CDP One

Migrating Data to CDP One

Date published: 2022-06-03
Date modified: 2022-08-15

https://docs.cloudera.com/

Legal Notice

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.

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

Overview.. 5

Migrating data from CDH to CDP One.. 5

Migrating HDFS and Hive data from CDH to CDP One.. 5
Migration prerequisites.. 5
Introduction to Replication Manager.. 7
HDFS data migration from CDH to CDP One...13
Hive migration from CDH to CDP One.. 18
Migrating Oozie workflows from CDH to CDP One...22
About Migrating Oozie workloads...23
Migration prerequisites.. 23
Setting up an external account... 23
Migrating Hue databases from CDH to CDP One... 23
Migrating HDFS native permissions to CDP One..25
Extracting HDFS native permissions... 25
Converting HDFS native permissions into Ranger HDFS policies........................... 26
Transforming Ranger HDFS policies into Ranger S3 policies..................................27
Importing Ranger AWS S3 policies... 28
Migrating workflows directly created in Oozie to CDP One.. 31
Migrating Sentry policies from CDH to CDP One.. 32
About Migrating Sentry policies.. 32

Migrating data from HDP to CDP One...40

Migrating HDFS data from HDP to CDP One...40
Migration prerequisites.. 40
About DistCp tool...41
Unbanning hdfs user in HDP cluster... 41
Before migrating... 42
HDFS data migration from HDP to CDP One... 42
Migrating HDFS native permissions to CDP One..42
Extracting HDFS native permissions... 43
Converting HDFS native permissions into Ranger HDFS policies........................... 44
Transforming Ranger HDFS policies into Ranger S3 policies..................................45
Importing Ranger AWS S3 policies... 46
Migrating Ranger policies from HDP to CDP One...49
About Migrating Ranger policies...49
Performing Export and Transform operations... 50
Performing Import operation.. 51
Supported Input parameters for Export operation..55
Supported Input parameters for Transform operation..57
Migrating Hive data from HDP 2.x or HDP 3.x to CDP One... 60
Migration prerequisites.. 63
HMS Mirror command summary...67
Migrating Hive metadata..68
HMS Mirror generated files.. 70
Verifying metadata migration..71
Migrating actual Hive data..71

Adjust AVRO table schema URLs.......... 72
Verifying actual Hive data migration.......... 72
Table locations.......... 73
Fixing statistics.......... 73
Changes to HDP Hive tables.......... 73

Overview

You can migrate HDFS and Hive data from CDH and HDP to CDP One. You learn about semantic changes in component versions that affect migration.

Migrating data from CDH to CDP One

How to migrate data from CDH to CDP One.

Migrating HDFS and Hive data from CDH to CDP One

An overview of the migration process from CDH to CDP One prepares you to migrate HDFS and Hive data to the AWS S3 endpoint.

Once fulfilling migration prerequisites, you set up security for communications between the CDH on-premise cluster and the destination cloud services. You can use the CDP Replication Manager service to migrate HDFS and Hive data from CDH or HDP to CDP One

Migration prerequisites

You need to know the prerequisites for migrating HDFS and Hive data from CDH to CDP One.

You must meet the following general prerequisites before starting the migration process:

- AWS is your Cloud provider. Migrating to Azure / GCP is currently not supported.
- Classic clusters (CDH/HDP/CDP Private Cloud Base) are on-premises clusters that host the data to be replicated to the CDP deployment. These clusters are on-premises versions of Cloudera Data Platform and must be registered on the Management Console before they can be used for data migration purposes. For more information, see Support Matrix for Replication Manager.
- Validate the CDH source cluster as per cluster requirements.
- CDH clusters must have been created using Cloudera Manager.
- Clusters that are not managed by Cloudera Manager cannot be registered to CDP.
- You must log in as the admin user to Cloudera Manager in the CDH cluster.
- Replication user needs to have hadoop admin privilege on source cluster.
- On the source cluster which is running CDH, make sure the user we specify on the replication manager source cluster side has hadoop admin privileges.
- Your source cluster supports compatible versions of Cloudera Manager and Cloudera Runtime shown in the following table:

Source cluster	Earliest supported Cloudera Manager	Earliest supported Cloudera Runtime
CDH 5	6.3.0	5.1.0
CDH 6	6.3.0	6.1.0
CDH 6	7.3.1	6.3.3

The only required migration tool is Cloudera Replication Manager, which is pre-installed in Cloudera Manager and licensed for use in Data Hub.

Specifically note that:

- Your CDH cluster is registered on the CDP Management Console. For more information, see Adding a CDH cluster.
- Open communication ports between the source CDH cluster and CDP list.

- Set up line-of-sight from the CDH cluster to the AWS endpoint.
- Ensure adequate network connectivity between the CDH cluster and AWS endpoint.
- Create an external account on the CDH cluster that has a non-expiring access credentials and secret key pair.
- You must have the license to perform your tasks in Replication Manager. To understand more about Cloudera license requirements, see Managing Licenses.
- Allocate storage on HDFS to create HDFS snapshots that allows administrators to modify the active file-system.
- You must be a part of the super-group on each host on the CDH cluster.

Ports for Replication Manager on CDP Public Cloud

You must open certain ports on the CDH source host to allow communication between the source on-premises cluster and CDP.

Connectivity required	Default Port	Type	Description	Required for the following replication scenarios
Data transfer from CDH cluster hosts to AWS S3	80 or 443 (TLS)	Outbound	Outgoing port. All CDH nodes must be able to access S3 endpoints	HDFS to Cloud replication (S3 Target) Hive Replication
Cloudera Manager Admin Console HTTP	7180 or 7183 when TLS enabled	Inbound	Incoming port. Open on the source cluster to enable the target Cloudera Manager (in cloud) to communicate to the on-premises Cloudera Manager.	Hive Replication (Not applicable for HDFS migration)
Classic cluster setup (if using Replication Manager App)	Varies based on CCMv1 (6000-6049) or CCMv2 (443)	Outbound	Connecting the source CDH cluster to the CDP Management Console via Cluster Connectivity Manager (CCM) Outbound network access for CCM documented here and CCM overview is documented here	Required when using the Replication Manager UI/ App

Setting up an external account

As CDH Administrator, you need to create an External Account on the CDH cluster that has an access key and secret key pair for accessing the cloud storage used in the CDP One environment.

About this task

Using the external account along with the cloud storage path, administrators can create one or more HDFS replication schedules to migrate the HDFS data from the CDH cluster to the cloud storage used in the CDP One environment.You must have valid Amazon S3 credentials to access cloud storage for migrating data.

You must raise a support case with Cloudera to provide the key pair.

You must have valid Amazon S3 credentials to access cloud storage for migrating data. You add the access key / secret key pair to the account to access CDP cloud storage s3a://<data lake bucket path>. The access key / secret key pair ensures that the external account configured on CDH can successfully access the S3 buckets used by Data Lake.

Procedure

1. In Cloudera Manager#Administration, click Add Access Key Credentials.
2. In Edit Credential, name the credential and enter the AWS Access Key ID and AWS Secret Key.
3. Save changes,

Setting up SSL/TLS certificate exchange

In order to replicate data securely, you must configure an SSL/TLS certificate exchange between the two Cloudera Manager instances that manage the source and target clusters.

About this task

The following sample commands use the open-jdk-11 java version. Use the Java version that you use in CDP clusters in these commands.

Procedure

1. List the contents of the trustore file located in the source cluster Cloudera Manager.

   ```
   /usr/lib/jvm/java-openjdk-11/bin/keytool -list -keystore /var/lib/
   cloudera-scm-agent/agent-cert/cm-auto-global_truststore.jks -store
   pass [***password***]
   ```

   The global truststore password is available in the /etc/hadoop/conf/ssl-client.xml file.
2. Export the certificate contents to the cert.txt file.

   ```
   /usr/java/default/bin/keytool -exportcert -keystore /var/lib/cloudera-scm-
   agent/agent-cert/cm-auto-global_truststore.jks -alias cmrootca-0 -file ./c
   ert.txt -storepass [***truststore_pass***]
   ```

3. Copy the cert.txt file to all the hosts of the destination cluster Cloudera Manager securely.

   ```
   $ mv cert.txt othercert.txt
   $ scp
   ```

4. Import the certificate into the trustore file on all the hosts of the destination cluster Cloudera Manager.

   ```
   /usr/java/default/bin/keytool -importcert -noprompt -v -trustcac
   erts -keystore /var/lib/cloudera-scm-agent/agent-cert/cm-auto-gl
   obal_truststore.jks -alias cmrootca-1 -file ./othercert.txt --storep
   ass [***truststore_password***]
   ```

5. Run steps 1 and 2 in the target cluster Cloudera Manager, and then copy the cert.txt file to all the hosts in the source cluster Cloudera Manager securely, and import the certificate into the trustore file on all the hosts of the source cluster Cloudera Manager (steps 3 and 4).

Cloudera license requirements for Replication Manager

You must have the required license to perform your tasks in Replication Manager.

Specifically, the source cluster Cloudera Manager must have a Cloudera Enterprise license in order to use the Replication Manager.

To learn more about Cloudera license requirements, see Managing Licenses.

Introduction to Replication Manager

Replication Manager is a service to copy and migrate data from CDH 5.13+ version and above clusters to CDP One endpoint. The currently supported Public Cloud service is Amazon S3.

Accessing the Replication Manager service

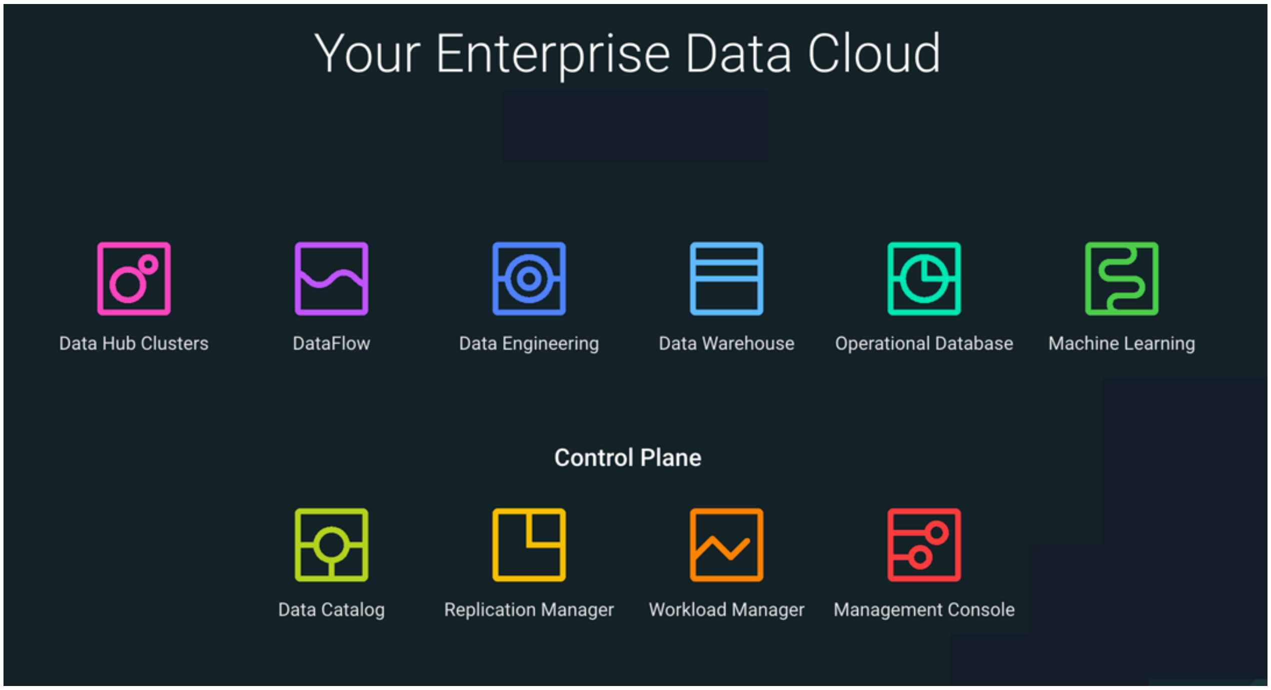
You can access the Replication Manager service by logging into Cloudera Data Platform.

When you log into Cloudera Data Platform, the CDP Public Cloud web interface appears. Click Replication Manager. The Overview page of the Replication Manager appears.

The Replication Manager has the following pages:

- Overview
- Classic Clusters
- Cloud Credentials
- Replication Policies

The following image shows the CDP Public Cloud web interface:



You can also access the Replication Manager service by logging into the Management Console. In the Management Console, click the

▦

icon and select Replication Manager.

How replication policies work

In Replication Manager, you create replication policies to establish the rules you want applied to your replication jobs.

The policy rules you set can include which cluster is the source and which is the destination, what data is replicated, what day and time the replication job occurs, the frequency of job runs, and bandwidth restrictions.

The first time you run a job (an instance of a policy) with data that has not been previously replicated, the Replication Manager creates a new folder or database and bootstraps the data. During a bootstrap operation, all data is replicated from the source cluster to the destination. As a result, the initial execution of a job can take a significant amount of time, depending on how much data is being replicated, network bandwidth, and so on. So you should plan the bootstrap operation accordingly.

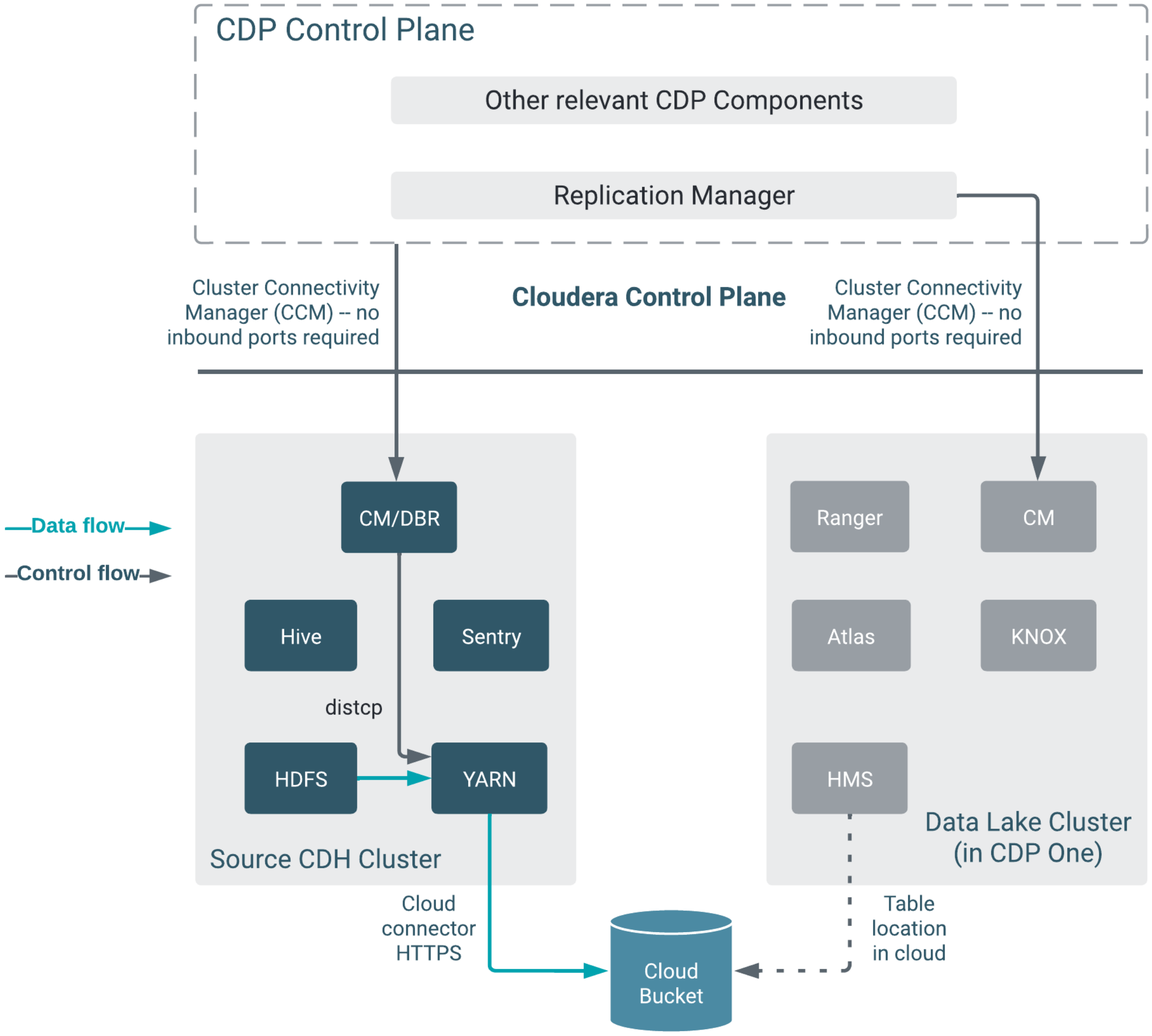
After the bootstrap operation succeeds, an incremental data replication is automatically performed. This job synchronizes, between the source and destination clusters, any events that occurred during the bootstrap process. After the data is synchronized, the replicated data is ready for use on the destination. Data is in a consistent state only after incremental replication has captured any new changes that occurred during bootstrap.

Subsequent replication jobs from the same source location to the same target on the destination are incremental, so only the changed data is copied.

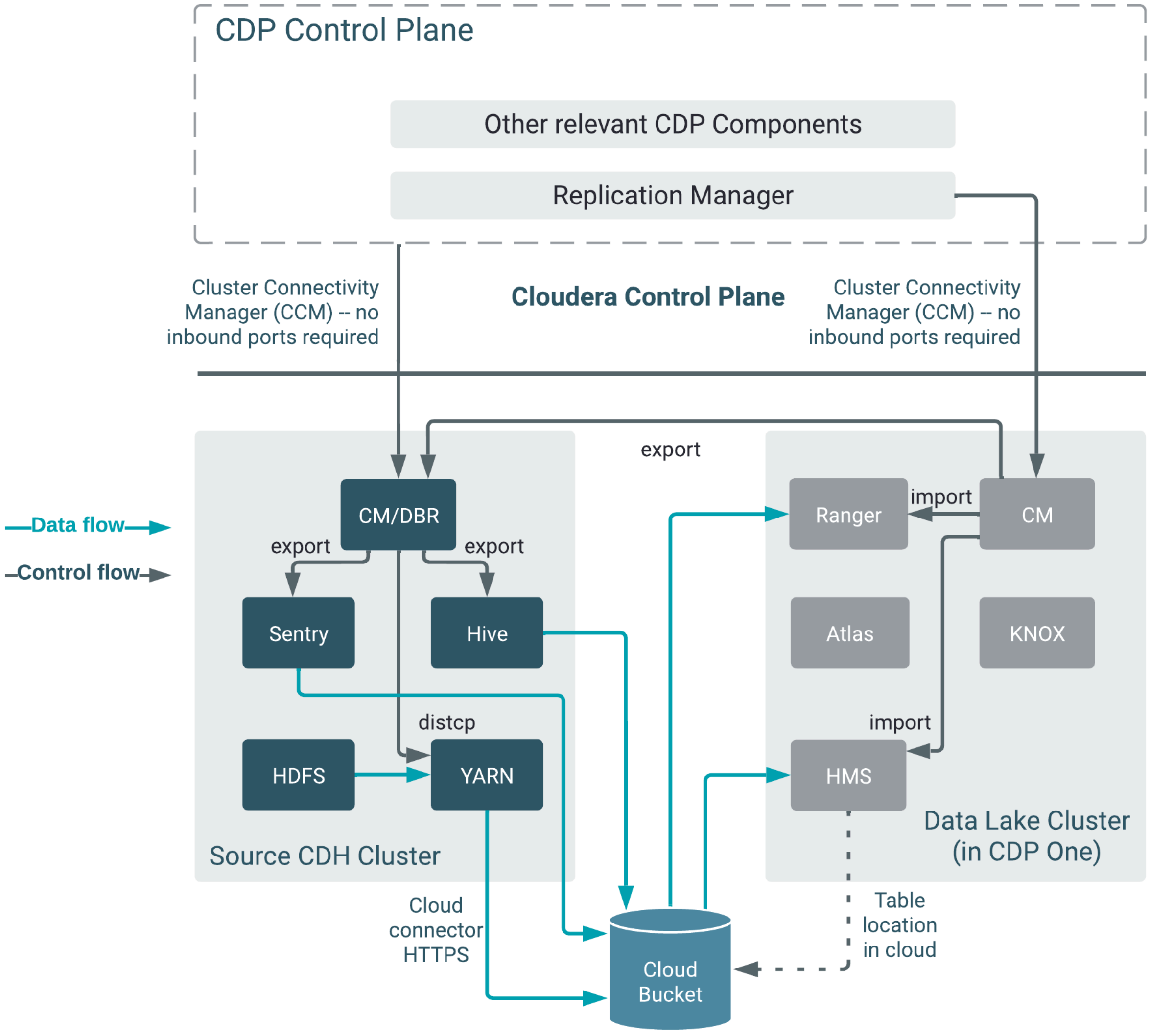
When a bootstrap operation is interrupted, such as due to a network failure or an unrecoverable error, the Replication Manager does not retry the job instead it runs the job at the next scheduled interval, if available. Therefore, if the bootstrap operation is interrupted, you must manually correct the issue and then run the policy.

When scheduling how often you want a replication job to run, you should consider the recovery point objective (RPO) of the data being replicated; that is, what is the acceptable lag time between the active site and the replicated data on the destination.

The following diagram shows the HDFS replication architecture:



The following diagram shows the Hive replication architecture:



Replication policy considerations

You should take into consideration certain guidelines when creating or modifying a replication policy. It is important for you to understand the security restrictions and encryption policies within Replication Manager.

The guidelines you need to consider before you create or modify a replication policy includes:

Data security

To use an S3 cluster for your policy, register your credentials on the Cloud Credentials page.

A user with access to the Replication Manager user interface has the ability to browse, within the Replication Manager UI, the folder structure of any clusters enabled for Replication Manager.

Therefore, users can view folders, files, and databases in the Replication Manager user interface, that they might not have access to in HDFS. Users cannot view from the Replication Manager UI the content of files on the source or destination clusters. Nor do these administrators have the ability to modify or delete folders or files that are viewable from the Replication Manager UI.

Replication Manager policy properties and settings

Consider the recovery point objective (RPO) of the data being replicated when you schedule a replication policy. The RPO is the acceptable lag time between the active site and the replicated data on the destination. Ensure that the frequency is set so that a job finishes before the next job starts.

Jobs based on the same policy cannot overlap. If a job is not completed before another job starts, the second job does not execute and is given the status Skipped. If a job is consistently skipped, you might need to modify the frequency of the job.

Specify bandwidth per map, in MBps. Each map is restricted to consume only the specified bandwidth. This is not always exact. The map throttles back its bandwidth consumption during a copy in such a way that the net bandwidth used tends towards the specified value.

Cluster requirements

- Pair the clusters before you include them in a replication policy.
- With a single cluster, you can replicate data on-premises to the cloud.
- With a single cluster, you cannot replicate data on-premises to on-premises.
- If the clusters are Replication-Manager enabled, it appears in the Source Cluster or Destination in the Create Policy wizard. You must ensure that the clusters you select are healthy before you start a policy instance (job).

Working with cloud credentials

The Cloud Credentials page shows the registered cloud credentials for Replication Manager. To replicate data to a storage cloud account, you must register the cloud credentials, so that the Replication Manager can access your cloud account. The supported cloud storage accounts are Amazon S3.

On the Cloud Credentials page, you can add, update, or delete cloud credentials. Before you register an Amazon S3 cloud account, ensure the cloud bucket requirements are met.

Adding cloud credentials

You can perform the following tasks on the Cloud Credentials page to manage cloud credentials.

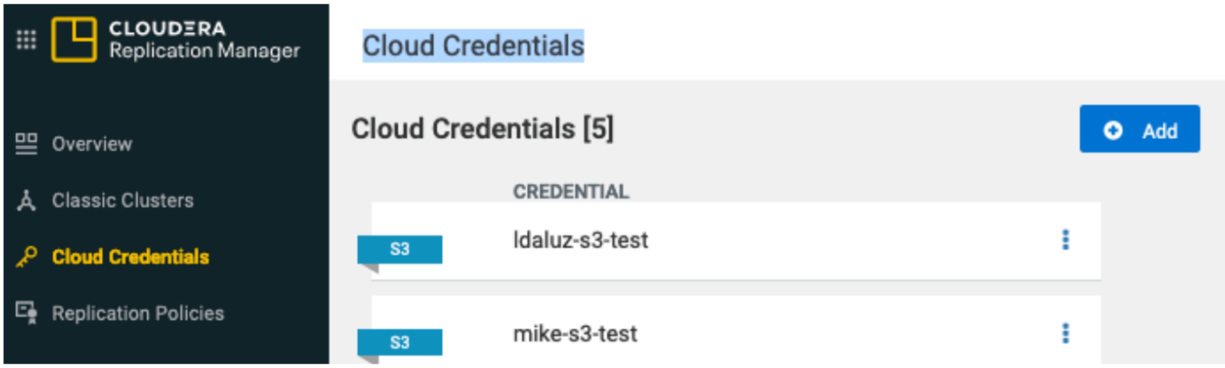
About this task

You can add cloud credentials to migrate data to the cloud.

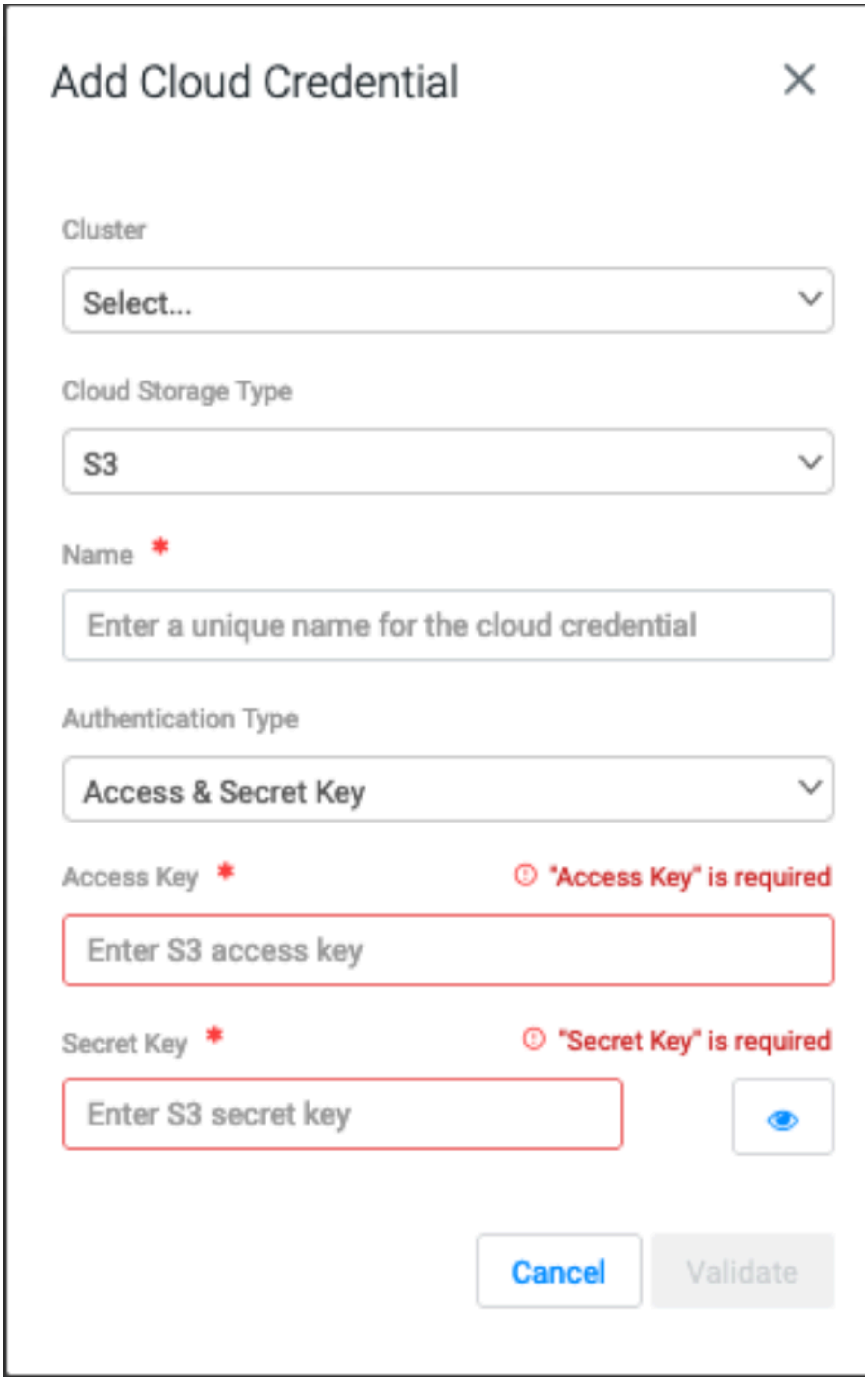
Unregistered credentials can impact the replication process. Credentials associated with a cluster node that do not have updated credentials are called unregistered credentials. For example, if a node is down when the credentials are changed on a bucket or when the node is brought up that has the old credentials.

Procedure

1. Log into CDP public cloud Management Console > Replication Manager.
2. Click Cloud Credentials on the left pane, click Add.



3. Fill out the information as seen in the image and click Validate.



Add Cloud Credential

Cluster: Select...

Cloud Storage Type: S3

Name *: Enter a unique name for the cloud credential

Authentication Type: Access & Secret Key

Access Key *: Enter S3 access key — "Access Key" is required

Secret Key *: Enter S3 secret key — "Secret Key" is required

4. Once validated successfully, click Save.

Update cloud credentials

You can update the cloud credentials based on various factors.

When the bucket configuration such as secret or access keys, bucket name or endpoint, and encryption type is changed, it can affect the Replication Manager replication policy run and might require an update to the Replication Manager cloud credentials.

Credential changes are picked up by the next run of the policy. When you change the credentials, the in-progress policy runs might fail but the succeeding runs pick up the changes.

To update a cloud credential, click Actions > Update option.

Delete cloud credentials

You can delete unwanted credentials from the Replication Manager.

When you delete cloud credentials, the replication policies that use the deleted cloud credentials might fail. To avoid failures, delete the Replication Manager cloud policies associated with the deleted credentials and recreate the policies with the new credentials. You can view a list of policies associated with specific credentials on the Cloud Credentials page.

To delete a cloud credential, click Actions > Delete option.

HDFS data migration from CDH to CDP One

To create a HDFS replication policy from on-premises to the cloud account, you must register your cloud account credentials with Replication Manager service, so that Replication Manager can access your cloud storage.

Before performing HDFS replication using CDH clusters, see Working with cloud credentials.

Replication Manager supports replication of HDFS data from cluster to cloud storage. The replication policy runs on the cluster and pushes the data from cloud storage. The cluster can be an on-premises or IaaS cluster with data on local HDFS. The cluster requires HDFS, YARN, Ranger, and Knox services to perform replication.

You can schedule taking HDFS snapshots for replication in the Replication Manager. HDFS snapshots are read-only point-in-time copies of the filesystem. You can enable snapshots on the entire filesystem, or on a subtree of the filesystem. In Replication Manager, you take snapshots at a dataset level. Understanding how snapshots work and some of the benefits and costs involved can help you to decide whether or not to enable snapshots.

For more information, see HDFS snapshots.

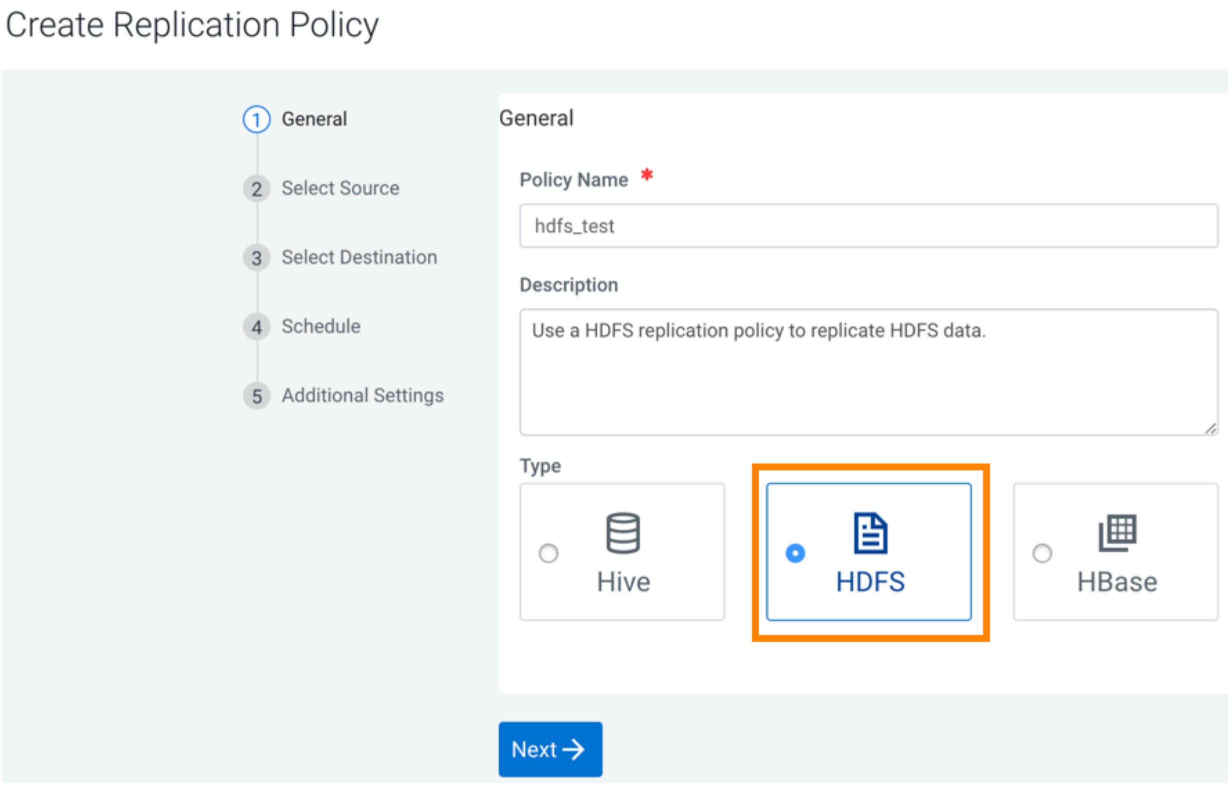
Creating a HDFS replication policy

You can use a HDFS replication policy to replicate data from on-premises to cloud. Before you create a new replication policy, you must register a cloud account with the Replication Manager service.

Procedure

1. On the Management Console > Replication Manager > Replication Policies page, click Add Policy.

2. In the Create Replication Policy wizard, select HDFS.



Create Replication Policy

1 General
2 Select Source
3 Select Destination
4 Schedule
5 Additional Settings

General

Policy Name *

hdfs_test

Description

Use a HDFS replication policy to replicate HDFS data.

Type

○ Hive

● HDFS

○ HBase

Next →

3. Enter the HDFS replication Policy Name and Description. Click Next.
4. Select Source Cluster.
5. Enter the Source Path where the source data resides.
6. In the Run As Username (on source) field, enter the source user.
7. Click Next.
8. Choose the destination Type as S3.
9. Select a Cloud Credential on Source. You can also add cloud credentials using the Add Cloud Credential link.
10. In the Path field, enter the values based on the Type you need and choose S3 type, provide a folder path in the bucket_name/path format.
11. Click Validate Policy.
    The Replication Manager verifies the data with a status Validate Policy Source and Destination information.
12. Click Next to schedule the replication policy.
13. On the Schedule page, choose one of the following options:
    - Run Now (Default) - The replication policy is immediately submitted and processed.
    - Schedule Run - The replication policy can be scheduled to run at specified time intervals.
14. (Optional) In the Repeat field, you can choose one of the following options:
    - Does Not Repeat

      Custom - In the Custom Recurrence dialog box, choose the time, date, and the frequency to run the policy.
15. Click Next.

16. On the Additional Settings page, enter or choose the values as necessary:

Option	Description
YARN Queue Name	Enter the name of the YARN queue for the cluster to which the replication job is submitted if you are using Capacity Scheduler queues to limit resource consumption. The default value for this field is default.
Maximum Maps Slots	Set the maximum number of map tasks (simultaneous copies) per replication job. The default value is 20.
Maximum Bandwidth	Adjust this setting so that each map task is throttled to consume only the specified bandwidth. Each map task ((simultaneous copy) is restricted to consume only the specified bandwidth. This is not always exact. The map throttles back its bandwidth consumption during a copy in such a way that the net bandwidth used tends towards the specified value. You can adjust this setting so that each map task is throttled to consume only the specified bandwidth so that the net bandwidth used tends towards the specified value. The default value for the bandwidth is 100MB per second for each mapper.
Path Exclusion	Enter a regular expression-based path. When you add an exclusion, include the snapshotted relative path for the regex. For example, to exclude the /user/bdr directory, use the `.*/user/\.snapshot/.+/bdr.*` regular expression, which includes the snapshots for the `bdr` directory. To exclude top-level directories from replication in a globbed source path, you can specify the relative path for the regex without including .snapshot in the path. For example, to exclude the bdr directory from replication, use the `.*/user+/bdr.*` regular expression. You can add more than one regular expression to exclude.
Replication Strategy	Choose one of the following replication strategies to determine whether the file replication tasks should be distributed among the mappers statically or dynamically. • Static distributes file replication tasks among the mappers up front to achieve an uniform distribution based on the file sizes. • Dynamic distributes the file replication tasks in small sets to the mappers, and as each mapper completes its tasks, it dynamically acquires and processes the next unallocated set of tasks. The default replication strategy is Dynamic.
MapReduce Service	Choose the MapReduce or YARN service to use.
Log Path	Enter an alternate path for the logs, if required.

Option	Description
Error Handling	Select the following options as necessary: • Skip Checksum Checks - Determines whether to skip checksum checks on the copied files. If selected, checksums are not validated. Checksums are checked by default. **Note:** You must skip checksum checks to prevent replication failure due to non-matching checksums in the following cases: • Replications from an encrypted zone on the source cluster to an encrypted zone on a destination cluster. • Replications from an encryption zone on the source cluster to an unencrypted zone on the destination cluster. • Replications from an unencrypted zone on the source cluster to an encrypted zone on the destination cluster. Checksums are used for two purposes: • To skip replication of files that have already been copied. If Skip Checksum Checks is selected, the replication job skips copying a file if the file lengths and modification times are identical between the source and destination clusters. Otherwise, the job copies the file from the source to the destination. • To redundantly verify the integrity of data. However, checksums are not required to guarantee accurate transfers between clusters. HDFS data transfers are protected by checksums during transfer and storage hardware also uses checksums to ensure that data is accurately stored. These two mechanisms work together to validate the integrity of the copied data. • Skip Listing Checksum Checks - Whether to skip checksum check when comparing two files to determine whether they are same or not. If skipped, the file size and last modified time are used to determine if files are the same or not. Skipping the check improves performance during the mapper phase. Note that if you select the Skip Checksum Checks option, this check is also skipped. • Abort on Error - Whether to abort the job on an error. If selected, files copied up to that point remain on the destination, but no additional files are copied. Abort on Error is not selected by default. • Abort on Snapshot Diff Failures - If a snapshot diff fails during replication, the replication policy uses a complete copy to replicate data. If you select this option, the policy aborts the replication when it encounters an error instead.

Option	Description
Preserve	Choose the required options to preserve the block size, replication count, permissions (including ACLs), and extended attributes (XAttrs) as they exist on the source file system, or to use the settings as configured on the destination file system. By default source system settings are preserved. When Permission is selected, and both the source and destination clusters support ACLs, replication preserves ACLs. Otherwise, ACLs are not replicated. When Extended attributes is selected, and both the source and destination clusters support extended attributes, replication preserves them. (This option only displays when both source and destination clusters support extended attributes.) If you select one or more of the Preserve options and you are replicating to S3, the values all of these items are saved in metadata files on S3. When you replicate from S3 to HDFS, you can select which of these options you want to preserve. **Note:** To preserve permissions to HDFS, you must be running as a superuser on the destination cluster. Use the Run As Username option to set the username.
Delete Policy	Choose the required options to determine whether the files that were deleted on the source should also be deleted from the destination directory. This policy also determines the handling of files in the destination location that are unrelated to the source. Options include: • Keep Deleted Files - Retains the destination files even when they no longer exist at the source. This is the default option. • Delete to Trash - If the HDFS trash is enabled, files are moved to the trash folder. This is not supported when replicating to S3 • Delete Permanently - Uses the least amount of space; use with caution.
Alerts	Choose the required options to generate alerts for various state changes in the replication workflow. You can alert on failure, on start, on success, or when the replication workflow is aborted.

17. Click Create.

Once the replication policy runs successfully, you can view the replication job status on the Replication Policies page. Verify whether the job starts and runs as expected.

Verifying HDFS data migration

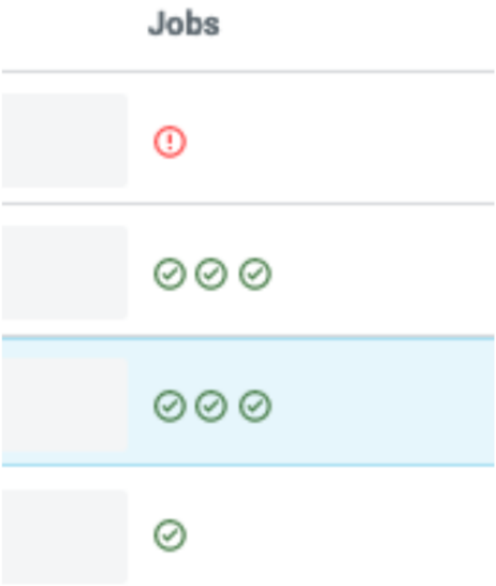
Post-migration process, you must check if the migration process is completed successfully by verifying the data on the AWS S3 endpoint.

There are two ways to verify the HDFS data replication:

1. **Verification from Replication Manager App.**

   Once the replication policy is created and scheduled, go to CDP public cloud -> choose replication manager -> click Replication Policies on the left pane.

   You can view the number of replication policies that was created and the replication jobs status as seen from the following image:

   The green check mark indicates successful replication jobs. The red escalation mark indicates the failed replication jobs.
2. **Verifying from the AWS S3 Console.**

   Log into AWS S3 > click Buckets on left pane -> search the bucket name in the search box of right pane.

   After accessing the target bucket, search the path name where the HDFS data is migrated. View if the HDFS data in source CDH cluster is replicated in the target S3 bucket path

Hive migration from CDH to CDP One

To create a Hive replication policy from on-premises to the cloud account, you must register your cloud account credentials with the Replication Manager service, so that Replication Manager can access your cloud storage.

Before performing Hive replication using CDH clusters, see Working with cloud credentials.

Creating a Hive replication policy

To replicate Hive metadata from on-premises to cloud, you must set the Ranger policy in Ranger, and then create the Hive replication policy in Replication Manager.

About this task

The Apache Ranger access policy model consists of the following components:

- Specification of the resources that you can apply to a replication policy which includes the HDFS files and directories; Hive databases, tables, and columns; and HBase tables, column-families, and columns.
- Specification of access conditions for specific users and groups.

Note: You must set the Ranger policy for the hdfs user on the target cluster to perform all operations on all databases and tables. The same user role is used to import the Hive Metastore. The hdfs user should have access to all Hive datasets, including all operations. Otherwise, the Hive import fails during the replication process.

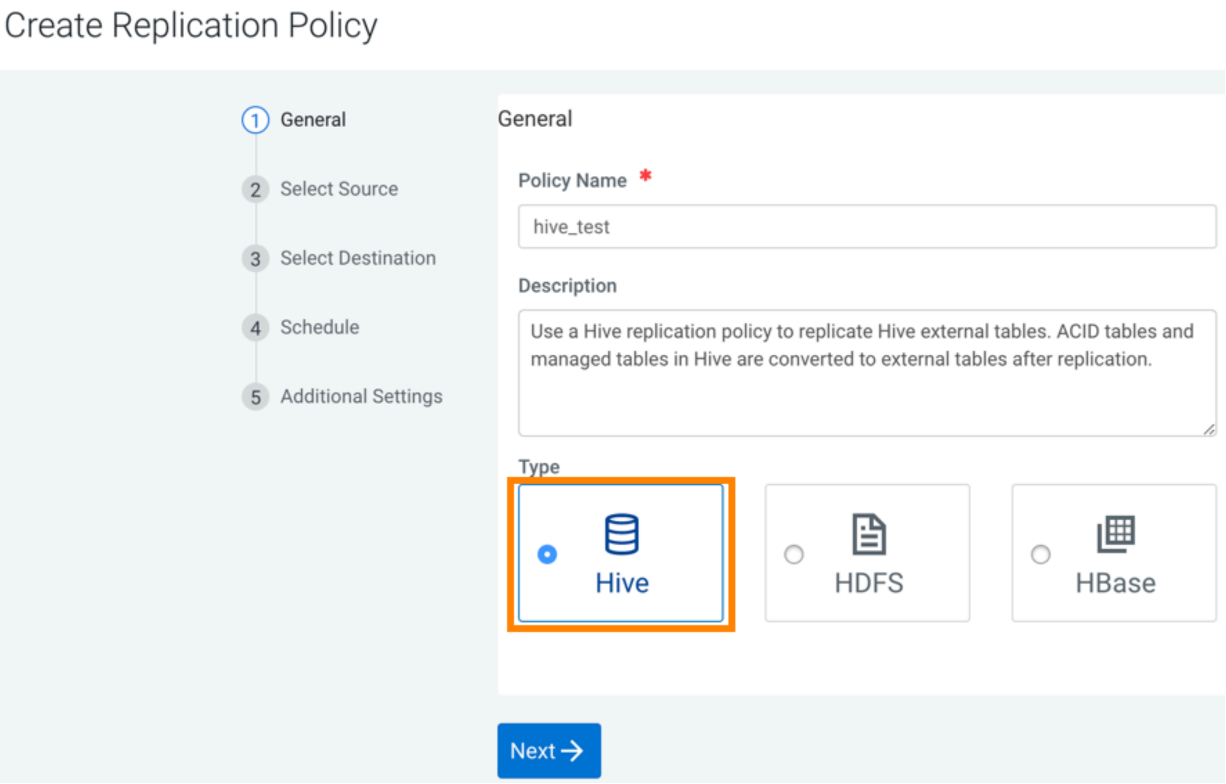
On the target cluster, the hive user must have Ranger admin privileges. The same hive user performs the metadata import operation.

1. Log in to the Ranger Admin UI.
2. In the Hadoop_SQL section, provide the hdfs user permission to "all-database, table, column" in hdfs.

Note: You can replicate data from on-premises to cloud with a single cluster. The Metastore must be running on the cloud.

Procedure

1. On the Management Console > Replication Manager > Replication Policies, click Add Policy.
2. In the Create Replication Policy wizard, select Hive.



Create Replication Policy

General

Policy Name *

hive_test

Description

Use a Hive replication policy to replicate Hive external tables. ACID tables and managed tables in Hive are converted to external tables after replication.

Type

● Hive ○ HDFS ○ HBase

3. Enter the Hive replication Policy Name and Description. Click Next.
4. Select the Source Cluster.
5. Enter the database name and table name in Source Databases and Tables. Click the + icon to enter more databases and tables as necessary.
6. Enter the username in Source User. Ensure that the user has the necessary permissions to replicate data. The user should have Read access to the data.
7. Click Next.
8. Select the Destination Data Lake cluster. The Managed Warehouse Path and the Hive External Table Base Directory path for the Data Lake appears.

    a. Administrators can edit the Hive External Table Base Directory field to add another path to override the default storage location for replicated Hive external tables. Before you add another path to override the default storage location, ensure that the following steps are complete in the Ranger UI:

        1. Alter the ranger policy Default: Hive warehouse locations in cm_s3 service to allow the Hive service to access the updated locations of S3 bucket path.
        2. Manually update the Ranger and Sentry permissions.

9. Select a Cloud Credential on Source. You can also add cloud credentials using the Add Cloud Credentials link.
10. Enter the source cluster user name in Run as Username.
11. Click Validate Policy.

    The Replication Manager verifies the data with a status Validate Policy Source and Destination information.
12. Click Next to schedule the replication policy.
13. On the Schedule page, choose one of the following options:
    - Run Now (Default) - The replication policy is immediately submitted and processed.
    - Schedule Run - The replication policy can be scheduled to run at specified time interval.
14. In the Repeat field, you can choose one of the following options:
    - Does Not Repeat
    - Custom - In the Custom Recurrence dialog box, choose the time, date, and the frequency to run the policy.
15. Click Next.
16. On the Additional Settings page, enter the values as necessary:

Option	Description
YARN Queue Name	Enter the name of the YARN queue for the cluster to which the replication job is submitted if you are using Capacity Scheduler queues to limit resource consumption. The default value for this field is default.
Maximum Maps Slots	Set the maximum number of map tasks (simultaneous copies) per replication job. The default value is 20.
Maximum Bandwidth	Adjust this setting so that each map task is throttled to consume only the specified bandwidth. Each map task ((simultaneous copy) is restricted to consume only the specified bandwidth. This is not always exact. The map throttles back its bandwidth consumption during a copy in such a way that the net bandwidth used tends towards the specified value. You can adjust this setting so that each map task is throttled to consume only the specified bandwidth so that the net bandwidth used tends towards the specified value. The default value for the bandwidth is 100MB per second for each mapper.
Path Exclusion	Enter a regular expression-based path. When you add an exclusion, include the snapshotted relative path for the regex. For example, to exclude the /user/bdr directory, use the `.*/user/\.snapshot/.+/bdr.*` regular expression, which includes the snapshots for the `bdr` directory. To exclude top-level directories from replication in a globbed source path, you can specify the relative path for the regex without including .snapshot in the path. For example, to exclude the bdr directory from replication, use the `.*/user+/bdr.*` regular expression. You can add more than one regular expression to exclude.
Replication Strategy	Choose one of the following replication strategies to determine whether the file replication tasks should be distributed among the mappers statically or dynamically. • Static distributes file replication tasks among the mappers up front to achieve an uniform distribution based on the file sizes. • Dynamic distributes the file replication tasks in small sets to the mappers, and as each mapper completes its tasks, it dynamically acquires and processes the next unallocated set of tasks. The default replication strategy is Dynamic.
MapReduce Service	Choose the MapReduce or YARN service to use.
Log Path	Enter an alternate path for the logs, if required.

Error Handling	Select the following options as necessary: • Skip Checksum Checks - Determines whether to skip checksum checks on the copied files. If selected, checksums are not validated. Checksums are checked by default. **Note:** You must skip checksum checks to prevent replication failure due to non-matching checksums in the following cases: • Replications from an encrypted zone on the source cluster to an encrypted zone on a destination cluster. • Replications from an encryption zone on the source cluster to an unencrypted zone on the destination cluster. • Replications from an unencrypted zone on the source cluster to an encrypted zone on the destination cluster. Checksums are used for two purposes: • To skip replication of files that have already been copied. If Skip Checksum Checks is selected, the replication job skips copying a file if the file lengths and modification times are identical between the source and destination clusters. Otherwise, the job copies the file from the source to the destination. • To redundantly verify the integrity of data. However, checksums are not required to guarantee accurate transfers between clusters. HDFS data transfers are protected by checksums during transfer and storage hardware also uses checksums to ensure that data is accurately stored. These two mechanisms work together to validate the integrity of the copied data. • Skip Listing Checksum Checks - Whether to skip checksum check when comparing two files to determine whether they are same or not. If skipped, the file size and last modified time are used to determine if files are the same or not. Skipping the check improves performance during the mapper phase. Note that if you select the Skip Checksum Checks option, this check is also skipped. • Abort on Error - Whether to abort the job on an error. If selected, files copied up to that point remain on the destination, but no additional files are copied. Abort on Error is not selected by default. • Abort on Snapshot Diff Failures - If a snapshot diff fails during replication, the replication policy uses a complete copy to replicate data. If you select this option, the policy aborts the replication when it encounters an error instead.
Preserve	Choose the required options to preserve the block size, replication count, permissions (including ACLs), and extended attributes (XAttrs) as they exist on the source file system, or to use the settings as configured on the destination file system. By default source system settings are preserved. When Permission is selected, and both the source and destination clusters support ACLs, replication preserves ACLs. Otherwise, ACLs are not replicated. When Extended attributes is selected, and both the source and destination clusters support extended attributes, replication preserves them. (This option only displays when both source and destination clusters support extended attributes.) If you select one or more of the Preserve options and you are replicating to S3 or ADLS, the values all of these items are saved in metadata files on S3 or ADLS. When you replicate from S3 or ADLS to HDFS, you can select which of these options you want to preserve. **Note:** To preserve permissions to HDFS, you must be running as a superuser on the destination cluster. Use the Run As Username option to set the username.

Delete Policy	Choose the required options to determine whether the files that were deleted on the source should also be deleted from the destination directory. This policy also determines the handling of files in the destination location that are unrelated to the source. Options include: • Keep Deleted Files - Retains the destination files even when they no longer exist at the source. This is the default option. • Delete to Trash - If the HDFS trash is enabled, files are moved to the trash folder. This is not supported when replicating to S3 or ADLS. • Delete Permanently - Uses the least amount of space; use with caution.
Alerts	Choose the required options to generate alerts for various state changes in the replication workflow. You can alert on failure, on start, on success, or when the replication workflow is aborted.
Sentry Permissions	Choose the following Sentry permissions as necessary: • Include Sentry Permissions with Metadata - Select this option to migrate Sentry permissions during the replication job. • Exclude Sentry Permissions from Metadata - Select this option if you do not want to migrate Sentry permissions during the replication job. • Skip URI Privileges - Select this option if you do not want to include URI privileges when you migrate Sentry permissions. During migration, the URI privileges are translated to point to an equivalent location in S3. If the resources have a different location in Amazon S3, do not migrate the URI privileges because the URI privileges might not be valid.
Replication Option	Specify metadata and data, or metadata only.
Directory for metadata file	The folder path in the destination cluster to save the metadata file. If the folder does not exist, Replication Manager creates a new folder.

17. Click Create.

Verifying Hive data migration

After the replication policy runs successfully, you can view the replication job status on the Replication Policies page.

An administrator can verify whether the job starts and runs as expected using the following steps:

- For HDFS data replication, check the cloud storage path (S3 bucket path) to verify whether the data was successfully copied in the specified bucket.
- For Hive metadata replication, verify whether the specified source database, along with tables, partitions, UDFs, and column stats are available in the Data Lake HMS instance. For this, the administrator can use a Data Hub cluster and run the corresponding queries using Hue or beeline.
- For Ranger policies, query the Ranger policies to ensure that the Sentry policies are properly mapped as Ranger policies for the right users and groups.

Note: If the CDH source database contains functions, you must explicitly run the reload function command to view the migrated replication functions in the target location. You can perform and monitor various tasks after running a replication policy. You can view the job progress, notifications, and replication logs. You can use the advanced options to optimize a job run. You can suspend a job and also activate a suspended job. You can edit the replication policy as necessary.

Migrating Oozie workflows from CDH to CDP One

Hue stoores the workflows within the Hue database which is created using Hue. The data residing in Hue is migrated to CDP One.

About Migrating Oozie workloads

The Oozie database on the source CDH cluster is automatically configured to purge data that is older than 30 days. If you need to retain this data, Cloudera recommends that you back up the data and make it available for running simple SQL queries. See Back Up the Databases.

Oozie workflows, query history, and saved query are some of the workloads that get migrated to CDP One.

Most Oozie workloads and configurations will need to be migrated manually to the destination cluster. Although you can copy the XML and property files from the source cluster, these files require extensive modifications before they can be used in the destination cluster. Cloudera Professional Services can assist you with these migrations.

Migration prerequisites

You need to know the prerequisites for migrating Hue databases from CDH to CDP One.

You must meet the following general prerequisites before starting the migration process:

- AWS is your Cloud provider. Migrating to Azure / GCP is currently not supported.
- You can log in as the admin user to Cloudera Manager in the CDH cluster.
- CDH clusters must have been created using Cloudera Manager.

Setting up an external account

As CDH Administrator, you need to create an External Account on the CDH cluster that has an access key and secret key pair for accessing the cloud storage used in the CDP One environment.

About this task

You must raise a support case with Cloudera to provide the key pair.

You must have valid Amazon S3 credentials to access cloud storage for migrating data. You add the access key / secret key pair to the account to access CDP cloud storage s3a://<data lake bucket path>. The access key / secret key pair ensures that the external account configured on CDH can successfully access the S3 buckets used by Data Lake.

Procedure

1. In Cloudera Manager#Administration, click Add Access Key Credentials.
2. In Edit Credential, name the credential and enter the AWS Access Key ID and AWS Secret Key.
3. Save changes,

Migrating Hue databases from CDH to CDP One

You must migrate Hue data from the CDH cluster to CDP One.

About this task

On your CDH environment enabled with Hue cluster migrate the Hue database:

Procedure

1. In the Hue Web UI, click the home icon to view the documents that you plan to migrate.
2. In Cloudera Manager, stop the Hue service. Go to Hue and select Actions > Stop.

   Note: Refresh the web page to confirm that the Hue service is stopped.
3. From Cloudera Manager -> Hue -> Select Actions > Dump Database and click Dump Database.

   This exports a JSON file that is database-agnostic. The file is written to

   /tmp/hue_database_dump.json

   on the host running the Hue service.

4. Log on to the host of the Hue server in a command-line terminal. You can find the hostname on the Dump Database window and at Hue > Hosts.
5. Remove the following lines from the hue_database_dump.json file.

   ALERT: This appears to be a CM Managed environment

   ALERT: HUE_CONF_DIR must be set when running hue commands in CM Managed environment

   ALERT: Please run 'hue <command> --cm-managed'

   On your CDP environment:

   a. In Cloudera Manager for the CDP environment that is, DataHub Cloudera Manager (not Data Lake Cloudera Manager), set up the database for Hue.
   b. Synchronize: Select Actions > Synchronize Database and click Synchronize Database.
   c. Truncate all the records in the Hue tables available under the Hue database.

      Note: Must be performed directly on the Hue database; there are no Hue commands or options in the UI to perform this task.

   Additionally, on the CDP environment: As no information is stored in the Hue database yet:

   a. Copy the hue_database_dump.json from the CDH cluster to the CDP cluster.
   b. Place the JSON file taken out from the legacy CDH Hue Cluster and place it in /tmp on the new CDP Hue server host.

      Note: The import process can only be run from one Hue server. Pick one to run the import process on and temporarily turn off the other Hue server instances.

      Attention: The file name should not be changed from hue_database_dump.json.You must load the Hue dump using the CLI method and must NOT use the UI option via Hue Server.
   c. Stop the Hue Service.
   d. Run the following commands:

      ```
      cd /opt/cloudera/parcels/CDH/lib/hue
      ./build/env/bin/hue loaddata --ignorenonexistent /tmp/<HUE_DATABASE_JSON_FILE> --cm-managed
      ```
   e. Start the Hue service.
   f. From Cloudera Manager -> Hue -> Select Actions -> Synchronize: Select Actions > Synchronize Database and click Synchronize Database.
   g. Start the Hue Service.

Performing post-migration tasks

You must complete the post-migration activities on your CDP cluster.

Note: The “userid” should NOT change while performing the post-migration task. If they are different, the documents do not populate for the user, as the documents are mapped with the specific userid.

Login to Hue using the same credentials used while using the source CDH Hue cluster.

The saved documents must be populated.

Attention: While carrying out the post-migration task, if the userid is different from the one that was used earlier, use the following method to resolve the differences.

Post Migration requirements

You must note the list of manual refactoring changes that should be applied to the Oozie jobs to get them working in the new environment.

- Name-node settings for each workflow
- Resource-manager settings for each workflow
- Paths, if the locations are being changed
- JDBC URLs, including Hive must point to the new database
- DFS paths may need to be replaced with S3
- New cluster-specific configurations for the Workflow (for example, Amazon S3 credentials.)
- Coordinator Jobs that have future sdf runs scheduled. Recreate these jobs to match what was included in any other cluster. For such Jobs, the Workflow and its configuration might be stored in Hue
- Copy any custom sharelib jars from the source to the destination cluster. The jars are stored here: /user/oozie/share/lib/lib_{TIMESTAMP}/{COMPONENT}
- If you have custom application jars used by any of your workflows, please make sure you recompile them with the 3rd-party libraries used by the latest Cloudera runtime.

For example on older versions of HDP, Oozie was using Guava 11, but in CDP 7 it is using Guava 28 or higher. If you compiled your application with Guava 11 you must also upgrade Guava.

- On the destination cluster, update the sharelib using the following command:

oozie admin -oozie {URL} -sharelibupdate

Migrating HDFS native permissions to CDP One

If you have HDFS native permissions in your CDH or HDP clusters, you learn how to convert the native permissions into Ranger policy format and import the policies to CDP One. You can choose to ignore this migration process if you do not have any HDFS native native permissions.

Migrating HDFS native permissions from CDH or HDP to CDP One involves a combination of Extract Convert Transform Import operations.

The HDFS native permissions are first extracted as a file in a specific format, which are then converted and transformed into Ranger AWS S3 policies using the Ranger policy migration utility. The transformed Ranger policies are finally imported into CDP One using the authzmigrator tool.

Extracting HDFS native permissions

Learn how to use the HDFS Permissions Export utility to extract HDFS native POSIX permissions from a source cluster running CDH or HDP. The extracted HDFS permissions are then used to create Ranger S3 policies that can be used in CDP One.

About this task

The HDFS Permissions Export utility exports the permissions to a .csv file in the following format with the permissions sorted in ascending order:

```
Syntax:
"/resource/path"|"username"|"groupname"|"permission"
```

where,

- "/resource/path" refers to resource entities
- "username" refers to user entries
- "groupname" refers to group entries
- "permission" refers to the HDFS permission entries

```
Example:
"/dir1"|"hdfs"|"supergroup, public"|"read, execute"
"/dir1/dir1"||"supergroup, public"|"read, execute"
"/dir1/dir1"|"hdfs"||"read, write, execute"
```

Before you begin

- The source and target migration clusters must be running the following supported versions:
  - Source cluster: HDP 2.6.5, HDP 3.1.5, CDH 5.16, CDH 6.3
  - Target cluster: CDP 7.2.15 or higher
- Secure Copy (SCP) the HDFS keytab file from the source cluster to your local system
- Run the klist command to view the list of Kerberos principals available in the keytab — ${RANGER_ADMIN_KEYTAB_PATH} and then run the kinit command to authenticate the user.

```
klist -kt ${RANGER_ADMIN_KEYTAB_PATH}

kinit -kt ${RANGER_ADMIN_KEYTAB_PATH} rangeradmin/_HOST@REALM
```

Procedure

1. Download the HDFSPermissionsExportTool-<version>.jar file from Cloudera Archive.
2. Run the following command to extract HDFS permissions from the source cluster.

   ```
   hadoop jar ./<path>/HDFSPermissionsExportTool-<version>.jar hdfs_namenod
   e_host:hdfs_port source_directories
   ```

   where,

   - <path> refers to the location of the downloaded .jar file.
   - hdfs_namenode_host:hdfs_port refers to the URI of the active NameNode and the HDFS NameNode port. For example, active-namenode.example.com:8020
   - source_directories refers to the root directory from where you want the permissions scan to begin. For example, "/" or "/dir1". You can also specify multiple directories — "/hbase/yarn"

   ```
   hadoop jar ./target/HDFSPermissionsExportTool-<version>.jar active-namen
   ode.example.com:8020 /user /ranger
           /hbase
   ```

   This creates a HDFS_Permissions_Export_<timestamp>.csv file that contains the required HDFS permissions.

   ⚠ **Important:** Be sure to use the URI of the active NameNode. Using the passive NameNode URI or the NameNode HA URI results in a failure.
3. Use the scp command and copy HDFS_Permissions_Export_<timestamp>.csv from the source cluster to the migration node of the target cluster.

   ⚠ **Important:** Since the .csv filename contains a colon (:), ensure that you provide the full path in the scp command to prevent the command from treating ":" as a separator.

What to do next

Convert the extracted HDFS native permissions into Ranger HDFS policies and then transform them into Ranger S3 policies.

Related Information

Converting HDFS native permissions into Ranger HDFS policies

Converting HDFS native permissions into Ranger HDFS policies

Learn how to convert the extracted HDFS native permissions into Ranger HDFS service policies, which can then be transformed into Ranger S3 policies.

Before you begin

- The source and target migration clusters must be running the following supported versions:
  - Source cluster: HDP 2.6.5, HDP 3.1.5, CDH 5.16, CDH 6.3
  - Target cluster: CDP 7.2.15 or higher
- You must have copied the extracted HDFS native permissions (.csv file) into the migration cluster node.
- You must have copied the Ranger policy migration utility, ranger-<version>-policymigration.tar.gz file to the migration cluster node.
  1. Log in to the migration cluster's node (or a Data Lake node of a cloud cluster) where the ranger-admin binaries are available. For example, /opt/cloudera/parcels/CDH/lib/ranger-admin
  2. Copy /opt/cloudera/parcels/CDH/lib/ranger-admin/policymigration/ranger-<version>-policymigration.tar.gz to some location of the migration cluster machine from where you want to run the Ranger policy migration utility.
- You must have read, write, and execute permissions on the directory where the Ranger policy migration utility is copied.
- You must have SUDO or ROOT access to perform the migration process.

Procedure

1. Log in to the migration cluster node where you have copied the ranger-<version>-policymigration.tar.gz file. Untar the file and navigate to the ranger-<version>-policymigration directory.
2. Update the “Transform” part of the env.sh file with the right set of values according to your environment.

   For more information, see Supported Input parameters for Transform operation.
3. Run the transform.sh script with the convert command to convert the extracted HDFS native permissions to Ranger HDFS policy format.

   ```
   Syntax:
   ./transform.sh convert $native_hdfs_permissions_csv_file_path
   ```

   ```
   Example:
   ./transform.sh convert /tmp/HDFS_Permissions_Export_<timestamp>.csv
   ```

Results

The command generates a HDFS_Permissions_Export_<timestamp>_convert.json file, which is then transformed into the required Ranger S3 policies. If you encounter any errors, see the logs/ranger-policymigration.log file.

What to do next

Transform the converted Ranger HDFS policies into Ranger S3 policies.

Related Information

Supported Input parameters for Transform operation

Extracting HDFS native permissions

Transforming Ranger HDFS policies into Ranger S3 policies

Transforming Ranger HDFS policies into Ranger S3 policies

Learn how to transform Ranger HDFS service policies into Ranger AWS S3 policies, which can then be imported into the CDP cluster Ranger.

Before you begin

- You must have converted the extracted Ranger HDFS native permissions (.csv file) into Ranger HDFS policies (.json file) using the transform.sh convert command.

- You must have copied the Ranger policy migration utility, ranger-<version>-policymigration.tar.gz file to the migration cluster node.
    1. **1.** Log in to the migration cluster's node (or a Data Lake node of a cloud cluster) where the ranger-admin binaries are available. For example, /opt/cloudera/parcels/CDH/lib/ranger-admin
    2. **2.** Copy /opt/cloudera/parcels/CDH/lib/ranger-admin/policymigration/ranger-<version>-policymigration.tar.gz to some location of the migration cluster machine from where you want to run the Ranger policy migration utility.
- You must have read, write, and execute permissions on the directory where the Ranger policy migration utility is copied.
- You must have SUDO or ROOT access to perform the migration process.

Procedure

1. Log in to the migration cluster node where you have copied the ranger-<version>-policymigration.tar.gz file. Untar the file and navigate to ranger-<version>-policymigration directory.

2. Update the "Transform" part of the env.sh file by updating the following parameters or as required for your environment.

- S3_BUCKET_NAME
- SERVICE_NAME_FOR_NATIVE_POLICIES

> **Note:** If the usernames in the target cluster are different from the usernames in the source cluster, provide a user mapping file as an input by updating the RANGER_POLICYMIGRATION_USERS_MAPPING_FILE parameter in the env.sh file.

For more information, see Supported Input parameters for Transform operation.

3. Run the transform.sh script with the transform command to convert the Ranger HDFS policies to Ranger AWS S3 policy format.

```
Syntax:
./transform.sh transform $converted_ranger_policies_file_path
```

```
Example:
./transform.sh transform /tmp/HDFS_Permissions_Export_<timestamp>_conve
rt.json
```

Results

The command generates a HDFS_Permissions_Export_<timestamp>_convert_transform.json file, which can then be imported using the Authzmigrator utility into the AWS S3 service of the target CDP cluster. If you encounter any errors, see the logs/ranger-policymigration.log file.

What to do next

Import the transformed HDFS native permissions into the Ranger AWS S3 service of the target CDP cluster using the Authzmigrator tool.

Related Information

Supported Input parameters for Transform operation

Extracting HDFS native permissions

Converting HDFS native permissions into Ranger HDFS policies

Importing Ranger AWS S3 policies

Learn how to import the transformed HDFS native permissions into the Ranger AWS S3 service of the target CDP cluster using the Authzmigrator tool.

Before you begin

- You must have extracted, converted, and transformed the native HDFS permissions into Ranger AWS S3 policy format.
- Ensure that the RANGER_ADMIN role is up and running in the Cloudera Manager/CDP One portal of the target environment.

Procedure

1. Log in to the RANGER_ADMIN portal with administrator user credentials or user privileges as ADMIN.
   a) From Settings User/Groups/Roles Search , search for the "ranger" user.
   b) Click the "ranger" user and change its role to "Admin", and save the changes.
2. Download and copy the authz-ingest.zip file to the /opt directory of the migration cluster node (or a Data Lake node of the cloud cluster).
3. Extract the authz-ingest.zip file into the /opt directory.

   Note: /opt is referred to as an example directory.

   The authz-ingest file contains the following directories:

   - config-files
   - scripts
4. If Ranger is TLS/SSL enabled using Cloudera Manager AUTO-TLS, then copy the cm-auto-global-truststore.jks file from the Ranger admin process directory of the Ranger admin host to /opt/authz-ingest/config-files/ of the migration tool host. Perform these steps to find the location of the Ranger admin process directory:
   a) Run the following command to get to the Ranger admin process:

      ps -ef | grep proc_rangeradmin
   b) Search for the -cp option from the output of the above command and look for "/var/run/cloudera-scm-agent/process/<ID>-ranger-RANGER_ADMIN"

   If you have set up SSL manually or using a different method, then copy the file that you have created for trust stores to the /opt/authz-ingest/config-files directory.

   For example, cp /path/to/cm-auto-global_truststore.jks /opt/authz-ingest/config-files/
5. Export the following environment variables to the target cluster Ranger:

   - JAVA_HOME - Java home path used on the cluster
   - RANGER_ADMIN_URL - URL of the Ranger Admin portal
   - RANGER_ADMIN_SSL_ENABLED - Indicates if SSL is enabled or disabled
   - RANGER_ADMIN_TRUSTORE_PASSWORD - If Ranger is TLS/SSL enabled using Cloudera Manager AUTO-TLS, then the password can be found in the Ranger host file, /etc/hadoop/conf/ssl-client.xml, property — 'ssl.client.truststore.password'

   ```
   export JAVA_HOME=/usr/java/jdk1.8.0_232-cloudera/
   export RANGER_ADMIN_URL=<ranger admin url with port>
   export RANGER_ADMIN_SSL_ENABLED=<true or false>
   export RANGER_ADMIN_TRUSTORE_PASSWORD=<ssl.client.truststore.password>
   ```
6. Go to /opt/authz-ingest/config-files and open the authorization-migration-site.xml file.

   The authorization-migration-site.xml contains the following properties:

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <configuration>
     <property>
       <name>authorization.migration.export.target_services</name>
       <value>HIVE,KAFKA</value>
     </property>
   ```

```
  <property>
    <name>authorization.migration.export.migration_objects</name>
    <value />
  </property>
  <property>
    <name>authorization.migration.export.output_file</name>
    <value>hdfs:///user/sentry/export-permissions/permissions.json</value>
  </property>
  <property>
    <name>authorization.migration.ingest.is_dry_run</name>
    <value>false</value>
  </property>
  <property>
    <name>authorization.migration.role.permissions</name>
    <value>true</value>
  </property>
  <property>
    <name>authorization.migration.translate.url.privileges</name>
    <value>false</value>
  </property>
  <property>
    <name>authorization.migration.ingest.merge.ifexists</name>
    <value>true</value>
  </property>
  <property>
    <name>authorization.migration.migrate.url.privileges</name>
    <value>true</value>
  </property>
</configuration>
```

7. Search for the authorization.migration.ranger.import.input_file property in the authorization-migration-site.xml file and provide the location of the transformed HDFS permissions file as the property value.

```
<property>
 <name>authorization.migration.ranger.import.input_file</name>
<value>file:///tmp/HDFS_Permissions_Export_<timestamp>_convert_transform.j
son</value>
</property>
```

If you are unable to find this property, then add the property in the file.

8. For Kerberos-based authentication, perform the following steps:

Important: If the Key Distribution Center (KDC) server is not installed in your environment or imported without the Kerberos authentication, you can skip this procedure and use the process provided for non-Kerberos authentication. Also, if PAM authentication is enabled in the target cluster, then Kerberos authentication is recommended to import ranger policies quickly.

a) Find the Ranger admin keytab location and provide it in the following command:
export RANGER_ADMIN_KEYTAB_PATH=/path/to/ranger.keytab

In the CDEP cluster, ranger keytab is available in the ranger host location /cdep/keytabs/rangeradmin.keytab or under the ranger admin process directory — /var/run/cloudera-scm-agent/process/<ranger-process-ID>-ranger-RANGER_ADMIN

b) If your ranger node and migration node are different, copy the ranger keytab to the migration node.

c) Run the klist command on the migration node to view the list of Kerberos principals available in the keytab — ${RANGER_ADMIN_KEYTAB_PATH}
klist -kt ${RANGER_ADMIN_KEYTAB_PATH}

d) Run the following kinit command and update _HOST and REALM correctly.
kinit -kt ${RANGER_ADMIN_KEYTAB_PATH} rangeradmin/_HOST@REALM

9. For non-Kerberos authentication, add the following properties in the /opt/authz-ingest/config-files/authorization-migration-site.xml file with the appropriate values, and then save and close the file.

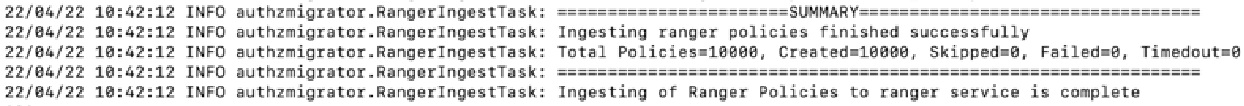
```
<property>
    <name>ranger.admin.username</name>
    <value>admin</value>
</property>

<property>
    <name>ranger.admin.passwd</name>
    <value>admin123</value>
</property>
<property>
<name>authorization.migration.kerberos.authentication</name>
<value>false</value>
</property>
```

10. Go to the /opt/authz-ingest/scripts directory and run the import script.
sh authz-import.sh

If the import operation fails, look for details in the output of the command printed in the terminal window or see the target Ranger admin log files.

Results

After the import operation is complete, the authzmigrator tool displays the total policies count, created, skipped, and failed policies count. You can refer to the console output logs to verify if policies are imported or not.



```
22/04/22 10:42:12 INFO authzmigrator.RangerIngestTask: =======================SUMMARY===================================
22/04/22 10:42:12 INFO authzmigrator.RangerIngestTask: Ingesting ranger policies finished successfully
22/04/22 10:42:12 INFO authzmigrator.RangerIngestTask: Total Policies=10000, Created=10000, Skipped=0, Failed=0, Timedout=0
22/04/22 10:42:12 INFO authzmigrator.RangerIngestTask: ===============================================================
22/04/22 10:42:12 INFO authzmigrator.RangerIngestTask: Ingesting of Ranger Policies to ranger service is complete
```

Related Information

Extracting HDFS native permissions

Converting HDFS native permissions into Ranger HDFS policies

Transforming Ranger HDFS policies into Ranger S3 policies

Migrating workflows directly created in Oozie to CDP One

The oozie workflows present on HDFS must be migrated to CDP One.

Oozie workflows created manually or outside of the Hue Workflow editor require specific review and manual updates. No automated tool or a process is required for updating manually created workflows and property files.

The Oozie workflows are available in HDFS within CDH and HDP clusters. To migrate the Oozie workflows from these clusters to CDP SaaS involves a separate workflow process.

While using the HDP clusters, you can use the DistCp tool to migrate the Oozie workflows present in HDFS. While using the CDH clusters, you can employ the Replication Manager App to migrate the Oozie workflows present in HDFS

Specifically, some manual updates are required to process the Oozie migration created outside of the Hue Workflow. Before you proceed further, you must understand:

- During the migration process you must copy across all Oozie job files (workflow.xml, job.properties, and any supporting JARs).
- Which Oozie workflow files must be copied or migrated to CDP One. Identify the workflow.xml file and job.properties file for each Oozie workload that must be migrated. These files are stored in HDFS and must be copied to the CDP One endpoint.
- The job.properties file must be updated with the appropriate CDP One endpoints.

- Optionally, the workflow.xml file needs to be updated. For example, while currently using the legacy “hive action” requires an update to the newer “hive2 action”.

Depending on where (the location) you have stored your Oozie workflow data in HDFS, note the following information:

The workflow.xml and any job JAR files reside within HDFS in the source cluster. These will have to be copied across into a CDP One endpoint.

The job.properties file for a job contains a reference to the location of where the workflow files are stored. The job .properties file will need to be updated during a migration with the new target environment settings / locations.

Note: The job.properties file must be run from "local" and not HDFS, but some customers may have these files also stored on HDFS in the source cluster.

Attention: Using the Oozie Hive Action with HiveServer2 is deprecated. If using the legacy HivAction in your Oozie workflow, you must modify your workflow.xml to use the Hive 2 Action instead.

- If required, the refactoring of the component level code. For example, Oozie job executing Spark 1.6 code requires an update to the newer Spark 2.4 version. For more information, see Migrating Spark workloads to CDP.

Related Information

Migrating HDFS and Hive data from CDH to CDP One

Migrating HDFS data from HDP to CDP One

Migrating Sentry policies from CDH to CDP One

In the CDH environment consisting of Sentry policies, permissions need to be migrated from Sentry to Ranger. This migration process is supported by the Authzmigrator tool.

About Migrating Sentry policies

Migrating Sentry policies to CDP One is a two step process.

Export the Sentry policies from the CDH cluster and copy the generated output file(s) to the target environment. Later, the import utility (Authzmigrator tool) performs the Transform and Import operations to complete the Sentry migration process. Export operation is performed using the CDH cluster.

The following policy migrations are included as part of the migration process:

- Kafka and Hive Sentry permissions
- URL transformation to the AWS S3 endpoint
- Default AWS S3 policy creation for Hive Warehouse location

Migration prerequisites

You need to know the prerequisites for migrating Sentry policies from CDH to CDP One.

You must meet the following general prerequisites before starting the migration process:

- AWS is your Cloud provider. Migrating to Azure / GCP is currently not supported.
- The source and target migration clusters must be running the following supported versions:
  - Source cluster: CDH 5.16, CDH 6.3
  - Target cluster: CDP 7.2.15 or higher
- You can log in as the admin user to Cloudera Manager in the CDH cluster.
- You must have SUDO or ROOT access to perform the migration process.
- CDH clusters must have been created using Cloudera Manager. Clusters that are not managed by Cloudera Manager cannot be registered to CDP.
- Create an external account on the CDH cluster that has non-expiring access credentials and secret key pair.

- Your source cluster supports compatible versions of Cloudera Manager and Cloudera Runtime shown in the following table:

Source cluster	Earliest supported Cloudera Manager	Earliest supported Cloudera Runtime
CDH 5	6.3.0	5.1.0
CDH 6	6.3.0	6.1.0
CDH 6	7.3.1	6.3.3

Specifically note that:

- Your CDH cluster is registered on the CDP Management Console. For more information, see Adding a CDH cluster.
- Open communication ports between the source CDH cluster and CDP list.
- When you create databases, you must assign valid roles for tables and grant adequate permissions as well. And the same applies to Kafka.

Setting up an external account

As CDH Administrator, you need to create an External Account on the CDH cluster that has an access key and secret key pair for accessing the cloud storage used in the CDP One environment.

About this task

Using the external account along with the cloud storage path, administrators can migrate the Sentry policies from the CDH cluster to the cloud storage used in the CDP One environment.You must have valid Amazon S3 credentials to access cloud storage for migrating data.

You must raise a support case with Cloudera to provide the key pair.

You must have valid Amazon S3 credentials to access cloud storage for migrating data. You add the access key / secret key pair to the account to access CDP cloud storage s3a://<data lake bucket path>. The access key / secret key pair ensures that the external account configured on CDH can successfully access the S3 buckets used by Data Lake.

Procedure

1. In Cloudera Manager#Administration, click Add Access Key Credentials.
2. In Edit Credential, name the credential and enter the AWS Access Key ID and AWS Secret Key.
3. Save changes,

Exporting Sentry permissions

You must export the Sentry policies from CDH cluster to CDP cluster Ranger.

Procedure

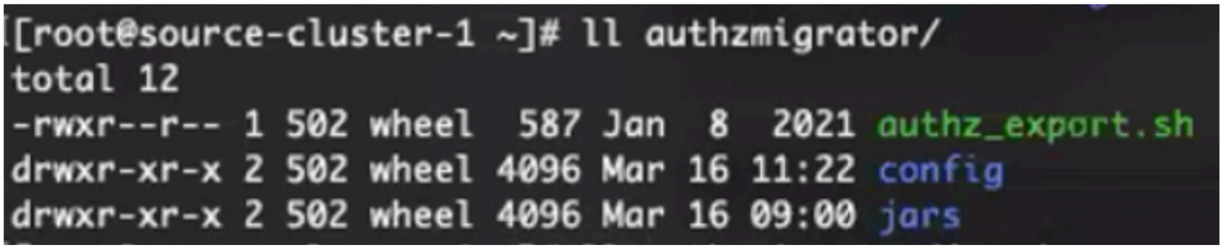
1. SSH to the Sentry host of the source cluster.

2. Copy the file located here to the Sentry host and extract it to a suitable location.

   tar -xvf authz_export.tar.gz

   Later, a directory named authzmigrator is created which contains the following files:

   - Jars
   - Config
   - Authz_export.sh

   ```
   [root@source-cluster-1 ~]# ll authzmigrator/
   total 12
   -rwxr--r-- 1 502 wheel  587 Jan  8  2021 authz_export.sh
   drwxr-xr-x 2 502 wheel 4096 Mar 16 11:22 config
   drwxr-xr-x 2 502 wheel 4096 Mar 16 09:00 jars
   ```

3. Verify the SENTRY_SERVER process path

   ps -ef | grep SENTRY_SERVER
4. Replace sentry-site.xml and core-site.xml in the extracted file with config files from Sentry-run directory.
   For example,

   - cp /var/run/cloudera-scm-agent/process/<process-id>-sentry-SENTRY_SERVER/sentry-site.xml authzmigrator/config/
   - cp /var/run/cloudera-scm-agent/process/<process-id>-sentry-SENTRY_SERVER/core-site.xml authzmigrator/config/
5. Update ‘Sentry.store.jdbc.user’ and sentry.store.jdbc.password in sentry-site.xml.

   The mandatory values are the Sentry "database user" and "database user password" in clear text.
6. Remove the property hadoop.security.credential.provider.path in sentry-site.xml.
7. Update the value for the property fs.defaultFS to file:/// in core-site.xml.
8. Make sure that the following configurations are updated in

   authorization-migration-site.xml present in the authzmigrator/config/

   - • authorization.migration.export.target_services= HIVE,KAFKA should have a list of services for which permissions are needed to export the Sentry permissions. Valid values: HIVE, KAFKA

- authorization.migration.export.output_file=<path> should be updated to the absolute location of the file where permissions should be exported

To set up role based permissions, you must add the following properties:

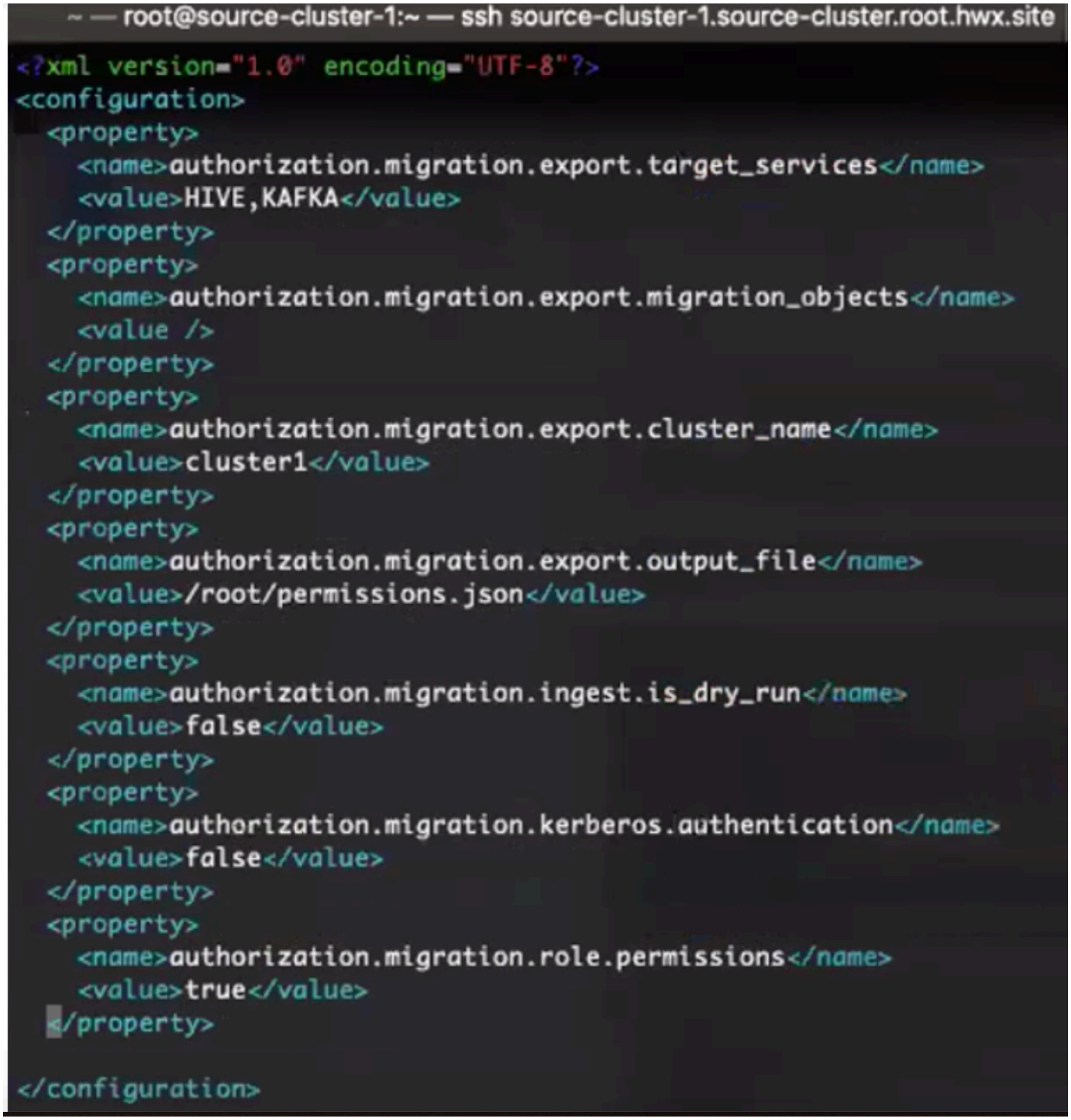
<property>

<name>authorization.migration.role.permissions</name>

<value>true</value>

</property>

An example property file:



~ — root@source-cluster-1:~ — ssh source-cluster-1.source-cluster.root.hwx.site

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
  <property>
    <name>authorization.migration.export.target_services</name>
    <value>HIVE,KAFKA</value>
  </property>
  <property>
    <name>authorization.migration.export.migration_objects</name>
    <value />
  </property>
  <property>
    <name>authorization.migration.export.cluster_name</name>
    <value>cluster1</value>
  </property>
  <property>
    <name>authorization.migration.export.output_file</name>
    <value>/root/permissions.json</value>
  </property>
  <property>
    <name>authorization.migration.ingest.is_dry_run</name>
    <value>false</value>
  </property>
  <property>
    <name>authorization.migration.kerberos.authentication</name>
    <value>false</value>
  </property>
  <property>
    <name>authorization.migration.role.permissions</name>
    <value>true</value>
  </property>

</configuration>
```

9. Export JAVA_HOME variable

10. Run the script:

    ```
    cd authzmigrator/
    ```

    ```
    sh      authz_export.sh
    ```

    Exporting the permissions is completed.
11. SCP the exported JSON to the target cluster. For more information about logging into CDP One clusters, see Using SSH to access the cluster.

    ```
    scp <exported json> root@<target_clustert>:/root
    ```

Related Information

Importing Sentry permissions into Ranger

Importing Sentry permissions into Ranger

You must import the Sentry permissions to CDP cluster Ranger.

Procedure

1. Login to the RANGER_ADMIN portal with administrator user credentials or user privileges as ADMIN
    a. Under Settings > Users/Groups/Roles > Search for the ranger user.
    b. Click on the ranger user and change its role to Admin and save the changes.
2. You must log into the CDP One environment. For more information about logging into CDP One clusters, see Using SSH to access the cluster.
3. Download and copy authz-ingest.zip to the /opt directory of the migration node.

    For example: CDP Data Lake or any relevant target cluster.
4. Unzip the authz-ingest.zip into the /opt directory, which contains the following files:

    Note: /opt is referred to as an example directory.

    - config-files
    - scripts

    The config-files contains the following:

    authorization-migration-site.xml

    ranger-hive-security.xml

    ranger-kafka-security.xml

    ranger-plugin-policymgr-ssl.xml

    Note: cm-auto-global_truststore.jks is directly not available. If you have an SSL/TLS enabled cluster, you must manually add cm-auto-global_truststore.jks.

5. Ranger is TLS/SSL enabled using Cloudera Manager AUTO-TLS. From the Ranger admin host, copy cm-auto-global_truststore.jks (from the Ranger-admin process directory) to the following location:

   /opt/authz-ingest/config-files/ of the migration tool host.

   Steps to find the ranger admin process directory:

   - • Execute the below command to get the RANGER_ADMIN process.

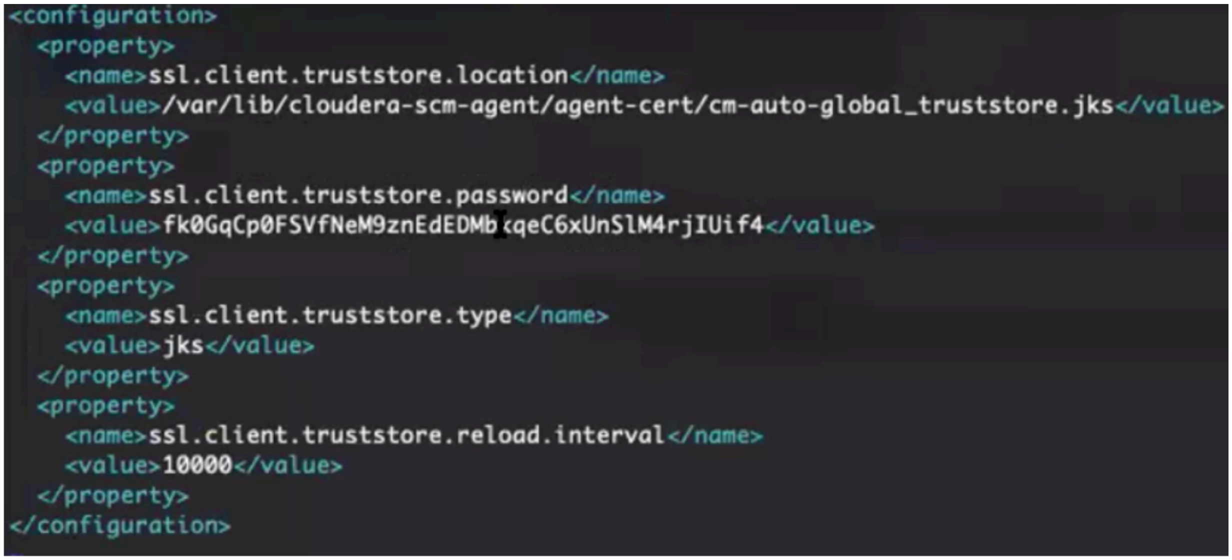
     ```
     ps -ef | grep      proc_rangeradmin
     ```
   - Search for -cp option from the above command output and look for "/var/run/cloudera-scm-agent/process/<ID>-ranger-RANGER_ADMIN"

   If you have done SSL setup manually or in a different method, copy the filename which you created for truststores.

   For example: cp /path/to/cm-auto-global_truststore.jks   /opt/authz-ingest/config-files/

6. Export the following variables:

   - JAVA_HOME - Java home path used on the cluster
   - RANGER_ADMIN_URL - Ranger Admin portal url
   - RANGER_ADMIN_SSL_ENABLED - either true or false
   - RANGER_ADMIN_TRUSTORE_PASSWORD - If Ranger is TLS/SSL enabled using Cloudera Manager AUTO-TLS, then the password can be found on the Ranger host /etc/hadoop/conf/ssl-client.xml property 'ssl.client.truststore.password'



```
<configuration>
  <property>
    <name>ssl.client.truststore.location</name>
    <value>/var/lib/cloudera-scm-agent/agent-cert/cm-auto-global_truststore.jks</value>
  </property>
  <property>
    <name>ssl.client.truststore.password</name>
    <value>fk0GqCp0FSVfNeM9znEdEDMbxqeC6xUnSlM4rjIUif4</value>
  </property>
  <property>
    <name>ssl.client.truststore.type</name>
    <value>jks</value>
  </property>
  <property>
    <name>ssl.client.truststore.reload.interval</name>
    <value>10000</value>
  </property>
</configuration>
```

EXAMPLE:

```
export JAVA_HOME=/usr/java/jdk1.8.0_232-cloudera/
export RANGER_ADMIN_URL=<ranger admin url with      port>
export RANGER_ADMIN_SSL_ENABLED=<true or      false>
export RANGER_ADMIN_TRUSTORE_PASSWORD=<ssl.client.truststore.password>
```

7. For the Kerberos-based authentication, use the following process:

a. Find the Ranger admin keytab location and provide it in the following command:

```
export          RANGER_ADMIN_KEYTAB_PATH=/path/to/rangeradmin.keytab
```

In the CDEP cluster ranger keytab shall be available in the "/cdep/keytabs/rangeradmin.keytab" or under the ranger admin process directory "/var/run/cloudera-scm-agent/process/<ranger-process-number>-ranger-RANGER_ADMIN" location.

Steps to find the ranger admin process directory:

- Execute the below command to get the RANGER_ADMIN process.

```
ps -ef | grep proc_rangeradmin
```

- Search for -cp option from the above command output and look for "/var/run/cloudera-scm-agent/process/<ID>-ranger-RANGER_ADMIN"

If your ranger node and migration node are different, copy the ranger keytab to the migration node.

On the migration node run the following klist command to fetch a list of principals available in

${RANGER_ADMIN_KEYTAB_PATH}

```
klist -kt        ${RANGER_ADMIN_KEYTAB_PATH}
```

Run the following kinitcommand using

${RANGER_ADMIN_KEYTAB_PATH} with rangeradmin/_HOST@REALM principle.

Update _HOST and REALM using the following command:

```
kinit -kt ${RANGER_ADMIN_KEYTAB_PATH}          rangeradmin/_HOST@REALM
```

8. Go to location /opt/authz-ingest/config-files and open the file authorization-migration-site.xml

9. Add below property and provide the path of the sentry permissions.json file which needs to be imported.

```
<property>
    <name>authorization.migration.export.output_file</name>
<value>file:///root/permissions.json</value>
</property>
```

Note: If you have permission which is role-based, you have to set this property

```
<property>
    <name>authorization.migration.role.permissions</name>
<value>true</value>
</property>
```

10. Set the following properties to create AWS S3 policy for Hive Warehouse location:

```
<property>
    <name>authorization.migration.create.s3.policy</name>
<value>true</value>
</property>
<property>
    <name>authorization.migration.target.s3.bucket.name</name>
<value>bucket_name</value>
</property>
```

11. Set the following properties which ensures that URI/URL policies are appropriately converted into AWS S3 URL Hive policy:

```
<property>
<name>authorization.migration.translate.url.privileges</name>
<value>true</value>
</property>
<property>
<name>authorization.migration.migrate.url.privileges</name>
<value>true</value>
</property>
<property>
<name>authorization.migration.destination.location.prefix</name>
<value>s3a://mybucket</value>
</property>
```

12. (Optional) Transforming old user name into new user name requires you to set the following properties:

```
<property>
<name>authorization.migration.translate.user</name>
<value>true</value>
</property>
<property>
<name>authorization.migration.usermap.file.path</name>
<value>file:///root/UserMapping.json</value>
</property>
```

Use the following information to provide the valid user mapping:

Format:

=> For the JSON file format we can have multiple users also in the JSON object formatted like :

{“testuser1”:”testuser1@host.com”, “testuser2”:”testuser2@host.com”}

=> And with the CSV format, it's pipe-delimited with one entry per line.

testuser1,testuser1@host.com

testuser2,testuser2@host.com

Note: If user mapping is provided in CSV format, set the following property

```
<property>
<name>authorization.migration.usermap.separator</name>
<value>,</value>
</property>
```

13. A flag-based new functionality is added in Authzmigrator, which skips {OWNER} policy while migrating from Sentry to Ranger. You must add the below property in authorization-migration-site.xml to enable this feature.

By default, this feature is disabled by setting the below value to false.

```
<property>
```

```
    <name>authorization.migration.skip.owner.policy</name>
    <value>true</value>
    </property>
```

14. Save the file and exit
15. Run the script:

    cd authz-ingest/scripts

    sh authz-import.sh

    A confirmation message appears.

    "Ingesting Ranger policies finished successfully"

    "Ingesting of permissions to Ranger service is complete"

Migrating data from HDP to CDP One

How to migrate data from HDP to CDP One.

Migrating HDFS data from HDP to CDP One

An overview of the migration process from HDP to CDP One prepares you to migrate HDFS data to the AWS S3 endpoint.

The HDP to CDP One migration migrates HDFS data using the CDP Replication Manager service. Once fulfilling migration prerequisites, you set up security for communications between the HDP cluster and the destination cloud services.

Migration prerequisites

You need to know the prerequisites for migrating HDFS data from HDP to CDP One.

You must meet the following general prerequisites before starting the migration process:

- You can log in as the “hdfs” user to Ambari in the HDP cluster.
- Set up line-of-sight from the HDP cluster to the AWS endpoint.
- Ensure adequate network connectivity between the HDP cluster and AWS S3 endpoint.
- You can add cloud credentials and configure them in one of the two ways:
  - Add them into the core-site.xml configuration.
  - Provide as a runtime parameter when running the job (that job can be DistCp, Spark job and others).
- Make sure the HDFS service ports are configured.
- As the migration of HDFS data to CDP SaaS is directly to S3 via a push mechanism, the key network requirement is to ensure that the YARN / HDFS nodes have network access to the AWS S3 endpoints (port 80 or port 443 for TLS).
- Must allocate compute resources to execute YARN (DistCp) jobs to transfer the HDFS data to the S3 bucket.
- Any user running the DistCp job should not be a banned user and must be a part of a supergroup on each host in the cluster.
- Since there is no superuser on each host, the administrator needs to unban ‘hdfs’ user in the HDP cluster. For more information, see Unbanning hdfs user in HDP cluster on page 41.

Once fulfilling migration prerequisites, you set up security for communications between the HDP cluster and the destination cloud services.

Hadoop DistCp tool is used to migrate the HDFS data (and metadata) from an on-prem HDP cluster to the cloud storage (AWS S3) used in the CDP One environment.

About DistCp tool

DistCp (distributed copy) is a tool used for large inter/intra-cluster copying.

DistCp uses MapReduce to enable its distribution, error handling and recovery, and reporting. It expands a list of files and directories into input to map tasks, each of which will copy a partition of the files specified in the source list.

Using the DistCp tool

Use DistCp to copy files between various clusters.

The most common use of DistCp is an inter-cluster copy:

hadoop distcp hdfs://nn1:8020/source hdfs://nn2:8020/destination

Where hdfs://nn1:8020/source is the data source, and hdfs://nn2:8020/ destination is the destination.

This will expand the name space under /source on NameNode "nn1" into a temporary file, partition its contents among a set of map tasks, and start copying from "nn1" to "nn2". Note that DistCp requires absolute paths.

You can also specify multiple source directories:

```
hadoop distcp hdfs://nn1:8020/source/a hdfs://nn1:8020/source/b hdfs://
         nn2:8020/destination
```

Or specify multiple source directories from a file with the -f option:

```

hadoop distcp -f hdfs://nn1:8020/srclist hdfs://nn2:8020/destination
```

Where srclist contains:

hdfs://nn1:8020/source/a

hdfs://nn1:8020/source/b

Note: The DistCp tool is also used to migrate data to the cloud storage (AWS S3, Microsoft Azure, and Google Cloud).

Unbanning hdfs user in HDP cluster

The hdfs user must be unbanned and allowed full access to the HDP cluster.

About this task

Procedure to unban the hdfs user.

Procedure

1. Remove the hdfs entry from the following file, from line banned.users on the ambari-server host.

   ```
   /var/lib/ambari-server/resources/common-services/YARN/2.1.0.2.0/package/
   templates/container-executor.cfg.j2
   ```

   The updated file contents are as follows

   yarn.nodemanager.local-dirs={{nm_local_dirs}}

   yarn.nodemanager.log-dirs={{nm_log_dirs}}

   yarn.nodemanager.linux-container-executor.group={{yarn_executor_container_group}}

   banned.users=yarn,mapred,bin

   min.user.id={{min_user_id}}

2. Check the YARN service configuration on Ambari UI for ‘container-executor configuration template’ configuration. Ensure that the hdfs user is not mentioned in the banned.users on the ambari-server host. If it is still present, remove it from the configuration, save the changes and restart stale services.
3. Restart ambari-server. On the ambari-server host, execute the following command:

   service ambari-server restart
4. Restart ambari-agent. On each ambari-agent host of the cluster, execute the following command:

   service ambari-agent restart
5. Add hdfs user in YARN ACLs configuration by editing

   yarn.admin.acl
6. Allow hdfs user to submit YARN jobs by editing the following configuration

   yarn.scheduler.capacity.root.acl_submit_applications
7. Restart the YARN service in the Ambari UI.

Before migrating

Before you commence to migrate HDFS data from HDP clusters, you must note about some of the underlying requirements.

To run the Hadoop DistCp tool to copy the HDFS data from the source HDP cluster to the CDP One S3 bucket, the following additional details must be met:

- The access key, secret key pair of the specified S3 bucket must be available.
  - Before you run the DistCp tool on your on-premises cluster, you must have valid cloud credentials to access and use AWS. Create a Support JIRA that states the time period for the migration.
  - The AWS access secret key will be disabled when the migration time expires.
- SSM-KMS is enabled on the S3 bucket, the KMS key is also required.
- Perform the appropriate kinit to the Hadoop DistCp tool for authentication by the Kerberos system.
  - For hdfs user, use the kinit command with hdfs user’s keytab file. (Look for keytab in /etc/security/keytabs)

HDFS data migration from HDP to CDP One

Use the DistCp tool to migrate the data.

An example DistCp command:

```
$ sudo -u hdfs hadoop distcp -Dfs.s3a.access.key=###########
          -Dfs.s3a.secret.key=######### -Dfs.s3a.server-side-encryption-alg
orithm=SSE-KMS
          -Dfs.s3a.server-side-encryption.key=arn:aws:kms:<region>:######
##### /data/datafiles/
          s3a://demo2-cdp-private-default-saas/cdp-saas/hdp-data/
```

Additional configuration:

The following AWS S3 specific configurations are recommended, apart from configuring the default settings like the numbers of mappers and bandwidth per mapper.

1. fs.s3a.fast.upload=true, this configuration allows to speed up the data transfer
2. Setting up s3 proxy server configurationsfs.s3a.proxy.* configs
3. fs.s3a.connection.ssl.enabled=true, this configuration enables SSL connection to the S3 endpoint.

Migrating HDFS native permissions to CDP One

If you have HDFS native permissions in your CDH or HDP clusters, you learn how to convert the native permissions into Ranger policy format and import the policies to CDP One. You can choose to ignore this migration process if you do not have any HDFS native native permissions.

Migrating HDFS native permissions from CDH or HDP to CDP One involves a combination of Extract Convert Transform Import operations.

The HDFS native permissions are first extracted as a file in a specific format, which are then converted and transformed into Ranger AWS S3 policies using the Ranger policy migration utility. The transformed Ranger policies are finally imported into CDP One using the authzmigrator tool.

Extracting HDFS native permissions

Learn how to use the HDFS Permissions Export utility to extract HDFS native POSIX permissions from a source cluster running CDH or HDP. The extracted HDFS permissions are then used to create Ranger S3 policies that can be used in CDP One.

About this task

The HDFS Permissions Export utility exports the permissions to a .csv file in the following format with the permissions sorted in ascending order:

```
Syntax:
"/resource/path"|"username"|"groupname"|"permission"
```

where,

- "/resource/path" refers to resource entities
- "username" refers to user entries
- "groupname" refers to group entries
- "permission" refers to the HDFS permission entries

```
Example:
"/dir1"|"hdfs"|"supergroup, public"|"read, execute"
"/dir1/dir1"||"supergroup, public"|"read, execute"
"/dir1/dir1"|"hdfs"||"read, write, execute"
```

Before you begin

- The source and target migration clusters must be running the following supported versions:
  - Source cluster: HDP 2.6.5, HDP 3.1.5, CDH 5.16, CDH 6.3
  - Target cluster: CDP 7.2.15 or higher
- Secure Copy (SCP) the HDFS keytab file from the source cluster to your local system
- Run the klist command to view the list of Kerberos principals available in the keytab — ${RANGER_ADMIN_KEYTAB_PATH} and then run the kinit command to authenticate the user.

  ```
  klist -kt ${RANGER_ADMIN_KEYTAB_PATH}

  kinit -kt ${RANGER_ADMIN_KEYTAB_PATH} rangeradmin/_HOST@REALM
  ```

Procedure

1. Download the HDFSPermissionsExportTool-<version>.jar file from Cloudera Archive.

2. Run the following command to extract HDFS permissions from the source cluster.

   ```
   hadoop jar ./<path>/HDFSPermissionsExportTool-<version>.jar hdfs_namenod
   e_host:hdfs_port source_directories
   ```

   where,

   - <path> refers to the location of the downloaded .jar file.
   - hdfs_namenode_host:hdfs_port refers to the URI of the active NameNode and the HDFS NameNode port. For example, active-namenode.example.com:8020
   - source_directories refers to the root directory from where you want the permissions scan to begin. For example, "/" or "/dir1". You can also specify multiple directories — "/hbase/yarn"

   ```
   hadoop jar ./target/HDFSPermissionsExportTool-<version>.jar active-namen
   ode.example.com:8020 /user /ranger
           /hbase
   ```

   This creates a HDFS_Permissions_Export_<timestamp>.csv file that contains the required HDFS permissions.

   Important: Be sure to use the URI of the active NameNode. Using the passive NameNode URI or the NameNode HA URI results in a failure.

3. Use the scp command and copy HDFS_Permissions_Export_<timestamp>.csv from the source cluster to the migration node of the target cluster.

   Important: Since the .csv filename contains a colon (:), ensure that you provide the full path in the scp command to prevent the command from treating ":" as a separator.

What to do next

Convert the extracted HDFS native permissions into Ranger HDFS policies and then transform them into Ranger S3 policies.

Related Information

Converting HDFS native permissions into Ranger HDFS policies

Converting HDFS native permissions into Ranger HDFS policies

Learn how to convert the extracted HDFS native permissions into Ranger HDFS service policies, which can then be transformed into Ranger S3 policies.

Before you begin

- The source and target migration clusters must be running the following supported versions:
  - Source cluster: HDP 2.6.5, HDP 3.1.5, CDH 5.16, CDH 6.3
  - Target cluster: CDP 7.2.15 or higher
- You must have copied the extracted HDFS native permissions (.csv file) into the migration cluster node.
- You must have copied the Ranger policy migration utility, ranger-<version>-policymigration.tar.gz file to the migration cluster node.
  1. Log in to the migration cluster's node (or a Data Lake node of a cloud cluster) where the ranger-admin binaries are available. For example, /opt/cloudera/parcels/CDH/lib/ranger-admin
  2. Copy /opt/cloudera/parcels/CDH/lib/ranger-admin/policymigration/ranger-<version>-policymigration.tar.gz to some location of the migration cluster machine from where you want to run the Ranger policy migration utility.
- You must have read, write, and execute permissions on the directory where the Ranger policy migration utility is copied.
- You must have SUDO or ROOT access to perform the migration process.

Procedure

1. Log in to the migration cluster node where you have copied the ranger-<version>-policymigration.tar.gz file. Untar the file and navigate to the ranger-<version>-policymigration directory.
2. Update the “Transform” part of the env.sh file with the right set of values according to your environment.

   For more information, see Supported Input parameters for Transform operation.
3. Run the transform.sh script with the convert command to convert the extracted HDFS native permissions to Ranger HDFS policy format.

   ```
   Syntax:
   ./transform.sh convert $native_hdfs_permissions_csv_file_path
   ```

   ```
   Example:
   ./transform.sh convert /tmp/HDFS_Permissions_Export_<timestamp>.csv
   ```

Results

The command generates a HDFS_Permissions_Export_<timestamp>_convert.json file, which is then transformed into the required Ranger S3 policies. If you encounter any errors, see the logs/ranger-policymigration.log file.

What to do next

Transform the converted Ranger HDFS policies into Ranger S3 policies.

Related Information

Supported Input parameters for Transform operation

Extracting HDFS native permissions

Transforming Ranger HDFS policies into Ranger S3 policies

Transforming Ranger HDFS policies into Ranger S3 policies

Learn how to transform Ranger HDFS service policies into Ranger AWS S3 policies, which can then be imported into the CDP cluster Ranger.

Before you begin

- You must have converted the extracted Ranger HDFS native permissions (.csv file) into Ranger HDFS policies (.json file) using the transform.sh convert command.
- You must have copied the Ranger policy migration utility, ranger-<version>-policymigration.tar.gz file to the migration cluster node.

  1. Log in to the migration cluster's node (or a Data Lake node of a cloud cluster) where the ranger-admin binaries are available. For example, /opt/cloudera/parcels/CDH/lib/ranger-admin
  2. Copy /opt/cloudera/parcels/CDH/lib/ranger-admin/policymigration/ranger-<version>-policymigration.tar.gz to some location of the migration cluster machine from where you want to run the Ranger policy migration utility.
- You must have read, write, and execute permissions on the directory where the Ranger policy migration utility is copied.
- You must have SUDO or ROOT access to perform the migration process.

Procedure

1. Log in to the migration cluster node where you have copied the ranger-<version>-policymigration.tar.gz file. Untar the file and navigate to ranger-<version>-policymigration directory.

2. Update the "Transform" part of the env.sh file by updating the following parameters or as required for your environment.

    - S3_BUCKET_NAME
    - SERVICE_NAME_FOR_NATIVE_POLICIES

    Note: If the usernames in the target cluster are different from the usernames in the source cluster, provide a user mapping file as an input by updating the RANGER_POLICYMIGRATION_USERS_MAPPING _FILE parameter in the env.sh file.

    For more information, see Supported Input parameters for Transform operation.

3. Run the transform.sh script with the transform command to convert the Ranger HDFS policies to Ranger AWS S3 policy format.

    ```
    Syntax:
    ./transform.sh transform $converted_ranger_policies_file_path
    ```

    ```
    Example:
    ./transform.sh transform /tmp/HDFS_Permissions_Export_<timestamp>_conve
    rt.json
    ```

Results

The command generates a HDFS_Permissions_Export_<timestamp>_convert_transform.json file, which can then be imported using the Authzmigrator utility into the AWS S3 service of the target CDP cluster. If you encounter any errors, see the logs/ranger-policymigration.log file.

What to do next

Import the transformed HDFS native permissions into the Ranger AWS S3 service of the target CDP cluster using the Authzmigrator tool.

Related Information

Supported Input parameters for Transform operation

Extracting HDFS native permissions

Converting HDFS native permissions into Ranger HDFS policies

Importing Ranger AWS S3 policies

Learn how to import the transformed HDFS native permissions into the Ranger AWS S3 service of the target CDP cluster using the Authzmigrator tool.

Before you begin

- You must have extracted, converted, and transformed the native HDFS permissions into Ranger AWS S3 policy format.
- Ensure that the RANGER_ADMIN role is up and running in the Cloudera Manager/CDP One portal of the target environment.

Procedure

1. Log in to the RANGER_ADMIN portal with administrator user credentials or user privileges as ADMIN.
    a) From Settings User/Groups/Roles Search , search for the "ranger" user.
    b) Click the "ranger" user and change its role to "Admin", and save the changes.
2. Download and copy the authz-ingest.zip file to the /opt directory of the migration cluster node (or a Data Lake node of the cloud cluster).

3. Extract the authz-ingest.zip file into the /opt directory.

   Note: /opt is referred to as an example directory.

   The authz-ingest file contains the following directories:

   - config-files
   - scripts
4. If Ranger is TLS/SSL enabled using Cloudera Manager AUTO-TLS, then copy the cm-auto-global-truststore.jks file from the Ranger admin process directory of the Ranger admin host to /opt/authz-ingest/config-files/ of the migration tool host. Perform these steps to find the location of the Ranger admin process directory:

   a) Run the following command to get to the Ranger admin process:

      ps -ef | grep proc_rangeradmin

   b) Search for the -cp option from the output of the above command and look for "/var/run/cloudera-scm-agent/process/<ID>-ranger-RANGER_ADMIN"

   If you have set up SSL manually or using a different method, then copy the file that you have created for trust stores to the /opt/authz-ingest/config-files directory.

   For example, cp /path/to/cm-auto-global_truststore.jks /opt/authz-ingest/config-files/
5. Export the following environment variables to the target cluster Ranger:

   - JAVA_HOME - Java home path used on the cluster
   - RANGER_ADMIN_URL - URL of the Ranger Admin portal
   - RANGER_ADMIN_SSL_ENABLED - Indicates if SSL is enabled or disabled
   - RANGER_ADMIN_TRUSTORE_PASSWORD - If Ranger is TLS/SSL enabled using Cloudera Manager AUTO-TLS, then the password can be found in the Ranger host file, /etc/hadoop/conf/ssl-client.xml, property — 'ssl.client.truststore.password'

   ```
   export JAVA_HOME=/usr/java/jdk1.8.0_232-cloudera/
   export RANGER_ADMIN_URL=<ranger admin url with port>
   export RANGER_ADMIN_SSL_ENABLED=<true or false>
   export RANGER_ADMIN_TRUSTORE_PASSWORD=<ssl.client.truststore.password>
   ```
6. Go to /opt/authz-ingest/config-files and open the authorization-migration-site.xml file.

   The authorization-migration-site.xml contains the following properties:

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <configuration>
     <property>
       <name>authorization.migration.export.target_services</name>
       <value>HIVE,KAFKA</value>
     </property>
     <property>
       <name>authorization.migration.export.migration_objects</name>
       <value />
     </property>
     <property>
       <name>authorization.migration.export.output_file</name>
       <value>hdfs:///user/sentry/export-permissions/permissions.json</value>
     </property>
     <property>
       <name>authorization.migration.ingest.is_dry_run</name>
       <value>false</value>
     </property>
     <property>
       <name>authorization.migration.role.permissions</name>
       <value>true</value>
     </property>
   ```

```
  <property>
    <name>authorization.migration.translate.url.privileges</name>
    <value>false</value>
  </property>
  <property>
    <name>authorization.migration.ingest.merge.ifexists</name>
    <value>true</value>
  </property>
  <property>
    <name>authorization.migration.migrate.url.privileges</name>
    <value>true</value>
  </property>
</configuration>
```

7. Search for the authorization.migration.ranger.import.input_file property in the authorization-migration-site.xml file and provide the location of the transformed HDFS permissions file as the property value.

```
<property>
 <name>authorization.migration.ranger.import.input_file</name>
<value>file:///tmp/HDFS_Permissions_Export_<timestamp>_convert_transform.j
son</value>
</property>
```

If you are unable to find this property, then add the property in the file.

8. For Kerberos-based authentication, perform the following steps:

Important: If the Key Distribution Center (KDC) server is not installed in your environment or imported without the Kerberos authentication, you can skip this procedure and use the process provided for non-Kerberos authentication. Also, if PAM authentication is enabled in the target cluster, then Kerberos authentication is recommended to import ranger policies quickly.

a) Find the Ranger admin keytab location and provide it in the following command:
export RANGER_ADMIN_KEYTAB_PATH=/path/to/ranger.keytab

In the CDEP cluster, ranger keytab is available in the ranger host location /cdep/keytabs/rangeradmin.keytab or under the ranger admin process directory — /var/run/cloudera-scm-agent/process/<ranger-process-ID>-ranger-RANGER_ADMIN

b) If your ranger node and migration node are different, copy the ranger keytab to the migration node.

c) Run the klist command on the migration node to view the list of Kerberos principals available in the keytab — ${RANGER_ADMIN_KEYTAB_PATH}
klist -kt ${RANGER_ADMIN_KEYTAB_PATH}

d) Run the following kinit command and update _HOST and REALM correctly.
kinit -kt ${RANGER_ADMIN_KEYTAB_PATH} rangeradmin/_HOST@REALM

9. For non-Kerberos authentication, add the following properties in the /opt/authz-ingest/config-files/authorization-migration-site.xml file with the appropriate values, and then save and close the file.

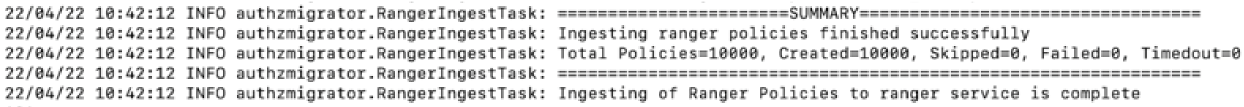
```
<property>
    <name>ranger.admin.username</name>
    <value>admin</value>
</property>

<property>
    <name>ranger.admin.passwd</name>
    <value>admin123</value>
</property>
<property>
<name>authorization.migration.kerberos.authentication</name>
<value>false</value>
</property>
```

10. Go to the /opt/authz-ingest/scripts directory and run the import script.
    sh authz-import.sh

    If the import operation fails, look for details in the output of the command printed in the terminal window or see the target Ranger admin log files.

Results

After the import operation is complete, the authzmigrator tool displays the total policies count, created, skipped, and failed policies count. You can refer to the console output logs to verify if policies are imported or not.



```
22/04/22 10:42:12 INFO authzmigrator.RangerIngestTask: =======================SUMMARY==================================
22/04/22 10:42:12 INFO authzmigrator.RangerIngestTask: Ingesting ranger policies finished successfully
22/04/22 10:42:12 INFO authzmigrator.RangerIngestTask: Total Policies=10000, Created=10000, Skipped=0, Failed=0, Timedout=0
22/04/22 10:42:12 INFO authzmigrator.RangerIngestTask: ================================================================
22/04/22 10:42:12 INFO authzmigrator.RangerIngestTask: Ingesting of Ranger Policies to ranger service is complete
```

Related Information

Extracting HDFS native permissions

Converting HDFS native permissions into Ranger HDFS policies

Transforming Ranger HDFS policies into Ranger S3 policies

Migrating Ranger policies from HDP to CDP One

In the HDP environment consisting of Ranger policies, the migration is performed using the combination of Export -> Transform -> Import operations.

About Migrating Ranger policies

Migrating Ranger policies involves export and transform operations.

Firstly, the Ranger policies available in the source HDP clusters are exported and copied to the target cluster, followed by the transform and import operations. The export and transform operations are performed using the policy migration utility.

The Ranger Policy Migration utility files in the directory

ranger-<version>-policymigration consists of the following scripts:

- env.sh
- exportPolicy.sh
- transform.sh

The contents of the policy migration utility would be copied to the source HDP cluster from which the Ranger policies are to be migrated. The output file from the transform process would be transitioned to the import operations. The import process consists of the import utility (Authzmigrator tool).

Note: Currently, the migration process does not support migrating Group / Role name transformation.

Migration prerequisites

You need to know the prerequisites for migrating Ranger policies from HDP to CDP One.

You must meet the following general prerequisites before starting the migration process:

- AWS is your Cloud provider. Migrating to Azure / GCP is currently not supported.
- The source and target migration clusters must be running the following supported versions:
  - Source cluster: HDP 2.6.5, HDP 3.1.5.
  - Target cluster: CDP 7.2.15 or higher.
  - The minimum JAVA version. Default is 1.8.

Copying Policy Migration utility to the source cluster

Policy migration utility must be available on the source cluster.

About this task

You must copy the contents of the policy migration tool to the source cluster.

Procedure

1. Sign-in to the Migration cluster's node (or a Data Lake node of the cloud cluster) where the ranger-admin binaries are available.
   For example: /opt/cloudera/parcels/CDH/lib/ranger-admin
2. Copy /opt/cloudera/parcels/CDH/lib/ranger-admin/policymigration/ranger-<version>-policymigration.tar.gz to the source HDP cluster machine from where you want to export the HDP Ranger policies.

   Additionally, copy the same file to some other location of the migration cluster machine (or a Data Lake node of the cloud cluster) from where you want to execute the Ranger policy migration utility to transform Ranger policies.
3. Login to the HDP source cluster where you have copied the ranger-<version>-policymigration.tar.gz file. Untar the file and navigate to ranger-<version>-policymigration directory.

Performing Export and Transform operations

Using the Policy Migration utility, you must perform export and transform operations.

About the export operation

Call exportPolicy.sh script to export Ranger policies.

The exportPolicy.sh script performs the following tasks:

- Reads the env.sh file and loads the input parameters values
- Makes a CURL request to ranger/export JSON REST API and sends the required parameters
- Downloads and saves the exported Ranger policies at the location provided in $EXPORT_OUTPUT_LOCATION. It also downloads and saves the exported Ranger services at the same location with the name exportedRangerServices.json

Note: Valid for HDP cluster only.

Running the export operation

You must export the Ranger policies.

Procedure

1. Update the "export" part of the env.sh file with the right set of values as per your environment.
2. Run the exportPolicy.sh file.

   Running sh.exportPolicy.sh could throw an error because env.sh file is not available. You can resolve this condition by running the script file with the following usage: ./exportPolicy.sh
3. Verify if the exported Ranger policies file was created or not.
4. Verify if the exported Ranger services file was created or not.

   If any error refers to the source ranger-admin logs.
5. Copy the exported Ranger policies JSON file and exported Ranger services JSON file to the migration cluster node (or a Data Lake node of cloud cluster).

Related Information

Supported Input parameters for Export operation

About the transform operation

Call transform.sh to transform Ranger HDFS policies to Ranger AWS S3 policies and replace policy field’s value as per the target environment.

The transform.sh script performs the following tasks:

- Reads env.sh file and loads the input parameters values
- Checks Java versions
- Creates provided system user and group
- Provides ownership to the above created system user on install files and folders
- Update given parameters values to ranger-policymigration-site.xml
- Exports JAVA_OPTS and adds required options in the environment.
- Can call the Ranger PolicyMigration jar/code for convert/transform as requested

Running the transform operation

You must transform the Ranger policies.

Procedure

1. Login to the migration cluster node (or a Data Lake node of the cloud cluster) where you have copied the ranger-<version>-policymigration.tar.gz file. Untar the file and navigate to ranger-<version>-policymigration directory.
2. Update the “Transform” part of the env.sh file with the right set of values as per your environment.
3. Transform Ranger policies: Run the transform.sh file.

   a. Syntax: ./transform.sh transform $exported_ranger_policies_file_path $exported_ranger_services_file_path
   b. Example: ./transform.sh transform /tmp/policies.json /tmp/services.json

   Note: $exported_ranger_policies_file_path and $exported_ranger_services_file_path are the paths of exported Ranger policies and services file in JSON format received from Running the export operation on page 50.

   Attention: Output of the above command is in the JSON format and this JSON file is required for Performing Import operation on page 51. If any error is found, refer to logs/ranger-policymigration.log file.

Related Information

Supported Input parameters for Transform operation

Performing Import operation

You must import the Ranger policies to CDP cluster Ranger.

Note: If the users that are defined within the policy do not exist on the target cluster / storage, they are auto-created, import operation succeeds, and no further user action is required.

Procedure

1. Login to the RANGER_ADMIN portal with administrator user credentials or user privileges as ADMIN.

   a. Under Settings > Users/Groups/Roles > Search for the ranger user.
   b. Click on the ranger user and change its role to Admin and save the changes.
2. Download and copy the file authz-ingest.zip to the /opt directory of migration node (target data lake Ranger node).

   For example: CDP Data Lake or any relevant target cluster.

3. Unzip the authz-ingest.zip file into the /opt directory which contains the following files:

    - config-files
    - scripts

    The config-files contains the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
  <property>
    <name>authorization.migration.export.target_services</name>
    <value>HIVE,KAFKA</value>
  </property>
  <property>
    <name>authorization.migration.export.migration_objects</name>
    <value />
  </property>
  <property>
    <name>authorization.migration.export.output_file</name>
    <value>hdfs:///user/sentry/export-permissions/permissions.json</value>
  </property>
  <property>
    <name>authorization.migration.ingest.is_dry_run</name>
    <value>false</value>
  </property>
  <property>
    <name>authorization.migration.role.permissions</name>
    <value>true</value>
  </property>
  <property>
    <name>authorization.migration.translate.url.privileges</name>
    <value>false</value>
  </property>
  <property>
    <name>authorization.migration.ingest.merge.ifexists</name>
    <value>true</value>
  </property>
  <property>
    <name>authorization.migration.migrate.url.privileges</name>
    <value>true</value>
  </property>
</configuration>
```

    Attention: While using permissions.json file, you might encounter an error or a warning message while using the default configuration. You can ignore this and continue with the process.

4. If Ranger is TLS/SSL enabled using Cloudera Manager AUTO-TLS, you must then from the Ranger admin host, copy cm-auto-global_truststore.jks (from the Ranger-admin process directory).

   Note: In CDP One or CDP Public Cloud environment TLS is always enabled by default.

   For example:

   /var/run/cloudera-scm-agent/process/<ranger-process-ID>-ranger-RANGER_ADMIN) to the location /opt/authz-ingest/config-files/ of the migration tool host.

   Follow these steps to find the Ranger admin process directory:

   - Execute the below command to get to the RANGER_ADMIN process:

   ```
   ps -ef | grep proc_rangeradmin
   ```

   - Search for -cp option from the above command output and look for

     "/var/run/cloudera-scm-agent/process/<ID>-ranger-RANGER_ADMIN"

   If you have done SSL setup manually or in a different method, copy the filename which you created for truststores.

   For example:

   ```
   cp /path/to/cm-auto-global_truststore.jks       /opt/authz-ingest/config-files/
   ```

5. Export the following variables to the target Ranger:

   - **a.** JAVA_HOME - Java home path used on the cluster
     b. RANGER_ADMIN_URL - Ranger Admin portal url
     c. RANGER_ADMIN_SSL_ENABLED - either true or false
     d. RANGER_ADMIN_TRUSTORE_PASSWORD - If Ranger is TLS/SSL enabled using Cloudera Manager AUTO-TLS, then the password can be found on the Ranger host /etc/hadoop/conf/ssl-client.xml property 'ssl.client.truststore.password'

6. Go to the location /opt/authz-ingest/config-files and open the file authorization-migration-site.xml

   Locate the property “authorization.migration.ranger.import.input_file” and provide the location of the transformed output file generated from Running the transform operation on page 51.

   Note: If the property is not available then you must add this property in the file.

   Example: file:///tmp/exportedRangerPolicies_transform.json

   ```
   <property>
   <name>authorization.migration.ranger.import.input_file</name>
   <value>file:///tmp/exportedRangerPolicies_transform.json</value></property>
   ```

7. For the Kerberos-based authentication (or in a non CDP-SaaS environment) use the following process

   If the KDC server is not installed in your environment or imported without the Kerberos authentication, you can skip this procedure and use the process provided for non-Kerberos authentication. Also if PAM auth is enabled in the target cluster then Kerberos authentication is recommended to import Ranger policies quickly.

   - Find the Ranger admin keytab location and provide it in the following command:

   export RANGER_ADMIN_KEYTAB_PATH=/path/to/ranger.keytab

   In the CDEP cluster, Ranger keytab shall be available in the Ranger host location "/cdep/keytabs/rangeradmin.keytab" or under the Ranger admin process directory

   "/var/run/cloudera-scm-agent/process/<ranger-process-ID>-ranger-RANGER_ADMIN" location.

   Steps to find the Ranger admin process directory:

   - Execute the following command to get to the RANGER_ADMIN process.

   ps -ef | grep proc_rangeradmin

   - Search for -cp option from the above command output and look for "/var/run/cloudera-scm-agent/process/<ID>-ranger-RANGER_ADMIN"
   - If your Ranger node and migration node (target data lake Ranger node) are different, copy the Ranger keytab to the migration node (target data lake Ranger node).
   - Run the below klist command on the migration node (target data lake Ranger node) to get the list of principals available in ${RANGER_ADMIN_KEYTAB_PATH}

   klist -kt ${RANGER_ADMIN_KEYTAB_PATH}

   - Run the below kinit command using the ${RANGER_ADMIN_KEYTAB_PATH} with rangeradmin/_HOST@REALM principle.
   - Update the _HOST and the REALM correctly using the below command.

   kinit -kt ${RANGER_ADMIN_KEYTAB_PATH} rangeradmin/_HOST@REALM
8. For non-Kerberos authentication, add the following properties and provide relevant values. After providing the values to configurations, save, and close the file.

```
<property>
    <name>ranger.admin.username</name>
    <value>admin</value>
</property>
<property>
<name>authorization.migration.kerberos.authentication</name>
<value>false</value>
</property>
```

9. Run the migration script:

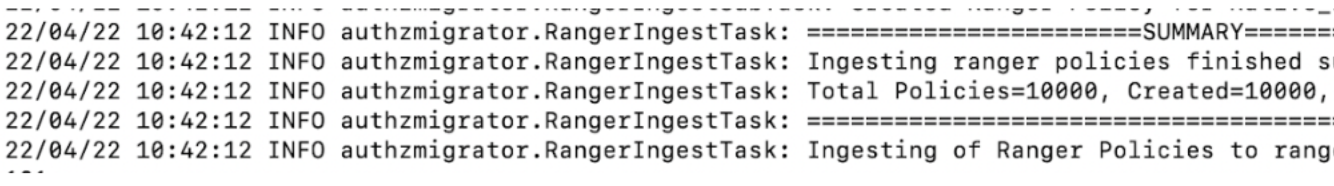
   cd authz-ingest/scripts

   sh authz-import.sh

   Note: If import operation fails, locate for details in the output of the command printed on the terminal window or refer to the target Ranger admin log files.

10. After execution, the authMigrator utility shall print total policies count, skipped policies count and the failed policy counts on the same console window. Users

should refer to the console output logs to ensure policies have been imported or not.



```
22/04/22 10:42:12 INFO authzmigrator.RangerIngestTask: =======================SUMMARY======
22/04/22 10:42:12 INFO authzmigrator.RangerIngestTask: Ingesting ranger policies finished s
22/04/22 10:42:12 INFO authzmigrator.RangerIngestTask: Total Policies=10000, Created=10000,
22/04/22 10:42:12 INFO authzmigrator.RangerIngestTask: ====================================
22/04/22 10:42:12 INFO authzmigrator.RangerIngestTask: Ingesting of Ranger Policies to rang
```

Supported Input parameters for Export operation

When you perform the export operation, depending on your cluster environment, you must make sure to configure the env.sh file for the supported input values accordingly.

Input Parameters	Description
POLICYMGR_EXTERNAL_URL	This configuration is used for the Ranger administrator URL. Ranger admin URL should be given along with protocol and port. Multiple Ranger admin URLs are not supported. Example: http://testhost.root.hwx.site:6080 Default : http://localhost:6080 Mandatory: Yes
POLICYMGR_KERBEROS_ENABLED	For kerberos authentication, set this configuration value to true. Kinit with right principal/keytab for authentication before executing the exportPolicy.sh script. Default : false Mandatory: No
POLICYMGR_HTTPS_ENABLED	For SSL/HTTPS authentication, set this configuration value to true. If this property is set to true, provide a valid certificate file path in the configuration POLICYMGR_HTTPS_CERT_FILE Default : false Mandatory: No
POLICYMGR_HTTPS_CERT_FILE	For SSL/HTTPS authentication set config POLICYMGR_HTTPS_ENABLED value to true and provide a valid certificate file path for this config Example: /tmp/cacerts.pem Default : empty string Mandatory: No
POLICYMGR_ADMIN	This configuration is for Ranger admin user loginID. loginID provided in this config must exist in ranger-admin and must have an ‘admin’ role. For example: If POLICYMGR_KERBEROS_ENABLED is set to true, POLICYMGR_ADMIN does not need to be configured with a user id. It is inherited from the authenticated Kerberos principal." This is optional for secured/kerberos/knoxsso authentication. Default : empty string Mandatory: No

Input Parameters	Description
POLICYMGR_PWD	This configuration is for the login password of the loginID provided in config POLICYMGR_ADMIN. This is optional for secured/kerberos/knoxsso authentication. Default : empty string Mandatory: No
SOURCE_SERVICE_NAME	This configuration is for Ranger service names of which Ranger policies need to be exported. Multiple service names can be given separated by comma(,) characters. If an empty string is provided then the utility will try to export Ranger policies of all services. Example: hadoopdev,hivedev Default : empty string Mandatory: No
EXPORT_OUTPUT_LOCATION	This configuration is for the output file of exported Ranger policies. If a location is given then the user must have write permission on that file/directory. If an empty string is given then the location shall be the current location. Only local file system path is supported and HDFS file system path is not supported. If directory path is given then file name shall be “exportedRangerPolicies_$timestamp”. If path contains the file name also then it will create/overwrite the existing file with the exported Ranger policies JSON content. example : /tmp Default : empty string Mandatory: No
POLICYMGR_KNOXSSO_TOKEN	This config is to provide the KNOX SSO Token for knox sso based authentication to source Ranger-admin. Example : eyJhbGciOiJSUzI1NiJ9.eyJzdWIiOiJhZG1pbjEiLCJpc3MiOiJLTk9YU1NPIiwiZXhwIjox gB-AVIbnfoKQoJobHTAAXgHiV-6dehFK2oWKWDX7ZXwj5Oxf-11a7ptBkRlRiPeJW7M Default : empty string Mandatory: No

Note: To retrieve the Knox SSO token make a CURL request to ranger-admin and fetch details.

Example : curl -iku ‘admin’:’admin’ "https://{KNOX-HOST}:8443/gateway/knoxsso/api/v1/websso?originalUrl=http://{RANGER-ADMIN-HOST}:6080/service/plugins/services/1"

Sample output:

```
HTTP/1.1 307 Temporary Redirect

Date: Wed, 09 Mar 2022 04:11:28 GMT

X-Frame-Options: DENY

Set-Cookie: KNOXSESSIONID=eabe18mw4qrs1f58xjg9d000q;Path=/gateway/knoxsso;Secure;HttpOnly

Expires: Thu, 01 Jan 1970 00:00:00 GMT

Set-Cookie: rememberMe=deleteMe; Path=/gateway/knoxsso; Max-Age=0; Expires=Tue, 08-Mar-2022 04:11:28 GMT

Set-Cookie: hadoop-jwt=eyJhbGciOiJSUzI1NiJ9.eyJzdWIiOiJhZG1pbjEiLCJpc3MiOiJLTk9YU1NPIiwiZXhwIjoxNjQ2ODk5MDg4fQ.TZf18N0zHMU
```

gB-AVIbnfoKQoJobHTAAXgHiV-6dehFK2oWKWDX7ZXwj5Oxf-11a7ptBkRlRiPeJW7MmHuZgdJIKmauJMmp8Z_67diImU__6wj

Location: http://localhost:6080/service/plugins/services/1

Content-Length: 0

Server: Jetty(9.2.15.v20160210)

Supported Input parameters for Transform operation

When you perform the transform operation, depending on your cluster environment, you must make sure to configure the env.sh file for the supported input values accordingly.

Input paramaters	Description
PYTHON_COMMAND_INVOKER	This configuration is for the python command name or path to the python command file. Python 2.x and 3.x both versions are supported. If python version 3.X command is not aliased with python then python3 must be entered here. If python 2 is installed then python or python2 must be provided as a value to this config. Default : python Mandatory: Yes
RANGER_POLICYMIGRATION_EXCLUDE_SERVICE_TYPES	This configuration is to exclude the Ranger policies of given Ranger service types during the transformation process. Ranger policies of given service-types will not be added in the transformed output file. **Note:** Currently “tag” migration is not supported, hence it is recommended that users must add "tag" to exclude service types. Example: storm,tag Default : storm,tag Mandatory: No
SOURCE_DESTINATION_RANGER_SERVICE_TYPE_MAPPING	This config is to tell the utility the mapping of source and target Ranger service type. Based on the given source service type, Ranger policies will be transformed to a given target service type. Source and target service type are separated by colon (:) At present, Only Ranger hdfs service policies can be converted to s3 only. Example: {hdfs:s3} Default : {hdfs:s3} Mandatory: No
SOURCE_DESTINATION_RANGER_SERVICE_NAME_MAPPING	This configuration is to provide the mapping of source and target Ranger service names to the utility. Based on the given service name mapping, during transformation service names of given Ranger policies will be replaced with the target service names. Source and target service names are separated by colon (:). Multiple entries are allowed and need to be separated by comma(,) character. If no mapping is provided then service names will remain the same even after the transformation process. Example: {cm_hdfs:cm_s3,c1_yarn:cm_yarn} Default : empty string Mandatory: No

Input paramaters	Description
SOURCE_DESTINATION_RANGER_ZONE_NAME_MAPPING	This configuration is to provide the mapping of source and target Ranger zone names to the utility. Based on the given zone name mapping, during transformation, zone names of given Ranger policies will be replaced with the target zone names. Source and target zone names are separated by colon (:). Multiple entries are allowed and need to be separated by comma(,) character. If no mapping is provided then zone names will remain the same even after the transformation process. Example: {oldzone1:newzone1} Default : empty string Mandatory: No

RANGER_POLICYMIGRATION_RESOURCE_MAPPING_FILE	This configuration is to provide the source and target service resource mappings to the utility. Based on the given resources mapping, during transformation resources of given Ranger policies will be replaced with the target resources. If no mapping is provided then the resource string will remain the same even after the transformation process.
	File format is CSV. Local filesystem path is supported only.
	Example value: /path to /resourceMappings.csv
	Default : empty string
	Mandatory: No
	Resource mapping for hdfs policy resource path=> For the JSON file format we can have multiple resource also in the JSON object formatted like :
	Sample resource mapping entries for a json file:
	`{"/srcdir1/srcfile1":"/tardir1/tarfile1",`
	`"/srcdir2/srcfile2":"/tardir2/tarfile2"}`
	=> And with the CSV format, it's pipe-delimited with one entry per line.
	Sample resource mapping entries for a csv file:
	/srcdir1/srcfile1|/tardir1/tarfile1
	/srcdir2/srcfile1|/tardir2/tarfile1

Input paramaters	Description
RANGER_POLICYMIGRATION_HIVE_URL_MAPPING_FILE	This config is to provide the source and target HIVE service URL policy resource mappings to the utility. Based on the given URL mapping, during transformation the URL resource of given Ranger Hive URL policies will be replaced with the target URL value. If no mapping is provided then the resource string will remain the same even after the transformation process. File format is CSV. Local filesystem path is supported only. Example value: /path to /resourceMappings.csv Default : empty string Mandatory: No Resource mapping for hive policy URL resource=> For the JSON file format we can have multiple resource also in the JSON object formatted like : Sample resource mapping entries for a json file: {"/user/hive/warehouse/customers/customers":"s3a://mybucket/user/hive/warehouse/customers/customers","hdfs://dysentry01-3.dysentry01.root.hwx.site:8020":"s3a://mybucket1"} => And with the CSV format, it's pipe-delimited with one entry per line. Sample resource mapping entries for a csv file: /user/hive/warehouse|s3a://mybucket/user/hive/warehouse hdfs://localhost:8020|s3a://mybucket1
S3_BUCKET_NAME	This configuration is for S3 bucket name which shall be assigned as a value to the resource type “bucket” key of each transformed policy of S3. If the target Ranger service type is S3 then value for this property must be provided. Default : myS3Bucket Mandatory: No
RANGER_POLICYMIGRATION_USERS_MAPPING_FILE	This configuration is to provide the mapping of usernames in the source and target Ranger policies. Based on the given user names mapping, during transformation user names in the given Ranger policies will be replaced with the target user names. If no mapping is provided then user names will remain the same even after the transformation process. File format is CSV. Local filesystem path is supported only. Example: /path/to/usersMappings.csv Default : empty string Mandatory: No Format: => For the JSON file format we can have multiple users also in the JSON object formatted like : {“testuser1”:”testuser1@host.com”, “testuser2”:”testuser2@host.com”} => And with the CSV format, it's pipe-delimited with one entry per line. testuser1|testuser1@host.com testuser2|testuser2@host.com

Input paramaters	Description
RANGER_POLICYMIGRATION_TRANSFORM_OUTPUT_FILE	This configuration is for the output file of transformed Ranger policies. If a location is given then the user must have write permission on that file/directory. If an empty string is given then the location shall be the same as the location of the input file. Local file system path should be used for this configuration. If a directory path is given then the suffix "transform" will be added in the input file name. If path contains file name also then utility shall create/overwrite the existing file with the transformed Ranger policies JSON content. Example : /tmp Default : empty string Mandatory: No
SERVICE_NAME_FOR_NATIVE_POLICIES	This configuration is to assign a service name to the HDFS Ranger policies. These Ranger policies are those Ranger policies which are transformed from the native HDFS permissions. Example : cm_hdfs Default : cm_hdfs Mandatory: Yes
RANGER_POLICYMIGRATION_LOG_DIR	Path to log file directory. Transform utility shall write the log files in this directory.

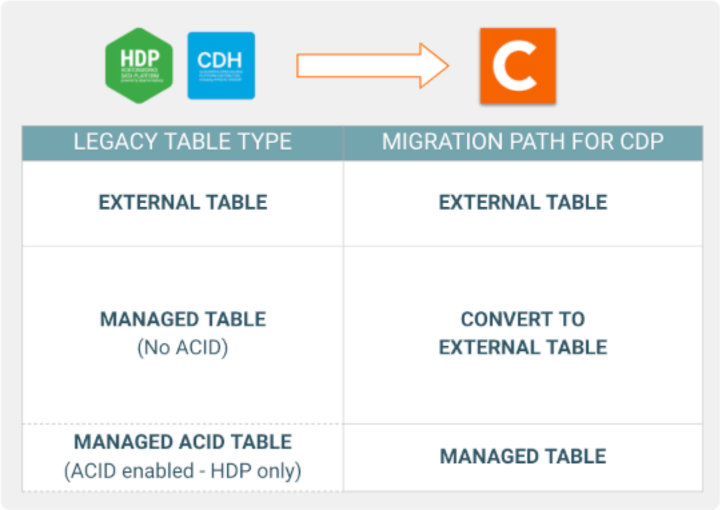
POLICYMIGRATION_INSTALL_DIR	Path of migration utility parents directory, default is present working directory.
	Example : ${PWD}
	Default : ${PWD}
	Mandatory: No

LOGFILE	Name of the log file. Transform utility shall write all the logs to this file.
LOGFILES	Name of the log file. Presently there is a single log file for all operations of the migration utility.
RANGER_POLICYMIGRATION_MAX_HEAP_SIZE	This allocates Max heap size for the transform utility execution. Default value is 4GB.
UNIX_USER	This configuration is for the user who will be the owner of Ranger policy migration utilities directory and files.
UNIX_USER_PWD	
UNIX_GROUP	This configuration is for the group name which shall have the same permissions as the owner of the Ranger policy migration utility directory and files.

Migrating Hive data from HDP 2.x or HDP 3.x to CDP One

The recommended way to migrate Hive data from HDP to CDP One depends on the types of tables you are migrating: external, legacy-managed, or ACID (managed) tables.

The default behavior in CDP has changed so that all newly created managed tables are transactional tables by default. To minimize impact to applications during upgrade and keep legacy behavior, all non-acid legacy tables (both managed and external) will be converted to external tables in CDP during migration. Legacy ACID tables will be migrated to managed ACID tables in CDP.



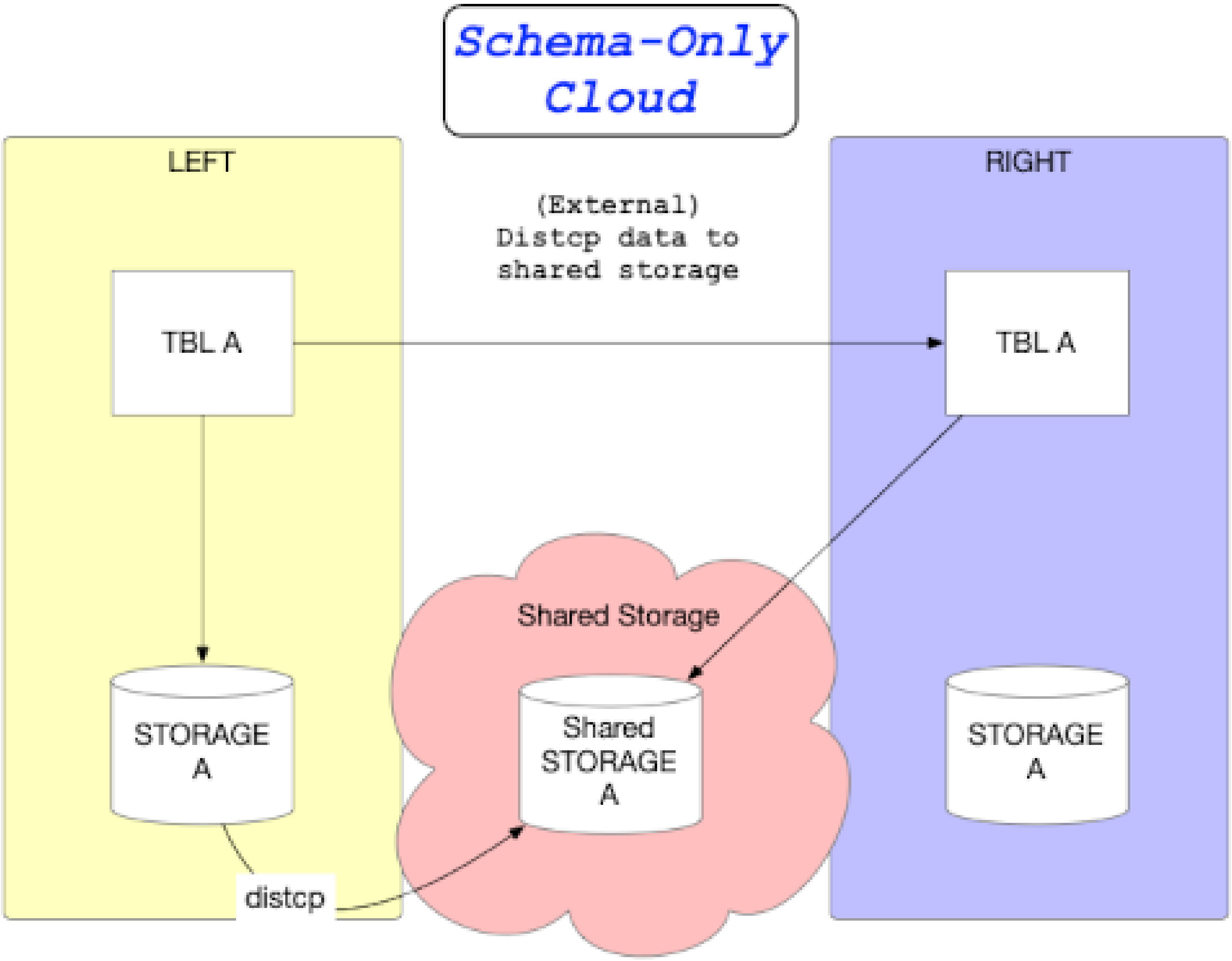
LEGACY TABLE TYPE	MIGRATION PATH FOR CDP
EXTERNAL TABLE	EXTERNAL TABLE
MANAGED TABLE (No ACID)	CONVERT TO EXTERNAL TABLE
MANAGED ACID TABLE (ACID enabled - HDP only)	MANAGED TABLE

Scenario 1: Migrating Non-ACID tables (SCHEMA_ONLY + distcp)

All managed and external Non-ACID tables will be migrated to external tables in CDP using the following components:

- HMS-Mirror: using the SCHEMA_ONLY mode to transfer metadata
- DistCP

In this scenario, HMS-Mirror and distcp are triggered from the source cluster (LEFT):

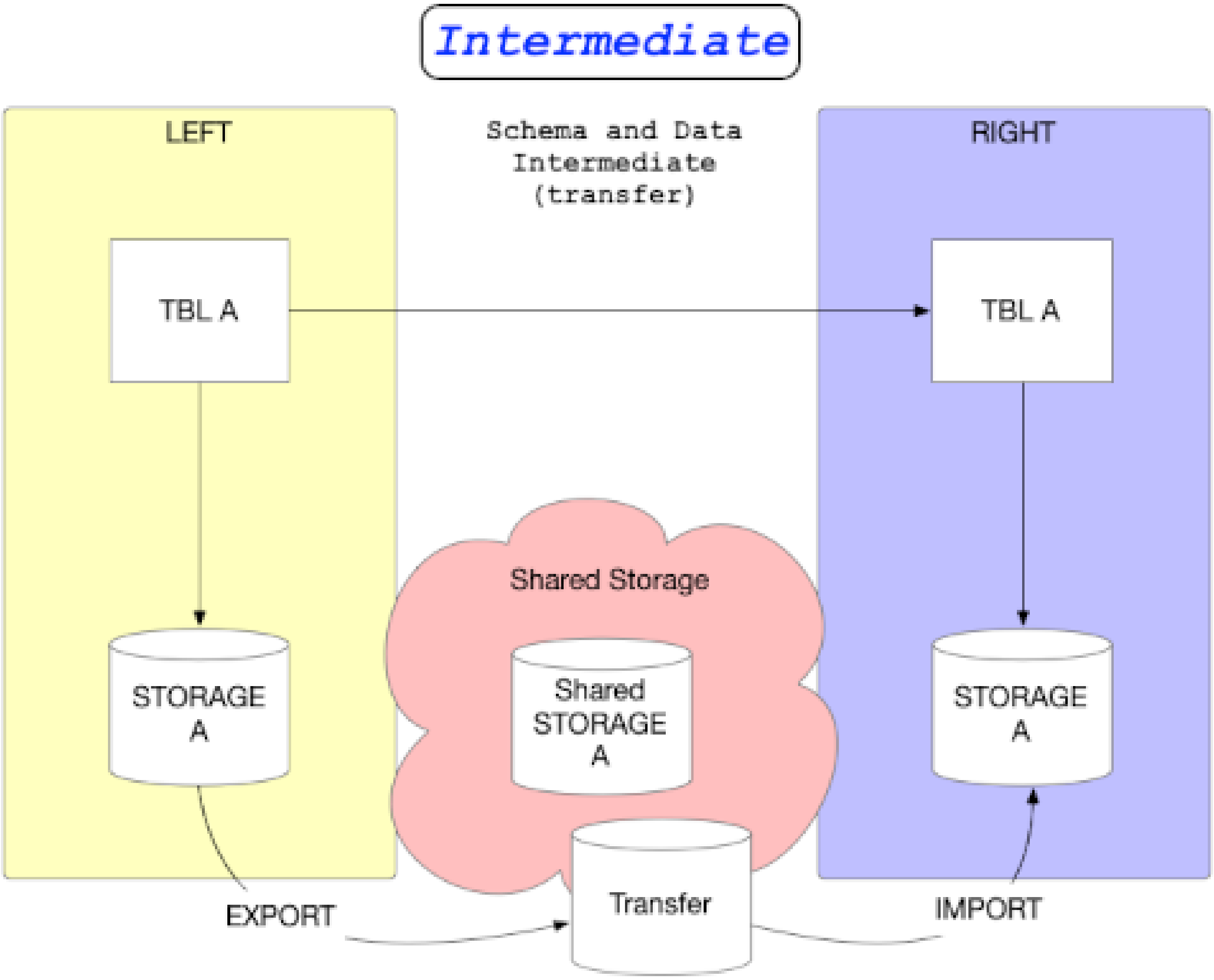


If direct Hive connectivity from on-prem to CDP is not available, you can dump of the schema to export it, and then preform a re-run manually in the target cluster.

Scenario 2: Migrating ACID tables (HYBRID + MIGRATE_ACID)

All managed ACID source tables (applicable to HDP) are migrated to ACID tables in CDP using HMS Mirror in the HYBRID migrate-acid mode with the intermediate-storage option. This approach migrates both data and metadata using Hive queries, and stages the data in an intermediate-storage location on S3. This approach avoids the need to link the target cloud environment with the on-prem source.

In this scenario, HMS-Mirror is triggered from the source cluster (LEFT), as shown in the following diagram:



Migration prerequisites

Migrating Hive metadata to CDP One requires setup tasks, some of which are related to security.

In addition to the step-by-step set up tasks listed below, put the SSL certificate on the HDP cluster edge node.You need to migrate Ranger policies from HDP to CDP, set up AVRO table migration if you have Hive tables in AVRO format. You need to setup Hive JDBC standalone JARS for connecting the HDP and CDP clusters.

After performing the following setup tasks, you simply run the HMS Mirror command to migrate the data.

Setting up Hive JDBC standalone JARS

You must have the Hive JDBC standalone jar compatible with your HDP version and the latest Hive JDBC standalone jar compatible with CDP One on the edge node between the HDP and CDP One cluster.

Procedure

1. On the HDP cluster, navigate to /usr/hdp/current/hive-client/lib check that the Apache Hive standalone JAR was installed by HDP on your cluster.
   For example, on HDP 2.6.5 you find a JAR having a name something like hive-jdbc-1.2.1000.2.6.5.5000-33-standalone.jar.

   On HDP 3.1.5, you find a JAR having a name something like hive-jdbc-3.1.0-315.133-2-standalone.jar.

   ⚠ **Important:** Hive 3 jars do not work on an HDP 2.x cluster.

2. If the JAR is not on the cluster, using the version number of the JAR in /usr/hdp/current/hive-client/lib, search for and download the Apache Hive JDBC driver hive-jdbc from the driver archive.
3. On the CDP cluster, navigate to /opt/cloudera/parcels/<version>/jars/hive-jdbc-3<version>, and check that the Apache Hive standalone JAR is on the cluster.
4. If the JAR is not on the cluster, download the latest Hive JDBC driver from the Cloudera Downloads page.
5. Make a note of the path to the JAR to connect the HDP and CDP clusters later.

Saving Hive metastore on HDP by dumping

You must dump the Hive metastore metadata immediately before upgrading or migrating. Dumping the metastore data to a file is critical to prevent data loss.

Procedure

1. On the node where the database you use for Hive Metastore resides, dump Hive Metastore metadata before upgrading/migrating to HDP.
   For example, in MySQL, dump each database as follows:

   ```
   mysqldump <hive_db_schema_name> > </path/to/dump_file>
   ```

   In Postgres, dump each database as follows:

   ```
   pg_dump --opt <metastore_db_name> > /path/to/dump_file
   ```
2. Proceed to upgrade/migrate HDP, assuming no Hive update, delete, or merge occurred after compaction; otherwise, repeat the compaction and Hive Metastore dump procedures, and then upgrade/migrate from HDP.

Taking a mandatory snapshot of HDP tables

Taking a snapshot of Hive tables is mandatory before upgrading or migrating. You also need to keep track of how many tables you have before and after for comparison.

Procedure

1. In Ambari, go to Services/Hive/Configs, and check the value of hive.metastore.warehouse.dir to determine the location of the Hive warehouse, /apps/hive/warehouse by default.
2. On any node in the cluster, as the HDFS superuser, enable snapshots.

   ```
   $ sudo su - hdfs
   $ hdfs dfsadmin -allowSnapshot /apps/hive/warehouse
   ```

   ```
   Allowing snapshot on /apps/hive/warehouse succeeded
   ```
3. Create a snapshot of the Hive warehouse.

   ```
   $ hdfs dfs -createSnapshot /apps/hive/warehouse
   ```

   Output includes the name and location of the snapshot.

   ```
   Created snapshot /apps/hive/warehouse/.snapshot/s20181204-164645.898
   ```
4. Start Hive as a user who has SELECT privileges on the tables.

   ```
   $ beeline
   beeline> !connect jdbc:hive2://
   Enter username for jdbc:hive2://: hive
   Enter password for jdbc:hive2://: *********
   ```

   ```
   Connected to: Apache Hive (version 1.2.1000.2.6.5.0-292)
   ```

```
Driver: Hive JDBC (version 1.2.1000.2.6.5.0-292)
```

5. Identify all tables outside /apps/hive/warehouse/.

   ```
   hive> USE my_database;
   hive> SHOW TABLES;
   ```

6. Determine the location of each table using the DESCRIBE command. For example:

   ```
   hive> DESCRIBE FORMATTED my_table partition (dt='20181130');
   ```

7. Create a snapshot of the directory shown in the location section of the output.
8. Repeat steps 5-7 for each database and its tables outside /apps/hive/warehouse/.

Setting up security

You need to turn off Kerberos and provide an authentication certificate on respective clusters before running HMS Mirror.

Procedure

1. Copy the cert.txt file to all the hosts in the destination CDP One cluster.

   ```
   $ mv cert.txt othercert.txt
   $ scp othercert.txt <path on CDP SaaS>
   ```

2. Import the certificate into the keystore file on all the hosts of the destination cluster.

   ```
   /usr/java/default/bin/keytool -importcert -noprompt -v -trustcacerts -ke
   ystore /var/lib/cloudera-scm-agent/agent-cert/cm-auto-global_truststore.
   jks -alias cmrootca-1 -file ./othercert.txt --storepass [***truststore_p
   assword***]
   ```

Installing and configuring HMS Mirror

You can perform the migration when HMS Mirror is installed on the HDP cluster.

About this task

You download, install, and configure HMS Mirror software. Configuration consists of generating and customizing an HMS Mirror config file in YAML format. For example, you set the path of the hcfsNamespace, which is the HDFS file system path.

Before you begin

- Remove any previous installation of HMS Mirror. For example, as root run the following command. rm -f /usr/local/hms-mirror/lib/*.jar. For more information, see setup.sh generated by a dry run of HMS Mirror.>>
- On the HDP cluster, from the Ambari configuration in /etc/ambari-server conf, get the hcfsNamespace.
- On the CDP One cluster, get the path to S3 object storage.
- On the CDP One cluster, get the HiveServer (HS2) URI from the SaaS interface.
- On the CDP One cluster, get the Java KeyStore (jks) file name and path to set the sslTrustStore.

Procedure

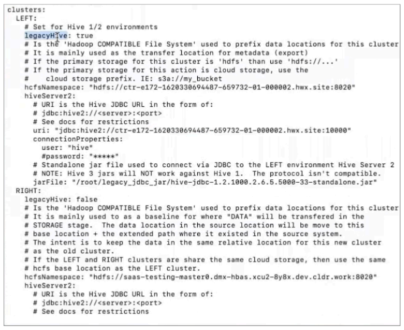
1. On the source HDP cluster, go to https://github.com/cloudera-labs/hms-mirror.
2. Download the HMS Mirror distribution, and extract the files.
3. Go to a directory other than the HMS Mirror installation directory.

   You should not run HMS Mirror from the directory where you installed the software.

4. Generate a config file .hms-mirror/cfg/default.yaml using the -setup option.

```
hms-mirror -setup
```

The config file is generated.



```
clusters:
  LEFT:
    # Set for Hive 1/2 environments
    legacyHive: true
    # Is the 'Hadoop COMPATIBLE File System' used to prefix data locations for this cluster
    # It is mainly used as the transfer location for metadata (export)
    # If the primary storage for this cluster is 'hdfs' than use 'hdfs://...'
    # If the primary storage for this action is cloud storage, use the
    #    cloud storage prefix. IE: s3a://my_bucket
    hcfsNamespace: "hdfs://ctr-e172-1620330694487-659732-01-000002.hwx.site:8020"
    hiveServer2:
      # URI is the Hive JDBC URL in the form of:
      # jdbc:hive2://<server>:<port>
      # See docs for restrictions
      uri: "jdbc:hive2://ctr-e172-1620330694487-659732-01-000002.hwx.site:10000"
      connectionProperties:
        user: "hive"
        #password: "*****"
      # Standalone jar file used to connect via JDBC to the LEFT environment Hive Server 2
      # NOTE: Hive 3 jars will NOT work against Hive 1.  The protocol isn't compatible.
      jarFile: "/root/legacy_jdbc_jar/hive-jdbc-1.2.1000.2.6.5.5000-33-standalone.jar"
  RIGHT:
    legacyHive: false
    # Is the 'Hadoop COMPATIBLE File System' used to prefix data locations for this cluster
    # It is mainly used to as a baseline for where "DATA" will be transfered in the
    # STORAGE stage.  The data location in the source location will be move to this
    # base location + the extended path where it existed in the source system.
    # The intent is to keep the data in the same relative location for this new cluster
    # as the old cluster.
    # If the LEFT and RIGHT clusters are share the same cloud storage, then use the same
    # hcfs base location as the LEFT cluster.
    hcfsNamespace: "hdfs://saas-testing-master0.dmx-hbas.xcu2-8y8x.dev.cldr.work:8020"
    hiveServer2:
      # URI is the Hive JDBC URL in the form of:
      # jdbc:hive2://<server>:<port>
      # See docs for restrictions
```

5. Scroll to the cluster information about both LEFT (HDP) and RIGHT (CDP) clusters at the bottom of the file.
6. Enter Boolean values for legacyHive to identify the source cluster.
   For example:

   - On the LEFT (HDP) cluster, set legacyHive: true
   - On the RIGHT (CDP) cluster, you set legacyHive: false
7. On the RIGHT cluster, enter the hostname and port for S3 object storage.
   For example: S3a://bucket_name/path
8. Enter the HiveServer (HS2) server name and port for the jdbc url of the LEFT source HDP cluster.
   For example, jdbc:hive2://<hs2 server name>: 10000
9. Enter the HiveServer URI of the RIGHT destination CDP cluster, including the sslTrustStore path and trustStorePassword.
   The URI looks something like this:

```
"jdbc:hive2://saas/testing-master0.dmx:443/;ssl=true;transportMode=http:
httpPath=saas-testing/cdp-proxy-token/hive;sslTrustStore=/root/cert/gate
way-client-trust.jks;trustStorePassword=changeit;"
```

10. Save changes to the YAML config.

Sample YAML configuration file

You customize a YAML file generated by the HMS Mirror dry run for your migration by looking at an example YAML.

```
clusters:
  LEFT:
    # Set for Hive 1/2 environments
    legacyHive: false
    # Is the 'Hadoop COMPATIBLE File System' used to prefix data locations f
or this cluster.
    # It is mainly used as the transfer location for metadata (export)
    # If the primary storage for this cluster is 'hdfs' than use 'hdfs://...
'
    # If the primary storage for this action is cloud storage, use the
    #    cloud storage prefix. IE: s3a://my_bucket
    hcfsNamespace: "hdfs://hdp_test:8020"
    hiveServer2:
      # URI is the Hive JDBC URL in the form of:
      # jdbc:hive2://<server>:<port>
      # See docs for restrictions
      uri: "jdbc:hive2://<Hive Server2 Host>:10000"
      connectionProperties:
        user: "hive"
        password: "hive"
      # Standalone jar file used to connect via JDBC to the LEFT environment
 Hive Server 2
      # NOTE: Hive 3 jars will NOT work against Hive 1.  The protocol isn't
compatible.
      jarFile: "/root/.hms-mirror/aux_libs/hive-jdbc-3.1.0.3.1.5.6091-7-stan
dalone.jar"
  RIGHT:
    legacyHive: false
    # Is the 'Hadoop COMPATIBLE File System' used to prefix data locations
 for this cluster.
    # It is mainly used to as a baseline for where "DATA" will be transfered
 in the
```

Testing the YAML and the cluster connection

It makes sense to test the YAML and cluster connection before starting the migration using HMS Mirror. Connection troubleshooting is easier before, rather than after, running HMS Mirror.

Procedure

1. Using a Hive client, such as DBeaver, beeline, or Tableau, connect the left cluster to the right cluster, and then vice versa.
2. If you use ZooKeeper Service Discovery on the HDP source cluster, use a plain URL in your connection string. For example:jdbc:hive2://<hive-server2-hostname>:10000/

HMS Mirror command summary

There are several critical options you need to use with the hms-mirror command to migrate HDP to CDP migration.

Run the following HMS Mirror help command, or go to the Cloudera HMS Mirror github site, to see a description of all command options:

```
hms-mirror -h
```

For the HDP to CDP migration, you generally use the subset of options shown in the following command:

```
hms-mirror -db <database name> -ma -mnn -e
```

-asm

Migrate avro based schema tables.

-d,--data-strategy <strategy>

Specify how the data will follow the schema. [SCHEMA_ONLY, LINKED, SQL, EXPORT_IMPORT, HYBRID, CONVERT_LINKED, STORAGE_MIGRATION, COMMON]

-da, -downgrade-acid

Downgrade ACID tables to EXTERNAL tables with purge.

-db [<db1,db2,..>]

Comma separated list of databases (up to 100) where dbn is the database name. The database name is optional, only required if the database value is null in the YAML configuration file.

-dc, --distcp

Generate the distcp commands.

-e, --execute

Executes HMS Mirror. Omit for a dry run.

--hadoop-classpath

Required for migrating from a Kerberos-enabled HDP cluster. For more information, see HMS Mirror documentation.

-ma | --migrate-acid

Manage managed tables, translating source metadata to the destination.

-mao

Migrate only acid tables.

-mnn

Manage non-native tables, such as HBase/Kafka/JDBC backed hive tables, translating source metadata to the destination.

-mnno

Migrate only non-native tables.

-o <output directory>

Output directory location.

-s, --sync

Drop metadata on the destination CDP One cluster if it does not match the definition on the HDP source cluster. (Ideally, there should not be any data on the destination cluster before metadata migration.)

--setup

Creates a default.yaml file in the $HOME/.hms-mirror/cfg/ directory by prompting you for information about your migration. You customize the YAML manually for your migration.

-v | --views-only

Migrates only VIEWs between two environments. This option alone must be run to separately generate views.

You can run any combination of options, except -v, which only migrates views. Run HMS Mirror to migrate views separately. VIEW creation requires dependent tables to exist. Run hms-mirror to create all the target tables before running it with the -v option.

This flag is an OR for processing views OR tables. They are NOT processed together.

Requirements:

- The dependent tables must exist in the RIGHT cluster
- When using -dbp|--db-prefix option, VIEW definitions are not modified and will most likely cause VIEW creation to fail.

Migrating Hive metadata

You first perform a dry run of HMS Mirror, and check the output. You see how to execute HMS Mirror to migrate the Hive metadata in HMS on the HDP cluster to the CDP as a Service cluster.

About this task

The dry run of HMS Mirror generates scripts, action files, reports, and a blueprint of what migration will take place.

Tip: To bypass Ranger security checks, which speeds up the migration, in Cloudera Manager, set ranger.plugin.hive.urlauth.filesystem.schemes=file in Cloudera Manager HiveServer (HS2) for Hive on TEZ.

In this procedure, you run the HMS Mirror command as shown in the step 6. In summary, use the Migrate ACID flag option -ma to migrate Hive ACID tables. Alternatively, you can migrate the data by specifying the HYBRID data strategy. An HMS Mirror command that sets the intermediate storage flag stages data in S3, as required for transfers in the public cloud. The target CDP SaaS environment uses the S3 object store in the cloud instead of the on-prem HDFS because the target does not have direct access to the on-prem HDFS.

Before you begin

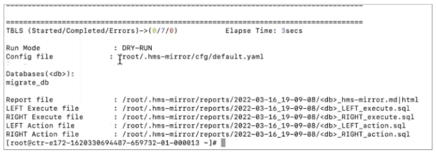
- If you are going to migrate non-native Hive tables, such as HBase, Kafka, or JDBC-backed tables, make these services available on the CDP One cluster; otherwise metadata migration does not work.
- If you have many databases to migrate, separate them into batches of 100 databases, or fewer. HMS Mirror does not accept a larger batch.
- A batching strategy is recommended. Inspect the number of tables/partitions within each database and develop a general criterion for batching.

Procedure

1. Stop all workloads on the source HDP and destination CDP clusters.
2. Run HMS Mirror to perform a dry run by omitting the -e option.

```
hms-mirror -db migrate_db -mnn
```

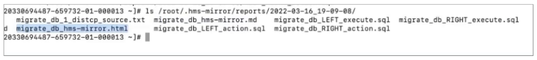
The output looks something like this:



```
--------------------------------------------------------------------------------

================================================================================
TBLS (Started/Completed/Errors)->(0/7/0)            Elapse Time: 3secs

Run Mode                  : DRY-RUN
Config file              : /root/.hms-mirror/cfg/default.yaml

Databases(<db>):
migrate_db

Report file               : /root/.hms-mirror/reports/2022-03-16_19-09-08/<db>_hms-mirror.md|html
LEFT Execute file         : /root/.hms-mirror/reports/2022-03-16_19-09-08/<db>_LEFT_execute.sql
RIGHT Execute file        : /root/.hms-mirror/reports/2022-03-16_19-09-08/<db>_RIGHT_execute.sql
LEFT Action file          : /root/.hms-mirror/reports/2022-03-16_19-09-08/<db>_LEFT_action.sql
RIGHT Action file         : /root/.hms-mirror/reports/2022-03-16_19-09-08/<db>_RIGHT_action.sql
[root@ctr-e172-1620330694487-659732-01-000013 ~]#
```

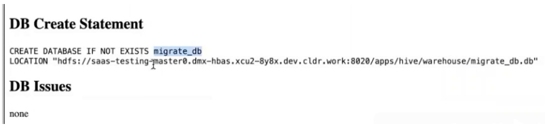
3. List the generated reports.
For example,

```
ls /root/.hms-mirror/reports/2022-03-16_19-09-08/
```



```
20330694487-659732-01-000013 ~]# ls /root/.hms-mirror/reports/2022-03-16_19-09-08/
   migrate_db_1_distcp_source.txt  migrate_db_hms-mirror.md    migrate_db_LEFT_execute.sql  migrate_db_RIGHT_execute.sql
d  migrate_db_hms-mirror.html      migrate_db_LEFT_action.sql  migrate_db_RIGHT_action.sql
20330694487-659732-01-000013 ~]#
```

4. Open the HMS Mirror report in the migrate_db_hms-mirror.html file, and scroll down to see the DB Create Statement.



DB Create Statement

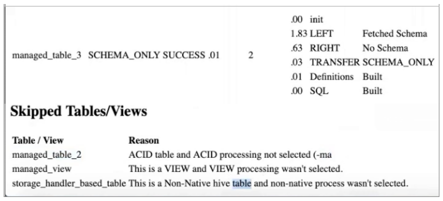
```
CREATE DATABASE IF NOT EXISTS migrate_db
LOCATION "hdfs://saas-testing-master0.dmx-hbas.xcu2-8y8x.dev.cldr.work:8020/apps/hive/warehouse/migrate_db.db"
```

DB Issues

none

Reports follow the DB Create Statement.

5. Scroll down to Skipped Tables/Views.



managed_table_3	SCHEMA_ONLY	SUCCESS	.01	2	.00	init
					1.83	LEFT Fetched Schema
					.63	RIGHT No Schema
					.03	TRANSFER SCHEMA_ONLY
					.01	Definitions Built
					.00	SQL Built

Skipped Tables/Views

Table / View	Reason
managed_table_2	ACID table and ACID processing not selected (-ma
managed_view	This is a VIEW and VIEW processing wasn't selected.
storage_handler_based_table	This is a Non-Native hive table and non-native process wasn't selected.

Your managed tables will not be listed in Skipped Tables/Views, if you omit the -ma option, for example.

6. Run the HMS Mirror command to execute the migration (-e option) of the migrate_db database. For example:

- Migrating standard

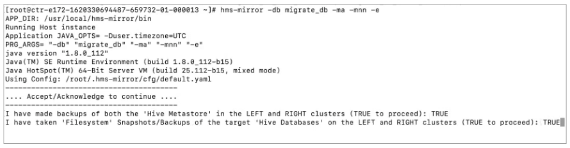
  hms-mirror -db migrate_db -e
- Migrating ACID tables

  ```
  hms-mirror -db migrate_db -ma -is s3a://intermediate_storage_location -e
   -d HYBRID
  ```
- Migrating non-native tables (HBase, Kafka, JDBC)

  ```
  hms-mirror -db migrate_db -mnn -e
  ```

Note: The -d HYBRID migrates the data). The intermediate data is staged in S3 via hms-mirror, required in a public cloud transfer scenario because the target CDP SaaS env does not have direct access to the on-prem HDFS).

7. At the prompt to confirm that you backed up the Hive Metastore (HMS) in both clusters, enter TRUE.



```
[root@ctr-e172-1620330694487-659732-01-000013 ~]# hms-mirror -db migrate_db -ma -mnn -e
APP_DIR: /usr/local/hms-mirror/bin
Running Host instance
Application JAVA_OPTS= -Duser.timezone=UTC
PRG_ARGS= "-db" "migrate_db" "-ma" "-mnn" "-e"
java version "1.8.0_112"
Java(TM) SE Runtime Environment (build 1.8.0_112-b15)
Java HotSpot(TM) 64-Bit Server VM (build 25.112-b15, mixed mode)
Using Config: /root/.hms-mirror/cfg/default.yaml
----------------------------------------
.... Accept/Acknowledge to continue ....
----------------------------------------
I have made backups of both the 'Hive Metastore' in the LEFT and RIGHT clusters (TRUE to proceed): TRUE
I have taken 'Filesystem' Snapshots/Backups of the target 'Hive Databases' on the LEFT and RIGHT clusters (TRUE to proceed): TRUE
```

8. At the prompt to confirm that you configured the file system TRASH, enter TRUE.

HMS Mirror generates reports in the directory shown in the output.

The metadata migration process completes.

HMS Mirror generated files

You use generated scripts, reports, and other files as you migrate the table metadata and data. Reports generated by HMS Mirror include information about HMS Mirror configurations. A list of tables on left HDP and translated tables on right CDP cluster are also generated.

The following files are generated by a dry run of HMS Mirror:

- <dbname>_1_distcp_source.txt

  A list of source cluster table file locations.
- <dbname>_LEFT_execute.sql

  A list of commands for creation of databases and tables under the database for the HDP cluster.
- <dbname>_LEFT_action.sql

  Actions that may be required on the HDP cluster to achieve the optimum state for migration.
- <dbname>_RIGHT_execute.sql

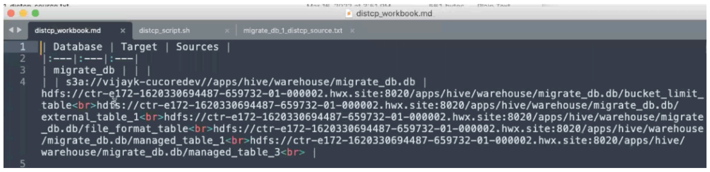
  A list of commands for creation of databases and tables under the database for the CDP SaaS cluster.
- <dbname>_RIGHT_action.sql

  Actions that may be required on the CDP SaaS cluster to achieve the optimum state for migration.
- <dbname>_hms-mirror.md Options you use for the hms-mirror command, config yaml, database and tables present on left cluster, translated tables on right cluster in an md file. You can use any markdown file viewer to view this file.
- <dbname>_hms-mirror.html — This is the html version of the above md file.
- distcp_script.sh

  The distcp commands needed to migrate actual table data from HDP to S3 location. Moving HDFS data from the HDP to the S3 object store for the CDP cluster is fully documented in Migrating data from HDP to CDP One on page 40.

- distcp_workbook.md

  Shows the HDP source and CDP One destination locations that we need to migrate to.

  ```
  | Database | Target | Sources |
  |:---|:---|:---|
  | migrate_db | | |
  | | s3a://vijayk-cucoredev//apps/hive/warehouse/migrate_db.db | hdfs://ctr-e172-1620330694487-659732-01-000002.hwx.site:8020/apps/hive/warehouse/migrate_db.db/bucket_limit_table<br>hdfs://ctr-e172-1620330694487-659732-01-000002.hwx.site:8020/apps/hive/warehouse/migrate_db.db/external_table_1<br>hdfs://ctr-e172-1620330694487-659732-01-000002.hwx.site:8020/apps/hive/warehouse/migrate_db.db/file_format_table<br>hdfs://ctr-e172-1620330694487-659732-01-000002.hwx.site:8020/apps/hive/warehouse/migrate_db.db/managed_table_1<br>hdfs://ctr-e172-1620330694487-659732-01-000002.hwx.site:8020/apps/hive/warehouse/migrate_db.db/managed_table_3<br> |
  ```

  This information can help you build a transfer job to specify multiple sources using distcp -f.

Verifying metadata migration

After running the HMS Mirror tool to migrate Hive metadata from HDP to CDP One, you take a look at the migrated metadata on the CDP One cluster.

About this task

Procedure

1. From a JDBC client, such as DBeaver, create a connection to the CDP cluster.
2. Generate a passcode token to connect to the CDP cluster.
3. Run SQL queries and check the output to verify that the databases and tables are migrated.
   For example:

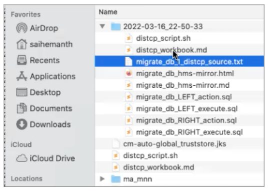
   ```
   SHOW DATABASES;
   USE migrate_db;
   SHOW TABLES;
   ```

Migrating actual Hive data

Although HMS Mirror migrates only Hive metastore (HMS) metadata, not the actual Hive data, the tool generates a script for running the Hadoop DistCp tool. Using DistCp, you migrate actual data from HDP to CDP One.

About this task

DistCp is fully documented in HDP to CDP SaaS HDFS Migration. When you run the HMS Mirror tool, you generated a number of distcp files:



The distcp_script.sh contains a template and instructions for running the script to migrate the actual Hive data. The template looks something like this:

```
hadoop distcp ${DISTCP_OPTS} -f ${HCFS_BASE_DIR}/migrate_db_1_distcp_source.
txt s3a://myserver//apps/hive/warehouse/migrate_db.db
```

In this task, you customize the DistCp command to substitute values specific to your migration for the variables above.

Before you begin

Give the user who is migrating the data permissions to read/write/execute on both the HDP and CDP clusters.

Procedure

1. From the information in the distcp_workbook.md file, get the names of source files from the Sources column to migrate from the <dbname>_1_distcp_source.txt
2. On the source HDP cluster, export the environment variable HCFS_BASE_DIR that represents the path to the source files.

   ${HCFS_BASE_DIR} must be available to the user running DistCp.
3. Export the DISTCP_OPTS environment variable to customize job settings.
   For example, you might adjust memory settings for large jobs.
4. Using information from steps 1-3 plus your S3 access key and password, customize the DistCp template.
5. On the HDP source cluster edge node, run the customized DistCp commands.
   For example:

```
hadoop distcp -Dfs.s3a.access.key=AKIAR2YKDFBDE64CJ5XG -Dfs.s3a.secret.k
ey=<mykey> hdfs://ctr-e172-1620330694487-659732-01-000002.hwx.site:8020/
apps/hive/warehouse/migrate_db.db/managed_table_1  s3a://myserver//apps/
hive/warehouse/migrate_db.db
hadoop distcp -Dfs.s3a.access.key=AKIAR2YKDFBDE64CJ5XG -Dfs.s3a.secret.
key=<mykey> hdfs://ctr-e172-1620330694487-659732-01-000002.hwx.site:8020/
apps/hive/warehouse/migrate_db.db/file_format_table  s3a://myserver//apps/
hive/warehouse/migrate_db.db
hadoop distcp -Dfs.s3a.access.key=AKIAR2YKDFBDE64CJ5XG -Dfs.s3a.secret.k
ey=<mykey> hdfs://ctr-e172-1620330694487-659732-01-000002.hwx.site:8020/
apps/hive/warehouse/migrate_db.db/external_table_1  s3a:/myserver//apps/
hive/warehouse/migrate_db.db
```

6. Check the status of the migration in HTML reports generated by DistCP to determine success or failure.
7. If an error occurs, check $HOME/.hms-mirror/logs/hms-mirror.log.

Adjust AVRO table schema URLs

If any of your Hive tables are in AVRO format, before migration already created the AVRO schema on the CDP object store. After migration, you must change the URLs in table properties to point to the CDP location.

Procedure

On the destination CDP cluster, correct the path of AVRO schema URL table property.

```
ALTER TABLE my_table SET TBLPROPERTIES('avro.schema.url'='S3a:///mybucket/my
path/my_table.avsc');
```

Verifying actual Hive data migration

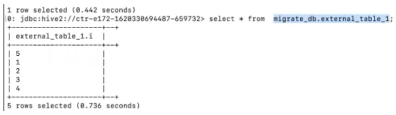
After running the DistCp to migrate Hive data from HDP to CDP One, you take a look at the migrated data on the CDP One cluster.

Procedure

1. From a JDBC client, such as DBeaver, create a connection to the CDP cluster.
2. Generate a passcode token to connect to the CDP cluster.

3. Run SQL SELECT queries and check the output to verify that the external table data is migrated.
   For example:

   ```
   SELECT * FROM migrate_db.external_table_1;
   ```

   ```
   1 row selected (0.442 seconds)
   0: jdbc:hive2://ctr-e172-1620330694487-659732> select * from  migrate_db.external_table_1;
   +----------------------+--+
   | external_table_1.i   |
   +----------------------+--+
   | 5                    |
   | 1                    |
   | 2                    |
   | 3                    |
   | 4                    |
   +----------------------+--+
   5 rows selected (0.736 seconds)
   ```

4. Run SELECT queries and check the output to verify that managed data is migrated.

   ```
   SELECT * FROM migrate_db.managed_table_3;
   ```

Table locations

After migration from HDP 2.x, the relative paths for databases, tables, and partitions on the destination CDP One cluster are different from HDP 2.x.

- HDP 2.x managed table location: /apps/hive/warehouse
- HDP 2.x external table location: /user/hive/warehouse”

After migration from HDP 3.x, the relative paths for databases, tables, partitions on the destination CDP One cluster are the same as the HDP 3.x:

- HDP 3.x and CDP One managed table location: /warehouse/tablespace/managed/hive
- HDP 3.x and CDP One external table location: /warehouse/tablespace/external/hive

Fixing statistics

Upgrading or migrating from Hive 1 or Hive 2 to Hive 3 might result in missing statistics. In Hive 3, these missing statistics, when detected by the cost-based optimizer (CBO), could cause datasets to be disregarded. You need to fix these statistics after upgrading.

Procedure

1. Run DESCRIBE FORMATTED <table>, and check the value of numrows.
   If the value is 0, you must fix statistics.
2. Run ANALYZE on the tables and columns to fix the statistics.

   ```
   ANALYZE TABLE credit_card_01.cc_acct COMPUTE STATISTICS[FOR COLUMNS];
   ```

Changes to HDP Hive tables

You need to locate and use your Apache Hive 3 tables after moving to CDP. You also need to understand the changes that occur during the process.

Managed, ACID tables that are not owned by the hive user remain managed tables in CDP, but hive becomes the owner.

In CDP, the format of a Hive table is the same as it was in HDP. For example, native or non-native tables remain native or non-native, respectively.

In CDP, the location of managed tables or partitions do not change under any one of the following conditions:

- The old HDP 2.x table or partition directory was not in its default location /apps/hive/warehouse before the upgrade.
- The old table or partition is in a different file system than the new warehouse directory.
- The old table or partition directory is in a different encryption zone than the new warehouse directory.

Otherwise, in CDP managed files reside in the Hive warehouse /warehouse/tablespace/managed/hive. The location of external files in CDP does not change. By default, Hive places any new external tables you create in /warehouse/tablespace/external/hive. In CDP, the hive.metastore.warehouse.dir property is set to this location, designating it the Hive warehouse location.

Changes to table references using dot notation

Upgrading to CDP includes the Hive-16907 bug fix, which rejects `db.table` in SQL queries. The dot (.) is not allowed in table names. To reference the database and table in a table name, both must be enclosed in backticks as follows: `db`.`table`.

Changes to ACID properties

Hive 3.x supports transactional and non-transactional tables. Transactional tables have atomic, consistent, isolation, and durable (ACID) properties. In Hive 2.x, the initial version of ACID transaction processing was ACID v1. In Hive 3.x, the mature version of ACID is ACID v2, which is the default table type in CDP.

Native and non-native storage formats

Storage formats are a factor in changes to table types. Hive 2.x and 3.x support the following native and non-native storage formats:

- Native: Tables with built-in support in Hive, such as those in the following file formats:
  - Text
  - Sequence File
  - RC File
  - AVRO File
  - ORC File
  - Parquet File
- Non-native: Tables that use a storage handler, such as the DruidStorageHandler or HBaseStorageHandler

Changes to HDP table types

The following table compares Hive table types and ACID operations from HDP 2.x to CDP. The ownership of the Hive table file is a factor in determining table types and ACID operations in CDP.

Table 1: HDP 2.x and CDP Table Type Comparison

HDP 2.x				CDP	
Table Type	**ACID v1**	**Format**	**Owner (user) of Hive Table File**	**Table Type**	**ACID v2**
External	No	Native or non-native	hive or non-hive	External	No
Managed	Yes	ORC	hive or non-hive	Managed, updatable	Yes
Managed	No	ORC	hive	Managed, updatable	Yes
			non-hive	External, with data delete	No
Managed	No	Native (but non-ORC)	hive	Managed, insert only	Yes
			non-hive	External, with data delete	No
Managed	No	Non-native	hive or non-hive	External, with data delete	No