Cloudera Runtime 7.3.2

Configuring Streams Replication Manager

Date published: 2019-08-22
Date modified: 2026-03-31



https://docs.cloudera.com/

Legal Notice

© Cloudera Inc. 2026. All rights reserved.

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

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

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

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

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

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

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

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

Contents

Add Streams Replication Manager to an existing cluster..... 5

Enable high availability for Streams Replication Manager..... 8

Enabling prefixless replication..... 9

Enabling exactly-once semantics for replication flows..... 10

Defining and adding clusters for replication..... 11
Defining external Kafka clusters..... 13
Defining co-located Kafka clusters using a service dependency..... 17
Defining co-located Kafka clusters using Kafka credentials..... 17
Adding clusters to Streams Replication Manager's configuration..... 21

Configuring replications..... 22

Configuring the driver role target clusters..... 23

Configuring the service role target cluster..... 24

Configuring properties not exposed in Cloudera Manager..... 24

Suppressing internal metrics topics..... 25

Configuring replication specific Kafka Connect REST servers..... 26

Configuring Remote Querying..... 27
Enabling Remote Querying..... 27
Configuring the advertised information of the Streams Replication Manager Service role..... 29

Configuring Streams Replication Manager Driver retry behaviour..... 29

Configuring Streams Replication Manager Driver heartbeat emission..... 30

Configuring automatic group offset synchronization..... 32

Configuring Streams Replication Manager Driver for performance tuning.....33

Hardware..... 33
Task count.....33
Consumer and producer configurations..... 34
Connect worker configurations..... 35
Metric calculation..... 36

New topic and consumer group discovery..... 36

Configuration examples..... 37

Bidirectional replication example of two active clusters.....37
Cross data center replication example of multiple clusters..... 39

Enabling reverse checkpointing..... 43

Add Streams Replication Manager to an existing cluster

Streams Replication Manager can be installed on an existing cluster managed by Cloudera Manager. To do this, you need to define your external clusters using Kafka credentials, add Streams Replication Manager to the cluster, and configure a number of mandatory properties related to clusters, replications, and role targets.

About this task

Streams Replication Manager is comprised of three roles:

- Streams Replication Manager Driver role: This role is responsible for connecting to the specified clusters and performing replication between them. The driver can be installed on one or more hosts.
- Streams Replication Manager Service role: This role consist of a REST API and a Kafka Streams application to aggregate and expose cluster, topic and consumer group metrics. The service can be installed on or more hosts.
- Streams Replication Manager Gateway role: This role can be used to deploy Streams Replication Manager client configuration on hosts that do not have a Driver or Service role running on them.



Note: Do not confuse Streams Replication Manager, which is a service managed by Cloudera Manager, with the Streams Replication Manager Service role, which is a role within Streams Replication Manager.

You can install Streams Replication Manager independent of the clusters that replication is happening between.

The following steps walk you through the process of adding Streams Replication Manager to your cluster. The configuration examples on this page are simple examples that are meant to demonstrate the type of information that you have to enter. These steps and examples do not go into detail of how security is configured or present all configuration options. For additional information regarding the steps presented in this document, Streams Replication Manager security, and Streams Replication Manager configuration in general, see the *Related information* section at the bottom of this page

Before you begin

- If you are planning on replicating data to or from a Kafka service running in either a CDH 5.x or 6.x cluster and you are using Sentry for authorization, make sure that the streamsrepmgr user is added to the Kafka Super users property. You can find the Super users property by going to Kafka service Configuration . Do this on all CDH 5.x or 6.x clusters where data replication will happen.
- If you are planning on replicating data to or from a Kafka service running in Runtime 7.x and you are using Ranger for authorization, make sure that the streamsrepmgr user has all required permissions assigned to it in Ranger. Do this on all Runtime 7.x clusters where data replication will happen.

Procedure

1. In Cloudera Manager, go to Administration External Accounts .
2. Go to the Kafka Credentials tab.

   The Kafka Credentials tab enables you to create, configure, and manage Kafka credentials. A Kafka credential is an item that contains the connection properties required by Streams Replication Manager to establish a connection with a cluster. You can think of a Kafka credential as the definition of a single cluster. On this tab you will create a credential for each external cluster taking part in the replication process.
3. Create and configure Kafka credentials for the external clusters:

   Repeat this step for all external clusters.

   a) Click Add Kafka credentials.
   b) Configure the Kafka credential:

      The security configuration of the external cluster determines which of the available properties you must set. At minimum, the following properties must be set:

- Name
- Bootstrap servers
- Security protocol

For example, assume that you have two clusters US-West and US-East. The Streams Replication Manager service you are configuring is located on US-West. This means that, US-East is an external cluster and US-West is co-located. In a case like this you would have to specify US-East with a Kafka credential.

```
Name=useast
Bootstrap servers=useast-1.cloudera.com:9092, useast-2.cloudera.com:9092
, useast-3.cloudera.com:9092
Security protocol=PLAINTEXT
```

Note: Click the i icon next to each property in Cloudera Manager to reveal additional information about each property.

c) Click Add.

If credential creation is successful, a new entry corresponding to the Kafka credential you specified appears on the page.

4. From the Cloudera Manager Home page, select the drop-down to the right of your cluster, and select Add a Service.
5. Select Streams Replication Manager from the list of services and click Continue.
6. Select dependencies

Select the option that has the Kafka service listed as an optional dependency.

7. Assign role instances to hosts:

Select at least 2 hosts for both the Driver and the Service role if you want to enable high availability mode for Streams Replication Manager.

a) Click the field below SRM Driver to display a dialog box containing a list of hosts.
b) Select 1 or more hosts that the SRM Driver should be assigned to and click Ok.
c) Click the field below SRM Service to display a dialog box containing a list of hosts.
d) Select 1 or more host that the SRM Service should be assigned to and click Ok.
e) Click the field below SRM Gateway to display a dialog box containing a list of hosts.
f) Select 1 or more hosts that the SRM Gateway role should be assigned to and click Ok.

8. Click Continue.
9. Add the clusters defined with Kafka credentials to Streams Replication Manager's configuration:

a) Find and configure the External Kafka Accounts property.
b) Click the add button and add new lines for each Kafka credential you created.
c) Add the names of all Kafka credentials.

Each Kafka credential must be added to a new line. For example:

```
useast
```

10. Find and configure the Streams Replication Manager Co-located Kafka Cluster Alias property.

The alias you configure represents the co-located cluster. Enter an alias that is unique and easily identifiable. For example:

```
uswest
```

11. Find and configure the Streams Replication Manager Cluster alias property.

Add all cluster aliases. This includes aliases present in the External Kafka Accounts and Streams Replication Manager Co-located Kafka Cluster Alias properties. Delimit the aliases with commas. For example:

```
useast, uswest
```

12. Add and enable replications:

a) Find the Streams Replication Manager's Replication Configs property.

b) Click the add button and add new lines for each unique replication you want to add and enable.

c) Add and enable your replications.

For example:

```
uswest->useast.enabled=true
useast->uswest.enabled=true
```

13. Specify the Streams Replication Manager Service role target cluster:

a) Find the Streams Replication Manager Service Target Cluster property.

b) Add the target cluster alias. For example:

```
uswest
```

The target cluster is where the Service role gathers replication information from. Cloudera recommends that you deploy the Service role on every cluster and configure each instance of the Service role to target the cluster that it is running on.

14. Enable prefixless replication by selecting the Enable Prefixless Replication property.

The default replication policy used by Streams Replication Manager is the `DefaultReplicationPolicy`. This replication policy renames remote (replicated) topics on target clusters and adds the source cluster alias as a prefix. For example, useast.topic1.

If you do not want your topics to get renamed, you can choose to use `IdentityReplicationPolicy` instead, which does not rename topics during replication. This type of replication is referred to as prefixless replication. The `IdentityReplicationPolicy` has a number of limitations, see *Replication flows and replication policies* or *Enabling prefixless replication* for more information.

15. Specify the Streams Replication Manager Driver role target clusters:

a) Find the Streams Replication Manager Driver Target Cluster property.

b) Add the cluster aliases that you want the driver role to target. For example:

```
uswest,useast
```

You can use the Streams Replication Manager Driver Target Cluster property to specify a subset of clusters that the driver should target or in other words write data to. When this property is left empty (default) the driver will read from and write to all clusters added to Streams Replication Manager’s configuration. When this property is set, the driver will collect data from all clusters, but will only write to the clusters specified in this property. However, in order for monitoring to function correctly, this property has to contain the target as well as source clusters. As a result, custom configuration of this property is considered an advanced configuration practice, which is only viable in complex replication scenarios. Cloudera recommends that you either leave this property empty or add all clusters taking part in the replication.

16. Configure properties not exposed in Cloudera Manager:

Streams Replication Manager accepts a number of additional configuration properties that are not available in Cloudera Manager. Depending on your requirements you may need to configure these properties as well. You can find a comprehensive list of these properties in Configuration Properties Reference for Properties not Available in Cloudera Manager.

a) Find the Streams Replication Manager's Replication Configs property.

b) Click the add button and add new lines for each additional property you want to configure.

c) Add configuration properties

For example:

```
offset.flush.timeout.ms=30000
```

17. Depending on your requirement, review and configure other properties available on this page.

18. Click Continue and wait until the process is finished
19. Click Continue then click Finish.
20. Start Streams Replication Manager:
    a) On the Cloudera Manager Home page, select the drop-down to the right of Streams Replication Manager, and select Start.
    b) Click Startand wait until the process is finished.
    c) Click Close.

Results

- Replicating data to or from the specified clusters is now possible.
- The Streams Replication Manager service REST API Swagger UI is available at one of the following addresses:
  - `http://<srm-service-host>:<srm-service-port/swagger`
  - `https://<srm-service-host>:<srm-service-port/swagger`

What to do next

- Use the srm-control tool to kick off replication by adding topics or groups to the allowlist.
- If you plan to use Streams Messaging Manager to monitor Kafka cluster replications, configure SMM to communicate with Streams Replication Manager. For information, see *Configuring SMM for Monitoring Streams Replication Manager Replications*.

Related Information

Configuration examples
srm-control
Configuration Properties Reference for Properties not Available in Cloudera Manager
Defining and adding clusters for replication
Configuring the driver role target clusters
Configuring the service role target cluster
Configuring properties not exposed in Cloudera Manager
Configuring replications
Replication flows and replication policies
Enabling prefixless replication
Integrating Streams Replication Manager with Streams Messaging Manager

Enable high availability for Streams Replication Manager

Streams Replication Manager is capable of running in high availability mode. This can be enabled by deploying multiple instances of the driver and service role in a cluster.

Streams Replication Manager is capable of running in high availability mode. By enabling high availability mode for Streams Replication Manager, you can ensure that the replication of data and the monitoring of replication continues even in the case of host failure. To enable Streams Replication Manager high availability, you must deploy multiple instances of the driver and service roles on the hosts in a cluster.

In Cloudera Base on premises this can be done when installing a new cluster, when adding Streams Replication Manager to an existing cluster, or when Streams Replication Manager is already deployed and running on a cluster.



Note: Expect an increased load when running Streams Replication Manager in high availability mode.

Enable high availability for Streams Replication Manager during cluster installation

To enable high availability for Streams Replication Manager during cluster installation, follow the cluster installation instructions available in Cloudera Base on premises Installation Guide . During Step 7: Set Up a Cluster Using the Wizard, when prompted to assign roles to hosts, assign multiple role instances of the Streams Replication Manager driver and service roles to your hosts.

Enable high availability when adding Streams Replication Manager to an existing cluster

To enable high availability when adding Streams Replication Manager to an existing cluster, follow the service installation instructions available in Add Streams Replication Manager to an existing cluster. When assigning role instances to hosts, assign multiple role instances of the Streams Replication Manager driver and service roles to your hosts.

Enable high availability for an already running instance of Streams Replication Manager

To enable high availability for an already running instance of Streams Replication Manager, you must add additional driver and service role instances to your hosts. For more information, see Adding a Role Instance in the Cloudera Manager documentation.

Related Information

Streams Replication Manager Driver

Task architecture and load-balancing

Enabling prefixless replication

Learn how to enable prefixless replication with the IdentityReplicationPolicy. If this replication policy is in use, remote topics on target clusters retain their original name and are not prefixed.

About this task

The default replication policy used by Streams Replication Manager is the `DefaultReplicationPolicy`. This replication policy renames remote (replicated) topics on target clusters and adds the source cluster alias as a prefix. For example, useast.topic1. If you do not want your topics to get renamed, you can choose to use `IdentityReplicationPolicy` instead, which does not rename topics during replication. This type of replication is referred to as prefixless replication.

You can configure Streams Replication Manager to use the `IdentityReplicationPolicy` by selecting the Enable Prefixless Replication Streams Replication Manager service property in Cloudera Manager.

Before you begin

- Ensure that the Remote Topics Discovery With Internal Topic property remains selected after you enable prefixless replication. This property must be selected for prefixless replication to function. This property is selected by default and is also automatically selected when you enable prefixless replication with Enable Prefixless Replication.
- Do not use replication.policy.class to set `IdentityReplicationPolicy` as the replication policy. Use Enable Prefixless Replication instead.
- Using a mix of replications policies in a deployment with multiple instances of the Streams Replication Manager service is not supported.
- Transitioning an already running Streams Replication Manager service from the `DefaultReplicationPolicy` or any other replication policy to the `IdentityReplicationPolicy` is not recommended.
- The `IdentityReplicationPolicy` has the following limitations:
  - Replication loop detection is not supported. As a result, you must ensure that topics are **not** replicated in a loop between your source and target clusters.

- The /v2/topic-metrics/{target}/{downstreamTopic}/{metric} endpoint of Streams Replication Manager Service v2 API does not work properly with prefixless replication. Use the /v2/topic-metrics/{source}/{target}/{upstreamTopic}/{metric} endpoint instead.
- The replication metric graphs shown on the **Topic Details** page of the Streams Messaging Manager UI do not work with prefixless replication.
- Reverse checkpointing is not compatible with the `IdentityReplicationPolicy` because the feature relies on cluster prefixes to identify replica topics and correctly transform replica topic names back to their original counterparts.

As a result of these limitations, Cloudera recommends that you use the `DefaultReplicationPolicy` whenever possible.

Procedure

1. In Cloudera Manager, select the Streams Replication Manager service.
2. Go to Configuration.
3. Find and select the Enable Prefixless Replication property.
4. Click Save Changes.
5. Restart the service.

Results

Streams Replication Manager is configured to use the `IdentityReplicationPolicy`. Remote topics on target clusters are no longer prefixed with the source cluster alias.

Enabling exactly-once semantics for replication flows

You can enable exactly-once semantics (EOS) for replication flows in Streams Replication Manager to ensure that transactions are replicated without duplication.

About this task

The progress of `MirrorSourceConnector` is tracked by periodically committing the offsets of the processed messages. If the connector fails, uncommitted messages are reprocessed after the connector starts running again.

Using EOS, source connectors are able to handle offset commits and message produces in a single transaction. This either results in a successful operation where messages are produced to the target topic along with offset commits, or a rollback of the whole operation. EOS is enabled in Streams Replication Manager with the exactly.once.source.support property.

If transactional producers are writing messages to the source topic, Cloudera recommends that you filter records from the aborted transactions out from the replicated data. Otherwise, aborted transactions are marked as committed in the target, which results in invalid data. This is configured in your `MirrorSourceConnector` with the isolation.level property. You set the property to read_committed.



Important: Due to the periodic nature of checkpointing, EOS does not apply to failover and failback scenarios. Duplicate messages are expected.

Procedure

1. Enable EOS in Cloudera Manager.
   a) In Cloudera Manager, select the SRM service and go to Configuration.
   b) Find the Streams Replication Manager's Replication Configs property.
   c) Add the exactly.once.source.support property.

   Configuration differs for newly deployed resources and existing resources.

- For new deployments, set the value to enabled.
- For existing deployments, you must follow a two-step process to avoid downtime:
  1. Set the value to preparing.
  2. Click Save Changes.
  3. Restart the Streams Replication Manager service.
  4. Wait for the replication to stabilize.
  5. Update the value to enabled.

2. Set the consumer isolation level.
   a) In the same Streams Replication Manager's Replication Configs property, configure the isolation level.
   b) To set it globally for all connectors, add:

```
connectors.consumer.isolation.level=read_committed
```

   c) Alternatively, to set it for a specific replication flow, add:

```
[SOURCE]->[TARGET].consumer.isolation.level=read_committed
```

   Replace [SOURCE] and [TARGET] with the cluster aliases.

   Note:

   Setting the isolation.level comes with caveats. If the connector reaches a message written by an uncommitted transaction, it stops reading until the transaction is either committed or rolled back. This can cause significant lag in replication. You can limit this by applying an appropriate application timeout, however, the timeout you set will depend on the application and use case.

3. Click Save Changes.
4. Restart the Streams Replication Manager service.

Related Information

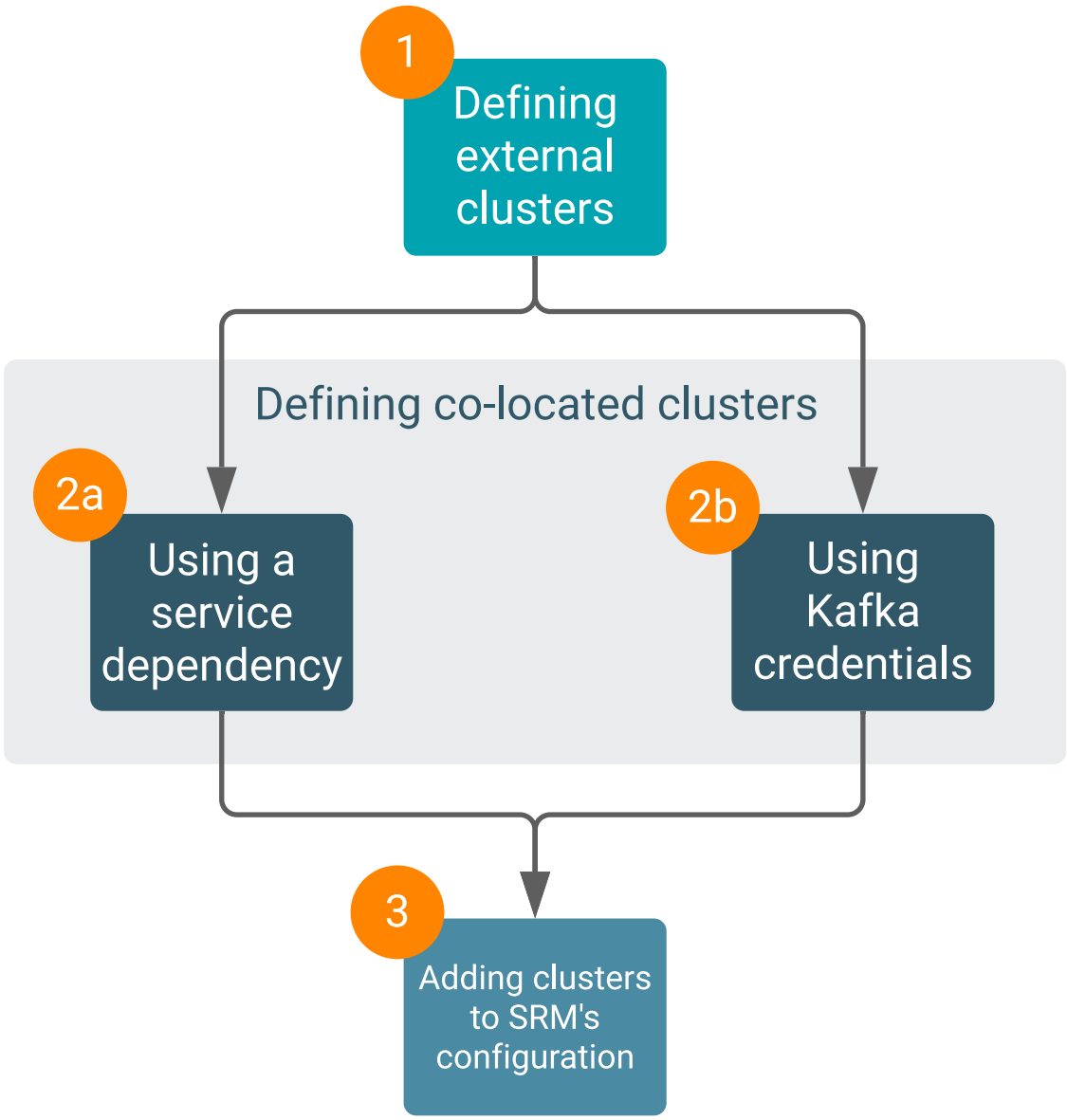
Replication flows and replication policies

Streams Replication Manager Architecture

Defining and adding clusters for replication

Before you can start replicating data with Streams Replication Manager, you must define the clusters that take part in the replication process and add them to Streams Replication Manager's configuration. Defining and adding the clusters provides Streams Replication Manager with the necessary connection and security information it needs to replicate data.

In order for Streams Replication Manager to replicate data between clusters, you have to define these clusters in Cloudera Manager and add them to the Streams Replication Manager service's configuration. In order to correctly configure your clusters, you need to complete multiple configuration tasks. The configuration workflow is the following:



1 Defining external clusters

In any given scenario, you start with defining your external clusters. External clusters are defined by creating and configuring Kafka credentials. A Kafka credential is an item that contains the connection properties required by Streams Replication Manager to establish a connection with a cluster. You can think of a Kafka credential as the definition of a single cluster. It contains the name (alias), address (bootstrap servers), and credentials that Streams Replication Manager can use to access a specific cluster.

Kafka credentials are created and configured in Cloudera Manager on the Administration External Accounts Kafka Credentials page. How you configure each Kafka credential depends on the security configuration of the cluster that it defines. Once a Kafka credential is created, the properties needed to access that cluster become available to Streams Replication Manager.

2 Defining co-located clusters

Co-located Kafka clusters can be defined in either of two ways. You can define these clusters using a service dependency or by using Kafka credentials. Cloudera recommends that you follow the

recommendations provided and choose the method based on the security configuration of the co-located Kafka cluster. Additionally, Cloudera recommends that you use the service dependency method whenever possible. The method you choose also impacts how you configure the srm-control tool.

Note: It is possible that your deployment does not have any co-located clusters, or if there are multiple instances of Streams Replication Manager, only some have a co-located cluster. If there is no co-located cluster, you do not need to define it.

2a Defining co-located clusters using a service dependency

The service dependency method works by enabling a service dependency between the Streams Replication Manager service and the co-located Kafka cluster, specifying an alias for the co-located cluster, and configuring security related properties.

Cloudera recommends that you use this method if:

- The co-located Kafka cluster is not secured.
- The co-located Kafka cluster uses Kerberos for authentication.

2b Defining co-located clusters using Kafka credentials

Kafka credentials can be used to define co-located clusters. The steps to define a co-located cluster with a Kafka credential are identical to the steps you take when defining external clusters.

Cloudera recommends that you use this method if the co-located Kafka cluster uses an authentication mechanism that is not Kerberos, for example PLAIN.

3 Adding clusters to Streams Replication Manager's configuration

Once both external and co-located clusters are defined, you must add all defined clusters to Streams Replication Manager's configuration. This is done by configuring various configuration properties in Cloudera Manager.

Continue reading for step-by-step instructions on how to complete each of the configuration tasks.

Defining external Kafka clusters

Before you can start replicating data with Streams Replication Manager, you must define the external Kafka clusters that take part in the replication process. This is done using Cloudera Manager by creating Kafka credentials.

About this task

The following list of steps walk you through how you can define the external Kafka clusters that Streams Replication Manager connects to.

Before you begin

- Ensure that you have reviewed Defining and adding clusters for replication and understand that the following list of steps are only one part of the full configuration workflow.
- When creating and configuring Kafka Credentials you will be presented with various configuration properties. A comprehensive list of these properties is provided in Kafka credentials property reference. Cloudera recommends having this resource open during configuration.

Procedure

1. In Cloudera Manager, go to Administration External Accounts .
2. Go to the Kafka Credentials tab.

   The Kafka Credentials tab enables you to create, configure, and manage Kafka credentials. On this tab you will create a credential for each external cluster taking part in the replication process.
3. Create and configure Kafka credentials for the external clusters:

Repeat this step for all external clusters.

a) Click Add Kafka credentials.
b) Configure the Kafka credentials:

The security configuration of the external cluster determines which of the available properties you must set. To see which exact properties are required for your scenario, click one of the following tabs matching the security protocol of the cluster you are defining:

For PLAINTEXT

- Name
- Bootstrap servers
- Security protocol

For SSL

Encryption only

- Name
- Bootstrap servers
- Security protocol
- Truststore Password
- Truststore Path
- Truststore Type

Encryption and authentication

- Name
- Bootstrap servers
- Security protocol
- Truststore Password
- Truststore Path
- Truststore Type
- Key Password
- Keystore Password
- Keystore Path
- Keystore Type

For SASL_PLAINTEXT

- Name
- Bootstrap servers
- Security protocol
- JAAS Secret [1-3]
- JAAS Template
- Kerberos Service Name
- SASL Mechanism

For SASL_SSL

- Name
- Bootstrap servers
- Security protocol
- Truststore Password
- Truststore Path
- Truststore Type
- JAAS Secret [1-3]

- JAAS Template
- Kerberos Service Name
- SASL Mechanism

Note: Click the i icon next to each property in Cloudera Manager to reveal additional information about each property.

c) Click Add.

If credential creation is successful, a new entry corresponding to the Kafka credential you specified appears on the page.

The following tabs collect examples of different Kafka credentials as well as some notes regarding configuration. Review these examples to better understand how to correctly configure a Kafka credential.

For PLAINTEXT

```
name=uswest
Bootstrap servers=uswest-node1.cluster.com:9092,uswest-node2.clu
ster.com:9092,uswest-node3.cluster.com:9092
Security protocol=PLAINTEXT
```

For SSL

Encryption only

```
Name=uswest
Bootstrap servers=uswest-node1.cluster.com:9093,uswest-node2.clu
ster.com:9093,uswest-node3.cluster.com:9093
Security protocol=SSL
Truststore Password=password
Truststore Path=/opt/srm/us-west-truststore.jks
Truststore Type=JKS
```

Encryption and authentication

```
Name=uswest
Bootstrap servers=uswest-node1.cluster.com:9093,uswest-node2.clu
ster.com:9093,uswest-node3.cluster.com:9093
Security protocol=SSL
Truststore Password=password
Truststore Path=/opt/srm/us-west-truststore.jks
Truststore Type=JKS
Keystore Password=password
Keystore Path=/opt/srm/us-west-keystore.jks
Keystore Type=JKS
Key Password=password
```

Note: The truststore and keystore files must be deployed on all Streams Replication Manager hosts and Streams Replication Manager must be able to access these files.

For SASL_PLAINTEXT

Kerberos

```
Name=uswest
Bootstrap servers=uswest-node1.cluster.com:9092,uswest-node2.clu
ster.com:9092,uswest-node3.cluster.com:9092
Security protocol=SASL_PLAINTEXT
JAAS Template=com.sun.security.auth.module.Krb5LoginModule requ
ired useKeyTab=true keyTab="/opt/srm/streamsrepmgr.keytab" princ
ipal="streamsrepmgr@REALM.COM";
Kerberos Service Name=kafka
```

```
SASL Mechanism=GSSAPI
```

PLAIN

```
Name=uswest
Bootstrap servers=uswest-node1.cluster.com:9092,uswest-node2.clu
ster.com:9092,uswest-node3.cluster.com:9092
Security protocol=SASL_PLAINTEXT
JAAS Secret 1=password
JAAS Template=org.apache.kafka.common.security.plain.PlainLoginM
odule required username="username" password="##JAAS_SECRET_1##";
SASL Mechanism=PLAIN
```



Note: The Kerberos principal must not include host names. Additionally, the keytab file must be deployed on all Streams Replication Manager hosts and Streams Replication Manager must be able to access this file.

For SASL_SSL

Kerberos

```
Name=uswest
Bootstrap servers=uswest-node1.cluster.com:9093,uswest-node2.clu
ster.com:9093,uswest-node3.cluster.com:9093
Security protocol=SASL_SSL
Truststore Password=password
Truststore Path=/opt/srm/us-west-truststore.jks
Truststore Type=JKS
JAAS Template=com.sun.security.auth.module.Krb5LoginModule requi
red useKeyTab=true keyTab="/opt/srm/streamsrepmgr.keytab" princi
pal="streamsrepmgr@REALM.COM";
Kerberos Service Name=kafka
SASL Mechanism=GSSAPI
```

PLAIN

```
Name=uswest
Bootstrap servers=uswest-node1.cluster.com:9093,uswest-node2.clu
ster.com:9093,uswest-node3.cluster.com:9093
Security protocol=SASL_SSL
Truststore Password=password
Truststore Path=/opt/srm/us-west-truststore.jks
Truststore Type=JKS
JAAS Secret 1=password
JAAS Template=org.apache.kafka.common.security.plain.PlainLoginMo
dule required username="username" password="##JAAS_SECRET_1##";
SASL Mechanism=PLAIN
```



Note: The Kerberos principal must not include host names. Additionally, the keytab and truststore files must be deployed on all Streams Replication Manager hosts and Streams Replication Manager must be able to access these files.

Results

The external Kafka clusters are defined using Kafka credentials.

What to do next

Define the co-located Kafka cluster either with a Kafka credential or through a service dependency.

Related Information

Defining co-located Kafka clusters using a service dependency

Defining co-located Kafka clusters using Kafka credentials

Defining co-located Kafka clusters using a service dependency

Before you can start replicating data with Streams Replication Manager, you must define the co-located Kafka clusters that take part in the replication process. This can be done by enabling a service dependency between the Streams Replication Manager service and the co-located Kafka cluster, specifying an alias for the co-located cluster, and configuring security related properties.

About this task

The following list of steps walk you through how you can define the co-located Kafka cluster using a service dependency.

Before you begin

Ensure that you have reviewed Defining and adding clusters for replication and understand that the following list of steps are only one part of the full configuration workflow.

Procedure

1. In Cloudera Manager, go to Clusters and select the Streams Replication Manager service.
2. Go to Configuration.
3. Configure the co-located cluster:
   a) Find and enable the Kafka Service property.
   b) Find and configure the Streams Replication Manager Co-located Kafka Cluster Alias property.
      The alias you configure represents the co-located cluster. Enter an alias that is unique and easily identifiable. For example:

      ```
      uswest
      ```

   c) If the co-located cluster uses any type of security, enable the relevant security feature toggles.
      - If the cluster uses TLS/SSL, enable the following properties:
        - Enable TLS/SSL for SRM Driver
        - Enable TLS/SSL for SRM Service
      - If the cluster uses Kerberos authentication, enable the Enable Kerberos Authentication property.
      - If the cluster uses TLS/SSL and Kerberos, enable all three properties.

Results

The co-located Kafka cluster is defined using a service dependency.

What to do next

Add both external and co-located Kafka clusters to Streams Replication Manager's configuration.

Related Information

Adding clusters to Streams Replication Manager's configuration

Defining co-located Kafka clusters using Kafka credentials

Before you can start replicating data with Streams Replication Manager, you must define the co-located Kafka clusters that take part in the replication process. This can be done using Cloudera Manager by creating Kafka credentials.

About this task

The following list of steps walk you through how you can define the co-located Kafka cluster using Kafka credentials.

Before you begin

- Ensure that you have reviewed Defining and adding clusters for replication and understand that the following list of steps are only one part of the full configuration workflow.

Procedure

1. In Cloudera Manager, go to Administration External Accounts .
2. Go to the Kafka Credentials tab.

   The Kafka Credentials tab enables you to create, configure, and manage Kafka credentials. On this tab you will create a credential for your co-located Kafka cluster.
3. Create and configure Kafka credentials for the external clusters:
   a) Click Add Kafka credentials.
   b) Configure the Kafka credential:

      The security configuration of the co-located cluster determines which of the available properties you must set. To see which exact properties are required for your scenario, click one of the following tabs matching the security protocol of the cluster you are defining:

      For PLAINTEXT
      - Name
      - Bootstrap servers
      - Security protocol

      For SSL
      Encryption only
      - Name
      - Bootstrap servers
      - Security protocol
      - Truststore Password
      - Truststore Path
      - Truststore Type

      Encryption and authentication
      - Name
      - Bootstrap servers
      - Security protocol
      - Truststore Password
      - Truststore Path
      - Truststore Type
      - Key Password
      - Keystore Password
      - Keystore Path
      - Keystore Type

      For SASL_PLAINTEXT
      - Name
      - Bootstrap servers
      - Security protocol

- JAAS Secret [1-3]
- JAAS Template
- Kerberos Service Name
- SASL Mechanism

For SASL_SSL

- Name
- Bootstrap servers
- Security protocol
- Truststore Password
- Truststore Path
- Truststore Type
- JAAS Secret [1-3]
- JAAS Template
- Kerberos Service Name
- SASL Mechanism



Note: Click the i icon next to each property in Cloudera Manager to reveal additional information about each property.

c) Click Add.

If credential creation is successful, a new entry corresponding to the Kafka credential you specified appears on the page.

The following tabs collect examples of different Kafka credentials as well as some notes regarding configuration. Review these examples to better understand how to correctly configure a Kafka credential.

For PLAINTEXT

```
name=uswest
Bootstrap servers=uswest-node1.cluster.com:9092,uswest-node2.clu
ster.com:9092,uswest-node3.cluster.com:9092
Security protocol=PLAINTEXT
```

For SSL

Encryption only

```
Name=uswest
Bootstrap servers=uswest-node1.cluster.com:9093,uswest-node2.clu
ster.com:9093,uswest-node3.cluster.com:9093
Security protocol=SSL
Truststore Password=password
Truststore Path=/opt/srm/us-west-truststore.jks
Truststore Type=JKS
```

Encryption and authentication

```
Name=uswest
Bootstrap servers=uswest-node1.cluster.com:9093,uswest-node2.clu
ster.com:9093,uswest-node3.cluster.com:9093
Security protocol=SSL
Truststore Password=password
Truststore Path=/opt/srm/us-west-truststore.jks
Truststore Type=JKS
Keystore Password=password
Keystore Path=/opt/srm/us-west-keystore.jks
Keystore Type=JKS
```

```
Key Password=password
```

Note: The truststore and keystore files must be deployed on all Streams Replication Manager hosts and Streams Replication Manager must be able to access these files.

For SASL_PLAINTEXT

Kerberos

```
Name=uswest
Bootstrap servers=uswest-node1.cluster.com:9092,uswest-node2.clu
ster.com:9092,uswest-node3.cluster.com:9092
Security protocol=SASL_PLAINTEXT
JAAS Template=com.sun.security.auth.module.Krb5LoginModule requ
ired useKeyTab=true keyTab="/opt/srm/streamsrepmgr.keytab" princ
ipal="streamsrepmgr@REALM.COM";
Kerberos Service Name=kafka
SASL Mechanism=GSSAPI
```

PLAIN

```
Name=uswest
Bootstrap servers=uswest-node1.cluster.com:9092,uswest-node2.clu
ster.com:9092,uswest-node3.cluster.com:9092
Security protocol=SASL_PLAINTEXT
JAAS Secret 1=password
JAAS Template=org.apache.kafka.common.security.plain.PlainLoginM
odule required username="username" password="##JAAS_SECRET_1##";
SASL Mechanism=PLAIN
```

Note: The Kerberos principal must not include host names. Additionally, the keytab file must be deployed on all Streams Replication Manager hosts and Streams Replication Manager must be able to access this file.

For SASL_SSL

Kerberos

```
Name=uswest
Bootstrap servers=uswest-node1.cluster.com:9093,uswest-node2.clu
ster.com:9093,uswest-node3.cluster.com:9093
Security protocol=SASL_SSL
Truststore Password=password
Truststore Path=/opt/srm/us-west-truststore.jks
Truststore Type=JKS
JAAS Template=com.sun.security.auth.module.Krb5LoginModule requi
red useKeyTab=true keyTab="/opt/srm/streamsrepmgr.keytab" princi
pal="streamsrepmgr@REALM.COM";
Kerberos Service Name=kafka
SASL Mechanism=GSSAPI
```

PLAIN

```
Name=uswest
Bootstrap servers=uswest-node1.cluster.com:9093,uswest-node2.clu
ster.com:9093,uswest-node3.cluster.com:9093
Security protocol=SASL_SSL
Truststore Password=password
Truststore Path=/opt/srm/us-west-truststore.jks
Truststore Type=JKS
JAAS Secret 1=password
JAAS Template=org.apache.kafka.common.security.plain.PlainLoginMo
dule required username="username" password="##JAAS_SECRET_1##";
```

```
SASL Mechanism=PLAIN
```

Note: The Kerberos principal must not include host names. Additionally, the keytab and truststore files must be deployed on all Streams Replication Manager hosts and Streams Replication Manager must be able to access these files.

You can check whether Streams Replication Manager has access to required keytab files by running the following commands.

```
sudo -u streamsrepmgr kinit -kt [***PATH TO KEYTAB***] streams
repmgr@[***DOMAIN.REALM***]
klist
```

If access is not provided, you can grant access with the following commands.

```
chown streamsrepmgr:streamsrepmgr [***PATH TO KEYTAB***]
chmod 400 [***PATH TO KEYTAB***]
```

Results

The co-located Kafka cluster is defined using Kafka credentials.

What to do next

Add both external and co-located Kafka clusters to Streams Replication Manager's configuration.

Related Information

Adding clusters to Streams Replication Manager's configuration

Adding clusters to Streams Replication Manager's configuration

After both the external and co-located are defined, you must add them to the Streams Replication Manager service's configuration. This is done by configuring various configuration properties in Cloudera Manager.

About this task

Defining the clusters that you want to replicate is not sufficient. You must add the clusters you defined to the Streams Replication Manager service's configuration. This is required because Streams Replication Manager does not automatically pick up the clusters that you define.

Before you begin

Ensure that you have reviewed Defining and adding clusters for replication and understand that the following list of steps are only one part of the full configuration workflow.

Procedure

1. In Cloudera Manager, go to Clusters and select the Streams Replication Manager service.
2. Go to Configuration.
3. Add the clusters defined with Kafka credentials to Streams Replication Manager's configuration:
   a) Find and configure the External Kafka Accounts property.
   b) Click the add button and add new lines for each Kafka credential you created.
   c) Add the names of all Kafka credentials.

   Each Kafka credential must be added to a new line. For example:

```
useast
usnorth
```

Note: The alias representing the co-located cluster should only be included in this list if the co-located cluster was defined with a Kafka credential. If you used the service dependency method to define the co-located cluster, this list should only contain the external clusters.

4. Find and configure the Streams Replication Manager Cluster alias property.

   Add all cluster aliases including:

   - Aliases present in the External Kafka Accounts property
   - If the co-located cluster was defined with a service dependency, include the alias present in the Streams Replication Manager Co-located Kafka Cluster Alias property.

   Delimit the aliases with commas. For example:

   ```
   useast, usnorth, uswest
   ```

5. Click Save Changes.
6. Restart the Streams Replication Manager service.

Results

Both external and co-located Kafka clusters are added to Streams Replication Manager's configuration.

What to do next

Add and configure replications. Alternatively, if replications are already configured, you can continue with configuring the srm-control tool.

Related Information

Configuring replications

Configuring srm-control

Configuring replications

In order to replicate data between clusters, you must configure replications for Streams Replication Manager. Each replication specifies a source->target cluster pair and the direction in which data is replicated. Replications can be configured in Cloudera Manager.

About this task

Configuring replications does not start the replication of data. When you configure your replications, Streams Replication Manager sets up communication with the clusters specified within each replication, but does not automatically replicate any data. To start replicating data, you must specify which topics and groups you want to replicate. This is done using the srm-control command line tool. Unless the tool is used to add topics and consumer groups to the replication allowlists, the replications you configure remain empty (inactive).

Procedure

1. In Cloudera Manager, select Streams Replication Manager.
2. Go to Configuration.
3. Add and enable replications:
   a) Find the Streams Replication Manager's Replication Configs property.
   b) Click the add button and add new lines for each unique replication you want to add and enable.
   c) Add and enable your replications. For example:

   ```
   primary->secondary.enabled=true
   secondary->primary.enabled=true
   ```

4. Enter a Reason for change, and then click Save Changes to commit the changes.
5. Restart Streams Replication Manager.

Results
Replications are set up and configured for the specified cluster pairs.

What to do next
Use the srm-control tool to start replication by adding topics or groups to the allowlist.

Related Information
srm-control

Configuring the driver role target clusters

The Streams Replication Manager Driver role's target clusters are the clusters that the driver is writing data to. You can configure these target clusters for each instance of the driver with the Streams Replication Manager Driver Target Cluster property. Custom configuration of these targets is only recommended in advanced deployments.

About this task

The Streams Replication Manager Driver role is responsible for connecting to the specified clusters and performing replication between them. The driver can be installed on one or more hosts within a cluster.

The clusters the driver connects to are the clusters that you specify with the Streams Replication Manager Cluster alias and Streams Replication Manager's Replication Configs properties.

Target clusters of the driver are clusters that the driver writes data to. By default when the driver is started it will connect to all clusters, gather data from them, and write to all of them. In other words, by default a driver targets all clusters in your configuration. You can limit the number of clusters that each driver targets. This can be done with the Streams Replication Manager Driver Target Cluster property, which allows you to specify which cluster or clusters the driver targets.

When you specify a driver target, the driver still connects to all clusters and gathers data from them, but will only write to the clusters specified.

However, in order for monitoring to function correctly, the driver has to target all clusters taking part in the replication. That is, it has to contain the actual target, the cluster you want to write data to, as well as the source clusters for that target, where data is being pulled from. If the source clusters are not specified, you will not be able to monitor your replications. As a result of this, configuring driver targets and limiting the number of clusters each instance of the driver writes to is considered an advanced configuration practice. This practice is only viable in complex replication scenarios that involve a high number clusters and replications. Therefore, Cloudera recommends that you either leave this property empty or add all clusters taking part in the replication.

By default the Streams Replication Manager Driver Target Cluster property is left empty, meaning that all clusters are targeted. The property accepts any cluster alias that is specified in Streams Replication Manager Cluster alias. When adding multiple cluster aliases, delimit them with a comma.

Procedure

1. In Cloudera Manager, select Streams Replication Manager.
2. Go to Configuration.
3. Find the Streams Replication Manager Driver Target Cluster property.
4. Add the cluster aliases that you want the driver role to target. For example:

```
primary, secondary
```

5. Enter a Reason for change, and then click Save Changes to commit the changes.

6. Restart Streams Replication Manager.

Results
Driver targets are configured. Drivers only write data to the targeted clusters.

Configuring the service role target cluster

The Streams Replication Manager Service role's target cluster is the cluster from which metrics are gathered and exposed. Multiple targets can be configured for each instance of the service with the Streams Replication Manager Service Target Cluster property. Configuration is mandatory.

About this task

The Streams Replication Manager Service role consists of a REST API and a Kafka Streams application that aggregates and exposes cluster, topic, and consumer group metrics. With the help of these metrics, users can monitor replications. The service can be installed on one or more hosts per cluster.

Each instance of the service is associated with at least one target cluster. The target is the cluster that the service gathers and exposes metrics from. Each instance of the service can have multiple targets. As a result, it is possible to use a single instance of the Service role to target and monitor multiple clusters.

Service role target clusters are configured with the Streams Replication Manager Service Target Cluster property. The property accepts any cluster alias that is specified in Streams Replication Manager Cluster alias as long as data is being replicated to that cluster. Configuring at least a single service target is mandatory.

Procedure

1. In Cloudera Manager, select Streams Replication Manager.
2. Go to Configuration.
3. Find the Streams Replication Manager Service Target Cluster property.
4. Add your target cluster aliases.

   The number of aliases you add depends on whether you want the Service role to target and monitor a single cluster or multiple clusters.

   If you want to target a single cluster, add a single cluster alias. For example:

   ```
   secondary
   ```

   If you want to target multiple clusters, add multiple aliases separated by commas. For example:

   ```
   primary,secondary,tertiary
   ```
5. Enter a Reason for change, and then click Save Changes to commit the changes.
6. Restart Streams Replication Manager.

Results
The Service role targets are configured. The Service role gathers and exposes metrics from the specified clusters.

Configuring properties not exposed in Cloudera Manager

There are a number of configuration properties that Streams Replication Manager accepts, but are not exposed directly in Cloudera Manager. You can configure these properties with the Streams Replication Manager's Replication Configs property.

About this task

In addition to the configuration properties exposed directly for configuration through Cloudera Manager, Streams Replication Manager accepts number of additional properties including Streams Replication Manager specific properties, Kafka Connect worker properties, as well as Kafka properties available in the version of Kafka that you are using. Properties not exposed directly in Cloudera Manager can be set through the Streams Replication Manager's Replication Configs property.

The following steps demonstrate how these properties can be configured using the Streams Replication Manager's Replication Configs property. If you want to learn more about the properties that you can configure, see the links provided in the *Related information* section at the bottom of this page.

Before you begin

Make sure that cluster aliases and replications are configured. Otherwise cluster or replication level configuration is not possible.

Procedure

1. In Cloudera Manager, select Streams Replication Manager.
2. Go to Configuration.
3. Configure properties not exposed in Cloudera Manager:
   a) Find the Streams Replication Manager's Replication Configs property.
   b) Click the add button and add new lines for each additional property you want to configure.
   c) Add configuration properties. For example:

   ```
   offset.flush.timeout.ms=20000
   ```
4. Enter a Reason for change, and then click Save Changes to commit the changes.
5. Restart Streams Replication Manager.

Results

Configuration properties not directly exposed in Cloudera Manager are configured.

Related Information

Configuration Properties Reference for Properties not Available in Cloudera Manager

Configuring replication specific Kafka Connect REST servers

Understanding Streams Replication Manager properties, their configuration and hierarchy

Suppressing internal metrics topics

Suppress the creation of unused internal metrics topics to avoid unnecessary topic creation.

About this task

By default, the Streams Replication Manager Service eagerly creates a number of different srm-metrics topics in advance for all possible replication flows. This ensures that the Service does not crash when there are no source topics if the Service starts earlier than the Driver.

Because the Service does not have information of all the exact replications running in the cluster, this behavior generates unused topics. You can configure Streams Replication Manager to suppress the eager creation of these topics by setting the metrics.topic.creation.for.possible.flows.enabled property to false.

When you set this property to false, only the Driver creates the srm-metrics topics.

Important: If you set metrics.topic.creation.for.possible.flows.enabled to false and do not configure replication, the Streams Replication Manager Service restarts repeatedly until you configure a replication flow and the Driver creates the topics.

Procedure

1. In Cloudera Manager, select the Streams Replication Manager service.
2. Go to Configuration.
3. Find the Additional Configs For Streams Application Running Inside SRM Service property.
4. Add the following configuration property:

```
metrics.topic.creation.for.possible.flows.enabled=false
```

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

Configuring replication specific Kafka Connect REST servers

The Streams Replication Manager Driver role uses a single REST server to handle inter-worker communication for all replications. This REST server enables communication between the different instances of the Streams Replication Manager driver and ensures that replication does not fail when there are multiple driver instances present in a single cluster.

About this task

If required, you can configure the REST server that is set up by the driver.

Important:

The legacy per-flow REST server implementation is deprecated and will be removed in a future release. In the previous implementation, the Streams Replication Manager Driver role started a dedicated REST server for each replication flow, configured via the ***ALIAS***->***ALIAS***.worker. prefix. Cloudera recommends that you migrate your Streams Replication Manager clusters to the new single-server implementation.

REST server properties can be configured through the Streams Replication Manager's Replication Configs property by using the following prefix:

listeners

Configures the host address and port of the REST server. The value must be in protocol://host:port format. For example:

```
listeners=http://localhost:4445
```

listeners.https.

This prefix can be used to configure SSL related properties. You can use this prefix to override the SSL settings that the REST server inherits from the service configuration. For example, if the REST server needs to use a different keystore location than the one provided in the service configuration, you can use the following property:

```
listeners.https.ssl.keystore.location=***CUSTOM KEYSTORE LOCATIO
N***
```

Procedure

1. In Cloudera Manager, select the Streams Replication Manager service.
2. Go to Configuration.
3. Find the Streams Replication Manager's Replication Configs property.
4. Click the add button and add new lines for each additional property you want to configure.
5. Add configuration properties. For example:

```
listeners.https.ssl.keystore.location=***CUSTOM KEYSTORE LOCATION***
```

6. Enter a Reason for change, and then click Save Changes to commit the changes.
7. Restart Streams Replication Manager.

Results

Custom configuration of the Kafka Connect REST server is complete.

Configuring Remote Querying

Remote Querying in Streams Replication Manager refers to the Streams Replication Manager Service's capability of querying other, remote Streams Replication Manager Services to fetch the remote cluster replication metrics. This allows users to monitor all replications of a deployment that has multiple instances of Streams Replication Manager through a single Streams Replication Manager Service.

Related Information

Remote Querying

Enabling Remote Querying

Remote Querying can be enabled for a cluster of Streams Replication Manager Service roles (Streams Replication Manager Service cluster). When this feature is enabled, the configured Streams Replication Manager Service cluster acts as a monitoring gateway and fetches metrics from other, remote Streams Replication Manager Service clusters. This enables you to monitor the replications of your whole deployment using a single Streams Replication Manager Service cluster.

About this task

You enable Remote Querying for the Streams Replication Manager Service clusters that you want to designate as monitoring gateways. Once enabled, the designated Streams Replication Manager Service cluster discovers and communicates with other, remote Streams Replication Manager Service clusters and fetches the metrics collected by the remote Streams Replication Manager Service clusters. After configuration is complete, you can use the designated Streams Replication Manager Service cluster to monitor all replications in your deployment.

Configuration is done in Cloudera Manager by providing a list of external Kafka cluster aliases in the Streams Replication Manager Service Remote Target Clusters property. These clusters must be defined in Cloudera Manager using Kafka credentials.

Before you begin

- If any of the Streams Replication Manager installations are located behind a proxy, complete Configuring the advertised information of the Streams Replication Manager Service role
- Define all external Kafka clusters that you want to monitor.
    - External clusters are defined by creating and configuring Kafka credentials. Kafka credentials are created and configured in Cloudera Manager on the Administration External Accounts Kafka Credentials page. For more information, see Defining external Kafka clusters.

  - Ensure that the Name property of each Kafka credential you create for the external Kafka clusters matches the alias set for that Kafka cluster in the configuration of that Kafka cluster's co-located Streams Replication Manager instance. The alias can be found in the Streams Replication Manager Co-located Kafka Cluster Alias property.
  - Note down the Name property of each Kafka credential you create for the external Kafka clusters. You need to provide these during configuration.
- Ensure that network connectivity and DNS resolution are established between the clusters in your deployment. Additionally, the Streams Replication Manager Service's HTTPS port (default 6671) must have both inbound and outbound traffic enabled.

Procedure

1. In Cloudera Manager, select the Streams Replication Manager service.
2. Go to Configuration.
3. Find and configure the Streams Replication Manager Service Remote Target Clusters property.

   Add a comma separated list of the external Kafka cluster aliases that you want to monitor. For example:

   ```
   remote_1, remote_2
   ```

   Tip: Cluster aliases are specified during Kafka credential creation. A cluster alias is equivalent to the Name of the Kafka credential. Ensure that the aliases you add match the names of the Kafka credentials you created. Additionally, ensure that the aliases are identical with the aliases used on the remote clusters that you want to monitor.
4. Click Save Changes.
5. Restart the Streams Replication Manager service.

Results

Remote Querying is enabled and configured for the selected Streams Replication Manager Service cluster. The Streams Replication Manager Service cluster discovers and then communicates with other, remote Streams Replication Manager Service clusters and fetches metrics available on the remote Streams Replication Manager Service clusters.

What to do next

- If any of the remote Streams Replication Manager Services have Basic Authentication enabled, complete Configuring Basic Authentication for Remote Querying.
- Access the **Replications** page on the Streams Messaging Manager UI. Remote replications will be visible in the UI.
- Query metrics using the Streams Replication Manager REST API. For example:

  1. Go to Streams Replication Manager Web UI SRM Service Swagger UI .
  2. Find and open the /v2/replications endpoint.
  3. Click Try it out then click Execute.

     The response includes all discovered replications, replicated topics, and various other metrics. This includes replications that target remote clusters.

Related Information

Remote Querying

Configuring the advertised information of the Streams Replication Manager Service role

If your Streams Replication Manager deployment is located behind a proxy, you must configure the information advertised by a cluster of Streams Replication Manager Service roles (Streams Replication Manager Service cluster). Otherwise, the Streams Replication Manager Service cluster cannot be discovered by other Streams Replication Manager Service clusters and Remote Querying will not be possible.

About this task

Each Streams Replication Manager Service cluster advertises itself automatically through its target Kafka cluster. Advertised information includes properties like the protocol, host, port, and root API path of the Streams Replication Manager Service cluster. This information is used by Streams Replication Manager Service clusters that have Remote Querying enabled to connect to and communicate with remote Streams Replication Manager Service clusters and fetch the metrics collected by them.

However, if a Streams Replication Manager Service cluster is located behind a proxy, the information advertised by default cannot be used to connect to them. In such a case, the advertised protocol, host, port, and root API path must be configured to match the properties of the proxy. If this configuration is not completed, the Streams Replication Manager Service clusters cannot be discovered.

Procedure

1. In Cloudera Manager, select the Streams Replication Manager service.
2. Go to Configuration.
3. Find and configure the following properties:
   - SRM Service Advertised Protocol For Remote Queries
   - SRM Service Advertised Host For Remote Queries
   - SRM Service Advertised Port For Remote Queries
   - SRM Service Advertised API Root Path For Remote Queries
4. Click Save Changes.
5. Restart the Streams Replication Manager service.

Results

The Streams Replication Manager Service cluster advertises the configured protocol, host, port and API root path.

Related Information

Remote Querying

Configuring Streams Replication Manager Driver retry behaviour

If a target Kafka cluster is not available at startup, the replications targeting that cluster are not set up. In a case like this, the Streams Replication Manager Driver sets up the replications for available clusters and retries replication setup for the unavailable clusters. Retry behaviour can be configured in Cloudera Manager.

About this task

By default the Streams Replication Manager Driver is configured to retry indefinitely with a retry interval of 10000 ms (10 seconds). Both the retry count and retry interval can be configured in Cloudera Manager using the Retry Count for SRM Driver and Retry Delay for SRM Driver properties.

Retry Count for SRM Driver

Configures the maximum number of retries. The default value of -1 means that retries are carried out indefinitely. If the configured retry count is reached, the Streams Replication Manager Driver will stop. Additionally, setting the property to 0 is the same as setting it to 1.

Retry Delay for SRM Driver

Configures the interval at which retries happen, this is the delay (in ms) between each retry.

Note: When retries are happening, at least one of the configured replications is not running as expected. As a result, the DISTRIBUTED_HERDER_STATUS health test returns that replication status is bad. Based on the health test result, Cloudera Manager will display that the Streams Replication Manager Driver is in bad (red) health.

Procedure

1. In Cloudera Manager, select the Kafka service.
2. Go to Configuration.
3. Find and configure the following properties:
   - Retry Count for SRM Driver
   - Retry Delay for SRM Driver
4. Click Save Changes.
5. Restart the Streams Replication Manager service.

Results

Retry behaviour is configured for the Streams Replication Manager Driver. The Streams Replication Manager Driver uses the configured values if a Kafka cluster is unavailable at startup.

Configuring Streams Replication Manager Driver heartbeat emission

The Streams Replication Manager Driver spins up a Kafka Connect worker for each possible replication in your deployment. However, some of these workers created for disabled replications might be unnecessary. Unnecessary workers can be deactivated by configuring heartbeat emission. This can help with minimizing unnecessary performance overhead.

About this task

At startup the Streams Replication Manager Driver creates a Kafka Connect worker for each possible replication based on the cluster aliases you have configured. This means that a Connect worker is created not only for the replications that you specifically enable, but also for the ones that are disabled. The difference between an enabled and disabled replication is that for disabled replications not all possible connectors are set up. For disabled replications only the MirrorHeartbeatConnector is created which is responsible for heartbeat emission. For more information on Connect workers and connector instances created by the Streams Replication Manager Driver, see *Streams Replication Manager Architecture*.

In certain scenarios, some of the Connect workers set up for a disabled replication are unnecessary. In such a case, you can configure Streams Replication Manager to not create these Connect workers and to fully deactivate unutilized replications. This can be done by disabling heartbeat emission for these replications. If heartbeat emission is disabled, the Connect worker and MirrorHeartbeatConnector are not created. As a result, the replication associated with that worker is fully deactivated. This can help in minimizing the performance overhead caused by unnecessary Connect workers.

Heartbeat emission is configured with the emit.heartbeats.enabled property. This property can be set on a global as well as replication specific level. On a global level, heartbeat emission can be turned on and off using the Enable Heartbeats Cloudera Manager property. On a replication level, configuration is done by adding emit.heartbeats.enab

led to Streams Replication Manager's Replication Configs with a replication prefix. For example, A->B.emit.heartbeats.enabled=true.

Which disabled replications are unnecessary differs from deployment to deployment. To operate correctly, Streams Replication Manager requires that all clusters that act as a source have a heartbeats topic. The heartbeats topic is created by Connect workers that target the cluster. This means that at minimum, each source cluster must be targeted by at least a single enabled or disabled replication. All other disabled replications can be fully deactivated.

For example, assume that you have three clusters, A, B, and C. The Streams Replication Manager Driver is targeting all three clusters. Because the Streams Replication Manager Driver spins up a Connect worker for each possible replication, the following Connect workers are created:

- B->A
- C->A
- A->B
- C->B
- A->C
- B->C

Now, assume that the enabled replications, the ones that replicate data, are A->B, B->C, and C->A. This means that all three clusters act as sources. Therefore, heartbeat emission must be enabled in at least a single replication that is targeting the clusters. All other replications can be fully deactivated. In this example, the enabled replications A->B, B->C, and C->A target one of the clusters each. As a result, these replications take care of heartbeating. Out of all replications, the following can be deactivated:

- B->A
- C->B
- A->C

This can be achieved by for example enabling heartbeat emission on a global level with Enable Heartbeats and adding the following entries to Streams Replication Manager's Replication Configs:

```
B->A.emit.heartbeats.enabled=false
C->B.emit.heartbeats.enabled=false
A->C.emit.heartbeats.enabled=false
```

Consider another example. Assume that you have the same setup, but the enabled replications are A->B and B->C. Cluster A acts as a source but is not targeted by any enabled replication. In a case like this, you can only fully deactivate replications C->B and A->C. One of the replications targeting cluster A (B->A or C->A) must remain disabled and cannot be deactivated. Otherwise, a heartbeats topic will not be created in cluster A.

This can be achieved by for example enabling heartbeat emission on a global level with Enable Heartbeats and adding the following entries to Streams Replication Manager's Replication Configs:

```
C->B.emit.heartbeats.enabled=false
A->C.emit.heartbeats.enabled=false
B->A.emit.heartbeats.enabled=false
```

In this example, replication B->A was fully deactivated (heartbeat emission disabled) and replication C->A was left disabled. Although replication C->A is disabled, it is still actively emitting heartbeats as a result of Enable Heartbeats being enabled.

Procedure

1. In Cloudera Manager, select the Streams Replication Manager Service.
2. Go to Configuration.
3. Configure heartbeat emission for your replications.

   You have two options. You can enable heartbeat emission on a global level and disable it for the required replications, or disable heartbeat emission on a global level and enable it for the required replications.

For Enable globally and disable per replication

a. Find and enable the Enable Heartbeats property.

b. Find and configure the Streams Replication Manager's Replication Configs property.

Add the emit.heartbeats.enabled property with a correct replication prefix. For example:

```
A->B.emit.heartbeats.enabled=false
```

Repeat this step for all replications that you want to disable heartbeat emission for.

For Disable globally and enable per replication

a. Find and disable the Enable Heartbeats property

b. Find and configure the Streams Replication Manager's Replication Configs property.

Add the emit.heartbeats.enabled property with correct replication prefix. For example:

```
A->B.emit.heartbeats.enabled=true
```

Repeat this step for all replications that you want to enable heartbeat emission for.

4. Click Save Changes.
5. Restart Streams Replication Manager.

Results

Heartbeat emission is configured. Unnecessary replications are fully deactivated.

Related Information

Streams Replication Manager Architecture

Configuring automatic group offset synchronization

Learn how to enable automatic group offset synchronization in Streams Replication Manager. Enabling this feature can simplify the manual steps that you need to take to migrate consumer groups in a failover or failback scenario.

About this task

By default, automatic group offset synchronization is disabled. It can be enabled in Cloudera Manager by adding new entries to the Streams Replication Manager's Replication Configs property.

Depending on your use case, you can set up automatic group offset synchronization on a global level or for specific replications only.

Procedure

1. In Cloudera Manager, select Streams Replication Manager.
2. Go to Configuration.
3. Find the Streams Replication Manager's Replication Configs and add the following configuration entries:

```
sync.group.offsets.enabled = true
sync.group.offsets.interval.seconds = [***TIME IN SECONDS***]
```

In this example, all properties are set on a global level. This means that automatic group offset synchronization is applied to all replications. Depending on your setup, you can also choose to set these properties for specific replications using appropriate replication prefixes.

The sync.group.offsets.enabled property enables automatic group offset synchronization. The sync.group.offsets.interval.seconds property controls how frequently offsets are synched. You only need to specify this property if you want to customize synchronization frequency.

4. Enter a Reason for change, and then click Save Changes to commit the changes.
5. Restart Streams Replication Manager.

Results

Automatic group offset synchronization is enabled. From now on, Streams Replication Manager automatically exports the translated offsets of the configured source clusters and applies them to the configured target clusters.

Related Information

Automatic group offset synchronization

Configuring Streams Replication Manager Driver for performance tuning

Learn about the configuration options available for tuning the performance of Streams Replication Manager Driver.

To perform performance tuning for Streams Replication Manager Driver, it is important that you understand the task architecture of Streams Replication Manager. For more information, see Streams Replication Manager Architecture.

It is also important that you are familiar with the task load balancing method used by Streams Replication Manager. For more information, see Task architecture and load-balancing.

You need to tune the following main groups of configurations for the Streams Replication Manager Driver when running a replication flow with a heavy load:

- Hardware
- Task count
- Consumer and producer configurations of MirrorSourceTask
- Connect worker configurations

Hardware

Learn about the hardware-related aspects of Streams Replication Manager performance tuning, such as memory requirements.

For high workloads, Streams Replication Manager Drivers need to run on dedicated nodes. Streams Replication Manager Drivers are typically network-bound, but due to configuration tweaks, memory consumption can also be high.

For small workloads, a maximum heap size of 8 GB is sufficient, however, for larger workloads, Cloudera recommends a higher maximum heap size. Streams Replication Manager Driver performance can significantly degrade when limited by memory or due to garbage collection pauses. Always make sure that Streams Replication Manager Drivers have enough memory. You can define the heap size using the SRM_HEAP_OPTS property in Cloudera Manager.

Task count

Learn about configuring optimal task counts in relation to the number of topic partitions, Streams Replication Manager Drivers, and Streams Replication Manager Driver nodes.

The task count specifies the level of parallelism of the data replication. One task corresponds to (approximately) one worker thread on the Streams Replication Manager Driver cluster. Each task runs a consumer and a producer instance, meaning more connections to the source and target Kafka clusters.

To increase the parallelism of the work, you can increase the Tasks Max property in Cloudera Manager. Since the unit of work is the source topic-partition, there is no need for the value of Tasks Max to exceed the number of replicated topic-partitions, as it has no effect on the throughput of the system.

When Streams Replication Manager Drivers are running on dedicated nodes, Cloudera recommends setting the value of Tasks Max approximately to the number of Streams Replication Manager Drivers times the number of Streams Replication Manager Driver node cores.



Tip: Consider changing this configuration when the consumer and producer batching configurations are already tweaked based on the source data bandwidth, and the CPU is underutilized on the Streams Replication Manager Driver nodes.

However, increasing Tasks Max may affect CPU usage. Make sure to monitor CPU usage metrics when tweaking this property.

Consumer and producer configurations

Learn about consumer and producer properties you can tweak to increase throughput and decrease overhead.

A MirrorSourceTask manages one consumer and one producer to drive the data replication. Similarly to custom Kafka client applications, you need to tweak these consumer and producer instances when expecting a high throughput on the replication flow. The following list is only an overview of properties that are typically tweaked for the consumer and the producer, as well as their application for Streams Replication Manager. On some workloads, you might also need to tweak broker configuration.

Consumer configurations

To configure the consumer inside the MirrorSourceTask, use the cluster->cluster.consumer. prefix in Streams Replication Manager's Replication Configs in Cloudera Manager. You can tweak the following consumer properties to increase throughput:

fetch.max.bytes

The maximum size of a fetch response. You can increase this to increase the consumer throughput and reduce the overhead of fetching.

max.poll.records

The maximum number of records returned in a fetch response. You can increase this to increase the consumer throughput and reduce the overhead of fetching.

receive.buffer.bytes

The size of the TCP receive buffer used by the consumer. When the average response size increases due to tweaking the previous configurations, you also need to increase the receive buffer bytes to reduce the overhead of buffering.

max.partition.fetch.bytes

The maximum amount of data returned for one topic partition in a fetch response. In some cases, where there are source topics with high ingestion rate to be replicated, you need to increase this configuration to allow the consumer to keep up with the ingestion rate. If the source topics have a similar ingestion rate, you do not need to change this.

Producer configurations

To configure the producer inside the MirrorSourceTask, use the cluster->cluster.producer.override. prefix in Streams Replication Manager's Replication Configs in Cloudera Manager. You can tweak the following producer properties to increase throughput:

Producer Batch Size

The maximum batch size created by the producer. Batches are created for each topic partition. You can increase this to increase the producer throughput and reduce the overhead of producing. When tweaking this property, make sure to check the value of max.request.size, as well as message.max.bytes on the broker side.

send.buffer.bytes

The size of the TCP send buffer used by the producer. When the average request size increases due to tweaking the previous configurations, you also need to increase the send buffer bytes to reduce the overhead of buffering.

Producer Compression Type

The compression type to use in the producer. You can use producer side compression to achieve higher throughput.

max.request.size

The maximum request size sent by the producer. The batch size cannot exceed this configuration. When tweaking this property, make sure to check the value of Producer Batch Size, as well as message.max.bytes on the broker side.

Producer Buffer Memory

The maximum amount of memory the producer can use for buffering records. For high throughput replications, heavy batching is necessary on the producer side. To allow heavy batching, you need to increase the buffer memory.



Note: Cloudera does not recommend changing the following Connect producer properties, because the changes might lead to violating Streams Replication Manager guarantees.

- acks (default: all)
- retries (default: int max)
- max.in.flight.requests.per.connection (default: 1)

Related Information

Streams Replication Manager Architecture

Connect worker configurations

Learn about the Connect worker properties you can change to facilitate heavy loads on Streams Replication Manager.

The Connect worker does not significantly influence the throughput of Streams Replication Manager. However, when running a heavy load with Streams Replication Manager, you might need to tweak the following properties.

offset.flush.timeout.ms

Affects the timeout when waiting for in-flight produce requests to finish before committing the source offsets. The default value is five seconds, which is a low timeout. It might be exceeded when there is a high load on a single task. You can safely increase this value, as the timeout does not affect the throughput of the actual flow, and data replication continues even when the Connect framework is waiting for a safe source offset commit.



Tip: Consider changing this when the ERROR level message Failed to flush, timed out while waiting for producer to flush outstanding X messages appears in the Streams Replication Manager Driver logs .

Offset Flush Interval

Controls the interval of the source offset flush. The offset flush does not have a significant impact in the data replication throughput. A possible reason to tweak this configuration is the average offset flush duration getting close to the interval. In such a case, increasing the interval might reduce unnecessary work.

Tip: Consider changing this when the average flush duration seems to be close to the flush interval, based on the DEBUG level message Finished offset commitOffsets successfully in X ms in the Streams Replication Manager Driver logs.

Metric calculation

The calculation of the following metrics might impact performance.

replication-records-lag

You can tweak the performance impact of the replication-records-lag metric by tweaking the values of the following properties in Streams Replication Manager's Replication Configs in Cloudera Manager.

replication.records.lag.calc.enabled

Controls whether the replication-records-lag metric is calculated. This metric provides information regarding the replication lag based on offsets. The metric is available both on the cluster and the topic level. The calculation of this metric might add latency to replications and impact Streams Replication Manager performance. If you are experiencing performance issues, you can try setting this property to false to disable the calculation of replication-records-lag. Alternatively, you can try fine-tuning how Streams Replication Manager calculates replication-records-lag with the replicat ion.records.lag.calc.period.ms and replication.records.lag.end.offset.timeout.ms properties.

replication.records.lag.calc.period.ms

Controls how frequently Streams Replication Manager calculates the replication-records-lag metric. The default value of 0 means that the metric is calculated continuously. Cloudera recommends configuring this property to 15000 ms (15 seconds) or higher if you are experiencing issues related to the calculation of replication-records-lag. A calculation frequency of 15 seconds or more results in the metric being available for consumption without any significant impact on Streams Replication Manager performance.

replication.records.lag.end.offset.timeout.ms

Specifies the Kafka end offset timeout value used for calculating the replication-records-lag metric. Setting this property to a lower value than the default 6000 ms (1 minute) might reduce latency in calculating replication-records-lag, however, replication-records-lag calculation might fail. A value higher than the default can help avoid metric calculation failures, but might increase replication latency and lower Streams Replication Manager performance.

New topic and consumer group discovery

Kafka topics or consumer groups may not get replicated instantly when they are added to white and blacklists. This is due to the default behaviour of how topics and consumer groups are discovered by Streams Replication Manager.

The discovery and replication of newly created topics or consumer groups is not instantaneous. Streams Replication Manager checks source clusters for new topics and consumer groups periodically, as controlled by the Refresh Topics Interval Seconds and Refresh Groups Interval Seconds properties. By default both properties are set to 10 minutes. As a result, the discovery and replication of new topics or groups can take up to 10 minutes.

Cloudera does not recommend using a refresh interval lower than the default value for production environments as it can lead to severe performance degradation.

Related Information

srm-control Topics and Groups Subcommand

Configuration examples

These configuration examples give step-by-step instructions on how you can set up and configure typical deployments of Streams Replication Manager. Reviewing these examples can help you gain a better understanding of how your specific setup can be configured.

Bidirectional replication example of two active clusters

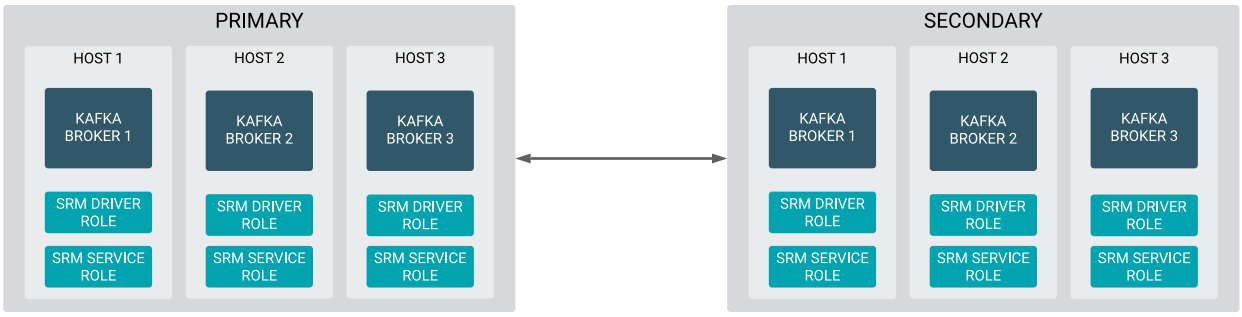
Review the bidirectional replication example to learn how you can configure and start replication with Streams Replication Manager in a deployment with two active clusters configured with bidirectional replication.

About this task

In a typical scenario, you may have two active Kafka clusters within the same region but in separate availability zones. With bidirectional replication, clients can connect to either cluster in case one is temporarily unavailable.

This example demonstrates the steps required to configure the deployment shown below. Additionally, it also provides example commands to start replication between clusters.

Figure 1: Bidirectional Replication of Active Clusters



Before you begin

- The following list of steps assume that both clusters are unsecured.
- The following list of steps assume that the Streams Replication Manager Service role is running on all Kafka broker hosts and is targeting its co-located cluster (the cluster it is running in).
- Steps 1 through 6 must be carried out on all clusters for all Streams Replication Manager services. The steps highlight when the configuration differs from cluster to cluster or if it is identical on all clusters.
- The following list of steps assume that the `DefaultReplicationPolicy` is in use.

Procedure

1. Define external clusters:
   a) In Cloudera Manager, go to Administration External Accounts .
   b) Go to the Kafka Credentials tab.
      On this tab you will create a credential for each external cluster taking part in the replication process.
   c) Click Add Kafka credentials.
   d) Configure the Kafka credential:
      On the primary cluster you have to add a credential that defines secondary. For example:

      ```
      Name=secondary
      ```

```
Bootstrap servers=secondary-1.cloudera.com:9092, secondary-2.cloudera
.com:9092, secondary-3.cloudera.com:9092
Security protocol=PLAINTEXT
```

On the secondary cluster you have to add a credential that defines primary. For example:

```
Name=primary
Bootstrap servers=primary-1.cloudera.com:9092, primary-2.cloudera.com:9
092, primary-3.cloudera.com:9092
Security protocol=PLAINTEXT
```

e) Click Add.

If credential creation is successful, a new entry corresponding to the Kafka credential you specified appears on the page.

2. Define the co-located Kafka cluster:

a) In Cloudera Manager, go to Clusters and select the Streams Replication Manager service.
b) Go to Configuration.
c) Find and enable the Kafka Service property.
d) Find and configure the Streams Replication Manager Co-located Kafka Cluster Alias property.

The alias you configure represents the co-located cluster. Enter an alias that is unique and easily identifiable. On the primary cluster:

```
primary
```

On the secondary cluster:

```
secondary
```

3. Add the clusters defined with Kafka credentials to Streams Replication Manager's configuration:

a) Find and configure the External Kafka Accounts property.
b) Click the add button and add new lines for each Kafka credential you created.
c) Add the names of all Kafka credentials.

Each Kafka credential must be added to a new line. On the primary cluster:

```
secondary
```

On the secondary cluster:

```
primary
```

4. Find and configure the Streams Replication Manager Cluster alias property.

Add all cluster aliases to this property. This includes the aliases present in both the External Kafka Accounts and Streams Replication Manager Co-located Kafka Cluster Alias properties. Delimit the aliases with commas. In the case of this example the configuration is identical on both clusters:

```
primary, seconadry
```

5. Configure Driver role target clusters:

a) Find the Streams Replication Manager Driver Target Cluster property.
b) Add the cluster aliases that you want the driver role to target.

In the case of this example, each Driver should target its co-located cluster. On the primary cluster:

```
primary
```

On the secondary cluster:

```
secondary
```

6. Add and enable replications:
   a) Find the Streams Replication Manager's Replication Configs property.
   b) Click the add button and add new lines for each unique replication you want to add and enable.
   c) Add and enable your replications.

   In the case of this example, the configuration will be identical on both clusters:

```
primary->secondary.enabled=true
secondary->primary.enabled=true
```

7. Replicate data between clusters with the following commands:

   Important: The following commands assume that the `DefaultReplicationPolicy` is in use. Running the following commands with the `IdentityReplicationPolicy` would result in replication loops.

```
srm-control topics --source primary --target secondary --add ".*"
```

```
srm-control topics --source secondary --target primary --add ".*"
```

Cross data center replication example of multiple clusters

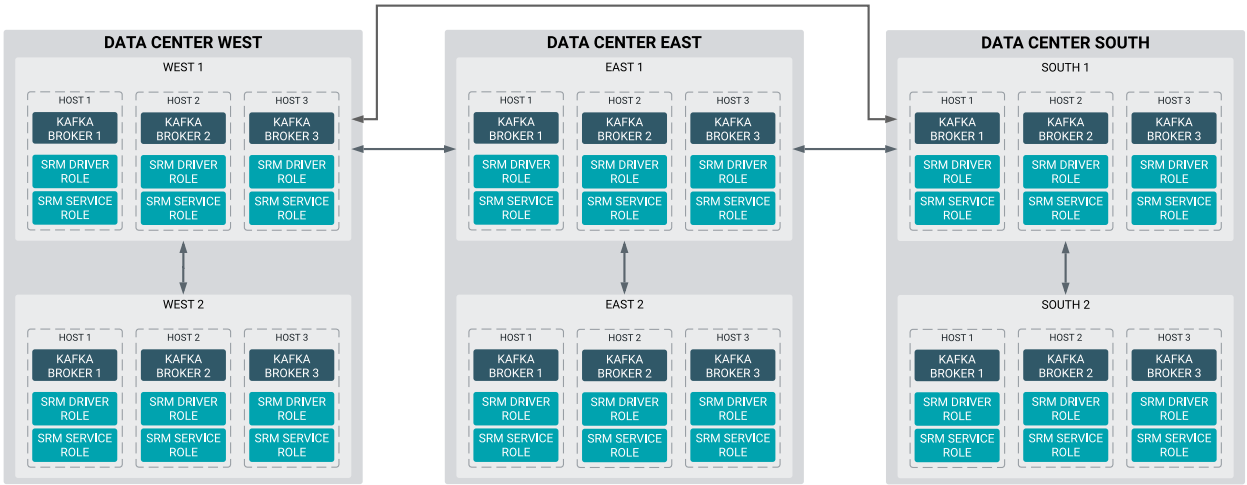
Review the cross data center replication example to understand how you can configure and start replication with Streams Replication Manager in a deployment with three data centers that each have two Kafka clusters.

About this task

In more advanced deployments, you may have multiple Kafka clusters in each of several data centers. To prevent creating a fully-connected mesh of all Kafka clusters, Cloudera recommends leveraging a single Kafka cluster in each data center for cross data center replication.

This example demonstrates the steps required to configure the deployment shown below. Additionally, it provides example commands that start bidirectional replication of all topics within each data center as well as example commands that replicate a single topic across all data centers

Figure 2: Cross Data Center Replication of Multiple Clusters



Before you begin

- The following list of steps assume that all clusters are unsecured.
- The following list of steps assume that both Streams Replication Manager Service and Driver roles are running on all Kafka broker hosts. Additionally, both roles have a single target which is their co-located cluster (the cluster they are running in).
- Clusters West 1, East 1, and South 1 are collectively referred to as primary clusters, while clusters West 2, East 2, and South 2 are collectively referred to as secondary clusters.
- Steps 1 through 8 must be carried out on each of the clusters individually. That is, the properties presented in these steps must be configured on all clusters. However, how you configure the properties for each individual cluster (the values that you set) will be different. The steps provide an explanation for each configuration and provide explicit examples for the clusters in Data Center West.
- The following list of steps assume that the `DefaultReplicationPolicy` is in use.

Procedure

1. Define the external Kafka cluster:

External clusters are configured with Kafka credentials. On each primary cluster you need to create a total of three Kafka credentials. Two representing the other two primary clusters and another one representing the secondary cluster that is in the same data center. On each secondary cluster you need to create a single Kafka credential representing the primary cluster that is running in the same data center.

a) In Cloudera Manager, go to Administration External Accounts .
b) Go to the Kafka Credentials tab.
c) Click Add Kafka credentials.
d) Configure the Kafka credentials.

For example, in West 1, you need to create credentials for East 1, South 1, and West 2. In West 2, you need to create a credential for West 1.

- West 1

For East 1 Credential

```
Name=east1
Bootstrap servers=east1-host1.cloudera.com:9092, east1-host2.clou
dera.com:9092, east1-host2.cloudera.com:9092
Security protocol=PLAINTEXT
```

For South 1 Credential

```
Name=south1
Bootstrap servers=south1-host1.cloudera.com:9092, south1-host2.c
loudera.com:9092, south1-host2.cloudera.com:9092
Security protocol=PLAINTEXT
```

For West 2 Credential

```
Name=west2
Bootstrap servers=west2-host1.cloudera.com:9092, west2-host2.clou
dera.com:9092, west2-host2.cloudera.com:9092
Security protocol=PLAINTEXT
```

- West 2

For West 1 Credential

```
Name=west1
```

```
Bootstrap servers=west1-host1.cloudera.com:9092, west1-host2.clou
dera.com:9092, west1-host2.cloudera.com:9092
Security protocol=PLAINTEXT
```

e) Click Add.

If credential creation is successful, a new entry corresponding to the Kafka credential you specified appears on the page.

2. Define the co-located Kafka cluster:
   a) In Cloudera Manager, go to Clusters and select the Streams Replication Manager service.
   b) Go to Configuration.
   c) Find and enable the Kafka Service property.
   d) Find and configure the Streams Replication Manager Co-located Kafka Cluster Alias property.

   The alias you configure represents the co-located cluster. Enter an alias that is unique and easily identifiable.

   For example:

   - West 1

     ```
     west1
     ```

   - West 2

     ```
     west2
     ```

3. Add the clusters defined with Kafka credentials to Streams Replication Manager's configuration:
   a) Find and configure the External Kafka Accounts property.
   b) Click the add button and add new lines for each Kafka credential you created.
   c) Add the names of all Kafka credentials.

   Add each Kafka credential that you created in the cluster that you are configuring. When configured correctly, the property will include three credential names in each primary and a single credential name in each secondary cluster. Each Kafka credential must be added to a new line.

   For example, in West 1 you must add the names of the Kafka credentials that you created in West 1:

   ```
   east1
   south1
   west2
   ```

   In West 2, you must add a the name of the single credential created for West 1:

   ```
   west1
   ```

4. Find and configure the Streams Replication Manager Cluster alias property.

   Add all cluster aliases to this property. This includes the aliases present in both the External Kafka Accounts and Streams Replication Manager Co-located Kafka Cluster Alias properties. Delimit the aliases with commas. For example:

   - West 1

     ```
     west1, west2, east1, south1
     ```

   - West 2

     ```
     west1, west2
     ```

5. Add and enable replications:
   a) Find the Streams Replication Manager's Replication Configs property.
   b) Click the add button and add new lines for each unique replication you want to add and enable.

c) Add and enable your replications.

In the case of this example, you need to enable both cross data center replication and replication within each data center. This can be done by configuring three replications in each primary cluster and a single replication in each of the secondary clusters. Each of the replications that you configure should target the cluster that you are configuring. For example:

- West 1

```
east1->west1.enabled=true
south1->west1.enabled=true
west2->west1.enabled=true
```

- West 2

```
west1->west2.enabled=true
```

6. Click Save Changes.
7. Restart the Streams Replication Manager service.
8. Start replication with the srm-control tool:

The following command examples demonstrate how you can start replication of a single topic across all data centers and how you can replicate all topics within each data center.



Important: The following commands assume that the `DefaultReplicationPolicy` is in use. Running the following commands with the `IdentityReplicationPolicy` would result in replication loops.

a) SSH into one of the hosts in West 1 and run the following commands:

Replicate topic1 across data centers.

```
srm-control topics --source east1 --target west1 --add topic1,east2.topi
c1
srm-control topics --source south1 --target west1 --add topic1,south2
.topic1
```

Replicate all topics within the data center.

```
srm-control topics --source west1 --target west2 --add ".*"
srm-control topics --source west2 --target west1 --add ".*"
```

b) SSH into one of the hosts in East 1 and run the following commands:

Replicate topic1 across data centers.

```
srm-control topics --source west1 --target east1 --add topic1,west2.topi
c1
srm-control topics --source south1 --target east1 --add topic1,south2
.topic1
```

Replicate all topics within the data center.

```
srm-control topics --source east1 --target east2 --add ".*"
srm-control topics --source east2 --target east1 --add ".*"
```

c) SSH into one of the hosts in South 1 and run the following commands:

Replicate topic1 across data centers.

```
srm-control topics --source west1 --target south1 --add topic1,west2.top
ic1
srm-control topics --source east1 --target south1 --add topic1,east2.top
ic1
```

Replicate all topics within the data center.

```
srm-control topics --source south1 --target south2 --add ".*"
srm-control topics --source south2 --target south1 --add ".*"
```

Enabling reverse checkpointing

Learn how to enable the reverse checkpointing feature. This feature minimizes message duplication in failback scenarios.

Before you begin

The reverse checkpointing feature expects that the `DefaultReplicationPolicy` is used, and so replica topic names are prefixed with the source cluster alias. For this reason, reverse checkpointing is incompatible with prefixless replication through the `IdentityReplicationPolicy`. Make sure that the Enable Prefixless Replication property is deselected in Cloudera Manager.

Procedure

1. In Cloudera Manager, select Streams Replication Manager.
2. Go to Configuration.
3. Find and select the Reverse Checkpointing Enabled property in Cloudera Manager.
4. Enable bidirectional replication.
   a) Find the Streams Replication Manager's Replication Configs property.
   b) Click the add button, and add the following lines:

   ```
   [***SOURCE CLUSTER ALIAS***]->[***TARGET CLUSTER ALIAS***]=true
   [***TARGET CLUSTER ALIAS***]->[***SOURCE CLUSTER ALIAS***]=true
   ```
5. Enter a Reason for change, and then click Save Changes to commit the changes.
6. Restart Streams Replication Manager.
7. Open a terminal on a host where the Streams Replication Manager client is installed.
8. Using the srm-control tool, perform the following steps:
   a) Make sure that the required topics are added to the replication allowlist.

   ```
   srm-control topics --source [***SOURCE CLUSTER ALIAS***] --targe
   t [***TARGET CLUSTER ALIAS***] --add [***TOPIC***]
   ```
   b) Add the topics you want to be reverse checkpointed.

   ```
   srm-control reverse-checkpointed-topics --source [***TARGET CLUSTER
    ALIAS***] --target [***SOURCE CLUSTER ALIAS***] --add [***SOURCE
    CLUSTER ALIAS***].[***TOPIC***]
   ```

   Replace *[***TOPIC***]* with the name of the original topic. The full value to add is the replica topic name as it appears on the target (backup) cluster, which is the original topic name prefixed with the source cluster alias. For example, if your primary cluster alias is *[***P***]* and your original topic is Payments, add P.Payments.

c) Enable consumer group replication in both directions.

```
srm-control groups --source [***SOURCE CLUSTER ALIAS***] --targe
t [***TARGET CLUSTER ALIAS***] --add [***CONSUMER GROUP***]
```

```
srm-control groups --source [***TARGET CLUSTER ALIAS***] --targe
t [***SOURCE CLUSTER ALIAS***] --add [***CONSUMER GROUP***]
```

Results

Reverse checkpointing is enabled, and offsets are tracked for the specified topics and groups in both directions.

What to do next

By default, consumer offset sync from checkpoints to the __consumer_offsets topic is not automatic. To enable automatic consumer offset syncing, configure the following properties in the Streams Replication Manager's Replication Configs property in Cloudera Manager:

- sync.group.offsets.enabled — Enables automatic consumer offset syncing from checkpoints.
- sync.group.offsets.interval.seconds — Controls the interval at which offsets are synced.

The following properties are also essential for controlling message duplication after a failover or failback:

- offset.lag.max — Controls the maximum offset lag. Affects the amount of records reprocessed after a failover or failback.
- offset.flush.interval.ms — Controls how often an offset sync is produced, even if offset.lag.max is not reached.