Cloudera Runtime 7.2.11

Configuring Streams Replication Manager

Date published: 2019-09-13
Date modified: 2021-09-09



https://docs.cloudera.com/

Legal Notice

© Cloudera Inc. 2025. All rights reserved.

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

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

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

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

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

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

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

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

Contents

Enable high availability for Streams Replication Manager 4

Defining and adding clusters for replication 4
Defining external Kafka clusters 6
Defining co-located Kafka clusters using a service dependency 10
Defining co-located Kafka clusters using Kafka credentials 11
Adding clusters to SRM's configuration 15

Configuring replications 16

Configuring the driver role target clusters 16

Configuring the service role target cluster 17

Configuring properties not exposed in Cloudera Manager 18

Configuring replication specific Kafka Connect REST servers 19

Configuring automatic group offset synchronization 20

New topic and consumer group discovery 21

Configuration examples 21
Bidirectional replication example of two active clusters 21
Cross data center replication example of multiple clusters 24

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

In CDP Public Cloud this can be done when provisioning a new cluster with Data Hub.

Note: Expect an increased load when running SRM in high availability mode.

Enable high availability mode in CDP Public Cloud

In CDP Public Cloud, high availability mode can be enabled by provisioning a Data Hub cluster that has multiple instances of the SRM driver and service. In terms of high availability, the default Streams Messaging cluster definitions behave in the following way:

Streams Messaging Light Duty

High availability mode is enabled by default. Any cluster that you provision with this definition will start SRM in high availability mode. Note that it is only the driver that is started in high availability mode, the service role is only provisioned on a single host. This however, is still considered as a highly available deployment of SRM.

Streams Messaging Heavy Duty

In the heavy duty definition, SRM has its own host group. However, by default, the SRM host group is not provisioned. When provisioning a cluster with the heavy duty definition, you can decide how many nodes this host group should have. To provision SRM in high availability mode with this cluster definition, you have to set the instance count of the SRM nodes host group to at least 2.

For more information on creating a Streams Messaging cluster with Data Hub, see Creating your first Streams Messaging cluster.

Related Information

Creating your first Streams Messaging cluster

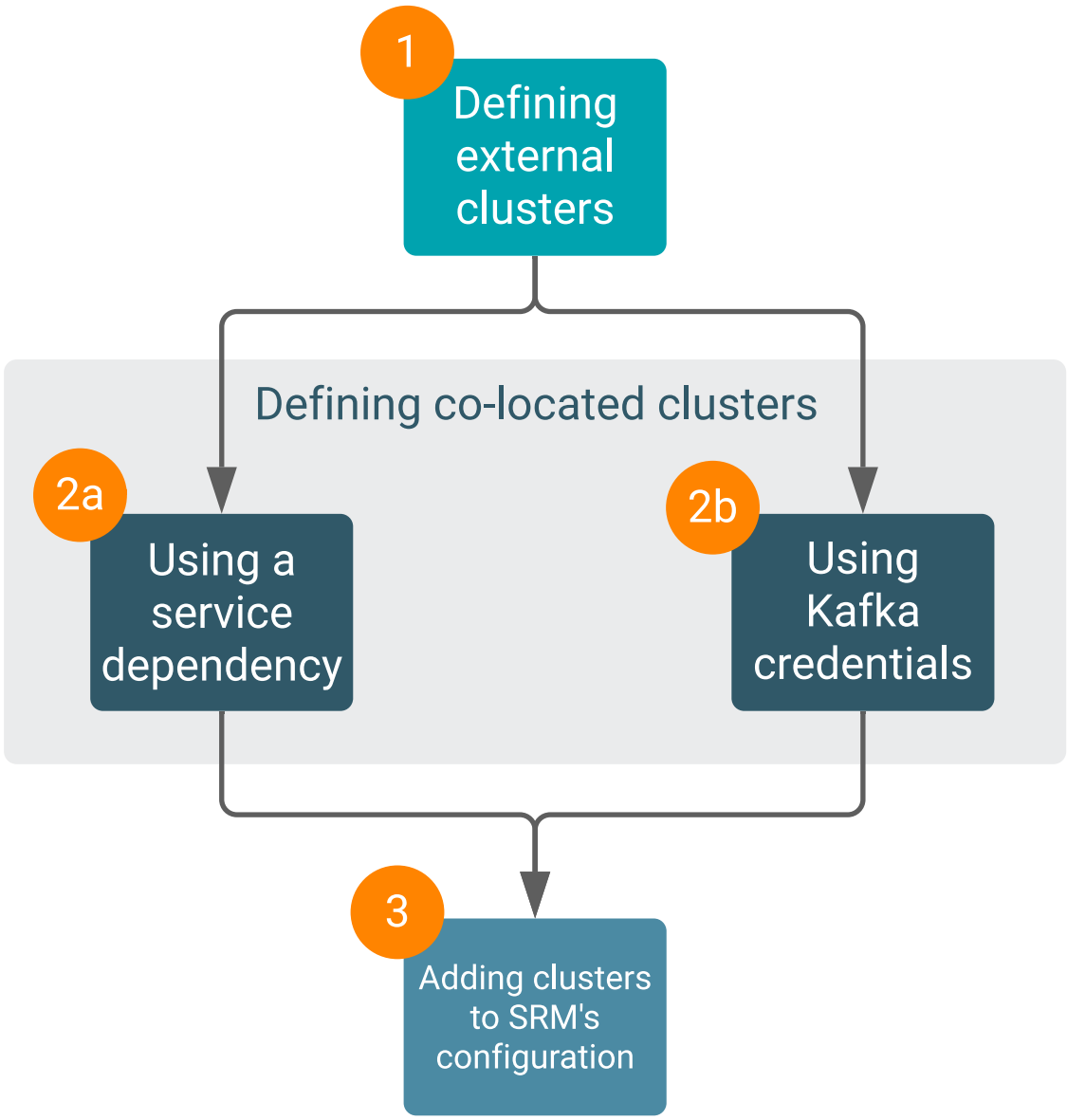
Streams Replication Manager Driver

Task architecture and load-balancing

Defining and adding clusters for replication

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

In order for SRM to replicate data between clusters, you have to define these clusters in Cloudera Manager and add them to the SRM 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 SRM 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 SRM can use to access a specific cluster.

Kafka credentials are created and configured in Cloudera Manager on the AdministrationExternal AccountsKafka 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 SRM.

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 SRM, 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 SRM 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.

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 Kerberos for authentication.
- The co-located Kafka cluster uses an authentication mechanism that is not Kerberos, for example PLAIN.

3 Adding clusters to SRM's configuration

Once both external and co-located clusters are defined, you must add all defined clusters to SRM'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 (SRM), 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 SRM 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 AdministrationExternal 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 SRM hosts and SRM 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 SRM hosts and SRM 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 SRM hosts and SRM 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 (SRM), 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 SRM service and the co-located Kafka cluster, specifying an alias for the co-located cluster, and configuring security related properties.

About this task

Important: Due to a known issue, defining co-located Kafka clusters using a service dependency is not supported for Kerberos enabled co-located Kafka clusters. Follow the instructions available in Defining co-located Kafka clusters using Kafka credentials on page 11 to define a Kerberized co-located Kafka cluster. For more information regarding this issue, see OPSAPS-61814 in Known Issues in Streams Replication Manager.

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 cluster uses TLS/SSL, enable the following properties:
      - Enable TLS/SSL for SRM Driver
      - Enable TLS/SSL for SRM Service

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 SRM's configuration.

Related Information

Adding clusters to SRM's configuration

Defining co-located Kafka clusters using Kafka credentials

Before you can start replicating data with SRM, 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 AdministrationExternal 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 SRM hosts and SRM 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 SRM hosts and SRM 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 SRM hosts and SRM must be able to access these files.

Results

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

What to do next

Add both external and co-located Kafka clusters to SRM's configuration.

Related Information

Adding clusters to SRM's configuration

Adding clusters to SRM's configuration

After both the external and co-located are defined, you must add them to the Streams Replications Manager (SRM) 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 SRM service's configuration. This is required because SRM 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 SRM'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 SRM service.

Results

Both external and co-located Kafka clusters are added to SRM'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 (SRM). 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, SRM 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. A single target 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 only be installed on one host per cluster.

Each instance of the service is associated with a single target cluster. The target is the cluster that the service gathers and exposes metrics from. Because each instance of the service can only target and expose metrics from a single cluster, monitoring multiple clusters requires the deployment of multiple instances of the service.

The target cluster of the service is 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 a 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 the target cluster alias. For example:

```
secondary
```

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

Results

The service target is set. The service gathers and exposes metrics from the specified cluster.

Configuring properties not exposed in Cloudera Manager

There are a number of configuration properties that Streams Replication Manager (SRM) 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, SRM accepts number of additional properties including SRM 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 SRM properties, their configuration and hierarchy

Configuring replication specific Kafka Connect REST servers

The SRM Driver role starts a dedicated Kafka Connect REST server for each replication that you set up and configure. These REST servers make communication possible between the different instances of the SRM driver. Communication between the driver instances in turn 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 each replication's dedicated REST server that is set up by the driver.

REST server properties can be configured through the Streams Replication Manager's Replication Configs property by using two specific prefixes. These prefixes are the following:

*****ALIAS***->***ALIAS***.worker.**

This prefix can be used to set any Kafka Connect REST server property for a replication's dedicated REST server. The first element of the prefix determines which replication's REST server is being configured. For example, the primary->secondary.worker. prefix can be used to configure the prim ary->secondary replication's REST server. While configuration of all REST server properties is supported, the main use case for this prefix is to set the ports of the REST servers. The reason for this is that by default the REST servers will bind to port 0, that is, any available port. This behaviour may not be suitable for your deployment.



Important: The rest.host.name and rest.port properties should not be used with this prefix. Use the listeners property instead if you want to configure ports.

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:

```
primary->secondary.worker.listeners=HTTPS://myhost:8084
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 automatic group offset synchronization

Automatic group offset synchronization is a feature in Streams Replication Manager (SRM) that automates the export and application of translated consumer group offsets. 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

SRM automatically translates consumer group offsets between clusters. While the offset mappings are created by SRM, they are not applied by default to the consumer groups of the target cluster. As a result, by default, migrating consumer groups from one cluster to another involves running the srm-control offsets and kafka-consumer-groups tools. The srm-control offsets tool exports translated offsets, kafka-consumer-groups resets and applies the translated offsets on the target cluster.

This process can be automated by enabling automatic group offset synchronization. If automatic group offset synchronization is enabled, the translated group offsets of the source cluster are automatically exported from the source cluster and are applied on the target cluster (they are written to the __consumer_offsets topic). If you choose to enable this feature, running srm-control offsets and kafka-consumer-groups is not required to migrate consumer groups. You only need to restart and redirect consumers to consume from the new cluster.

Although automatic group offset synchronization can simplify migrating consumer groups, ensure that you understand the following about its behavior:

- Automatic group offset synchronization does not fully automate a failover or failback process. It only allows you to skip certain manual steps required in the process. Consumers must be restarted and redirected to the new cluster even if the feature is enabled.
- Offsets are synced at a configured interval. As a result, it is not guaranteed that the latest translated offsets are applied. If you want to have the latest offsets applied, Cloudera recommends that you export and apply consumer group offsets manually instead. The exact interval depends on sync.group.offsets.interval.seconds and emit.checkpoints.interval.seconds.
- The checkpointing frequency configured for SRM can have an effect on the group offset synchronization frequency. The frequency of group offset synchronization can be configured with sync.group.offsets.interval.seconds. However, specifying an interval using this property might not result in the offsets being synchronized at the set frequency. This depends on how emit.checkpoints.interval is configured. The emit.checkpoints.interval property specifies how frequently offset information is fetched (checkpointing). Because offset synchronization can only happen after offset information is available, the frequency configured with the emit.checkpoints.interval might introduce additional latency. For example, assume that you set offset synchronization to 60 seconds (default), but have checkpointing set to 120 seconds. In a case like this, offset synchronization will happen every 60 seconds, but because offset information is only refreshed every 120 seconds, in practice offsets are only synchronized every 120 seconds.
- Offsets are only synchronized for the consumers that are inactive in the target cluster. This is done so that the SRM does not override the offsets in the target cluster.

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 only 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, SRM automatically exports the translated offsets of the configured source clusters and applies them to the configured target clusters.

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

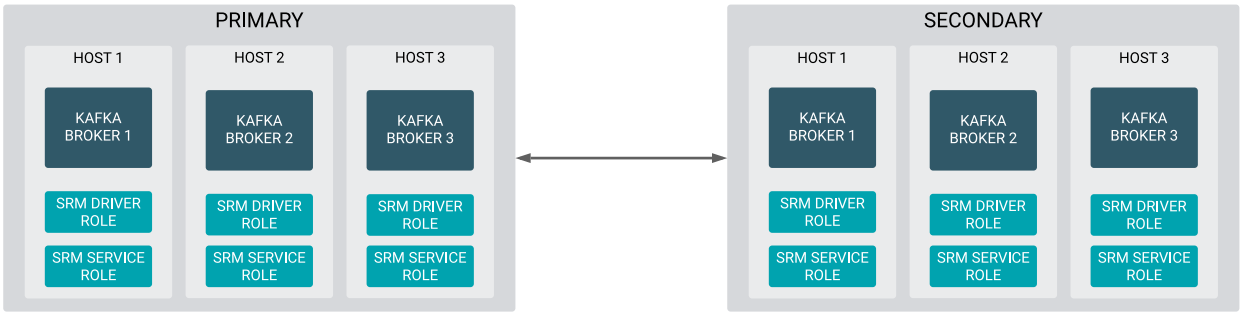
Review the bidirectional replication example to learn how you can configure and start replication with Streams Replication Manager in a deployment with two active clusters configured with bidirectional replication.

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 SRM services. The steps highlight when the configuration differs from cluster to cluster or if it is identical on all clusters.

Procedure

1. Define external clusters:
   a) In Cloudera Manager, go to AdministrationExternal 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 SRM'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:

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

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

Cross data center replication example of multiple clusters

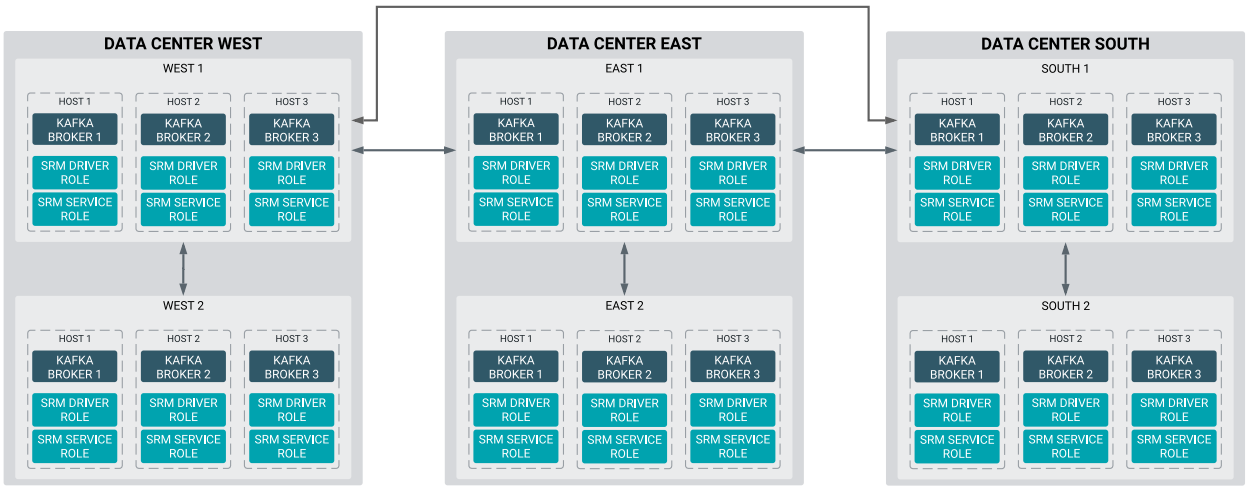
Review the cross data center replication example to understand how you can configure and start replication with Streams Replication Manager in a deployment with three data centers that each have two Kafka clusters.

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.

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 AdministrationExternal 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 SRM'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 SRM 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.

   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 ".*"
```