Cloudera Runtime 7.3.3

Using Cloudera Iceberg REST Catalog for Data Sharing

Date published: 2020-07-28
Date modified: 2026-12-13



https://docs.cloudera.com/

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

How data sharing with Cloudera Iceberg REST Catalog works.....4

Setting up Cloudera Iceberg REST Catalog for data sharing.....4

Configuring Hive Metastore as a REST Catalog for Ozone storage.....7
Installing the Knox IDBroker.....8
Configuring Ozone STS for on premises Data Sharing.....10
Creating the Ranger role for Ozone STS.....12
Configuring the Knox IDBroker for Ozone S3 environments.....16
Declaring Knox topologies for on premises Data Sharing.....19
Trusting the Ozone STS certificate in the Knox IDBroker truststore.....22
Creating a Data Share.....22
Registering external clients for Cloudera on premises.....23
Managing Ranger policies for Data Shares.....25

Supported REST Catalog APIs for accessing the data.....26

How data sharing with Cloudera Iceberg REST Catalog works

Cloudera Iceberg REST Catalog enables your clients to run their workloads from data platforms, such as Databricks or Snowflake to fetch data from Cloudera environments for analytical purposes.

Data sharing involves the creation of a Data Share with the necessary authorization and authentication mechanisms. A Data Share is an organizational unit of data, a collection of data assets. You can then share this data with your clients so that they can access the Iceberg table data created within the Cloudera environment.

The following sections describe the high-level workflow of the processes:

Data Share creation

The following tasks are part of the Data Share creation process:

- As a resource owner, generate a token by calling the client-ID service exposed by the cdp-share-management Knox topology. This unique Token ID is referred to as the CLIENT_ID and the generated passcode is the CLIE NT_SECRET.
- You then create a Ranger group named after the CLIENT_ID and a Ranger role for the client, and add the group to the role. This role represents the client with whom the data is shared.
- Create and maintain policies for the set of databases and tables to be shared for the Ranger role and group and thereby create a Data Share.
- Maintain the SELECT permission for the databases or tables to allow READ-only access.
- You can then share the CLIENT_ID and the CLIENT_SECRET with your clients so that they can access the shared data in the Cloudera environment.

Data Share access

After you have created a Data Share, which includes creating tokens, authoring a read-only policy within Ranger, and shared the CLIENT_ID and CLIENT_SECRET with your client, the client makes use of these credentials in their workloads to establish a handshake with Cloudera and to access the shared data. In exchange for the CLIENT_ID and the CLIENT_SECRET, Knox can return an Access Token, that can be used in requests to Iceberg REST Catalog (Hive MetaStore).

When the shared data resides on Cloudera Object Store (powered by Apache Ozone) storage, Cloudera Iceberg REST Catalog also vends temporary Ozone S3 credentials to the client through the Knox IDBroker, which obtains them from the Ozone Security Token Service (STS).

Related Information

Enabling data sharing with Cloudera Iceberg REST Catalog

Creating a Data Share

Introduction to Ozone

Setting up Cloudera Iceberg REST Catalog for data sharing

Learn how to perform the preparatory configurations in Cloudera to enable data sharing. These configurations are required for the creation of a data share in Cloudera and allow your clients to access data in Cloudera environments.

Installation and Upgrade Scenarios

Note:

- On premises, Data Sharing supports only Iceberg tables stored on Cloudera Object Store (powered by Apache Ozone). Iceberg tables stored on HDFS are not supported.
- In the following processes, run all commands within the network of your Cloudera Runtime or through a VPN.

Different versions of Cloudera Runtime and Data Lakes require different steps to enable data sharing.

- Fresh 7.3.2.20000 or Higher Cloudera Base on premises Installation on page 5
- Upgrade from a lower version of Cloudera Base on premises to a 7.3.2.20000 or Higher version of Cloudera Base on premises on page 6

Fresh 7.3.2.20000 or Higher Cloudera Base on premises Installation

Complete the following steps after a fresh installation ofa 7.3.2.20000 or higher version of Cloudera Base on premises to enable data sharing.

Note: For more information on installing Cloudera Base on premises, see:

- Version and Download Information
- Cloudera Base on premises Installation Guide
- Production Installation

1. Installing the Metering V2 Service

   Install and configure the Metering V2 service. This is a prerequisite for Cloudera Data Sharing on Ozone storage. If Metering V2 is TLS-enabled, also import the Metering V2 certificate authority into the Hive Metastore truststore.
2. HMS REST Catalog configuration for Ozone storage

   Enable the Iceberg REST Catalog in the Hive Metastore and point it at your Ozone Object Manager and S3 Gateway so Cloudera Data Sharing can vend temporary S3 credentials through the gateway. Only Iceberg tables whose storage locations use s3a:// paths are shareable; tables registered with ofs:// locations only cannot be shared this way.
3. Installing the Knox IDBroker

   Add the IDBroker role to the Knox service. On premises deployments do not provision the IDBroker automatically, so you must install it manually before you can configure it for Ozone S3 credential vending.
4. Configuring Ozone STS for on premises Data Sharing

   Enable the Ozone Security Token Service (STS) and the Ranger action-matcher condition, and then complete the Ranger role, Knox IDBroker, certificate, and topology configuration required so that REST Catalog can vend temporary Ozone S3 credentials.
5. Creating the Ranger role for Ozone STS

   Create the sts-assume-role Ranger role and grant the hive user **Assume Role** access so that Ozone STS can generate session credentials for the REST Catalog.
6. Configuring the Knox IDBroker for Ozone S3 environments

   Configure the Knox IDBroker STS endpoint override, session policy template, and AWS user mapping so that IDBroker can vend temporary Ozone S3 credentials to external clients.
7. Trusting the Ozone STS certificate in the Knox IDBroker truststore

   Import the cluster's SCM Local CA into the Knox IDBroker truststore so that IDBroker can call the Ozone STS and S3 Gateway endpoint over HTTPS.
8. Declaring Knox topologies for on premises Data Sharing

Declare the cdp-datashare-access and cdp-share-management Knox topologies so external clients can reach REST Catalog through Knox and administrators can generate the CLIENT_ID and CLIENT_SECRET used by external clients.

9. Registering external clients for on premises deployments

   Generate a CLIENT_ID and CLIENT_SECRET for the external client through the cdp-share-management Knox topology, create the Ranger group and role that represent the client, and then create the Data Share policy in Managing Ranger policies to grant that role SELECT access to the shared Iceberg tables.

Upgrade from a lower version of Cloudera Base on premises to a 7.3.2.20000 or Higher version of Cloudera Base on premises

Complete the following configurations to enable Cloudera Data Sharing when upgrading Cloudera Base on premises from a supported version to a 7.3.2.20000 or higher versionthe Data Lake from 7.2.18.x with no REST Catalog or Cloudera Data Sharing configured in that version.

Note: For more information on upgradingCloudera Base on premises, see the following:

- Cloudera Upgrade and Migrations Paths
- Supported in-place upgrade paths

1. Installing the Metering V2 Service

   Install and configure the Metering V2 service. This is a prerequisite for Cloudera Data Sharing on Ozone storage. If Metering V2 is TLS-enabled, also import the Metering V2 certificate authority into the Hive Metastore truststore.
2. HMS REST Catalog configuration for Ozone storage

   Enable the Iceberg REST Catalog in the Hive Metastore and point it at your Ozone Object Manager and S3 Gateway so Data Sharing can vend temporary S3 credentials through the gateway. Only Iceberg tables whose storage locations use s3a:// paths are shareable; tables registered with ofs:// locations only cannot be shared this way.
3. Installing the Knox IDBroker

   Add the IDBroker role to the Knox service. On premises deployments do not provision the IDBroker automatically, so you must install it manually before you can configure it for Ozone S3 credential vending.
4. Configuring Ozone STS for on premises Data Sharing

   Enable the Ozone Security Token Service (STS) and the Ranger action-matcher condition, and then complete the Ranger role, Knox IDBroker, certificate, and topology configuration required so that REST Catalog can vend temporary Ozone S3 credentials.
5. Creating the Ranger role for Ozone STS

   Create the sts-assume-role Ranger role and grant the hive user **Assume Role** access so that Ozone STS can generate session credentials for the REST Catalog.
6. Configuring the Knox IDBroker for Ozone S3 environments

   Configure the Knox IDBroker STS endpoint override, session policy template, and AWS user mapping so that IDBroker can vend temporary Ozone S3 credentials to external clients.
7. Trusting the Ozone STS certificate in the Knox IDBroker truststore

   Import the cluster's SCM Local CA into the Knox IDBroker truststore so that IDBroker can call the Ozone STS and S3 Gateway endpoint over HTTPS.
8. Declaring Knox topologies for on premises Data Sharing

   Declare the cdp-datashare-access and cdp-share-management Knox topologies so external clients can reach REST Catalog through Knox and administrators can generate the CLIENT_ID and CLIENT_SECRET used by external clients.
9. Registering external clients for on premises deployments

Generate a CLIENT_ID and CLIENT_SECRET for the external client through the cdp-share-management Knox topology, create the Ranger group and role that represent the client, and then create the Data Share policy in Managing Ranger policies to grant that role SELECT access to the shared Iceberg tables.

Configuring Hive Metastore as a REST Catalog for Ozone storage

To share Iceberg tables on Cloudera Object Store (powered by Apache Ozone) through the S3 Gateway and vended credentials, enable Cloudera Iceberg REST Catalog in Hive Metastore and point it at your Ozone Object Manager and S3 Gateway. Only tables whose storage locations use s3a:// paths are shareable; tables registered with ofs:// locations only cannot be shared this way.

Before you begin

This procedure applies to Cloudera on premises deployments where the Hive Metastore stores Iceberg metadata on Cloudera Object Store (powered by Apache Ozone) (ofs://) on a Kerberized cluster.

Ensure that your Cloudera Runtime is of 7.3.2.20000 version or later.

Procedure

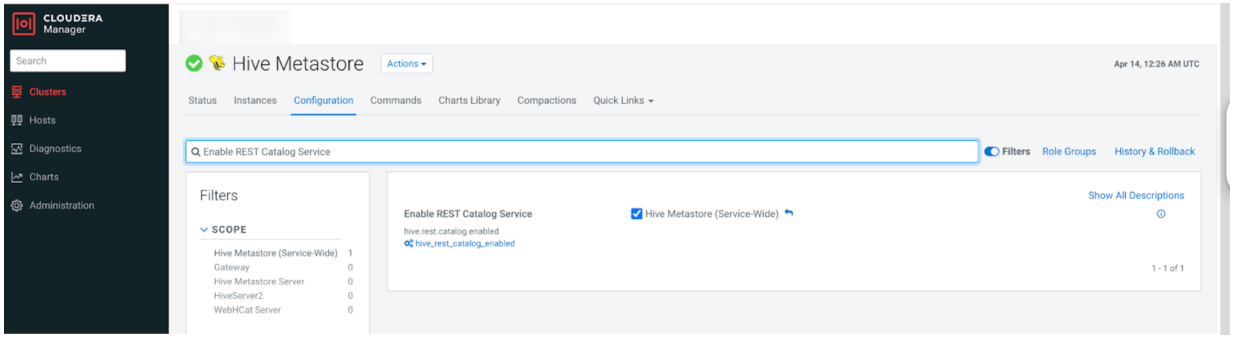
1. Log in to Cloudera Manager and go to Clusters HDFS-1 Configuration and search for HDFS Cluster-wide Advanced Configuration Snippet (Safety Valve) for core-site.xml.
2. Add the Ozone Object Manager (OM) service and file system properties so that HDFS clients can resolve ofs:// paths:

```
<property><name>ozone.om.service.ids</name><valu
e>[***OZONE_OM_SERVICE_ID***]</value></property>
<property><name>ozone.om.nodes.[***OZONE_OM_SERVICE_ID***]</name><value>om
1,om2,om3</value></property>
<property><name>ozone.om.address.[***OZONE_OM_SERVICE_ID***].om1</name><
value>[***OM1_HOST***]:9862</value></property>
<property><name>ozone.om.address.[***OZONE_OM_SERVICE_ID***].om2</name><
value>[***OM2_HOST***]:9862</value></property>
<property><name>ozone.om.address.[***OZONE_OM_SERVICE_ID***].om3</name><
value>[***OM3_HOST***]:9862</value></property>
<property><name>fs.ofs.impl</name><value>org.apache.hadoop.fs.ozone.Roo
tedOzoneFileSystem</value></property>
<property><name>fs.AbstractFileSystem.ofs.impl</name><value>org.apache.had
oop.fs.ozone.RootedOzoneFs</value></property>
```



Note: You can find the value for ozone.om.service.ids in Clusters Ozone Instances Ozone Manager Processes , in conf/ozone-site.xml under the ozone.om.service.ids property. Use the same service ID to fill in the remaining node and address properties.

3. Click Save Changes, and then run Actions Deploy Client Configuration and restart the affected services.
4. Log in to Cloudera Manager and click Clusters Hive Metastore Configuration .
5. Search for **Enable REST Catalog Service** and enable it to set up the REST Catalog Service in Hive Metastore.



6. Go to Clusters HIVE-1 Configuration and search for Hive Service Advanced Configuration Snippet (Safety Valve) for hive-site.xml.
7. Add the REST Catalog, Ozone data sharing, and Knox IDBroker credential-vending properties:

```
<property><name>hive.rest.catalog.enabled</name><value>true</value></pro
perty>
<property><name>hive.metastore.catalog.ozone.enabled</name><value>true</v
alue></property>
<property><name>hive.metastore.catalog.ozone.om.address</name><valu
e>[***OZONE_OM_SERVICE_ID***]</value></property>
<property><name>hive.metastore.catalog.ozone.s3.gateway</name><value>htt
ps://[***OZONE_S3_GATEWAY_HOST***]:9879</value></property>
<property><name>rest.catalog.env.cloud.provider</name><value>ozone</val
ue></property>
<property><name>hive.metastore.catalog.idbroker.url</name><value>http
s://[***KNOX_IDBROKER_HOST***]:8444/gateway</value></property>
```

If clients read or write tables using s3a:// locations instead of ofs://, also add the following properties on both the **HIVE** and **HIVE_ON_TEZ** services:

```
<property><name>fs.s3a.endpoint</name><value>htt
ps://[***OZONE_S3_GATEWAY_HOST***]:9879</value></property>
<property><name>fs.s3a.bucket.probe</name><value>0</value></property>
<property><name>fs.s3a.path.style.access</name><value>true</value></prope
rty>
<property><name>fs.s3a.change.detection.mode</name><value>none</value></
property>
<property><name>fs.s3a.change.detection.version.required</name><value>fal
se</value></property>
```

8. Click Save Changes, and then run Actions Deploy Client Configuration and restart the Hive Metastore service.

What to do next

Continue with Installing the Knox IDBroker.

Installing the Knox IDBroker

Add the IDBroker role to the Knox service so that Cloudera Iceberg REST Catalog can vend temporary Ozone S3 credentials to external clients.

About this task

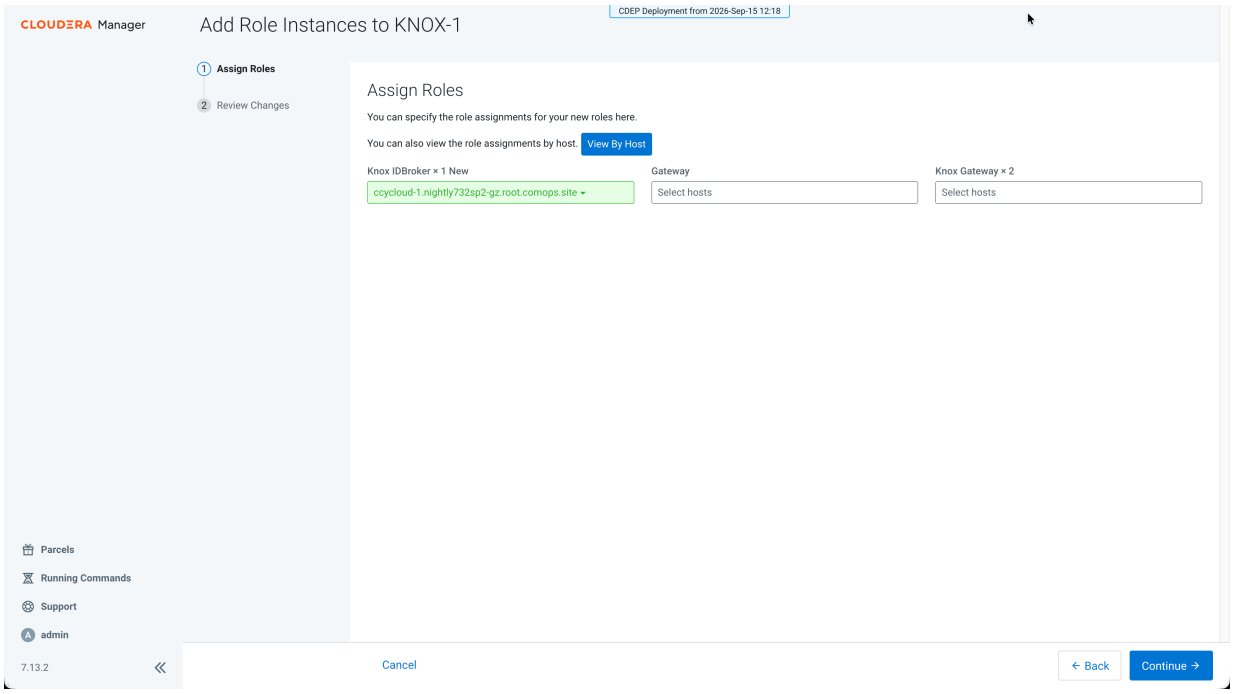
In Cloudera on cloud environments, the Knox IDBroker is provisioned automatically. In on premises environments, you must add the IDBroker role to the Knox service manually before you can configure it for Ozone S3 credential vending.

Procedure

1. Login to Cloudera Manager as Administrator.
2. Go Clusters KNOX-1 , then select the **Instances** tab.
3. Add the IDBroker role to the Knox service.

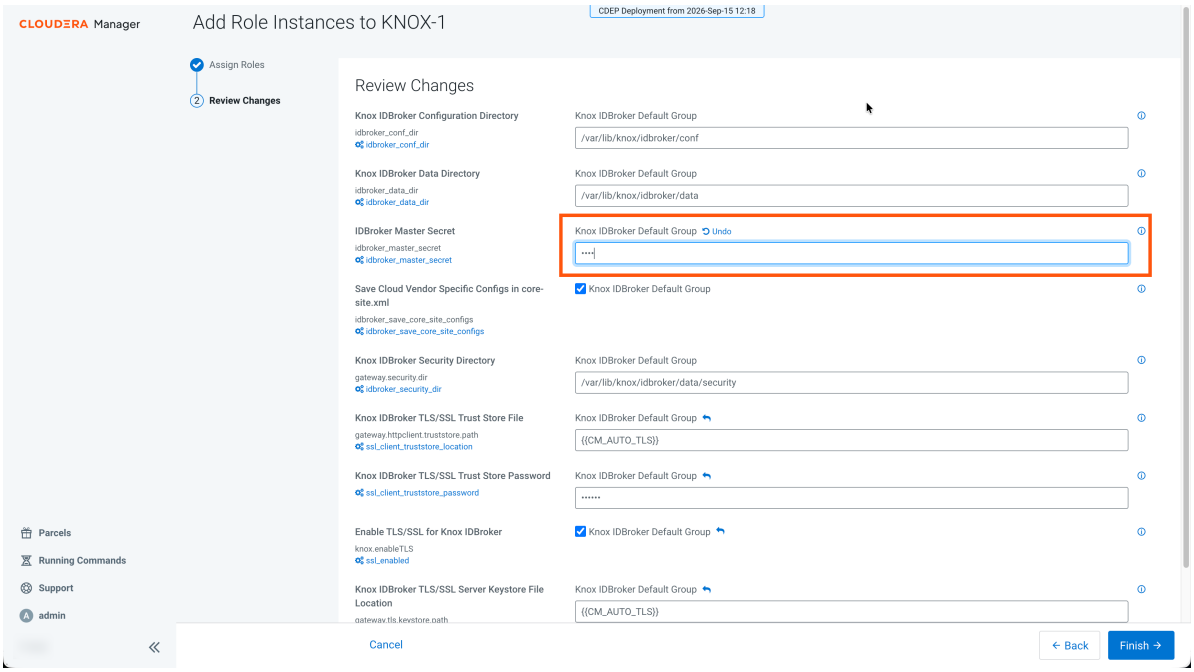
   Click Add Role Instances, and assign the IDBroker role to the host running Knox. Continue through the wizard.

   Figure 1: Assigning the IDBroker role to the Knox host



4. On the **Review Changes** page, set the IDBroker master secret.
   a) Provide a value for **IDBroker Master Secret** (idbroker_master_secret)

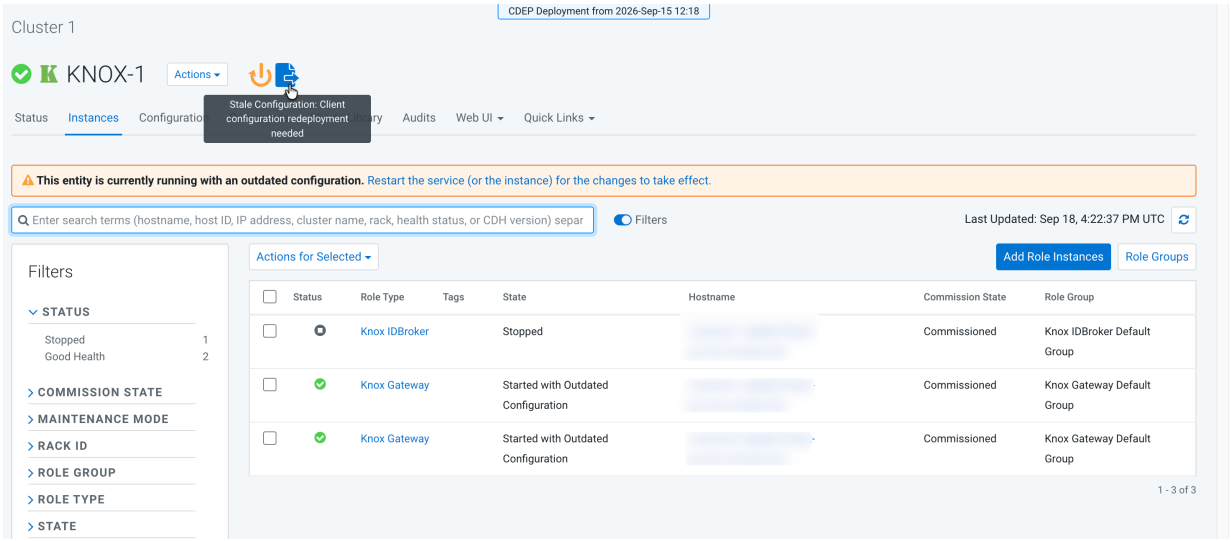
   Figure 2: IDBroker master secret on the Review Changes page



b) Click Finish.

5. Restart all affected services from Cloudera Manager.

Figure 3: Restarting affected services after adding IDBroker



What to do next

Continue with Configuring Ozone STS for on premises Data Sharing.

Configuring Ozone STS for on premises Data Sharing

Enable the Ozone Security Token Service (STS) and Ranger action-matcher support so that REST Catalog can vend temporary Ozone S3 credentials for on premises Data Sharing.

About this task

Data sharing on Cloudera Object Store (powered by Apache Ozone) storage requires Ozone to issue AWS STS-compatible, short-lived credentials (AssumeRole) that the Knox IDBroker exchanges for external clients. This configuration applies only when the data being shared is stored on Ozone S3 in a Cloudera on premises deployment.

Before you begin

- Complete Configuring Hive Metastore as a REST Catalog for Ozone storage, including the Ozone safety-valve properties.
- Complete Installing the Metering V2 Service.
  - If Metering V2 is TLS-enabled, import the Metering V2 certificate authority (metering-scm-ca) into the Hive Metastore truststore.
- Ranger and Ozone must support the ASSUME_ROLE access type and inline-policy grants. This is available in Cloudera Runtime 7.3.2.20000 and later service packs.
- Ozone STS and the Ranger action-matcher condition for Ozone service definitions must be available in your build.

Procedure

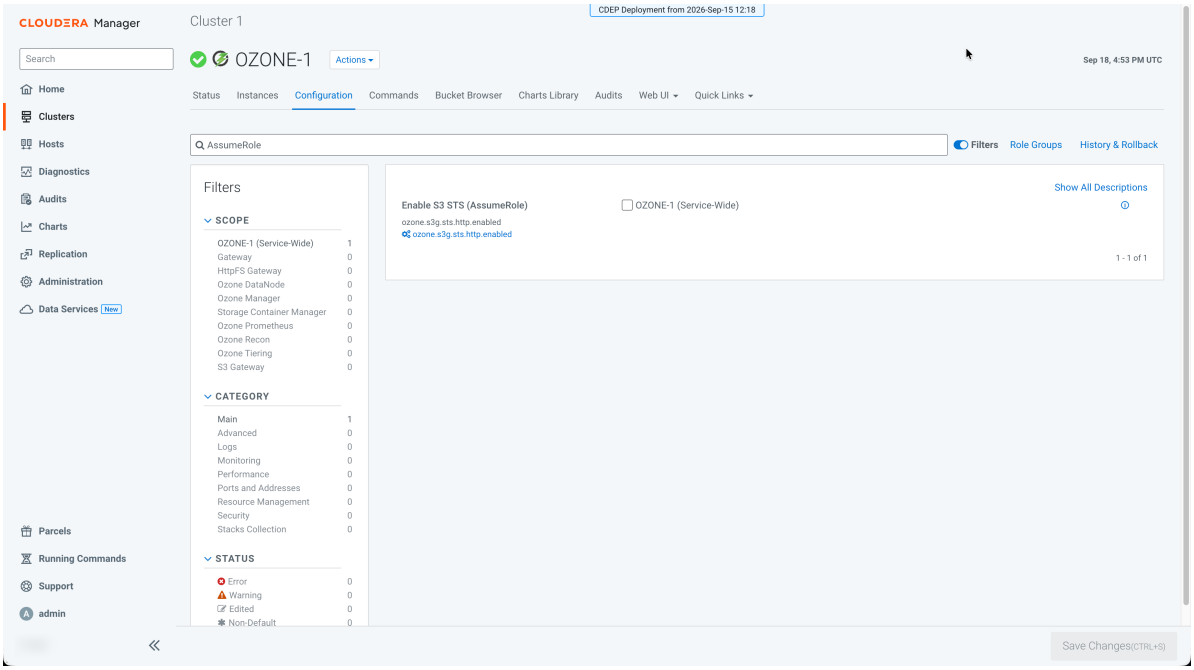
1. Enable the S3 STS (AssumeRole) feature.

   A single Cloudera Manager setting turns on both the Ozone STS feature flag (ozone.s3g.sts.http.enabled) and the Ranger action-matcher condition for Ozone service definitions (ranger.servicedef.ozone.enableActionMatcherI

nPoliciesCondition) at the same time. The action-matcher condition lets Ranger evaluate the ASSUME_ROLE access type when the Ozone S3 Gateway requests a session policy during STS token generation.

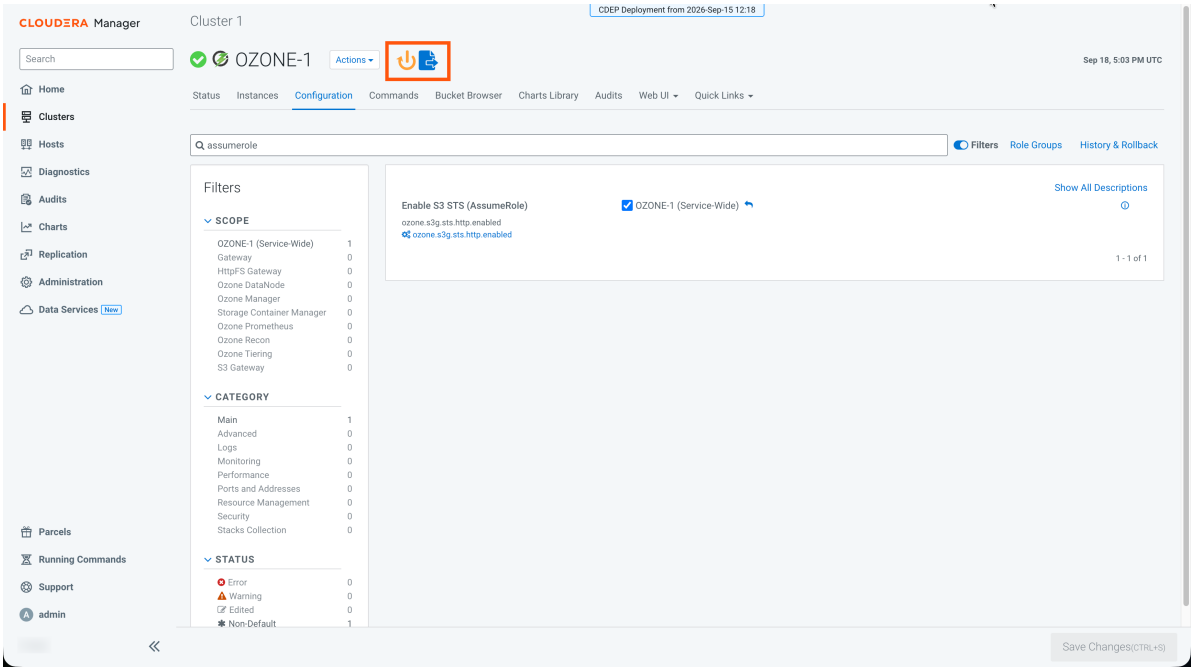
a) In Cloudera Manager, go to Clusters OZONE-1 Configuration and search for the word assumeRole.
b) Select Enable S3 STS (AssumeRole), then click Save Changes.

Figure 4: Enable S3 STS (AssumeRole) in Ozone configuration



c) Wait about 30 seconds until the stale configuration indicator appears next to the Actions dropdown, and then click the Stale Configuration: Restart needed icon.

Figure 5: Stale configuration restart needed after enabling AssumeRole



d) Click Restart Stale Services, ensure Regular Restart is selected, and then click Continue.
e) When the restart completes, click Finish.

2. Create the Ranger role that Ozone STS uses to generate session credentials for the REST Catalog, as described in Creating the Ranger role for Ozone STS.
3. Configure the Knox IDBroker for the Ozone S3 Gateway, as described in Configuring the Knox IDBroker for Ozone S3 environments.
4. Declare the cdp-datashare-access and cdp-share-management Knox topologies for Ozone data sharing, as described in Declaring Knox topologies for on premises Data Sharing.
5. Trust the Ozone STS and S3 Gateway certificate in the IDBroker truststore, as described in Trusting the Ozone STS certificate in the Knox IDBroker truststore.

Results

REST Catalog can now vend temporary Ozone S3 credentials to external clients. For a table stored on Ozone, the /credentials response returns an s3.endpoint for the Ozone S3 Gateway and temporary s3.access-key-id, s3.secret-access-key, and s3.session-token values. Clients use those credentials against the gateway; the vended storage path prefix is s3a:// and matches the translated Ozone link bucket name, not the original ofs:// table location. For a working example, see Supported REST Catalog APIs for accessing the data.

What to do next

Continue with Creating the Ranger role for Ozone STS.

Creating the Ranger role for Ozone STS

Create the Ranger role that Ozone STS uses to generate session credentials (access key and secret) for Cloudera Iceberg REST Catalog.

About this task

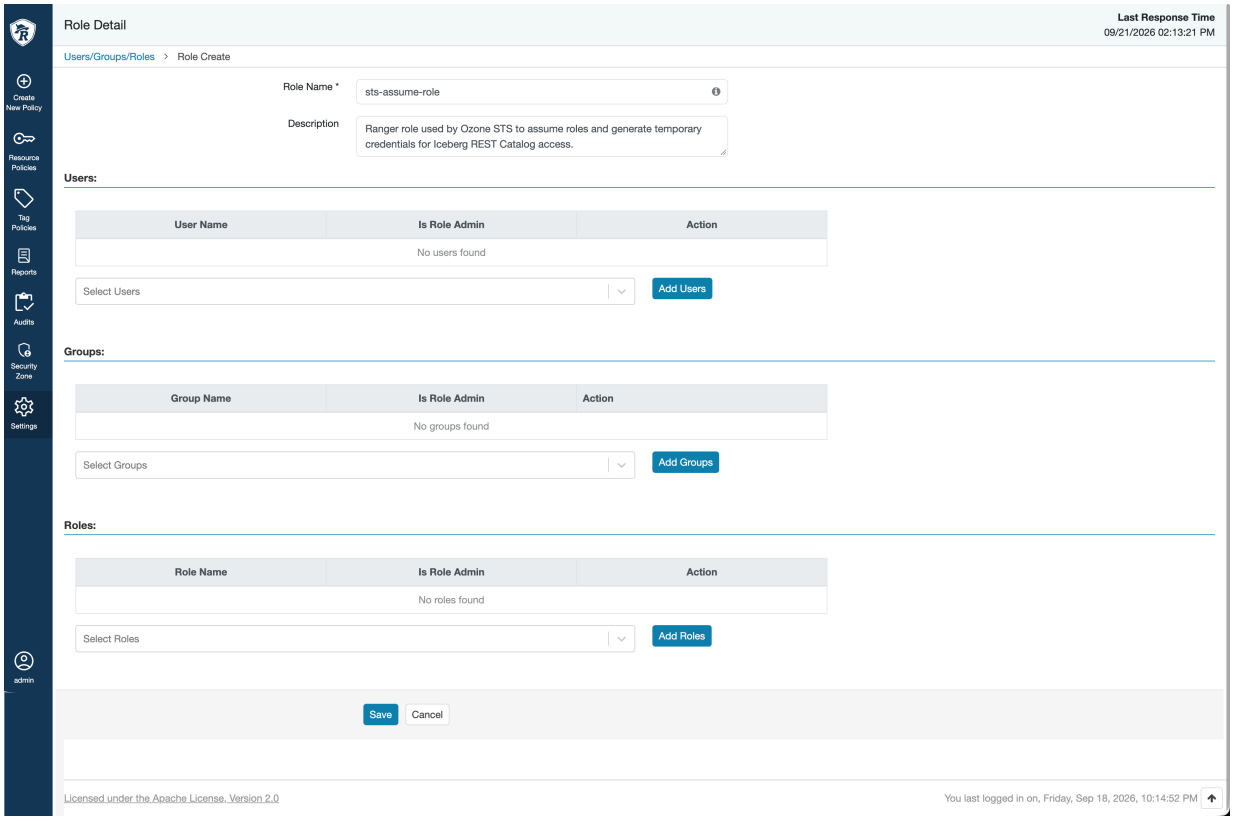
hive is the identity that Cloudera Iceberg REST Catalog uses to obtain short-lived credentials from IDBroker on behalf of external clients, so the Ranger role that Ozone STS uses to generate session credentials must grant that identity **Assume Role** access.

Procedure

1. In Ranger, go to Settings Roles , then click Add New Role to create the sts-assume-role role.

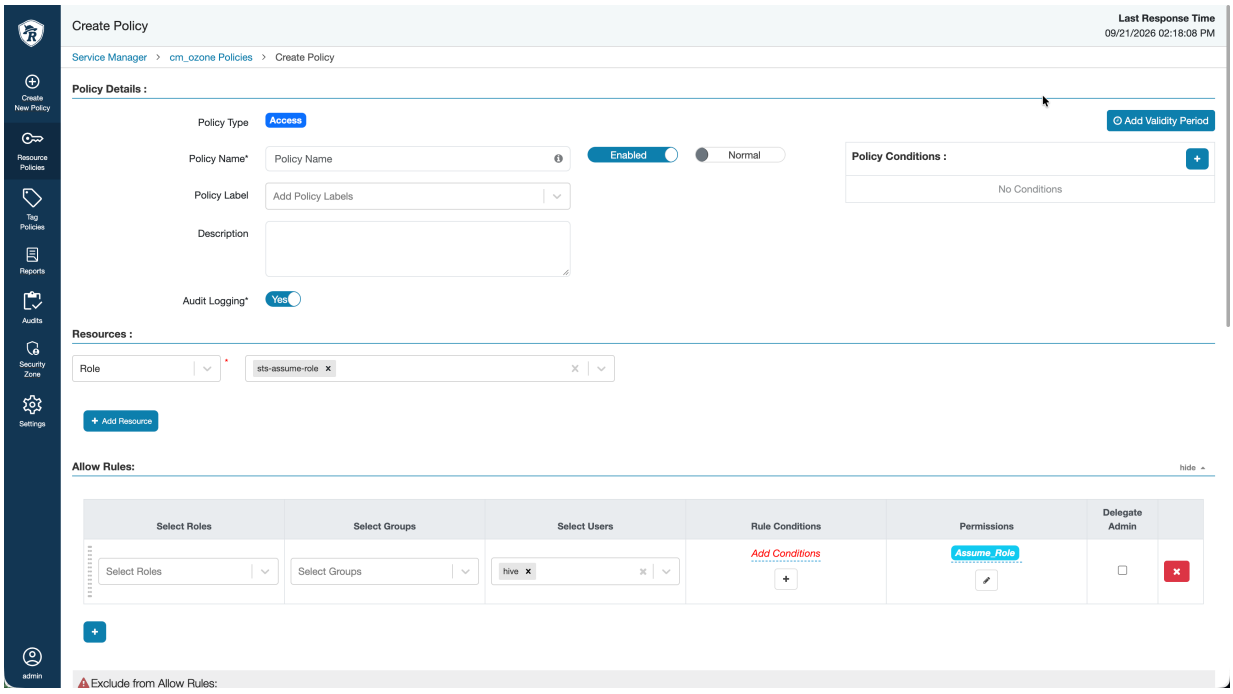
   Select the hive user when creating the role.

   Figure 6: Creating the sts-assume-role Ranger role



2. Go to Resource Policies cm_ozone service and create the Assume Role Policy.
   a) Add the sts-assume-role as role under **Resources**.
3. Grant the hive user **Assume Role** access on the sts-assume-role role.

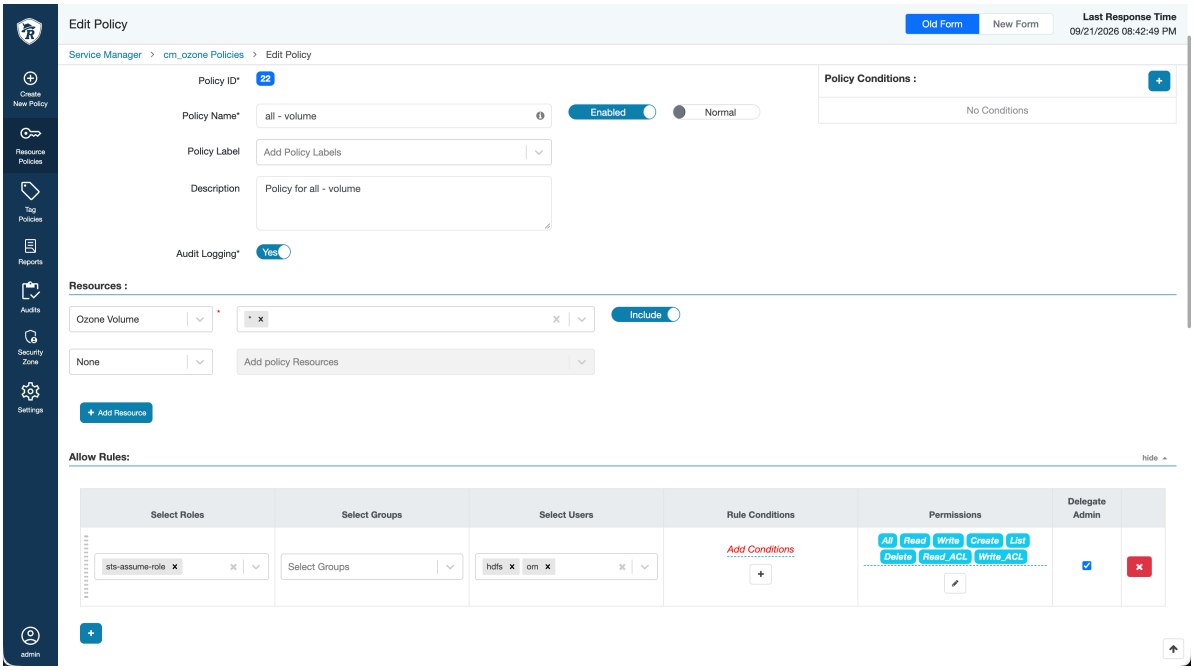
Figure 7: Creating the Assume Role policy



4. Grant the sts-assume-role role read access on the Cloudera Object Store (powered by Apache Ozone) volume, bucket, and key resources that back the shared tables.
   - This grant requires the Ranger action-matcher condition for the Ozone service definition to be enabled.
   - When you author resource paths for a policy that covers a table stored on Ozone, use the translated s3a:// link bucket path (for example, s3a://*[***VOLUME***]-[***BUCKET***]*/...), not the original ofs:// location. Cloudera Iceberg REST Catalog derives IDBroker policy paths and inline loadTable credential prefixes from this translated link bucket, so a policy written against the raw ofs:// path does not match.

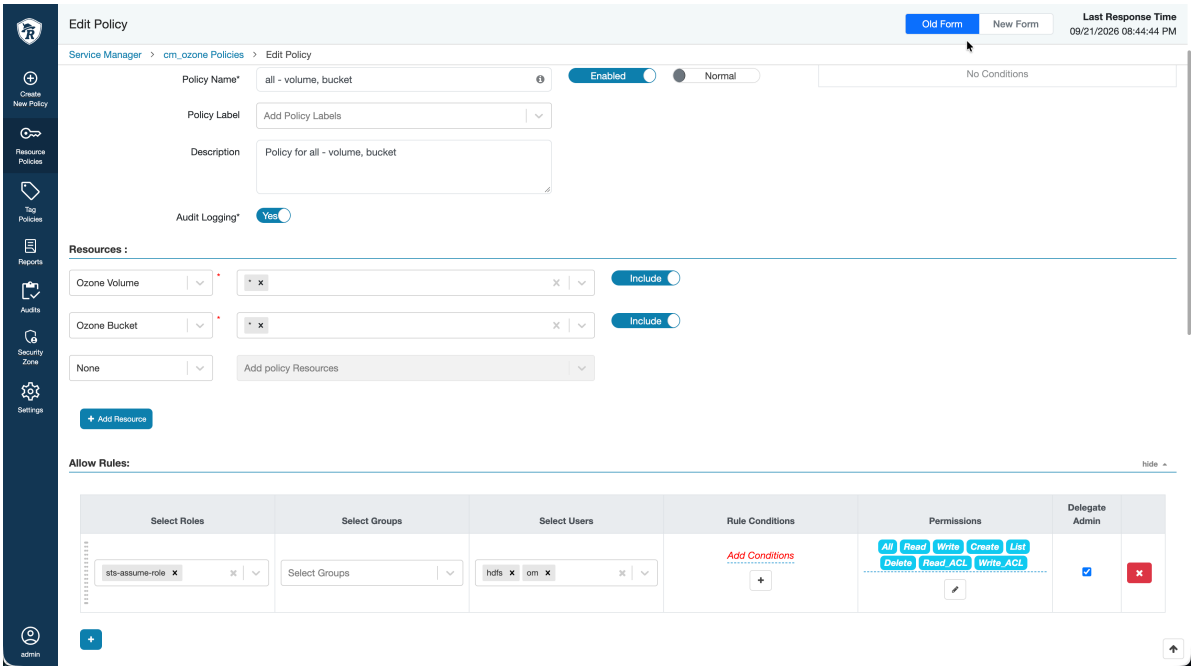
   a) Edit your Ozone volume policy (for example, the default all - volume policy) and add the sts-assume-role role to the **Allow Rules** with read permissions.

   Figure 8: Ozone volume policy



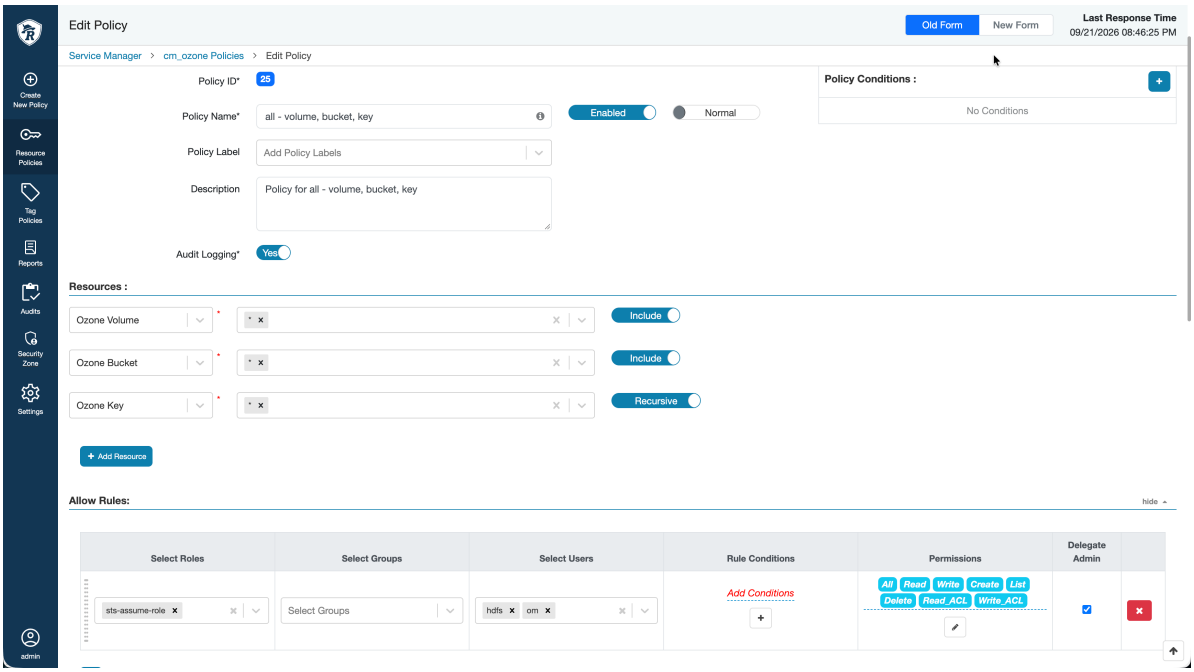
   b) Edit your Ozone bucket policy (for example, the default all - volume, bucket policy) and add the sts-assume-role role to the **Allow Rules** with read permissions.

   Figure 9: Ozone bucket policy



c) Edit your Ozone key policy (for example, the default all - volume, bucket, key policy) and add the sts-assume-role role to the **Allow Rules** with read permissions.

Figure 10: Ozone key policy



5. Generate the Ozone S3 access key and secret for the hive user.

   These credentials belong to the identity that was granted **Assume Role** access. Save them into the Knox IDBroker aliases in Configuring the Knox IDBroker for Ozone S3 environments.

   Authenticate as the hive Kerberos principal by using the Hive Metastore keytab present on the host you are configuring, and then retrieve the secret:

```
kinit -kt [***HIVE_KEYTAB_PATH***] hive/[***HMS_HOST***]@[***REALM***]
```

```
ozone s3 getsecret
```

Note: Cloudera Manager writes the keytab under a process directory whose numeric prefix (for example, .../process/142-hive-HIVEMETASTORE/hive.keytab) changes on each restart. Use the path currently present on the host.

The ozone s3 getsecret command returns the Ozone S3 access key (the Kerberos principal) and secret:

```
awsAccessKey=hive/[***HMS_HOST***]@[***REALM***]
awsSecret=[***OZONE_S3_SECRET_KEY***]
```

What to do next

Continue with Configuring the Knox IDBroker for Ozone S3 environments.

Configuring the Knox IDBroker for Ozone S3 environments

Configure the Knox IDBroker STS endpoint, session policy, and AWS user mapping to vend temporary Ozone S3 credentials to external clients.

About this task

For Cloudera Object Store (powered by Apache Ozone) environments, you must manually configure the IDBroker STS endpoint, session policy, and AWS user mapping. Complete this procedure after Installing the Knox IDBroker.

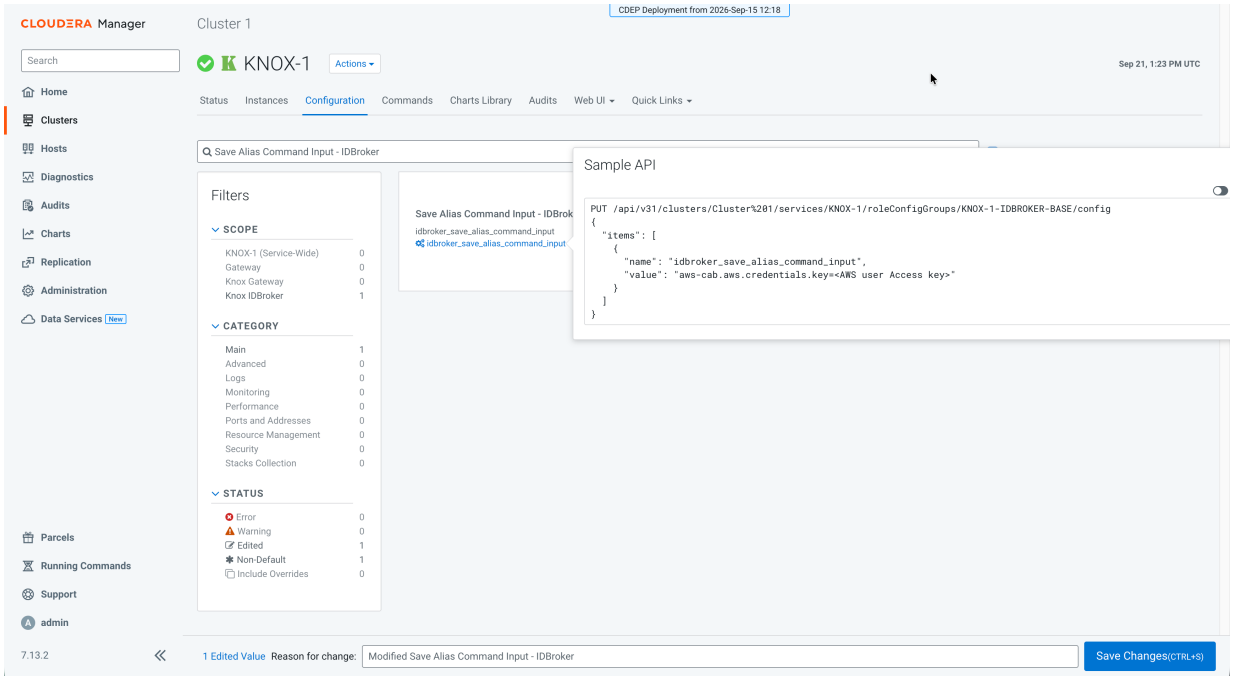
Before you begin

The sts-assume-role role must already exist in Ranger and be granted read access to the shared Ozone volume, bucket, and key resources, as described in Creating the Ranger role for Ozone STS.

Procedure

1. Go to Cloudera Manager Knox Configuration and search for Save Alias Command Input - IDBroker.
2. Enter the S3-compatible storage credential access key alias, click Save Changes, and then run Actions Save Alias - IDBroker .

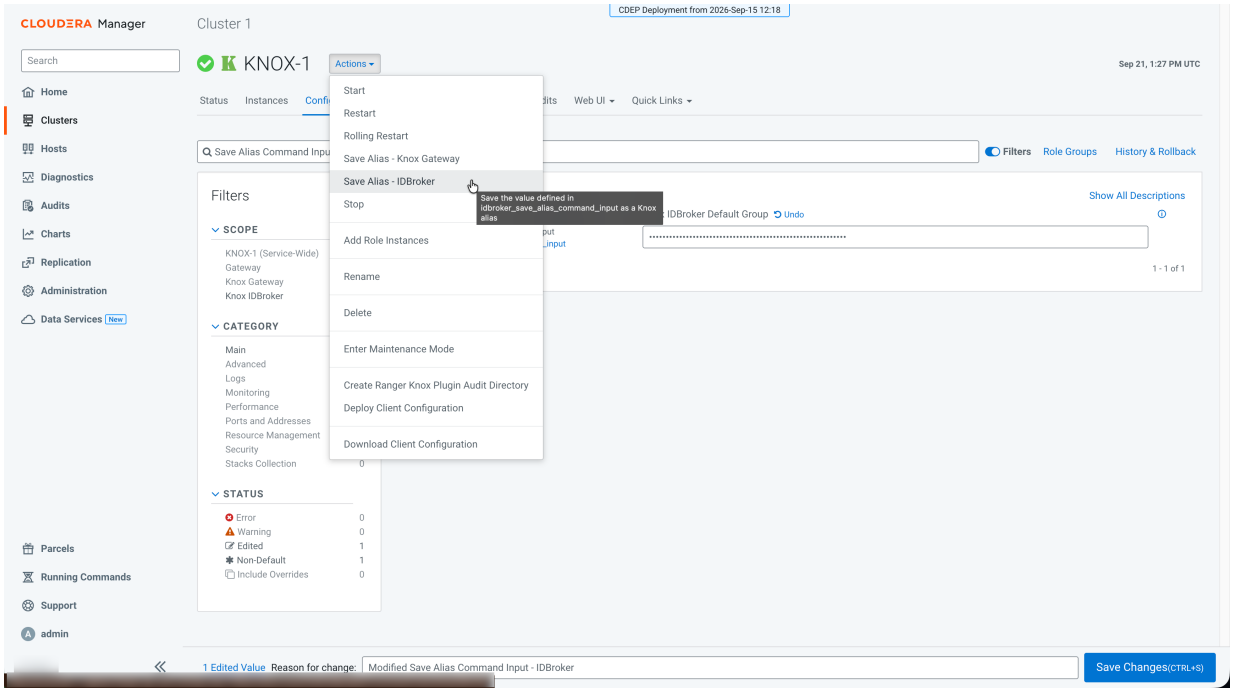
Figure 11: Save Alias Command Input - IDBroker property




```
aws-cab.aws.credentials.key=[***OZONE_S3_ACCESS_KEY***]
```

3. Search for Save Alias Command Input - IDBroker again, enter the S3-compatible storage credential secret access key alias, click Save Changes, and then run Actions Save Alias - IDBroker again.

Figure 12: Save Alias - IDBroker action




```
aws-cab.aws.credentials.secret=[***OZONE_S3_SECRET_KEY***]
```

4. Set the Save Alias Command Input - IDBroker property back to empty, and click Save Changes.

Note: Clearing the property prevents the plaintext secret from remaining in the Cloudera Manager configuration.

5. Go to Cloudera Manager Knox Configuration and search for Knox IDBroker Advanced Configuration Snippet (Safety Valve) for conf/cdp-resources.xml.
6. Add the following entry to override the IDBroker STS endpoint to point at the Ozone S3 Gateway:
   - Name:

     ```
     aws-cab
     ```

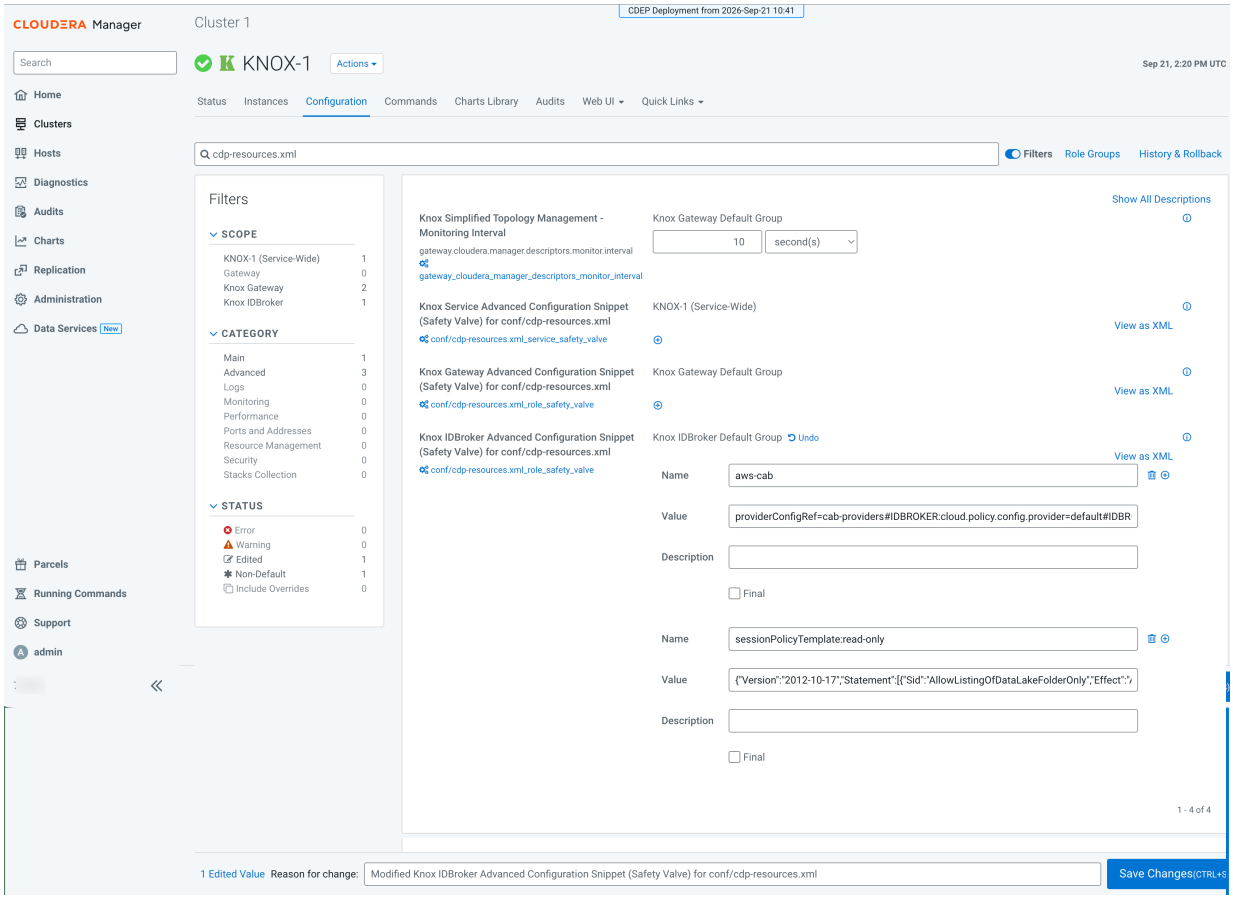
   - Value:

     ```
     providerConfigRef=cab-providers#IDBROKER:cloud.policy.config.provider=de
     fault#IDBROKER:cloud.client.provider=AWS#IDBROKER:aws.region.name=us-eas
     t-1#IDBROKER:org.apache.knox.idbroker.endpoint.override=
     https://[***OZONE_S3_GATEWAY_HOST***]:[***SECURE OZONE S3 GATEWAY API
      PORT (TLS/SSL)***]
     ```

   Name:

   Note: The Secure Ozone S3 Gateway API Port (TLS/SSL) and the Secure Ozone S3 Gateway STS HTTPS Port can be found under Cloudera Manager Clusters OZONE-1 Configuration
7. Add the read-only session IDBroker policy template to the Knox IDBroker Advanced Configuration Snippet (Safety Valve) for conf/cdp-resources.xml:

Figure 13: Read-only session policy template



- Name:

```
sessionPolicyTemplate:read-only
```

- Value:

```
{"Version":"2012-10-17","Statement":[{"Sid":"AllowListingOfDataLakeFolderOnly","Effect":"Allow","Action":["s3:List*"],"Resource":"arn:aws:s3:::${bucket}","Condition":{"StringEquals":{"s3:prefix":["${prefix}","${prefix}/"]}}},{"Sid":"AllowAccessToDataLakeFolder","Effect":"Allow","Action":["s3:Get"],"Resource":"arn:aws:s3:::${bucket}/${prefix}/*"}]}
```

8. In Cloudera Manager Knox Configuration , search for Knox IDBroker AWS User Mapping, set the mapping for the hive user to the Ranger role created for Ozone STS (for example, sts-assume-role), and click Save Changes.

   Map the hive Kerberos user to the STS role ARN. Use the Ranger role name from Creating the Ranger role for Ozone STS; change only the account ID and role name in the ARN to match your environment.

   ```
   hive=arn:aws:iam::[***ACCOUNT_ID***]:role/[***RANGER_ROLE_NAME***]
   ```

9. Click Save Changes, and then restart the Knox service.

   Note: Without these cdp-resources.xml entries, the Cloudera Iceberg REST Catalog credentials API cannot obtain Ozone S3 STS credentials for external clients. This restart is separate from the Save Alias - IDBroker command used earlier for the credential aliases, which does not persist these safety valve entries.

What to do next

Continue with Declaring Knox topologies for on premises Data Sharing.

Related Information

Configuring cloud credentials using Knox Alias

Declaring Knox topologies for on premises Data Sharing

Declare the cdp-datashare-access and cdp-share-management Knox topologies so external clients can reach Cloudera Iceberg REST Catalog and administrators can generate client IDs and secrets.

Before you begin

- Identify the share-administrative users who generate CLIENT_ID and CLIENT_SECRET values and who must be Knox proxy users and Ranger administrators.
- Identify the Hive Metastore hosts where Cloudera Iceberg REST Catalog is enabled and the Ranger Admin hosts that cdp-share-management proxies to.

About this task

The cdp-datashare-access topology proxies external REST Catalog and OAuth token requests to the Hive Metastore Iceberg REST servlet and issues bearer tokens for the IDBroker audience. The cdp-share-management topology exposes the Knox token service and Ranger Admin APIs that you use to generate CLIENT_ID and CLIENT_SECRET pairs and register external clients. Declaring the topologies does not create a client; it provides the endpoints you call later.

Add all four properties in this procedure to the same Knox Gateway Advanced Configuration Snippet (Safety Valve) for conf/cdp-resources.xml.

Procedure

1. In Cloudera Manager, go to Clusters Knox Configuration and search for Knox Gateway Advanced Configuration Snippet (Safety Valve) for conf/cdp-resources.xml.
2. Add the cdp-datashare-access provider configuration. Enter the following values:
   - **Name**:

     ```
     providerConfigs:cdp-datashare-access-provider
     ```
   - **Value**:

     ```
     role=federation#federation.name=JWTProvider#federation.enabled=true#fede
     ration.param.knox.token.exp.server-managed=true#role=identity-assertion#
     identity-assertion.name=Default#identity-assertion.enabled=true#identity
     -assertion.param.group.mapping.$PRIMARY_GROUP=(not (member username))#ro
     le=ha#ha.name=HaProvider#ha.param.ICEBERG-REST=enabled=true;maxFailoverA
     ttempts=3;failoverSleep=1000
     ```
3. Add the cdp-datashare-access topology descriptor. Set the following:
   - **Name**:

     ```
     cdp-datashare-access
     ```
   - **Value**:

     ```
     providerConfigRef=cdp-datashare-access-provider#KNOXTOKEN:knox.token.ttl
     =36000000#KNOXTOKEN:knox.token.exp.server-managed=true#KNOXTOKEN:knox.to
     ken.audiences=idbroker#KNOXTOKEN:gateway.knox.token.limit.per.user=-1#IC
     EBERG-REST:url=http://[***HMS_HOST***]:[***REST_CATALOG_PORT***]/icecli#
     ICEBERG-REST:url=http://[***HMS_HOST_2***]:[***REST_CATALOG_PORT***]/ice
     cli#
     ```

     Note: The default Iceberg REST Catalog servlet port is 18090 and the servlet path is icecli. If you run Hive Metastore on a single host, set *[***HMS_HOST_2***]* to the same value as *[***HMS_HOST***]* so the HA provider has two backend entries. For multiple Hive Metastore hosts, add one ICEBERG-REST :url=http://*[***HMS_HOST***]*:*[***REST_CATALOG_PORT***]*/icecli# fragment per host.

     Note: The value 36000000 for knox.token.ttl sets a 10-hour token lifetime. Adjust this value on both topologies if you need a different duration.
4. Add the cdp-share-management provider configuration. Enter the following values:
   - **Name**:

     ```
     providerConfigs:cdp-share-management-providers
     ```
   - **Value**:

     ```
     role=authentication#authentication.name=ShiroProvider#authentication.par
     am.main.invalidRequest=org.apache.shiro.web.filter.InvalidRequestFilter#
     authentication.param.main.invalidRequest.blockBackslash=false#authentica
     tion.param.main.invalidRequest.blockNonAscii=false#authentication.param.
     main.invalidRequest.blockSemicolon=false#authentication.param.main.pamRe
     alm=org.apache.knox.gateway.shirorealm.KnoxPamRealm#authentication.param
     .main.knoxAnonFilter=org.apache.knox.gateway.filter.AnonymousAuthFilter#
     authentication.param.urls./knoxtoken/api/v1/jwks.json=knoxAnonFilter#aut
     hentication.param.main.pamRealm.service=login#authentication.param.sessi
     onTimeout=30#authentication.param.urls./**=authcBasic#role=identity-asse
     rtion#identity-assertion.name=HadoopGroupProvider#identity-assertion.par
     am.hadoop.proxyuser.impersonation.enabled=true#identity-assertion.param.
     hadoop.proxyuser.[***KNOX-ADMIN-USER***].users=*#identity-assertion.para
     ```

```
m.hadoop.proxyuser.[***KNOX-ADMIN-USER***].groups=*#identity-assertion.p
aram.hadoop.proxyuser.[***KNOX-ADMIN-USER***].hosts=*#identity-assertion
.param.CENTRAL_GROUP_CONFIG_PREFIX=gateway.group.config.#role=authorizat
ion#authorization.name=XASecurePDPKnox#authorization.enabled=false#role=
ha#ha.name=HaProvider#ha.enabled=true#ha.param.RANGER=enableStickySessio
n=false;noFallback=false;enableLoadBalancing=true
```

Note: Replace *[***KNOX-ADMIN-USER***]* with a share-administrative user that generates external client IDs and secrets. This user must be configured as a proxy user (impersonation) and as a Ranger administrator.

5. If additional share-administrative users need access, duplicate the three identity-assertion.param.hadoop.proxyuser.[***KNOX-ADMIN-USER***].* fragments in the providerConfigs:cdp-share-management-providers value for each user.
6. Add the cdp-share-management topology descriptor. Set the following:
   - **Name**:

     ```
     cdp-share-management
     ```

   - **Value**:

     ```
     providerConfigRef=cdp-share-management-providers#RANGER:url=http
     s://[***RANGER-HOST***]:[***RANGER-PORT***]#KNOXTOKEN:knox.token.ttl=36
     000000#KNOXTOKEN:knox.token.type=JWT#KNOXTOKEN:knox.token.target.url=cdp
     -proxy-token#KNOXTOKEN:knox.token.audiences=cdp-proxy-token#KNOXTOKEN:kn
     ox.token.client.data=homepage_url=homepage/home?profile=token&topologies
     =cdp-proxy-token#KNOXTOKEN:knox.token.exp.tokengen.allowed.tss.backends=
     JDBCTokenStateService,AliasBasedTokenStateService#KNOXTOKEN:knox.token.l
     ifespan.input.enabled=true#KNOXTOKEN:knox.token.user.limit.exceeded.acti
     on=RETURN_ERROR#KNOXTOKEN:knox.token.exp.server-managed=true#KNOXTOKEN:g
     ateway.knox.token.limit.per.user=-1#KNOXTOKEN:knox.token.renewer.whiteli
     st=[***SHARE_ADMIN_USERS***]
     ```

   Note: Replace *[***RANGER-HOST***]* and *[***RANGER-PORT***]* with the hostname and HTTPS port of each Ranger Admin host (repeat RANGER:url=https://*[***RANGER-HOST***]*:*[***RANGER-PORT***]*# for high availability). By default, the Ranger HTTPS port is 6182.

   Note: Set *[***SHARE_ADMIN_USERS***]* to a comma-separated list of share-administrative users (for example, knoxui,hrt_qa,om,systest). The list must match the proxy users configured in providerConfigs: cdp-share-management-providers.
7. In the same Knox service, search for Knox Gateway Advanced Configuration Snippet (Safety Valve) for conf/gateway-site.xml and add the following properties:

   ```
   <property><name>gateway.knox.admin.users</name><valu
   e>[***SHARE_ADMIN_USERS***]</value></property>
   <property><name>gateway.knox.token.limit.per.user</name><value>-1</value
   ></property>
   ```

   Use the same comma-separated share-administrative user list as in knox.token.renewer.whitelist.
8. Click Save Changes, deploy client configuration if your environment requires it, and restart the Knox service.

What to do next

Continue with Trusting the Ozone STS certificate in the Knox IDBroker truststore.

Trusting the Ozone STS certificate in the Knox IDBroker truststore

Import the cluster's SCM Local CA into the Knox IDBroker JVM truststore so that the IDBroker can call the Ozone STS and S3 Gateway endpoint over HTTPS when vending temporary credentials to external clients.

About this task

The Knox IDBroker calls the Cloudera Object Store (powered by Apache Ozone) STS endpoint over HTTPS. If the certificate authority that signs the Ozone S3 Gateway and STS certificate is not trusted by the IDBroker JVM, the call fails with a PKIX path building failed: unable to find valid certification path to requested target error.

To prevent this error, import the cluster's SCM Local CA (which signs the Ozone S3 Gateway and STS certificate) into the cacerts truststore that the IDBroker AWS SDK uses. Complete this procedure after Configuring the Knox IDBroker for Ozone S3 environments.

Procedure

Retrieve the SCM Local CA certificate, import it under the ozone-scm-ca alias, and verify the import.

Replace *[***OZONE_S3_GATEWAY_HOST***]* with the host running the Ozone S3 Gateway, and adjust the JDK path if your cluster uses a different JVM. The awk c==2 selector keeps the second certificate in the chain, which is the SCM Local CA that signs the S3 Gateway and STS certificate.

```
export S3G=[***OZONE_S3_GATEWAY_HOST***]
export JHOME=/usr/lib/jvm/jdk1.17.0.11.0-openjdk-cloudera

# Grab the full chain and keep the CA (second cert in the chain)
echo | openssl s_client -connect "${S3G}:9881" -showcerts 2>/dev/null \
  | awk '/BEGIN CERT/{c++} c==2,/END CERT/' > /tmp/sts-ca.pem

# Verify that the extracted certificate is the SCM Local CA
openssl x509 -in /tmp/sts-ca.pem -noout -subject -issuer

sudo "$JHOME/bin/keytool" -importcert -noprompt \
  -alias ozone-scm-ca \
  -file /tmp/sts-ca.pem \
  -keystore "$JHOME/lib/security/cacerts" \
  -storepass changeit
"$JHOME/bin/keytool" -list \
  -alias ozone-scm-ca \
  -keystore "$JHOME/lib/security/cacerts" \
  -storepass changeit
```

Note: Run these steps on all hosts in the cluster, and then restart the cluster.

What to do next

Continue with Creating a Data Share.

Creating a Data Share

Learn how resource owners or Data Share administrators can share Iceberg tables in the Cloudera platform by registering external clients and creating Data Shares using Knox and Ranger API commands.

Resource owners or Data Share administrators who want to share their Iceberg tables in Cloudera with external clients must first register the external client (Data Consumer) in the Cloudera on cloud environment. This provisions a CLIE NT_ID and CLIENT_SECRET for the external user.

After registering the external user, the resource owner creates a Data Share. A Data Share packages specified data assets (Iceberg tables) into a shareable unit and optionally grants access to registered external users at creation time.

Registering external users (Data Consumers)

For more information, see Registering external clients for on premises deployments.

Creating a Data Share

For more information, see Managing Ranger policies.

Note:

The CDP CLI also provides commands to manage the full Data Share lifecycle, including listing, updating, activating, deactivating, and deleting shares, as well as managing asset membership and external user access.

Registering external clients for Cloudera on premises

For Cloudera on premises deployments, generate a CLIENT_ID and CLIENT_SECRET for an external client through the cdp-share-management Knox topology, and then create the Ranger group and role that represent the external client.

About this task

In Cloudera on premises deployments, external clients are registered directly against the cdp-share-management Knox topology gateway rather than a Cloudera on cloud Data Lake. The registration provisions a CLIENT_ID and CLIE NT_SECRET, which the external client later uses in an OAuth client credentials flow to authenticate to Cloudera Iceberg REST Catalog. You then create a Ranger group named after the CLIENT_ID and add it to a Ranger role. To grant that role SELECT access to the Iceberg tables you want to share, create a Data Share policy as described in Managing Ranger policies for Data Shares.

Before you begin

- Declare the cdp-datashare-access and cdp-share-management Knox topologies, as described in Declaring Knox topologies for on premises Data Sharing.
- The user that runs the following commands:
  - Must exist in the cluster
  - Must be configured as a Knox proxy (impersonation) user in the cdp-share-management topology
  - Must be granted the Ranger administrator role
- Run all commands within the network of your Cloudera Runtime deployment or through a VPN.

Procedure

1. Generate the CLIENT_ID and CLIENT_SECRET by calling the cdp-share-management Knox token service on the Knox Gateway:

   This CLIENT_ID acts as the unique principal identity for the external client (such as Snowflake or EMR) when it requests an OAuth token. The CLIENT_SECRET is the password that the external client uses to authenticate.

   ```
   curl -k -u [***RANGER-ADMIN-USER***]:[***PASSWORD***] "https://[***KNOX-
   GATEWAY-HOST***]:8443/gateway/cdp-share-management/knoxtoken/api/
   v1/token?doAs=external.user&comment=[***ADDITIONAL-INFO***]&md_cont
   act=[***EMAIL_ID***]&md_role=[***ROLE_OF_THE_CLIENT***]&md_type=CLIENT_
   ID" | jq -r '"CLIENT_ID: \(.token_id) SECRET: \(.passcode)"'
   ```

   - doAs=external.user - Sets the value of the external.user.
   - comment - Additional comments on the CLIENT.
   - md_contact - Client contact metadata, for example, an email address.
   - md_role - *[***ROLE FOR THE CLIENT_ID***]*.

- md_type - Token metadata type; set to CLIENT_ID for external client credentials.

Note: Replace *[***KNOX-GATEWAY-HOST***]* with the hostname of the Knox Gateway node.

```
curl -k -u systest:[***PASSWORD***] "https://my-knox-gateway-host.root.
comops.site:8443/gateway/cdp-share-management/knoxtoken/api/v1/token?doA
s=external.user&comment=carriers&md_contact=client_name@company.com&md_r
ole=UnitedAirlinesRole&md_type=CLIENT_ID" | jq -r '"CLIENT_ID: \(.token_
id) SECRET: \(.passcode)"'
```

2. Create a Ranger group whose name is the CLIENT_ID generated in the previous step:

   When the external client authenticates via the REST Catalog, Ranger evaluates its permissions based on its CLIE NT_ID. By creating a Ranger group that exactly matches the UUID string of the CLIENT_ID, you can manage its access in the Ranger ecosystem.

```
curl -k -u [***RANGER-ADMIN-USER***]:[***PASSWORD***] -H "Accept: applica
tion/json" -H "Content-Type: application/json" -X POST "https://[***KNOX-
GATEWAY-HOST***]:8443/gateway/cdp-share-management/ranger/service/xusers
/groups/" -d '{"name": "[***CLIENT_ID***]", "description": "group represe
nting a share for a CLIENT_ID"}'
```

   Note: Apache Ranger provides the following public API to create the group. For more information, see the Apache Ranger API documentation (`POST /xusers/groups`).

```
curl -k -u systest:[***PASSWORD***] -H "Accept: application/json" -H "C
ontent-Type: application/json" -X POST "https://my-knox-gateway-host.roo
t.comops.site:8443/gateway/cdp-share-management/ranger/service/xusers/gr
oups/" -d '{"name": "9a18e31e-c70d-4f84-8a5a-22d7fbce6124", "description":
 "group representing a share for a CLIENT_ID"}'
```

3. Create a new Ranger role for the external client, and add the group (the CLIENT_ID) to it.

   Instead of assigning table permissions directly to the unreadable CLIENT_ID group, best practice is to assign permissions to a human-readable role (such as SalesDeptRole). This step creates that role and links the external client to it so that it inherits the role's access.

```
curl -k -u [***RANGER-ADMIN-USER***]:[***PASSWORD***] -H "Accept:
 application/json" -H "Content-Type: application/json" -X POST "http
s://[***KNOX-GATEWAY-HOST***]:8443/gateway/cdp-share-management/range
r/service/public/v2/api/roles/" -d '{"name": "[***CLIENT_ROLE***]", "
description": "[***CLIENT_ROLE_DESCRIPTION***]", "groups": [ { "name":
 "[***CLIENT_ID***]", "isAdmin": false } ] }'
```

   Note: Apache Ranger provides the following public API to create a role. For more information, see the Apache Ranger API documentation (`POST /roles/roles`).

```
curl -k -u systest:[***PASSWORD***] -H "Accept: application/json" -H "C
ontent-Type: application/json" -X POST "https://my-knox-gateway-host.roo
t.comops.site:8443/gateway/cdp-share-management/ranger/service/public/v2
/api/roles/" -d '{"name": "SalesDeptRole", "description": "SalesRole des
cription", "groups": [ { "name": "9a18e31e-c70d-4f84-8a5a-22d7fbce6124",
 "isAdmin": false } ] }'
```

4. Alternatively, if you already have an existing Ranger role that manages access for this Data Share, add the group to that existing role instead.

   Use this alternative if you do not want to create a new role.

- Get the RoleId for the role name:

```
curl -k -u [***RANGER-ADMIN-USER***]:[***PASSWORD***] -H "Accept:
 application/json" -H "Content-Type: application/json" -X GET "https
://[***KNOX-GATEWAY-HOST***]:8443/gateway/cdp-share-management/ranger/se
rvice/public/v2/api/roles/name/[***CLIENT_ROLE***]" | jq -r '"RoleId:
\(.id)"'
```

- Add the group to the role by using the RoleId:

```
curl -k -u [***RANGER-ADMIN-USER***]:[***PASSWORD***] -H
"Accept: application/json" -H "Content-Type: application/
json" -X PUT "https://[***KNOX-GATEWAY-HOST***]:8443/gatewa
y/cdp-share-management/ranger/service/public/v2/api/role
s/[***ROLE_ID***]" -d '{"name": "[***CLIENT_ROLE***]", "descript
ion": "[***CLIENT_ROLE_DESCRIPTION***]", "groups": [ { "name":
 "[***CLIENT_ID***]", "isAdmin": false } ] }'
```

Results

The external client is registered with a CLIENT_ID and CLIENT_SECRET, and a Ranger role that contains the CLIE NT_ID group represents the Data Share. The client uses the CLIENT_ID and CLIENT_SECRET in an OAuth client credentials flow to authenticate to Cloudera Iceberg REST Catalog.

What to do next

Create the Data Share policy that grants the Ranger role SELECT access to the shared Iceberg tables, as described in Managing Ranger policies.

Managing Ranger policies for Data Shares

Learn how to manage your Ranger policies to authenticate your external users.

About this task

The Ranger Administrator must maintain policies for the set of databases and tables for the Ranger role and group to enable read access for these assets.

Before you begin



Note: Run all commands within the network of your Cloudera Runtime or through a VPN.

Procedure

Create a Data Share policy in Ranger that grants the Ranger role SELECT access to the databases and tables you want to share.

Create the policy for the role created in Registering external clients and creating a Data Share for on premises deployments:

```
curl -k -u [***RANGER-ADMIN-USER***]:[***PASSWORD***] -H "Accept: applica
tion/json" -H "Content-Type: application/json" -X POST "https://[***KNOX-
GATEWAY-HOST***]:8443/gateway/cdp-share-management/ranger/service/pu
blic/v2/api/policy/" -d '{"service":"cm_hive", "policyType": 0, "name":
 "[***DATA_SHARE_NAME***]", "description": "Policy for SELECT access
to the shared Iceberg tables", "isEnabled": true, "resources": { "da
tabase": { "values": ["[***DATABASE_NAME***]"] }, "table": { "values":
 ["[***TABLE_NAME***]"] }, "column": { "values": ["*"] } }, "policyItems":
```

```
 [ { "accesses": [ { "type": "select" } ], "users": [], "groups": [], "role
s": ["[***CLIENT_ROLE***]"], "conditions": [] } ] }'
```

Instead of managing access separately for every shared table, you define a central Ranger policy that groups the relevant tables together into a unified Data Share. This policy grants the required read access to the Ranger role you previously created, ensuring that any external client linked to that role can query the shared data.

Note: For more information about creating Ranger policies, see the Apache Ranger API documentation (`POST /public/v2/api/policy`).

The following policy grants the SalesDeptRole role SELECT access to the employees, departments, and dept_emp Iceberg tables, which together form the datashare1 Data Share:

```
curl -k -u systest:[***PASSWORD***] -H "Accept: application/json" -H "Conte
nt-Type: application/json" -X POST "https://my-knox-gateway-host.root.comops
.site:8443/gateway/cdp-share-management/ranger/service/public/v2/api/policy/
" -d '{"service":"cm_hive", "policyType": 0, "name": "datashare1", "descript
ion": "Policy for SELECT access to the shared Iceberg tables", "isEnabled":
true, "resources": { "database": { "values": ["emp_data"] }, "table": { "val
ues": ["employees", "departments", "dept_emp"] }, "column": { "values": ["*"
] } }, "policyItems": [ { "accesses": [ { "type": "select" } ], "users": [],
 "groups": [], "roles": ["SalesDeptRole"], "conditions": [] } ] }'
```

Related Information

Ranger Policies Overview

Supported REST Catalog APIs for accessing the data

To use the Iceberg data sharing in Cloudera, you must employ the REST client to perform specific operations.

The REST APIs from the specification defined by Apache Iceberg are available in the REST Catalog open API specification. Cloudera currently supports the REST APIs that allow read operations on Iceberg tables:

On premises gateway endpoints and Ozone S3 credential verification

For Cloudera on premises deployments, external clients and administrators reach the data-sharing gateway paths through the Knox topology instead of a cloud load balancer:

- https://[***KNOX-GATEWAY-HOST***]/[***DATALAKE-NAME***]/cdp-datashare-access/iceberg-rest/v1/namespaces — client ID and client secret generation.
- https://[***KNOX-GATEWAY-HOST***]/[***DATALAKE-NAME***]/cdp-datashare-access/iceberg-rest/... — REST Catalog namespace, table, and credentials endpoints.

When a table is stored on Cloudera Object Store (powered by Apache Ozone) S3, load the table and confirm that the response contains a s3a:// credential prefix that matches the translated Ozone link bucket, along with a session token:

```
curl -ivk -X GET -H "Content-Type: application/x-www-form-urlencoded" -H
 "Authorization: Bearer $[***MY-TOKEN-NAME***]" https://[***KNOX-GATEWAY-
HOST***]/[***DATALAKE-NAME***]/cdp-datashare-access/iceberg-rest/v1/namesp
aces/<namespace>/tables/<table>
```

A successful response includes a config block similar to the following:

```
{
  "s3.access-key-id": "ASIA...",
  "s3.secret-access-key": "...",
  "s3.session-token": "...",
```

```
  "s3.endpoint": "https://[***OZONE_S3_GATEWAY_HOST***]:9879",
  "s3.path-style-access": "true",
  "s3.remote-signing-enabled": "false"
}
```

Note: The credential prefix always matches the s3a:// link bucket name (*[***VOLUME***]-[***BUCKET***]*), not the original ofs:// table location. For the setup this verification depends on, see Configuring Ozone STS for on premises data sharing.

Retrieving the access token

Retrieving the access token works the same for all supported endpoints:

```
[***MY-TOKEN-NAME***]=$(curl -k -X POST -H  "Content-Type: applicatio
n/x-www-form-urlencoded" -d "client_id=[***CLIENT ID***]&client_secr
et=[***CLIENT SECRET***]&grant_type=client_credentials" "https://[***KNOX-
GATEWAY-HOST***]/[***DATALAKE-NAME***]/cdp-datashare-access/knoxtoken/api/
v2/token"  | jq -r '.access_token')
```

Using the endpoints must be always preceded by retrieving the access token.

List Databases: /v1/{prefix}/namespaces

List all namespaces at a certain level, optionally starting from a given parent namespace.

```
curl -ivk -X GET -H "Content-Type: application/x-www-form-urlencoded" -H
 "Authorization: Bearer $[***MY-TOKEN-NAME***]" https://[***KNOX-GATEWAY-
HOST***]/[***DATALAKE-NAME***]/cdp-datashare-access/iceberg-rest/v1/namesp
aces
```

```
My-token-name=$(curl -k -X POST -H  "Content-Type: application/x-www-form-ur
lencoded" -d "client_id=b9efc3dd-3695-4867-9e4c-523389c8a78b&client_secret=W
WpsbFptTXpaR1F0TXpZNU5TMDBPRFkzTFRsbE5HTXROVEl6TXpnNVl6aGhOemhpOjpNRGt5Tmpne
E1HUXRZak00TmkwMFpqTXhMVGczTTJZdFptTXhOekl6TmpVNFlqazI=&grant_type=client_cr
edentials" "https://my-cluster-gateway.example.cldr.work/my-cluster/cdp-data
share-access/knoxtoken/api/v2/token"  | jq -r '.access_token')

curl -ivk -X GET -H "Content-Type: application/x-www-form-urlencoded" -H "Au
thorization: Bearer $My-token-name" https://my-cluster-gateway.example.cldr.
work/my-cluster/cdp-datashare-access/iceberg-rest/v1/namespaces
```

List Tables: /v1/{prefix}/namespaces/{namespace}/tables

List all table identifiers under the specified namespace.

```
curl -ivk -X GET -H "Content-Type: application/x-www-form-urlencoded" -H
 "Authorization: Bearer $[***MY-TOKEN-NAME***]" https://[***KNOX-GATEWAY-
HOST***]/[***DATALAKE-NAME***]/cdp-datashare-access/iceberg-rest/v1/namesp
aces/<namespace>/tables
```

Example:

```
My-token-name=$(curl -k -X POST -H  "Content-Type: application/x-www-form-ur
lencoded" -d "client_id=b9efc3dd-3695-4867-9e4c-523389c8a78b&client_secret=W
WpsbFptTXpaR1F0TXpZNU5TMDBPRFkzTFRsbE5HTXROVEl6TXpnNVl6aGhOemhpOjpNRGt5Tmpne
E1HUXRZak00TmkwMFpqTXhMVGczTTJZdFptTXhOekl6TmpVNFlqazI=&grant_type=client_cr
edentials" "https://my-cluster-gateway.example.cldr.work/my-cluster/cdp-data
share-access/knoxtoken/api/v2/token"  | jq -r '.access_token')
```

```
curl -ivk -X GET -H "Content-Type: application/x-www-form-urlencoded" -H "Au
thorization: Bearer $My-token-name" https://my-cluster-gateway.example.cldr.
work/my-cluster/cdp-datashare-access/iceberg-rest/v1/namespaces/hive_rest_ai
rline_orc/tables
```

Load Tables: /v1/{prefix}/namespaces/{namespace}/tables/{table}

Load a table from the catalog.

```
curl -ivk -X GET -H "Content-Type: application/x-www-form-urlencoded" -H
 "Authorization: Bearer $[***MY-TOKEN-NAME***]" https://[***KNOX-GATEWAY-
HOST***]/[***DATALAKE-NAME***]/cdp-datashare-access/iceberg-rest/v1/namesp
aces/<namespace>/tables/icebergtable
```

Example:

```
My-token-name=$(curl -k -X POST -H  "Content-Type: application/x-www-form-ur
lencoded" -d "client_id=b9efc3dd-3695-4867-9e4c-523389c8a78b&client_secret=W
WpsbFptTXpaR1F0TXpZNU5TMDBPRFkzTFRsbE5HTXROVEl6TXpnNVl6aGhOemhpOjpNRGt5Tmpne
E1HUXRZak00TmkwMFpqTXhMVGczTTJZdFptTXhOekl6TmpVNFlqazI=&grant_type=client_cr
edentials" "https://my-cluster-gateway.example.cldr.work/my-cluster/cdp-data
share-access/knoxtoken/api/v2/token"  | jq -r '.access_token')

curl -ivk -X GET -H "Content-Type: application/x-www-form-urlencoded" -H "Au
thorization: Bearer $My-token-name" https://my-cluster-gateway.example.cldr.
work/my-cluster/cdp-datashare-access/iceberg-rest/v1/namespaces/hive_rest_ai
rline_orc/tables/airport_iceberg_external
```

Related Information

Iceberg REST Catalog API specification