

Ranger Authorization

Date published: 2020-07-28

Date modified: 2026-03-31



Legal Notice

© Cloudera Inc. 2026. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

Using Ranger to Provide Authorization in Cloudera.....	6
Ranger plugin overview.....	6
Ranger Hive Plugin.....	6
Ranger Kafka Plugin.....	8
Ranger-HBase Plugin.....	9
Ranger special entities.....	10
Enabling Ranger HDFS plugin manually on a Cloudera Data Hub.....	11
Ranger Policies Overview.....	13
Ranger tag-based policies.....	14
Tags and policy evaluation.....	15
Ranger access conditions.....	16
Using the Ranger Admin Web UI.....	19
Accessing the Ranger Admin Web UI.....	19
Ranger console navigation.....	21
Policy conditions in Apache Ranger.....	26
Evaluation context in policy conditions.....	26
ctx and _ctx object in policy conditions.....	28
Tag attributes with ctx or _ctx object for policy conditions.....	30
Ranger policy condition macros.....	31
Practical JavaScript usage examples.....	34
Resource-based Services and Policies.....	35
Configuring resource-based services.....	35
Configure a resource-based service: Atlas.....	36
Configure a resource-based service: HBase.....	38
Configure a resource-based service: HDFS.....	40
Configure a resource-based service: HadoopSQL.....	43
Configure a resource-based service: Kafka.....	45
Configure a resource-based service: Knox.....	48
Configure a resource-based service: NiFi.....	49
Configure a resource-based service: NiFi Registry.....	51
Configure a resource-based service: Solr.....	53
Configure a resource-based service: YARN.....	55
Configuring resource-based policies.....	57
Configure a resource-based policy: Atlas.....	58
Configure a resource-based policy: HBase.....	59

Configure a resource-based policy: HDFS.....	62
Configure a resource-based policy: HadoopSQL.....	64
Configure a resource-based storage handler policy: HadoopSQL.....	67
Configure a resource-based policy: Kafka.....	72
Configure a resource-based policy: Knox.....	73
Configure a resource-based policy: Kudu.....	75
Configure a resource-based policy: NiFi.....	77
Configure a resource-based policy: NiFi Registry.....	79
Configure a resource-based policy: S3.....	81
Configure a resource-based policy: Solr.....	83
Configure a resource-based policy: YARN.....	85
Wildcards and macros in resource-based policies.....	87
Adding a policy condition to a resource-based policy.....	91
Adding a policy label to a resource-based policy.....	93
Preloaded resource-based services and policies.....	94
Importing and exporting resource-based policies.....	100
Import resource-based policies for a specific service.....	103
Import resource-based policies for all services.....	105
Export resource-based policies for a specific service.....	108
Export all resource-based policies for all services.....	109
Row-level filtering and column masking in Hive.....	111
Row-level filtering in Hive with Ranger policies.....	112
Dynamic resource-based column masking in Hive with Ranger policies.....	115
Dynamic tag-based column masking in Hive with Ranger policies.....	119
Tag-based Services and Policies.....	123
Adding a tag-based service.....	123
Adding tag-based policies.....	125
Using tag attributes and values in Ranger tag-based policy conditions.....	127
Adding a policy condition to a tag-based policy.....	128
Adding a tag-based PII policy.....	129
Default EXPIRES ON tag policy.....	132
Importing and exporting tag-based policies.....	134
Import tag-based policies.....	136
Export tag-based policies.....	138
Configuring Ranger TagSync with AtlasREST.....	140
Enabling tag-based policies for Ranger RAZ.....	141
Handling inconsistent username and group name conventions for consistent authorization.....	142
Create a time-bound policy.....	142
Enabling IP-based access control in Ranger.....	144
Create a Hive authorizer URL policy.....	147
Hive authorizer URL policy with hdfs default value.....	149
Showing Role Grant definitions from Ranger HiveAuthorizer.....	149

Ranger Security Zones.....	150
Security Zones Administration.....	150
Security Zones Example Use Cases.....	151
Adding a Ranger security zone.....	153
 Administering Ranger Reports.....	 159
View Ranger reports.....	159
Search Ranger reports.....	160
Export Ranger reports.....	161
 Using Ranger client libraries.....	 162
 Using session cookies to validate Ranger policies.....	 163
 Configure optimized rename and recursive delete operations in Ranger	
Ozone plugin.....	163
 How to optimally configure Ranger RAZ client performance.....	 164

Using Ranger to Provide Authorization in Cloudera

Apache Ranger manages access control through a user interface that ensures consistent policy administration across Cloudera 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 resources (HDFS, HBase, Hive, etc.) and add access policies to those services. Ranger security zones enable you to organize service resources into multiple security zones. 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 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.

**Note:**

You can configure authorization using the Ranger UI, REST APIs, or client libraries. For more information about:

- Ranger REST APIs, see <https://ranger.apache.org/apidocs/index.html>.
- Ranger client libraries, see Using Ranger client libraries.

Ranger plugin overview

Ranger enforces authorization using a plugin model.

Ranger at the core has a centralized web application, which consists of the policy administration. These policies are enforced within the Hadoop ecosystem using lightweight Ranger Java plugins. These plugins run as part of the same process as the namenode (HDFS), HiveServer2(Hive), HiveMetaStore, HBase server (Hbase), Kafka, Solr, NiFi, Raz, RazS3, ADLS, Yarn and Knox server (Knox). Plugins are enabled by default for each of these components except (Solr) and can be disabled individually, using Cloudera Manager.

Ranger plugins exist in the path of the user request. Each plugin decides whether to allow or deny user requests for accessing. Each plugin also collects and stores the access request details as access audit log records.

Ranger plugins enforce the policies defined in the policy database. Ranger Admin users can create a policy for a specific set of resources and assign a specific set of permissions to a specific set of users, groups and roles. Ranger admin users manage policies using the Ranger Admin Web UI.

Ranger policies are independent from native permissions (os permission). Ranger uses native permissions to authorize user access in the case that an applicable Ranger policy does not exist in the policy database.

Ranger Hive Plugin

Describes how the Ranger Hive plugin enforces authorization.

Ranger Hive Plugin is enabled in HiveServer2 which helps in storage-based authorization and SQL-standard authorization. In storage-based authorization when a new table is created by running CREATE TABLE statement in Beeline, which will submit query to HiveServer2 for processing, and before HiveServer2 is able to run the query, it will check the policy cache file and make sure the user who submits the query has the appropriate permission to perform the task. Once the authorization passes, a query is submitted and a table created.

Upon successful creation of the new table, the following will be triggered by Ranger's Hive plugin:

1. Sends audit event to Solr and/or HDFS

SQL standard authorization provides grant/revoke functionality at database, table level. When a grant command is executed in beeline it updates/creates a policy for that user and when revoke is executed the user is added in the deny condition of the policy.

Ranger Hive Plugin Enforcement Example

Prerequisite

1. Create a database, table, column in hive service and also insert some data into it with hive user.
 - create database vehicle;
 - create table vehicle.cars(car_id int, car_name string, car_color string, car_price int);"
 - insert into vehicle.cars(car_id, car_name, car_color, car_price) VALUES (1,'car1','color1',100000), (2,'car2','color2',200000), (3,'car3','color3',300000), (4,'car4','color4',400000);
 - select * from vehicle.cars;
2. Create external user 'externaluser1'

Access Enforcement steps

1. Let's try to access the vehicle.cars table using 'externaluser1'.

'externaluser1' will be denied access, because 'externaluser1' lacks permission to access the vehicle.cars table.
2. Lets create a policy in ranger-hive for the user:
 - Resource : [database=vehicle, table=cars, column=*]
 - allow policy item : [user='externaluser1', permission=select]
3. Let's try to access the vehicle.cars table using 'externaluser1'.

'externaluser1' will be allowed access, because 'externaluser1' now has permission to access the vehicle.cars table.
4. You can check the logs related to these actions, using Ranger Admin Web UI Access Audit tab.

Masking Enforcement steps

Suppose you don't want to show the car_price to 'externaluser1' user so we can mask the data of that column for that user.

1. Lets create a masking policy in ranger-hive for the user:
 - Resource : [database=vehicle, table=cars, column=car_price]
 - allow policy item : [user='externaluser1', permission=select, Select Masking Option=Partial mask: show last 4]
2. Now let's try to access the vehicle.cars table using 'externaluser1'

'externaluser1' will see the car_price - only last 4 digits - because 'externaluser1' has masked access to vehicle.cars table.

Row Level Filtering steps

Suppose you don't want to show the only one row to 'externaluser1' user so we can do it using the row filter policy.

1. Lets create a masking policy in ranger-hive for the user:
 - Resource : [database=vehicle, table=cars]
 - allow policy item : [user='externaluser1', permission=select, Row Level Filter=car_color = 'color4']
2. Now let's try to access the vehicle.cars table using 'externaluser1'.

'externaluser1' will see only the row whose car_color is 'color4'.

Table 1: Hive Commands to Ranger Permission Mapping

Permission	Action
SELECT	Gives read access to an object.
CREATE	Hive Create Table statement is used to create table.
UPDATE	Gives the ability to run update queries on an object (table).
ALTER	You can rename the table and column of existing Hive tables. You can add a new column to the table. Rename Hive table column. Add or drop table partition. Add Hadoop archive option to Hive table.
DROP	DROP TABLE command in the hive is used to drop a table inside the hive.
INDEX	An Index is nothing but a pointer on a particular column of a table. Creating an index means creating a pointer on a particular column of a table.
LOCK	Is used to lock the table.
Read	Read data from HDFS using hdfs or other cloud locations.
Write	Export Data to a location in hdfs or other cloud locations.
ReplAdmin	ReplAdmin privilege is related to REPL DUMP and REPL LOAD commands.
Service Admin	Enable hive ranger plugin to isolate various admin operations, in this case "Kill Query". "Service Admin" won't be able to do DATABASE/TABLE/COLUMN operations as this will all be taken care by the existing DATABASE/TABLE/COLUMN level permission model.
Temporary UDF Admin	Temporary UDF Admin is needed for creating UDFs.
Refresh	Refresh is used by only impala.
ALL	This is for all the permission mentioned above.

Ranger Kafka Plugin

Describes how the Ranger Kafka plugin enforces authorization.

Ranger Kafka plugin is enabled in master.

Ranger Kafka Plugin Enforcement Example

Prerequisite

1. Create external user 'externaluser3'

Access Enforcement steps

1. Let's try to create a topic and send some data using 'externaluser3', he will be denied as he doesn't have permission to create it.
2. Lets create a policy in ranger-kafka for the user
 - Resource : [Topic=topicstest01]
 - allow policy item : [user='externaluser3', permission=publish, consume, describe, create]
3. Let's try to create a topic and send some data using 'externaluser3', he will be allowed as he gets permission to access it.
4. You can check the logs related to these actions, using Ranger Admin Web UI Access Audit tab.

Table 2: Kafka Commands to Ranger Permission Mapping

Permission	Action
Resource = topic	
Publish, Describe, Create	To produce topic and publish
Describe, Create	To describe topic
Describe	sending message to topic
Publish	To publish topic
Consume	To read data (consume)
Describe	To list topic
Configure	To alter config of topic
Delete	To delete topic
Describe Config	To describe config of topic
Alter Config	To alter config
Resource = consumergroup	
Describe	To describe topic
Consume	To consume topic
Resource = cluster	
Create	To create topic
Describe	To describe topic
Idempotent Write	To write idempotently
Resource = transactionid	
Describe, Publish	To publish and describe

Ranger-HBase Plugin

Describes how the Ranger HBase plugin enforces authorization.

Ranger HBase Plugin is enabled in master which helps in authorization of the column-oriented database management system.

Ranger HBase Plugin Enforcement Example

Prerequisite

1. Create a sample table called "data", with two column families, and add some rows to the table with hbase user.
 - create 'Testtable1', 'personal', 'medical','finance'
 - put 'Testtable1', '1', 'personal:fname', 'Mike'
 - get 'Testtable1','1','personal:fname'
2. Create external user 'externaluser2'

Access Enforcement steps

1. Let's try to access the 'Testtable1' table using 'externaluser2', he will be denied as he don't have permission to access it.

2. Lets create a policy in ranger-hive for the user
 - Resource : [HBase Table=Testtable1, Column-family=*, Column=*]
 - allow policy item : [user='externaluser2', permission=read]
3. Let's try to access the vehicle.cars table using 'externaluser2', he will be allowed as he gets permission to access it.
4. You can check the logs related to these actions, using Ranger Admin Web UI Access Audit tab.

Table 3: HBase Commands to Ranger Permission Mapping

Permission	Action
Read (R)	can read data at the given scope
Write (W)	can write data at the given scope
Execute (X)	can execute coprocessor endpoints at the given scope
Create (C)	can create tables or drop tables (even those they did not create) at the given scope
Admin (A)	can perform cluster operations such as balancing the cluster or assigning regions at the given scope

Ranger special entities

Ranger in Cloudera has specific, internal groups and entities that affect user authorization and access to all services in Cloudera.

In addition to any users, group, roles and permissions that you define using Ranger, you must understand the following Ranger special entities:

"public" group

A special, internal group within Ranger that consists of all users, including future users. Membership is implicit and automatic.



Note: All users belong to "public" group. Any policies granted to this group provide access to everyone.

The following, default policies give permissions to members of group "public":

- all - database > public > create permission
- default database tables columns > public > create permission
- Information_schema database tables columns > public > select permission

You can remove "public" from these default policies to further restrict user access, based on your security requirements.

{OWNER} special entity

A special Ranger entity attached to a user based on their actions. For example, when a user "bob" creates a table, "bob" becomes the {OWNER} of that table and would get any permissions provided to {OWNER} on that table across all the policies. The following default policies have permissions for {OWNER}:

- all - database, table, column > {OWNER} > all permissions
- all - database, table > {OWNER} > all permissions
- all - database, udf > {OWNER} > all permissions
- all - database > {OWNER} > all permissions

Although not recommended, you can modify access to special entity {OWNER}, based on your security requirements. Removing the default {OWNER} permissions may require adding additional,

specific policies for each object owner, which may increase your policy management operational burden.

Enabling Ranger HDFS plugin manually on a Cloudera Data Hub

How to enable an HDFS plugin for Ranger, service-wide, on a Cloudera Data Hub using Cloudera Manager.

About this task

The Ranger HDFS plug-in helps to centralize HDFS authorization policies. Apache Ranger plugins validate the access of a user against the authorization policies defined in the Apache Ranger policy administration server, and stored in the HDFS service instance, also called a repository. When you enable the Ranger HDFS plugin and an HDFS service user attempts access, Ranger checks whether a policy exists granting or denying the user access. If no policy exists, Ranger defaults to use the native permissions model in HDFS. Access control rules configured through this combination of Ranger HDFS plugin and native file system permissions apply.

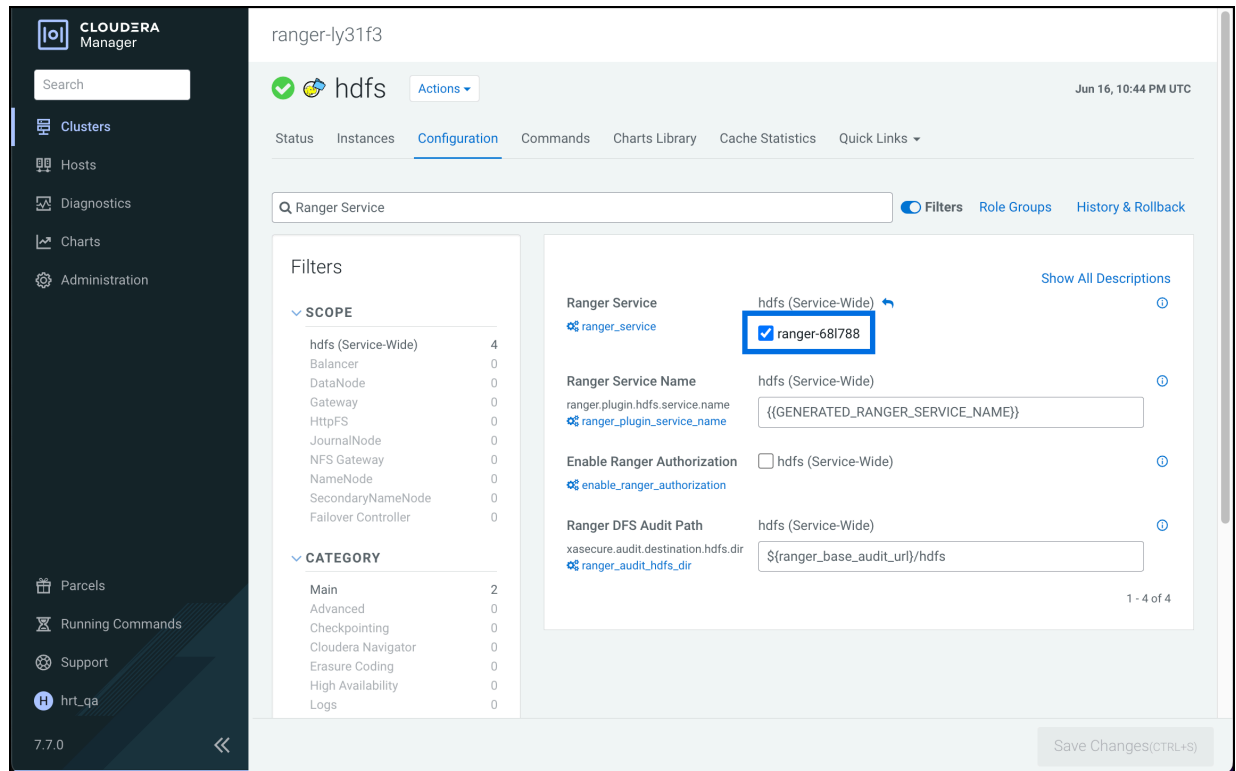
To enable users define Ranger authorization policies, using an HDFS service plugin:

Procedure

1. In a Cloudera Data Hub, go to Cloudera Manager HDFS Configuration .

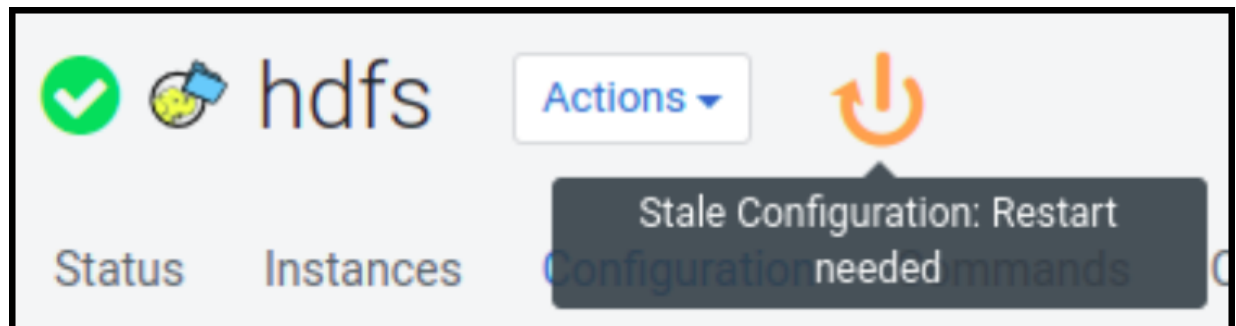
2. In Search, type Ranger Service, then click the box to enable the `hdfs` (service-wide) parameter for Ranger Service.

Figure 1: Enabling the HDFS Ranger plugin parameter on a Cloudera Data Hub



A stale configuration icon displays for the hdfs service.

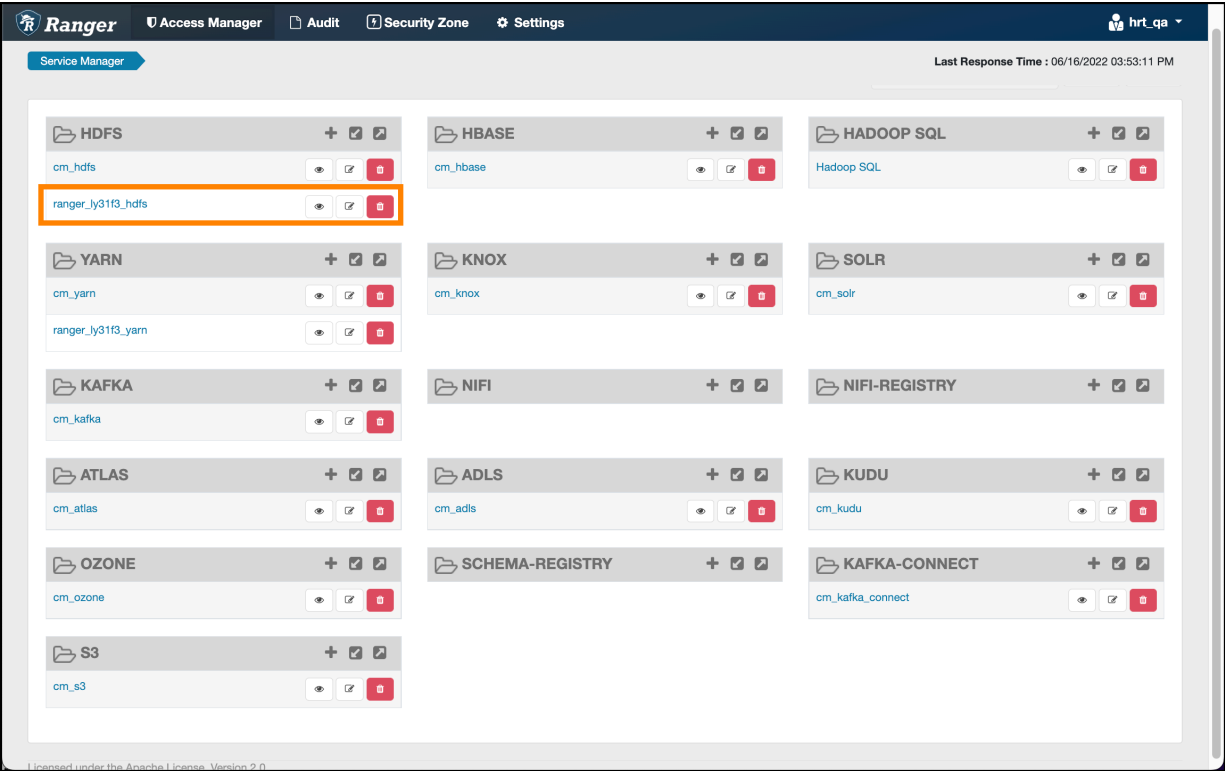
Figure 2: Result of adding a new configuration parameter



3. Before restarting Hdfs service for stale configurations, choose **HDFS Actions Create Ranger Repository** . After progress completes, close the Create Repository dialog.
4. Now proceed to restart the Hdfs service. Click **HDFS Actions Restart** . After progress completes, close the Restart dialog.
5. On the Data Lake, log in to Ranger.
6. Go to **Admin Web UI Access Manager** .

7. In Service Manager HDFS , confirm that (Cloudera Data Hub cluster name)_hdfs appears.

Figure 3: Confirming HDFS plugin added

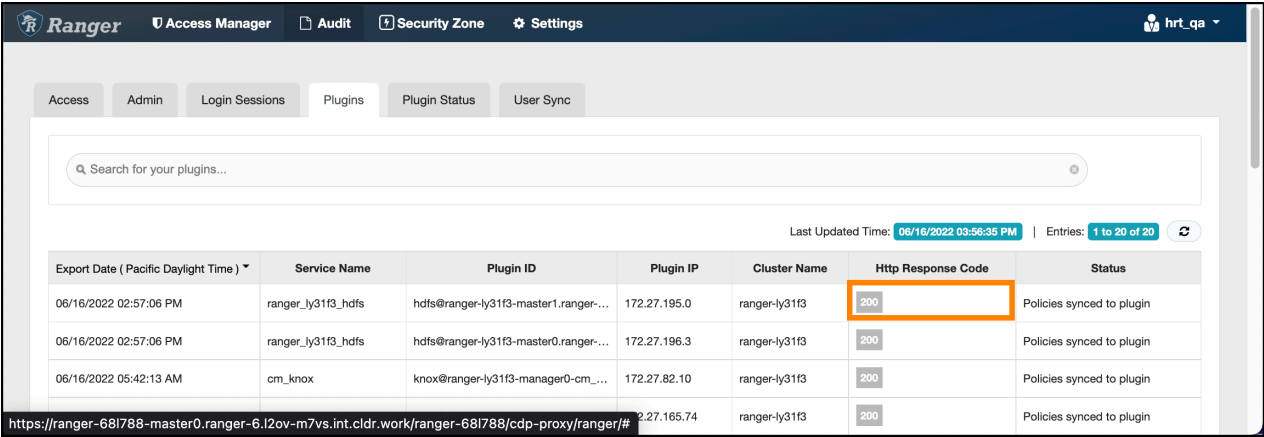


8. Go to Audit Plugins .

Results

Confirm that the Http response code for the Ranger Hdfs plugin enabled on the Cloudera Data Hub Hdfs service displays 200 (successful).

Figure 4: Confirming successful http response



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.



Note: Tags applied on a Hive table are propagated to the views created from that table. Hence, if any Ranger tag based access or masking policies are associated with those tags, then the views also have those policies applied.

- 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.



Note: Ranger tag-based policies are used to restrict data access; they do not affect metadata in metadata services, such as Cloudera Data Catalog. If an asset is protected by a tag-based policy, you can use a resource-based policy on Atlas instead of tag-based policies to manage access control on metadata.

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

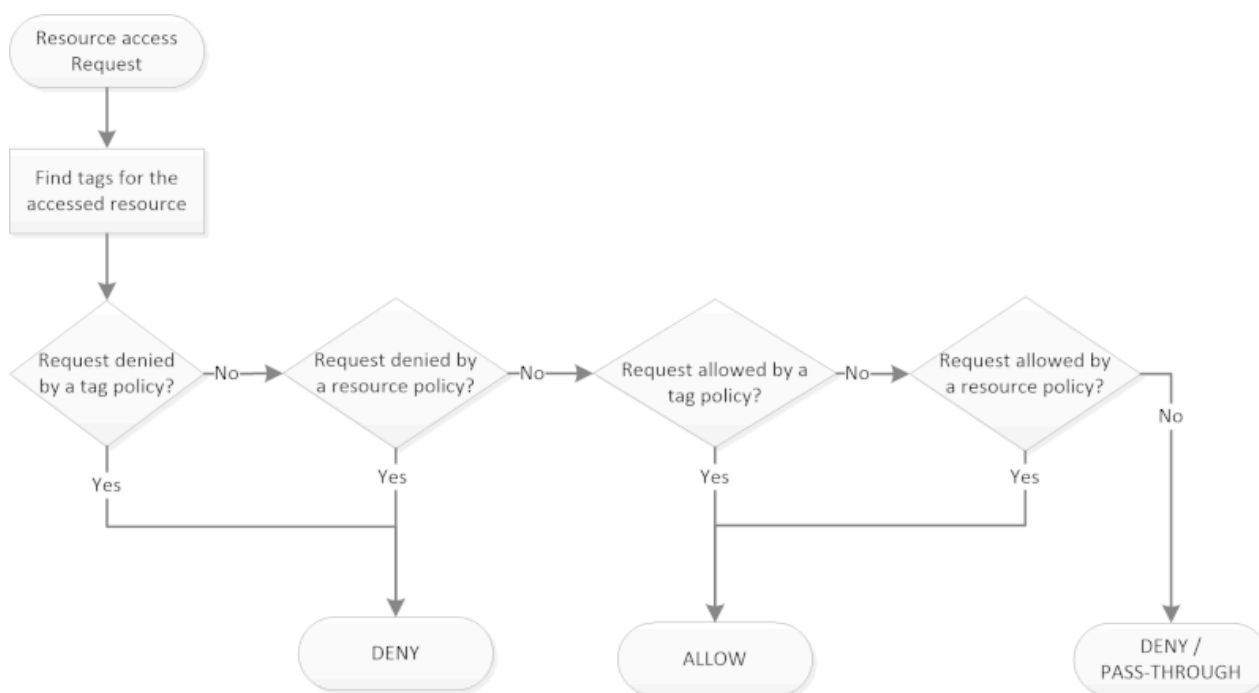
Ranger TagSync is used to synchronize the tag store with an external metadata service such as Apache Atlas. TagSync is a daemon process similar to the Ranger UserSync process.

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](#)

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 * **enabled**

Hive Database * **Include**

table * **Include** ← **Resource**

Hive Column * **Include**

Description

Audit Logging **YES**

Allow Conditions :

Select Group	Select User	Permissions	Delegate Admin	
finance	Select User	All edit	☒	X

Exclude from Allow Conditions :

Deny Conditions :

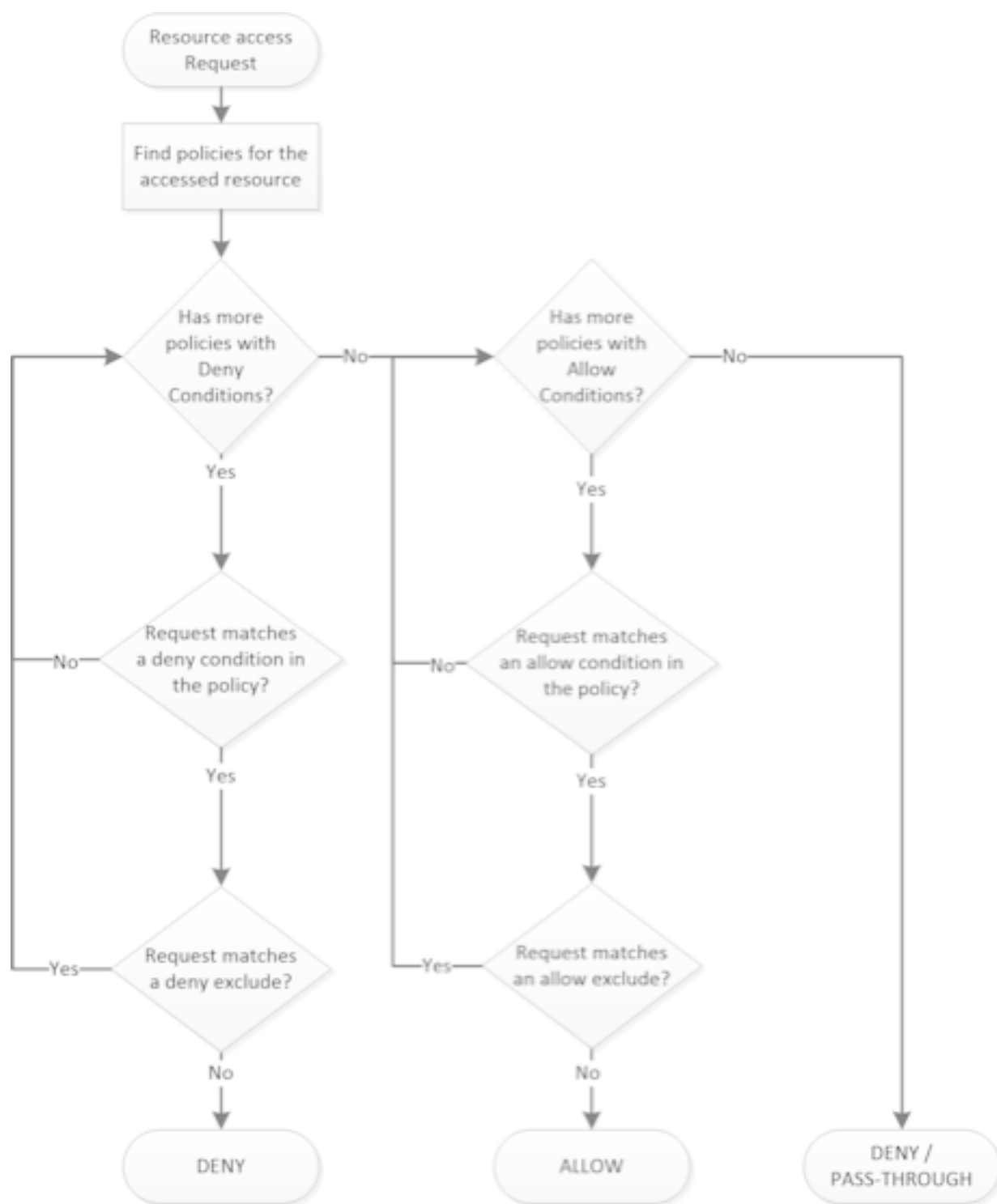
Select Group	Select User	Permissions	Delegate Admin	
interns	Select User	All edit	☒	X

Exclude from Deny Conditions :

Select Group	Select User	Permissions	Delegate Admin	
Select Group	X SCOR	select edit	☐	X

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

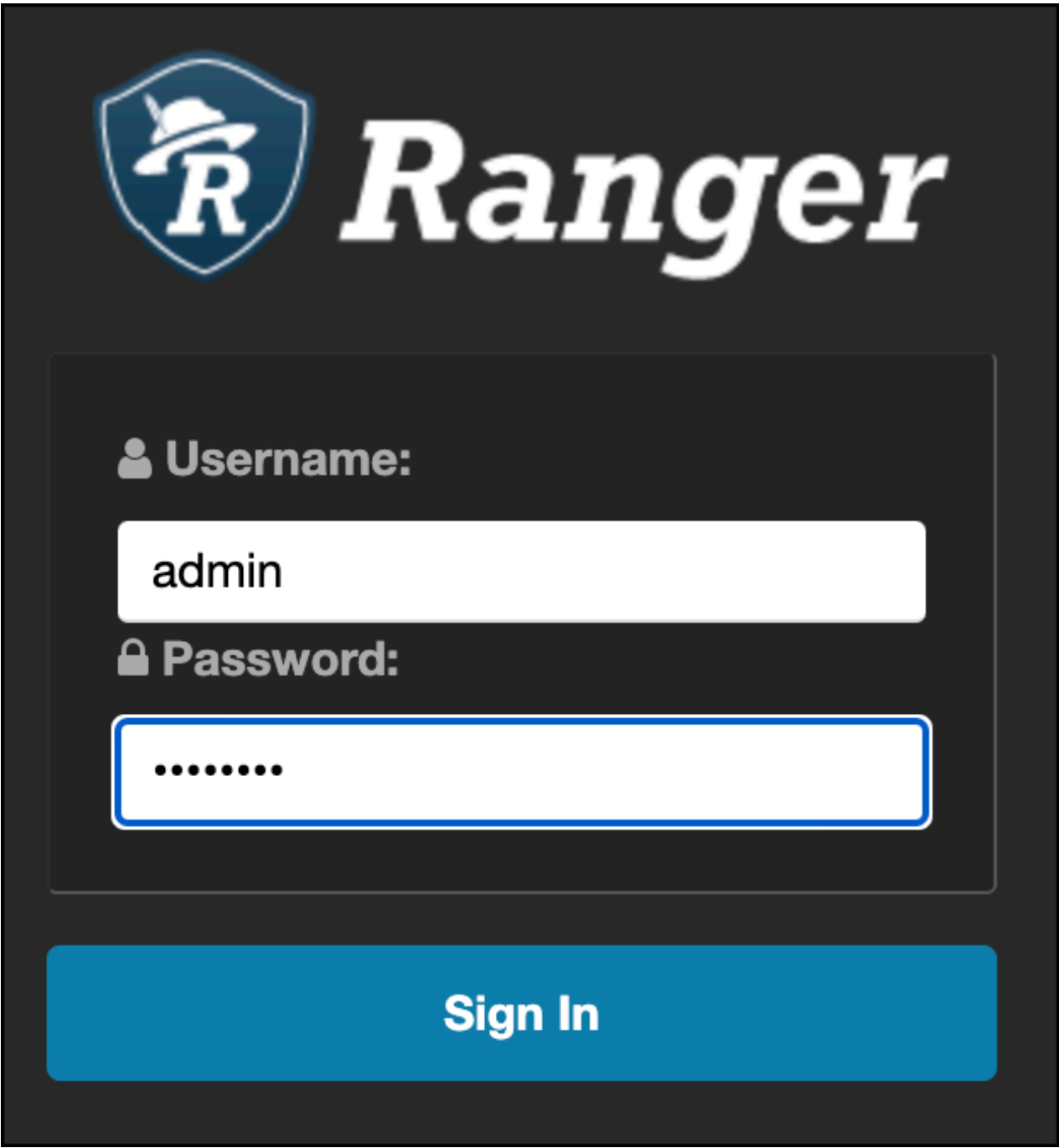
Using the Ranger Admin Web UI

Log in through the Ranger Admin Web UI as a Ranger administrator user to administer auditing, services for Cloudera resources, access policies for those services and permissions to the Ranger Admin modules for other Ranger users, groups and roles.

Accessing the Ranger Admin Web UI

How to access the Ranger Admin Web UI.

To access the Ranger Admin Web UI, go to Cloudera Manager Ranger Ranger Admin Web UI , type your user name and password, and then click Sign In.



The image shows the Ranger Admin Web UI Home Page. At the top, there is a logo consisting of a blue shield with a white 'R' and a white ranger hat, followed by the word 'Ranger' in a large, white, serif font. Below the logo, there is a dark gray rectangular box containing the login fields. The first field is labeled 'Username:' with a person icon, and it contains the text 'admin'. The second field is labeled 'Password:' with a lock icon, and it contains a series of dots. Below these fields is a large blue button with the text 'Sign In' in white.

 **Ranger**

 **Username:**

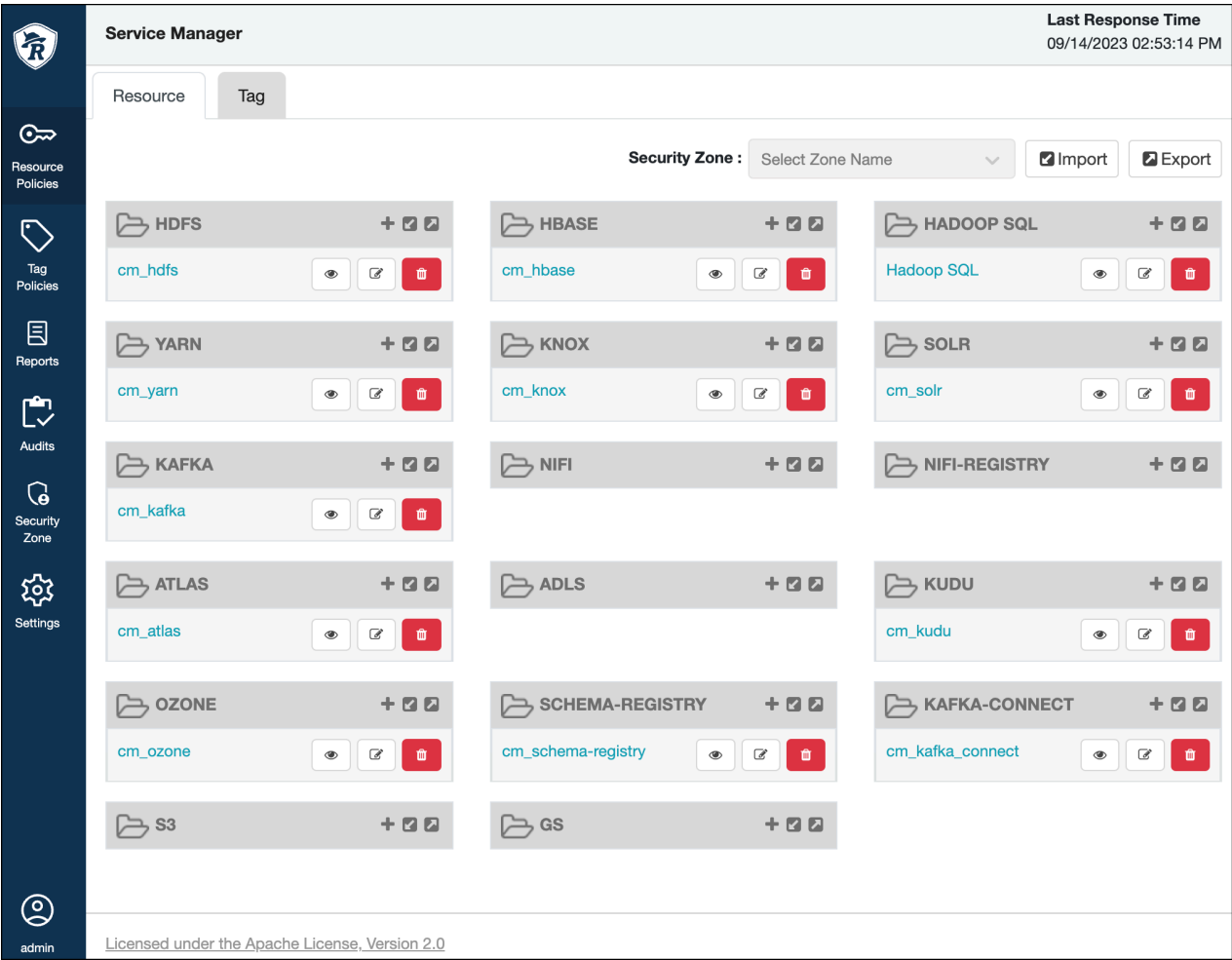
admin

 **Password:**

.....

Sign In

Ranger Admin Web UI Home Page



After you log in, your user name is displayed at the lower left of the Ranger Admin Web UI.

Ranger console navigation

Explains the basic Ranger console/GUI.

- The Service Manager for Resource Based Policies page displays when you log in to Ranger Admin Web UI. You can use Service Manager to create services for Cloudera resources (HDFS, HBase, Hive, etc.) and add access policies to those resources.

Service Manager Last Response Time
09/14/2023 02:53:14 PM

Resource **Tag**

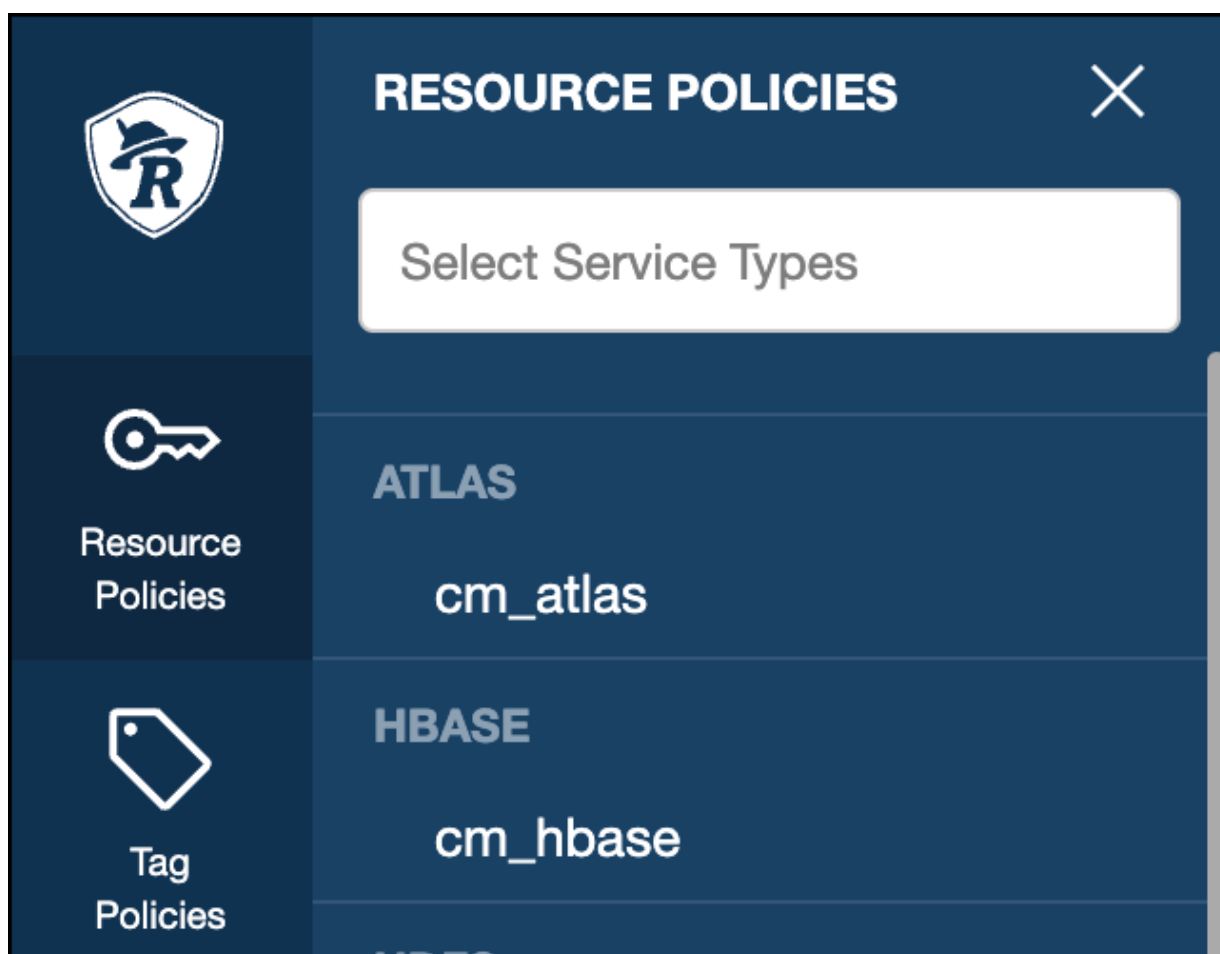
Security Zone : ☒ Import ☒ Export

HDFS + [check] [lock] cm_hdfs [eye] [edit] [trash]	HBASE + [check] [lock] cm_hbase [eye] [edit] [trash]	HADOOP SQL + [check] [lock] Hadoop SQL [eye] [edit] [trash]
YARN + [check] [lock] cm_yarn [eye] [edit] [trash]	KNOX + [check] [lock] cm_knox [eye] [edit] [trash]	SOLR + [check] [lock] cm_solr [eye] [edit] [trash]
KAFKA + [check] [lock] cm_kafka [eye] [edit] [trash]	NIFI + [check] [lock]	NIFI-REGISTRY + [check] [lock]
ATLAS + [check] [lock] cm_atlas [eye] [edit] [trash]	ADLS + [check] [lock]	KUDU + [check] [lock] cm_kudu [eye] [edit] [trash]
OZONE + [check] [lock] cm_ozone [eye] [edit] [trash]	SCHEMA-REGISTRY + [check] [lock] cm_schema-registry [eye] [edit] [trash]	KAFKA-CONNECT + [check] [lock] cm_kafka_connect [eye] [edit] [trash]
S3 + [check] [lock]	GS + [check] [lock]	

admin Licensed under the Apache License, Version 2.0

Use the left navigation panel to navigate the Ranger Admin Web UI.

To return to the Service Manager home page, click the Ranger icon at the upper left corner of the Ranger Admin Web UI page.



- Resource Policies -- Clicking Resource Policies displays a list of resource-based policies. Click a specific policy name to open policy management page for the selected service. You can use the policy page to administer access policies for that service.
- Tag Policies -- Clicking Tag Policies displays a list of resource-based policies. Click a specific policy name to open policy management page for the selected tag-based policy. You can use the Tag policy page to administer access policies for tag-based policies.
- Reports -- Clicking Reports opens the Reports page. You can use the Reports page to generate user access reports for resource and tag-based policies based on search criteria such as policy name, resource, group, and user name.
- Audits -- Click Audits, then select Access, Admin, Login Sessions, Plugins, Plugin Status or User Sync to access the Audit page Access, Admin, Login Sessions, Plugins, Plugin Status, and User Sync tabs. These UIs provide

administrator access to monitor user activity at the resource level, and also to set up conditional auditing based on users, groups, or time.

Audits

Access

Admin

Login Sessions

Plugins

Plugin Status

User Sync

Metrics

Q

START DATE : 09/14/2023

Exclude Service Users: ☐

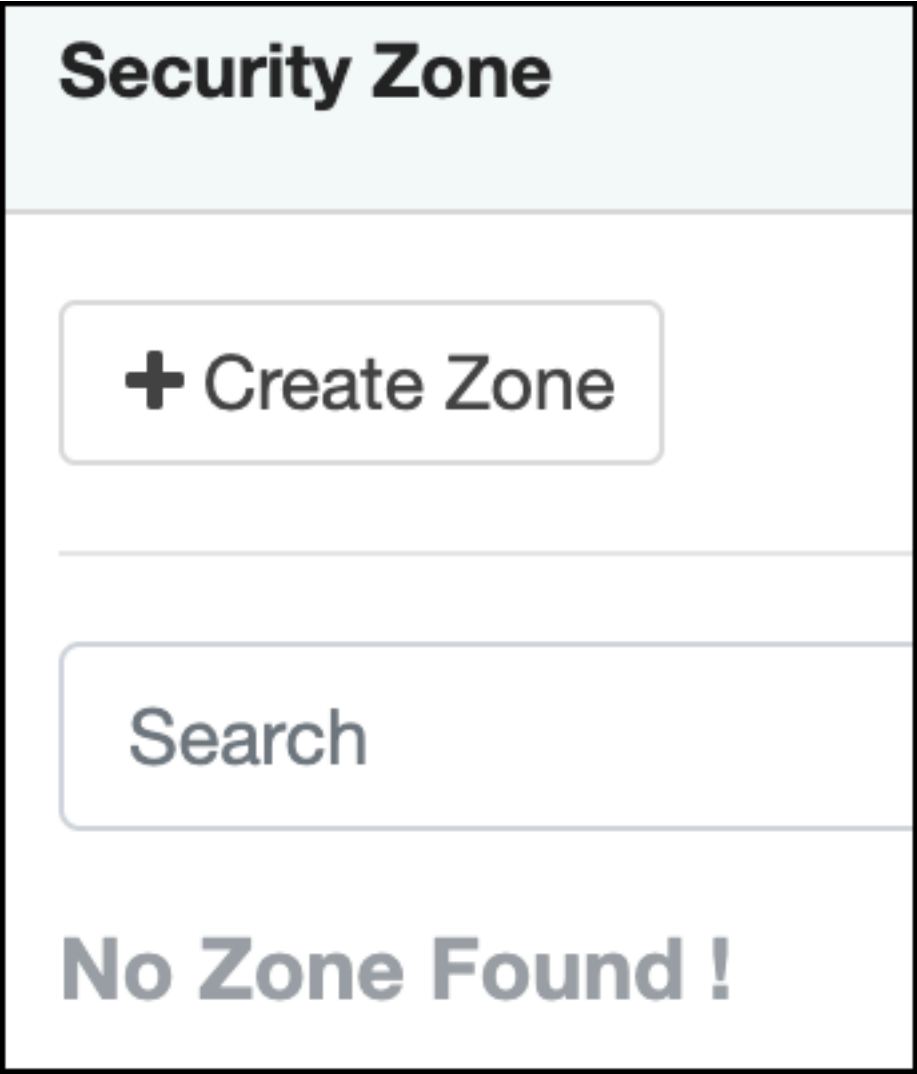
Last Updated Time: 09/14/2023 03:41:23 PM

Entries: 1 to 25 of 2057192

Columns ▾

Policy ID	Policy Version	Event Time ▾	Application	User	Service (Name / Type)	Resource (Name / Type)	Access Type	Permission	Result	Access Enforcer
26	1	09/14/2023 3:41:09 PM	kafka	streamsrepmgr	cm_kafka kafka	srm-meta.internal topic	describe	describe	Allowed	ranger-acl
20	1	09/14/2023 3:41:08 PM	ozone	hue	cm_ozone ozone	s3v/cloudera-health-m... bucket	read	read	Allowed	ranger-acl
27	1	09/14/2023 3:41:07 PM	kafka	streamsrepmgr	cm_kafka kafka	secondary-mm2 consumergroup	consume	consume	Allowed	ranger-acl

- Security Zone -- Lets you organize resource and tag-based services and policies into separate security zones. You can assign one or more administrators for each security zone. Security zone administrators can then create and update policies for their security zone.



- Users/Groups/Roles

Last Response Time 09/14/2023 03:47:00 PM

Users

Groups

Roles

Add New User

Set Visibility ▾

🗑️

☐	User Name	Email Address	Role	User Source	Sync Source	Groups	Visibility	Sync Details
☐	admin	--	Admin	Internal	--	--	Visible	--
☐	rangerusersync	--	Admin	Internal	--	--	Visible	--
☐	rangertagsync	--	Admin	External	Unix	rangertagsync	Visible	
☐	hdfs	--	User	External	Unix	hadoop hdfs	Visible	
☐	hive	--	User	External	Unix	hive	Visible	

25

Policy conditions in Apache Ranger

Policy conditions in Apache Ranger enable dynamic, context-aware access control by using JavaScript expressions to evaluate requests at the policy and policy-item levels. Instead of just specifying who can access what, you can also control under which conditions access is allowed.

Apache Ranger enables dynamic policy conditions through JavaScript expressions evaluated at request time to decide whether to evaluate the policy or a policy item.

These expressions must evaluate to boolean values and are used in the following configurations:

- Resource-based policies (allow and deny)
- Tag-based policies (allow and deny)
- Row-level filters
- Column masking

If the condition evaluates to true, access continues to be evaluated. If false, the policy or a policy item is skipped.



Note: Policy conditions can be set at the policy level, for example, policy details, as well as at policy-item level, for example, rules. The policy-level condition acts as a master gateway for all rules within the policy. Policy-item-level conditions apply to specific rules within a policy and are only evaluated if the policy-level condition evaluates to True. The evaluation is sequential and acts as a logical AND.

Evaluation order and precedence

Apache Ranger evaluates a policy that has conditions at both policy level and policy-item level in the following order:



Note:

- Policy-level conditions have absolute precedence.
- Policy-item-level conditions cannot override policy-level conditions.

1. Matches the policy.

Apache Ranger first identifies a list of policies, where the requested resource, for example, an HDFS path or a Hive table, matches the policy resource definition.

2. Evaluates the policy-level condition.

Apache Ranger then evaluates the custom condition set at the main policy level.

- If this condition evaluates to False, the entire policy is skipped. None of the individual allow or deny rules within it are even considered, and Apache Ranger moves to the next policy.
- If this condition evaluates to True or if no policy-level condition exists, Apache Ranger proceeds to evaluate the individual rules (policy items) within that policy.

3. Evaluates the policy-item-level condition.

Apache Ranger evaluates the specific condition attached to each rule (policy item) within the matched policy.

- If the rule-level condition evaluates to True, the policy engine in that rule checks for the user, group, and access privileges in that specific policy item.
- If the rule-level condition evaluates to False, that specific rule is ignored.

Evaluation context in policy conditions

The evaluation context in Apache Ranger enables dynamic, attribute-based access control (ABAC) by gathering and enriching request details for precise policy decisions.

When a user tries to access a resource, such as a Hive table or an HDFS file, the Apache Ranger plugin intercepts this action. It gathers all relevant details about the request into a package of information. This package is called the context enricher.

Context enricher

The context enricher is a package of information gathered by the Apache Ranger plugin for every resource access request. It serves as the foundation for the policy engine to make a precise access decision. It contains all the live data points related to the access attempt.

The context enricher carries answers to the following key contextual questions:

Table 4: Evaluation context elements

Context element	Description	Example data
Who (Identity)	The principal (user and associated groups) making the request.	user: john_doe, groups: finance_analysts
What (Resource)	The specific resource the user is attempting to access.	resource: /data/finance/salaries.csv (HDFS path)
How (Action)	The operation being performed.	action: read
Where (Origin)	The network location of the client making the request.	clientIPAddress: 10.0.2.15
When (Time)	The current time of the request.	currentTimestamp: 2026-01-29T21:18:00Z

During request processing, the context enricher adds more specialized information to the base context package, enabling more complex policy conditions.

Table 5: Context enrichers

Enricher	Data added to context	Description
Tag enricher	TAGS, TAG_OBJECT	Incorporates metadata tags associated with the resource.
User store enricher	USERSTORE attributes	Adds user or group attributes fetched from external stores, for example, LDAP.
GDS enricher	Governance, data sharing information	Integrates data governance and sharing details.
Zone enricher	Security zone information	Identifies the security zone the resource belongs to.

Context population lifecycle

The evaluation context is built through the following distinct stages within the plugin:

- Initial setup – The plugins gather basic request information, for example, resource path and requested action.
- Enrichment phase – Context enrichers run, adding specialized data, such as tags and user attributes, to the context.
- Evaluation phase – The policy engine checks the enriched context against defined policies. Evaluators might also potentially modify the context.
- Cleanup phase – Any temporary data added to the context is removed.



Note: Context evaluation takes place on the plugin side, at the component level, for example, within the Hive, HDFS, or Kafka plugin.

Custom conditions and attribute-based access control (ABAC)

The ctx object is the key to implementing dynamic, attribute-based access control (ABAC). It allows policies to go beyond simple identity and resource checks to evaluate environmental, user, or time-based attributes.

- **Static rule example (without ctx):** Allow group `finance_analysts` to read table `salaries`.
- **Dynamic ABAC rule example (with ctx):** Allow group `finance_analysts` to read table `salaries` only if the request is coming from an internal IP address and it is during business hours.

You can write custom conditions as JavaScript-like expressions in the Custom Conditions box within the Ranger Policy UI.

Table 6: Example expressions using ctx

Policy objective	Example expression
Restrict access by IP Address	<code>ctx.request.clientIPAddress == '10.0.2.15'</code>
Restrict access by Time	<code>ctx.request.currentTimestamp.getHours() >= 9 && ctx.request.currentTimestamp.getHours() <= 17</code>

This expression evaluates to true only if the request originates from the specified IP address and occurs during the specified hours. If the expression evaluates to true, the policy condition is met, and the access decision (allow/deny) is processed.

ctx and _ctx object in policy conditions

The `ctx` and `_ctx` objects in Apache Ranger enable dynamic, attribute-based policy conditions by providing the evaluation context. These objects allow for advanced access control decisions through JavaScript.

By leveraging the `ctx` object, the policy conditions evaluate factors such as how, when, and where the access request is made. This approach enables the application of a unified rule across all permissions within the policy.

When you define a condition at the policy level, you establish a master gateway. This gateway is a single JavaScript expression that must evaluate to True for Apache Ranger to even consider any of the individual allow or deny rules within that policy.

Policy-level conditions in Apache Ranger utilize JavaScript scripts to assess contextual information. The context evaluation system offers a comprehensive set of data through the `ctx` or `_ctx` context objects, which JavaScript scripts can use to make advanced policy decisions.

The following table outlines the key differences between the `ctx` and `_ctx` objects:

Table 7: Comparing ctx and _ctx

Aspect	ctx	_ctx
Type	Java object with methods	JavaScript object with data
Access pattern	<code>ctx.methodName()</code>	<code>_ctx.property.subproperty</code>
Best for	Built-in utility functions	Direct data access
Examples	<code>ctx.hasTag('PCI')</code>	<code>_ctx.request.user</code>

Best practices

The dual-context system in Apache Ranger offers policy authors maximum flexibility when writing JavaScript conditions. It provides convenience methods through `ctx` and raw data access through `_ctx`, enabling efficient and versatile policy creation. Cloudera recommends using the following best practices:

- Use `ctx` to complete the following tasks:
 - Perform time-based validations, such as `ctx.isAccessedAfter()`
 - Run built-in checks, including `ctx.hasTag()` and `ctx.isInGroup()`
 - Execute complex utility functions

- Use `_ctx` to complete the following tasks:
 - Access data directly, for example `_ctx.request.user`
 - Perform complex data comparisons
 - Apply custom logic on raw data
- Use a hybrid approach:
 - Combine both `ctx` and `_ctx` objects for optimal performance and readability
 - Leverage `ctx` methods for intensive operations and use `_ctx` for direct data access

Context structure available in JavaScript

The `_ctx` object serves as a gateway to the following types of categorized information:

- Request information (`_ctx.request`)
 - Resource details – Access resource-specific data using `_ctx.request.resource.*`
 - User information – Retrieve user-related details such as `_ctx.request.user`, `_ctx.request.userGroups`, and `_ctx.request.userRoles`
 - Access details – Understand access specifics through `_ctx.request.accessType` and `_ctx.request.action`
 - User attributes – Access user-specific attributes through `_ctx.request.userAttributes.*`
 - Group attributes – Retrieve group-specific attributes using `_ctx.request.userGroupAttributes.*`
- Tag information (`_ctx.tags`, `_ctx.tag`)
 - All tags – Access all tags by using `_ctx.tags`
 - Current tag – Retrieve the current tag with `_ctx.tag`
 - Tag attributes – Access specific tag attributes such as `_ctx.tag.attr1` or `_ctx.tags['PCI'].attr1`
- Context attributes
 - Custom context – Retrieve custom context attributes by using `ctx.getRequestContextAttribute('key')`

JavaScript examples for policy-level conditions

- Resource-based conditions

```
// Check database access
_ctx.request.resource.database == 'sensitive_db'
// Check multiple resources
_ctx.request.resource.database == 'hr' && _ctx.request.resou
rce.table == 'employees'
// Pattern matching
_ctx.request.resource.database.indexOf('prod') != -1
```

- User and group-based conditions

```
// Check specific user
_ctx.request.user == 'admin'
// Check user groups
_ctx.request.userGroups.indexOf('data_scientists') != -1
// Multiple group check
_ctx.request.userGroups.length >= 2 &&
_ctx.request.userGroups.indexOf('managers') != -1
// User roles check
_ctx.request.userRoles.indexOf('dba') != -1
```

- User or group attribute-based conditions

```
// Check user attributes
_ctx.request.userAttributes['department'] == 'finance'
// Check multiple user attributes
```

```

    _ctx.request.userAttributes['clearance_level'] == 'top_secret' &&
    _ctx.request.userAttributes['location'] == 'headquarters'
    // Group attribute check
    _ctx.request.userGroupAttributes['finance']['budget_access']
    == 'unlimited'

```

- Tag-based conditions

```

<codeblock id="codeblock_tyd_jgx_c3c">
  // Check specific tag type
  _ctx.tag._type == 'PCI'
  // Check tag attributes
  _ctx.tag.level == 'high'
</codeblock>

```

Tag attributes with ctx or _ctx object for policy conditions

Tag attributes in the Atlas-Ranger integration enable a transition from basic resource-based access control to a highly scalable and efficient attribute-based access control (ABAC) model.

Tag attributes enhance basic classifications by adding key-value metadata. For example, rather than merely tagging a Hive table as PII, you can tag it with a classification such as DATA_SENSITIVITY and add attributes such as level='HIGH'.

The Ranger policy engine can read these attribute values during access evaluation. This allows you to create a single, dynamic policy, such as "Allow the compliance_team to access any data where the DATA_SENSITIVITY level is HIGH," instead of creating separate policies for every sensitive data asset.

Creating a tag with attributes in Apache Atlas

You first define the schema of your tag called a classification in Apache Atlas and its potential attributes.

1. Go to the Classifications tab in the Atlas UI.
2. Create a New Classification named DATA_CLASSIFICATION.
3. Add attributes to it by clicking Add Attribute and defining the metadata you want to store.

Example

- Attribute Name: sensitivity_level
 - Data Type: string
4. Apply the tag to a data asset.

For example, find a resource such as a Hive table named customer_contact_info. Apply the new DATA_CLASSIFICATION tag to it. When you do so, the Atlas UI will prompt you to set the values for your attributes.

- Set sensitivity_level to HIGH.

This metadata is now managed in Apache Atlas and will be synchronized to Apache Ranger through TagSync.

Using tag attributes in a Apache Ranger policy condition

After the tag information is in Apache Ranger, you can use it in your policy conditions to access its attributes.

Create a policy that only allows members of the security_auditors group to SELECT data that has been explicitly classified with a CRITICAL sensitivity level.

1. In Ranger Admin Web UI, create a new tag-based policy for the Hive service.
2. Select the tag in the TAG field and select DATA_CLASSIFICATION.
3. Define the policy item (rule) in Allow Conditions.

For example, "Grant SELECT permission to the group security_auditors."

4. Add the custom condition in the Custom Conditions text box and enter the following JavaScript:

```
// This condition checks the value of the 'sensitivity_level' attribute on
// the 'DATA_CLASSIFICATION' tag associated with the resource.
// The policy item will only apply if the value is 'CRITICAL'.

GET_TAG_ATTR('sensitivity_level') == HIGH;
OR
_ctx.tag.sensitivity_level == 'HIGH';
OR
tagAttr.sensitivity_level == 'HIGH';
OR
ctx.tagAttr('sensitivity_level') == 'HIGH';
```

Ranger policy condition macros

Apache Ranger policy macros provide pre-defined JavaScript functions to evaluate tag, user, group, role, and time-based attributes during access requests. Use these macros to simplify the creation of dynamic, context-aware policy conditions without manual string manipulation or complex custom coding.

Table 8: Tag-related macros

Macro	Expands to	JavaScript usage example	Description	Example return
GET_TAG_NAMES()	ctx.tagNames	ctx.tagNames()	Get all tag names as CSV	"PII,FINANCE"
GET_TAG_NAMES_Q()	ctx.tagNamesQ	ctx.tagNamesQ()	Get all tag names as quoted CSV	""PII','FINANCE""
GET_TAG_ATTR_NAMES()	ctx.tagAttrNames	ctx.tagAttrNames()	Get all tag attribute names as CSV	"level,type,owner"
GET_TAG_ATTR_NAMES_Q()	ctx.tagAttrNamesQ	ctx.tagAttrNamesQ()	Get all tag attribute names as quoted CSV	""level','type','owner""
GET_TAG_ATTR('attrName')	ctx.tagAttr	ctx.tagAttr('sensitiveLevel')	Get attribute values from all tags as CSV	"10,5"
GET_TAG_ATTR_Q('attrName')	ctx.tagAttrQ	ctx.tagAttrQ('sensitiveLevel')	Get attribute values from all tags as quoted CSV	""10','5""
GET_TAG_ATTR_CSV('attrName')	ctx.tagAttrCsv	ctx.tagAttrCsv('owner')	Get attribute values as CSV (alias)	"john,mary"
GET_TAG_ATTR_Q_CSV('attrName')	ctx.tagAttrQCsv	ctx.tagAttrQCsv('classification')	Get attribute values as quoted CSV (alias)	""confidential','public""
TAG_ATTR_NAMES_CSV	ctx.tagAttrNamesCsv()	ctx.tagAttrNamesCsv()	Get tag attribute names as CSV	"level,type,owner"
TAG_ATTR_NAMES_Q_CSV	ctx.tagAttrNamesCsvQ()	ctx.tagAttrNamesCsvQ()	Get tag attribute names as quoted CSV	""level','type','owner""
TAG_NAMES_CSV	ctx.tagNamesCsv()	ctx.tagNamesCsv()	Get tag names as CSV	"PII,FINANCE"
TAG_NAMES_Q_CSV	ctx.tagNamesCsvQ()	ctx.tagNamesCsvQ()	Get tag names as quoted CSV	""PII','FINANCE""
HAS_TAG('tagName')	ctx.hasTag	ctx.hasTag('PII')	Check if resource has specific tag	true/false
HAS_ANY_TAG	ctx.hasAnyTag()	ctx.hasAnyTag()	Check if resource has any tags	true/false
HAS_NO_TAG	!ctx.hasAnyTag()	!ctx.hasAnyTag()	Check if resource has no tags	true/false

Macro	Expands to	JavaScript usage example	Description	Example return
HAS_TAG_ATTR('attrName')	ctx.hasTagAttr	ctx.hasTagAttr('sensitiveLevel')	Check if any tag has specific attribute	true/false
JavaScript usage example: <pre>HAS_TAG('PII') && GET_TAG_ATTR('sensitiveLevel') >= 10</pre>				

Table 9: User-related macros

Macro	Expands to	JavaScript usage example	Description	Example return
GET_USER_ATTR_NAMES()	ctx.userAttrNames	ctx.userAttrNames()	Get all user attribute names as CSV	"dept,level,state"
GET_USER_ATTR_NAMES_Q()	ctx.userAttrNamesQ	ctx.userAttrNamesQ()	Get all user attribute names as quoted CSV	""dept','level','state""
GET_USER_ATTR('attrName')	ctx.userAttr	ctx.userAttr('clearanceLevel')	Get user attribute value	"10"
GET_USER_ATTR_Q('attrName')	ctx.userAttrQ	ctx.userAttrQ('department')	Get user attribute value with quotes	""FINANCE""
USER_ATTR_NAMES_CSV()	ctx.userAttrNamesCsv()	ctx.userAttrNamesCsv()	Get user attribute names as CSV	"dept,level,state"
USER_ATTR_NAMES_Q_CSV()	ctx.userAttrNamesCsvQ()	ctx.userAttrNamesCsvQ()	Get user attribute names as quoted CSV	""dept','level','state""
HAS_USER_ATTR('attrName')	ctx.hasUserAttr	ctx.hasUserAttr('clearanceLevel')	Check if user has specific attribute	true/false
JavaScript usage example: <pre>GET_USER_ATTR('clearanceLevel') >= GET_TAG_ATTR('sensitiveLevel')</pre>				

Table 10: Group-related macros

Macro	Expands to	JavaScript usage example	Description	Example return
GET_UG_NAMES()	ctx.ugNames	ctx.ugNames()	Get user's group names as CSV	"admins,finance"
GET_UG_NAMES_Q()	ctx.ugNamesQ	ctx.ugNamesQ()	Get user's group names as quoted CSV	""admins','finance""
GET_UG_ATTR_NAMES()	ctx.ugAttrNames	ctx.ugAttrNames()	Get all group attribute names as CSV	"dept,site,level"
GET_UG_ATTR_NAMES_Q()	ctx.ugAttrNamesQ	ctx.ugAttrNamesQ()	Get all group attribute names as quoted CSV	""dept','site','level""
GET_UG_ATTR('attrName')	ctx.ugAttr	ctx.ugAttr('department')	Get group attribute values as CSV	"FINANCE,SALES"
GET_UG_ATTR_Q('attrName')	ctx.ugAttrQ	ctx.ugAttrQ('location')	Get group attribute values as quoted CSV	""NY','CA""
GET_UG_ATTR_CSV('attrName')	ctx.ugAttrCsv	ctx.ugAttrCsv('costCenter')	Get group attribute values as CSV (alias)	"CC001,CC002"
GET_UG_ATTR_Q_CSV('attrName')	ctx.ugAttrCsvQ	ctx.ugAttrCsvQ('region')	Get group attribute values as quoted CSV (alias)	""East','West""
UG_NAMES_CSV	ctx.ugNamesCsv()	ctx.ugNamesCsv()	Get group names as CSV	"admins,finance"

Macro	Expands to	JavaScript usage example	Description	Example return
UG_NAMES_Q_CSV	ctx.ugNamesCsvQ()	ctx.ugNamesCsvQ()	Get group names as quoted CSV	"admins','finance'"
UG_ATTR_NAMES_CSV	ctx.ugAttrNamesCsv()	ctx.ugAttrNamesCsv()	Get group attribute names as CSV	"dept,site,level"
UG_ATTR_NAMES_Q_CSV	ctx.ugAttrNamesCsvQ()	ctx.ugAttrNamesCsvQ()	Get group attribute names as quoted CSV	"'dept','site','level'"
IS_IN_GROUP('groupName')	ctx.isInGroup	ctx.isInGroup('finance')	Check if user is in specific group	true/false
IS_IN_ANY_GROUP	ctx.isInAnyGroup()	ctx.isInAnyGroup()	Check if user is in any group	true/false
IS_NOT_IN_ANY_GROUP	!ctx.isInAnyGroup()	!ctx.isInAnyGroup()	Check if user is not in any group	true/false
HAS_UG_ATTR('attrName')	ctx.hasUgAttr	ctx.hasUgAttr('department')	Check if user's groups have specific attribute	true/false
JavaScript usage example: <pre>IS_IN_GROUP('finance') && GET_UG_ATTR('dept') == 'FINANCE'</pre>				

Table 11: Role-related macros

Macro	Expands to	JavaScript usage example	Description	Example return
GET_UR_NAMES()	ctx.urNames	ctx.urNames()	Get user's role names as CSV	"analyst,dba"
GET_UR_NAMES_Q()	ctx.urNamesQ	ctx.urNamesQ()	Get user's role names as quoted CSV	"'analyst','dba'"
UR_NAMES_CSV	ctx.urNamesCsv()	ctx.urNamesCsv()	Get role names as CSV	"analyst,dba"
UR_NAMES_Q_CSV	ctx.urNamesCsvQ()	ctx.urNamesCsvQ()	Get role names as quoted CSV	"'analyst','dba'"
IS_IN_ROLE('roleName')	ctx.isInRole	ctx.isInRole('data_scientist')	Check if user has specific role	true/false
IS_IN_ANY_ROLE	ctx.isInAnyRole()	ctx.isInAnyRole()	Check if user has any role	true/false
IS_NOT_IN_ANY_ROLE	!ctx.isInAnyRole()	!ctx.isInAnyRole()	Check if user has no roles	true/false
JavaScript usage example: <pre>IS_IN_ROLE('analyst') IS_IN_ROLE('admin')</pre>				

Table 12: Time-related macros

Macro	Expands to	JavaScript usage example	Description	Example return
IS_ACCESS_TIME_AFTER('dateTime')	ctx.isAccessTimeAfter	ctx.isAccessTimeAfter('2020/01/01') ctx.isAccessTimeAfter('09:00')	Check if access time is after specified time	IS_ACCESS_TIME_AFTER('2020/01/01')
IS_ACCESS_TIME_BEFORE('dateTime')	ctx.isAccessTimeBefore	ctx.isAccessTimeBefore('2025/01/01') ctx.isAccessTimeBefore('18:00')	Check if access time is before specified time	IS_ACCESS_TIME_BEFORE('2025/01/01')

Macro	Expands to	JavaScript usage example	Description	Example return
IS_ACCESS_TIME_BETWEEN('09:00','17:00')	ctx.isAccessTimeBetween('09:00','17:00')	ctx.isAccessTimeBetween('09:00','17:00')	Check if access time is within range (2023/01/01, 2023/12/31)	IS_ACCESS_TIME_BETWEEN('09:00','17:00')
JavaScript usage example: <pre>IS_ACCESS_TIME_BETWEEN('09:00', '17:00')</pre>				

Combined conditions

JavaScript usage example

```
HAS_TAG('SENSITIVE') && IS_IN_GROUP('privileged-users') && GET_USER_ATTR('clearanceLevel') >= GET_TAG_ATTR('sensitiveLevel')
```

Important notes

- Q suffix: Macros ending with Q return values enclosed in single quotes.
- CSV suffix: Macros ending with CSV return values separated by commas.
- Parameter format: Macros that require parameters use parentheses, for example, GET_TAG_ATTR('attrName').
- String parameter format: Always use single quotes for string parameters, such as ctx.tagAttr('attributeName').
- Case sensitivity: Attribute names and values are case-sensitive.
- Boolean returns: HAS_* and IS_* macros return true/false.
- Boolean logic: Combine conditions using &&, ||, and ! for complex conditions.
- Multiple values: If multiple tags or groups share the same attribute, the returned values are comma-separated.
- Array operations: To convert CSV values into arrays for advanced processing, use .split(',').
- Type conversion: For numeric comparisons, apply parseInt() or Number().

Best practices

- Ensure that the JavaScript conditions are straightforward and easy to test.
- Apply ctx.result=false to deny access and ctx.result=true to allow access, based on the specific condition type.
- Use macros and attributes to reduce hardcoding of values.
- Refrain from using intricate nested logic unless absolutely required.

Practical JavaScript usage examples

Ranger policy condition examples provide JavaScript snippets for implementing access control based on tags, user attributes, groups, roles, and time. Use these templates to enforce complex authorization logic, such as attribute-based comparisons and time-restricted data access.

Use case	Policy conditions
Tag-based policy conditions	
Check if resource has PII tag and sensitivity level is high	ctx.hasTag('PII') && parseInt(ctx.tagAttr('sensitiveLevel')) >= 10
Check if any tag has a specific owner	ctx.tagAttr('owner').split(',').includes('john.doe')
Verify user clearance against tag sensitivity	parseInt(ctx.userAttr('clearanceLevel')) >= Math.max(...ctx.tagAttr('sensitiveLevel').split(',').map(Number))
User attribute conditions	
Check user department and clearance	ctx.userAttr('department') === 'FINANCE' && parseInt(ctx.userAttr('clearanceLevel')) >= 5

Use case	Policy conditions
Verify user has required attributes	<code>ctx.hasUserAttr('clearanceLevel') && ctx.hasUserAttr('department')</code>
Group-based conditions	
Check if user is in privileged group with specific department attribute	<code>ctx.isInGroup('privileged_users') && ctx.ugAttr('department').split(',') .includes('SECURITY')</code>
Verify group attributes match requirements	<code>ctx.hasUgAttr('costCenter') && ctx.ugAttr('region') === 'US'</code>
Role-based Conditions	
Check if user has admin or analyst role	<code>ctx.isInRole('admin') ctx.isInRole('analyst')</code>
Ensure user has at least one role	<code>ctx.isInAnyRole()</code>
Time-based conditions	
Business hours access only	<code>ctx.isAccessTimeBetween('09:00', '17:00')</code>
Weekend restriction	<code>!ctx.isAccessTimeBetween('2023/04/01 00:00', '2023/04/02 23:59')</code>
Complex combined conditions	
Multi-factor authorization	<code>ctx.hasTag('SENSITIVE') && ctx.isInGroup('data_analysts') && parseInt(ctx.userAttr('clearanceLevel')) >= parseInt(ctx.tagAttr('sensitiveLevel')) && ctx.isAccessTimeBetween('08:00', '18:00')</code>
Department-based data access	<code>ctx.ugAttr('department').split(',').some(dept => ctx.tagAttr('allowedDepartments').split(',').includes(dept))</code>

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 displays the **Resource Policies** **Resource** page when you log in to the Ranger Admin Web UI. You can also access this page by selecting **Service Manager** **Resource Policies**, or by clicking the Ranger icon at the upper left of the Ranger Admin Web UI. You can use Resource to add, edit or delete services for Hadoop resources (HDFS, HBase, HadoopSQL, etc.) and add access policies for those resources.

- To add a new resource-based service, click Add () in the applicable box on Service Manager. Enter the required configuration settings, then click Add.
- To edit a resource-based service, click Edit () at the right of the service. Edit the service settings, then click Save to save your changes.
- To delete a resource-based service, click Delete () at the right of the service. Deleting a service also deletes all of the policies for that service.

Resource Policies

Tag Policies

Reports

Audits

Security Zone

Settings

admin

Service Manager

Last Response Time
09/19/2023 10:07:05 AM

Resource

Tag

Security Zone :
Select Zone Name

Import

Export

HDFS

cm_hdfs

+

🔍

✎

🗑

YARN

cm_yarn

+

🔍

✎

🗑

KAFKA

cm_kafka

+

🔍

✎

🗑

ATLAS

cm_atlas

+

🔍

✎

🗑

OZONE

cm_ozone

+

🔍

✎

🗑

S3

+

🔍

✎

🗑

HBASE

cm_hbase

+

🔍

✎

🗑

KNOX

cm_knox

+

🔍

✎

🗑

NIFI

+

🔍

✎

🗑

ADLS

+

🔍

✎

🗑

SCHEMA-REGISTRY

cm_schema-registry

+

🔍

✎

🗑

GS

+

🔍

✎

🗑

HADOOP SQL

Hadoop SQL

+

🔍

✎

🗑

SOLR

cm_solr

+

🔍

✎

🗑

NIFI-REGISTRY

+

🔍

✎

🗑

KUDU

cm_kudu

+

🔍

✎

🗑

KAFKA-CONNECT

cm_kafka_connect

+

🔍

✎

🗑

Configure a resource-based service: Atlas

How to add an Atlas service.

Procedure

1.
- On Service Manager Resource Policies , click Add New Service () next to Atlas.
The Create Service page appears.

Create Service

Last Response Time
09/26/2025 12:39:31 PM

[Service Manager](#) > Create Service

Service Details :

Service Name *

Display Name

Description

Active Status

☒ Enabled ☐ Disabled

Tag Service

Select Tag Service

Config Properties :

Username *

Password *

atlas.rest.address *

Common Name for Certificate

Policy Download Users

Select Users

Tag Download Users

Select Users

Service Admin Users

Select Users

Service Admin Groups

Select Groups

Superusers

Select Users

Superuser Groups

Select Groups

Userstore Download Users

Select Users

Add New Custom Configurations

Name	Value

Audit Filter : ☒

Is Audited	Access Result	Resources	Operations	Permissions	Users	Groups	Roles
Click on below + button to add audit filter !!							

+

Test Connection

Add

Cancel

2. On Create Service, enter the following information:

Table 13: Service Details

Field name	Description
Service Name	The name of the service; required when configuring agents.
Display Name	The name which appears on Service Manager.
Description	A description of the service.
Active Status	Enabled or Disabled.

Field name	Description
Tag Service	Select a tag-based service to apply the service and its tag-based policies to Atlas.

Table 14: Configuration 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 (SSL environment)	Atlas host and port. Update in the Ranger plugin services to get resource lookup working. Value is https://<Atlas-server>:<https-port>. Default port is 31443.
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.
Policy Download Users	Selected users can download policies in the service.
Tag Download Users	Selected users can download tags in the service.
Service Admin Users	Selected users can create/update/delete/read policies in the service.
Service Admin Groups	Users in the selected groups can create/update/delete/read policies in the service.
Superusers	The plugin grants all accesses on all resources to the selected users.
Superuser Groups	The plugin grants all accesses on all resources to users in the selected groups.
Userstore Download Users	Selected users can download user and group details.
Add New Configurations	Add any other new configuration(s).

3. Click Test Connection.
4. Click Add.

Configure a resource-based service: HBase

How to add an HBase service.

Procedure

1.
- On Service Manager Resource Policies , click Add New Service (
- Create Service

Last Response Time
09/26/2025 12:54:07 PM

[Service Manager](#) > Create Service

Service Details :

Service Name *

Display Name

Description

Active Status

☒ Enabled ☐ Disabled

Tag Service

Select Tag Service

Config Properties :

Username *

Password *

hadoop.security.authentication *

hbase.master.kerberos.principal

hbase.security.authentication *

hbase.zookeeper.property.clientPort *

hbase.zookeeper.quorum *

zookeeper.znode.parent *

Common Name for Certificate

Policy Download Users

Tag Download Users

Service Admin Users

Service Admin Groups

Superusers

Superuser Groups

Userstore Download Users

Add New Custom Configurations

Name	Value

Audit Filter : ☒

Is Audited	Access Result	Resources	Operations	Permissions	Users	Groups	Roles
Click on below  button to add audit filter !!							

Test Connection

Add

Cancel

2.

On Create Service, enter the following information:

Table 15: Service Details

Field name	Description
Service Name	The name of the service; required when configuring agents.

39

Field name	Description
Display Name	The name which will appear on Service Manager.
Description	A description of the service.
Active Status	Enabled or Disabled.
Tag Service	Select a tag-based service to apply the service and its tag-based policies to HBase.

Table 16: Configuration 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.authentication	Values are simple and kerberos.
hbase.master.kerberos.principal	The Kerberos principal for the HBase Master. (Required only if Kerberos authentication is enabled). Update in the Ranger plugin services to get resource lookup working. For example, hbase/_HOST@<REALM>.
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.
Policy Download Users	Selected users can download policies in the service.
Tag Download Users	Selected users can download tags in the service.
Service Admin Users	Selected users can create/update/delete/read policies in the service.
Service Admin Groups	Users in the selected groups can create/update/delete/read policies in the service.
Superusers	The plugin grants all accesses on all resources to the selected users.
Superuser Groups	The plugin grants all accesses on all resources to users in the selected groups.
Userstore Download Users	Selected users can download user and group details.
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 an HDFS service.

Procedure

1.
- On Service Manager Resource Policies , click Add New Service () next to HDFS.
- The Create Service page appears.

Create Service

Last Response Time
09/26/2025 01:00:32 PM

[Service Manager](#) > Create Service

Service Details :

Service Name *

Display Name

Description

Active Status

☒ Enabled ☐ Disabled

Tag Service

Select Tag Service

Config Properties :

Username *

Password *

Namenode URL *

Authorization Enabled *

Authentication Type *

hadoop.security.auth_to_local

dfs.datanode.kerberos.principal

dfs.namenode.kerberos.principal

dfs.secondary.namenode.kerberos.principal

RPC Protection Type

Common Name for Certificate

Policy Download Users

Tag Download Users

Service Admin Users

Service Admin Groups

Superusers

Superuser Groups

Userstore Download Users

Add New Custom Configurations

Name	Value

Audit Filter : ☒

Is Audited	Access Result	Resources	Operations	Permissions	Users	Groups	Roles
Click on below  button to add audit filter !!							

Test Connection

Add

Cancel

2. On Create Service, enter the following information:

Table 17: Service Details

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

Table 18: Configuration 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.
Policy Download Users	Selected users can download policies in the service.
Tag Download Users	Selected users can download tags in the service.

Field name	Description
Service Admin Users	Selected users can create/update/delete/read policies in the service.
Service Admin Groups	Users in the selected groups can create/update/delete/read policies in the service.
Superusers	The plugin grants all accesses on all resources to the selected users.
Superuser Groups	The plugin grants all accesses on all resources to users in the selected groups.
Userstore Download Users	Selected users can download user and group details.
Add New Configurations	Add any other new configuration(s).

3. Click Test Connection.
4. Click Add.

Configure a resource-based service: HadoopSQL

How to add a HadoopSQL service.

Procedure

1.
- On Service Manager Resource Policies , click Add New Service () next to HadoopSQL. Create Service appears.

Create Service

Last Response Time
09/26/2025 01:10:53 PM

[Service Manager](#) > Create Service

Service Details :

Service Name *

Display Name

Description

Active Status

☒ Enabled ☐ Disabled

Tag Service

Select Tag Service

Config Properties :

Username *

Password *

jdbc.driverClassName *

jdbc.url *

Common Name for Certificate

Policy Download Users

Select Users

Tag Download Users

Select Users

Service Admin Users

Select Users

Service Admin Groups

Select Groups

Superusers

Select Users

Superuser Groups

Select Groups

Userstore Download Users

Select Users

Add New Custom Configurations

Name	Value

Audit Filter : ☒

Is Audited	Access Result	Resources	Operations	Permissions	Users	Groups	Roles
Click on below + button to add audit filter !!							

+

Test Connection

AddCancel

2.
- On Create Service, enter the following information:

Table 19: Service Details

Field name	Description
Service Name	The name of the service; required when configuring agents.
Display Name	The name which will appear on Service Manager.
Description	A description of the service.

Field name	Description
Active Status	Enabled or Disabled.
Tag Service	Select a tag-based service to apply the service and its tag-based policies to Hive.

Table 20: Configuration 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
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.
Policy Download Users	Selected users can download policies in the service.
Tag Download Users	Selected users can download tags in the service.
Service Admin Users	Selected users can create/update/delete/read policies in the service.
Service Admin Groups	Users in the selected groups can create/update/delete/read policies in the service.
Superusers	The plugin grants all accesses on all resources to the selected users.
Superuser Groups	The plugin grants all accesses on all resources to users in the selected groups.
Userstore Download Users	Selected users can download user and group details.
Add New Configurations	Add any other new configuration(s).

3. Click Test Connection.
4. Click Add.

Configure a resource-based service: Kafka

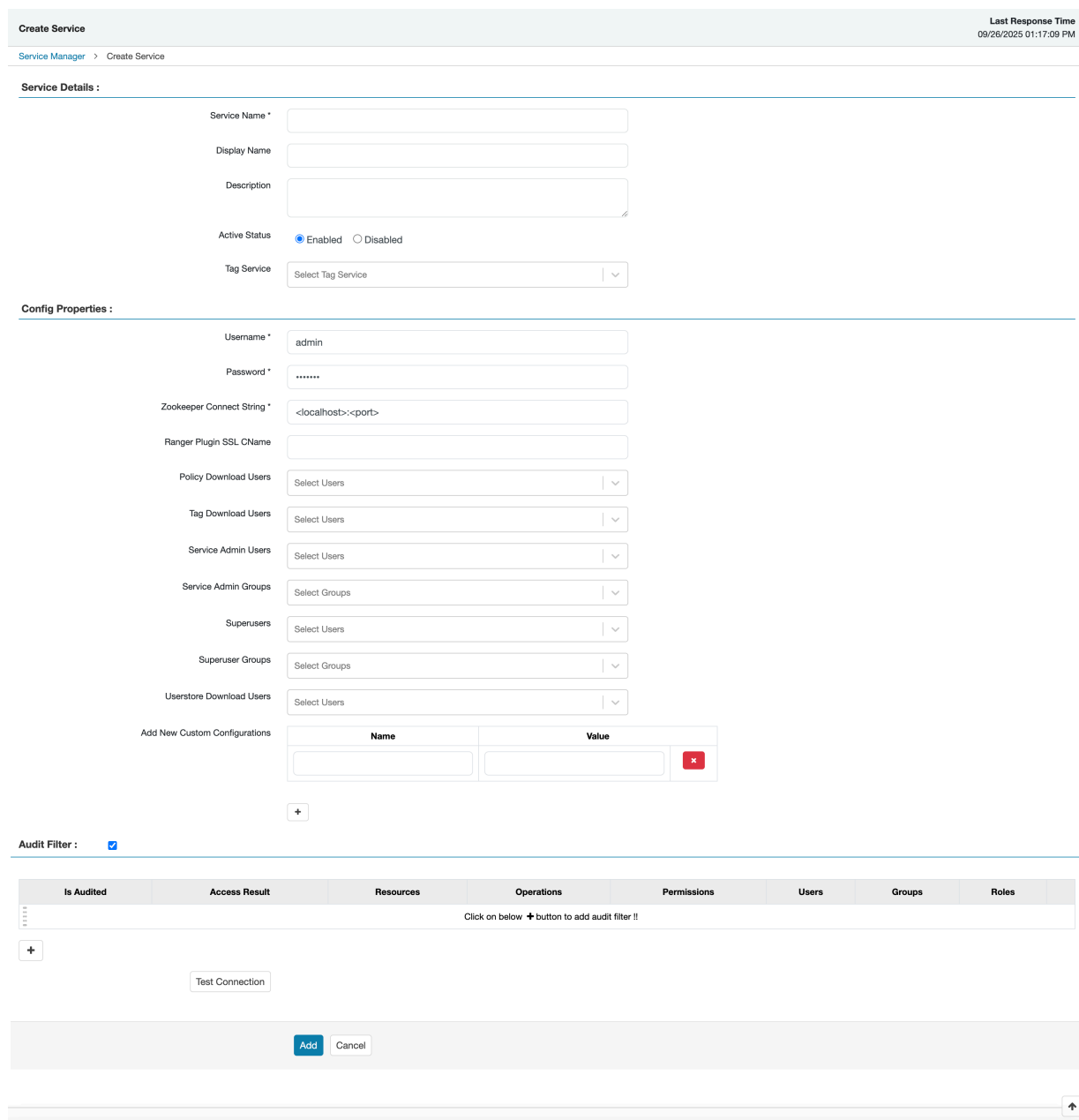
How to add a Kafka service.

Procedure

1.

On Service Manager Resource Policies, click Add New Service () next to Kafka.

The Create Service page appears.



Create Service Last Response Time
09/26/2025 01:17:09 PM

[Service Manager](#) > Create Service

Service Details :

Service Name *

Display Name

Description

Active Status ☒ Enabled ☐ Disabled

Tag Service

Config Properties :

Username *

Password *

Zookeeper Connect String *

Ranger Plugin SSL CName

Policy Download Users

Tag Download Users

Service Admin Users

Service Admin Groups

Superusers

Superuser Groups

Userstore Download Users

Add New Custom Configurations

Name	Value

Audit Filter : ☒

Is Audited	Access Result	Resources	Operations	Permissions	Users	Groups	Roles
Click on below + button to add audit filter !!							

2. On Create Service, enter the following information:

Table 21: Service Details

Field name	Description
Service Name	The name of the service; required when configuring agents.
Display Name	The name which will appear on Service Manager.
Description	A description of the service.

Field name	Description
Active Status	Enabled or Disabled.
Tag Service	Select a tag-based service to apply the service and its tag-based policies to Kafka.

Table 22: Configuration 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).
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.
Policy Download Users	Selected users can download policies in the service.
Tag Download Users	Selected users can download tags in the service.
Service Admin Users	Selected users can create/update/delete/read policies in the service.
Service Admin Groups	Users in the selected groups can create/update/delete/read policies in the service.
Superusers	The plugin grants all accesses on all resources to the selected users.
Superuser Groups	The plugin grants all accesses on all resources to users in the selected groups.
Userstore Download Users	Selected users can download user and group details.
Add New Configurations	Add any other new configuration(s).
For non-SSL environment + Kerberos - Update in the Ranger plugin services to get resource lookup working.	
bootstrap.servers	The value is host1:9092,host2:9092,host3:9092.
security.protocol	The value is SASL_PLAINTEXT.
sasl.mechanism	The value is GSSAPI.
kafka.keytab	The value is <path to keytab>.  Note: It should be present in all the Ranger admin nodes.
kafka.principal	The value is kafka@<REALM>.
For SSL environment + Kerberos - Update in the Ranger plugin services to get resource lookup working.	
bootstrap.servers	The value is host1:9093,host2:9093,host3:9093.
security.protocol	The value is SASL_SSL.
sasl.mechanism	The value is GSSAPI.
kafka.keytab	The value is <path to keytab>.  Note: It should be present in all the Ranger admin nodes.
kafka.principal	The value is kafka@<REALM>.

- 3. Click Test Connection.
- 4. Click Add.

Configure a resource-based service: Knox

How to add a Knox service.

Procedure

- 1. On Service Manager Resource Policies , click Add New Service () next to Knox. The Create Service page appears.

Create Service

Last Response Time
09/26/2025 01:22:38 PM

[Service Manager](#) > Create Service

Service Details :

Service Name *

Display Name

Description

Active Status

☒ Enabled ☐ Disabled

Tag Service

Select Tag Service

Config Properties :

Username *

Password *

knox.url *

Common Name for Certificate

Policy Download Users

Select Users

Tag Download Users

Select Users

Service Admin Users

Select Users

Service Admin Groups

Select Groups

Superusers

Select Users

Superuser Groups

Select Groups

Userstore Download Users

Select Users

Add New Custom Configurations

Name	Value

Audit Filter : ☒

Is Audited	Access Result	Resources	Operations	Permissions	Users	Groups	Roles
Click on below + button to add audit filter !!							

Test Connection

Add

Cancel

2. On Create Service, enter the following information:

Table 23: Service Details

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

Table 24: Configuration 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.
Policy Download Users	Selected users can download policies in the service.
Tag Download Users	Selected users can download tags in the service.
Service Admin Users	Selected users can create/update/delete/read policies in the service.
Service Admin Groups	Users in the selected groups can create/update/delete/read policies in the service.
Superusers	The plugin grants all accesses on all resources to the selected users.
Superuser Groups	The plugin grants all accesses on all resources to users in the selected groups.
Userstore Download Users	Selected users can download user and group details.
Add New Configurations	Add any other new configuration(s).

3. Click Test Connection.
4. Click Add.

Configure a resource-based service: NiFi

How to add a NiFi service.

Procedure

1.
- On Service Manager Resource Policies , click Add New Service () next to NiFi. The Create Service page appears.

Create Service

Last Response Time09/26/2025 01:24:53 PM

[Service Manager](#) > Create Service

Service Details :

Service Name *

Display Name

Description

Active Status

☒ Enabled ☐ Disabled

Tag Service

Select Tag Service

Config Properties :

NiFi URL *

Authentication Type *

Use Ranger's Default SSL Context *

Keystore

Keystore Type

PKCS12

Keystore Password

.....

Truststore

Truststore Type

Truststore Password

Policy Download Users

Select Users

Tag Download Users

Select Users

Service Admin Users

Select Users

Service Admin Groups

Select Groups

Superusers

Select Users

Superuser Groups

Select Groups

Userstore Download Users

Select Users

Add New Custom Configurations

Name	Value

Audit Filter :

Select "Audit Filter" to save/add audit filter !!

Test Connection

Add

Cancel

2.
- On Create Service, enter the following information:

Table 25: Service Details

Field name	Description
Service Name	The name of the service; required when configuring agents.
Display Name	The name which will appear on Service Manager.

Field name	Description
Description	A description of the service.
Active Status	Enabled or Disabled.
Tag Service	Select a tag-based service to apply the service and its tag-based policies to NiFi.

Table 26: Configuration Properties

Field name	Description
NiFi URL	The complete NiFi host URL.
Authentication Type	None or SSL.
Use Ranger's Default SSL Context	If true, then Ranger's keystore and truststore will be used to communicate with NiFi. If false, the keystore and truststore properties must be provided.
Keystore	The keystore to use when Ranger makes an https connection to NiFi. This keystore contains the certificate that represents the Ranger server.
Keystore Type	The keystore type (JKS or PKCS12).
Keystore Password	The keystore password.
Truststore	The truststore to use when Ranger makes an https connection to NiFi. This truststore contains the public key of the certificate authority that signed the NiFi server certificates.
Truststore Type	The truststore type (JKS or PKCS12).
Truststore Password	The truststore password.
Policy Download Users	Selected users can download policies in the service.
Tag Download Users	Selected users can download tags in the service.
Service Admin Users	Selected users can create/update/delete/read policies in the service.
Service Admin Groups	Users in the selected groups can create/update/delete/read policies in the service.
Superusers	The plugin grants all accesses on all resources to the selected users.
Superuser Groups	The plugin grants all accesses on all resources to users in the selected groups.
Userstore Download Users	Selected users can download user and group details.
Add New Configurations	Add any other new configuration(s).

3. Click Test Connection.
4. Click Add.

Configure a resource-based service: NiFi Registry

How to add a NiFi Registry service.

Procedure

1.
- On Service Manager Resource Policies , click Add New Service () next to NiFi Registry. The Create Service page appears.

Create Service

Last Response Time
09/26/2025 01:33:36 PM

[Service Manager](#) > Create Service

Service Details :

Service Name *

Display Name

Description

Active Status

☒ Enabled ☐ Disabled

Tag Service

Select Tag Service

Config Properties :

NiFi Registry URL *

Authentication Type *

Use Ranger's Default SSL Context *

Keystore

Keystore Type

PKCS12

Keystore Password

.....

Truststore

Truststore Type

Truststore Password

Policy Download Users

Select Users

Tag Download Users

Select Users

Service Admin Users

Select Users

Service Admin Groups

Select Groups

Superusers

Select Users

Superuser Groups

Select Groups

Userstore Download Users

Select Users

Add New Custom Configurations

Name	Value

Audit Filter :

Select "Audit Filter" to save/add audit filter !!

Test Connection

Add

Cancel

2.
- On Create Service, enter the following information:

Table 27: Service Details

Field name	Description
Service Name	The name of the service; required when configuring agents.
Display Name	The name which will appear on Service Manager.

Field name	Description
Description	A description of the service.
Active Status	Enabled or Disabled.
Tag Service	Select a tag-based service to apply the service and its tag-based policies to NiFi.

Table 28: Configuration Properties

Field name	Description
NiFi Registry URL	The complete NiFi Registry URL.
Authentication Type	None or SSL.
Use Ranger's Default SSL Context	If true, then Ranger's keystore and truststore will be used to communicate with NiFi Registry. If false, the keystore and truststore properties must be provided.
Keystore	The keystore to use when Ranger makes an https connection to the NiFi Registry. This keystore contains the certificate that represents the Ranger server.
Keystore Type	The keystore type (JKS or PKCS12).
Keystore Password	The keystore password.
Truststore	The truststore to use when Ranger makes an https connection to the NiFi Registry. This truststore contains the public key of the certificate authority that signed the NiFi server certificates.
Truststore Type	The truststore type (JKS or PKCS12).
Truststore Password	The truststore password.
Policy Download Users	Selected users can download policies in the service.
Tag Download Users	Selected users can download tags in the service.
Service Admin Users	Selected users can create/update/delete/read policies in the service.
Service Admin Groups	Users in the selected groups can create/update/delete/read policies in the service.
Superusers	The plugin grants all accesses on all resources to the selected users.
Superuser Groups	The plugin grants all accesses on all resources to users in the selected groups.
Userstore Download Users	Selected users can download user and group details.
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 Solr service.

Procedure

1.
- On Service Manager Resource Policies , click Add New Service () next to Solr.
The Create Service page appears.

Create Service

Last Response Time
09/26/2025 01:52:32 PM

[Service Manager](#) > Create Service

Service Details :

Service Name *

Display Name

Description

Active Status

☒ Enabled ☐ Disabled

Tag Service

Select Tag Service

Config Properties :

Username *

Password *

Solr Zookeeper Quorum

Solr URL *

Ranger Plugin SSL CName

Policy Download Users

Select Users

Tag Download Users

Select Users

Service Admin Users

Select Users

Service Admin Groups

Select Groups

Superusers

Select Users

Superuser Groups

Select Groups

Userstore Download Users

Select Users

Add New Custom Configurations

Name	Value

Audit Filter : ☒

Is Audited	Access Result	Resources	Operations	Permissions	Users	Groups	Roles
Click on below + button to add audit filter !!							

+

Test Connection

Add

Cancel

2.
- On Create Service, enter the following information:

Table 29: Service Details

Field name	Description
Service Name	The name of the service; required when configuring agents.
Display Name	The name which will appear on Service Manager.
Description	A description of the service.

Field name	Description
Active Status	Enabled or Disabled.
Tag Service	Select a tag-based service to apply the service and its tag-based policies to Solr.

Table 30: Configuration Properties

Field name	Description
Username	The end system username that can be used for connection.
Password	The password for the username entered above.
Solr Zookeeper Quorum	Solr-Zookeeper znode connection string used to store information about the Solr service where Ranger Solr plugin is enabled.
Solr URL	http://Solr_host:8983
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.
Policy Download Users	Selected users can download policies in the service.
Tag Download Users	Selected users can download tags in the service.
Service Admin Users	Selected users can create/update/delete/read policies in the service.
Service Admin Groups	Users in the selected groups can create/update/delete/read policies in the service.
Superusers	The plugin grants all accesses on all resources to the selected users.
Superuser Groups	The plugin grants all accesses on all resources to users in the selected groups.
Userstore Download Users	Selected users can download user and group details.
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 YARN service.

Procedure

1.
- On Service Manager Resource Policies , click Add New Service () next to YARN.
- The Create Service page appears.

Create Service

Last Response Time
09/26/2025 02:27:46 PM

[Service Manager](#) > Create Service

Service Details :

Service Name *

Display Name

Description

Active Status ☒ Enabled ☐ Disabled

Tag Service

Select Tag Service

Config Properties :

Username *

Password *

YARN REST URL *

Authentication Type

Simple

Common Name for Certificate

Policy Download Users

Select Users

Tag Download Users

Select Users

Service Admin Users

Select Users

Service Admin Groups

Select Groups

Superusers

Select Users

Superuser Groups

Select Groups

Userstore Download Users

Select Users

Add New Custom Configurations

Name	Value

Audit Filter : ☐

Select "Audit Filter" to save/add audit filter !!

Test Connection

Add

Cancel

2.
- On Create Service, enter the following information:

Table 31: Service Details

Field name	Description
Service Name	The name of the service; required when configuring agents.
Display Name	The name which will appear on Service Manager.
Description	A description of the service.
Active Status	Enabled or Disabled.

Field name	Description
Tag Service	Select a tag-based service to apply the service and its tag-based policies to YARN.

Table 32: Configuration 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://<RESOURCEMANAGER_FQDN>:<http-port> https://<RESOURCEMANAGER_FQDN>:<https-port>
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.
Policy Download Users	Selected users can download policies in the service.
Tag Download Users	Selected users can download tags in the service.
Service Admin Users	Selected users can create/update/delete/read policies in the service.
Service Admin Groups	Users in the selected groups can create/update/delete/read policies in the service.
Superusers	The plugin grants all accesses on all resources to the selected users.
Superuser Groups	The plugin grants all accesses on all resources to users in the selected groups.
Userstore Download Users	Selected users can download user and group details.
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 **Service Manager** **Resource Policies** . List of Policies displays the list of existing policies for that service, 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 Edit () for the policy. Edit the policy settings, then click Save to save your changes.

- To delete a resource-based policy, click Delete () for the policy.

HBASE Policies

Service : cm_hbase

Select Zone Name

Manage Service

Last Response Time
09/26/2023 09:25:51 AM

Search for your policy...

Add New Policy

Policy ID	Policy Name	Policy Label	Status	Audit Logging	Roles	Groups	Users	Actions
5	all - table, column-family, column	--	Enabled	Enabled	--	--	hbase rangerlookup	
6	ATLAS_HBASE_TABLES	--	Enabled	Enabled	--	--	atlas	Edit
59	grant-1695052642270	--	Enabled	Enabled	--	--	atlas	
60	grant-1695052642467	--	Enabled	Enabled	--	--	atlas	Delete

Related Information

Importing and exporting resource-based policies

Configure a resource-based policy: Atlas

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

Procedure

1. On Service Manager, select an existing Atlas service.
List of Policies displays a list of the policies defined for Atlas service.
2. Click Add New Policy.
Create Policy displays controls for creating details for a new policy.

Create Policy

Last Response Time
09/26/2023 10:33:47 AM

Service Manager

cm_atlas Policies

Create Policy

Policy Details

Policy Type: Access

Policy Name*: Policy Name

Policy Label: Select...

Type Category: Select...

Type Category: Entity Type, Atlas Service, Relationship Type

Audit Logging*: Yes

Enabled, Normal, Include, Include

Add Validity Period

Allow Conditions:

Select Roles: Select...

Select Groups: Select...

Select Users: Select...

Permissions: Add Permissions

Delegate Admin: ☐

☐

3. Complete the Create Policy page as follows:

Table 33: Policy Details

Field	Description
Policy Name	Enter an appropriate policy name. This name cannot be duplicated across the system. This field is mandatory.

Field	Description
normal/override	Enables you to specify an override policy. When override is selected, the access permissions in the policy override the access permissions in existing policies. This feature can be used with Add Validity Period to create temporary access policies that override existing policies.
type-category	Select type-category, entity-type, atlas-service, or relationship-type.
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.
Add Validity Period	Specify a start and end time for the policy.

Table 34: Allow Conditions

Label	Description
Select Role	Specify the roles to which this policy applies. To designate a role as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Select Group	Specify the groups to which this policy applies. To designate a group as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy. The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify the users to which this policy applies. To designate a user as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Permissions	Add or edit permissions: Create Type, Update Type, Delete Type, Select/Deselect All.
Delegate Admin	You can use Delegate Admin to assign administrator privileges to the roles, groups, or users specified in the policy. Administrators can edit or delete the policy, and can also create child policies based on the original policy.

4. You can use + 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. You can use Deny All Other Accesses to deny access to all other users, groups, and roles other than those specified in the allow conditions for the policy.
Combination of Deny All Other Accesses and Ranger macros (for example, {USER}) are not supported.
6. Click Add.

Related Information

[Wildcards and macros in resource-based policies](#)

Configure a resource-based policy: HBase

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

Procedure

1. On Service Manager, select an existing HBase service.
List of Policies displays a list of the policies defined for Hbase service.
2. Click Add New Policy.

Create Policy displays controls for creating details for a new policy.

Create Policy Last Response Time
09/26/2023 10:20:35 AM

Service Manager > cm_hbase Policies > Create Policy

Policy Details

Policy Type: **Access** Add Validity Period

Policy Name*: Enabled ☒ Normal ☐

Policy Label:

HBase Table *: Include ☒

HBase Column-family *: Include ☒

HBase Column *: Include ☒

Description:

Audit Logging*: ☒ Yes

Allow Conditions: hide

Select Roles	Select Groups	Select Users	Permissions	Delegate Admin	
Select...	Select...	Select...	Add Permissions +	☐	☒

+

3. Edit fields on Create Policy, as follows:

Table 35: Policy Details

Label	Description
Policy Name	Enter an appropriate policy name. This name cannot be duplicated across the system. This field is mandatory.
normal/override	Enables you to specify an override policy. When override is selected, the access permissions in the policy override the access permissions in existing policies. This feature can be used with Add Validity Period to create temporary access policies that override existing policies.
HBase Table	Select the appropriate database. Multiple databases can be selected for a particular policy. This field is mandatory.  Note: Policies must use '<namespace>:*' for the table name to apply to the namespace as a whole. You can define a namespace in the HBase table field. Valid formats for a namespace-specific, HBase policy include: <namespace>:<table> for example, "myNameSpace:table1" Further, note that <namespace>:<tablePrefix>* (default value) does not work, per https://issues.apache.org/jira/browse/RANGER-1226. All other namespaces except the default one work.
HBase Column-family	For the selected table, specify the column families to which the policy applies.

Label	Description
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.
Add Validity Period	Specify a start and end time for the policy.

Table 36: Allow Conditions

Label	Description
Select Role	Specify the roles to which this policy applies. To designate a role as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Select Group	Specify the groups to which this policy applies. To designate a group as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy. The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify the users to which this policy applies. To designate a user as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Permissions	Add or edit permissions: Read, Write, Create, Admin, Select/ Deselect All.
Delegate Admin	You can use Delegate Admin to assign administrator privileges to the roles, groups, or users specified in the policy. Administrators can edit or delete the policy, and can also create child policies based on the original policy.

4. You can use + 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. You can use Deny All Other Accesses to deny access to all other users, groups, and roles other than those specified in the allow conditions for the policy.
Combination of Deny All Other Accesses and Ranger macros (for example, {USER}) are not supported.
6. 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 macros 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 using [Service Manager Resource Policies](#), and to have restrictive permissions at the HDFS level.

Procedure

1. On [Service Manager Resource Policies](#), select an existing HDFS service.

List of Policies displays a list of the policies defined for HDFS service.

2. Click Add New Policy.

Create Policy displays controls for creating details for a new policy.

The screenshot shows the 'Create Policy' interface. At the top, there's a breadcrumb trail: 'Service Manager > cm_hdfs Policies > Create Policy'. The 'Policy Details' section contains the following elements:

- Policy Type:** A button labeled 'Access'.
- Policy Name*:** A text input field with the placeholder 'Policy Name'.
- Policy Label:** A dropdown menu with 'Select...' as the placeholder.
- Resource Path*:** A dropdown menu with 'Select...' as the placeholder.
- Description:** A large text area for describing the policy.
- Audit Logging*:** A toggle switch set to 'Yes'.
- Enabled/Normal:** Two radio buttons, with 'Enabled' selected.
- Recursive:** A toggle switch set to 'Recursive'.
- Add Validity Period:** A button with a clock icon.

Below the form, the 'Allow Conditions' section is visible. It includes a table with columns for 'Select Roles', 'Select Groups', and 'Select...'. A dropdown menu is open, showing options: 'Read', 'Write', 'Execute', and 'Select All'. To the right, there's a table with columns for 'Permissions', 'Delegate Admin', and a red 'X' button. The 'Permissions' column has a blue '+' button and the text 'Add Permissions...'. The 'Delegate Admin' column has a checkbox.

3. Complete the Create Policy page as follows:

Table 37: Policy Details

Field	Description
Policy Name	Enter a unique name for this policy. The name cannot be duplicated anywhere in the system.
normal/override	Enables you to specify an override policy. When override is selected, the access permissions in the policy override the access permissions in existing policies. This feature can be used with Add Validity Period to create temporary access policies that override existing policies.
Resource Path	Define the resource path for the policy folder/file. The default Recursive setting specifies that the resource path is recursive; you can also specify a non-recursive path.
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.
Add Validity Period	Specify a start and end time for the policy.

Table 38: Allow Conditions

Label	Description
Select Role	Specify the roles to which this policy applies. To designate a role as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.

Label	Description
Select Group	Specify the groups to which this policy applies. To designate a group as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy. The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify the users to which this policy applies. To designate a user as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Permissions	Add or edit permissions: Read, Write, Execute, Select/Deselect All.
Delegate Admin	You can use Delegate Admin to assign administrator privileges to the roles, groups, or users specified in the policy. Administrators can edit or delete the policy, and can also create child policies based on the original policy.

4. You can use + 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. You can use Deny All Other Accesses to deny access to all other users, groups, and roles other than those specified in the allow conditions for the policy.
Combination of Deny All Other Accesses and Ranger macros (for example, {USER}) are not supported.
6. Click Add.

Related Information

[Wildcards and macros in resource-based policies](#)

Configure a resource-based policy: HadoopSQL

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

Procedure

1. On Service Manager Resource Policies , select an existing HadoopSQL service.

List of Policies displays a list of the policies defined for HadoopSQL service.



Note: Service_name remains cm_hive. Display name is HadoopSQL.

2. Click Add New Policy.

Create Policy displays controls for creating details for a new policy.

The screenshot shows the 'Create Policy' interface. At the top, there's a breadcrumb trail: 'Service Manager > Hadoop SQL Policies > Create Policy'. The 'Policy Details' section includes a 'Policy Type' dropdown set to 'Access'. Below it are input fields for 'Policy Name*', 'Policy Label', 'Hive Database', 'Hive Table', and 'Hive Column', each with an 'Include' button. A 'Description' text area and an 'Audit Logging*' toggle (set to 'Yes') are also present. The 'Allow Conditions' section has 'Select Roles' and 'Select Groups' dropdowns. On the right, a 'Permissions' table is visible with columns 'Permissions', 'Delegate Admin', and 'Add Permissions'. A dropdown menu is open, listing various permissions like 'select', 'update', 'Create', etc.

3. Complete the Create Policy page as follows:

Table 39: 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.
normal/override	Enables you to specify an override policy. When override is selected, the access permissions in the policy override the access permissions in existing policies. This feature can be used with Add Validity Period to create temporary access policies that override existing policies.
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/udf	Specifies a table-based or UDF-based policy. Select table or udf, then type in the applicable table or 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 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.

Field	Description
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.
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/*
Description	(Optional) Describe the purpose of the policy.
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.
Add Validity Period	Specify a start and end time for the policy.

Table 40: Allow Conditions

Label	Description
Select Role	Specify the roles to which this policy applies. To designate a role as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Select Group	Specify the groups to which this policy applies. To designate a group as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy. The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify the users to which this policy applies. To designate a user as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Permissions	Add or edit permissions: Select, Update, Create, Drop, Alter, Index, Lock, All, ReplAdmin, Service Admin, Temp UDF Admin, Refresh, RW Storage, Select/Deselect All. Service Admin is used in conjunction with Hive Service Name and the kill query API: kill query <queryID> .

Label	Description
Delegate Admin	You can use Delegate Admin to assign administrator privileges to the roles, groups, or users specified in the policy. Administrators can edit or delete the policy, and can also create child policies based on the original policy.

- You can use + 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.
- You can use Deny All Other Accesses to deny access to all other users, groups, and roles other than those specified in the allow conditions for the policy.
Combination of Deny All Other Accesses and Ranger macros (for example, {USER}) are not supported.
- 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 macros in resource-based policies](#)

Configure a resource-based storage handler policy: HadoopSQL

How to configure a policy that allows authorized users to create data tables using storage-handlers.

About this task

Ranger includes “storage-type” and “storage-url” resources in HadoopSQL Service that support only the “RW Storage” permission. Ranger authorizes users to create or alter tables against this resource policy. Users are allowed to create/alter the table in the respective storage if they have the required “RW Storage” permission on the resource representing the storage-type and storage-url .

Procedure

- 1. On Service Manager Resource Policies , select an existing HadoopSQL service.
List of Policies displays a list of the policies defined for HadoopSQL service.



Note: Service_name remains cm_hive. Display name is HadoopSQL.

- 2. Select Add New Policy to create a new policy.
 - a) Within Create Policy, select storage-type as shown in the following example:

Create Policy

Last Response Time
09/26/2023 11:34:47 AM

Service Manager > Hadoop SQL Policies > Create Policy

Policy Details

Policy Type

Access

Policy Name*

test storage handler policy

Enabled

Normal

Policy Label

Select...

Storage Type

Select...

Hive Database

URL

Hive Service

Global

Storage Type

Include

ge handler policy for HadoopSQL

Add Validity Period

Allow Conditions:

hide

Select Roles	Select Groups	Select Users	Permissions	Delegate Admin
Select...	Select...	hive beacon dpprofiler hue impala admin	RW Storage	✓ ✗

- b) Complete the required* fields.
 - c) Under Allow Conditions, select users and add the RW Storage permission, as shown in the preceeding example.
 - d) Scroll to the bottom of Create Policy, then click Add.

- To configure an existing policy named all - storage-type, storage-url, click Edit.

The Edit Policy page appears.

Edit Policy Last Response Time
11/29/2023 03:22:15 PM

Service Manager > Hadoop SQL Policies > Edit Policy

Policy ID* 11

Policy Name* Enabled Normal

Policy Label

Storage Type Include

Storage URL* Include

Description

Audit Logging* Yes

Allow Conditions: hide ^

Select Roles	Select Groups	Select Users	Permissions	Delegate Admin	
Select...	Select...	hive beacon dpprofiler hue admin impala systest	RW Storage x	☒	x

+

- Complete the Edit Policy page as shown in the preceding example using the follow policy detail descriptions:

Table 41: 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.
normal/override	Enables you to specify an override policy. When override is selected, the access permissions in the policy override the access permissions in existing policies. This feature can be used with Add Validity Period to create temporary access policies that override existing policies.
Policy Label	Specify a label for this policy. You can search reports and filter policies based on these labels.
storage-type	Type in the applicable storage type. * allows authorizes users to create any table in the specified storage type..
storage url	Type in the applicable storage URL. * allows authorizes users to create any table in the specified storage URL. Select Exclude to deny access.
Description	(Optional) Describe the purpose of the policy.
Audit Logging	Specify whether this policy is audited. (De-select to disable auditing).

Field	Description
Add Validity Period	Specify a start and end time for the policy.

Table 42: Allow Conditions

Label	Description
Select User	Specify the users to which this policy applies. To designate a user as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Permissions	Add or edit permissions: RW Storage, You can assign read and select permissions to rangerlookup user.
Delegate Admin	You can use Delegate Admin to assign administrator privileges to the roles, groups, or users specified in the policy. Administrators can edit or delete the policy, and can also create child policies based on the original policy.

Example

Example StorageHandler Policy Definitions and Use Cases:

HBase StorageHandler policy:

Storage Type: hbase

Storage URL: hbase-cluster:port/hbase-table

Storage create table command:

```
CREATE [EXTERNAL] table foo(...)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
TBLPROPERTIES ('hbase.table.name' = 'bar');
```

```
e.g:
CREATE TABLE hive_hbase_test_1(key int, value string) STORED BY
'org.apache.hadoop.hive.hbase.HBaseStorageHandler' WITH SERDEPR
PERTIES ("hbase.columns.mapping" = "cf:string", "hbase.table.name" = "hbase_test_1");
```

Iceberg StorageHandler policy:

Storage type: iceberg

Storage URL: DBname/Table* , or

Storage URL: DBname/*

JDBC StorageHandler policy:

Storage Type: jdbc:mysql

Storage URL: mysql-host:port/DBname/Table , or

Storage URL: mysql-host:port/DBname/*

**Note:**

Policy and table definitions must be in sync regarding the port definition, even for default port numbers. For example, if port number 3306 is defined in the policy for mysql and this port number is left out from the URL as default value for the JDBC Driver, you must use the same reference as defined in the policy when creating the external table.

Using an explicit table name allows only to reference that specific table with `hive.sql.table` while using `*` allows not only to reference any tables from the database but also allows you to write a custom query against this database, for example using `hive.sql.query`.

Storage create table command:

```
CREATE [EXTERNAL] TABLE student_jdbc
(
  name string,
  age int,
  gpa double
)
STORED BY 'org.apache.hive.storage.jdbc.JdbcStorageHandler'
TBLPROPERTIES (
  "hive.sql.database.type" = "MYSQL",
  "hive.sql.jdbc.driver" = "com.mysql.jdbc.Driver",
  "hive.sql.jdbc.url" = "jdbc:mysql://localhost/sample",
  "hive.sql.dbcp.username" = "hive",
  "hive.sql.dbcp.password" = "hive",
  "hive.sql.table" = "STUDENT",
  "hive.sql.dbcp.maxActive" = "1"
);
```

Kafka StorageHandler policy:

Storage Type: kafka

Storage URL: bootstrap-server:port/kafka-topic

Phoenix StorageHandler policy:

Storage Type: phoenix

Storage URL: phoenix-cluster:port/table-name

Storage create table command:

```
CREATE [EXTERNAL] TABLE phoenix_table (
  s1 string,
  i1 int,
  f1 float,
  d1 double
)
STORED BY 'org.apache.phoenix.hive.
PhoenixStorageHandler'
TBLPROPERTIES (
  "phoenix.table.name" = "phoenix_table",
  "phoenix.zookeeper.quorum" = "localhost",
  "phoenix.zookeeper.znode.parent" = "/hbase",
  "phoenix.zookeeper.client.port" = "2181",
  "phoenix.rowkeys" = "s1, i1",
  "phoenix.column.mapping" = "s1:s1, i1:i1, f1:f1, d1:d1",
```

```
TS=10, DATA_BLOCK_ENCODING='DIFF' "phoenix.table.options" = "SALT_BUCKE
);
```

Configure a resource-based policy: Kafka

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

Procedure

1. On Service Manager Resource Policies, select an existing Kafka service.

List of Policies displays a list of the policies defined for Kafka service.

2. Click Add New Policy.

Create Policy displays controls for creating details for a new policy.

The screenshot shows the 'Create Policy' interface. The 'Policy Details' section includes a 'Policy Type' dropdown set to 'Access'. The 'Policy Name' field is labeled 'Policy Name'. The 'Policy Label' field is a dropdown menu. The 'Topic' field is a dropdown menu with a list of options: Transactional Id, Cluster, Delegation Token, and Consumer Group. There are toggle switches for 'Enabled' (selected) and 'Normal', and an 'Include' toggle. A 'Policy Conditions' section shows 'No Conditions'. At the bottom, there are sections for 'Select Roles', 'Select Groups', 'Select Users', 'Policy Conditions', 'Permissions', 'Delegate Admin', and a 'hide' button.

3. Complete the Create Policy page as follows:

Table 43: Policy Details

Field	Description
Policy Name	Enter an appropriate policy name. This name cannot be duplicated across the system. This field is mandatory.
normal/override	Enables you to specify an override policy. When override is selected, the access permissions in the policy override the access permissions in existing policies. This feature can be used with Add Validity Period to create temporary access policies that override existing policies.
Policy Label	Specify a label for this policy. You can search reports and filter policies based on these labels.
Topic	Kafka resource type. A topic is a category or feed name to which messages are published.

Field	Description
Transactional ID	Kafka resource type, uniquely identifies producers in a persistent way.
Cluster	Kafka resource type.
Delegation Token	Kafka resource type for authentication.
Description	(Optional) Describe the purpose of the policy.
Audit Logging	Specify whether this policy is audited. (De-select to disable auditing).
Add Validity Period	Specify a start and end time for the policy.
Policy Conditions (applied at the policy level)	Click +, then specify an IP address range.

Table 44: Allow Conditions

Label	Description
Select Role	Specify the roles to which this policy applies. To designate a role as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Select Group	Specify the groups to which this policy applies. To designate a group as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy. The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify the users to which this policy applies. To designate a user as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Policy Conditions (applied at the item level)	Specify an IP address range.
Permissions	Add or edit permissions: Publish, Consume, Configure, Describe, Create, Delete, Describe Configs, Alter Configs, Select/Deselect All.
Delegate Admin	You can use Delegate Admin to assign administrator privileges to the roles, groups, or users specified in the policy. Administrators can edit or delete the policy, and can also create child policies based on the original policy.

- You can use + 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.
- You can use Deny All Other Accesses to deny access to all other users, groups, and roles other than those specified in the allow conditions for the policy.
Combination of Deny All Other Accesses and Ranger macros (for example, {USER}) are not supported.
- Click Add.

Related Information

[Wildcards and macros in resource-based policies](#)

Configure a resource-based policy: Knox

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

Procedure

1. On Service Manager Resource Policies , select an existing Knox service.

List of Policies displays a list of the policies defined for Knox service.

2. Click Add New Policy.

Create Policy displays controls for creating details for a new policy.

Create Policy Last Response Time
09/26/2023 02:02:09 PM

Service Manager > cm_knox Policies > Create Policy

Policy Details

Policy Type: **Access** Add Validity Period

Policy Name*: Enabled ☒ Normal ☐

Policy Label:

Knox Topology *: Include ☒

Knox Service *: Include ☒

Description:

Audit Logging*: **Yes** ☒

Policy Conditions : +

No Conditions

Allow Conditions: hide

Select Roles	Select Groups	Select Users	Policy Conditions	Permissions	Delegate Admin
Select...	Select...	Select...	Add Conditions +	Add Permissions +	☐ ×

3. Complete the Create Policy page as follows:

Table 45: Policy Details

Field	Description
Policy Name	Enter an appropriate policy name. This name cannot be duplicated across the system. This field is mandatory.
normal/override	Enables you to specify an override policy. When override is selected, the access permissions in the policy override the access permissions in existing policies. This feature can be used with Add Validity Period to create temporary access policies that override existing policies.
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).
Policy Label	Specify a label for this policy. You can search reports and filter policies based on these labels.
Add Validity Period	Specify a start and end time for the policy.

Field	Description
Policy Conditions (applied at the policy level)	Click the + icon, then enter an IP address range using Classless Inter-Domain Routing (CIDR) notation (for example 192.168.0.0/24) or a single IP address (for example 192.168.0.100).

Table 46: Allow Conditions

Label	Description
Select Role	Specify the roles to which this policy applies. To designate a role as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Select Group	Specify the groups to which this policy applies. To designate a group as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy. The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify the users to which this policy applies. To designate a user as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Policy Conditions (applied at the item level)	Specify an IP address range.
Permissions	Add or edit permissions: Allow
Delegate Admin	You can use Delegate Admin to assign administrator privileges to the roles, groups, or users specified in the policy. Administrators can edit or delete the policy, and can also create child policies based on the original 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.

4. You can use + 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. You can use Deny All Other Accesses to deny access to all other users, groups, and roles other than those specified in the allow conditions for the policy.

Combination of Deny All Other Accesses and Ranger macros (for example, {USER}) are not supported.

6. Click Add.

Related Information

[Wildcards and macros in resource-based policies](#)

Configure a resource-based policy: Kudu

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

Procedure

1. On **Service Manager Resource Policies**, select an existing Kudu service.
List of Policies displays a list of the policies defined for Kudu service.

2. Click Add New Policy.
- Create Policy displays controls for creating details for a new policy.

Resource Policies

Tag Policies

Reports

Audits

Security Zone

Settings

Create Policy

Last Response Time
08/23/2024 04:08:16 PM

Service Manager > cm_kudu Policies > Create Policy

Policy Details

Policy Type

Access

Policy Name*

Policy Name

Enabled

Normal

Policy Label

Select...

Include

Database *

Select...

Include

Table

Select...

Include

Column

Select...

Include

Description

Audit Logging*

Yes

Add Validity Period

Allow Conditions:

hide

Select Roles

Select Groups

Select Users

Permissions

Delegate Admin

Select...

Select...

Select...

Add Permissions

3. Complete the Create Policy page as follows:

Table 47: Policy Details

Field	Description
Policy Name	Enter an appropriate policy name. This name cannot be duplicated across the system. This field is mandatory.
normal/override	Enables you to specify an override policy. When override is selected, the access permissions in the policy override the access permissions in existing policies. This feature can be used with Add Validity Period to create temporary access policies that override existing policies.
Policy Label	(Optional) Specify a label for this policy. You can search reports and filter policies based on these labels.
Database	Kudu resource type. A Kudu Database to which this policy applies.
Table/None	Kudu resource type to indentify the table to which this policy applies. Select None if the policy is needed for all the tables in the database.
Column	Kudu resource type to identify the column to which this policy applies. Select None if the policy is needed for all the table/tables.
Description	(Optional) Describe the purpose of the policy.
Audit Logging	Specify whether this policy is audited. (De-select to disable auditing).
Add Validity Period	Specify a start and end time for the policy.

Table 48: Allow Conditions

Label	Description
Select Role	Specify the roles to which this policy applies. To designate a role as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.

Label	Description
Select Group	Specify the groups to which this policy applies. To designate a group as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy. The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify the users to which this policy applies. To designate a user as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Permissions	Add or edit permissions: Select, Insert, Update, Delete, Alter, create, Drop, Metadata, All, Select All.
Delegate Admin	You can use Delegate Admin to assign administrator privileges to the roles, groups, or users specified in the policy. Administrators can edit or delete the policy, and can also create child policies based on the original policy.

- You can use + 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.
- You can use Deny All Other Accesses to deny access to all other users, groups, and roles other than those specified in the allow conditions for the policy.
Combination of Deny All Other Accesses and Ranger macros (for example, {USER}) are not supported.
- Click Add.

Configure a resource-based policy: NiFi

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

Procedure

- On **Service Manager** **Resource Policies**, select an existing NiFi service.
List of Policies displays a list of the policies defined for NiFi service.

2. Click Add New Policy.

Create Policy displays controls for creating details for a new policy.

Create Policy

Last Response Time
09/26/2023 02:20:53 PM

[Service Manager](#) > [NiFi Policies](#) > Create Policy

Policy Details

Policy Type

Access

Add Validity Period

Policy Name*

Policy Name

Enabled

Normal

Policy Label

Select...

NiFi Resource Identifier *

Select...

Description

Audit Logging*

Yes

Allow Conditions:

hide

Select Roles	Select Groups	Select Users	Permissions	Delegate Admin	
Select...	Select...	Select...	Add Permissions		
			+		x

+

Save

Cancel

3. Complete the Create Policy page as follows:

Table 49: Policy Details

Field	Description
Policy Name	Enter an appropriate policy name. This name cannot be duplicated across the system. This field is mandatory.
normal/override	Enables you to specify an override policy. When override is selected, the access permissions in the policy override the access permissions in existing policies. This feature can be used with Add Validity Period to create temporary access policies that override existing policies.
NiFi Resource Identifier	In a NiFi cluster, all nodes must be granted the ability to view and modify component data in order for user to list or empty queues in processor component outbound connections. With Ranger this can be accomplished by using a wildcard to grant all of the NiFi nodes read and write access to the /data/* NiFi resource.
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.

Field	Description
Add Validity Period	Specify a start and end time for the policy.

Table 50: Allow Conditions

Label	Description
Select Role	Specify the roles to which this policy applies. To designate a role as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Select Group	Specify the groups to which this policy applies. To designate a group as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy. The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify the users to which this policy applies. To designate a user as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Permissions	Add or edit permissions: Read, Write, Select/Deselect All.
Delegate Admin	You can use Delegate Admin to assign administrator privileges to the roles, groups, or users specified in the policy. Administrators can edit or delete the policy, and can also create child policies based on the original policy.

- You can use + 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.
- You can use Deny All Other Accesses to deny access to all other users, groups, and roles other than those specified in the allow conditions for the policy.
Combination of Deny All Other Accesses and Ranger macros (for example, {USER}) are not supported.
- Click Add.

Configure a resource-based policy: NiFi Registry

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

Procedure

- On **Service Manager** **Resource Policies**, select an existing NiFi Registry service.
List of Policies displays a list of the policies defined for NiFi Registry service.

2. Click Add New Policy.

Create Policy displays controls for creating details for a new policy.

Create Policy

Last Response Time
09/26/2023 02:31:17 PM

[Service Manager](#) > [NiFi Registry Policies](#) > Create Policy

Policy Details

Policy Type

Access

Policy Name*

Policy Name

Enabled

Normal

Policy Label

Select...

NiFi Registry Resource Identifier *

Select...

Required

Description

Audit Logging*

Yes

Add Validity Period

Allow Conditions:

hide

Select Roles	Select Groups	Select Users	Permissions	Delegate Admin	
Select...	Select...	Select...	Add Permissions +	☐	×

+

3. Complete the Create Policy page as follows:

Table 51: Policy Details

Field	Description
Policy Name	Enter an appropriate policy name. This name cannot be duplicated across the system. This field is mandatory.
normal/override	Enables you to specify an override policy. When override is selected, the access permissions in the policy override the access permissions in existing policies. This feature can be used with Add Validity Period to create temporary access policies that override existing policies.
NiFi Registry Resource Identifier	In a NiFi cluster, all nodes must be granted the ability to view and modify component data in order for user to list or empty queues in processor component outbound connections. With Ranger this can be accomplished by using a wildcard to grant all of the NiFi nodes read and write access to the /data/* NiFi resource.
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.

Field	Description
Add Validity Period	Specify a start and end time for the policy.

Table 52: Allow Conditions

Label	Description
Select Role	Specify the roles to which this policy applies. To designate a role as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Select Group	Specify the groups to which this policy applies. To designate a group as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy. The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify the users to which this policy applies. To designate a user as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Permissions	Add or edit permissions: Read, Write, Delete, Select/Deselect All.
Delegate Admin	You can use Delegate Admin to assign administrator privileges to the roles, groups, or users specified in the policy. Administrators can edit or delete the policy, and can also create child policies based on the original policy.

- You can use + 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.
- You can use Deny All Other Accesses to deny access to all other users, groups, and roles other than those specified in the allow conditions for the policy.
Combination of Deny All Other Accesses and Ranger macros (for example, {USER}) are not supported.
- Click Add.

Related Information

[SQL Standard Based Hive Authorization](#)

Configure a resource-based policy: S3

How to add a new policy to an existing S3 service.

Procedure

- On **Service Manager** **Resource Policies**, select an existing S3 service.
List of Policies displays a list of the policies defined for S3 service.

2. Click Add New Policy.

Create Policy displays controls for creating details for a new policy

3. Complete the Create Policy page as follows:

Table 53: Policy Details

Field	Description
Policy Name	Enter a unique name for this policy. The name cannot be duplicated anywhere in the system.
Policy Label	An optional label for the policy. You can search reports and filter policies based on these labels.
Enabled/Disabled	Enables or disables the policy.
Normal/Override	Enables you to specify an override policy. When override is selected, the access permissions in the policy override the access permissions in existing policies. This feature can be used with Add Validity Period to create temporary access policies that override existing policies.
S3 Bucket	The S3 bucket.
Path	Specify the path for the policy. The default Recursive setting specifies that the path is recursive; you can also specify a non-recursive path.
Description	(Optional) Describe the purpose of the policy.
Audit Logging	Specify whether this policy is audited. (De-select to disable auditing).

Field	Description
Add Validity Period	Specify a start and end time for the policy.

Table 54: Allow Conditions

Label	Description
Select Role	Specify the roles to which this policy applies. To designate a role as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Select Group	Specify the groups to which this policy applies. To designate a group as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy. The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify the users to which this policy applies. To designate a user as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Permissions	Add or edit permissions: Read, Write, Select/Deselect All.
Delegate Admin	You can use Delegate Admin to assign administrator privileges to the roles, groups, or users specified in the policy. Administrators can edit or delete the policy, and can also create child policies based on the original policy.

- You can use + 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.
- You can use Deny All Other Accesses to deny access to all other users, groups, and roles other than those specified in the allow conditions for the policy.
Combination of Deny All Other Accesses and Ranger macros (for example, {USER}) are not supported.
- Click Add.

Configure a resource-based policy: Solr

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

Procedure

- On **Service Manager** **Resource Policies**, select an existing Solr service.
List of Policies displays a list of the policies defined for Solr service.

2. Click Add New Policy.

Create Policy displays controls for creating details for a new policy.

The screenshot shows the 'Create Policy' interface. At the top right, it says 'Last Response Time 09/26/2023 02:44:45 PM'. The breadcrumb trail is 'Service Manager > cm_solr Policies > Create Policy'. The 'Policy Details' section includes:

- Policy Type:** Access (selected)
- Policy Name:** Policy Name (input field)
- Policy Label:** Select... (dropdown)
- Solr Admin:** security (selected)
- Description:** (text area)
- Audit Logging:** Yes (selected)
- Policy Conditions:** No Conditions (button to add conditions)

Below the details is the 'Allow Conditions' section with a 'Select Roles' dropdown menu open, showing options: Query, Update, and Select All. The 'Permissions' table shows 'Add Permissions' button and a 'Delegate Admin' checkbox.

3. Complete the Create Policy page as follows:

Table 55: Policy Details

Field	Description
Policy Name	Enter an appropriate policy name. This name cannot be duplicated across the system. This field is mandatory.
normal/override	Enables you to specify an override policy. When override is selected, the access permissions in the policy override the access permissions in existing policies. This feature can be used with Add Validity Period to create temporary access policies that override existing policies.
Resource Type	collection - click to select from a list of dynamic values config - click to select from a list of dynamic values schema - click to select from a list of dynamic values admin - click to select COLLECTIONS, CORES, METRICS, SECURITY or AUTOSCALING
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.
Add Validity Period	Specify a start and end time for the policy.

Field	Description
Policy Conditions (applied at the policy level)	Click +, then specify an IP address range.

Table 56: Allow Conditions

Label	Description
Select Role	Specify the roles to which this policy applies. To designate a role as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Select Group	Specify the groups to which this policy applies. To designate a group as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy. The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify the users to which this policy applies. To designate a user as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Policy Conditions (applied at the item level)	Specify an IP address range.
Permissions	Add or edit permissions: Query, Update
Delegate Admin	You can use Delegate Admin to assign administrator privileges to the roles, groups, or users specified in the policy. Administrators can edit or delete the policy, and can also create child policies based on the original policy.

- You can use + 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.
- You can use Deny All Other Accesses to deny access to all other users, groups, and roles other than those specified in the allow conditions for the policy.
Combination of Deny All Other Accesses and Ranger macros (for example, {USER}) are not supported.
- Click Add.

Related Information

[Wildcards and macros in resource-based policies](#)

Configure a resource-based policy: YARN

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

Procedure

- On **Service Manager** **Resource Policies**, select an existing YARN service.
List of Policies displays a list of the policies defined for YARN service.

2. Click Add New Policy.
Create Policy displays controls for creating details for a new policy.

Service Manager

>

cm_yarn Policies

>

Edit Policy

Last Response Time

11/29/2023 01:20:07 PM

Policy Details

Policy Type

Access

Add Validity Period

Policy ID*

67

Policy Name*

yarn queue policy

Enable

Normal

Policy Label

Select...

Queue *

root.queue1

Recursive

Description

Audit Logging*

Yes

Allow Conditions:

hide

Select Roles	Select Groups	Select Users	Permissions	Delegate Admin	
Select...	sys	admin	submit-app admin-queue		

3. Complete the Create Policy page as follows:

Table 57: Policy Details

Field	Description
Policy Name	Enter an appropriate policy name. This name cannot be duplicated across the system. This field is mandatory.
normal/override	Enables you to specify an override policy. When override is selected, the access permissions in the policy override the access permissions in existing policies. This feature can be used with Add Validity Period to create temporary access policies that override existing policies.
Queue	The YARN queue to which the policy applies. For example, enter root.xyz for the xyz queue.
Recursive	The default recursive setting specifies that the policy will also be applied to all sub-queues; you can also specify a non-recursive path.
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.

Field	Description
Add Validity Period	Specify a start and end time for the policy.

Table 58: Allow Conditions

Label	Description
Select Role	Specify the roles to which this policy applies. To designate a role as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Select Group	Specify the groups to which this policy applies. To designate a group as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy. The public group contains all users, so granting access to the public group grants access to all users.
Select User	Specify the users to which this policy applies. To designate a user as an Administrator, select Delegate Admin. Administrators can edit or delete the policy, and can also create child policies based on the original policy.
Permissions	Add or edit permissions: submit-app, admin-queue, Select/Deselect All.
Delegate Admin	You can use Delegate Admin to assign administrator privileges to the roles, groups, or users specified in the policy. Administrators can edit or delete the policy, and can also create child policies based on the original policy.

- You can use + 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.
- You can use Deny All Other Accesses to deny access to all other users, groups, and roles other than those specified in the allow conditions for the policy.
Combination of Deny All Other Accesses and Ranger macros (for example, {USER}) are not supported.
- Click Add.

Related Information

[Wildcards and macros in resource-based policies](#)

Wildcards and macros in resource-based policies

Learn about wildcards and macros in resource-based policies.

Wildcard characters in Ranger authorization resource policy

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

Macros in Ranger policy authoring

In many enterprises, it is customary to create user-accessible data resources (HDFS files/directories, Hive databases/tables, and so on) encoded with the username or some other user-specific attribute value in their names. In such cases, a Ranger administrator needs to author multiple policies addressing these distinct resource names.

USER macro in Ranger provides a generic way to author Ranger policies addressing such relationships between resource names and access permissions for its users to achieve an equivalent access control regime with a single policy or a small number of policies.

Similarly, the OWNER macro in Ranger provides a way to define an access policy for the resources that are owned by the user, thereby restricting others who are not the owner from accessing them unless given specific access.

These two macro definitions make Ranger policy authoring robust and add clarity to the mapping of enterprise-wide and high-level access control regimes to Ranger policy specifications. Fewer policies, in general, lead to significant improvement in capacity and performance in Ranger administration as well as Ranger-enabled components.

USER and OWNER macros in Ranger policy authoring

Learn about USER and OWNER macros and how to use them in Ranger policies to create user-accessible data resources.

USER macro

Let us use the following example to understand the usage of USER macro.

The screenshot displays the 'Create Policy' interface in the Cloudera Ranger console. The breadcrumb trail shows 'Service Manager > cm_hdfs Policies > Create Policy'. The 'Last Response Time' is 07/19/2021 12:40:09 PM. The policy is configured as follows:

- Policy Type:** Access
- Policy Name:** USER macro policy (with an 'i' icon for help)
- Policy Label:** Policy Label
- Resource Path:** /user/{USER} (with a 'x' icon for help)
- Description:** (empty text area)
- Audit Logging:** Yes
- Buttons:** 'Enabled' (selected) and 'Normal' are visible next to the Policy Name. 'Recursive' (selected) is visible next to the Resource Path. 'Add Validity Period' is in the top right.

Below the main configuration, the 'Allow Conditions' section is expanded, showing a table with the following data:

Select Role	Select Group	Select User	Permissions	Delegate Admin
Select Roles	Select Groups	{USER}	Execute Read Write	☐

In the preceding example of the USER macro policy in the HDFS service, the policy has /user/{USER} as the Resource Path. This Resource Path has the {USER} macro, which will be replaced by the username when an access request is authorized using this policy. In Allow Conditions, to give Read/Write/Execute access, there is a {USER} macro defined for the user value, and this macro will be expanded to the corresponding user who accesses the resource. Resource Path macro and the corresponding Allow Condition macro will determine the user's access to the particular resource path.

For example, if the user systest is reading data from /user/systest, this USER macro policy will allow read access. If the user systest tries to read data from /user/hive, access will be denied.

By defining this single policy, enterprises can take advantage of controlling access to the user data.

Customizability of USER macro in Ranger

Ranger requires that the {USER} string be used to represent an accessing user as the user in the policy item in a Ranger policy. However, Ranger provides a 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 the '{' and '}' characters as delimiters for the string USER; however, Ranger supports a 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, the path names contains '{' or '}' as valid characters, but not the '%' character, then the service definition for HDFS can be specified as follows:

```
"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"
}
]
```

The corresponding Ranger policy for the use case for HDFS will be written as follows:

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

The following customizable matcherOptions are available for this feature.

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

Advanced use of USER macro

This feature, at its core, supports general-purpose identification of special patterns in the resource specification and their replacement at runtime with other strings to derive the name of the resource, before matching it with the resource being accessed by the user. Therefore, it is not limited to the replacement of string `USER` with the current user's name; it is just something that is offered out of the box.

In order for Ranger users to use the underlying core functionality, they need to be familiar with interfaces provided by Ranger for customizing Ranger, such as `RangerContextEnricher`, `RangerAccessRequest`, and `RangerConditionEvaluator`.

The following methods are provided to populate and reference context in the `RangerAccessRequest` object that represents access requests in the policy evaluation engine:

- `RangerAccessRequestUtil.setTokenInContext(Map<String, Object> context, String tokenName, Object tokenValue);`
- Object `RangerAccessRequestUtil.getTokenFromContext(Map<String, Object> context, String tokenName);`

An advanced Ranger user needs to provide the following to use the core functionality offered by this feature:

- Provide an implementation of `RangerContextEnricher` and include it in the service definition. The implementation of `RangerContextEnricher.enrich()` method needs to get the context of the `RangerAccessRequest` provided to it and use the `RangerAccessRequestUtil.setTokenInContext()` API to populate it with a specific tokenName (such as USER) and its value, derived based on some runtime information and enricher's configuration parameters.
- Customize the component's service definition with appropriate `matcherOptions` for each resource definition supported by it as described above in the customizability section.
- Provide an implementation of `RangerConditionEvaluator` and include it in the service definition. The implementation of the `RangerConditionEvaluator.isMatched()` API needs to retrieve the value of the tokenName from the request's context using the `RangerAccessRequestUtil.getTokenFromContext()` API, and return the appropriate result.
- Author a Ranger policy with the resource specification containing the tokenName to implement appropriate access control for the resource name built at runtime by Ranger.

OWNER macro

Let us use the following example to understand the usage of OWNER macro.

The screenshot shows the 'Create Policy' form in the Cloudera Ranger Policy Manager. The policy is named 'OWNER policy' and is set to 'Enabled'. The policy label is 'Policy Label'. The resource specification includes 'database', 'table', and 'column' with wildcards, each with an 'Include' button. The description field is empty. Audit logging is set to 'Yes'. The 'Allow Conditions' section shows a table with columns for 'Select Role', 'Select Group', 'Select User', 'Permissions', 'Delegate Admin', and a 'Delete' button. The 'Select User' field contains the macro '{OWNER}'. The 'Permissions' field is set to 'All'.

Select Role	Select Group	Select User	Permissions	Delegate Admin	
Select Roles	Select Groups	{OWNER}	All	☐	X

When resources are created in respective services (HDFS, Hive, HBase, and so on), those resources are marked with the owner information. When the resource's owner accesses those resources, the owner should have all permissions on those resources.

- In the Allow Conditions Ranger policy, for the user entry, the {OWNER} macro is used with the permission All.
- This Owner policy gives the necessary permission to the resources (database/table/column) that the user owns.
- Other resources which are not owned by the user are not allowed to be accessed.
- This single policy gives the flexibility to provide access to the owner of the resources.



Note: Services that take advantage of the Ranger owner policy should support the concept of an owner in the object creation, and the corresponding Ranger plugin for that service should have the request enriched with owner information.

Adding a policy condition to a resource-based policy

You can add a condition to a resource-based policy, using Ranger Admin Web UI when creating a new, or editing an existing policy.

About this task

Ranger Admin Web UI supports adding the following policy conditions to a new or existing resource-based policy for Knox, Kafka and Kafka-connect services.

- IP Address Range for example - xx.xxx.xxx, yy.yyy.yy
- Boolean expression for example - Country_Name="XYZ"

The Policy Conditions dialog prompts for inputs using uhint JSON. For populating "IP-range" for example, we are using JSON like this:

```
{
  "itemId": 1,
  "name": "ip-range",
  "evaluator": "org.apache.ranger.plugin.conditionevaluator.RangerIpMat
cher",
  "evaluatorOptions": {},
  "validationRegEx": "",
  "validationMessage": "",
  "uiHint": "{ \"isMultiValue\":true }",
  "label": "IP Address Range",
  "description": "IP Address Range"
}
```



Important: The uiHint attribute impacts the UI, based on the configured value, as follows:

- "uiHint": "{ \"isMultiline\": true }"
Displays a multi-line text area (for example, for JavaScript expressions).
- "uiHint": "{ \"singleValue\": true }"
Displays a single-select dropdown.
- "uiHint": "{ \"isMultiValue\": true }"
Displays a multi-select dropdown.

Procedure

1. In Service Manager Resource Policies cm_knox_policies (for example), choose one of the following:

Add New Policy

to add a new, tag-based policy.

Policy ID

click a policy ID to edit an existing policy.

2. In either **Create Policy** or **Edit Policy Policy Conditions** , click +.

Edit Policy Last Response Time: 10/23/2023 03:20:11 PM

Service Manager > cm_tag Policies > Edit Policy

Policy Details

Policy Type: **Access** + Add Validity Period

Policy ID*: **4**

Policy Name*: **EXPIRES_ON** Enable Normal

Policy Label: Select...

TAG *: **EXPIRES_ON**

Description: Policy for data with EXPIRES_ON tag

Audit Logging*: **Yes**

Policy Conditions : +

No Conditions

3. In **Policy Conditions**:

- In **IP Address Range** ?, enter or choose existing ip.address.values .
- In **Enter boolean expression**, enter an expression that evaluates to true or false.

Create Policy

Service Manager > cm_knox Policies > Create Policy

Policy Details

Policy Condition

IP Address Range:

enter expression

Enter boolean expression: 1

4. Click **Save**.

Adding a policy label to a resource-based policy

You can add a label to a resource-based policy, using Ranger Admin Web UI when creating a new, or editing an existing policy.

About this task

Ranger Admin Web UI supports adding one or more labels to a new or existing resource-based policy for Cloudera services. The Policy Label field displays all current labels created for a policy. Policy labels:

- Allow users to group sets of policies, using one or more labels.
- Enables users to search (filter) all policies using policy labels from both the [Service Manager Policy Details](#) or [Service Manager Reports](#) pages.
- Enables users to filter and export the filtered list of policies based on the policy label.

How to create a label for a policy

Procedure

1. In [Service Manager Resource Policies cm_hdfs_policies](#) (for example), choose one of the following:

Add New Policy

to add a new, tag-based policy.

Policy ID

click a policy ID to edit an existing policy.

2. In either [Create Policy](#) or [Edit Policy Policy Conditions Policy Label](#), type or select a policy label string.

The screenshot shows the 'Edit Policy' interface in the Ranger Admin Web UI. The breadcrumb navigation at the top reads 'Service Manager > HDFS Policies > Edit Policy'. The 'Policy Details' section includes the following fields and controls:

- Policy Type:** A blue button labeled 'Access'.
- Policy ID*:** A blue button labeled '31'.
- Policy Name*:** A text input field containing 'hdfs test policy' with an information icon.
- Policy Label:** A field containing 'test_policy_label' with a close button (x) and a dropdown arrow. This field is highlighted with an orange border.
- Resource Path*:** A text input field containing '*/tmp' with a close button (x) and a dropdown arrow.
- Description:** A text area containing the word 'label'.
- Audit Logging*:** A toggle switch set to 'Yes'.
- Enable/Normal:** Two toggle switches, 'Enable' (checked) and 'Normal' (unchecked).
- Recursive:** A toggle switch set to 'Recursive'.
- Add Validity Period:** A blue button with a circular icon.

3. Click Save.

Example

To filter existing policies by label from [Resource Policies Search](#), select [Policy Label](#), then type a label name, as shown in the following example:

Figure 5: Filtering Resource-based policies for HDFS service by label

HDFS Policies

Service : HDFS

Select Zone Name

Manage Service

Last Response Time
01/31/2024 04:56:33 PM

Q

POLICY LABEL : test

Add New Policy

Policy ID	Policy Name	Policy Label	Status	Audit Logging	Roles	Groups
31	hdfs test policy	test_policy_la...	Enabled	Enabled	--	--

Example

To export a set of policies having the same label, filter as shown, using Service Manager Reports Policy Label (select) Search in the following example. Then, choose Export.

Figure 6: Exporting a list of policies filtered by label

Reports

Last Response Time
01/31/2024 05:06:30 PM

Search Criteria

Policy Name

Enter Policy Name

Component

Policy Label

test_policy_label

X

Search By

Group

Select...

Policy Type

Access

Resource

Enter Resource Name

Zone Name

Select Zone Name

Q Search

Export

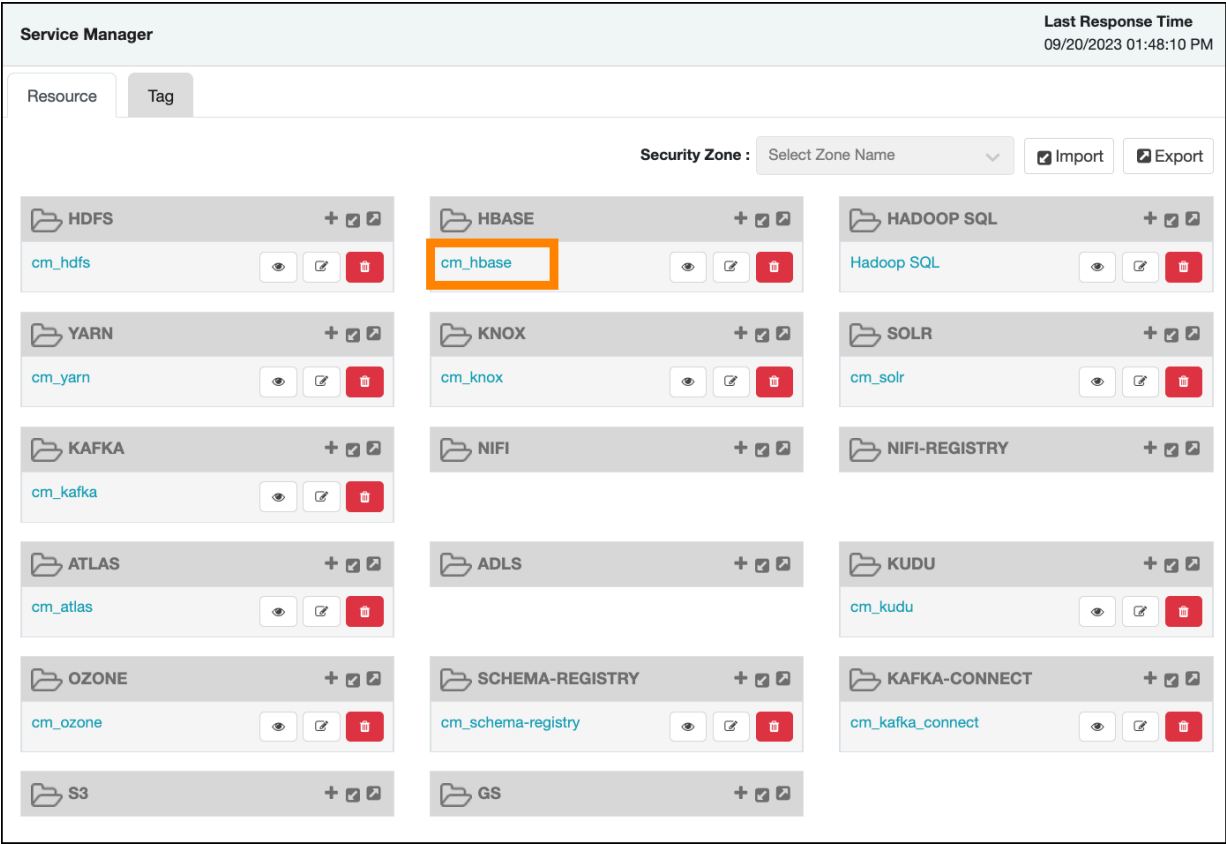
HDFS

Policy ID	Policy Name	Policy Label	Resources	Policy Type	Status	Zone Name
31	hdfs test policy	test_policy_label	path: */tmp	Access	Enabled	--

Preloaded resource-based services and policies

Apache Ranger includes preloaded resource-based services and policies.

- The preloaded resource-based services appear on the Service Manager page for resource-based policies. Service names are prefixed with "cm_", with the exception of Hadoop SQL, which applies to multiple SQL stack components (Hive, Impala, and Data Explorer).



- To view the policies for each preloaded service, click the service name. To view policy details, click the applicable edit icon or policy ID number.

HBASE Policies

Service : cm_hbase

Select Zone Name

Manage Service

Last Response Time
09/20/2023 01:57:55 PM

Q

Search for your policy...

Add New Policy

Policy ID	Policy Name	Policy Label	Status	Audit Logging	Roles	Groups	Users	Actions
5	all - table, column-family, column	--	Enabled	Enabled	--	--	hbase, rangerlookup	
6	ATLAS_HBASE_TABLES	--	Enabled	Enabled	--	--	atlas	
59	grant-1695052642270	--	Enabled	Enabled	--	--	atlas	
60	grant-1695052642467	--	Enabled	Enabled	--	--	atlas	

Index

- [cm_atlas](#)
- [cm_hbase](#)
- [cm_hdfs](#)
- [cm_kafka](#)
- [cm_knox](#)
- [cm_nifi](#)
- [cm_solr](#)

cm_yarn

Hadoop SQL

cm_atlas

all - entity-type, entity-classification, entity, entity-business-metadata

This is a default policy of type "entity" that gives access to all entities and their business metadata attributes for the following users and groups, with the specified permissions:

- admin, dpprofiler, beacon – Update Business Metadata
- rangertagsync, rangerlookup – Read entity
- public group – Read entity

all - entity-type, entity-classification, entity

This is a default policy of type "entity" that gives access to all entities and their classifications for the following users and groups, with the specified permissions:

- admin, dpprofiler, beacon – Read, Create, Update, Delete entity & Add, Update, Remove classification
- rangertagsync, rangerlookup – Read entity
- public group – Read entity

all - entity-type, entity-classification, entity, entity-label

This is a default policy of type "entity" that gives access to all entities and classifications and their labels for the following users and groups, with the specified permissions:

- admin, dpprofiler, beacon – Add, Remove label
- rangertagsync, rangerlookup – Read entity
- public group – Read entity

all - relationship-type, end-one-entity-type, end-one-entity-classification, end-one-entity, end-two-entity-type, end-two-entity-classification, end-two-entity

This is a default policy of type "relationship" that gives access to all to all Entity-Relationships between End1-Entity-Type, End1-Entity-Classification, End1-Entity-ID and End2-Entity-Type, End2-Entity-Classification, End2-Entity-ID for the following users and groups, with the specified permissions:

- admin, dpprofiler, beacon – Add, Update, and Remove relationship
- public group – Add, Update, and Remove relationship

all - atlas-service

This is a default policy of type "atlas-service" that gives access to all atlas-services [export, import, purge, server] for the following users, with the specified permissions:

- admin, dpprofiler, beacon – Admin Export and Admin Import

all - type-category, type

This is a default policy of type "type-category" that gives access to all type categories [ENUM, ENTITY, CLASSIFICATION, RELATIOSHIP, STRUCT] and type names for the following users, with the specified permissions:

- admin, dpprofiler, beacon – Create, Update, and Delete type

Allow users to manage favorite searches

This is a default policy of type "entity-type" that gives access to __AtlasUserProfile and __AtlasUserSavedSearch resources which are internal types for favorite search. This policy provides Read, Create, Update, and Delete Entity permissions to validated users who create a favorite search.

cm_hbase

all - table, column-family, column

Provides access to all HBase tables, column-families, and columns to the following users, with the specified permissions:

- hbase, rangerlookup – Read, Write, Create, Admin

ATLAS_HBASE_TABLES

Provides access to all HBase column-families and columns in the atlas_janus and ATLAS_ENTITY_AUDIT_EVENTS HBase tables, to the following user, with the specified permissions:

- atlas – Read, Write, Create, Admin

cm_hdfs

all - path

Provides access to all HDFS resource paths to the following users, with the specified permissions:

- hdfs, rangerlookup – Read, Write, Execute

kms-audit-path

Provides access to the /ranger/audit/kms resource path to the following user, with the specified permissions:

- keyadmin – Read, Write, Execute

cm_kafka

all - topic

Provides access to all topics to the following users, with the specified permissions:

- kafka, rangerlookup, streamsmgmgr, streamsrepmgr – Publish, Consume, Configure, Describe, Create, Delete, Describe Configs, Alter Configs

all - cluster

Provides access to all clusters to the following users, with the specified permissions:

- kafka, rangerlookup, streamsmgmgr, streamsrepmgr – Configure, Describe, Create, Kafka Admin, Idempotent Write, Describe Configs, Alter Configs

all - transactionalid

Provides transactionalid access to the following users, with the specified permissions:

- kafka, rangerlookup, streamsmgmgr, streamsrepmgr – Publish, Describe

all - delegationtoken

Provides delegationtoken access to the following users, with the specified permissions:

- kafka, rangerlookup, streamsmgmgr, streamsrepmgr – Describe

ATLAS_HOOK

Provides ATLAS_HOOK topic access to the following users, with the specified permissions:

- hbase, hive, impala, mlgov – Publish
- atlas – Create, Configure, and Consume

ATLAS_ENTITIES

Provides ATLAS_ENTITIES topic access to the following users, with the specified permissions:

- atlas – Create, Configure, and Publish
- rangertagsync – Consume

ATLAS_SPARK_HOOK

Provides ATLAS_SPARK_HOOK topic access to the following user, with the specified permissions:

- atlas – Create, Configure, and Consume

Also provides ATLAS_SPARK_HOOK topic access to the following group, with the specified permissions:

- public – Publish

cm_knox

all - topology, service

Provides access to all Knox topologies and services to the following users, with the specified permissions:

- admin, rangerlookup – Allow

cm_nifi

all - nifi-resource

Provides access to all NiFi resource identifiers to the following user, with the specified permissions:

- rangerlookup – Read, Write

cm_solr

all - collection

Provides access to all Solr collections to the following users, with the specified permissions:

- solr, rangerlookup, ranger, atlas – Query, Update, Others, Solr Admin

RANGER_AUDITS_COLLECTION

Provides access to the RANGER_AUDITS_COLLECTION Solr collection to the following users, with the specified permissions:

- atlas, hbase, hdfs, hive, impala, kafka, knox, nifi, ranger, storm, yarn – Query, Update, Others
- ranger – Query, Update, Others, Solr Admin

cm_yarn

all - queue

Provides access to all YARN queues to the following users, with the specified permissions:

- yarn, rangerlookup – submit-app, admin-queue

Hadoop SQL

all - global

Provides global access to the following users, with the specified permission:

- hive, beacon, dpprofiler, hue, admin, impala, rangerlookup – Temporary UDF Admin



Note: The Ranger web UI may show additional permissions for the all-global policy, but the only valid permission is Temporary UDF Admin.

all - database, table, column

Provides access to all databases, tables, and columns to the following users, with the specified permissions:

- hive, rangerlookup, impala – Select, Update, Create, Drop, Alter, Index, Lock, All, Read, Write, ReplAdmin, Service Admin, Temporary UDF Admin, Refresh
- {OWNER} – All

all - database, table

Provides access to all databases and tables to the following users, with the specified permissions:

- hive, rangerlookup, impala – Select, Update, Create, Drop, Alter, Index, Lock, All, Read, Write, ReplAdmin, Service Admin, Temporary UDF Admin, Refresh
- {OWNER} – All

all - storage-type, storage-url

Ranger introduces new resources “storage-type” and “storage-url” in HadoopSQL Service and supports only one permission “RW Storage”. When a user creates / alters a table, they will be authorized against this resource policy. Users granted “RW Storage” permission on the resource representing the storage-type + storage-url, can create/alter the table in the respective storage. Provides access to all databases to the following users, with the RW Storage permission only:

- hive, rangerlookup, impala, beacon, dpprofiler, hue, admin



Note: {OWNER} macro should NOT be configured for StorageHandler policies.

all - database

Provides access to all databases to the following users, with the specified permissions:

- hive, rangerlookup, impala – Select, Update, Create, Drop, Alter, Index, Lock, All, Read, Write, ReplAdmin, Service Admin, Temporary UDF Admin, Refresh
- {OWNER} – All

Also provides access to all databases to the following group, with the specified permissions:

- public – Create

all - hiveservice

Provides hiveservice access to the following users, with the specified permissions:

- hive, rangerlookup, impala – Select, Update, Create, Drop, Alter, Index, Lock, All, Read, Write, ReplAdmin, Service Admin, Temporary UDF Admin, Refresh

all - database, udf

Provides database and udf access to the following users, with the specified permissions:

- hive, rangerlookup, impala – Select, Update, Create, Drop, Alter, Index, Lock, All, Read, Write, ReplAdmin, Service Admin, Temporary UDF Admin, Refresh
- {OWNER} – All

all - url

Provides url access to the following users, with the specified permissions:

- hive, rangerlookup, impala – Select, Update, Create, Drop, Alter, Index, Lock, All, Read, Write, ReplAdmin, Service Admin, Temporary UDF Admin, Refresh

default database tables columns

Provides access to all tables and columns in the default database to the following user, with the specified permissions:

- impala – Create

Also provides access to all tables and columns in the default database to the following group, with the specified permissions:

- public – Create

information_schema database tables columns

Provides access to all tables and columns in the information_schema database to the following user, with the specified permissions:

- impala – Select

Also provides access to all tables and columns in the information_schema database to the following group, with the specified permissions:

- public – Select

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 [Ranger Admin Web UI Service Manager Resource](#) :

Service Manager

Last Response Time
09/21/2023 09:02:46 AM

Resource

Tag

Security Zone :

Select Zone Name

Import

Export

HDFS

cm_hdfs

HBASE

cm_hbase

HADOOP SQL

Hadoop SQL

YARN

cm_yarn

KNOX

cm_knox

SOLR

cm_solr

KAFKA

cm_kafka

NIFI

NIFI-REGISTRY

ATLAS

cm_atlas

ADLS

KUDU

cm_kudu

OZONE

cm_ozone

SCHEMA-REGISTRY

cm_schema-registry

KAFKA-CONNECT

cm_kafka_connect

S3

GS

You can also export policies from Ranger Admin Web UI Service Manager Reports :

Reports

Last Response Time
09/21/2023 09:08:09 AM

Search Criteria

Policy Name

Policy Type

Access

Component

Resource

Policy Label

Zone Name

Select Zone Name

Search By

Group

Select...

Search

Excel file

Export

CSV file

JSON file

Zone Name

HDFS

Policy ID	Policy Name	Policy Label	Resources	Policy Type		
1	all - path	--	path: /*	Access	Enabled	--
2	kms-audit-path	--	path: /ranger/audit/...	Access	Enabled	--

HBASE

Policy ID	Policy Name	Policy Label	Resources	Policy Type	Status	Zone Name
5	all - table, column...	--	column-family: * column: * table: *	Access	Enabled	--

Table 59: Export Policy Options

	Service 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 Service 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.



Important: When importing Ranger policies, it is crucial to understand the behavior of the import process regarding users and groups that do not exist in Ranger:

- Automatic user or group creation: The Ranger policy import process automatically creates any non-existing users or groups at the Ranger admin end as external users or groups. This is because the import process does not have access to the passwords of these users or groups, unlike users or groups created through the Ranger UI who are considered internal with known passwords.
- Handling of existing users: If a synchronization process is actively managing users or groups, any users or groups created during the policy import are updated with the details from the synchronization process as soon as they are synchronized. This ensures that users and groups information remains current and consistent with your identity management system.
- Impact on policy enforcement: While the absence of specific Hive resources or HDFS paths in the target system does not impede the policy import process, policies linked to these non-existent resources are not enforced until the resources are created. Therefore, Cloudera recommends to verify the existence of critical resources in the new environment to ensure that all policies function as expected after migration.

Related Information

[Importing and exporting tag-based policies](#)

Import resource-based policies for a specific service

How to import resource-based policies for a specific service (HBase, YARN, etc.).

Procedure

1. On the Service Manager page, click the Import icon for the service:



The Import Policy page appears.

2. Select the file to import.

You can only import policies in JSON format.

Security Zone : Select

Import Policy

Select File :

Select file  Override Policy : ☐

Ranger_Policies_20190717_190622.json ✖

Info All services gets listed on service destination when Zone destination is blank. When zone is selected at destination, then only services associated with that zone will be listed.

Specify Zone Mapping :

Source		Destination
	To	No zone selected

Specify Service Mapping :

Source		Destination
cm_hdfs	To	Select service name

Cancel Import

3. (Optional) Configure the import operation:

- The Override Policy option deletes all policies of the destination repositories.
- Zone Mapping – when no destination is selected, all services are imported. When a destination is selected, only the services associated with that security zone are imported.
- Service Mapping maps the downloaded file repository, i.e. source repository to destination repository. You can use the red x symbols to remove services from the import. Scroll down to view all service mappings.

Import Policy

Specify Zone Mapping :

Source: To Destination:

Specify Service Mapping :

Source	To	Destination
cm_hdfs	To	cm_hdfs
cm_hbase	To	cm_hbase
cm_yarn	To	cm_yarn
cm_hive	To	cm_hive
cm_knox	To	cm_knox
cm_storm	To	cm_storm

Cancel Import

4. Click Import.

A confirmation message appears after the file is imported.

Related Information

[Import resource-based policies for all services](#)

Import resource-based policies for all services

How to import policies for all services.

Procedure

1. On Service Manager Resource , click Import.

The screenshot displays the 'Service Manager' interface. At the top right, it shows the 'Last Response Time' as '09/21/2023 09:02:46 AM'. Below the header, there are tabs for 'Resource' and 'Tag'. A 'Security Zone' dropdown menu is set to 'Select Zone Name'. To the right of the dropdown are two buttons: 'Import' (which is highlighted with an orange box) and 'Export'. The main area contains a grid of service cards. Each card represents a service category (e.g., HDFS, YARN, KAFKA) and lists specific resources (e.g., cm_hdfs, cm_yarn, cm_kafka). Each resource card includes a folder icon, a name, and three action icons: a plus sign, a checkmark, and a trash can. The services listed include HDFS, HBASE, HADOOP SQL, YARN, KNOX, SOLR, KAFKA, NIFI, NIFI-REGISTRY, ATLAS, ADLS, KUDU, OZONE, SCHEMA-REGISTRY, KAFKA-CONNECT, S3, and GS.

The Import Policy page appears.

Security Zone : Select

Import Policy

Select File :

Select file 

Override Policy : ☐

Ranger_Policies_20190717_190622.json 

 All services gets listed on service destination when Zone destination is blank.
When zone is selected at destination, then only services associated with that zone will be listed.

Specify Zone Mapping :

Source

Destination

To

No zone selected 

Specify Service Mapping :

Source

Destination

cm_hdfs  

To

cm_hdfs   

Cancel

Import

2. Select the file to import.

You can only import policies in JSON format.

107

3. (Optional) Configure the import operation:

- The Override Policy option deletes all policies of the destination repositories.
- Zone Mapping – when no destination is selected, all services are imported. When a destination is selected, only the services associated with that security zone are imported.
- Service Mapping maps the downloaded file repository, i.e. source repository to destination repository. You can use the red x symbols to remove services from the import. Scroll down to view all service mappings.

Import Policy

Specify Zone Mapping :

Source: [Empty] To Destination: [No zone selected]

Specify Service Mapping :

Source	To	Destination
cm_hdfs	To	cm_hdfs
cm_hbase	To	cm_hbase
cm_yarn	To	cm_yarn
cm_hive	To	cm_hive
cm_knox	To	cm_knox
cm_storm	To	cm_storm

Buttons: Cancel, Import

4. Click Import.

A confirmation message appears after the file is imported.

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 would like to export in Excel or CSV format, export the policies from the Reports page dropdown menu.

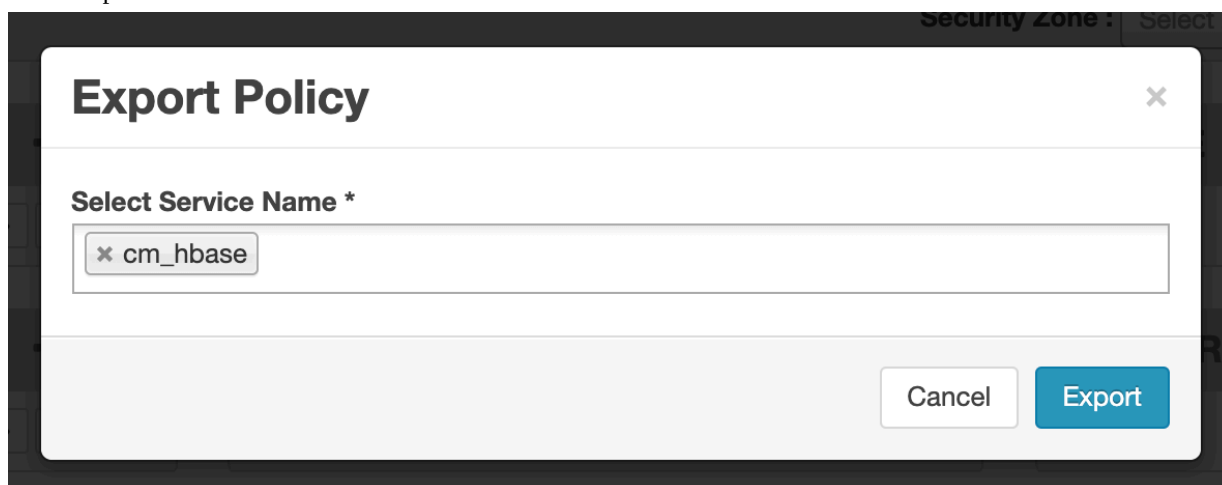
Procedure

1. On Service Manager Resource , click the Export icon for the service:



The Export Policy page appears.

2. Click Export.



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

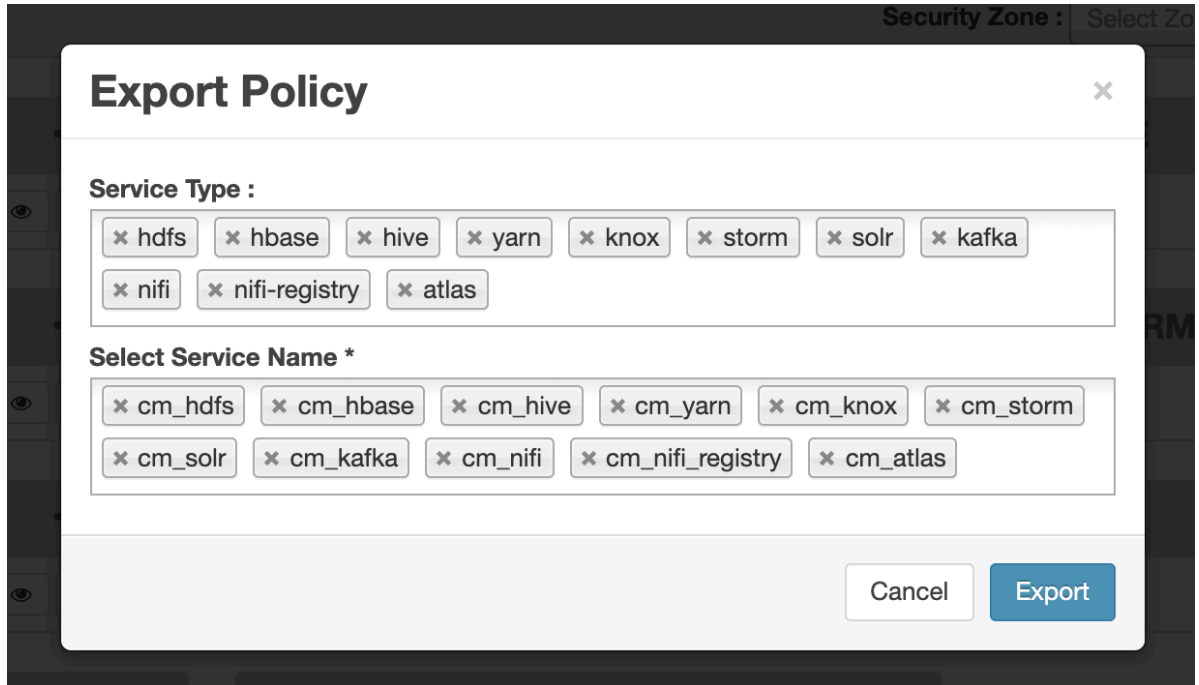
How to export the policies for all services.

About this task

If you would like to export in Excel or CSV format, export the policies from the Reports page drop-down menu.

Procedure

- From Service Manager Resource :
 - a) Click Export.
The Export Policy page appears.
 - b) Remove components or specific services, then click Export.

The image shows a screenshot of the 'Export Policy' dialog box in the Cloudera Service Manager. The dialog has a title bar with a close button (X). Below the title, there are two sections: 'Service Type :' and 'Select Service Name *'. The 'Service Type :' section contains a grid of buttons with 'x' icons, representing selected services: hdfs, hbase, hive, yarn, Knox, storm, solr, kafka, nifi, nifi-registry, and atlas. The 'Select Service Name *' section contains a similar grid of buttons with 'x' icons, representing selected service names: cm_hdfs, cm_hbase, cm_hive, cm_yarn, cm_knox, cm_storm, cm_solr, cm_kafka, cm_nifi, cm_nifi_registry, and cm_atlas. At the bottom right of the dialog, there are two buttons: 'Cancel' and 'Export'.

The file downloads in your browser as a JSON file.

- From Service Manager Reports :
 - Apply filters before exporting the file.
 - Open the Export drop-down menu:

Reports
Last Response Time
09/21/2023 09:08:09 AM

Search Criteria

Policy Name

Policy Type

Access

Component

Resource

Policy Label

Zone Name

Select Zone Name

Search By

Group

Select...

Q Search

Excel file

CSV file

JSON file

Export

HDFS

Policy ID	Policy Name	Policy Label	Resources	Policy Type	Status	Zone Name
1	all - path	--	path: /*	Access	Enabled	--
2	kms-audit-path	--	path: /ranger/audit/...	Access	Enabled	--

HBASE

Policy ID	Policy Name	Policy Label	Resources	Policy Type	Status	Zone Name
5	all - table, column...	--	column-family: * column: * table: *	Access	Enabled	--

- 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.



Note: Users with ALTER permissions on a table can override the policies set for row-level filtering, resource-based column masking, and tag-based column masking in that table. This means that users with ALTER permissions can rename tables or columns, even if masking or row-filtering policies are in place to restrict data access.

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 Service Manager Resource , select an existing Hadoop SQL service.
2. Select Row Level Filter , then click Add New Policy.

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

Table 60: 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.

Field	Description
normal/override	Enables you to specify an override policy. When override is selected, the access permissions in the policy override the access permissions in existing policies. This feature can be used with Add Validity Period to create temporary access policies that override existing policies.
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.
Add Validity Period	Specify a start and end time for the policy.

Table 61: Row Filter Conditions

Label	Description
Select Role	Specify the roles to which this policy applies.
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.

Label	Description
Add Row Filter	- To create a row filter for the specified users, groups, and roles, 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.

Create Policy

Last Response Time
09/21/2023 02:28:09 PM

Service Manager > Hadoop SQL Policies > Create Policy

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

Policy Details

Policy Type

Row Level Filter

○ Add Validity Period

Policy Name*

Policy Name i

Enabl

Norma

Policy Label

Select... ▼

Hive Database *

Select... ▼

Hive Table *

Select... ▼

Description

Audit Logging*

Yes ●

Row Filter Conditions:

hide ▲

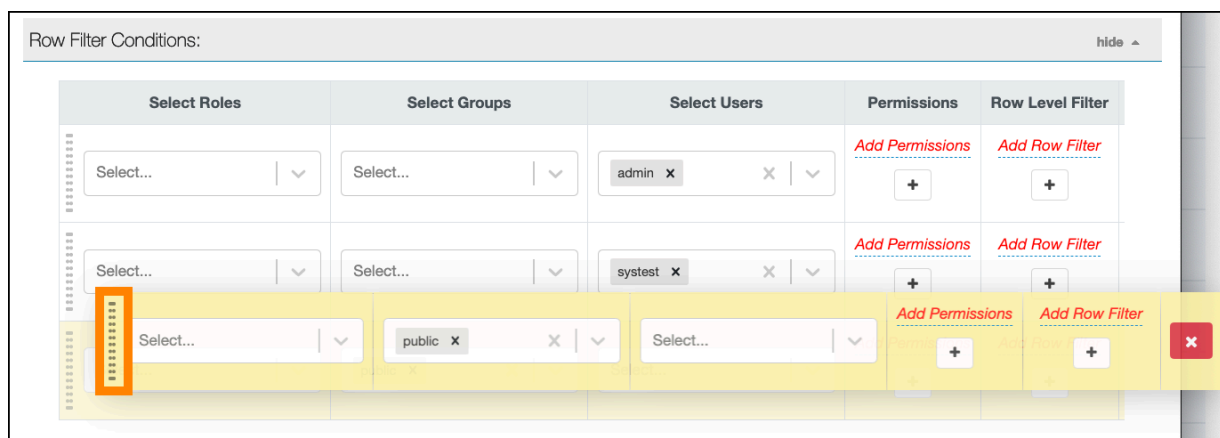
Select Roles	Select Groups	Select	Enter	Row Level Filter
Select... ▼	Select... ▼	admin ✕	enter expression	Add Row Filter +
Select... ▼	Select... ▼	sysrest ✕	✓ ✕ +	Add Row Filter +
Select... ▼	public ✕ ✕ ▼	Select... ▼	Add Permissions +	Add Row Filter +

+

Save

Cancel

- To move a condition in the Row Filter Conditions list (and therefore change the order in which the list 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.



- Click Add in Create Policy 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.



Note: Ranger enables masking functionality to be called on the Hive end. For details on supported column types, please refer to [Apache Hive documentation](#).

- 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.



Note: Operations like insert/update/delete/export are denied for users if row-filter or column-masking policies are applicable on the table for the user.

Procedure

- On Service Manager Resource , select an existing Hadoop SQL service.

2. Select Masking , then click Add New Policy.

The screenshot shows the 'HIVE Policies' management page. At the top, there's a header with 'Service : Hadoop SQL', 'Select Zone Name', and a 'Manage Service' button. Below this, there are three tabs: 'Access', 'Masking' (highlighted with an orange box), and 'Row Level Filter'. A search bar with the placeholder 'Search for your policy...' is present. To the right of the search bar is an 'Add New Policy' button, also highlighted with an orange box. Below the search bar is a table with columns: Policy ID, Policy Name, Policy Label, Status, Audit Logging, Roles, and Groups. The table is currently empty, showing a message 'No data to show!!'.

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

Table 62: 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.
normal/override	Enables you to specify an override policy. When override is selected, the access permissions in the policy override the access permissions in existing policies. This feature can be used with Add Validity Period to create temporary access policies that override existing policies.
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.
Add Validity Period	Specify a start and end time for the policy.

Table 63: Mask Conditions

Label	Description
Select Role	Specify the roles to which this policy applies.
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.

Label	Description
Select Masking Type	To create a row filter for the specified users, groups, and roles, click Select Masking Option, then select a masking type: • Redact – For Redact masking, the data types and mask values are as follows: • String: Mask all alphabetic characters with "x" and all numeric characters with "n". • Int: Mask all numeric values with "1". • Float: Mask all float values as "NULL". • 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). 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.

Create Policy

Last Response Time
09/21/2023 03:34:47 PM

[Service Manager](#) > [Hadoop SQL Policies](#) > Create Policy

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

Policy Details

Policy Type

Masking

Add Validity Period

Policy Name*

Policy Name

Enable

Normal

Policy Label

Select...

Hive Database *

Select...

Hive Table *

Select...

Hive Column *

Select...

Description

Audit Logging*

Yes

Mask Conditions:

Select Roles

Select Groups

Select Masking Options

Select...

Select...

hive

Select Masking Options

Select Masking Options

Save

Cancel

Enter

Redact

Partial mask: show last 4

Partial mask: show first 4

Hash

Nullify

Unmasked (retain original value)

Date: show only year

Custom

118

- To move a condition in the Mask Conditions list (and therefore change the order in which the list 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.

The screenshot shows the 'Mask Conditions' interface. It has a table with five columns: 'Select Roles', 'Select Groups', 'Select Users', 'Permissions', and 'Select Masking Option'. The third row is highlighted in yellow. On the left side of the first column, there are three rows of dotted icons, indicating that conditions can be reordered. The third row's dotted icon is highlighted with an orange box. The 'Select Users' column for the third row shows 'public' with a dropdown arrow. The 'Permissions' and 'Select Masking Option' columns for the third row have '+ Add Permissions' and '+ Select Masking Option' buttons respectively. A red 'X' button is visible on the far right of the third row.

- On Create Policy, 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, Ranger 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).
-  **Note:** Ranger depends on Hive/Impala's hashing functions for hash masking.
 - Impala uses sha512 in FIPS mode, sha256 in non-FIPS mode.
 - Hive uses sha256. Plans to update to sha512 in FIPS mode.
- 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

- On Service Manager Tag, select a tag-based service.
- On TAG Policies, select Masking, then click Add New Policy.

The screenshot shows the 'TAG Policies' interface. At the top, there's a 'Service : cm_tag' dropdown and a 'Select Zone Name' dropdown. A 'Manage Service' button is on the right. Below this, there's a tab bar with 'Access' and 'Masking' tabs. The 'Masking' tab is selected and highlighted with an orange box. Below the tabs is a search bar with the placeholder text 'Search for your policy...'. To the right of the search bar is an 'Add New Policy' button, which is also highlighted with an orange box. Below the search bar is a table with columns: 'Policy ID', 'Policy Name', 'Policy Label', 'Status', 'Audit Logging', 'Roles', and 'Grp'. The table is currently empty, with a message 'No data to show!!' at the bottom.

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

Table 64: Policy Details

Field	Description
Policy Type (required)	Set to Masking by default.
Policy Name (required)	Enter an appropriate policy name. This name cannot be duplicated across the system. The policy is enabled by default.
normal/override	Enables you to specify an override policy. When override is selected, the access permissions in the policy override the access permissions in existing policies. This feature can be used with Add Validity Period to create temporary access policies that override existing policies.
TAG (required)	Enter the applicable tag name, for example 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.
Add Validity Period	Specify a start and end time for the policy.
Policy Conditions (applied at the policy level)	Click + to add policy conditions. Currently "Accessed after expiry_date? (yes/no)" is the only available policy condition. "Accessed after expiry_date (yes/no)?" : To set this condition, type yes in the text box, then click check mark 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". Click Save to save the policy condition.

Table 65: Mask Conditions

Label	Description
Select Role	Specify the roles to which this policy applies.
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 (applied at the item level)	Click Add Conditions to add policy conditions. Currently "Accessed after expiry_date? (yes/no)" is the only available policy condition. "Accessed after expiry_date (yes/no)?" : To set this condition, type yes in the text box, then click check mark 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".
Permissions - Access Types	Currently select is the only available access type for the hive component.

Label	Description
Select Masking Option	To create a row filter for the specified users, groups, and roles, click Select Masking Option, then select a masking type: - 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). 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.

Create Policy Last Response Time
09/21/2023 04:10:36 PM

Service Manager > cm_tag Policies > Create 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** Add Validity Period

Policy Name*: Enabled Normal

Policy Label:

TAG *:

Description:

Audit Logging*: Yes

Mask Conditions:

Select Roles: Select Groups: Select Users:

Enter

- ☐ Redact
- ☐ Partial mask: show last 4
- ☐ Partial mask: show first 4
- ☐ Hash
- ☐ Nullify
- ☐ Unmasked (retain original value)
- ☐ Date: show only year
- ☐ Custom

Select Masking Option

Select Masking Option + ×

Save **Cancel**

- Click + 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](#)

Apply custom transformation to a column

In this section, you find the details of Apache Ranger policy to set a custom expression to transform the value of name_first column.

Data before transformation

The following example shows the data before transformation:

id	name_first	name_last	addr_country	date_of_birth
1	Mackenzzy	Smith	US	1993-12-18
2	Sherlyn	Miller	US	1975-03-22
3	Khiana	Wilson	US	1989-08-14
4	Jack	Thompson	US	1962-10-28
5	Felix	Lee	CA	1982-04-17
6	Adam	Brown	CA	1995-09-07
7	Lucas	Jones	CA	1999-02-06

Policy details

Add a policy with the following details to specify the expression, `initcap(reverse({col}))`, to transform `name_first` column values. Ensure that the data type of the expression is same as the data type of the column. Token `{col}` in the expression will be replaced by Apache Ranger policy engine by the name of the column on which masking is being applied.

Policy Details :

Policy Type

Masking

Policy Name *

masking: customer.name_first

enabled

Hive Database *

cust

Hive Table *

customer

Hive Column *

name_first

Audit Logging

YES

Description

masking policy for customer.name_first column

Mask Conditions :

Select Group

public

Select User

Select User

Access Types

select

Select Masking Option

Custom

initcap(reverse({col}))

Query Results

The query result shows transformed value `name_first` column, as shown in the following example:

[john@localhost ~]# beeline -u jdbc:hive2://localhost.localdomain:10000/cust

0: jdbc:hive2://localhost.localdomain:10000> select * from cust.customer;

id	name_first	name_last	addr_country	date_of_birth
1	Mackenzzy	Smith	US	1993-12-18
2	Sherlyn	Miller	US	1975-03-22
3	Khiana	Wilson	US	1989-08-14
4	Jack	Thompson	US	1962-10-28
5	Felix	Lee	CA	1982-04-17
6	Adam	Brown	CA	1995-09-07
7	Lucas	Jones	CA	1999-02-06

id	name_first	name_last	addr_country	date_of_birth
phone_num				
1	Yznekc	Smith	US	1993-12-18
123-456-7890	2	Nylreh	US	1975-03-22
234-567-8901	3	Anaihk	US	1989-08-14
345-678-9012	4	Kcaj	US	1962-10-28
456-789-0123	5	Xilef	CA	1982-04-17
321-654-0987	6	Mada	CA	1995-09-07
432-765-1098	7	Sacul	CA	1999-02-06
543-876-2109				

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

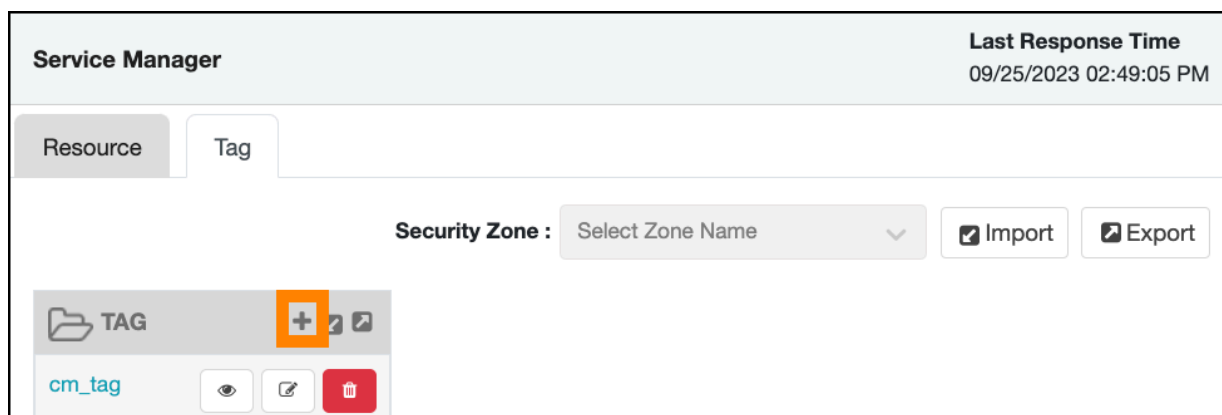
How to add a tag-based service to Ranger.

About this task

You can use [Ranger Admin Web UI Service Manager Tag Policies](#) to create tag-based services and add tag-based access policies that can be applied to Cloudera resources. Using tag-based policies enables you to control access to resources across multiple 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 [Ranger Admin Web UI Service Manager Tag Policies](#) , then click +.



2. On Create Service, 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.

Create Service

Last Response Time
09/26/2025 11:56:28 AM

[Service Manager](#) > Create Service

Service Details :

Service Name *

Display Name

Description

Active Status ☒ Enabled ☐ Disabled

Config Properties :

Policy Download Users

Tag Download Users

Service Admin Users

Service Admin Groups

Superusers

Superuser Groups

Userstore Download Users

Add New Custom Configurations

Audit Filter : ☒

Is Audited	Access Result	Resources	Operations	Permissions	Users	Groups	Roles
Yes	DENIED		Type Operations Name	Add Permissions	Select Users	Select Groups	Select Roles

Test Connection

AddCancel

3. The new tag service appears in Service Manager.

Service Manager

Last Response Time
09/25/2023 03:02:03 PM

Resource

Tag

Security Zone :

Select Zone Name

ImportExport

TAG

cm_tag

tag_service1

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 Service Manager Tag Policies , then select a tag-based service.
2. List of Policies displays existing Access policies by default. Click Add New Policy.

TAG Policies Service : cm_tag Select Zone Name Manage Service Last Response Time 09/27/2023 09:59:17 AM

Access Masking

Search for your policy...

Add New Policy

Policy ID	Policy Name	Policy Label	Status	Audit Logging	Roles	Groups	Users	Actions
4	EXPIRES_ON	--	Enabled	Enabled	--	public	--	

Create Policy displays controls for creating details for a new policy.

Create Policy Last Response Time 09/27/2023 10:06:28 AM

Service Manager > cm_tag Policies > Create Policy

Policy Details

Policy Type **Access** Add Validity Period

Policy Name* Policy Name Enabled Normal

Policy Label Select...

TAG * Select...

Description

Audit Logging* Yes

Allow Conditions: hide

Select Roles	Select Groups	Select Users	Policy Conditions	Permissions	
Select...	Select...	Select...	Add Conditions	Add Permissions	

+ Exclude from Allow Conditions:

Select Roles	Select Groups	Select Users	Policy Conditions	Permissions	
Select...	Select...	Select...	Add Conditions	Add Permissions	

+ Deny All Other Accesses: False

Deny Conditions: hide

Select Roles	Select Groups	Select Users	Policy Conditions	Permissions	
Select...	Select...	Select...	Add Conditions	Add Permissions	

+

3. Complete the Create Policy page as follows:

Table 66: Policy Details

Field	Description
Policy Type	Set to Access by default.
Policy Name	Enter a unique policy name. This name cannot be duplicated across the system. This field is mandatory.
normal/override	Enables you to specify an override policy. When override is selected, the access permissions in the policy override the access permissions in existing policies. This feature can be used with Add Validity Period to create temporary access policies that override existing policies.
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.
Add Validity Period	Specify a start and end time for the policy.
Policy Conditions (applied at the policy level)	Click the + icon to add policy conditions. Currently "Accessed after expiry_date? (yes/no)" is the only available policy condition. "Accessed after expiry_date (yes/no)?:": To set this condition, type yes in the text box, then click the 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". Click Save to save the policy condition.

Table 67: Allow, Exclude from Allow, Deny, and Exclude from Deny Conditions

Label	Description
Select Role	Specify the roles to which this policy applies.
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 (applied at the item level)	Click Add Conditions to add policy conditions. Currently "Accessed after expiry_date? (yes/no)" is the only available policy condition. "Accessed after expiry_date (yes/no)?:": To set this condition, type yes in the text box, then click the 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".

Label	Description
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. Click the check mark button to add the chosen component conditions to the policy.

4. You can use + 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. You can use Deny All Other Accesses to deny access to all other users, groups, and roles other than those specified in the allow conditions for the policy.
6. Click Add to add the new policy.

Related Information

[Using tag attributes and values in Ranger tag-based policy conditions](#)

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.RangerScriptCon
ditionEvaluator",
  "evaluatorOptions": {
    "engineName": "JavaScript",
    "ui.isMultiline": "true"
  },
  "uiHint": "{ \"isMultiline\": true }",
  "label": "Enter boolean expression",
  "description": "Boolean expression"
}
```



Important: The uiHint attribute impacts the UI, based on the configured value, as follows:

- "uiHint": "{ \"isMultiline\": true }"
Displays a multi-line text area (for example, for JavaScript expressions).
- "uiHint": "{ \"singleValue\": true }"
Displays a single-select dropdown.
- "uiHint": "{ \"isMultiValue\": true }"
Displays a multi-select dropdown.

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.

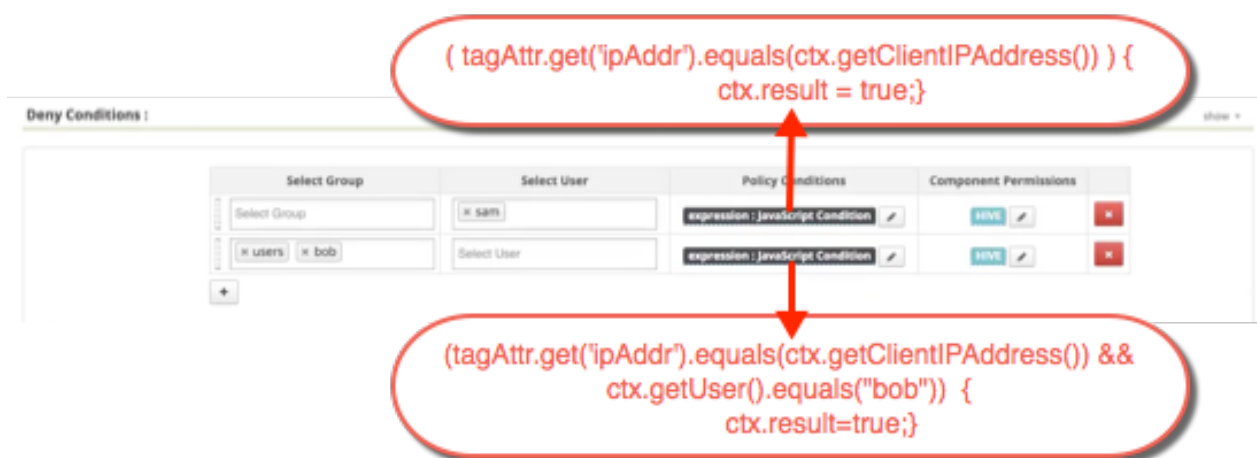
Set the deny condition for user sam with the following boolean expression:

```
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;
}
```



Adding a policy condition to a tag-based policy

You can add a condition to a tag-based policy, using Ranger Admin Web UI when creating a new, or editing an existing policy.

About this task

Ranger Admin Web UI supports adding one of the following policy conditions to a new or existing tag-based policy:

- Accessed after expiry_date ? for example - Yes/No
- Boolean expression for example - Country_Name="XYZ"

The Policy Conditions dialog prompts for inputs using uhint JSON. For populating "Accessed after expiry_date? (yes/no)" for example, we are using JSON like this:

```
{
  "itemId": 1,
  "name": "accessed-after-expiry",
  "evaluator": "org.apache.ranger.plugin.conditionevaluator.RangerScriptTemplateConditionEvaluator",
  "evaluatorOptions": {
    "scriptTemplate": "ctx.isAccessedAfter('expiry_date');"
  },
  "uiHint": "{ \"singleValue\":true }",
  "label": "Accessed after expiry_date (yes/no)?",
}
```

```
} "description": "Accessed after expiry_date? (yes/no)"
```



Important: The uiHint attribute impacts the UI, based on the configured value, as follows:

- "uiHint": "{ \"isMultiline\": true }"

Displays a multi-line text area (for example, for JavaScript expressions).

- "uiHint": "{ \"singleValue\": true }"

Displays a single-select dropdown.

- "uiHint": "{ \"isMultiValue\": true }"

Displays a multi-select dropdown.

Procedure

1. In Service Manager Tag Policies `cm_tag_policies`, choose one of the following:

Add New Policy

to add a new, tag-based policy.

Policy ID

click a policy ID to edit an existing policy.

2. In either Create Policy or Edit Policy Policy Conditions, click +.

Edit Policy Last Response Time
10/23/2023 03:20:11 PM

[Service Manager](#) > [cm_tag Policies](#) > Edit Policy

Policy Details

Policy Type: Access

Policy ID*: 4

Policy Name*: EXPIRES_ON
Enable
Normal

+ Add Validity Period

Policy Label: Select...

TAG *: EXPIRES_ON
x
v

Description: Policy for data with EXPIRES_ON tag

Audit Logging*: Yes

Policy Conditions : +

No Conditions

3. In Policy Conditions:
 - a) In Accessed after expiry date ?, select Yes or No.
 - b) In Enter boolean expression, enter an expression that evaluates to true or false.
Country_Name="XYZ"
4. Click Save.

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 Service Manager Tag Policies , then select a tag-based service.
2. List of Policies displays existing Access policies by default. Click Add New Policy.

TAG Policies Service : cm_tag Select Zone Name Manage Service Last Response Time 09/27/2023 09:59:17 AM

Access Masking

Search for your policy... Add New Policy

Policy ID	Policy Name	Policy Label	Status	Audit Logging	Roles	Groups	Users	Actions
4	EXPIRES_ON	--	Enabled	Enabled	--	public	--	

Create Policy displays controls for creating details for a new policy.

Create Policy Last Response Time 09/27/2023 10:06:28 AM

Service Manager > cm_tag Policies > Create Policy

Policy Details

Policy Type Access Add Validity Period

Policy Name* Policy Name Enabled Normal Policy Conditions : No Conditions

Policy Label Select...

TAG * Select...

Description

Audit Logging* Yes

Allow Conditions: hide

Select Roles	Select Groups	Select Users	Policy Conditions	Permissions	
Select...	Select...	Select...	Add Conditions	Add Permissions	

Exclude from Allow Conditions:

Select Roles	Select Groups	Select Users	Policy Conditions	Permissions	
Select...	Select...	Select...	Add Conditions	Add Permissions	

Deny All Other Accesses: * False

Deny Conditions: hide

Select Roles	Select Groups	Select Users	Policy Conditions	Permissions	
Select...	Select...	Select...	Add Conditions	Add Permissions	

3. Complete the Create Policy page as follows:

Table 68: Policy Details

Field	Description
Policy Type	Set to Access by default.
Policy Name	PII

Field	Description
TAG	PII
Audit Logging	YES
Description	Restrict access to resources with the PII tag.

Table 69: Allow Conditions

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

Table 70: Deny Conditions

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

Table 71: Exclude from Deny Conditions

Label	Description
Select Group	audit
Select User	<none>
Policy Conditions	<none>

Label	Description
Component Permissions	hive (select all permissions)

Create Policy Last Response Time
09/27/2023 10:06:28 AM

Service Manager > cm_tag Policies > Create Policy

Policy Details

Policy Type: **Access** Add Validity Period

Policy Name*: PII Enabled ☒ Normal ☐

Policy Label: Select... Policy Conditions: No Conditions

TAG *: PII +

Description: Restrict access to resources with the PII tag

Audit Logging*: ☒ Yes

Allow Conditions: hide

Select Roles	Select Groups	Select Users	Policy Conditions	Permissions	
Select...	audit x	Select...	Add Conditions	HIVE	x
+					

Exclude from Allow Conditions:

Select Roles	Select Groups	Select Users	Policy Conditions	Permissions	
Select...	Select...	Select...	Add Conditions	Add Permissions	x
+					

Deny All Other Accesses: ☐ False

Deny Conditions: hide

Select Roles	Select Groups	Select Users	Policy Conditions	Permissions	
Select...	public x	Select...	Add Conditions	HIVE	x
+					

Exclude from Deny Conditions:

Select Roles	Select Groups	Select Users	Policy Conditions	Permissions	
Select...	audit x	Select...	Add Conditions	HIVE	x
+					

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.

- Click Add to add the new policy.

Default EXPIRES_ON tag 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 **Service Manager Tag Policies** , then select a tag-based service.
2. On **List of Policies**, click **Edit** for the default EXIRES_ON policy.

TAG Policies

Service : cm_tag

Select Zone Name

Manage Service

Last Response Time
09/27/2023 10:36:57 AM

Access

Masking

Q

Search for your policy...

Add New Policy

Policy ID	Policy Name	Policy Label	Status	Audit Logging	Roles	Groups	Users	Actions
4	EXPIRES_ON	--	Enabled	Enabled	--	public	--	

Licensed under the Apache License, Version 2.0

The Edit Policy page appears:

Edit Policy

Last Response Time
09/27/2023 10:43:03 AM

Service Manager > cm_tag Policies > Edit Policy

Policy TypeAccess

Policy ID*4

Policy Name*EXPIRES_ON*i*

EnabledNormal

Policy LabelSelect...|v

TAG *EXPIRES_ONx|v

DescriptionPolicy for data with EXPIRES_ON tag

Audit Logging*Yes

Add Validity Period

Policy Conditions :+
No Conditions

Allow Conditions:

hide

Select RolesSelect GroupsSelect UsersPolicy ConditionsPermissions

Select...|v

Select...|v

Select...|v

Add Conditions+
+

Add Permissions+
+

x

+

Exclude from Allow Conditions:

Select RolesSelect GroupsSelect UsersPolicy ConditionsPermissions

Select...|v

Select...|v

Select...|v

Add Conditions+
+

Add Permissions+
+

x

+

Deny All Other Accesses: *False

Deny Conditions:

hide

Select RolesSelect GroupsSelect UsersPolicy ConditionsPermissions

Select...|v

public xX|v

rangerlookup xX|v

Accessed after expiry_data (yes/no)? : yes

ADLSKUDUSGGSHDFSHEBASEHIVEYARNKNOXKMSOLARKAFKANIFINIFI-REGISTRYATLASOZONESCHEMA-REGISTRYKAFKA-KCONNECT

x

+

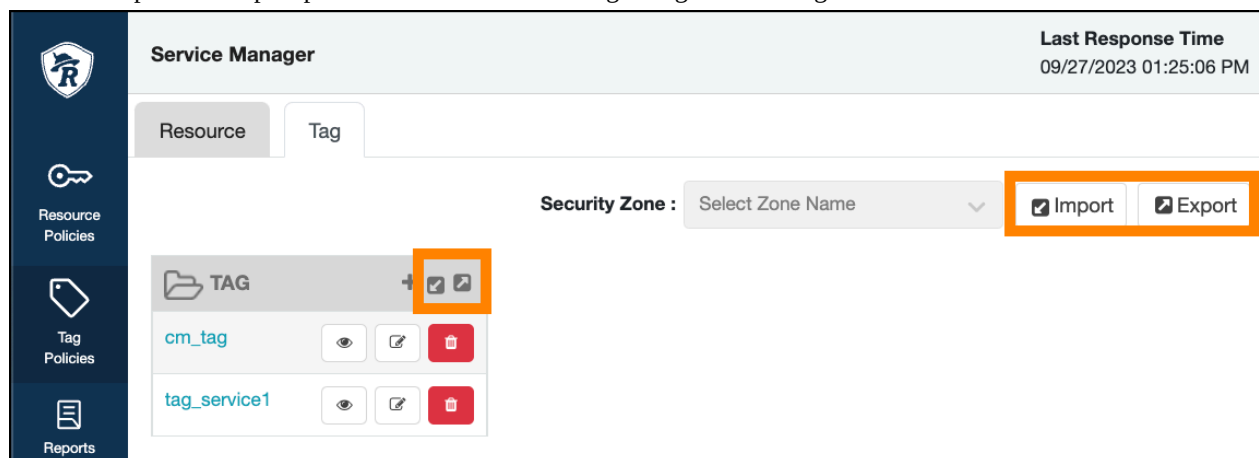
3. We can see that the default EXPIRES_ON policy denies access to all users, and for all components, after the expiry date specified in the Atlas tag attribute.

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 import or export 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 the Ranger Admin UI.

Interfaces

You can import and export policies from Service Manager Tag Policies Tag :



You can also export policies from Reports:

Reports

Last Response Time
09/21/2023 09:08:09 AM

Search Criteria

Policy Name

Enter Policy Name

Policy Type

Access

Component

Resource

Enter Resource Name

Policy Label

Zone Name

Select Zone Name

Search By

Group

Select...

Search

Excel file

CSV file

JSON file

Export

HDFS

Policy ID	Policy Name	Policy Label	Resources	Policy Type		
1	all - path	--	path: /*	Access	Enabled	--
2	kms-audit-path	--	path: /ranger/audit/...	Access	Enabled	--

HBASE

Policy ID	Policy Name	Policy Label	Resources	Policy Type	Status	Zone Name
5	all - table, column...	--	column-family: * column: * table: *	Access	Enabled	--

Table 72: Export Policy Options

	Service 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 Service 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+

**Note:**

Partial policy import is not supported.

Related Information

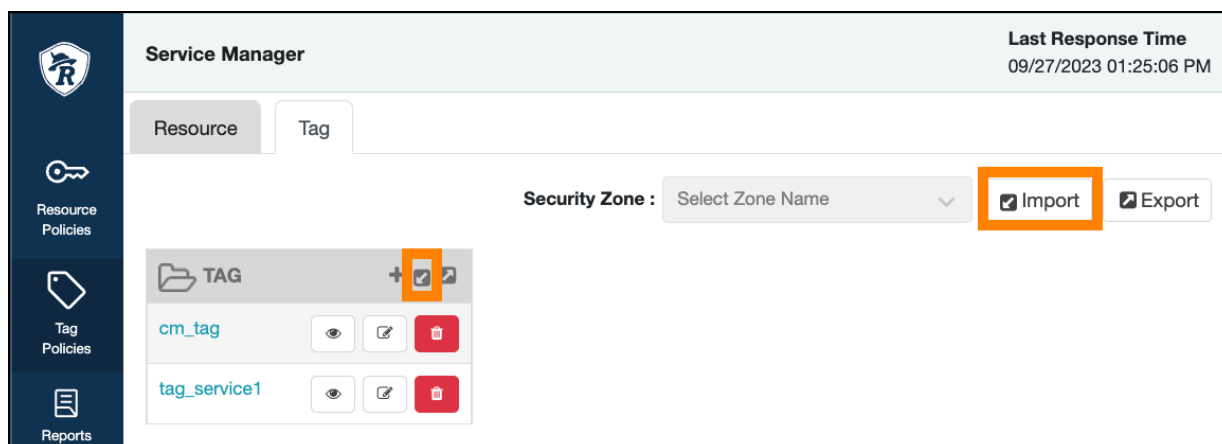
[Importing and exporting resource-based policies](#)

Import tag-based policies

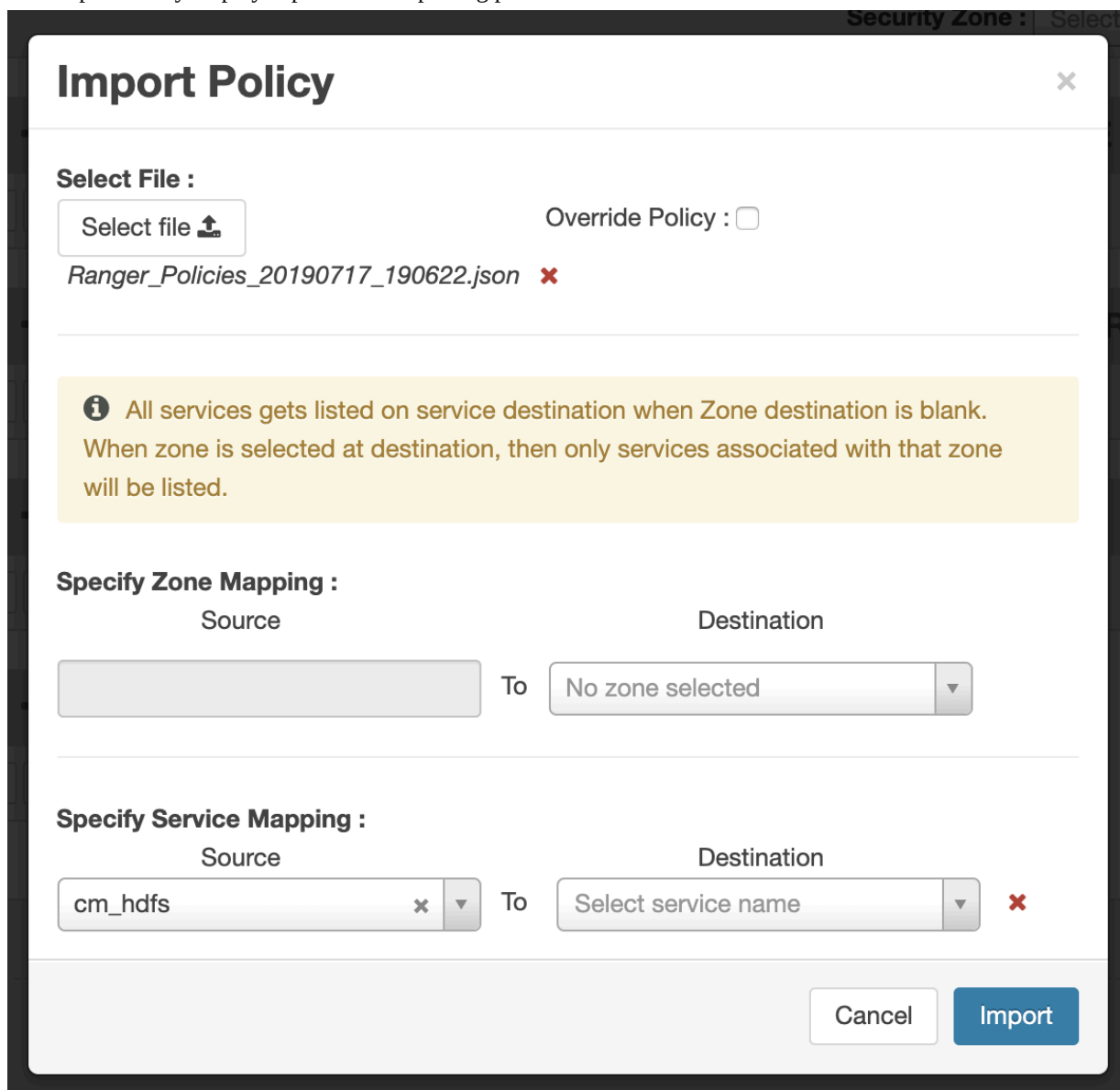
How to import tag-based policies.

Procedure

1. On Service Manager Tag Policies Tag , click one of the Import icons:



The Import Policy displays options for importing policies.



2. Select the file to import.

You can only import policies in JSON format.

3. (Optional) Configure the import operation:

- The Override Policy option deletes all policies of the destination repositories.
- Zone Mapping – when no destination is selected, all services are imported. When a destination is selected, only the services associated with that security zone are imported.
- Service Mapping maps the downloaded file repository, i.e. source repository to destination repository. You can use the red x symbols to remove services from the import. Scroll down to view all service mappings.

Import Policy

Specify Zone Mapping :

Source	To	Destination
	To	No zone selected

Specify Service Mapping :

Source	To	Destination
cm_hdfs	To	cm_hdfs
cm_hbase	To	cm_hbase
cm_yarn	To	cm_yarn
cm_hive	To	cm_hive
cm_knox	To	cm_knox
cm_storm	To	cm_storm

Cancel Import

4. Click Import.

A confirmation message appears after the file is imported.

Related Information

[Export tag-based policies](#)

Export tag-based policies

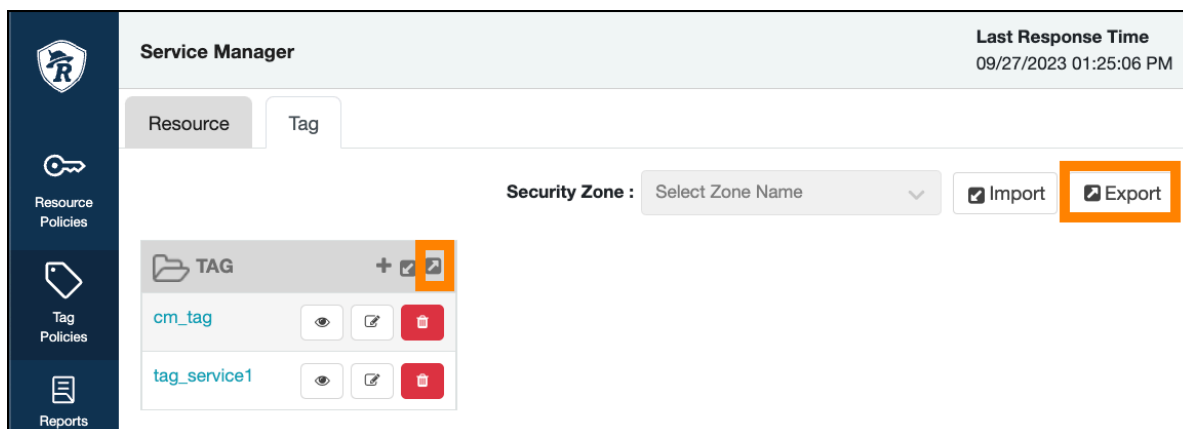
How to export all tag-based policies.

About this task

You can only export policies in JSON format from the Tag-based policies page. If you would like to export in Excel or CSV format, export the policies from the Reports page drop-down menu.

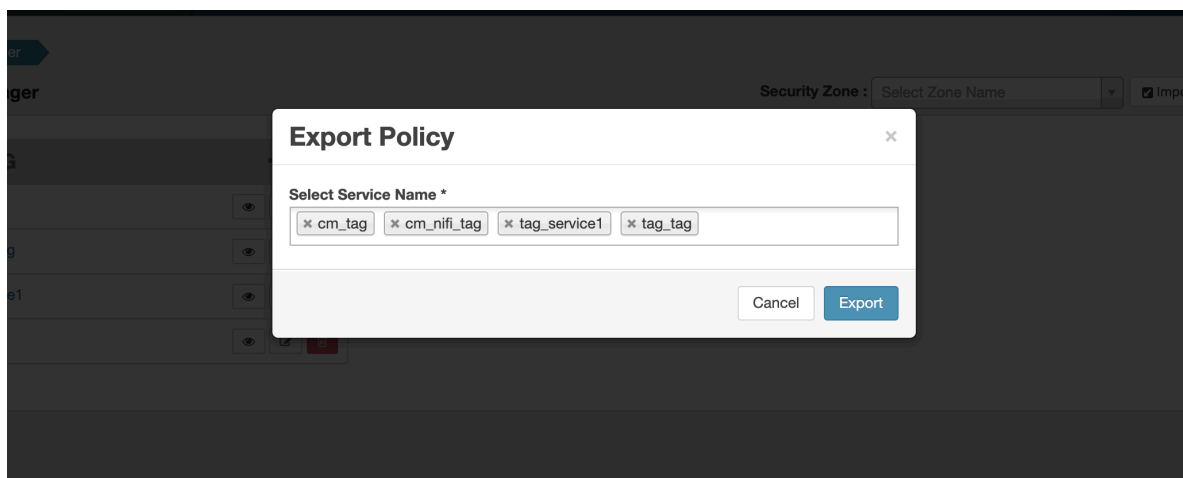
Procedure

- On Service Manager Tag Policies Tag, click one of the Export icons:
 - Click the Export button or icon:



Export Policy displays options for exporting policies.

- Remove components or specific services, then click Export.



- The file downloads in your browser as a JSON file.

- From Reports:
 - Filter Component to tag and click Search.
 - (Optional) Apply filters before exporting the file.
 - Open the Export drop-down menu:

The screenshot shows the 'Reports' section of the Cloudera interface. At the top right, it says 'Last Response Time 09/27/2023 01:56:48 PM'. Below this is a 'Search Criteria' section with several input fields: 'Policy Name' (placeholder: Enter Policy Name), 'Policy Type' (dropdown: Access), 'Component' (dropdown: tag x), 'Resource' (placeholder: Enter Resource Name), 'Policy Label' (dropdown), and 'Zone Name' (placeholder: Select Zone Name). There is also a 'Search By' section with a 'Group' dropdown and a 'Select...' button. A blue 'Search' button is at the bottom of the search criteria section. To the right of the search criteria is an 'Export' button with a dropdown menu showing 'Excel file', 'CSV file', and 'JSON file'. Below the search criteria is a table titled 'TAG' with columns: Policy ID, Policy Name, Policy Label, Resources, Policy Type, Status, Zone, and Conditions. The table contains two rows of data.

Policy ID	Policy Name	Policy Label	Resources	Policy Type	Status	Zone	Conditions
4	EXPIRES_ON	--	tag: EXPIRES_ON	Access	Enabled	--	+
67	EXPIRES_ON	--	tag: EXPIRES_ON	Access	Enabled	--	+

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

Configuring Ranger TagSync with AtlasREST

Learn how to configure Ranger TagSync with AtlasREST source.

About this task

To configure Ranger TagSync, select Ranger TagSync on the configuration properties page, and then specify a TagSync source. You can use Atlas, AtlasREST, or a file as the TagSync source. This topic describes how to add AtlasREST as the source.

Procedure

- Login to Cloudera Manager.
- Go to Ranger Configuration .
- Add the following properties in Ranger Tagsync Advanced Configuration Snippet (Safety Valve) for conf/ranger-tagsync-site.xml:

```
ranger.tagsync.source.atlasrest=true
ranger.tagsync.source.atlasrest.username=<atlas_admin_username>
ranger.tagsync.source.atlasrest.password=<atlas_admin_password>
ranger.tagsync.source.atlasrest.endpoint=https://<atlas_host>:31443
ranger.tagsync.source.atlasrest.ssl.config.filename={{CMF_CONF_DIR}}/conf/
ranger-tagsync-policymgr-ssl.xml
ranger.tagsync.source.atlasrest.keystore.filename={{CMF_CONF_DIR}}/cm-au
to-host-keystore.jks
```


4. Also, add the following properties in Ranger Tagsync Advanced Configuration Snippet (Safety Valve) for `conf/atlas-application.properties`:

```
atlas.enableTLS=true
truststore.file=/var/lib/cloudera-scm-agent/agent-cert/cm-auto-global_t
ruststore.jks
keystore.file={{CMF_CONF_DIR}}/cm-auto-host_keystore.jks
```

5. Restart Ranger.

Enabling tag-based policies for Ranger RAZ

Enable tag-based authorization for Ranger Remote Authorization Service (RAZ) to extend access control beyond traditional resource-based policies using Apache Atlas classifications. This workflow requires synchronizing S3 metadata using the S3 extractor, classifying resources in Atlas, and creating tag-based policies in Ranger Admin for synchronized AWS resources.

About this task

By default, RAZ supports resource-based authorization policies. This means that you configure access control using specific policies for resource types, such as S3 policies for AWS buckets or ADLS policies for Azure blobs.

The Ranger ecosystem, also supports tag-based policies. By associating classification tags with AWS resources, such as buckets, within Apache Atlas, these tags are automatically synchronized to Ranger Admin through TagSync.

This capability allows you to create tag-based policies for S3 buckets directly in Ranger Admin. Since RAZ is a Ranger plugin, it can download these tag-based policies and perform authorization checks based on the associated tags, extending access control capabilities beyond traditional resource-based policies.

Before you begin

To successfully implement this configuration, you must have working knowledge and operational expertise in the following areas:

- AWS Identity and Access Management (IAM)
- Cloudera Manager administration
- Apache Atlas metadata and classifications
- Ranger Admin authorization policies
- RAZ functionality and configuration

Procedure

1. Configure the S3 extractor and pull metadata of S3 buckets into Atlas.

For instructions, see *S3 Extractor configuration*.

2. Associate resources with appropriate classification tags in Atlas.

After the S3 metadata is in Atlas, classify your resources, for example, S3 buckets, objects, by associating them with the appropriate Atlas classification tags that will drive your authorization policies. For instructions, refer to *Creating classifications*.

3. Create a tag-based policy in Ranger.

After the classification tags are synchronized from Atlas to Ranger Admin through TagSync, create the tag-based policies. For instructions, refer to *Adding tag-based policies*.

Related Information

[S3 extractor configuration](#)

[Creating classifications](#)

[Adding tag-based policies](#)

Handling inconsistent username and group name conventions for consistent authorization

This document explains how user and group names are processed to ensure that Ranger policies are applied correctly, leading to seamless access to data and resources.

About this task

Cloudera offers a standardized method for managing usernames and group names to ensure consistent and accurate authorization across all Cloudera services. This approach is particularly useful when dealing with diverse naming conventions, including special characters such as whitespace and slashes.

You often use a variety of naming conventions for users and groups in your identity providers, for example, Active Directory or LDAP. These conventions can include special characters that, if not handled consistently, can lead to potential inconsistencies in authorization and increased administrative overhead.

Earlier, it was necessary to manually set the following user/group name conversion properties in the `ranger-admin-site.xml` file, and these values had to match those configured in the Ranger Usersync component:

- `ranger.plugins.conf.ldap.username.caseconversion`
- `ranger.plugins.conf.ldap.groupname.caseconversion`
- `ranger.plugins.conf.mapping.username.handler`
- `ranger.plugins.conf.mapping.groupname.handler`
- `ranger.plugins.conf.mapping.regex.separator`
- `ranger.plugins.conf.mapping.username.regex`
- `ranger.plugins.conf.mapping.groupname.regex`

From Cloudera Runtime 7.3.2.0 onwards, support has been added to expose the following Ranger Usersync configurations on Cloudera Manager and in the `ranger-admin-site.xml` file to manage usernames and group names seamlessly to ensure consistent and accurate authorization:

- `ranger.usersync.ldap.username.caseconversion`
- `ranger.usersync.ldap.groupname.caseconversion`
- `ranger.usersync.mapping.username.handler`
- `ranger.usersync.mapping.groupname.handler`
- `ranger.usersync.mapping.regex.separator`
- `ranger.usersync.mapping.username.regex`
- `ranger.usersync.mapping.groupname.regex`

To handle inconsistent user and group naming conventions, perform the following steps:

Procedure

Configure the safety valve at service level.

- a) Go to Cloudera Manager <Service> Configuration .
- b) Set the `ranger.plugin.<serviceType>.supports.name.transformation` safety valve to the service-level configuration.
For example, `ranger.plugin.hive.supports.name.transformation`.

Create a time-bound policy

Ranger policy validity periods enable you to configure a policy to be effective for a specified time range. You can add a validity period to both resource-based and tag-based policies.

About this task

Time-bound policy use-case examples:

- To restrict access to sensitive financial information until the earnings release date.
- To 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).
- To block a certain group for a specific time (e.g., excluding temporary employees from writing on resources during the holiday season).

 **Note:** The following procedure shows how to create a time-bound resource-based policy. The procedure is essentially the same for a tag-based resource policy.

Procedure

1. On Service Manager Resource Policies , select a service.
2. On <Service_name> Policies, click Add New Policy.
3. Complete the fields on Create Policy.
4. Click Add Validity Period.
5. On Policy Validity Period, specify a start time, end time, and time zone. To add additional validity periods, click +. Click Save to save the specified validity periods.

Policy Validity Period

Start Date	End Date	Time Zone ⓘ	
11-01-2024 00:00:00 X	11-30-2024 00:00:00 X	Africa/Abidjan (GMT) X v	X

+

Close

The JSON format for Policy Validity Period appears as follows:

```
"validitySchedules":[{"startTime":"2024/11/01 00:00:00","endTime":"2024/11/30 00:00:00","timeZone":"Africa/Abidjan"}]
```

6. If you would like the policy to override all other policies during its validity period, select override.

Create Policy Last Response Time
09/27/2023 02:21:51 PM

Service Manager > cm_hbase Policies > Create Policy

Policy Details

Policy Type: **Access** Add Validity Period

Policy Name*: temp employee override Enabled **Override**

Policy Label: Select...

HBase Table*: sales Include

HBase Column-family*: Select... Include

HBase Column*: Select... Include

Description:

Audit Logging*: Yes

Allow Conditions: hide

Select Roles	Select Groups	Select Users	Permissions	Delegate Admin
Select...	temp_employees	Select...	Read	☐
	No options			

Save **Cancel**

7. Click Add.

Enabling IP-based access control in Ranger

Learn how to extend Apache Ranger to support authorization based on the location, for example, country/state/city from which the resource is accessed.

To enable this feature, you need to perform the following tasks:

1. Prepare a location data file containing IP address-to-location detail mappings.
2. Register a context-enricher hook that adds the location details to the request.
3. Register a policy condition to verify that the client location matches the locations specified in the policy.
4. Create or update Apache Ranger policies to specify the locations to allow or deny the access.

IP location data file

IP location data file is a text file containing comma-separated fields. Each line in the file contains the location details for a range of IP addresses. The format of the IP location data file is as follows:

- Each line consists of comma-separated fields.
- The first line is treated as a header, containing names for each field.
- Subsequent lines have location details for a range of IP addresses.
 - The first field is the start IP address of the range.
 - The second field is the end IP address of the range.
 - Other fields have the location data for the IP range specified in first two fields (inclusive)

- IP addresses should be specified as long integers, but the context-enricher can read addresses in dot-notation when `IPInDotFormat` for the client IP address is true.

Example:

```
IP_FROM,IP_TO,COUNTRY_CODE,COUNTRY_NAME,REGION,CITY
10.0.0.255,10.0.3.0,US,United States,California,Santa Clara
20.0.100.80,20.0.100.89,US,United States,Colorado,Broomfield
20.0.100.110,20.0.100.119,US,United States,Texas,Irving
```

This data format is similar to commercially available data from providers like IP2Location. Depending on the requirements, the data file can either be sourced from a commercial data provider (like IP2Location) or created with deployment-specific details.

Register context enricher

When the Apache Ranger plugin receives an authorization request, the request is passed through registered context enrichers. The context enrichers have access to various request details, including the user, resource accessed, access type, IP address of the accessor, and so on. The context enrichers can update the request context with additional information that can be used while evaluating Ranger policies.

Context enricher `RangerFileBasedGeolocationProvider` adds geolocation data to the request context based on the location details available in a data file. To register the context enricher for a component (like HDFS/Hive/HBase/..), you need to update the component's service-def by including the following:

```
"contextEnrichers": [
  {
    "enricher": "org.apache.ranger.plugin.contextenricher.RangerFileBasedGe
olocationProvider",
    "enricherOptions": {
      "FilePath": "/etc/ranger/geo/geo.txt",
      "IPInDotFormat": "true"
    }
  }
]
```

Ensure that the data file is available to the components at the location specified in the above registration (`/etc/ranger/geo/geo.txt` in this example).

When `RangerFileBasedGeolocationProvider` receives an authorization request, it locates the record in the IP location data for the client IP address specified in the request. If a record is found, each field in the record will be added to the request context.

The following example describes the details of context data being added by the enricher:

- Client IP address: 20.0.100.85
- Matching record in IP location data:
 - 20.0.100.80,20.0.100.89,"US","United States","Colorado","Broomfield"
- IP location data header:
 - IP_FROM,IP_TO,COUNTRY_CODE,COUNTRY_NAME,REGION,CITY
- Entries added to the request context:
 - LOCATION_COUNTRY_CODE=US
 - LOCATION_COUNTRY_NAME=United States
 - LOCATION_REGION=Colorado
 - LOCATION_CITY=Broomfield

Note that the name of context entries are the field names, prefixed with `LOCATION_`.

Register policy condition

Apache Ranger provides policy-condition hooks to execute custom conditions while evaluating authorization requests. To determine the authorization result, the Apache Ranger policy engine evaluates the policies that are applicable to the accessed resource. Only when various criteria like user/group, access-type, and policy conditions specified in the policy match the request, the policy engine uses the policy to determine the result.

Policy condition `RangerContextAttributeValueNotInCondition` returns true only when the specified request context value does not match the values specified in the policy. This can be used to check if the location in request context (which is populated by the context enricher detailed earlier) is outside the values specified in the policy, for example, to deny access to requests that originate outside of specified countries. To register the policy condition for a component (like HDFS/Hive/HBase/..), please update the component's service-def by including the following:

```
"policyConditions": [
  {
    "itemId": 3,
    "name": "location-outside",
    "label": "Accessed from outside of location?",
    "description": "Accessed from outside of location?",
    "evaluator": "org.apache.ranger.plugin.conditionevaluator.RangerContextAttributeValueNotInCondition",
    "evaluatorOptions": {
      "attributeName": "LOCATION_COUNTRY_CODE"
    }
  }
]
```

After this policy condition is registered with Ranger, the policy editing UI prompts for condition values to be used during evaluation, as shown in the following image:

If you want to use allow rules instead, you need to update the service-def with the `RangerContextAttributeValueInCondition` policy condition. For example,

```
"policyConditions": [
  {
    "itemId": 1,
    "name": "access-location-within",
    "evaluator": "org.apache.ranger.plugin.conditionevaluator.RangerContextAttributeValueInCondition",
    "evaluatorOptions": {
      "attributeName": "LOCATION_COUNTRY_CODE"
    },
    "validationRegex": "",
    "validationMessage": "",
    "uiHint": "{ \"isMultiValue\": true }",
    "label": "Accessed from within location?",
    "description": "Accessed from within location"
  }
]
```

Create a Hive authorizer URL policy

You can create a Hive Authorizer URL policy in Ranger that maintains Read and Write permissions for a location or folder.

About this task

Hive supports several commands that include URLs which refer to a current or future data location. Such locations must authorize end user access to that location. Currently, you can create a Ranger HDFS policy that grants "All" permissions for a location, recursively. If no such policy exists, HDFS authorization "falls back" to the current ACL that defines access to a location or folder. By default the value of the parameter is "hdfs;file;wasb;adl:". If you remove "hdfs:", access requests will be authorized against the HIVE URL policy and won't check for hdfs plugin or Hadoop ACL. This solution requires maintaining many policies or ACLs at the storage level. You can create a Hive Authorizer URL policy in Ranger that maintains Read and Write permissions for a location or folder.

To create a Hive Authorizer policy:

Procedure

1. In Cloudera Manager HIVE-1 Configuration Search . type ranger-hive.
2. In Hive Service Advanced Configuration Snippet (Safety Valve) for ranger-hive-security.xml, click +.
 - a) Under HIVE-1, in Name, type: ranger.plugin.hive.urlauth.filesystem.schemes.
 - b) In Value, type: file:
 - c) Click Save Changes.
3. In Cloudera Manager Hive_On_Tez-1 Configuration Search . type ranger-hive.
4. In Hive Service Advanced Configuration Snippet (Safety Valve) for ranger-hive-security.xml, click +.
 - a) Under HIVE_ON_TEZ-1, in Name, type: ranger.plugin.hive.urlauth.filesystem.schemes.
 - b) In Value, type: file:
 - c) Click Save Changes.
5. In HIVE-1 Actions , click Restart.
6. In HIVE_ON_TEZ-1 Actions , click Restart.

By default the value of the parameter is "hdfs;file;wasb;adl:". If you remove "hdfs:", access requests will be authorized against the HIVE URL policy and won't check for hdfs plugin or Hadoop ACL.
7. In Ranger Resource Policies Hadoop SQL , click Add New Policy.
8. In Policy Details, select URL, then type the URL represents the location or folder to which you want Ranger to authorize access: hdfs://<hostname>.root.hwx.site:8020/demo/data.

9. In Allow Conditions, select user(s), then choose Read and Write permissions, as shown in the following example:

Figure 7: Creating a Hive Authorizer URL Policy

The screenshot shows the 'Create Policy' interface. At the top, it says 'Service Manager > Hadoop SQL Policies > Create Policy'. The 'Policy Details' section includes:

- Policy Type:** Access (selected)
- Policy Name*:** Policy Name
- Policy Label:** Select...
- URL:** hdfs://<hostname>-root.hwx.site:8020/demo/d... (with a dropdown arrow)
- Description:** (empty text area)
- Audit Logging*:** Yes (selected)
- Enabled/Normal:** Enabled (selected)
- Recursive:** (selected)
- Add Validity Period:** (button)

 The 'Allow Conditions' section is expanded, showing a table with columns: Select Roles, Select Groups, Select Users, Permissions, Delegate Admin, and a delete button. The 'Select Users' column contains 'hive'. The 'Permissions' column has 'Read' and 'Write' buttons. The 'Delegate Admin' column has a checkbox. The delete button is red with an 'x'.

This policy allows the user to READ / WRITE into the location defined by the URL.

```
CREATE EXTERNAL TABLE IF NOT EXISTS STUDENT (student_ID INT, FirstName STRING, LastName
STRING, year STRING, Major STRING) COMMENT 'Student Names' ROW FORMAT DELIMITED FIELDS
TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/demo/data';
```

This will create a table reading data from the location /demo/data provided user will have the necessary READ permission to the location along with CREATE permission for table STUDENT

If the storage system is S3A or ADFS, then URL policy would be maintained for the scheme. For example, s3a://<folder>, abfs://<folder>.

If the storage system is Ozone, then the URL policy would be maintained for the scheme ofs://volume/bucket/[key | folder], o3fs://volume/bucket/[key | folder].

Hive supports URL policies for the following commands that have URLs defined for the respective location:

CREATE TABLE

external table location

ALTER TABLE LOCATION

new location

ALTER PARTITION LOCATION

new partition location

ALTER TABLE ADD PARTITION

for partition location

LOAD

input location

For additional information about creating Hive commands with URL, see <https://cwiki.apache.org/confluence/display/RANGER/Hive+Commands+to+Ranger+Permission+Mapping>.

Hive authorizer URL policy with hdfs default value

Learn about the performance issue you face when the `ranger.plugin.hive.urlauth.filesystem.schemes` configuration includes `hdfs:` as a default filesystem scheme.

The default value of the `ranger.plugin.hive.urlauth.filesystem.schemes` configuration is `hdfs:,file:,wasb:,adl:`. When the `ranger.plugin.hive.urlauth.filesystem.schemes` configuration includes `hdfs:` as a default filesystem scheme in both HMS and HiveServer2, and you perform a `CREATE EXTERNAL TABLE` operation with a `LOCATION` clause, if both the Ranger Hive and HDFS plugins are enabled, the Ranger Plugin Authorizer performs authorization checks on all files within the target directory. This results in performance degradation that scales up linearly with the number of files. The operation can take a significantly long time to complete or even fail with an error if the specified location contains a large number of subdirectories.

If you remove `hdfs:` from the configuration, the request will be authorized against the HIVE URL policy and will not check for the Ranger HDFS plugin or Hadoop ACL. This will lead to faster responses and less waiting time.



Note: This performance issue appears to stem from the large number of connections that Ranger opens to the HDFS NameNode for performing authorization checks on the directory structure. Specifically, when a `LOCATION` clause is present, Ranger validates permissions on the URI by invoking `FileUtils.isOwnerOfFileHierarchy()`, which recursively checks ownership and permissions for all files and subdirectories under the specified path. As the number of files and subdirectories increases, the time taken for this check grows accordingly, leading to potential delays or failures in the operation.

Showing Role|Grant definitions from Ranger HiveAuthorizer

You can use beeline to show the roles granted to users, groups, and roles.

About this task

You can create roles in Ranger or in Hive. You create roles in HIVE using `ROLE` commands, such as `CREATE ROLE`, `GRANT / REVOKE ROLE`. You can create roles in Ranger, using the Ranger Admin Web UI, if you have Admin permissions. See related links for more information about creating roles. The Hive2 command line interface Beeline returns role grant definitions for a specific principal, such as a user, group or role.

Before you begin

Roles must be defined before using beeline to show role|grant definitions.

Procedure

1. Run beeline, (the hive2 command line interface) on the Ranger host.

```
beeline -u jdbc:hive2://<ranger_host_name>
```

2. Enter valid syntax to return the role definitions for a specific principal.

Syntax

```
SHOW ROLE GRANT (USER|GROUP|ROLE)    principal_name;
where
principal_name is USER | GROUP | ROLE name
```

Results

Beeline outputs query results, as shown in following examples:

Example

SHOW ROLE GRANT USER HDFS -> show roles for user "hdfs"

```
0: jdbc:hive2://rm-ranger-3.rm-ranger.root.hw> show role grant user hdfs;
INFO : Compiling command(queryId=hive_20211109235258_b9211cfe-0e78-47d7-8a2a-1b611ddcd18): show role grant user hdfs
INFO : Semantic Analysis Completed (retrial = false)
INFO : Created Hive schema: Schema(fieldSchemas:[FieldSchema(name:role, type:string, comment:from deserializer), FieldSchema(name:grant_option, _time, type:bigint, comment:from deserializer), FieldSchema(name:grantor, type:string, comment:from deserializer)], properties:null)
INFO : Completed compiling command(queryId=hive_20211109235258_b9211cfe-0e78-47d7-8a2a-1b611ddcd18); Time taken: 0.02 seconds
INFO : Executing command(queryId=hive_20211109235258_b9211cfe-0e78-47d7-8a2a-1b611ddcd18): show role grant user hdfs
INFO : Starting task [Stage-0:DDL] in serial mode
INFO : Completed executing command(queryId=hive_20211109235258_b9211cfe-0e78-47d7-8a2a-1b611ddcd18); Time taken: 0.008 seconds
INFO : OK
```

role	grant_option	grant_time	grantor
ITManager	false	1636501912000	

Example

SHOW ROLE GRANT ROLE -> show roles for role "ITManagers"

```
0: jdbc:hive2://rm-ranger-3.rm-ranger.root.hw> show role grant role ITManager;
INFO : Compiling command(queryId=hive_20211109235607_2d543c50-7c7b-4d54-bed0-159f67c24079): show role grant role ITManager
INFO : Semantic Analysis Completed (retrial = false)
INFO : Created Hive schema: Schema(fieldSchemas:[FieldSchema(name:role, type:string, comment:from deserializer), FieldSchema(name:grant_option, _time, type:bigint, comment:from deserializer), FieldSchema(name:grantor, type:string, comment:from deserializer)], properties:null)
INFO : Completed compiling command(queryId=hive_20211109235607_2d543c50-7c7b-4d54-bed0-159f67c24079); Time taken: 0.177 seconds
INFO : Executing command(queryId=hive_20211109235607_2d543c50-7c7b-4d54-bed0-159f67c24079): show role grant role ITManager
INFO : Starting task [Stage-0:DDL] in serial mode
INFO : Completed executing command(queryId=hive_20211109235607_2d543c50-7c7b-4d54-bed0-159f67c24079); Time taken: 0.007 seconds
INFO : OK
```

role	grant_option	grant_time	grantor
Managers	false	1636502074000	
TeamLeads	false	1636502122000	

Related Information

[Apache documentation on Role operations](#)

[Adding a role through Hive](#)

[Adding a role through Ranger](#)

Ranger Security Zones

Ranger security zones let you organize service resources into multiple security zones.

Security Zones allow carving/bucketing of resources in a service into multiple zones for better administration of security policies. Defining Security Zones can enable multiple administrators to setup security policies for a service – based on the zones to which they have been granted administration rights.

Security Zones Administration

A Security Zone enables a Ranger administrator to separate resource policies into different administrative zones.

What is a Security Zone?

Security Zones help simplify security policy administration, and allow a limited amount of policies to be checked when doing authorization against certain resources. Only policies under a particular zone that contains the requested resource are loaded and checked by Ranger.

For example, let us consider two security zones: finance and sales:

- Security zone finance includes all content in a Hive database named finance.

- Security zone sales includes all content in a sales database.
- Policies defined in a security zone apply only to resources of that zone.
- A zone can be extended to include resources from multiple services such as HDFS, Hive, HBase, Kafka, etc. Extending a zone across multiple services allows zone administrators to set up policies for resources owned by their organization across multiple services.

For example:

```
Zone: finance
  service: prod_hdfs; path=/finance/*, /taxes/*
  service: prod_hive; database=finance
  service: prod_kafka; topic=FIN_*
  service: test_hadoop; path=/finance/*, /taxes/*
Zone: sales
  service: prod_hadoop; path=/sales/*
  service: prod_hive; database=sales
  service: prod_kafka; topic=SALES_*
```

- As shown above, resources can be specified using wildcards (FIN_*, SALES_*).
- Sets of users and groups are designated as administrators in each security zone.
- Users are allowed to set up policies only in security zones in which they are administrators.
- A resource cannot map to more than one security zone. Ranger does not allow creating security zones that specify resources that match resources in another zone. For example, an attempt to update the finance zone in the above example with the HDFS path /sales/finance/* is not permitted, because this conflicts with the HDFS path /sales/* specified in the sales zone.
- A set of users and groups can be designated as administrators of a security zone. Administrators can create, update, and delete security policies for the resources in that security zone.
- A set of users and groups can be authorized to view audit logs for that security zone's resources. Other users are not allowed to view access-audit logs for that security zone's resources.
- The security zone name appears in the zonename column of the access-audit log.

Security Zone Administration

- Security zones can only be created, updated, or deleted by a user with the ROLE_SYS_ADMIN role in Ranger.
- Users can view, retrieve, and update policies only in security zones in which they have administrator privileges.
- Users can view/retrieve and cannot update zone policies for which they have zone auditor permission.

How are Security Zones Used in Authorization?

When a Ranger authorization plugin authorizes a resource access request, it first determines the zone in which the accessed resource resides. If the resource matches a security zone, only the policies of that security zone are used to authorize the access. If resource does not match any security zone, the policies in the default (unnamed) security zone are used to authorize the access.

Tag-based Policies in Security Zones

In a given service, each security zone can be configured to use tag-based policies from a specific security zone in a tag-service. This enables different tag-based authorization policies to be used, based on the security zone of the resource.

Audit Logs

Audit logs generated by Ranger include the name of the security zone in which the accessed resource resides. Only users who have been assigned as an Admin or Auditor for the security zone are allowed to view the audit logs.

Security Zones Example Use Cases

Four example use cases for administering security zones.

Based on the following example:

```

Zone: finance
    service: prod_hdfs; path=/finance/*, /taxes/*
    service: prod_hive; database=finance
    service: prod_kafka; topic=FIN_*
    service: test_hadoop; path=/finance/*, /taxes/*
Zone: sales
    service: prod_hadoop; path=/sales/*
    service: prod_hive; database=sales
    service: prod_kafka; topic=SALES_*

```

Use case 1 : Access HDFS path using zone policy

For example, let us access hdfs path using unixuser1 user from finance zone.

Finance zone resource:

Ranger Service : prod_hdfs
Resource : /finance/*

Finance zone policy:

Resource Path : /finance/*
User : unixuser1
Permission : read, write, execute

Now, when unixuser1 user tries to create dir in /finance dir, Ranger checks for zone with resource /finance and policy for that user in that zone and then allows access for that user. Also, access-audit logs for that operation appear in the Ranger Admin Web UI, Access Audit tab.

Use case 2 : Hive access policy and tag masking policy

For example, we want to manage access policies and masking policy for taxation-related information in multiple finance databases for an organization.

Zone Resource :

Zone Tag service: cm_tag
Ranger Service : prod_hive
Resource :
Database : finance

Zone policy resource

Tag policy
resource:TDS
Hive policy
Resource :
Database : finance

Now, the Admin and security zone admin can create access policies and masking policies for all the resources associated with tag TDS and as and when new tables on Hive/HBase are created for saving any taxation related data. They can associate a TDS tag with a related Hive/HBase column. This will enable zone admin to create policies for masking the confidential data of its organization.

Use case 3 : Knox topologies

For example, suppose we want to manage access to a service. We can manage access to a service using topology.

Zone Resource :

Ranger Service : prod_knox

Resource:

Knox Topology:cdp-proxy-api

Knox Service:WEBHDFS

Zone deny policy Resource:

Knox Topology:cdp-proxy-api

Knox Service:WEBHDFS

Without a security zone, access to webhdfs is allowed since the default policy has a 'public' group in it.

Use case 4 : Import and export of zone policy

We can import and export zone policies from stage to prod.

Suppose we want to have the same policy in production that exists on stage. We can export the zone policy from the stage where the exported json has a zone name as a parameter in the json. While importing, we can map the zone name of stage to prod and then import the policies.

Adding a Ranger security zone

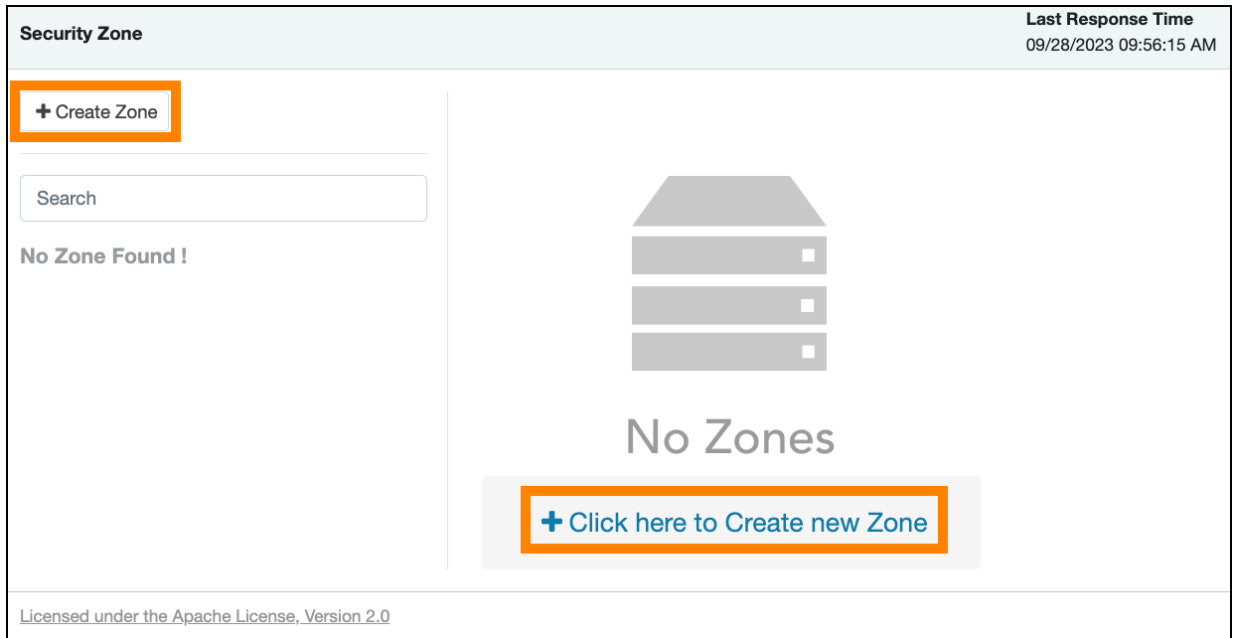
Ranger administrator users can create a Security Zone using the Ranger Admin Web UI.

Procedure

1. In Ranger Admin Web UI Service Manager , click Security Zone.

Security Zone displays existing security zones. If no zone exists, two options for creating a new zone display.

2. On Security Zone, click + Create (new) Zone.



Create Zonedisplays options for creating a new security zone.

Create Zone

Last Response Time
09/28/2023 11:25:39 AM

[Security Zone](#) > Create Zone

Zone Details:

Zone Name *

security-zone-1

Zone Description

Zone Administration:

Admin Users

Audrey x

Admin Usergroups

Select Group

Auditor Users

Audrey x

Auditor Usergroups

auditors x

Services:

Select Tag Services

cm_tag x

Select Resource Services *

cm_hive x

Service Name	Service Type	Resource
cm_hive	hive	+

Save

Cancel

Licensed under the Apache License, Version 2.0

3. On Create Zone, enter the following information:

Table 73: Zone Details

Field	Description
Zone Name	The security zone name.
Zone Description	An optional description.

Table 74: Zone Administration

Field	Description
Admin Users	The Admin users for the security zone.
Admin Usergroups	The Admin user groups for the security zone.
Auditor Users	The Auditor users for the security zone.

Field	Description
Auditor Usergroups	The Auditor user groups for the security zone.

Table 75: Services

Label	Description
Select Tag Services	Select tag-based services for the security zone.
Select Resource Services	Select resource-based services for the security zone.

4. Selected services are listed in Services. To add resources for each selected service, click + in the Resource column for the applicable service.

Create Zone

Last Response Time
09/28/2023 11:25:39 AM

[Security Zone](#) > Create Zone

Zone Details:

Zone Name *

security-zone-1

Zone Description

Zone Administration:

Admin Users

Audrey x

Admin Usergroups

Select Group

Auditor Users

Audrey x

Auditor Usergroups

auditors x

Services:

Select Tag Services

cm_tag x

Select Resource Services *

cm_hive x

Service Name	Service Type	Resource
cm_hive	hive	+

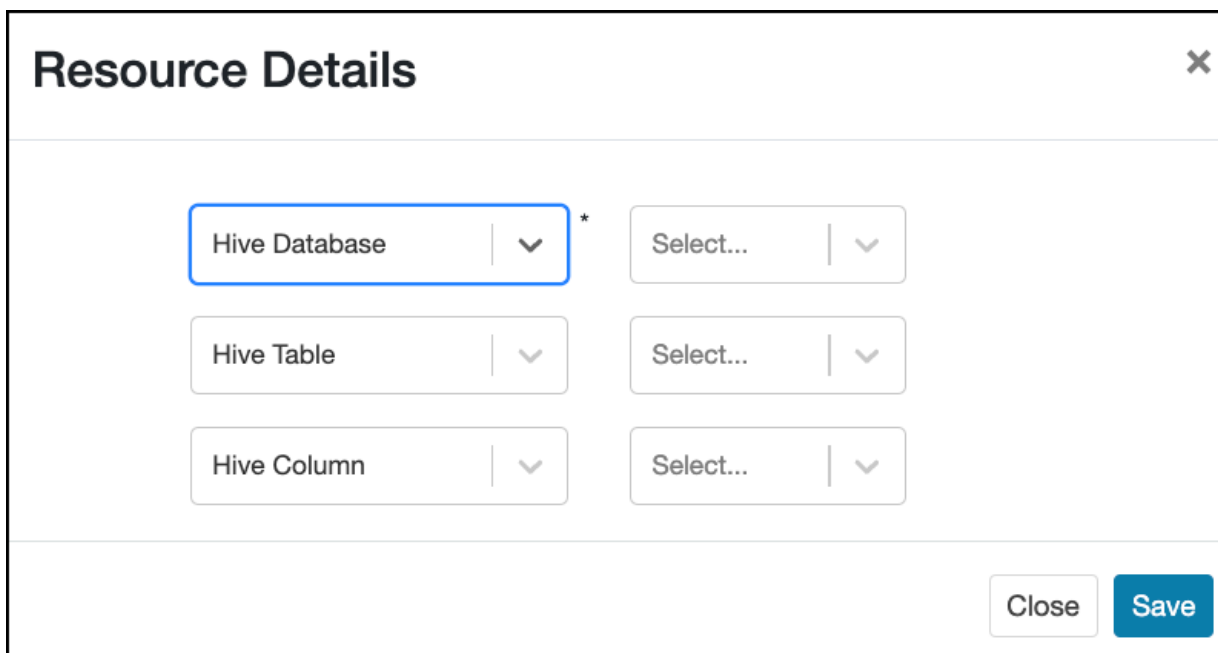
Add Resource

Save

Cancel

Licensed under the Apache License, Version 2.0

5. Use Resource Details to specify resources for the service, then click Save.



The 'Resource Details' dialog box is shown. It has a title bar with a close button (X). The main area contains three rows of dropdown menus. The first row has 'Hive Database' selected in the first dropdown and 'Select...' in the second, with an asterisk (*) next to the first dropdown. The second row has 'Hive Table' in the first dropdown and 'Select...' in the second. The third row has 'Hive Column' in the first dropdown and 'Select...' in the second. At the bottom right, there are 'Close' and 'Save' buttons.

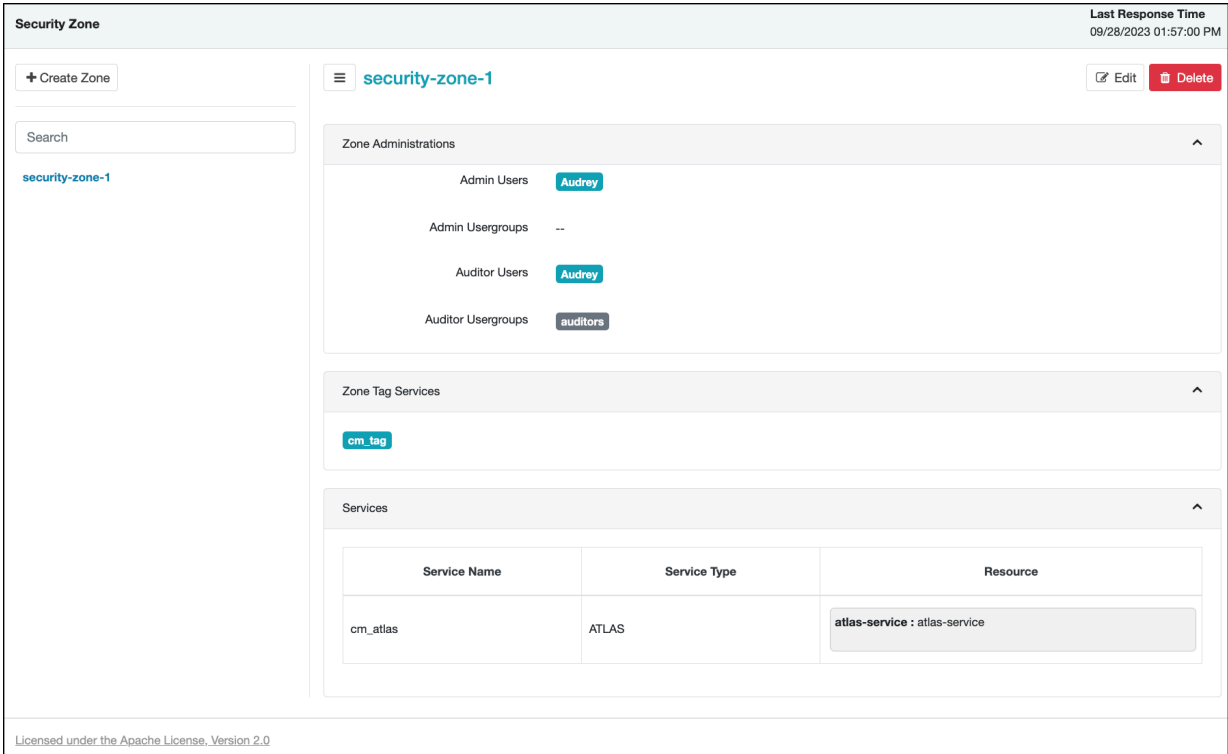
Selected resources appear as Resources for each Service in Create Zone.



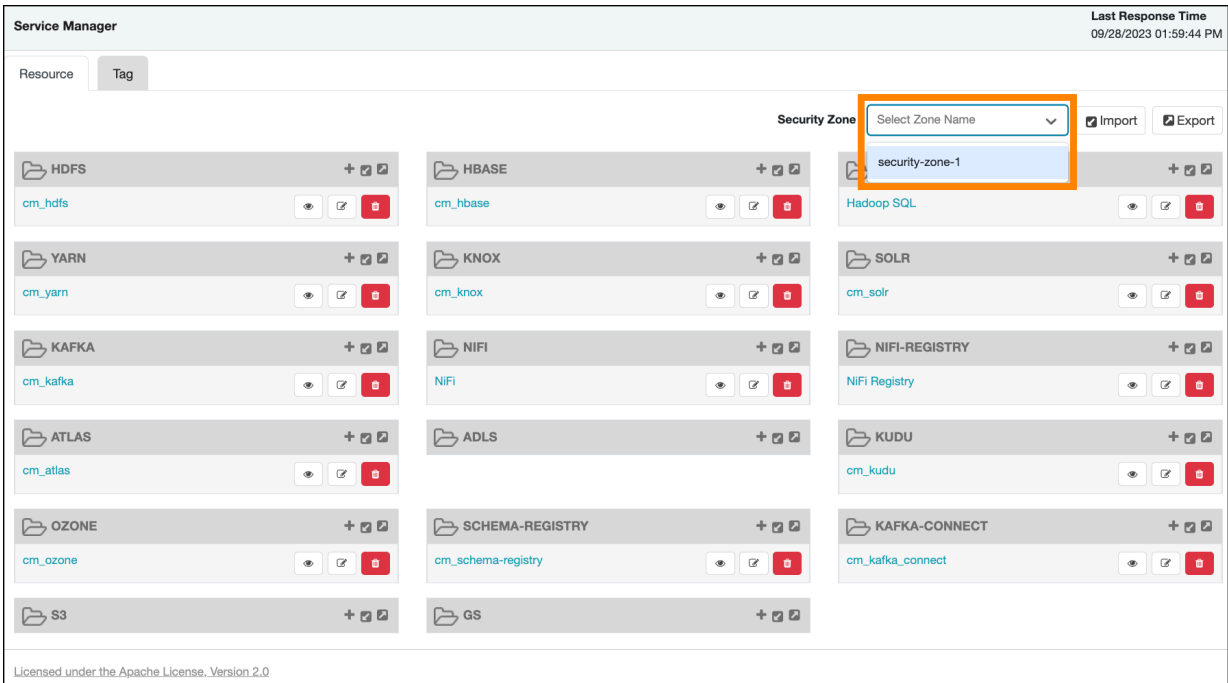
Note: The solr plugin supports fine-grained authorization similar to legacy Sentry privileges. A part of this support introduces the following new solr resources: collection, config, schema and admin. To perform any operation on a collection, a user also requires admin-level permission. To create a security zone for with the solr service that includes a collection resource, you must also add an admin resource. Currently, if you use one solr service to create a security zone that has a collection resource (and therefore includes an admin resource) you cannot create another solr security zone using another collection. (currently, only one admin resource can be used per solr security zone). This limitation exists for security zones in Cloudera Runtime-7.1.8.

6. Click Save at the bottom of Create Zone to save the new security zone.

7. The new security zone is listed on the Security Zone page.



8. To edit a security zone, click the security zone name in the Security Zones list, then click Edit.
9. After security zones have been created, you can use the Security Zone selection box on the Service Manager page to display the services assigned to the selected security zone. A Zone Name column appears in the table on Audit Access , and also in Service Manager Reports .



Administering Ranger Reports

You can use **Service Manager Reports** to help manage policies more efficiently as the number of policies increases. Reports lists all resource-based and tag-based policies.

Reports

Last Response Time
10/02/2023 09:15:04 AM

Search Criteria

Policy Name

Enter Policy Name

Policy Type

Access

Component

Resource

Enter Resource Name

Policy Label

Zone Name

Select Zone Name

Search By

Group

Select...

Search

Export

HDFS

Policy ID	Policy Name	Policy Label	Resources	Policy Type	Status	Zone Name	Policy Conditions
1	all - path	--	path: /	Access	Enabled	--	+
2	kms-audit-path	--	path: /ranger/audit/kms	Access	Enabled	--	+
3	hbase-archive	--	path: /hbase/archive	Access	Enabled	--	+

View Ranger reports

How to view reports for Ranger policies.

To view reports for one or more policies, select **Service Manager Reports** .

- To view Allow Condition details for each policy, click + in the Allow Conditions column. You can use the same method to view details for other policy conditions (Allow Exclude, Deny Conditions, etc.).
- To edit a policy from the Reports page, click the Policy ID.

Reports

Last Response Time
10/02/2023 09:15:04 AM

Search Criteria

Policy Name

Policy Type

Access

Component

Resource

Policy Label

Zone Name

Select Zone Name

Search By

Group

Select...

HDFS

Policy ID	Policy Name	Policy Label	Resources	Policy Type	Status	Zone Name	Policy Conditions
1	all - path	--	path: /	Access	Enabled	--	+
2	kms-audit-path	--	path: /ranger/audit/kms	Access	Enabled	--	+
3	hbase-archive	--	path: /hbase/archive	Access	Enabled	--	+

Search Ranger reports

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

You can search based on:

- Policy Name – The policy name.
- Policy Type – The policy type (Access, Masking, or Row Level Filter).
- Policy Label – The policy label.
- Component – The policy resource or tag component.
- Resource – The resource path used when creating the policy.
- Zone Name – The security zone name.
- Group, Username – The group or user name assigned to the policy.

Reports

Last Response Time
10/02/2023 09:15:04 AM

Search Criteria

Policy Name

Enter Policy Name

Policy Type

Access

Component

Resource

Enter Resource Name

Policy Label

Zone Name

Select Zone Name

Search By

Group

Select...

Search

Export

HDFS

Policy ID	Policy Name	Policy Label	Resources	Policy Type	Status	Zone Name	Policy Conditions
1	all - path	--	path: /	Access	Enabled	--	+
2	kms-audit-path	--	path: /ranger/audit/kms	Access	Enabled	--	+
3	hbase-archive	--	path: /hbase/archive	Access	Enabled	--	+

Export Ranger 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

Reports

Last Response Time
10/02/2023 09:15:04 AM

Search Criteria

Policy Name

Enter Policy Name

Policy Type

Access

Component

Resource

Enter Resource Name

Policy Label

Zone Name

Select Zone Name

Search By

Group

Select...

Search

Export

HDFS

Policy ID	Policy Name	Policy Label	Resources	Policy Type	Status	Zone Name	Policy Conditions
1	all - path	--	path: /	Access	Enabled	--	+
2	kms-audit-path	--	path: /ranger/audit/kms	Access	Enabled	--	+
3	hbase-archive	--	path: /hbase/archive	Access	Enabled	--	+

Excel file

CSV file

JSON file

Related Information

[Export tag-based policies](#)

[Export resource-based policies for a specific service](#)

[Export all resource-based policies for all services](#)

Using Ranger client libraries

Ranger now supports clients written in java and python which enable applications to access Ranger REST APIs programmatically. Using client library code simplifies access using java or python, compared with making direct HTTP requests to Ranger REST APIs.

Summary

Ranger client libraries:

- Provide idiomatic, hand-written code in Java and Python, making Ranger REST APIs simple and intuitive to use.
- Handle all low-level details of communication with the server including complexities involved in JSON parsing.
- Support installing the python client using the familiar package management tool pip.

Table 76: Ranger Client Installation Repo and Library Reference Links

Language	Installation	Library Reference
java	github source repository	java library reference
python	github source repository	python library reference

Authentication

The Apache Ranger release 2.2 client supports two authentication types:

- Basic authentication (username/password)
- Kerberos authentication

Java client prompts for the authentication mode to be used at runtime. For Kerberos-based authentications, a principal and keytab file path is required.

SSL

Java and Python clients support SSL/TLS-enabled ranger. To connect to HTTPS ranger using java client, provide the path to the SSL configuration file, as shown in this example:

```
$ ./run-sample-client.sh -n <ranger_admin_url>
SSL Configuration File: /path/to/config.xml
```

Sample SSL configuration file which requires values to be populated:

```
<configuration>
  <property>
    <name>xasecure.policymgr.clientssl.truststore</name>
    <value></value>
  </property>
  <property>
    <name>xasecure.policymgr.clientssl.truststore.credential.file</name>
    <value></value>
  </property>
  <property>
    <name>xasecure.policymgr.clientssl.truststore.type</name>
    <value></value>
  </property>
</configuration>
```

Environment variables

The Java client requires that you initialize the following environment variables:

```
$ export JAVA_HOME=/usr/java/<jdk_version>/bin
$ export PATH=$PATH:$JAVA_HOME
$ export HADOOP_CREDSTORE_PASSWORD=<hadoop_credstore_password>
```

Using session cookies to validate Ranger policies

Apache Ranger REST Client uses cookie sessions to download policies, tags and roles from Ranger Admin.

In earlier versions, each Ranger plugin used a kerberos login to request a ticket granting ticket (TGT) from the KDC/AD server in order to download policies, tags and roles. This caused high traffic levels when multiple Ranger plugins requested downloads.

Ranger Admin now supports cookie-based sessions. The flag used to enable cookie sessions, `ranger.plugin.<service-name>.policy.rest.client.cookie.enabled`, where `<service-name>` is the name of the service for which a Ranger plugin is enabled, such as `hive`, `solr`, or `kafka`, is set to "enabled" by default.

To check whether the cookie session is used, open the Ranger Admin `access.log` in the `/var/log/ranger/admin` folder. Any policy, tag, or role download call to Ranger Admin displays either a 200 or 304 value as response status. A 401 value for response status indicates the call to the KDC server for a TGT for authentication at service start or when the session cookie expires.

Configure optimized rename and recursive delete operations in Ranger Ozone plugin

You can enable a performance-optimized authorization approach for rename and recursive delete operations in the Ranger Ozone plugin.

About this task

Ozone introduced support for FSO (`FILE_SYSTEM_OPTIMIZED`) bucket layout. The FSO bucket layout is a Hierarchical FileSystem namespace view with directories and files. Similar to HDFS, with the FSO bucket layout, Ozone has an efficient directory rename and delete operations. Ranger supports not only authorization for rename and recursive delete operations but also provides an option to enable a performance-optimized solution when these operations are performed on a directory containing a large set of subpaths (directories/files) within it.

The property name is `ranger.plugin.ozone.optimized.subaccesspath.enabled`.

The default is set to false.

To enable authorization for rename and recursive delete operations in the Ranger Ozone plugin:

Procedure

1. In Cloudera Manager Ozone Ozone Manager Configuration Search, type `ranger-ozone-security.xml`.

2. In Ozone Service Advanced Configuration Snippet (Safety Valve) for ranger-ozone-security.xml, click +.
 - a) Under Ozone, in Name, type: ranger.plugin.ozone.optimized.subaccesspath.enabled.
 - b) In Value, type: true.
 - c) Click Save Changes.



Note: If the ranger.plugin.ozone.optimized.subaccesspath.enabled property is set to true, then the behavior for recursive operations (for example, command such as `rm -rf`) is as follows:

- Parent-level recursive policy is mandatory for the key being accessed.
- Tag-based deny policies are not supported for subpaths.
- Multi-resource policies are not supported for subpaths.

3. In Ozone Actions, click Restart.

Results

Ranger not only authorizes rename and recursive delete operations, but also provides an option to enable a performance-optimized solution when these operations are performed on a directory containing a large set of subpaths (directories/files) within it.

How to optimally configure Ranger RAZ client performance

How to find and set configurations for RAZ client performance.

About this task

This topic presents a set of best- balanced configs for Ranger RAZ clients, based on our past experience and testing. The majority of the time, the default config set for Ranger RAZ client is sufficient. We do not recommend any update in these default Ranger RAZ client configs, as it might result in unwanted outcomes.

In some cases, you may want to update these configs to optimize/suit your environment and if you are willing to take risks.

The following table lists some useful configs and short descriptions, which will help you optimize.

Procedure

1. Go to Cloudera Manager HDFS Configuration .
2. In Search, type core-site.xml.
3. Add the following configurations in Cluster-wide Advanced Configuration Snippet (Safety Valve) for core-site.xml:

Table 77: RAZ client configuration properties and default values

Configuration Name	Description	Default Value
ranger.raz.client.max.retry	Ranger Raz client retries (high layer), must not be negative	3
ranger.raz.client.rest.client.connection.pool.retry-count	Lower layer retries, must not be negative	3
ranger.raz.client.rest.client.connection.timeoutMs	Connection timeout in milliseconds	120000
ranger.raz.client.rest.client.read.timeoutMs	Read timeout in milliseconds	30000



Note: Changes in these default values might result in reduction of stability of a job (for example, yarn/spark) completion.