

CDP Data Center 7.1.1

## Schema Registry Security

Date published: 2020-06-12

Date modified: 2020-06-12

The Cloudera logo is displayed in a bold, orange, sans-serif font. The word "CLOUDERA" is written in all caps, with a stylized 'E' that has a horizontal bar extending to the right.

<https://docs.cloudera.com/>

# Legal Notice

© Cloudera Inc. 2025. 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

<b>TLS Encryption.....</b>	<b>4</b>
TLS Certificate Requirements and Recommendations.....	4
Configure TLS Encryption Manually for Schema Registry.....	5
Schema Registry TLS Properties.....	6
<b>Schema Registry Authorization through Ranger Access Policies.....</b>	<b>8</b>
Pre-defined Access Policies for Schema Registry.....	8
Add the user or group to a pre-defined access policy.....	9
Create a Custom Access Policy.....	10

## TLS Encryption

Transport Layer Security (TLS) is an industry standard set of cryptographic protocols for securing communications over a network. To encrypt sensitive information between the Cloudera Manager Server and cluster hosts, you must enable TLS.

You can choose to enable Auto-TLS or manually configure TLS.

Auto-TLS simplifies the process of enabling and managing TLS encryption on your cluster. When you enable Auto-TLS, an internal certificate authority (CA) is created and certificates deployed automatically across all cluster hosts. For more information on Auto-TLS, see *Configuring TLS Encryption for Cloudera Manager Using Auto-TLS*.

If you choose to enable TLS manually, you must create the TLS certificates making sure the certificates meet the requirements. Then configure Cloudera Manager and Schema Registry.

### Related Information

[Configuring TLS Encryption for Cloudera Manager Using Auto-TLS](#)

## TLS Certificate Requirements and Recommendations

If you choose to manually configure TLS, you must use your own certificates. The certificates must meet the requirements listed here.

### Certificate Requirements

Verify the following minimum requirements:

- The KeyStore must contain only one PrivateKeyEntry. Using multiple private keys in one KeyStore is not supported.
- The KeyStore password and key/certificate password must be the same or no password should be set on the certificate.
- The unique KeyStores used on each cluster node must use the same KeyStore password and key/certificate password. Ambari and Cloudera Manager do not support defining unique passwords per host.
- The X509v3 ExtendedKeyUsages section of the certificate must have the following attributes:
  - clientAuth - This attribute is for TLS web client authentication.
  - serverAuth - This attribute is for TLS web server authentication.
- The signature algorithm used for the certificate must be sha256WithRSAEncryption (SHA-256).
- The certificates must not use wildcards. Each cluster node must have its own certificate.
- Subject Alternate Names (SANs) are mandatory and should at least include the FQDN of the host.
- Additional names for the certificate/host can be added to the certificate as SANs.
  - Add the FQDN used for the CN as a DNS SAN entry.
  - If you are planning to use a load balancer, include the FQDN for the load balancer as a DNS SAN entry.
- The X509v3 KeyUsage section of the certificate must include the following attributes:
  - DigitalSignature
  - Key\_Encipherment

### Cloudera Recommendations

Cloudera recommends the following security protocols:

- Use certificates that are signed by a CA. Do not issue self-signed certificates.
- Generate a unique certificate per host.

## Configure TLS Encryption Manually for Schema Registry

If you do not want to enable Auto-TLS because for example, you need to use your own enterprise-generated certificates, you can manually enable TLS for Schema Registry.

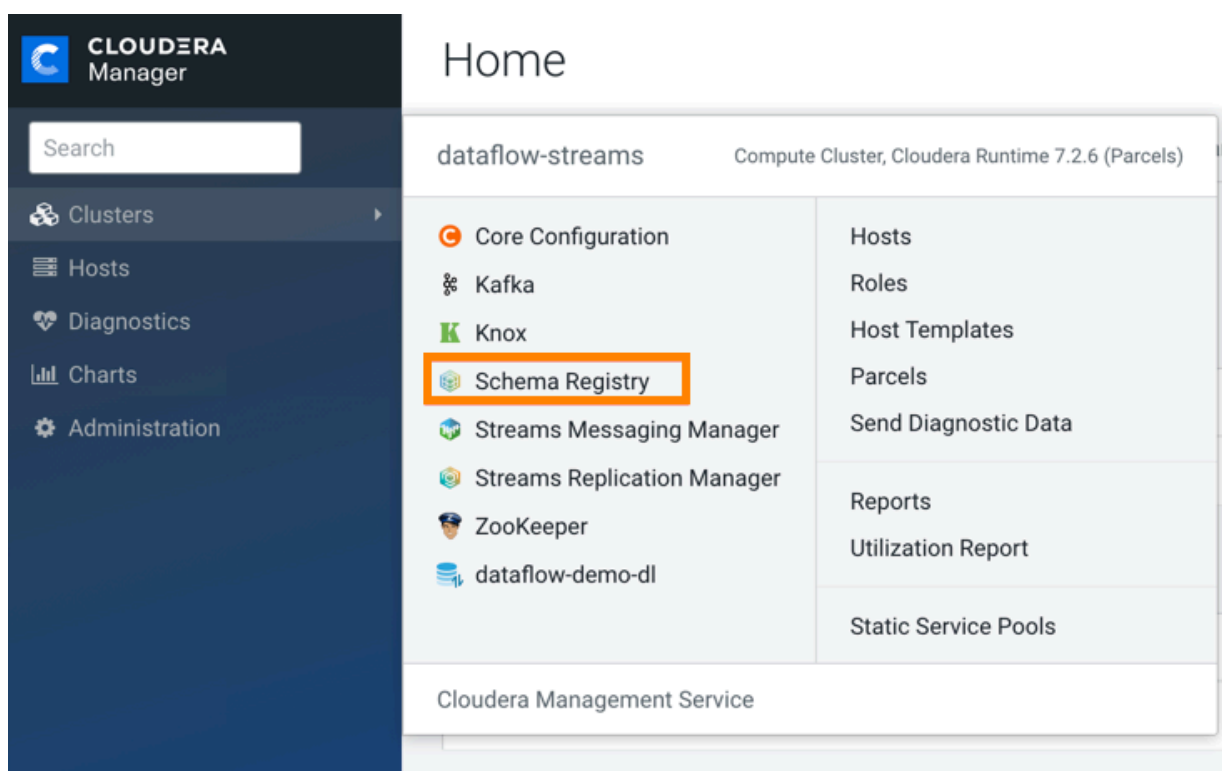
### Before you begin

Ensure you have set up TLS for Cloudera Manager:

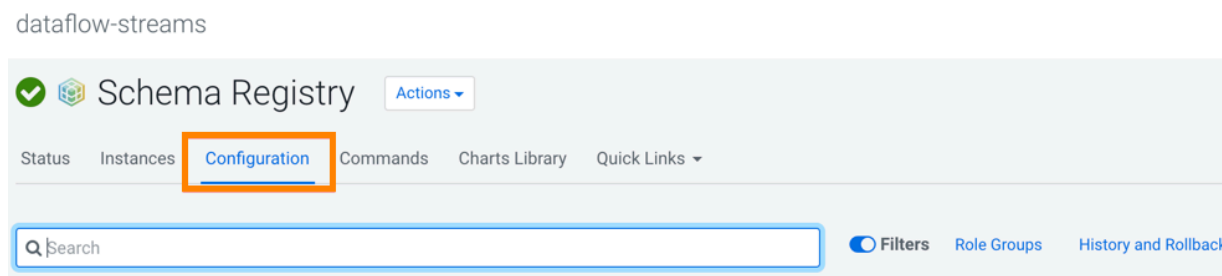
1. Review the requirements and recommendations for the certificates. See *TLS Certificate Requirements and Recommendations*.
2. Generate the TLS certificates and configure Cloudera Manager. See *Manually Configuring TLS Encryption for Cloudera Manager*.

### Procedure

1. From the Cloudera Manager UI, click Cluster Schema Registry .



2. Click the **Configuration** tab.



3. Enter `ssl` in the Search field.  
The Security properties for Schema Registry appear.
4. Edit the Security properties.

5. Click Save Changes.
6. Restart the Schema Registry service.

### Related Information

[TLS Certificate Requirements and Recommendations](#)

[Manually Configuring TLS Encryption for Cloudera Manager](#)

## Schema Registry TLS Properties

To enable and configure TLS manually for Schema Registry, edit the security properties according to the cluster configuration.

The following table lists the Security properties for Schema Registry:

Property	Description
Schema Registry Port (SSL) <code>schema.registry.ssl.port</code>	HTTPS port Schema Registry node runs on when SSL is enabled.
Schema Registry Admin Port (SSL) <code>schema.registry.ssl.adminPort</code>	HTTPS admin port Schema Registry node runs on when SSL is enabled.
SSL Keystore Type <code>schema.registry.ssl.keyStoreType</code>	The keystore type. It is blank by default but required if schema registry's ssl is enabled. e.g. PKCS12 or JKS. If it is left empty then this keystore type will come from CM settings.
SSL TrustStore Type <code>schema.registry.ssl.trustStoreType</code>	The truststore type. It is blank by default but required if schema registry's ssl is enabled. e.g. PKCS12 or JKS. If it is left empty then this keystore type will come from CM settings.
SSL ValidateCerts <code>schema.registry.ssl.validateCerts</code>	Whether or not to validate TLS certificates before starting. If enabled, it will refuse to start with expired or otherwise invalid certificates.
SSL ValidatePeers <code>schema.registry.ssl.validatePeers</code>	Whether or not to validate TLS peer certificates.
Version of oracle.net.ssl <code>schema.registry.oracle.net.ssl_version</code>	Oracle net ssl version.
Oracle TLS javax.net.ssl.keyStore <code>schema.registry.javax.net.ssl.keyStore</code>	Path to keystore file if enabling TLS using Oracle DB.
Oracle TLS javax.net.ssl.keyStoreType <code>schema.registry.javax.net.ssl.keyStoreType</code>	KeyStoreType type if enabling TLS using Oracle DB.

Property	Description
Oracle TLS javax.net.ssl.keyStorePassword <code>schema.registry.javax.net.ssl.keyStorePassword</code>	KeyStorePassword if enabling TLS using Oracle DB.
Oracle TLS javax.net.ssl.trustStore <code>schema.registry.javax.net.ssl.trustStore</code>	Required Path to truststore file if enabling TLS using Oracle DB.
Oracle TLS javax.net.ssl.trustStoreType <code>schema.registry.javax.net.ssl.trustStoreType</code>	Required Truststore type if enabling TLS using Oracle DB.
Oracle TLS javax.net.ssl.trustStorePassword <code>schema.registry.javax.net.ssl.trustStorePassword</code>	TrustStorePassword type if enabling TLS using Oracle DB.
Oracle TLS oracle.net.ssl_cipher_suites <code>schema.registry.oracle.net.ssl_cipher_suites</code>	Oracle net ssl cipher suites if enabling TLS using Oracle DB e.g. SSL_DH_DSS_WITH_DES_CBC_SHA.
Oracle TLS oracle.net.ssl_server_dn_match <code>schema.registry.oracle.net.ssl_server_dn_match</code>	Oracle ssl server domain name match if enabling TLS using Oracle DB.
Enable TLS/SSL for Schema Registry Server <code>ssl.enable</code>	Encrypt communication between clients and Schema Registry Server using Transport Layer Security (TLS) (formerly known as Secure Socket Layer (SSL)).
Schema Registry Server TLS/SSL Server JKS Keystore File Location <code>schema.registry.ssl.keyStorePath</code>	The path to the TLS/SSL keystore file containing the server certificate and private key used for TLS/SSL. Used when Schema Registry Server is acting as a TLS/SSL server.
Schema Registry Server TLS/SSL Server JKS Keystore File Password <code>schema.registry.ssl.keyStorePassword</code>	The password for the Schema Registry Server keystore file.
Schema Registry Server TLS/SSL Client Trust Store File <code>schema.registry.ssl.trustStorePath</code>	The location on disk of the trust store, in .jks format, used to confirm the authenticity of TLS/SSL servers that Schema Registry Server might connect to. This is used when Schema Registry Server is the client in a TLS/SSL connection. This trust store must contain the certificate(s) used to sign the service(s) connected to. If this parameter is not provided, the default list of well-known certificate authorities is used instead.
Schema Registry Server TLS/SSL Client Trust Store Password <code>schema.registry.ssl.trustStorePassword</code>	The password for the Schema Registry Server TLS/SSL Certificate Trust Store File. This password is not required to access the trust store; this field can be left blank. This password provides optional integrity checking of the file. The contents of trust stores are certificates, and certificates are public information.

# Schema Registry Authorization through Ranger Access Policies

User and group access to various Schema Registry functions is controlled through Apache Ranger.

Pre-defined access policies for Schema Registry allow the administrator to quickly add a user or user group to specify:

- Who can add/evolve schemas to a schema metadata.
- Who can view and edit schemas within a schema metadata.
- Who can upload the ser/des jar files.

If a higher level of granularity is necessary, the administrator can create an access policy and add the user or user-group to this custom policy.

## Related Information

[Pre-defined Access Policies for Schema Registry](#)

[Add the user or group to a pre-defined access policy](#)

[Create a Custom Access Policy](#)

## Pre-defined Access Policies for Schema Registry

Based on a user's responsibilities, you can add users or a user group to one or more of the following pre-defined access policies for Schema Registry and you can specify the type of permission such as Create, Read, Update, and Delete.

The following image shows the pre-defined access policies for Schema Registry:

Policy ID	Policy Name	Policy Labels	Status	Audit Logging	Roles	Groups	Users	Action
3	all - serde	--	Enabled	Enabled	--	--	streamsmgngnr kafka schemaregistry	View Edit Delete
5	all - schema-group, schema-metadata	--	Enabled	Enabled	--	--	streamsmgngnr kafka schemaregistry	View Edit Delete
6	all - schema-group, schema-metadata, schema-branch	--	Enabled	Enabled	--	--	streamsmgngnr kafka schemaregistry	View Edit Delete
7	all - registry-service	--	Enabled	Enabled	--	--	streamsmgngnr kafka schemaregistry	View Edit Delete
8	all - schema-group, schema-metadata, schema-branch, schema-version	--	Enabled	Enabled	--	--	streamsmgngnr kafka schemaregistry	View Edit Delete

The following table describes the pre-defined access policies for Schema Registry:

Access Policy	Description
all - serde	Allows users to store metadata for the format of how data should be read and how it should be written. Users can store JAR files for serializers and deserializers and then map the serdes to the schema.
all - schema-group, schema-metadata	Allows users to access the schema groups and schema metadata.
all - schema-group, schema-metadata, schema-branch	Allows users to access the schema groups, schema metadata, and schema branch.
all - registry-service	Allows users to access the schema registry service. If a user is added to this policy, the user can access all Schema Registry entities.
all - schema-group, schema-metadata, schema-branch, schema-version	Allows users to access the schema groups, schema metadata, schema branch, and schema version.

## Related Information

[Schema Registry Authorization through Ranger Access Policies](#)

[Add the user or group to a pre-defined access policy](#)

[Create a Custom Access Policy](#)

## Add the user or group to a pre-defined access policy

When an authenticated user attempts to view, create, edit, or delete a Schema Registry entity, the system checks whether the user has privileges to perform that action. These privileges are determined by the Ranger access policies that a user is associated with.

### Before you begin

For Ranger policies to work, you must have a user group named schemaregistry. If you use UNIX PAM, the schemaregistry user group must be on the node that hosts Schema Registry.

### About this task

Determine the permissions required by a user or user group and accordingly add the user or group to the appropriate pre-defined access policy.

Each pre-defined access policy controls access to one or more Schema Registry entities.

### Procedure

1. From the Cloudera Manager home page, click the Ranger link.  
The **Ranger** management page appears.
2. Click the Ranger Admin Web UI link.

The screenshot shows the Cloudera Manager interface for Cluster 1. The left sidebar contains navigation options: Clusters, Hosts, Diagnostics, Audits, Charts, Replication, Administration, and Private Cloud. The main content area displays the Ranger-1 management page. At the top, there is a 'RANGER-1' header with a status icon and an 'Actions' dropdown. Below this is a navigation bar with tabs: Status, Instances, Configuration, Commands, Charts Library, Audits, Ranger Admin Web UI (highlighted with an orange box), and Quick Links. The main content area is divided into sections: Health Tests (with a 'Show 3 Good' indicator and a 'Create Trigger' button), Status Summary (a table showing health status for Ranger Admin, Ranger Tagsync, Ranger Usersync, and Hosts), and Charts (an 'Informational Events' chart showing 0 events).

Service	Health Status
Ranger Admin	1 Good Health
Ranger Tagsync	1 Good Health
Ranger Usersync	1 Good Health
Hosts	1 Good Health

The **Ranger Log In** page appears.

3. Enter your user name and password to log in.  
The **Ranger Service Manager** page appears.

The page is organized by service. Each cluster is listed under its respective service. For example, the Schema Registry clusters in the environment are listed under Schema Registry.

- Select a cluster from the Schema Registry section.

The **List of Policies** page appears.

The screenshot shows the Ranger interface with the 'List of Policies' page for 'cm\_schema\_registry'. The page includes a search bar, an 'Add New Policy' button, and a table with the following data:

Policy ID	Policy Name	Policy Labels	Status	Audit Logging	Roles	Groups	Users	Action
3	all - serde	--	Enabled	Enabled	--	--	streamsmgmr kafka schemaregistry	👁️ ✎️ 🗑️
5	all - schema-group, schema-metadata	--	Enabled	Enabled	--	--	streamsmgmr kafka schemaregistry	👁️ ✎️ 🗑️
6	all - schema-group, schema-metadata, schema...	--	Enabled	Enabled	--	--	streamsmgmr kafka schemaregistry	👁️ ✎️ 🗑️
7	all - registry-service	--	Enabled	Enabled	--	--	streamsmgmr kafka schemaregistry	👁️ ✎️ 🗑️
8	all - schema-group, schema-metadata, schema...	--	Enabled	Enabled	--	--	streamsmgmr kafka schemaregistry	👁️ ✎️ 🗑️

- Click the ID for a policy.

The **Edit Policy** page appears.

- In the Allow Conditions section, add the user or group to the respective Select User or Select Group field.

The screenshot shows the 'Allow Conditions' section of the Ranger Edit Policy page. It includes a 'hide' link and a table with the following data:

Select Role	Select Group	Select User	Policy Conditions	Permissions	Delegate Admin
Select Roles	Select Groups	<input type="checkbox"/> streamsmgmr <input type="checkbox"/> kafka <input type="checkbox"/> schemaregistry	Add Conditions +	Create Read Update Delete	<input checked="" type="checkbox"/>

- From the Policy Conditions field, enter the appropriate IP address.

- From the Permissions field, select the appropriate permission.

- Click Save.

## Results

The user now has the rights according to the policy and the permission you assigned to the user. These rights apply to all objects in the entities unless you specified otherwise in the Policy Conditions field.

## Related Information

[Schema Registry Authorization through Ranger Access Policies](#)

[Pre-defined Access Policies for Schema Registry](#)

[Create a Custom Access Policy](#)

## Create a Custom Access Policy

You can create a custom access policy for a specific Schema Registry entity, specify an access type, and add a user or user-group to the policy.

### Before you begin

Determine the following information:

- The schema registry entity that the user needs access to.
- Whether the user requires all objects in the entity or specific objects.
- Whether the user needs read, view, edit, or delete permissions to the entity.
- If there are any IP addresses to include or exclude from the user's access.

### About this task

With a custom policy you can specify the Schema Registry entity and the type of access the user requires.

## Procedure

1. Go to the **Ranger List of Policies** page.
2. Click Add New Policy.

The screenshot shows the Ranger web interface. At the top, there is a navigation bar with 'Ranger', 'Access Manager', 'Audit', 'Security Zone', and 'Settings'. The user is logged in as 'admin'. Below the navigation bar, there is a breadcrumb trail: 'Service Manager > cm\_schema\_registry Policies'. The main heading is 'List of Policies : cm\_schema\_registry'. There 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 highlighted with a red box. Below the search bar is a table with the following columns: Policy ID, Policy Name, Policy Labels, Status, Audit Logging, Roles, Groups, Users, and Action. The table contains five rows of policy information.

Policy ID	Policy Name	Policy Labels	Status	Audit Logging	Roles	Groups	Users	Action
3	all - serde	--	Enabled	Enabled	--	--	streamsmgmr kafka schemaregistry	View Edit Delete
5	all - schema-group, schema-metadata	--	Enabled	Enabled	--	--	streamsmgmr kafka schemaregistry	View Edit Delete
6	all - schema-group, schema-metadata,...	--	Enabled	Enabled	--	--	streamsmgmr kafka schemaregistry	View Edit Delete
7	all - registry-service	--	Enabled	Enabled	--	--	streamsmgmr kafka schemaregistry	View Edit Delete
8	all - schema-group, schema-metadata,...	--	Enabled	Enabled	--	--	streamsmgmr kafka schemaregistry	View Edit Delete

The **Create Policy** page appears.

3. Enter a unique name for the policy.
4. Optionally, enter a keyword in the Policy Label field to aid in searching for a policy.
5. Select a Schema Registry entity. You can choose the Schema Registry service, schema group, or serde. Then, do one of the following tasks:
  - If you want the user to access all the objects in the entity, enter \*.
  - If you want to specify the objects in the entity that a user can access, enter the name of the object in the text field.
6. Optionally, enter a description.
7. In the Allow Conditions section, add the user or group to the respective Select User or Select Group field.

The screenshot shows the 'Allow Conditions' section of the Ranger web interface. It includes a 'hide' link on the right. Below the heading are several fields: 'Select Role' (with a 'Select Roles' dropdown), 'Select Group' (with a 'Select Groups' dropdown), 'Select User' (with a list of selected users: 'streamsmgmr', 'kafka', and 'schemaregistry'), 'Policy Conditions' (with an 'Add Conditions' button and a plus sign), 'Permissions' (with buttons for 'Create', 'Read', 'Update', 'Delete', and 'Delete'), and 'Delegate Admin' (with a checked checkbox and a red 'X' button).

8. Optionally, from the Policy Conditions field, enter the appropriate IP address.
9. From the Permissions field, select the appropriate permission.
10. Click Save.

## Results

The user now has the rights according to the policy and the permission you assigned to the user.

## Related Information

[Schema Registry Authorization through Ranger Access Policies](#)

[Pre-defined Access Policies for Schema Registry](#)

[Add the user or group to a pre-defined access policy](#)