e. source repository to destination repository.

Import Policy ✕

Select File :

Select file

Override Policy :

Ranger_Policies_20170209_192709.json ✕

Specify Service Mapping :

Source	To	Destination	
hbasedev1	To	hbase1 ✕ ▾	✕
<div style="border: 1px solid #ccc; padding: 5px; width: fit-content;">+</div>			

Cancel

Import

4. Click **Import.**

A confirmation message appears: “Success: File import successfully.”

Related Information

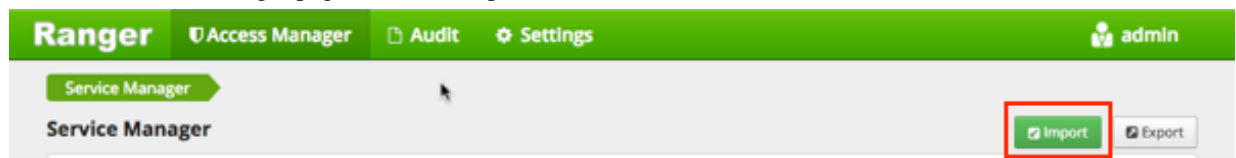
[Import Resource-Based Policies for All Services](#)

Import Resource-Based Policies for All Services

How to import the policies for all service.

Procedure

1. From the Access Manager page, click the Import button:



The Import Policy page opens.

Import Policy ✕

Service Type :

✕ hdfs
✕ hbase
✕ hive
✕ yarn
✕ Knox
✕ storm
✕ solr
✕ kafka

✕ atlas

Select File :

Select file

Override Policy :

No file chosen

Specify Service Mapping :

Source		Destination
Enter service name	To	Select service name ▼ ✕
<div style="border: 1px solid #ccc; padding: 5px; display: flex; align-items: center;"> + </div>		

Cancel
Import

2. Select the file to import.
You can only import policies in JSON format.
3. (Optional) Configure the import operation:
 - a) Service Types enables you to remove specific services from the import.
 - b) The Override Policy option deletes all policies of the destination repositories.
 - c) Service Mapping maps the downloaded file repository, i.e. source repository to destination repository.

Import Policy ✕

Service Type :

✕ hdfs ✕ hbase ✕ hive ✕ yarn ✕ Knox ✕ storm ✕ solr ✕ kafka

✕ atlas

Select file :

Select file 📁

Override Policy :

File Name : all_in_one.xls ✕

Specify service Mapping :

hadoopdev1	To	hdfs1	✕ ▼	✕
hivedev1	To	hive1	✕ ▼	✕
hbasedev1	To	hbase1	✕ ▼	✕

+

Cancel

Import

4. Click **Import**.

A confirmation message appears: “Success: File import successfully.”

Related Information

[Import Resource-Based Policies for a Specific Service](#)

Export Resource-Based Policies for a Specific Service

How to export the policies for a specific service (HBase, YARN, etc).

About this task

If you wish to export in Excel or CSV format, export the policies from the Reports page dropdown menu.

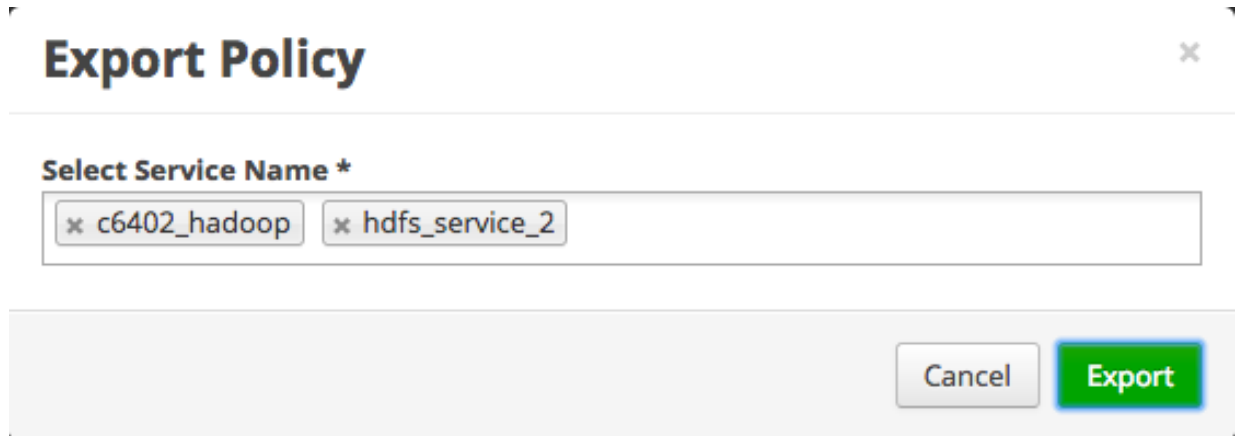
Procedure

1. From the Access Manager page, click the Export icon beside the service:



The Export Policy page opens.

2. Click the Export button.



The file downloads in your browser as a JSON file.

Related Information

[Export All Resource-Based Policies for All Services](#)

Export All Resource-Based Policies for All Services

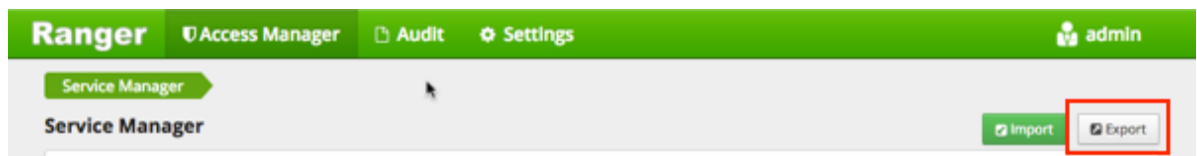
How to export the policies for all service.

About this task

If you wish to export in Excel or CSV format, export the policies from the Reports page dropdown menu.

Procedure

- From the Access Manager page:
 - a) Click the Export button:



The Export Policy page opens.

- b) Remove components or specific services and click Export.

Export Policy ✕

Service Type :

✕ hdfs
✕ hbase
✕ hive
✕ yarn
✕ Knox
✕ storm
✕ solr
✕ kafka

✕ atlas

Select Service Name *

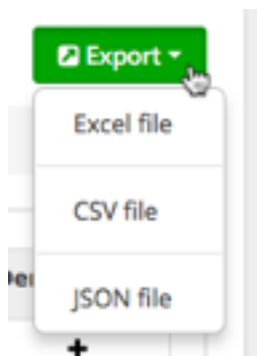
✕ c6402_hadoop
✕ hdfs_service_2
✕ c6402_hbase
✕ c6402_hive

✕ c6402_yarn
✕ c6402_knox
✕ c6402_atlas

Cancel
Export

The file downloads in your browser as a JSON file.

- From the Reports page:
 - a) Apply filters before exporting file.
 - b) Open the Export drop-down menu:



- c) Select the file format.
The file downloads in your browser.

Related Information

[Export Resource-Based Policies for a Specific Service](#)

Row-level Filtering and Column Masking in Hive

You can use Apache Ranger row-level filters to set access policies for rows in Hive tables. You can also use Ranger column masking to set policies that mask data in Hive columns, for example to show only the first or last four characters of column data.

Row-level Filtering in Hive with Ranger Policies

Row-level filtering helps simplify Hive queries. By moving the access restriction logic down into the Hive layer, Hive applies the access restrictions every time data access is attempted. This helps simplify authoring of the Hive query, and provides seamless behind-the-scenes enforcement of row-level segmentation without having to add this logic to the predicate of the query.

About this task

Row-level filtering also improves the reliability and robustness of Hadoop. By providing row-level security to Hive tables and reducing the security surface area, Hive data access can be restricted to specific rows based on user characteristics (such as group membership) and the runtime context in which this request is issued.

Typical use cases where row-level filtering can be beneficial include:

- A hospital can create a security policy that allows doctors to view data rows only for their own patients, and that allows insurance claims administrators to view only specific rows for their specific site.
- A bank can create a policy to restrict access to rows of financial data based on the employee's business division, locale, or based on the employee's role (for example: only employees in the finance department are allowed to see customer invoices, payments, and accrual data; only European HR employees can see European employee data).
- A multi-tenant application can create logical separation of each tenant's data so that each tenant can see only their own data rows.

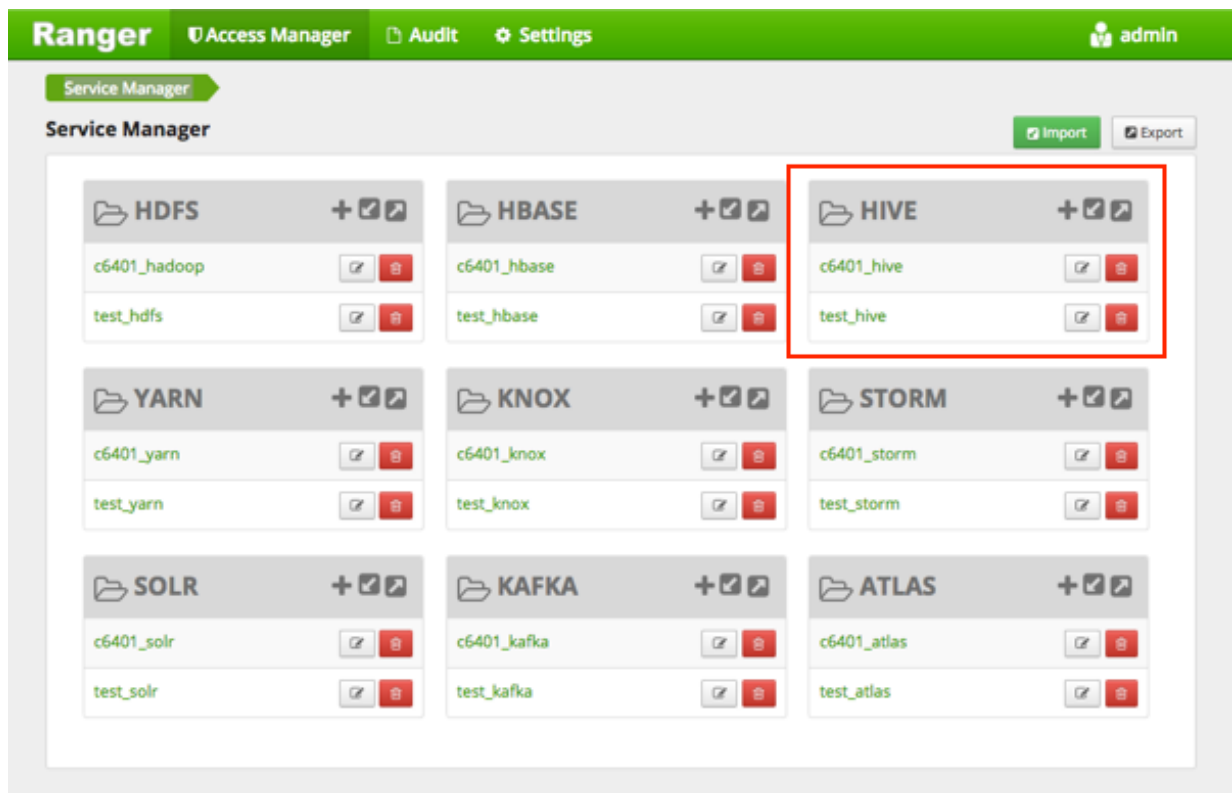
You can use Apache Ranger row-level filters to set access policies for rows in Hive tables. Row-level filter policies are similar to other Ranger access policies. You can set filters for specific users, groups, and conditions.

The following conditions apply when using row-level filters:

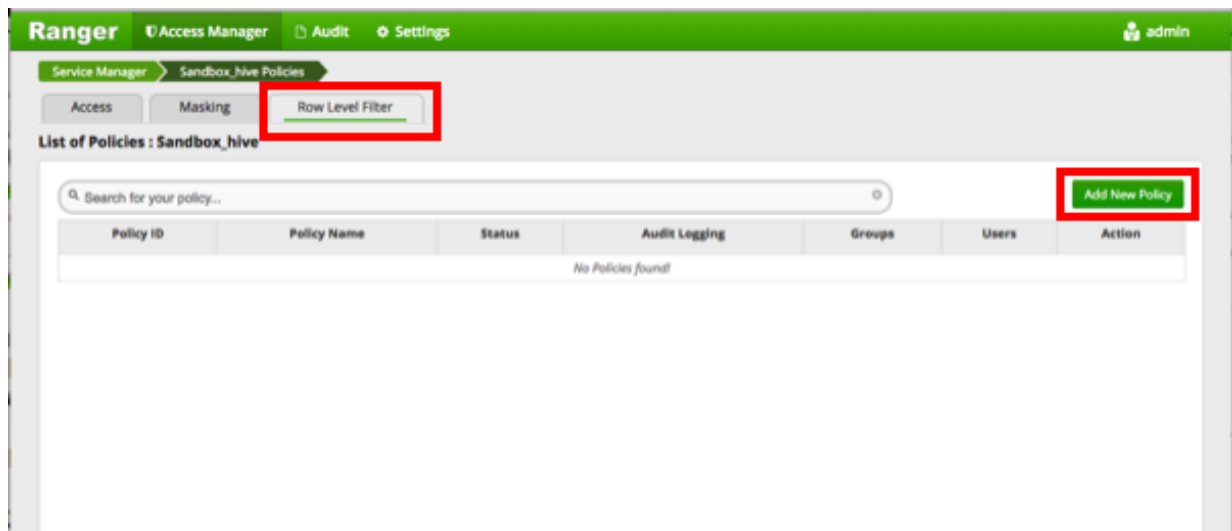
- The filter expression must be a valid WHERE clause for the table or view.
- Each table or view should have its own row-level filter policy.
- Wildcard matching is not supported on database or table names.
- Filters are evaluated in the order listed in the policy.
- An audit log entry is generated each time a row-level filter is applied to a table or view.

Procedure

1. On the Service Manager page, select an existing Hive Service.



2. Select the Row Level Filter tab, then click Add New Policy.



3. On the Create Policy page, add the following information for the row-level filter:

Table 39: Policy Details

Field	Description
Policy Name (required)	Enter an appropriate policy name. This name cannot be duplicated across the system. The policy is enabled by default.
Hive Database (required)	Type in the applicable database name. The auto-complete feature displays available databases based on the entered text.
Hive Table (required)	Type in the applicable table name. The auto-complete feature displays available tables based on the entered text.
Audit Logging	Audit Logging is set to Yes by default. Select No to turn off audit logging.
Description	Enter an optional description for the policy.

Table 40: Row Filter Conditions

Label	Description
Select Group	Specify the groups to which this policy applies. The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify one or more users to which this policy applies.
Access Types	Currently select is the only available access type. This will be used in conjunction with the WHERE clause specified in the Row Level Filter field.
Add Row Filter	<ul style="list-style-type: none"> To create a row filter for the specified users and groups, Click Add Row Filter, then type a valid WHERE clause in the Enter filter expression box. To allow Select access for the specified users and groups without row-level restrictions, do not add a row filter (leave the setting as "Add Row Filter"). Filters are evaluated in the order listed in the policy. The filter at the top of the Row Filter Conditions list is applied first, then the second, then the third, and so on.

Policy Details :

Policy Type **Row Level Filter**

Policy Name * row-filter:hr.employee **enabled**

Hive Database *

Hive Table *

Audit Logging **YES**

Description Row-level filter policy for hr.employee table

Row Filter Conditions :

Select Group	Select User	Access Types		
<input type="text" value="Select Group"/>	<input type="text" value="x admin"/>	<input type="button" value="select"/>	<input type="button" value="Add Row Filter"/>	<input type="button" value="x"/>
<input type="text" value="Select Group"/>	<input type="text" value="x ambari-qa"/>	<input type="button" value="select"/>	<input type="text" value="loc_state = 'CA'"/>	<input type="button" value="x"/>
<input type="text" value="x public"/>	<input type="text" value="Select User"/>	<input type="button" value="select"/>	<input type="text" value="loc_state in 'CA'"/>	<input type="button" value="x"/>

- To move a condition in the Row Filter Conditions list (and therefore change the order in which it is evaluated), click the dotted rows icon at the left of the condition row, then drag the condition to a new position in the list.

5. Click **Add** to add the new row-level filter policy.

Dynamic Resource-Based Column Masking in Hive with Ranger Policies

You can use Apache Ranger dynamic resource-based column masking capabilities to protect sensitive data in Hive in near real-time. You can set policies that mask or anonymize sensitive data columns (such as PII, PCI, and PHI) dynamically from Hive query output. For example, you can mask sensitive data within a column to show only the first or last four characters.

About this task

Dynamic column masking policies are similar to other Ranger access policies for Hive. You can set filters for specific users, groups, and conditions. With dynamic column-level masking, sensitive information never leaves Hive, and no changes are required at the consuming application or the Hive layer. There is also no need to produce additional protected duplicate versions of datasets.

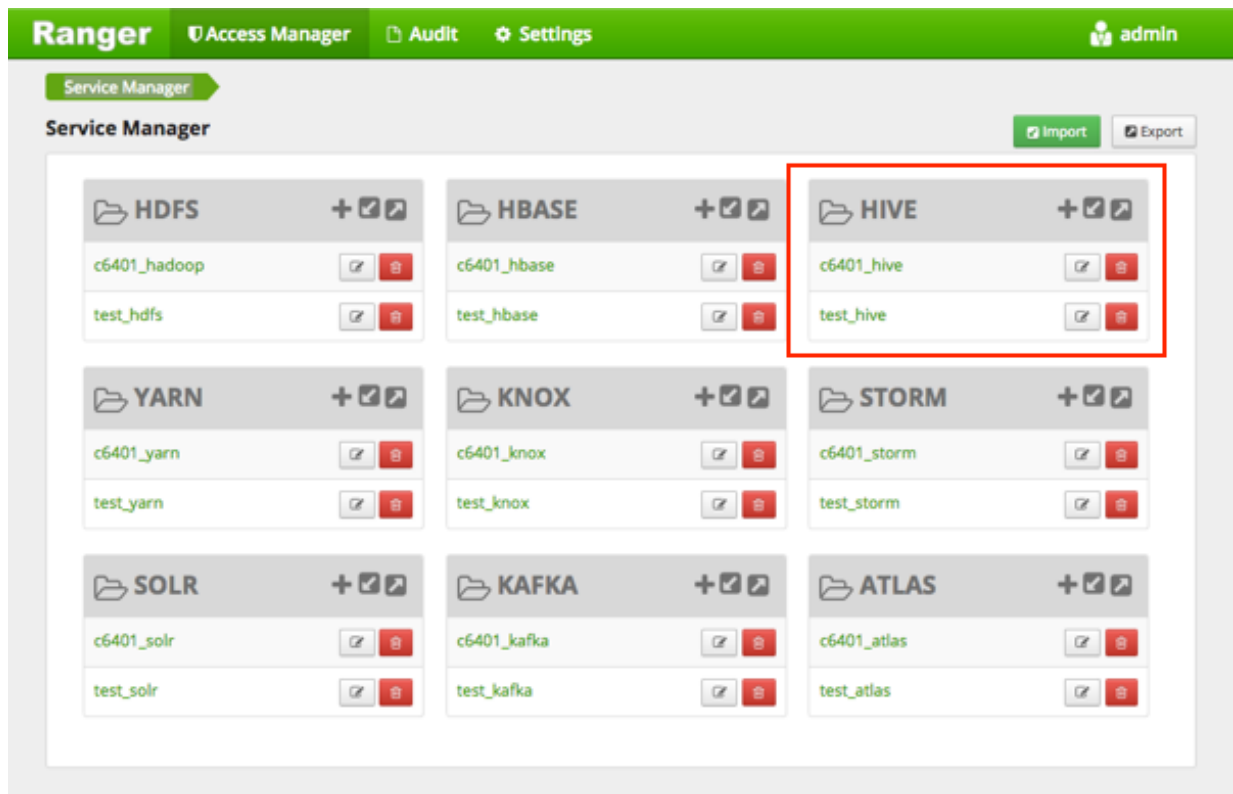
The following conditions apply when using Ranger column masking policies to mask data returned in Hive query results:

- A variety of masking types are available, such as show last 4 characters, show first 4 characters, Hash, Nullify, and date masks (show only year).
- You can specify a masking type for specific users, groups, and conditions.
- Wildcard matching is not supported.
- Each column should have its own masking policy.
- Masks are evaluated in the order listed in the policy.

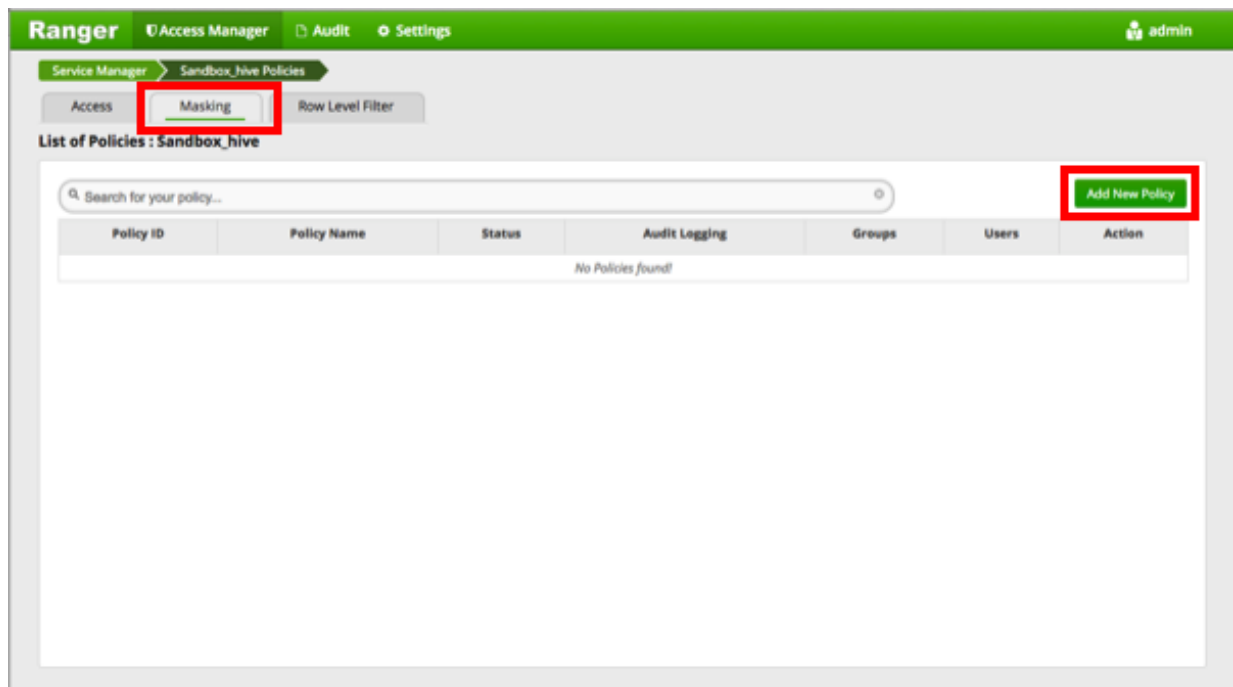
- An audit log entry is generated each time a masking policy is applied to a column.

Procedure

1. On the Service Manager page, select an existing Hive Service.



2. Select the Masking tab, then click Add New Policy.



3. On the Create Policy page, add the following information for the column-masking filter:

Table 41: Policy Details

Field	Description
Policy Name (required)	Enter an appropriate policy name. This name cannot be duplicated across the system. The policy is enabled by default.
Hive Database (required)	Type in the applicable database name. The auto-complete feature displays available databases based on the entered text.
Hive Table (required)	Type in the applicable table name. The auto-complete feature displays available tables based on the entered text.
Hive Column (required)	Type in the applicable column name. The auto-complete feature displays available columns based on the entered text.
Audit Logging	Audit Logging is set to Yes by default. Select No to turn off audit logging.
Description	Enter an optional description for the policy.

Table 42: Mask Conditions

Label	Description
Select Group	Specify the groups to which this policy applies. The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify one or more users to which this policy applies.
Access Types	Currently select is the only available access type.
Select Masking Type	<p>To create a row filter for the specified users and groups, click Select Masking Option, then select a masking type:</p> <ul style="list-style-type: none"> • Redact – mask all alphabetic characters with "x" and all numeric characters with "n". • Partial mask: show last 4 – Show only the last four characters. • Partial mask: show first 4 – Show only the first four characters. • Hash – Replace all characters with a hash of entire cell value. • Nullify – Replace all characters with a NULL value. • Unmasked (retain original value) – No masking is applied. • Date: show only year – Show only the year portion of a date string and default the month and day to 01/01 • Custom – Specify a custom masked value or expression. Custom masking can use any valid Hive UDF (Hive that returns the same data type as the data type in the column being masked). <p>Masking conditions are evaluated in the order listed in the policy. The condition at the top of the Masking Conditions list is applied first, then the second, then the third, and so on.</p>

Ranger Access Manager Audit Settings admin

Service Manager > Hive_service_1 Policies > Create Policy

Create Policy

Policy Details :

Policy Type: **Masking**

Policy Name: maskchr.employee.ssn **enabled**

Hive Database: hr

Hive Table: employee

Hive Column: ssn

Audit Logging: **YES**

Description: Masking for ssn column in hr.employee table

Mask Conditions :

Select Group	Select User	Access Types	Masking Option	Action
Select Group	admin	select	Unmasked (retain original value)	✖
Select Group	ambari-qa	select	Partial mask: show last 4	✖
public	Select User	select	Nullify	✖

Add **Cancel**

- To move a condition in the Mask Conditions list (and therefore change the order in which it is evaluated), click the dotted rows icon at the left of the condition row, then drag the condition to a new position in the list.

Ranger Access Manager Audit Settings admin

Service Manager Hive_service_1 Policies Create Policy

Create Policy

Policy Details :

Policy Type **Masking**

Policy Name * maskhr.employee.ssn **enabled**

Hive Database * hr

Hive Table * employee

Hive Column * ssn

Audit Logging **YES**

Description Masking for ssn column in hr.employee table

Mask Conditions :

Select Group	Select User	Access Types	Select Masking Option
Select Group	admin	select	Unmasked (retain original value)
Select Group	ambari-qa	select	Partial mask: show last 4
public	Select User	select	Nullify

Add **Cancel**

5. Click **Add** to add the new column masking filter policy.

Dynamic Tag-Based Column Masking in Hive with Ranger Policies

Where Ranger resource-based masking policy for Hive anonymizes data from a Hive column identified by the database, table, and column, tag-based masking policy anonymizes Hive column data based on tags and tag attribute values associated with Hive column (usually specified as metadata classification in Atlas).

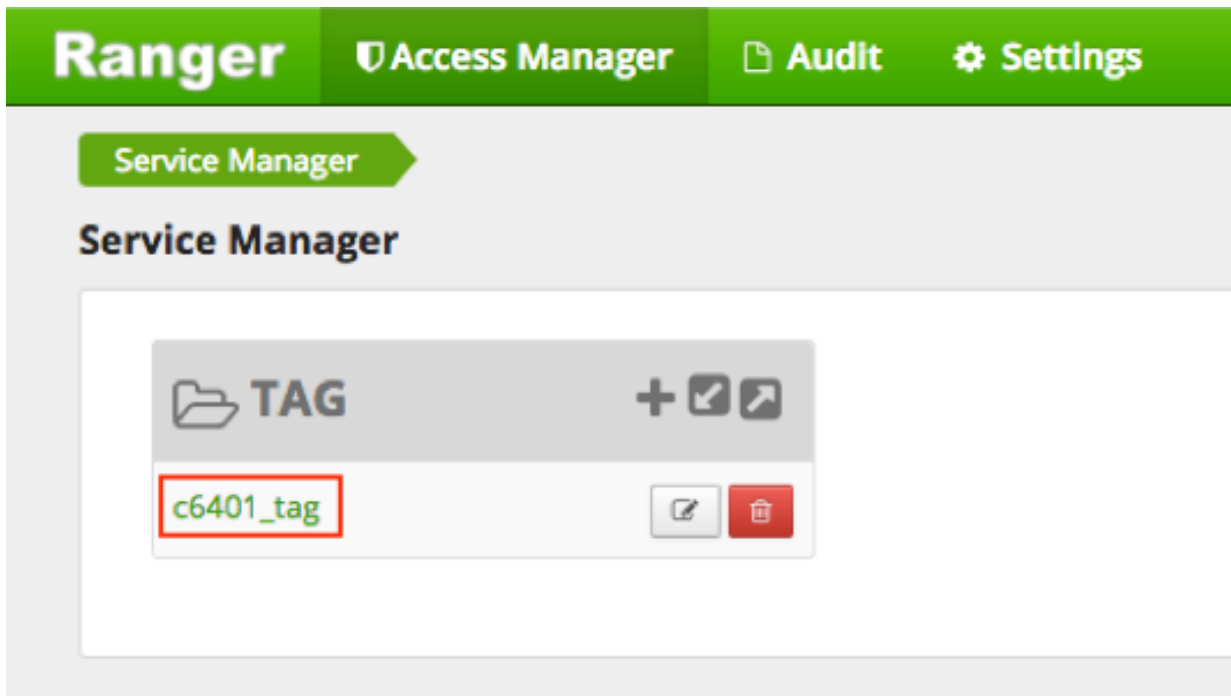
About this task

The following conditions apply when using Ranger column masking policies to mask data returned in Hive query results:

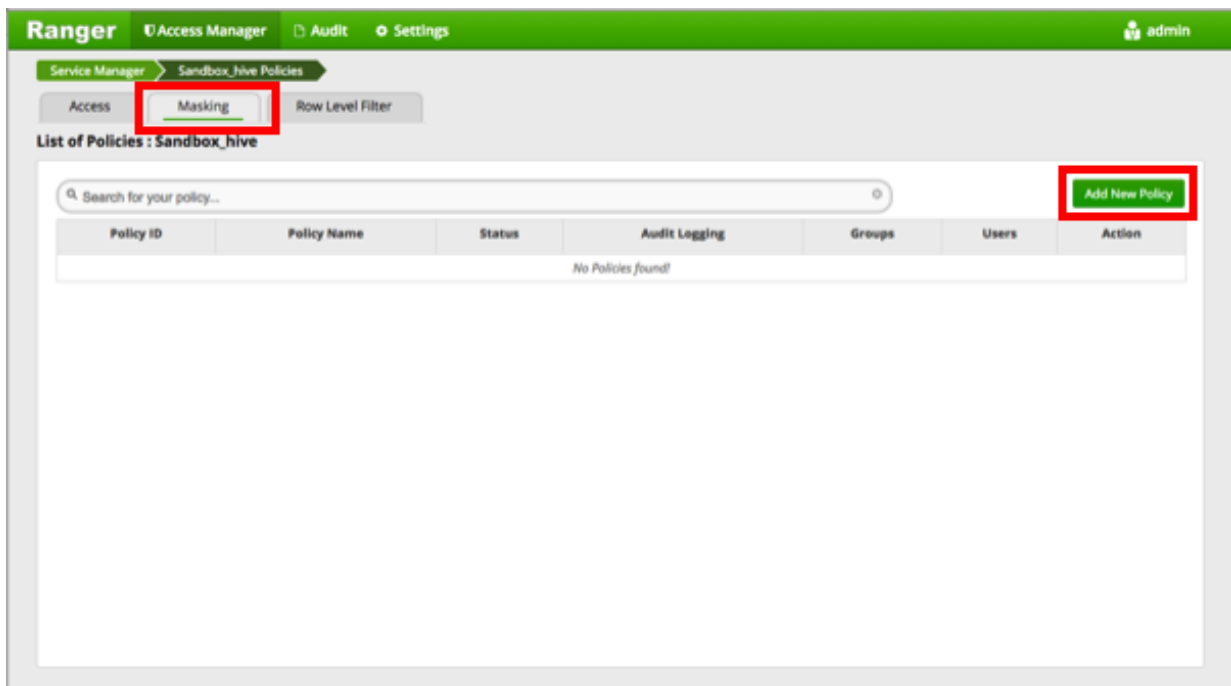
- A variety of masking types are available, such as show last 4 characters, show first 4 characters, Hash, Nullify, and date masks (show only year).
- You can specify a masking type for specific users, groups, and conditions.
- Wildcard matching is not supported.
- If there are multiple tag masking policies applied to the same Hive column, the masking policy with the lexicographically smallest policy-name is chosen for enforcement, E.G., policy "a" is enforced before policy "aa".
- Masks are evaluated in the order listed in the policy.
- An audit log entry is generated each time a masking policy is applied to a column.

Procedure

1. Select Access Manager > Tag Based Policies, then select a tag-based service.



2. Select the **Masking** tab, then click **Add New Policy**.



3. On the **Create Policy** page, add the following information for the column-masking filter:

Table 43: Policy Details

Field	Description
Policy Type (required)	Set to Hive by default.

Field	Description
Policy Name (required)	Enter an appropriate policy name. This name cannot be duplicated across the system. The policy is enabled by default.
TAG (required)	Enter the applicable tag name, E.G., MASK.
Audit Logging	Audit Logging is set to Yes by default. Select No to turn off audit logging.
Description	Enter an optional description for the policy.

Table 44: Mask Conditions

Label	Description
Select Group	Specify the groups to which this policy applies. The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify one or more users to which this policy applies.
Policy Conditions	Click Add Conditions to add or edit policy conditions. Currently "Accessed after expiry_date? (yes/no)" is the only available policy condition. To set this condition, type yes in the text box, then select the green check mark button to add the condition.
Access Types	Currently hive and select are the only available access types.
Select Masking Option	<p>To create a row filter for the specified users and groups, click Select Masking Option, then select a masking type:</p> <ul style="list-style-type: none"> • Redact – mask all alphabetic characters with "x" and all numeric characters with "n". • Partial mask: show last 4 – Show only the last four characters. • Partial mask: show first 4 – Show only the first four characters. • Hash – Replace all characters with a hash of entire cell value. • Nullify – Replace all characters with a NULL value. • Unmasked (retain original value) – No masking is applied. • Date: show only year – Show only the year portion of a date string and default the month and day to 01/01 • Custom – Specify a custom masked value or expression. Custom masking can use any valid Hive UDF (Hive that returns the same data type as the data type in the column being masked). <p>Masking conditions are evaluated in the order listed in the policy. The condition at the top of the Masking Conditions list is applied first, then the second, then the third, and so on.</p>

Ranger | Access Manager | Audit | Settings | admin

Service Manager > hive-tags Policies > Edit Policy

Edit Policy

Please ensure that users/groups listed in this policy have access to the tag via an Access Policy. This policy does not implicitly grant access to the tag.

Policy Details :

Policy Type: **Masking**

Policy ID: **21**

Policy Name *: masking policy [icon] **enabled**

TAG *: MASK

Audit Logging: **YES**

Description: Mask our Ranger resources marked with tag "MASK" for user

Mask Conditions :

Select Group	Select User	Policy Conditions	Access Types
Select Group	hive	Add Conditions	HIVE : Unmasked (retain original value)

Save Cancel Delete

- You can use the Plus (+) symbols to add additional conditions. Conditions are evaluated in the order listed in the policy. The condition at the top of the list is applied first, then the second, then the third, and so on.
- Click Add to add the new policy.

Tag-Based Services and Policies

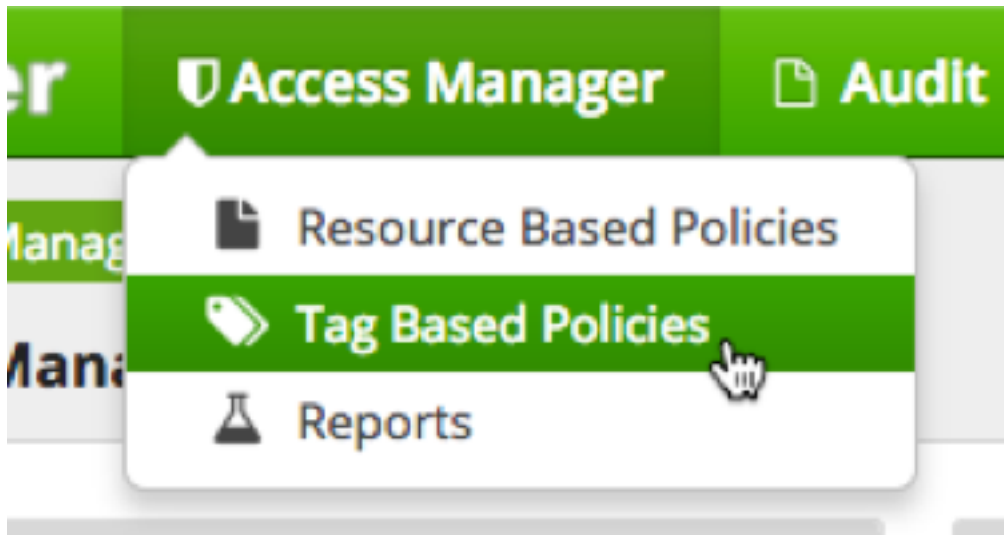
Ranger enables you to create tag-based services and add access policies to those services.

Adding a Tag-based Service

How to add a new tag-based service to Ranger.

About this task

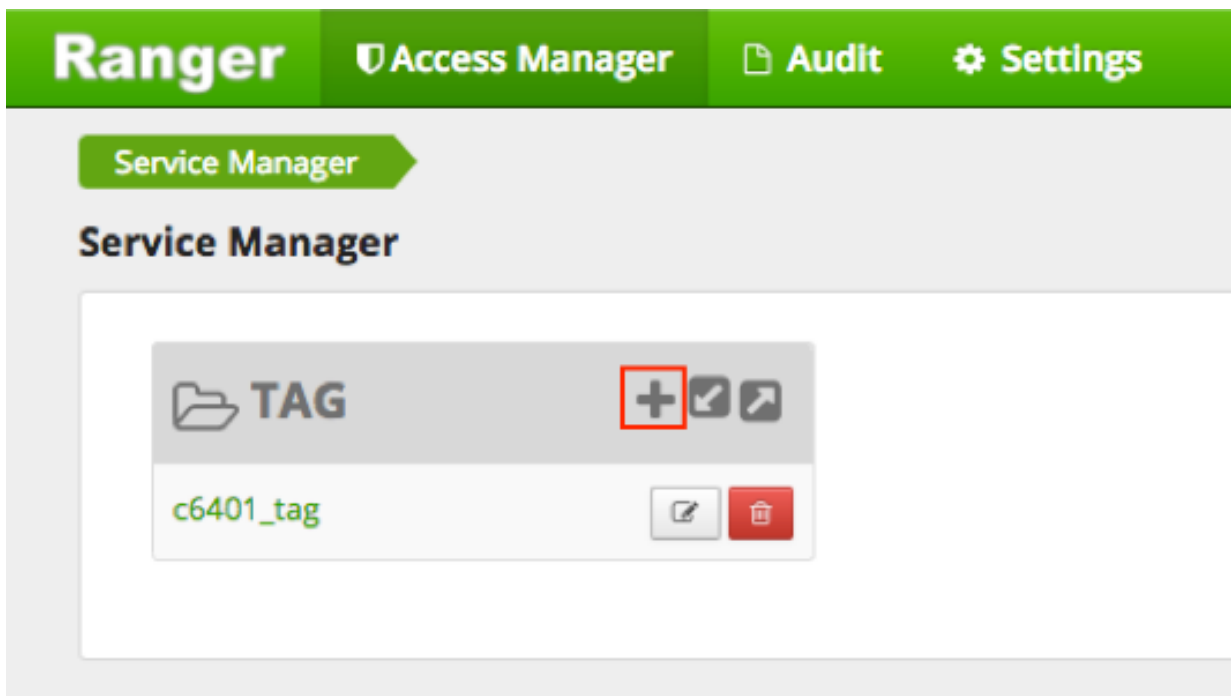
You can access the Service Manager for Tag-Based Policies page by selecting Access Manager > Tag Based Policies. You can use this page to create tag-based services and add tag-based access policies that can be applied to Hadoop resources. Using tag-based policies enables you to control access to resources across multiple Hadoop components without creating separate services and policies in each component. You can also use Ranger TagSync to synchronize the Ranger tag store with an external metadata service such as Apache Atlas.



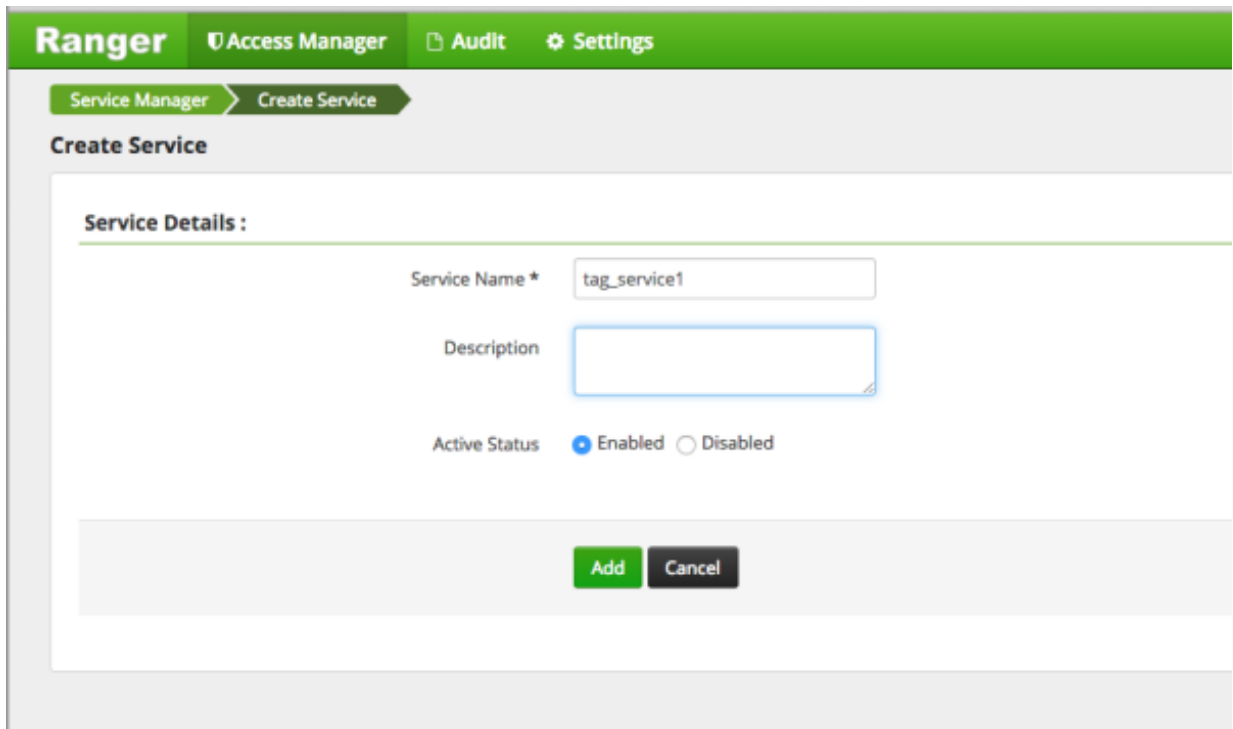
Procedure

1. Click the Add icon

()
in the TAG box on the Service Manager page.

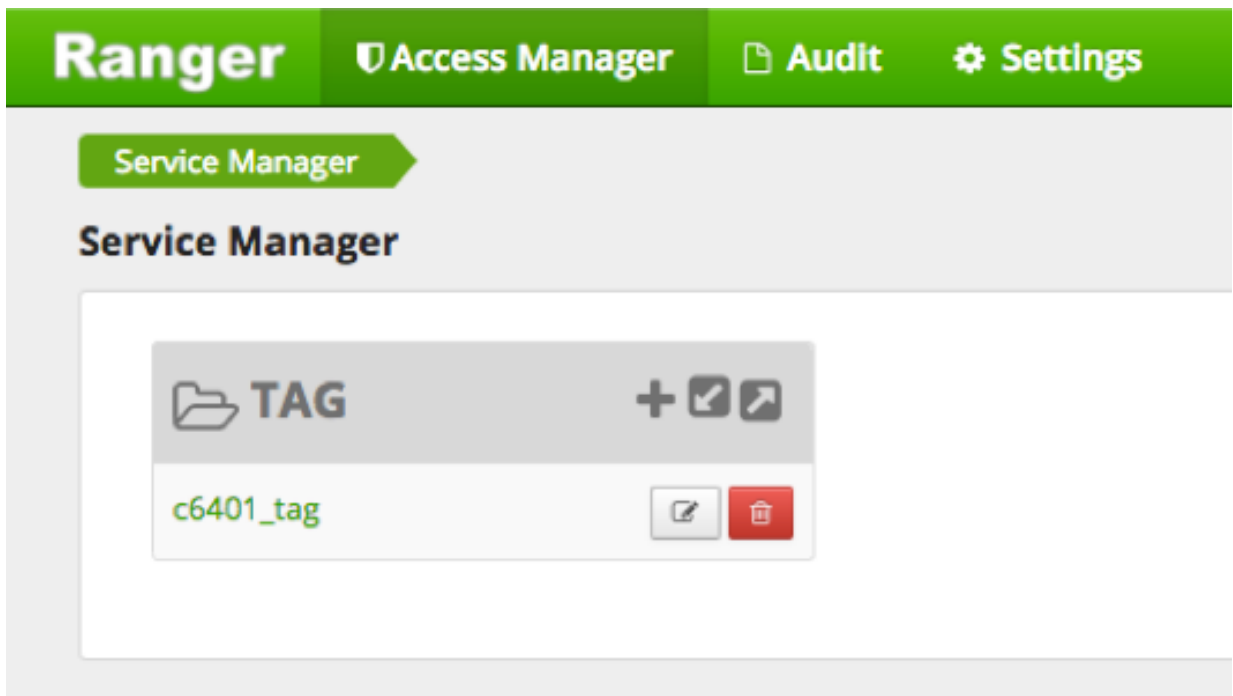


2. On the Service Details page, type in a service name and an optional description. The service is enabled by default, but you can disable it by selecting Disabled. To add the service, click **Add**.



The screenshot shows the 'Create Service' form in the Ranger interface. The top navigation bar includes 'Ranger', 'Access Manager', 'Audit', and 'Settings'. Below this, there are tabs for 'Service Manager' and 'Create Service'. The main heading is 'Create Service'. Under 'Service Details', there are three fields: 'Service Name *' with the value 'tag_service1', 'Description' (empty), and 'Active Status' with radio buttons for 'Enabled' (selected) and 'Disabled'. At the bottom of the form are 'Add' and 'Cancel' buttons.

3. The new tag service appears on the Service Manager page.

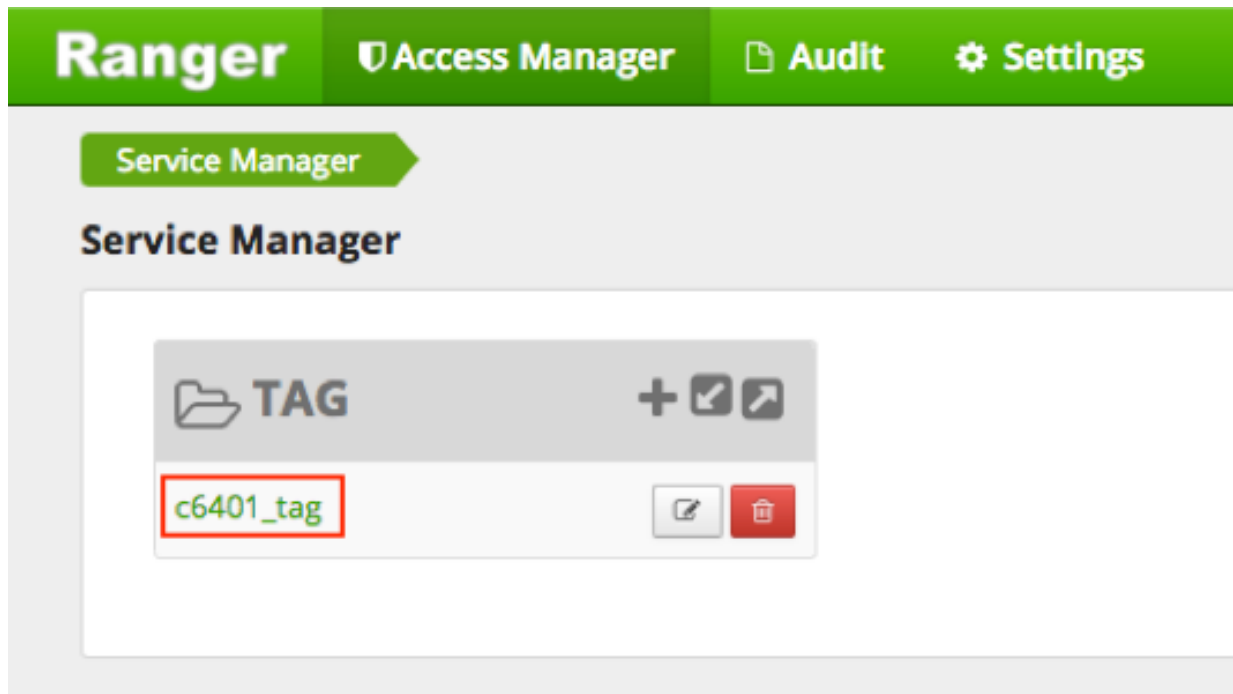


Adding Tag-Based Policies

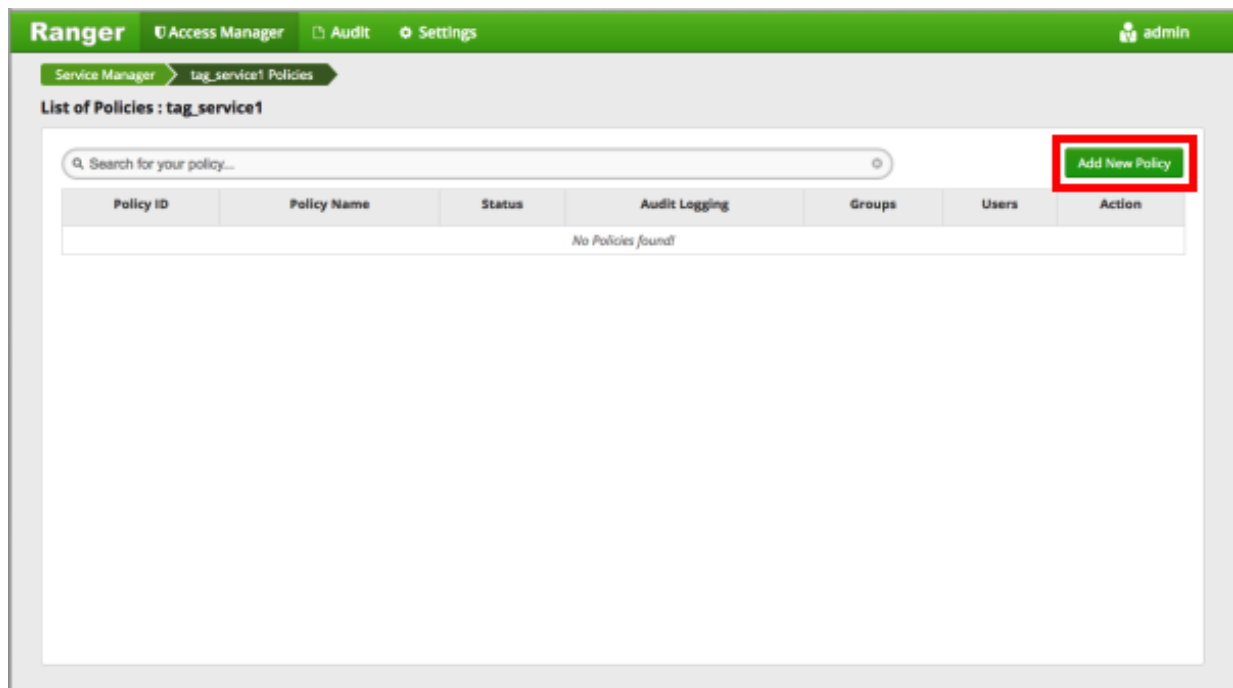
Tag-based policies enable you to control access to resources across multiple Hadoop components without creating separate services and policies in each component. You can also use Ranger TagSync to synchronize the Ranger tag store with an external metadata service such as Apache Atlas.

Procedure

1. Select Access Manager > Tag Based Policies, then select a tag-based service.



2. On the List of Policies page, click Add New Policy.



The Create Policy page appears:

3. Enter information on the Create Policy page as follows:

Table 45: Policy Details

Field	Description
Policy Type	Set to Access by default.
Policy Name	Enter an appropriate policy name. This name cannot be duplicated across the system. This field is mandatory.
TAG	Enter the applicable tag name.
Description	(Optional) Describe the purpose of the policy.
Audit Logging	Specify whether this policy is audited. (De-select to disable auditing).
Policy Label	Specify a label for this policy. You can search reports and filter policies based on these labels.

Table 46: Allow, Exclude from Allow, Deny, and Exclude from Deny Conditions

Label	Description
Select Group	Specify the group to which this policy applies. To designate the group as an Administrator for the chosen resource, specify Admin permissions. (Administrators can create child policies based on existing policies). The public group contains all users, so setting a condition for the public group applies to all users.
Select User	Specify a particular user to which this policy applies (outside of an already-specified group) OR designate a particular user as Admin for this policy. (Administrators can create child policies based on existing policies).
Policy Conditions	Click Add Conditions to add or edit policy conditions. " Accessed after expiry_date (yes/no)? ": To set this condition, type yes in the text box, then select the green check mark button to add the condition. Enter boolean expression: Available for allow or deny conditions on tag-based policies. For examples and details, see "Using Tag Attributes and Values in Ranger Tag-Based Policy Conditions".
Component Permissions	Click Add Permissions to add or edit component conditions. To add component permissions, enter the component name in the text box, then use the check boxes to specify component permissions. Select the green check mark button to add the chosen component conditions to the policy.

Note:

Currently there is no Atlas hook for HBase, HDFS, or Kafka. For these components, you must "Manually Creating Entities in Atlas". You can then associate tags with these entities and control access using Ranger tag-based policies.

- You can use the Plus (+) symbols to add additional conditions. Conditions are evaluated in the order listed in the policy. The condition at the top of the list is applied first, then the second, then the third, and so on.
- Click Add to add the new policy.

Related Information

[Using Tag Attributes and Values in Ranger Tag-Based Policy Conditions](#)

[Using Basic and Advanced Search](#)

Using Tag Attributes and Values in Ranger Tag-Based Policy Conditions

Enter boolean expression allows Ranger to use tag attributes and values when configuring tag-based policy Allow or Deny conditions. It allows admins to provide boolean expression(s) using tag attributes.

The policy condition is introduced in the tag service definition:

```
{
  "itemId":2,
  "name":"expression",
  "evaluator":
  "org.apache.ranger.plugin.conditionevaluator.RangerScriptConditionEvaluator",
  "evaluatorOptions" : {"engineName":"JavaScript",
  "ui.isMultiline":"true"},
  "label":"Enter boolean expression",
  "description": "Boolean expression"
}
```

The following variables can be referenced in the boolean expression:

- ctx: Context handler containing APIs to access metadata information from the request.
- tag: Information about the current tag.
- tagAttr: Map containing all the current tag attributes and corresponding values.

The following APIs available from the request:

- getUser(): Returns a string.
- getUserGroups(): Returns a set of strings containing groups.
- getClientIPAddress(): Returns a string containing client IP address.
- getAction(): Returns a string containing information about the action being requested.

For two scenarios:

- User “sam” needs to be denied a policy based on the IP address of the machine from where the resources are accessed.

Set the deny condition for user sam with the following boolean expression:

```
if ( tagAttr.get('ipAddr').equals(ctx.getClientIPAddress()) ) {
  ctx.result = true;
}
```

- Deny one particular user, “bob” from a group, “users”, only when this user is accessing resources from a particular IP defined as an tag attribute in Atlas.

Set the deny condition for group users with the following boolean expression:

```
if ( tagAttr.get('ipAddr').equals(ctx.getClientIPAddress()) &&
  ctx.getUser().equals("bob") ) {
  ctx.result=true;
}
```

The screenshot shows the "Deny Conditions" configuration interface. It features a table with four columns: "Select Group", "Select User", "Policy Conditions", and "Component Permissions".

- Row 1:**
 - Select Group: users
 - Select User: sam
 - Policy Conditions: `(tagAttr.get('ipAddr').equals(ctx.getClientIPAddress())) { ctx.result = true;}`
 - Component Permissions: DENY
- Row 2:**
 - Select Group: users
 - Select User: bob
 - Policy Conditions: `(tagAttr.get('ipAddr').equals(ctx.getClientIPAddress()) && ctx.getUser().equals('bob')) { ctx.result=true;}`
 - Component Permissions: DENY

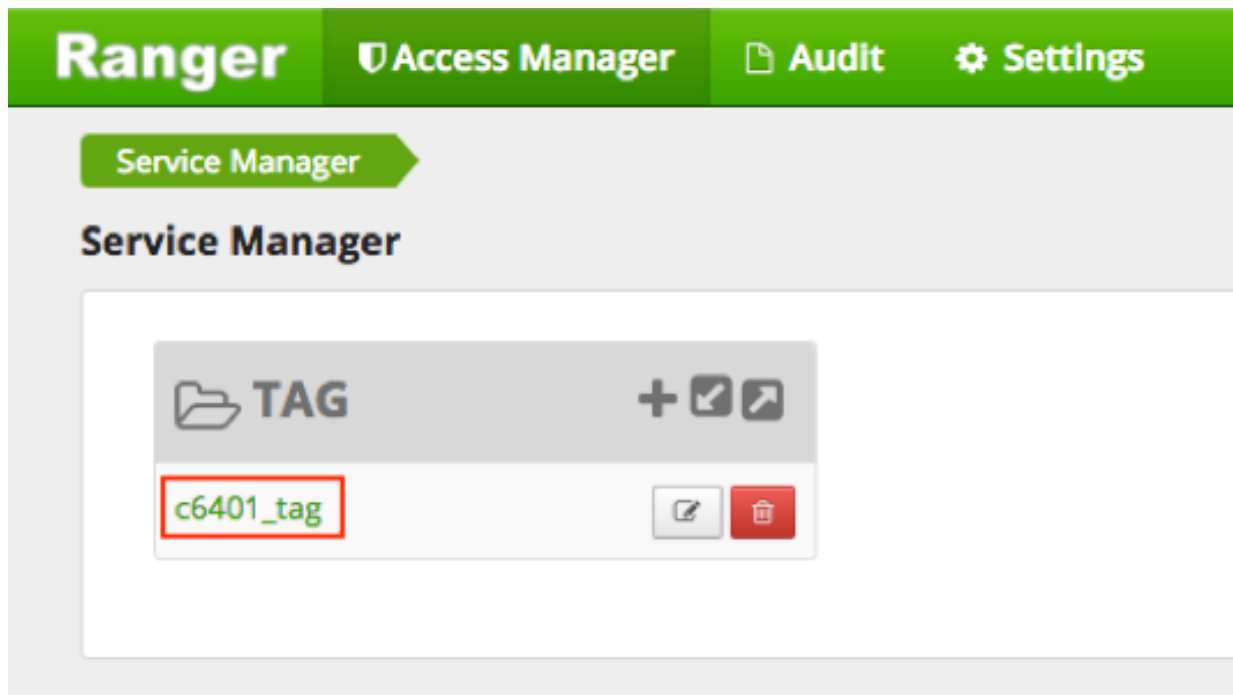
Red callout boxes highlight the JavaScript expressions in the "Policy Conditions" column, with arrows pointing from the text above and below to the corresponding cells in the table.

Adding a Tag-Based PII Policy

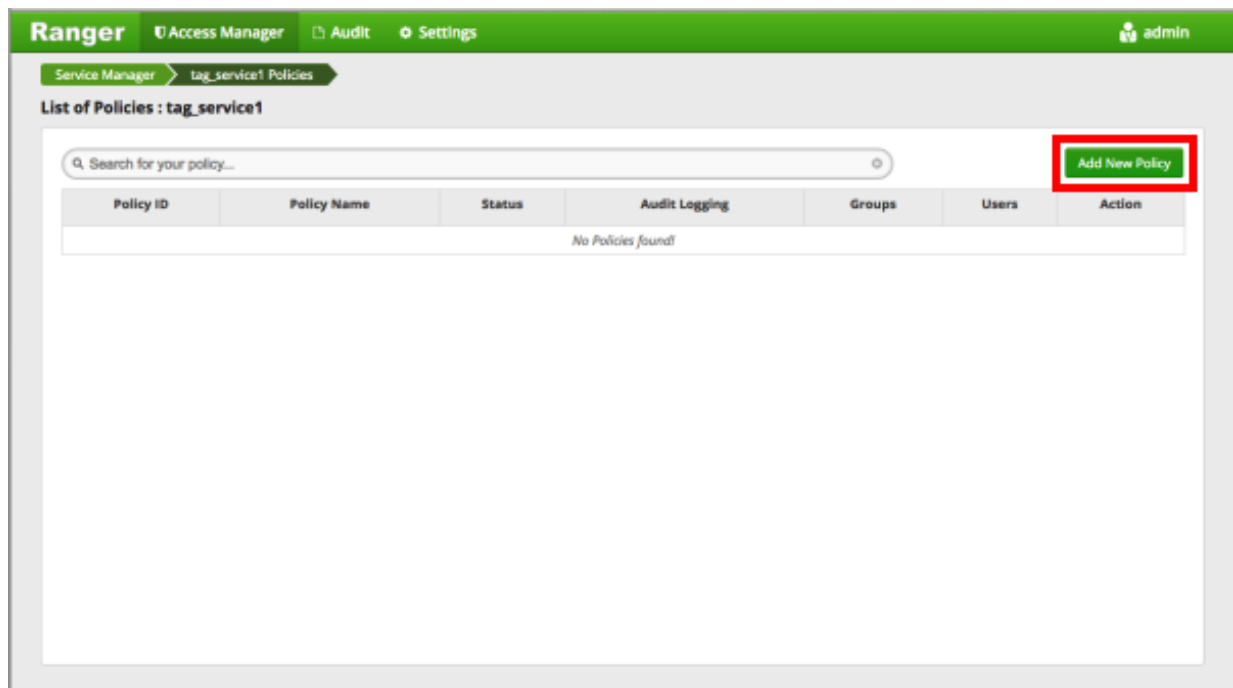
Example of how to add a PII tag-based policy. In this example we create a tag-based policy for objects tagged "PII" in Atlas. Access to objects tagged "PII" is allowed for members of the "audit" group. All other users (the "public" group) are denied access.

Procedure

1. Select Access Manager > Tag Based Policies, then select a tag-based service.



2. On the List of Policies page, click Add New Policy.



The Create Policy page appears:

3. Enter the following information on the Create Policy page:

Table 47: Policy Details

Field	Description
Policy Type	Set to Access by default.
Policy Name	PII
TAG	PII
Audit Logging	YES
Description	Restrict access to resources with the PII tag.

Table 48: Allow Conditions

Label	Description
Select Group	audit
Select User	<none>
Policy Conditions	<none>
Component Permissions	hive (select all permissions)

Table 49: Deny Conditions

Label	Description
Select Group	public
Select User	<none>
Policy Conditions	<none>
Component Permissions	hive (select all permissions)

Table 50: Exclude from Allow Conditions

Label	Description
Select Group	audit
Select User	<none>
Policy Conditions	<none>
Component Permissions	hive (select all permissions)

The screenshot shows the Ranger Access Manager interface for configuring a policy. The policy is named "PII" and is currently enabled. The TAG is set to "PII". Audit logging is turned on (YES). The description is "Restrict access to resources with the PII tag".

Allow Conditions:

Select Group	Select User	Policy Conditions	Component Permissions
audit	Select User	Add Conditions +	HIVE [edit] [delete]

Deny Conditions:

Select Group	Select User	Policy Conditions	Component Permissions
public	Select User	Add Conditions +	HIVE [edit] [delete]

Exclude from Deny Conditions:

Select Group	Select User	Policy Conditions	Component Permissions
audit	Select User	Add Conditions +	HIVE [edit] [delete]

At the bottom, there are "Add" and "Cancel" buttons.

In this example we used Allow Conditions to grant access to the "audit" group, and then used Deny Conditions to deny access to the "public" group. Because the "public" group includes all users, we then used Exclude from Deny Conditions to exclude the "audit" group, in effect reinstating the "audit" group's original Allow access condition.

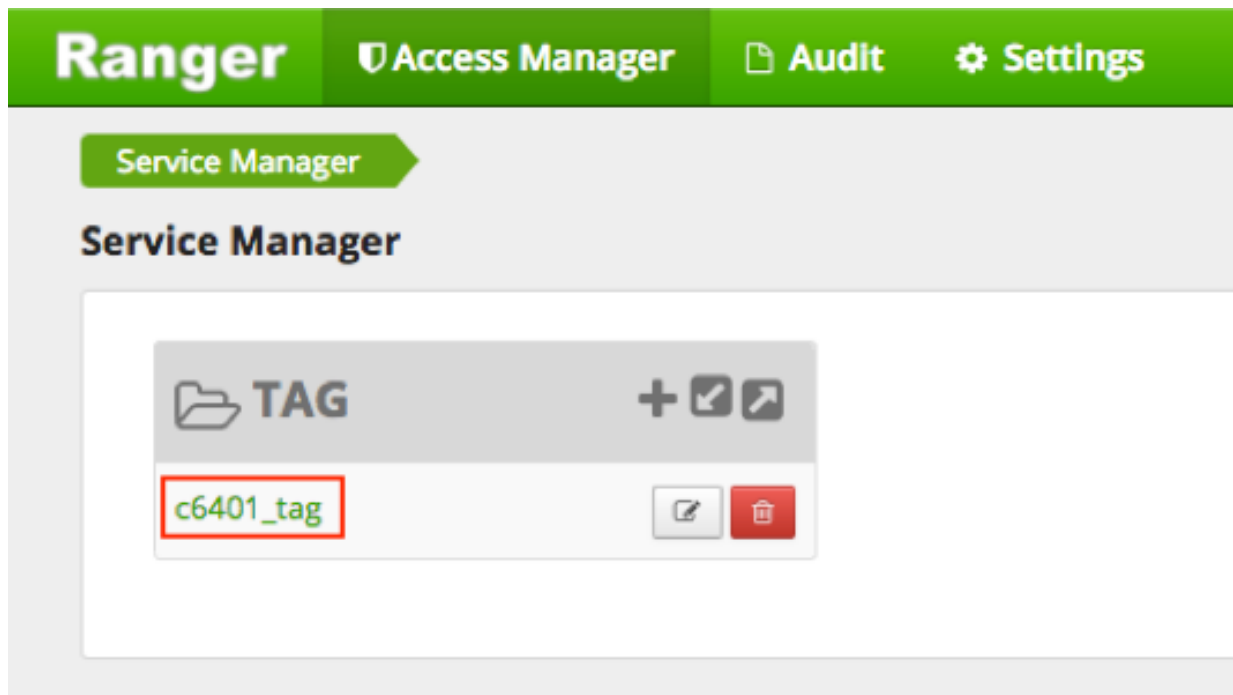
4. Click Add to add the new policy.

Tag Policy Default EXPIRES_ON Policy

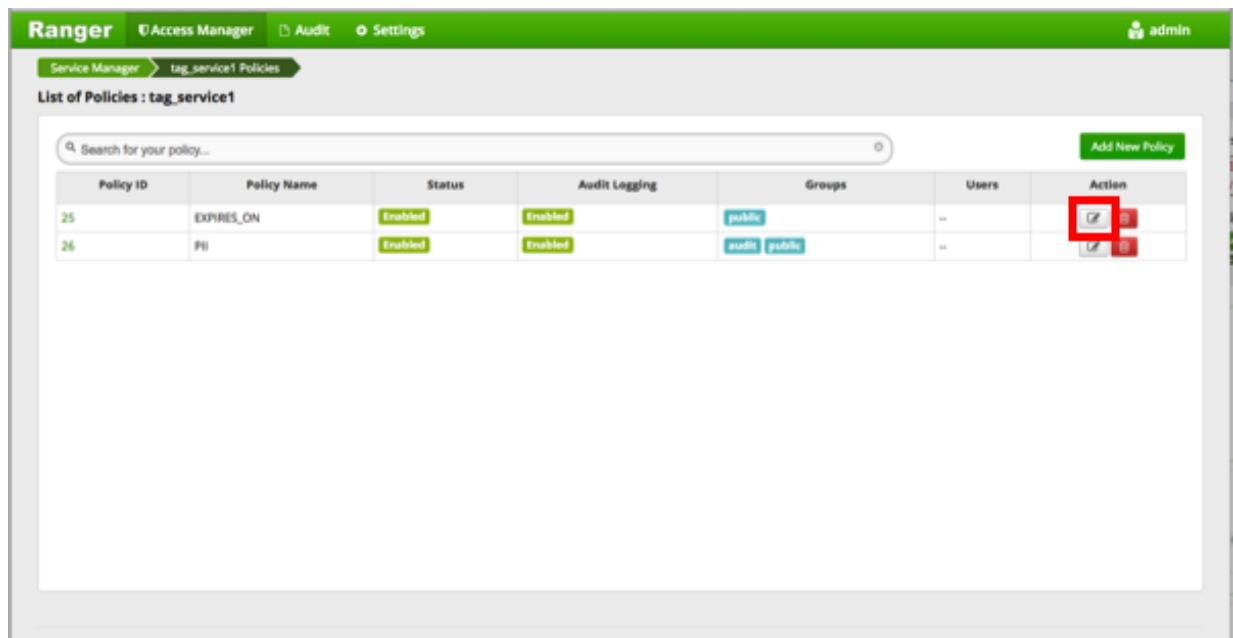
An EXPIRES_ON tag-based policy is created automatically when a tag service instance created. This default policy denies access to objects tagged with EXPIRES_ON after the expiry date specified in the Atlas tag attribute. You can use the following steps to review the default EXPIRES_ON policy.

Procedure

1. Select Access Manager > Tag Based Policies, then select a tag-based service.



2. On the List of Policies page, click the Edit icon for the default EXPIRES_ON policy.



The Edit Policy page appears:

The screenshot shows the 'Edit Policy' interface in Ranger. The top navigation bar includes 'Ranger', 'Access Manager', 'Audit', 'Settings', and a user profile 'admin'. The main content area is titled 'Edit Policy' and contains the following sections:

- Policy Details:**
 - Policy Type: Access
 - Policy ID: 25
 - Policy Name*: EXPIRES_ON (with an 'enabled' toggle)
 - TAG*: EXPIRES_ON
 - Audit Logging: YES
 - Description: Policy for data with EXPIRES_ON tag
- Allow Conditions:**
 - Select Group: Select Group
 - Select User: Select User
 - Policy Conditions: Add Conditions +
 - Component Permissions: Add Permissions +
 - Exclude from Allow Conditions: show +
- Deny Conditions:**
 - Select Group: is public
 - Select User: Select User
 - Policy Conditions: accessed-after-e:expiry:yes
 - Component Permissions: HDFS, HBASE, HIVE, KMS, KNOX, STORM, YARN, KAFKA, SOLR, ATLAS
 - Exclude from Deny Conditions: show +

At the bottom of the form are three buttons: Save (green), Cancel (grey), and Delete (red).

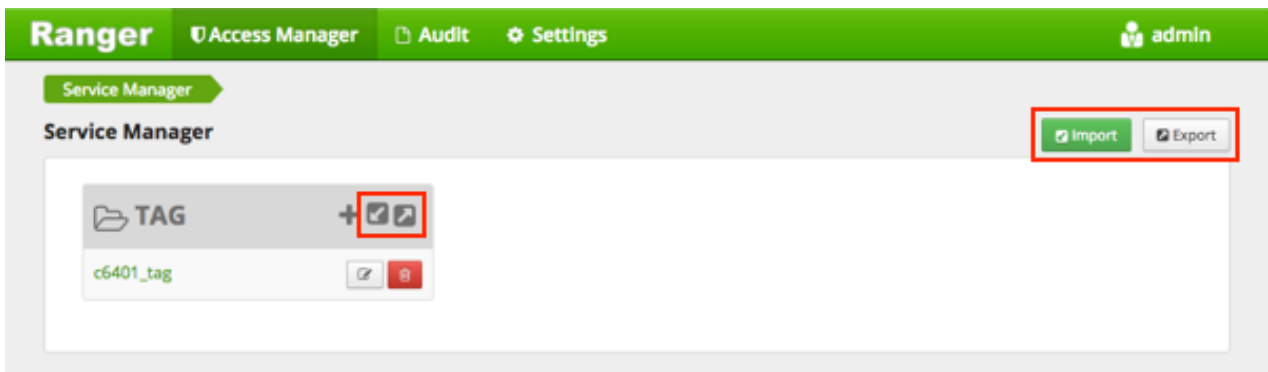
- We can see that the default EXPIRES_ON policy denies access to all users, and for all components, after the expiry date specified in the Atlas tag attribute.

Importing and Exporting Tag-Based Policies

You can export and import policies from the Ranger Admin UI for cluster resiliency (backups), during recovery operations, or when moving policies from test clusters to production clusters. You can export/import a specific subset of policies (such as those that pertain to specific resources or user/groups) or clone the entire repository (or multiple repositories) via Ranger Admin UI.

Interfaces

You can import and export policies from the Access Manager>Tag Based Policies page:



You can also export policies from the Reports page:

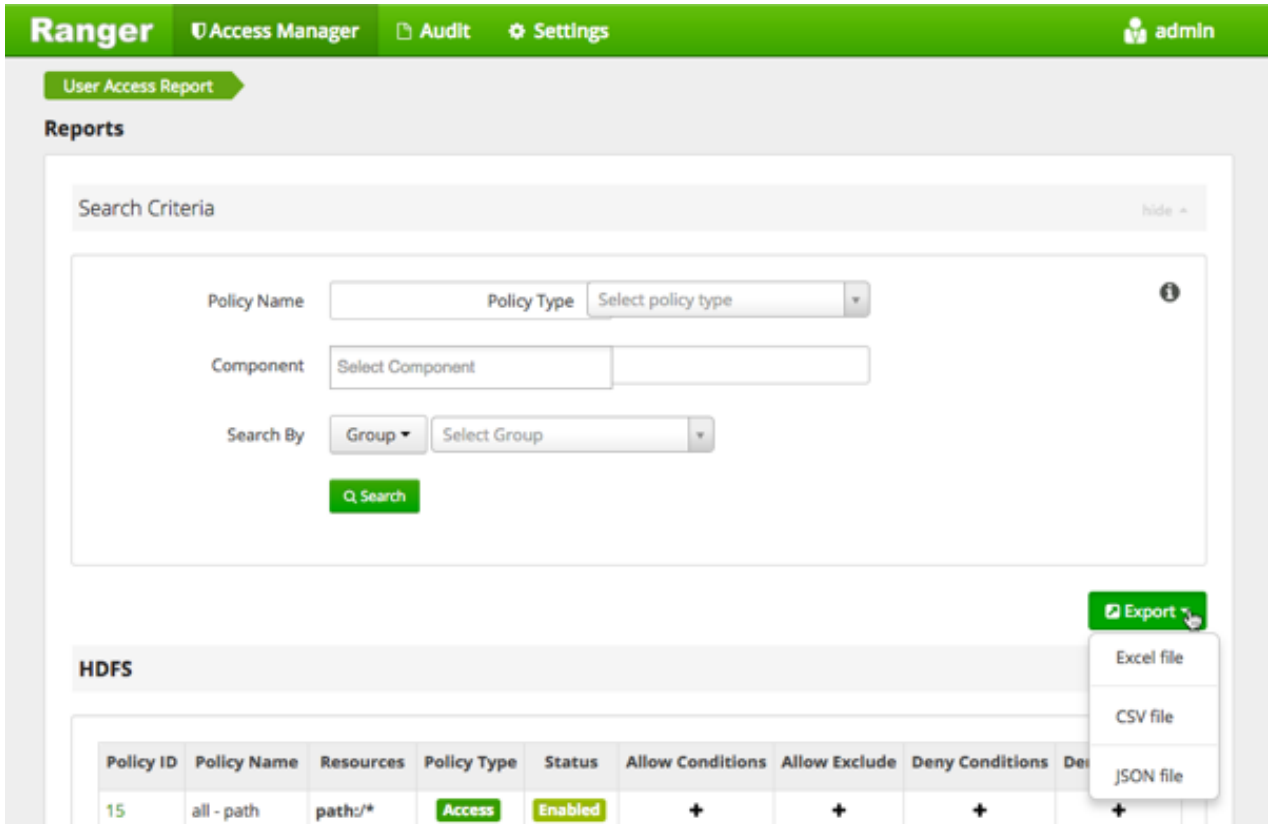


Table 51: Export Policy Options

	Access Manager Page	Reports Page
Formats	JSON	JSON Excel CSV
Filtering Supported	No	Yes
Specific Service Export	Yes	Via filtering

Filtering

When exporting from the Reports page, you can apply filters before saving the file.

Export Formats

You can export policies in the following formats:

- Excel
- JSON
- CSV

Note: CSV format is not supported for importing policies.

When you export policies from the Access Manager page, the policies are automatically downloaded in JSON format. If you wish to export in Excel or CSV format, export the policies from the Reports page dropdown menu.

Required User Roles

The Ranger admin user can import and export only Resource & Tag based policies. The credentials for this user are set in Ranger Configs > Advanced ranger-env in the fields labeled admin_username (default: admin/admin).

The Ranger KMS keyadmin user can import and export only KMS policies. The default credentials for this user are keyadmin/keyadmin.

Limitations

To successfully import policies, use the following database versions:

- MariaDB: 10.1.16+
- MySQL: 5.6.x+
- Oracle: 11gR2+
- PostgreSQL: 8.4+
- MS SQL: 2008 R2+

Partial policy import is not supported.

Related Information

[Importing and Exporting Resource-Based Policies](#)

Import Tag Based Policies

How to import tag-based policies.

Procedure


- Via the Import icon:
 - a) From the Access Manager>Tag Based Policies page, click the Import icon beside the service:



The Import Policy page opens.

Import Policy [Close]

Select File :

Select file  Override Policy :

No file chosen

Specify Service Mapping :

Source Destination

Enter service name To Select service name ▼ ✘

+

Cancel Import

b) Select the file to import.

You can only import policies in JSON format.

c) (Optional) Configure the import operation:

- The Override Policy option deletes all policies of the destination repositories.
- Service Mapping maps the downloaded file repository, i.e. source repository to destination repository.

For example:

Import Policy

Select File :

Select file

Override Policy :

Ranger_Policies_20170209_192920.json

Specify Service Mapping :

Source Destination

tagdev1 To tag1

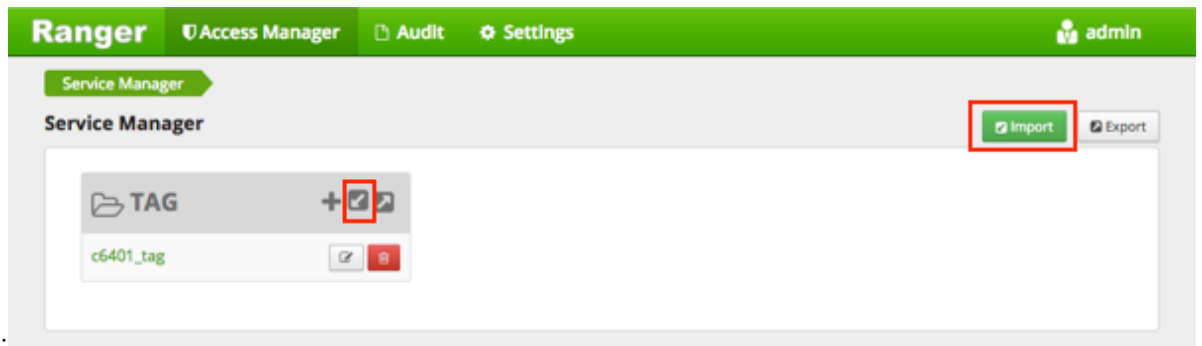
Cancel Import

d) Click **Import**.

A confirmation message appears: “Success: File import successfully.”

• Via the Import button:

a) From the Access Manager>Tag Based Policies page, click the Import



button:

The Import Policy page opens.

Import Policy ✕

Service Type :

✕ tag

Select File :

Select file

No file chosen

Override Policy :

Specify Service Mapping :

Source	To	Destination
Enter service name	To	Select service name ▼ ✕
<div style="border: 1px solid #ccc; padding: 2px; display: inline-block; margin-bottom: 10px;">+</div>		

Cancel

Import

b) Select the file to import.

You can only import policies in JSON format.

c) (Optional) Configure the import operation:

- Service Types enables you to remove specific services from the import.
- The Override Policy option deletes all policies of the destination repositories.
- Service Mapping maps the downloaded file repository, i.e. source repository to destination repository.

For example:

Import Policy ✕

Select File :

Select file

Override Policy :

Ranger_Policies_20170209_192920.json ✕

Specify Service Mapping :

Source	To	Destination	
tagdev1	To	tag1 ✕	✕
<div style="border: 1px solid #ccc; padding: 2px 10px; display: inline-block; text-align: center;">+</div>			

Cancel

Import

d) Click **Import**.

A confirmation message appears: “Success: File import successfully.”

Related Information

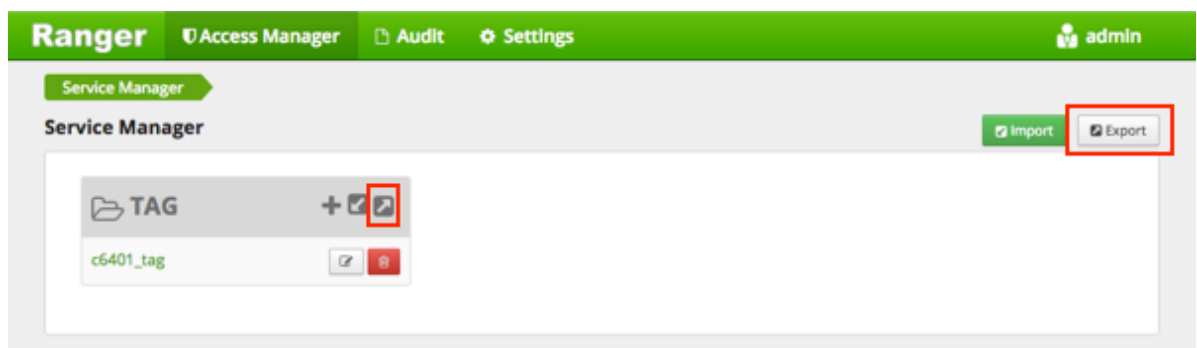
[Export Tag-Based Policies](#)

Export Tag-Based Policies

How to export all tag-based policies.

Procedure

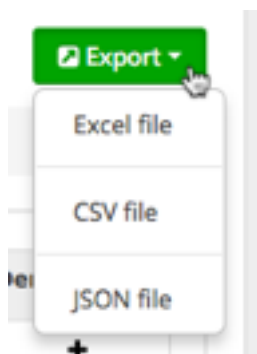
- From the Access Manager>Tag Based Policies page:
 - a) Click the Export button or icon:



The Export Policy page opens.

- b) Remove components or specific services and click Export.

- c) The file downloads in your browser as a JSON file.
If you wish to export in Excel or CSV format, export the policies from the Reports page dropdown menu.
- From the Reports page:
 - a) Filter **Component** to tag and click **Search**.
 - b) (Optional) Apply filters before exporting file.
 - c) Open the Export drop-down menu:



- d) Select the file format.
The file downloads in your browser.

Related Information

[Import Tag Based Policies](#)

Create a Time-bound Policy

Where Apache Ranger policies used to be permanent once authored, as of HDP 3.0, you can now create a time-bound policy. This enables you to configure a policy to be effective for a specified time range. You can add a validity period to resource- and tag-based policies.

About this task

For example, you may want to create a time-bound policy for:

- Financial information about earnings that is sensitive and restricted only until the earnings release date.
- Block a certain user for a specific time period (e.g., a compromised user account being investigated needs to be put on "hold" from accessing resources in Hadoop services).

- Block a certain group for a specific time (e.g., excluding temporary employees from writing on resources during the holiday season).

Procedure

- From Ranger, click **Access Manager > Resource Based Policies | Tag Based Policies > <select the service> > Add New Policy**

The image consists of two screenshots from the Ranger web interface. The top screenshot shows the 'Resource Based Policies' menu for the 'HBASE' service, with 'dwweekly_hbase' selected. The bottom screenshot shows the 'List of Policies : dwweekly_hbase' page with the 'Add New Policy' button highlighted.

Top Screenshot: Ranger - Resource Based Policies

Navigation: Service Manager > Resource Based Policies > dwweekly_hbase

Services and Policies:

- HDFS: dwweekly_hadoop
- HBASE: dwweekly_hbase (selected)
- HIVE: dwweekly_hive
- YARN: dwweekly_yarn
- KNOX: dwweekly_knox
- STORM
- SOLR
- KAFKA
- NIFI
- KYLIN
- SQOOP
- ATLAS: dwweekly_atlas

Bottom Screenshot: Ranger - dwweekly_hbase Policies

Navigation: Service Manager > dwweekly_hbase Policies

List of Policies : dwweekly_hbase

Search for your policy...

Add New Policy

Policy ID	Policy Name	Policy Labels	Status	Audit Logging	Groups	Users	Action
3	all - table, column-family, column	--	Enabled	Enabled	--	hbase	[View] [Edit] [Delete]
4	Service Check User Policy for Hbase	--	Enabled	Enabled	--	ambari-qa	[View] [Edit] [Delete]
11	grant-1532451641294	--	Enabled	Enabled	--	atlas	[View] [Edit] [Delete]
12	grant-1532451641509	--	Enabled	Enabled	--	atlas	[View] [Edit] [Delete]

- Complete the fields of the **Create Policy** page.
- Click **Add Validity Period**.
- In the **Policy Validity Period** dialog box, specify the **Start Time**, **End Time**, and **Time Zone**.

The screenshot shows the 'Create Policy' interface in the Ranger console. The 'Policy Details' section includes fields for Policy Type (Access), Policy Name (Temp Employees Override), Policy Label (Policy Label), HBase Table (sales), HBase Column-family, HBase Column, and Description. There are toggle buttons for 'enabled', 'override', 'include', and 'exclude'. The 'override' button is highlighted with a red box. A 'Policy Validity Period' dialog box is open, showing a table with columns for Start Time, End Time, and Time zone. The 'Add' button at the bottom of the 'Allow Conditions' section is highlighted with a green box.

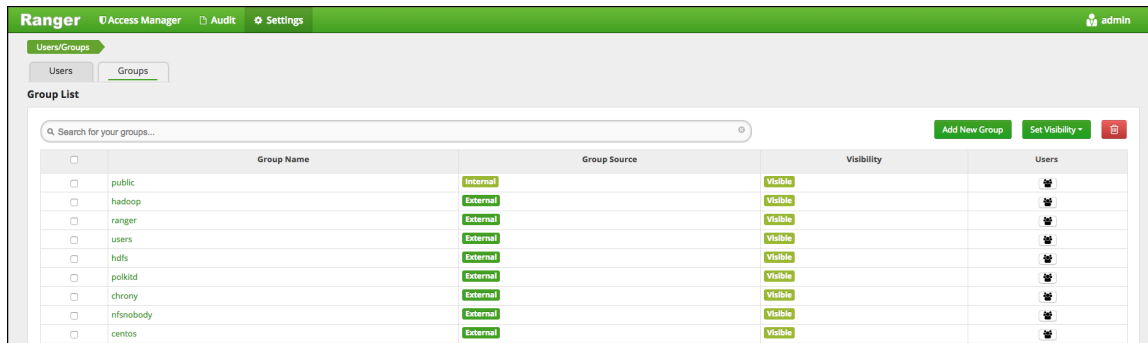
5. If you want this policy to take precedence over all other policies during its validity period, click **Override**. A decision from this policy stops further evaluation of policies.
6. Click **Add**.

Administering Ranger Users/Groups and Permissions

To view the list of users and groups who can access the Ranger portal or its services, select Settings > Users/Groups in the top menu.

The Users/Groups page lists:

- Internal users who can log in to the Ranger portal; created by the Ranger console Service Manager.
- External users who can access services controlled by the Ranger portal; created at other systems like Active Directory, LDAP or UNIX, and synched with those systems.
- Admins who are the only users with permission to create users and create services, run reports, and perform other administrative tasks. Admins can also create child policies based on the original policy (base policy).
- On the Groups page, you can click the people icon under **Users** and view the members of that group.



The screenshot shows the Ranger interface with the 'Users/Groups' page selected. The 'Groups' tab is active. A search bar is present at the top of the table. The table lists various groups with their names, sources, visibility, and associated users.

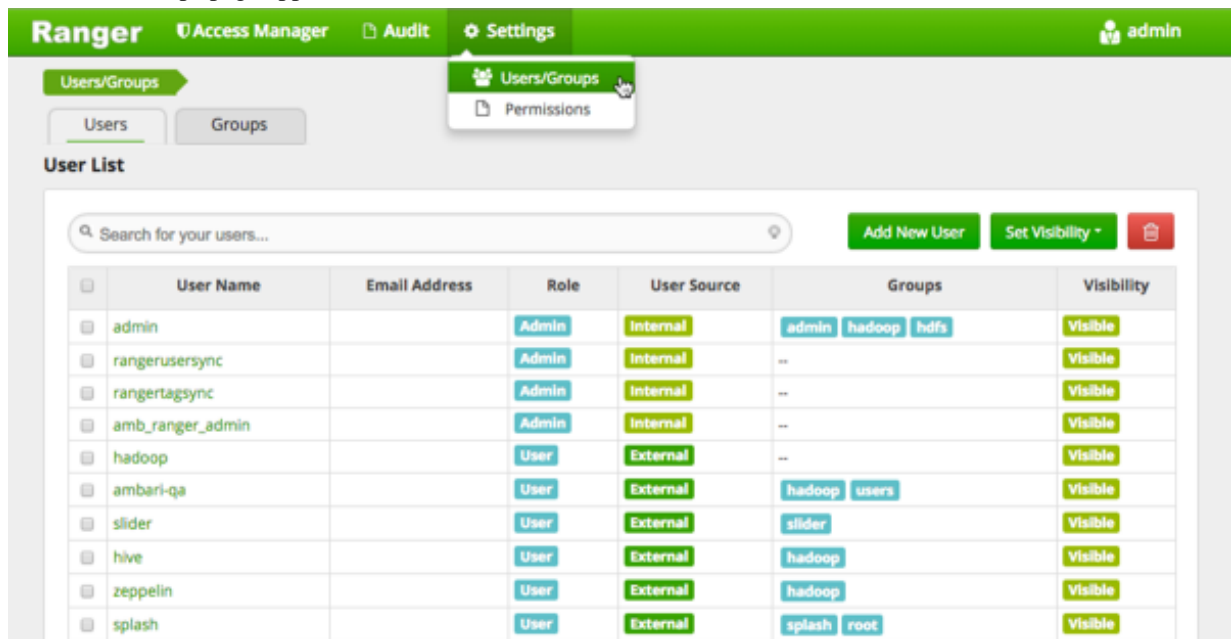
	Group Name	Group Source	Visibility	Users
<input type="checkbox"/>	public	Internal	Visible	
<input type="checkbox"/>	hadoop	External	Visible	
<input type="checkbox"/>	ranger	External	Visible	
<input type="checkbox"/>	users	External	Visible	
<input type="checkbox"/>	hdfs	External	Visible	
<input type="checkbox"/>	polkitd	External	Visible	
<input type="checkbox"/>	chrony	External	Visible	
<input type="checkbox"/>	nfsnobody	External	Visible	
<input type="checkbox"/>	centos	External	Visible	

Add a User

How to add a new user to the user list in Ranger.

Procedure

1. Select **Settings > Users/Groups**.
The Users/Groups page appears.



The screenshot shows the Ranger interface with the 'Settings' menu open and 'Users/Groups' selected. The 'Users' tab is active. A search bar is present at the top of the table. The table lists various users with their names, email addresses, roles, user sources, groups, and visibility.

	User Name	Email Address	Role	User Source	Groups	Visibility
<input type="checkbox"/>	admin		Admin	Internal	admin hadoop hdfs	Visible
<input type="checkbox"/>	rangerusersync		Admin	Internal	--	Visible
<input type="checkbox"/>	rangertagsync		Admin	Internal	--	Visible
<input type="checkbox"/>	amb_ranger_admin		Admin	Internal	--	Visible
<input type="checkbox"/>	hadoop		User	External	--	Visible
<input type="checkbox"/>	ambari-qa		User	External	hadoop users	Visible
<input type="checkbox"/>	slider		User	External	slider	Visible
<input type="checkbox"/>	hive		User	External	hadoop	Visible
<input type="checkbox"/>	zeppelin		User	External	hadoop	Visible
<input type="checkbox"/>	splash		User	External	splash root	Visible

2. Click **Add New User**.

The User Detail page appears.

The screenshot shows the Ranger web interface. The top navigation bar is green and contains the 'Ranger' logo, 'Access Manager', 'Audit', and 'Settings' links, along with a user profile icon for 'admin'. Below this, there are breadcrumb links for 'Users/Groups' and 'User Create'. The main content area is titled 'User Detail' and contains a form with the following fields:

- User Name *
- New Password *
- Password Confirm *
- First Name *
- Last Name
- Email Address
- Select Role * (Dropdown menu showing 'Admin')
- Group (Text: 'Please select', with a '+' button)

At the bottom of the form are 'Save' and 'Cancel' buttons.

3. Add the required user details, then click **Save**.
The user is immediately added to the list.

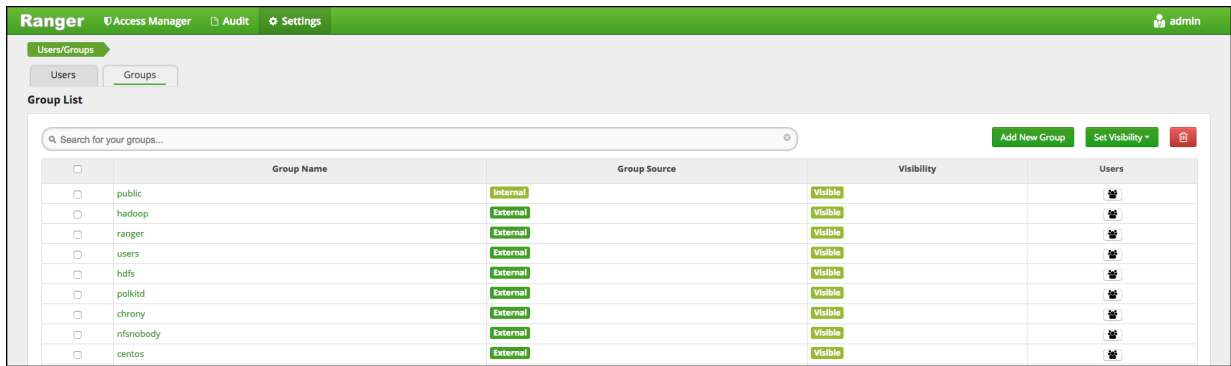
Edit a User

How to edit a user in Ranger.

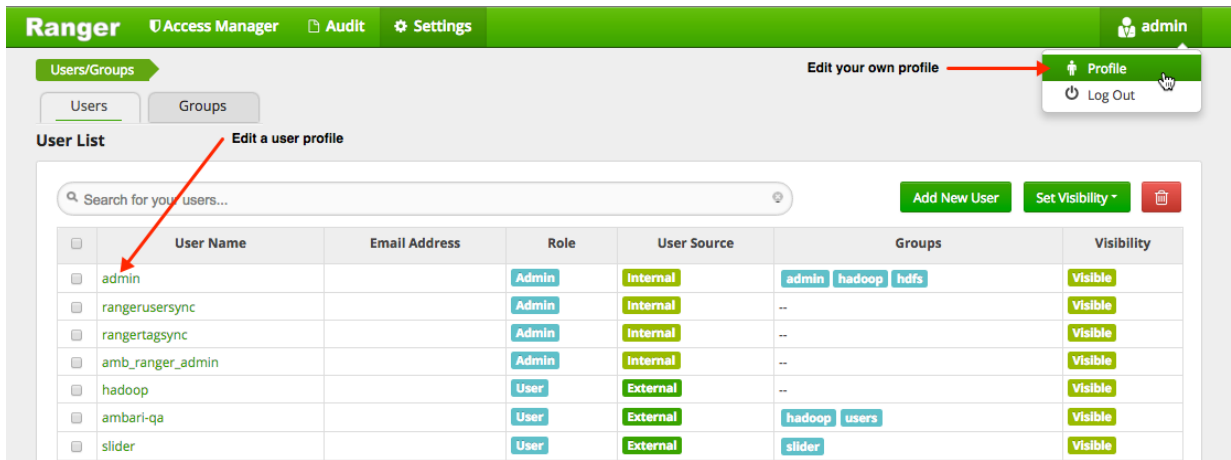
Procedure

1. Select **Settings > Users/Groups**.

The Users/Groups page opens to the Users tab.



2. Select a user profile to edit.



Note:

You can only fully edit internal users. For external users, you can only edit the user role.

Users/Groups > User Edit

User Detail

Basic Info Change Password


User Name *

First Name *

Last Name

Email Address

Select Role *

Group Marketing UX 

Users/Groups > User Edit

User Detail

User Name *

First Name

Last Name

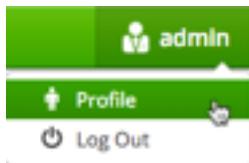
Email Address

Select Role *

Group group01

The User Detail page appears.

3. To edit your own user profile, click **Username > Profile**.



The User Profile page appears.

A screenshot of the 'User Profile' page. The page has a header with 'User Profile' and a sub-header with 'User Profile'. Below the sub-header are two tabs: 'Basic Info' (selected) and 'Change Password'. The 'Basic Info' tab contains four form fields: 'First Name *' with the value 'Admin', 'Last Name *' with the value 'Last Name', 'Email Address' with the value 'Email Address', and 'Select Role *' with a dropdown menu showing 'Admin'. At the bottom of the form are two buttons: 'Save' (green) and 'Cancel' (black).

4. Edit the appropriate details, then click **Save**.

Delete a User

How to permanently delete a user in Ranger.

Before you begin

Only users with role "admin" may delete a user.

Procedure

1. Select **Settings > Users/Groups**.
The Users/Groups page appears.

The screenshot shows the Ranger web interface. At the top, there is a navigation bar with 'Ranger', 'Access Manager', 'Audit', and 'Settings'. Below this, there are tabs for 'Users/Groups', 'Users', and 'Groups'. A dropdown menu is open under 'Users/Groups', showing 'Users/Groups' and 'Permissions'. The main content area is titled 'User List' and contains a search bar, 'Add New User', and 'Set Visibility' buttons. Below these is a table of users:

<input type="checkbox"/>	User Name	Email Address	Role	User Source	Groups	Visibility
<input type="checkbox"/>	admin		Admin	Internal	admin hadoop hdfs	Visible
<input type="checkbox"/>	rangerusersync		Admin	Internal	--	Visible
<input type="checkbox"/>	rangertagsync		Admin	Internal	--	Visible
<input type="checkbox"/>	amb_ranger_admin		Admin	Internal	--	Visible
<input type="checkbox"/>	hadoop		User	External	--	Visible
<input type="checkbox"/>	ambari-qa		User	External	hadoop users	Visible
<input type="checkbox"/>	slider		User	External	slider	Visible
<input type="checkbox"/>	hive		User	External	hadoop	Visible
<input type="checkbox"/>	zeppelin		User	External	hadoop	Visible
<input type="checkbox"/>	splash		User	External	splash root	Visible

- Select the check box of the user you want to delete and click the Delete icon

The screenshot shows the same 'User List' interface. A red box highlights the delete icon (a trash can) in the top right corner of the user list area. A red arrow points from this icon towards the 'splash' user row in the table below.

The screenshot shows the 'User List' interface with a confirmation dialog box overlaid. The dialog asks 'Are you sure you want to delete user 'splash?' and has 'Cancel' and 'OK' buttons. A red box highlights the 'OK' button. A red arrow points from the delete icon in the previous screenshot to the 'OK' button. Below the dialog, a green success message is displayed: 'Success User deleted successfully!'. A red box highlights the 'OK' button in the success message. A red box also highlights the 'splash' user row in the table, with a red arrow pointing to the 'OK' button in the success message.

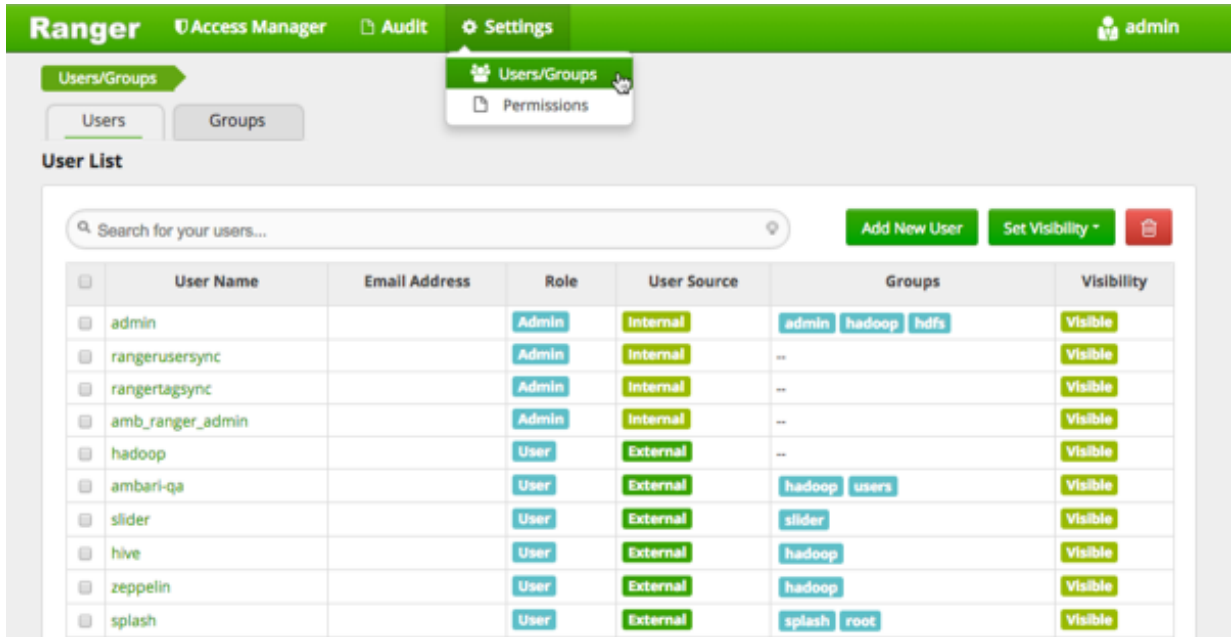
- You are prompted to confirm the user deletion; select OK.
You receive confirmation that the operation has succeeded.

Add a Group

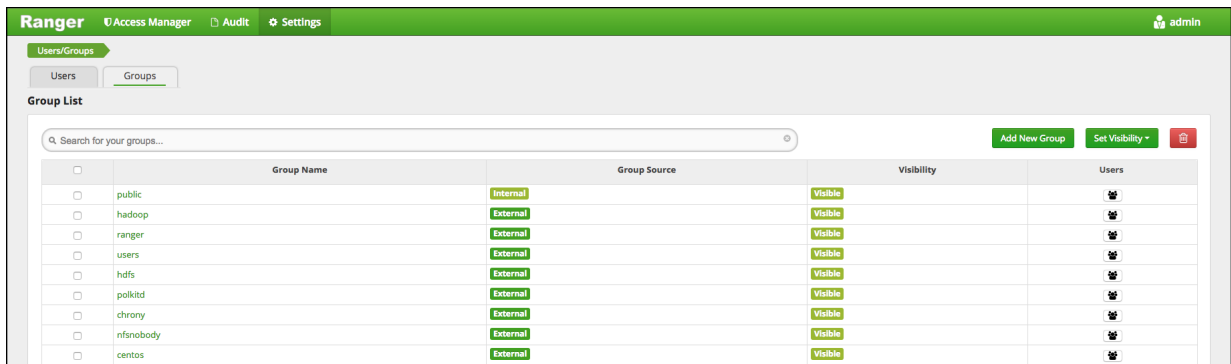
How to add a group in Ranger.

Procedure

1. Select **Settings > Users/Groups**.
The Users/Groups page opens to the Users tab.



2. Click the **Groups** tab.
The Groups page appears.



3. Click **Add New Group**.

The Group Create page

The screenshot shows the Ranger interface with the 'Users/Groups' section active. A 'Group Create' modal is open, displaying a form with 'Group Name *' and 'Description' fields, and 'Save' and 'Cancel' buttons. In the background, a 'User List' table is visible, and a red arrow points to the 'Add New User' button.

User Name	Email Address	Role	User Source	Groups	Visibility
admin		Admin	Internal	admin	Visible
rangerusersync		Admin	Internal	--	Visible
rangertagsync		Admin	Internal	--	Visible
amb_ranger_admin		Admin	Internal	--	Visible
hadoop		User	External	--	Visible
ambari-qa		User	External	hadoop users	Visible
slider		User	External	slider	Visible
hive		User	External	hadoop	Visible
zeppelin		User	External	hadoop	Visible
splash		User	External	splash root	Visible
HDP		User	External	root	Visible
hdfs		User	External	hadoop	Visible
ranger		User	External	hadoop	Visible

appears.

4. Enter a unique name for the group, and an optional description, then click **Save**.

Edit a Group

How to edit a group in Ranger.

Procedure

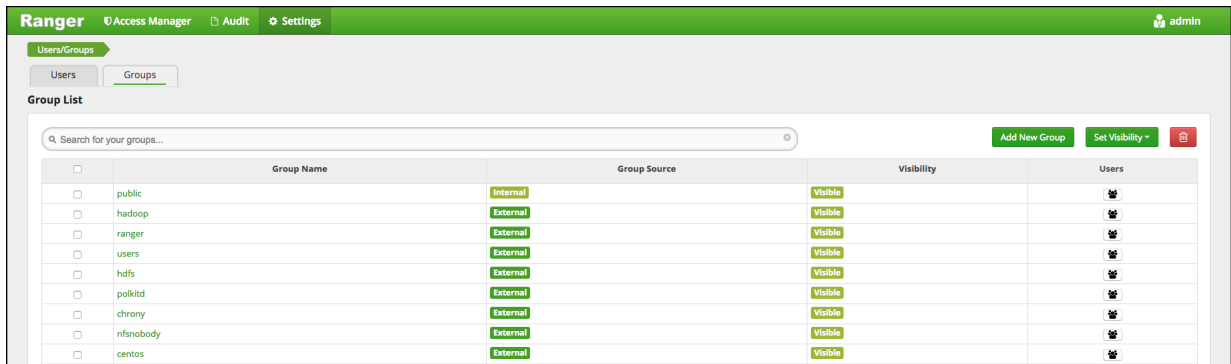
1. Select **Settings > Users/Groups**.
The Users/Groups page opens to the Users tab.

The screenshot shows the Ranger interface with the 'Settings' menu open. The 'Users/Groups' option is highlighted, and the 'Groups' tab is selected in the background. The 'User List' table is visible below.

User Name	Email Address	Role	User Source	Groups	Visibility
admin		Admin	Internal	admin hadoop hdfs	Visible
rangerusersync		Admin	Internal	--	Visible
rangertagsync		Admin	Internal	--	Visible
amb_ranger_admin		Admin	Internal	--	Visible
hadoop		User	External	--	Visible
ambari-qa		User	External	hadoop users	Visible
slider		User	External	slider	Visible
hive		User	External	hadoop	Visible
zeppelin		User	External	hadoop	Visible
splash		User	External	splash root	Visible

2. Click the **Groups** tab.

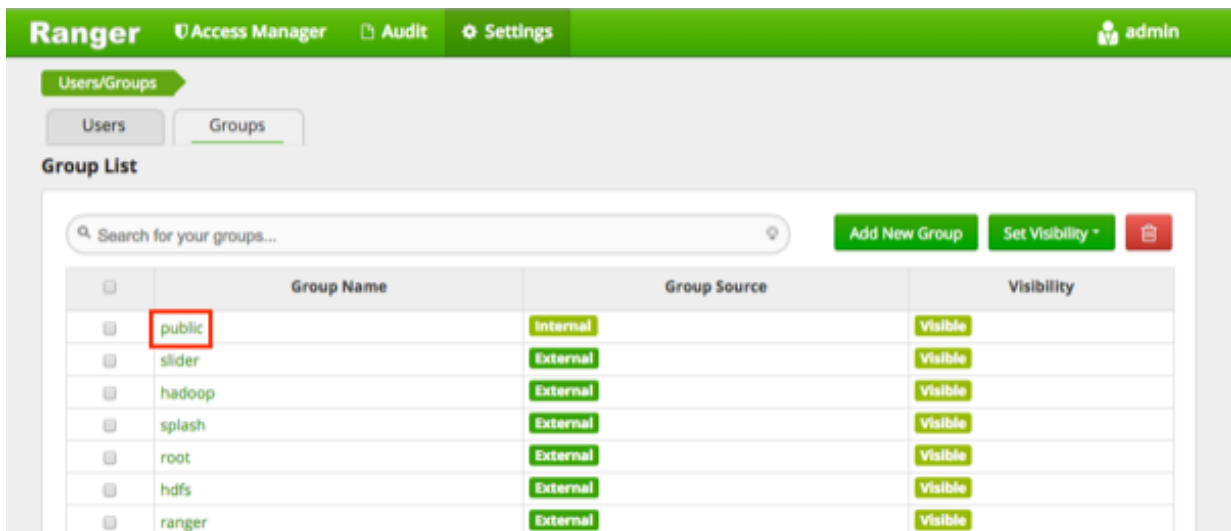
The Groups page appears.



The screenshot shows the Ranger web interface. The top navigation bar includes 'Ranger', 'Access Manager', 'Audit', and 'Settings'. The user 'admin' is logged in. The 'Users/Groups' section is active, with 'Groups' selected. The 'Group List' table contains the following data:

<input type="checkbox"/>	Group Name	Group Source	Visibility	Users
<input type="checkbox"/>	public	Internal	Visible	
<input type="checkbox"/>	hadoop	External	Visible	
<input type="checkbox"/>	ranger	External	Visible	
<input type="checkbox"/>	users	External	Visible	
<input type="checkbox"/>	hdfs	External	Visible	
<input type="checkbox"/>	polkitd	External	Visible	
<input type="checkbox"/>	chrony	External	Visible	
<input type="checkbox"/>	nfsnobody	External	Visible	
<input type="checkbox"/>	centos	External	Visible	

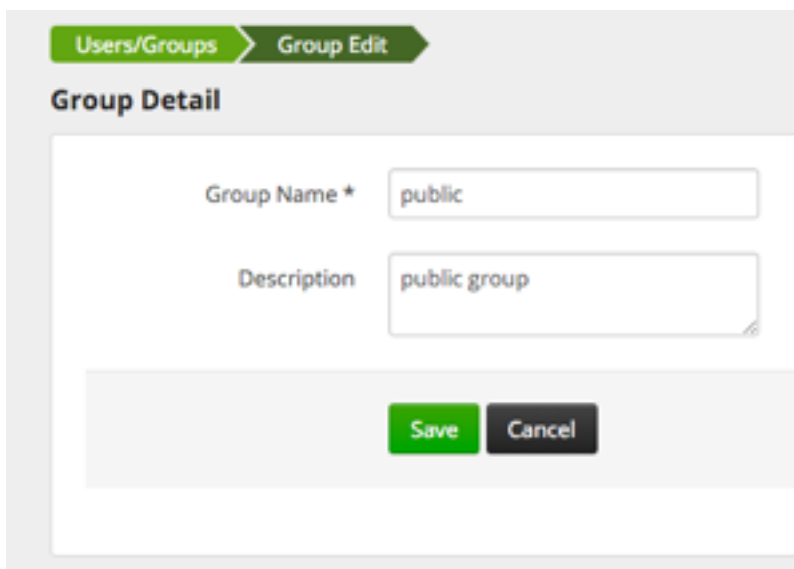
3. Select a group name to edit.



The screenshot shows the Ranger web interface. The top navigation bar includes 'Ranger', 'Access Manager', 'Audit', and 'Settings'. The user 'admin' is logged in. The 'Users/Groups' section is active, with 'Groups' selected. The 'Group List' table contains the following data:

<input type="checkbox"/>	Group Name	Group Source	Visibility	Users
<input type="checkbox"/>	public	Internal	Visible	
<input type="checkbox"/>	slider	External	Visible	
<input type="checkbox"/>	hadoop	External	Visible	
<input type="checkbox"/>	splash	External	Visible	
<input type="checkbox"/>	root	External	Visible	
<input type="checkbox"/>	hdfs	External	Visible	
<input type="checkbox"/>	ranger	External	Visible	

4. The Group Edit page appears.



The screenshot shows the Ranger web interface. The top navigation bar includes 'Ranger', 'Access Manager', 'Audit', and 'Settings'. The user 'admin' is logged in. The 'Users/Groups' section is active, with 'Group Edit' selected. The 'Group Detail' form contains the following data:

Group Name *

Description

5. Edit the group details, then click **Save**.

Delete a Group

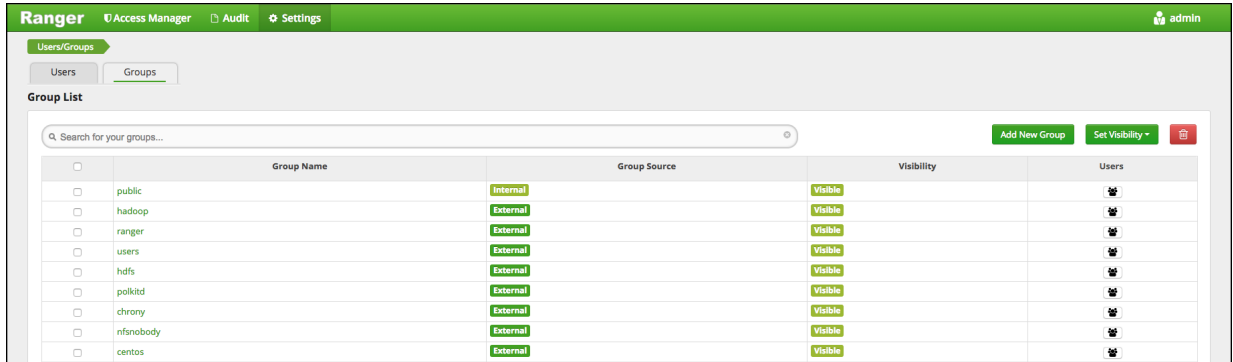
How to permanently delete a group in Ranger.

Before you begin

Only users with role "admin" may delete a group.

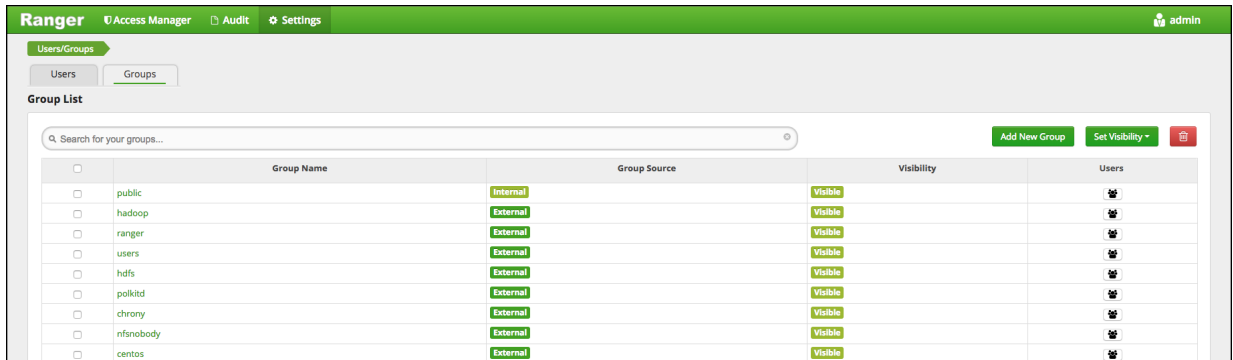
Procedure

1. Select **Settings > Users/Groups**.
The Users/Groups page appears.



2. Click the **Groups** tab.

The Groups page appears.



3. Select the check box of the group you want to delete and click the Delete icon



at the right of the **Group List** menu bar.

The screenshot shows the Ranger Admin console interface. At the top, there are navigation tabs for 'Access Manager', 'Audit', and 'Settings'. The 'Users/Groups' section is active, with 'Groups' selected. A search bar is present above a table of groups. The 'splash' group is selected, and a confirmation dialog is shown. A success message is displayed below the table.

	Group Name	Group Source	Visibility
<input type="checkbox"/>	public	Internal	Visible
<input type="checkbox"/>	slider		
<input type="checkbox"/>	hadoop		
<input checked="" type="checkbox"/>	splash		
<input type="checkbox"/>	root		
<input type="checkbox"/>	hdfs	External	Visible
<input type="checkbox"/>	ranger	External	Visible
<input type="checkbox"/>	users		Visible
<input type="checkbox"/>	packer		Visible
<input type="checkbox"/>	hue	External	Visible
<input type="checkbox"/>	nfsnobody	External	Visible
<input type="checkbox"/>	artmin	External	Visible

- You are prompted to confirm the group deletion; select OK.
You receive confirmation that the operation has succeeded.

What to do next

Users in a deleted group will be reassigned to no group. You can edit the user to reassign it to groups.

Related Information

[Edit a User](#)

Add/Edit Permissions

How to add or edit a user or group in Ranger.

Procedure

- Select **Settings > Permissions**.
The Users/Groups page opens to the Permissions page.

The screenshot shows the Ranger Admin console interface for the 'Permissions' page. A search bar is present above a table of permissions. The table lists various permissions and their associated groups and users.

Permissions	Groups	Users	Action
Resource Based Policies		admin, rangersonerptic, keyadmin, hue	Here...
Users/Groups		admin, rangersonerptic, antb, ranger, admin	Here...
Reports		admin, rangersonerptic, keyadmin, hue	Here...
Audit		admin, rangersonerptic, antb, ranger, admin	Here...
Key Manager		keyadmin	Here...

- Click the Edit icon



next to the permission you would like to edit.
The Edit Permission page appears.

- Edit the permission settings, then click **Save**.

You can select multiple users and groups from the drop-down menus.

Administering Ranger Reports

You can use the Reports page to help manage policies more efficiently as the number of policies increases. The page lists all HDFS, HBase, Hive, YARN, Knox, Storm, Solr, Kafka, Atlas, and tag-based policies.

Policy ID	Policy Name	Policy Labels	Resources	Policy Type	Status	Allow Conditions	Allow Exclude	Deny Conditions	Deny Exclude
1	all-path	--	path/*	Access	Enabled	--	+	-	+
	Deny Conditions	Groups	Users					Accesses	
			No records found						
	Allow Conditions	Groups	Users					Accesses	
			hdfs			read	write	execute	
2	kms-audit-path	--	path/ranger/audit/kms	Access	Enabled	+	+	+	+

View Ranger Reports

How to view reports on one or more policies in Ranger.

To view reports on one or more policies, select **Access Manager > Reports**.

More policy information is available when you click



below **Allow Conditions**.

Search Ranger Reports

Reference information for searching Ranger reports on one or more policies.

You can search based on:

- Policy Name – The policy name assigned to the policy.
- Policy Type – The policy type assigned to the policy (Access, Masking, or Row Level Filter).
- Policy Label – The label assigned to the policy.
- Component – The component assigned to the policy (HDFS, HBase, Hive, YARN, Knox, Storm, Solr, Kafka, Atlas, and tag).
- Resource – The resource path used when creating the policy.
- Group, Username – The group and the users to which the policy is assigned.

The screenshot shows the Ranger Reports page with the following search criteria:

- Policy Name:
- Policy Type: Access
- Component: Select Component
- Resource:
- Policy Label: Select Policy Label
- Search By: Group

The HDFS table displays the following data:

Policy ID	Policy Name	Policy Labels	Resources	Policy Type	Status	Allow Conditions	Allow Exclude	Deny Conditions	Deny Exclude
1	all - path	--	path/*	Access	Enabled	-	+	-	+
Deny Conditions									
Groups									
Users									
No records found									
Accesses									
Allow Conditions									
Groups									
Users									
Accesses									
hdfs									
read write execute									
2	kms-audit-path	--	path/ranger/audit/kms	Access	Enabled	+	+	+	+

Export Reports

Reference information for exporting Ranger reports on one or more policies.

You can export a list of reports in three file formats:

- CSV file
- Excel file
- JSON

The screenshot shows the Ranger Reports page with the following search criteria:

- Policy Name:
- Policy Type: Select policy type
- Component: Select Component
- Resource:
- Search By: Group

The HDFS table displays the following data:

Policy ID	Policy Name	Resources	Policy Type	Status	Allow Conditions
2	all - path	path/*	Access	Enabled	+
16	all - path	path/*	Access	Enabled	+

The **Export** button is highlighted with a red box.

For more information on exporting policies from the reports page, see links below.

Related Information

[Export Tag-Based Policies](#)

[Export Resource-Based Policies for a Specific Service](#)

[Export All Resource-Based Policies for All Services](#)

Editing Ranger Policies from the Reports Page

Reference information on how to edit Ranger policies from the Reports page.

You can edit policies from the Reports page by selecting the Policy ID.

The screenshot displays the Apache Ranger web interface. The top navigation bar includes 'Ranger', 'Access Manager', 'Audit', 'Settings', and a user profile 'admin'. The main content area is titled 'Reports' and contains a 'User Access Report' section. Below this, there is a 'Reports' section with a search criteria input field. A sidebar on the left shows a list of policies under 'HDFS' and 'HBASE'. A red arrow points to the 'Policy ID' field in the 'HDFS' section, which has the value '11'. The main content area shows the 'Edit Policy' form for Policy ID 1. The form includes the following fields:

- Policy Type: Access
- Policy ID: 1
- Policy Name: all - path
- Resource Path: *
- Recursive: ON
- Audit Logging: YES
- Description: Policy for all - path

Below the form is the 'Allow Conditions' section, which contains a table with the following columns: Select Group, Select User, Permissions, and Delegate Admin. The table shows a group 'hadoop' with users 'hadoop' and 'ambari-qa', and permissions 'Read', 'Write', and 'Execute'. The 'Delegate Admin' column has a blue checkmark and a red 'X' button. At the bottom of the form are 'Save', 'Cancel', and 'Delete' buttons.

Adding a New Component to Apache Ranger

How to add a new component to Apache Ranger.

Apache Ranger has three main components:

- Admin Tool -- Provides web interface & REST API for managing security policies.

- Custom Authorization Module for components -- Provides custom authorization within the (Hadoop) component to enforce the policies defined in Admin Tool.
- UserGroup synchronizer -- Enables the user/group information in Apache Ranger to synchronize with the Enterprise user/group information stored in LDAP or Active Directory.

In order to support new component authorization using Apache Ranger, the component details need to be added to Apache Ranger as follows:

- Add component details to the Admin Tool.
- Develop a custom authorization module for the new component.

Adding Component Details to the Admin Tool

The Apache Ranger Admin tool supports policy management via both a web interface (UI) and support for a (public) REST API. In order to support a new component in both the UI and the Server, the Admin Tool must be modified.

Required UI changes to support the new component:

1. Add a new component template to the Access Manager page (console home page):

Show new component on the Access Manager page i.e home page[#!/policymanager]. Apache Ranger needs to add table template to Service Manager page and make changes in corresponding JS files. Ranger also needs to create a new service type enum to distinguish the component for which the service/policy is created/updated.

For example: Add a table template to PolicyManagerLayout_tmpl.html file to view the new component on the Access Manager page and make changes in the PolicyManagerLayout.js file related to the new component, such as passing Knox service collection data to the PolicyManagerLayout_tmpl template. Also create a new service type enum (for example, ASSET_KNOX) in the XAEnums.js file.

2. Add new configuration information to the Service Form:

Add new configuration fields to Service Form [AssetForm.js] as per new component configuration information. This will cause the display of new configuration fields in the corresponding service Create/Update page. Please note that the AssetForm.js is a common file for every component to create/update the service.

For example: Add new field(configuration) information to AssetForm.js and AssetForm_tmpl.js.

3. Add a new Policy Listing page:

Add a new policy listing page for the new component in the View Policy list. For example: Create a new KnoxTableLayout.js file and add JS-related changes as per the old component[HiveTableLayout.js] to the View Policy listing. Also create a template page, KnoxTableLayout_tmpl.html.

4. Add a new Policy Create/Update page:

Add a Policy Create/Update page for the new component. Also add a policy form JS file and its template to handle all policy form-related actions for the new component. For example: Create a new KnoxPolicyCreate.js file for Create/Update Knox Policy. Create a KnoxPolicyForm.js file to add Knox policy fields information. Also create a corresponding KnoxPolicyForm_tmpl.html template.

5. Other file changes, as needed:

Make changes in existing common files as per our new component like Router.js, Controller.js, XAUtils.js, FormInputList.js, UserPermissionList.js, XAEnums.js, etc.

Required server changes for the new component:

Let's assume that Apache Ranger has three components supported in their portal and we want to introduce one new component, Knox:

1. Create New Service Type

If Apache Ranger is introducing new component i.e Knox, then they will add one new service type for Knox. i.e serviceType = "Knox". On the basis of service type, while creating/updating service/policy, Apache Ranger will distinguish for which component this service/policy is created/updated.

2. Add new required parameters in existing objects and populate objects

For Policy Creation/Update of any component (i.e HDFS, Hive, Hbase), Apache Ranger uses only one common object, `VXPolicy`. The same goes for the Service Creation/Update of any component: Apache Ranger uses only one common object `VXService`. As Apache Ranger has three components, it will have all the required parameters of all of those three components in `VXPolicy/VXService`. But for Knox, Apache Ranger requires some different parameters which are not there in previous components. Thus, it will add only required parameters into `VXPolicy/VXService` object. When a user sends a request to the Knox create/update policy, they will only send the parameters that are required for Knox to create/update the VXPolicy object.

After adding new parameters into VXPolixy/VXService, Apache Ranger populates the newly-added parameters in corresponding services, so that it can map those objects with Entity Object.

3. Add newly-added fields (into database table) related parameters into entity object and populate them

As Apache Ranger is using JPA-EclipseLink for database mapping into java, it is necessary to update the Entity object. For example, if for Knox policy Apache Ranger has added two new fields (`topology` and `service`) into db table `x_resource`, it will also have to update the entity object of table (i.e `XXResource`), since it is altering table structure.

After updating the entity object Apache Ranger will populate newly-added parameters in corresponding services (i.e XResourceService), so that it can communicate with the client using the updated entity object.

4. Change middleware code business logic

After adding and populating newly required parameters for new component, Apache Ranger will have to write business logic into file `AssetMgr`, where it may also need to do some minor changes. For example, if it wants to create a default policy while creating the Service, then on the basis of serviceType, Apache Ranger will create one default policy for the given service. Everything else will work fine, as it is common for all components.

Required database changes for the new component:

For service and policy management, Apache Ranger includes the following tables:

- x_asset (for service)
- x_resource (for service)

As written above, if Apache Ranger is introducing new component then it is not required to create individual table in database for each component. Apache Ranger has common tables for all components.

If Apache Ranger has three components and wants to introduce a fourth one, then it will add required fields into these two tables and will map accordingly with java object. For example, for Knox, Apache Ranger will add two fields (`topology`, `service`) into `x_resource`. After this, it will be able to perform CRUD operation of policy and service for our new component, and also for previous components.

Configuring Advanced Authorization Settings

How to customize the Ranger Advanced Settings when configuring authentication.

Ranger Admin Ranger User Info Ranger Plugin Ranger Audit Ranger Tagsync **Advanced**

▼ Admin Settings

Ranger Admin host dw-weekly.field.hortonworks.com

Ranger Admin username for Ambari 🔒 🟢 🔄

Ranger Admin user's password for Ambari 🔒

Location of Sql Connector Jar 🔄

▼ Ranger Settings

External URL 🔄

Authentication method

LDAP

ACTIVE_DIRECTORY

UNIX

NONE

⚠️

HTTP enabled 🔒 🔄

▼ Unix Authentication Settings

Allow remote Login 🔒 🔄

ranger.unixauth.enable 🔒 🔄

Developing a Custom Authorization Module

In the Hadoop ecosystem, each component (i.e., Hive, HBase) has its own authorization implementation and ability to plug in a custom authorization module. To implement the centralized authorization and audit feature for a component, the component should support a customizable (or pluggable) authorization module.

The custom component Authorization Plugin should do the following:

- Provide authorization based on Policies defined in Policy Admin Tool
- Provide audit information based on the authorization decisions

Implementing Custom Component Authorization

To implement the custom component authorization plugin, the Ranger common agent framework provides the following functionalities:

- Ability to read all policies from Service Manager for a given service-id
- Ability to log audit information

When the custom authorization module is initialized, the module should do the following:

1. Initiate a REST API call to the “Policy Admin Tool” to retrieve all policies associated with the specific component.
2. Once the policies are available, it should:
 - be built into a custom data structure for enabling the authorization module.

- kick off the policy updater thread to refresh policies from “Policy Admin Tool” at a regular interval.

When the custom authorization module is called to perform authorization of a component action (such as READ action) on a specific component resource (such as /app folder), the authorization module will:

- Identify authorization decision - For each policy:policyList:
 - If (resource in policy <match> auth-requested-resource)
 - If (action-in-policy <match>action-requested)
 - If (current-user or current-user-groups or public-group <allowed> for the policy), Return access-allowed
- Identify auditing needs - For each policy:policyList
 - If (resource in policy <match> auth-requested-resource), return policy.isAuditEnabled()

Special Requirements for High Availability Environments

In a High Availability (HA) environment, the primary and secondary NameNodes must be configured as described in the HDP System Administration Guide.

To enable Ranger in the HDFS HA environment, the HDFS plugin must be set up in each NameNode, and then pointed to the same HDFS service set up in the Security Manager. Any policies created within that HDFS service are automatically synchronized to the primary and secondary NameNodes through the installed Apache Ranger plugin. That way, if the primary NameNode fails, the secondary NameNode takes over and the Ranger plugin at that NameNode begins to enforce the same policies for access control.

When creating the service, you must include the `fs.default.name` property, and it must be set to the full host name of the primary NameNode. If the primary NameNode fails during policy creation, you can then temporarily use the `fs.default.name` of the secondary NameNode in the service details to enable directory lookup for policy creation.

If, while the primary NameNode is down, you wish to create new policies, there is a slight difference in user experience when specifying the resource path. If everything is normal, this is a drop-down menu with selectable paths; however, if your cluster is running from the failover node, there will be no drop-down menu, and you will need to manually enter the path.

Primary NameNode failure does not affect the actual policy enforcement. In this setup for HA, access control is enforced during primary NameNode failure by the Ranger plugs at the secondary NameNodes.

For **Test Connection** to be successful for HBase and HDFS in a Ranger HA environment, complete the following: In `/etc/ranger/admin`, create a symbolic link between `hbase-site.xml` and `hdfs-site.xml`:

```
cd /etc/ranger/admin
ln -s /etc/hadoop/conf/hdfs-site.xml hdfs-site.xml
ln -s /etc/hbase/conf/hbase-site.xml hbase-site.xml
```

Configure Advanced Usersync Settings

To access Usersync settings, select the Advanced tab on the Customize Service page. Usersync pulls in users from UNIX, LDAP, or AD and populates Ranger's local user tables with these users.

About this task

Configure advanced User Sync settings for the following:

- Unix
- (Required) LDAP/AD
- (Optional) LDAP/AD
- Automatically Assign ADMIN KEYADMIN Role for External Users

Procedure

- Unix: If you are using UNIX authentication, the default values for the Advanced ranger-ugsync-site properties are the settings for UNIX authentication:

▼ **Advanced ranger-ugsync-site**

ranger.usersync.ldap. bindkeystore	<input type="text"/>	🔒	🟢	
ranger.usersync.ldap. ldapbindpassword	<input type="password" value="Type password"/> <input type="password" value="Retype Password"/>	🔒		
ranger.usersync.group. memberattributename	<input type="text"/>	🔒	🟢	🔄
ranger.usersync.group. nameattribute	<input type="text"/>	🔒	🟢	🔄
ranger.usersync.group. objectclass	<input type="text"/>	🔒	🟢	🔄
ranger.usersync.group. searchbase	<input type="text"/>	🔒	🟢	🔄
ranger.usersync.group. searchenabled	<input type="text" value="false"/>	🔒	🟢	🔄
ranger.usersync.group. searchfilter	<input type="text"/>	🔒	🟢	🔄
ranger.usersync.group. searchscope	<input type="text"/>	🔒	🟢	🔄
ranger.usersync.group. usermapsyncenabled	<input type="text" value="false"/>	🔒	🟢	🔄
ranger.usersync.ldap. searchBase	<input type="text" value="dc=hadoop,dc=apache,dc=org"/>	🔒	🟢	🔄
ranger.usersync.source. impl.class	<input type="text" value="org.apache.ranger.unixusersync.process.UnixUserGroupBuilder"/>	🔒	🟢	🔄
ranger.usersync. credstore.filename	<input type="text" value="/usr/hdp/current/ranger-usersync/conf/ugsync.jceks"/>	🔒	🟢	🔄
ranger.usersync.enabled	<input type="text" value="true"/>	🔒	🟢	🔄
ranger.usersync. filesource.file	<input type="text" value="/tmp/usergroup.txt"/>	🔒	🟢	🔄
ranger.usersync. filesource.text.delimiter	<input type="text" value="."/>	🔒	🟢	🔄
ranger.usersync. keystore.file	<input type="text" value="/usr/hdp/current/ranger-usersync/conf/unixauthservice.jks"/>	🔒	🟢	🔄

- (Required) LDAP/AD
 - a) LDAP Advanced ranger-ugsync-site Settings

Table 52: LDAP Advanced ranger-ugsync-site Settings

Property Name	LDAP Value
ranger.usersync.ldap.bindkeystore	Set this to the same value as the ranger.usersync.credstore.filename property, i.e, the default value is /usr/hdp/current/ranger-usersync/conf/ugsync.jceks
ranger.usersync.ldap.bindalias	ranger.usersync.ldap.bindalias
ranger.usersync.source.impl.class	ldap

b) AD Advanced ranger-ugsync-site Settings

Table 53: AD Advanced ranger-ugsync-site Settings

Property Name	LDAP Value
ranger.usersync.source.impl.class	ldap

- (Optional) LDAP/AD. If you are using LDAP or Active Directory authentication, you may need to update the following properties, depending upon your specific deployment characteristics.

a) Advanced ranger-ugsync-site Settings for LDAP and AD

Table 54: Advanced ranger-ugsync-site Settings for LDAP and AD

Property Name	LDAP ranger-ugsync-site Value	AD ranger-ugsync-site Value
ranger.usersync.ldap.url	ldap://127.0.0.1:389	ldap://ad-conrowoller-hostname:389
ranger.usersync.ldap.binddn	cn=ldadmin,ou=users,dc=example,dc=com	cn=adadmin,cn=Users,dc=example,dc=com
ranger.usersync.ldap.ldapbindpassword	secret	secret
ranger.usersync.ldap.searchBase	dc=example,dc=com	dc=example,dc=com
ranger.usersync.source.impl.class	org.apache.ranger.ladpusersync.process.LdapUserGroupBuilder	
ranger.usersync.ldap.user.searchbase	ou=users, dc=example, dc=com	dc=example,dc=com
ranger.usersync.ldap.user.searchscope	sub	sub
ranger.usersync.ldap.user.objectclass	person	person
ranger.usersync.ldap.user.searchfilter	Set to single empty space if no value. Do not leave it as "empty"	(objectcategory=person)
ranger.usersync.ldap.user.nameattribute	uid or cn	sAMAccountName
ranger.usersync.ldap.user.groupnameattribute	memberof,ismemberof	memberof,ismemberof
ranger.usersync.ldap.username.caseconversion	none	none
ranger.usersync.ldap.groupname.caseconversion	none	none
ranger.usersync.group.searchenabled *	false	false
ranger.usersync.group.usermapsyncenabled *	false	false
ranger.usersync.group.searchbase *	ou=groups, dc=example, dc=com	dc=example,dc=com
ranger.usersync.group.searchscope *	sub	sub
ranger.usersync.group.objectclass *	groupofnames	groupofnames

Property Name	LDAP ranger-ugsync-site Value	AD ranger-ugsync-site Value
ranger.usersync.group.searchfilter *	needed for AD authentication	(member=CN={0}, OU=MyUsers, DC=AD-HDP, DC=COM)
ranger.usersync.group.nameattribute *	cn	cn
ranger.usersync.group.memberattributename *	member	member
ranger.usersync.pagedresultsenabled *	true	true
ranger.usersync.pagedresultssize *	500	500
ranger.usersync.user.searchenabled *	false	false
ranger.usersync.group.search.first.enabled *	false	false

* Only applies when you want to filter out groups.

After you have finished specifying all of the settings on the Customize Services page, click Next at the bottom of the page to continue with the installation.

- Automatically Assign ADMIN KEYADMIN Role for External Users. You can use usersync to mark specific external users, or users in a specific external group, with ADMIN or KEYADMIN role within Ranger. This is useful in cases where internal users are not allowed to login to Ranger.

a) From Ambari>Ranger>Configs>Advanced>Custom ranger-ugsync-site, select **Add Property**.

b) Add the following properties:

- ranger.usersync.role.assignment.list.delimiter = &

The default value is &.

- ranger.usersync.users.groups.assignment.list.delimiter = :

The default value is :.

- ranger.usersync.username.groupname.assignment.list.delimiter = ,

The default value is ,.

- ranger.usersync.group.based.role.assignment.rules =

ROLE_SYS_ADMIN:u:userName1,userName2&ROLE_SYS_ADMIN:g:groupName1,groupName2&ROLE_KEY_AD

c) Click Add.

d) Restart Ranger.

```
ranger.usersync.role.assignment.list.delimiter = &
ranger.usersync.users.groups.assignment.list.delimiter = :
ranger.usersync.username.groupname.assignment.list.delimiter = ,
ranger.usersync.group.based.role.assignment.rules :
&ROLE_SYS_ADMIN:u:ldapuser_12,ldapuser2
```

Related Information

[Set Up Hadoop Group Mapping for LDAP/AD](#)

Configure User Sync LDAP SSL

How to configure LDAP SSL using self-signed certs in the default Ranger User Sync TrustStore.

Procedure

- The default location is /usr/hdp/current/ranger-usersync/conf/mytruststore.jks for the ranger.usersync.truststore.file property.
- Alternatively, copy and edit the self-signed ca certs.

3. Set the `ranger.usersync.truststore.file` property to that new cacert file.

```
cd /usr/hdp/<version>/ranger-usersync
service ranger-usersync stop
service ranger-usersync start
```

Where `cert.pem` has the LDAPS cert.

Set Up Database Users Without Sharing DBA Credentials

If you do not wish to provide system Database Administrator (DBA) account details to the Ambari Ranger installer, you can use the `dba_script.py` Python script to create Ranger DB database users without exposing DBA account information to the Ambari Ranger installer. You can then run the normal Ambari Ranger installation without specifying a DBA user name and password.

Procedure

1. Download the Ranger rpm using the yum install command: `yum install ranger-admin`.
2. You should see one file named `dba_script.py` in the `/usr/hdp/current/ranger-admin` directory.
3. Get the script reviewed internally and verify that your DBA is authorized to run the script.
4. Execute the script by running the following command: `python dba_script.py`.
5. Pass all values required in the argument. These should include db flavor, JDBC jar, db host, db name, db user, and other parameters.

- If you would prefer not to pass runtime arguments via the command prompt, you can update the `/usr/hdp/current/ranger-admin/install.properties` file and then run: `python dba_script.py -q`

When you specify the `-q` option, the script will read all required information from the `install.properties` file.

- You can use the `-d` option to run the script in "dry" mode. Running the script in dry mode causes the script to generate a database script.

```
python dba_script.py -d /tmp/generated-script.sql
```

- Anyone can run the script, but it is recommended that the system DBA run the script in dry mode. In either case, the system DBA should review the generated script, but should only make minor adjustments to the script, for example, change the location of a particular database file. No major changes should be made that substantially alter the script -- otherwise the Ranger install may fail.

The system DBA must then run the generated script.

6. Run the Ranger Ambari install procedure, but set Setup Database and Database User to No in the Ranger Admin section of the Customize Services page.

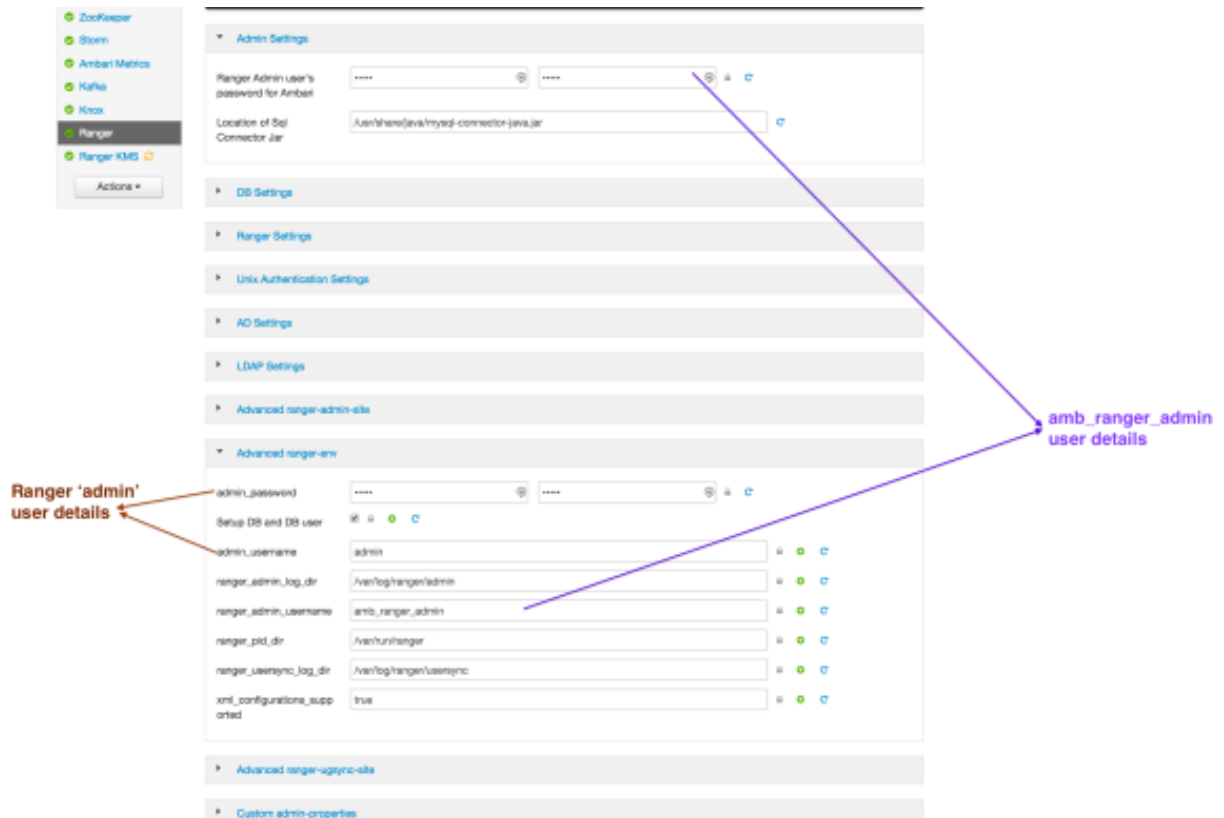
Updating Ranger Admin Passwords

For certain users, if you update the passwords on the Ranger Configs page, you must also update the passwords on the Configs page of each Ambari component that has the Ranger plugin enabled.

Individual Ambari component configurations are not automatically updated -- the service restart will fail if you do not update these passwords on each component.

- Ranger Admin user -- The credentials for this user are set in Configs > Advanced ranger-env in the fields labeled `admin_username` (default value: `admin`) and `admin_password` (default value: `admin`).
- Admin user used by Ambari to create repo/policies -- The user name for this user is set in Configs > Admin Settings in the field labeled Ranger Admin username for Ambari (default value: `amb_ranger_admin`). The password for this user is set in the field labeled Ranger Admin user's password for Ambari. This password is specified during the Ranger installation.

The following image shows the location of these settings on the Ranger Configs page:



Ranger Password Requirements

This topic lists password requirements for Ranger and Ranger KMS.

Ranger user password requirements:

- Minimum of 8 characters
- Must include at least one alphabetical and one numerical character
- Must not include the following unsupported special characters: " ' \ `

Ranger and Ranger KMS DB user password requirements:

- Must not include the following unsupported special characters: " ' \ `

Ranger database instance password requirements:

- Refer to the password requirements for the applicable database type (MySQL, PostgreSQL, Oracle, etc.)