

Using SRM

Date published: 2019-09-20

Date modified: 2019-09-20



Legal Notice

© Cloudera Inc. 2022. 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

SRM Command Line Tools.....	4
srm-driver.....	4
Options Reference.....	4
srm-control.....	4
Topics and Groups Subcommand.....	4
Offsets Subcommand.....	6
Options reference.....	7
srm-service.....	8
Options Reference.....	8
Monitoring Replication with Streams Messaging Manager.....	9
Replicating Data.....	9
How to Set up Failover and Failback.....	10
Configure SRM for Failover and Failback.....	10
Migrating Consumer Groups Between Clusters.....	12

SRM Command Line Tools

Overview of the command line tools shipped with SRM.

srm-driver

Learn how to use the `srm-driver` command line tool which is used to launch SRM.

SRM is launched with the `srm-driver` command line tool. The driver is responsible for connecting to the specified clusters and performing replication between them. The driver uses a configuration file to identify which clusters it should connect to. When not specified otherwise, it uses the default configuration file. If required, you can use the `--config` option to specify a different configuration file.

Run the driver with the following command:

```
srm-driver
```

By default the driver will read from and write to all clusters specified in the configuration file. Optionally you can use the `--clusters` option which is used to specify the clusters that the driver should target or in other words write to.

When the driver is started with the `--clusters` option it will still connect to and read data from all clusters specified in the configuration file, but will only write data to the clusters specified with the `--clusters` option. This allows you to distribute replication workloads.

Target specific clusters with the following command:

```
srm-driver --clusters [CLUSTER_1] [CLUSTER_2]
```

Although the driver launches SRM, it does not kick off replication. Data will only be replicated between clusters once the allowlist is populated with either topics or groups using the `srm-control` tool.

Options Reference

A collection of all options and their descriptions for the `srm-driver` command line tool.

Option	Description
<code>-h, --help</code>	Shows the help message.
<code>--clusters</code>	Specifies the target clusters for this node.
<code>--config</code>	Specifies the SRM configuration file to use.

srm-control

Learn how to use the `srm-control` command line tool which is used to manage replication of topics and consumer groups.

The `srm-control` tool enables users to manage replication of topics and consumer groups. The tool has three subcommands `topics`, `groups`, and `offsets`. The `topics` subcommand is used to control which topics are replicated. The `groups` subcommand is used to control which consumer groups are replicated. The `offsets` subcommand is used to export translated offsets for a `source->target` cluster pair.

Topics and Groups Subcommand

Learn how to use the `topics` and `groups` subcommand of the `srm-control` command line tool.

The topics and groups subcommands are used to manipulate the topic or group allowlists (whitelists) and denylists (blacklists). Both subcommands support the same set of command options.

Add topics or groups to a whitelist:

```
srmsrm-control topics --source [SOURCE_CLUSTER] --target [TARGET_CLUSTER] --add [TOPIC1],[TOPIC2]
```

```
srmsrm-control groups --source [SOURCE_CLUSTER] --target [TARGET_CLUSTER] --add [GROUP1],[GROUP2]
```

Remove topics or groups from a whitelist:

```
srmsrm-control topics --source [SOURCE_CLUSTER] --target [TARGET_CLUSTER] --remove [TOPIC1],[TOPIC2]
```

```
srmsrm-control groups --source [SOURCE_CLUSTER] --target [TARGET_CLUSTER] --remove [GROUP1],[GROUP2]
```

Add topics or groups to a denylist:

```
srmsrm-control topics --source [SOURCE_CLUSTER] --target [TARGET_CLUSTER] --add-blacklist [TOPIC1],[TOPIC2]
```

```
srmsrm-control groups --source [SOURCE_CLUSTER] --target [TARGET_CLUSTER] --add-blacklist [GROUP1],[GROUP2]
```

Remove topics or groups from a denylist:

```
srmsrm-control topics --source [SOURCE_CLUSTER] --target [TARGET_CLUSTER] --remove-blacklist [TOPIC1],[TOPIC2]
```

```
srmsrm-control groups --source [SOURCE_CLUSTER] --target [TARGET_CLUSTER] --remove-blacklist [GROUP1],[GROUP2]
```

Specifying topics or groups is also possible with regular expressions. The following example adds all topics to the allowlist, meaning that every topic on the source cluster will be replicated to the target cluster.

```
srmsrm-control topics --source [SOURCE_CLUSTER] --target [TARGET_CLUSTER] --add ".*" 
```

In addition to adding or removing items, you can also use the tool to look at the contents of a deny or allowlist.

```
srmsrm-control topics --source [SOURCE_CLUSTER] --target [TARGET_CLUSTER] --list
```

The topics and groups subcommands read configuration properties from a configuration file. When not specified otherwise, the subcommands use the default configuration file. If required, you can use the `--config` option to specify a different configuration file.

```
srmsrm-control --config [path/to/srm.properties] topics --source [SOURCE_CLUSTER] --target [TARGET_CLUSTER] --list
```

Client Override Options

The topics and groups subcommands support a number of client override options. Client override options allow users to temporarily specify or override configuration properties used for replication. These options also enable users to

issue `srm-control` commands even if the SRM's configuration file is not available on the host that the command is being issued from. While it is possible to specify a range of properties with the client override options, and they can prove to be a powerful tool in certain scenarios, Cloudera recommends that you use the `srm.properties` configuration file to manage client configuration options.

The following client override options are available:

- `--bootstrap-servers`: Specifies the bootstraps servers.
- `--producer-props`: Specifies producer configuration properties.
- `--consumer-props`: Specifies consumer configuration properties.
- `--props`: Specifies client configuration properties.



Note:

Client override options always take precedence over the `srm.properties` file. Additionally, the `--producer-props` and `--consumer-props` options take precedence over the `--props` option.

A simple example of using client override options is when you want to change the bootstrap server. This can be done in two ways.

You can specify the bootstrap server with the `--bootstrap-servers` option.

```
srm-control --bootstrap-servers localhost:9092 topics --source [SOURCE_CLUSTER] --target [TARGET_CLUSTER] --list
```

Alternatively, you can also use the `--props` option together with the `bootstrap.servers` Kafka property to define the bootstrap server.

```
srm-control --props bootstrap.servers=localhost:9092 topics --source [SOURCE_CLUSTER] --list
```

Offsets Subcommand

Learn how to use the `offsets` subcommand of the `srm-client` command line tool.

SRM automatically translates consumer group offsets between clusters. The offset mappings are created by SRM, but are not applied to the consumer groups of the target cluster directly. Consumers can be migrated from one cluster to another without losing any progress by using the `offsets` subcommand on the target cluster to export the translated offsets of the source cluster. For example:

```
srm-control offsets --source [SOURCE_CLUSTER] --target [TARGET_CLUSTER] --group [GROUP1] --export > out.csv
```

Exported offsets can then be applied to the consumer groups in the target cluster with the `kafka-consumer-groups` tool. For detailed steps on cluster migration, see *Migrating Consumer Groups Between Clusters*.

The `offsets` subcommand reads configuration properties from a configuration file. When not specified otherwise, it uses the default configuration file. If required, you can use the `--config` option to specify a different configuration file.

```
srm-control --config [path/to/srm.properties] offsets --source [SOURCE_CLUSTER] --target [TARGET_CLUSTER] --group [GROUP1] --export > out.csv
```

Client Override Options

The `offsets` subcommand supports client override options. Client override options allow users to temporarily specify or override configuration properties. These options also enable users to issue `srm-control` commands even if the SRM's configuration file is not available on the host that the command is being issued from. While it is possible to specify a range of properties with the client override options, and they can prove to be a powerful tool in

certain scenarios, Cloudera recommends that you use the `srm.properties` configuration file to manage client configuration options.

The following client override options are available:

- `--bootstrap-servers`: Specifies the bootstraps servers.
- `--props`: Specifies client configuration properties.



Note:

Client override options always take precedence over the `srm.properties` file.

A simple example of using client override options is when you want to change the bootstrap server. This can be done in two ways.

You can specify the bootstrap server with the `--bootstrap-servers` option.

```
srm-control --bootstrap-servers localhost:9092 offsets --source [SOURCE_CLUSTER] --group [GROUP] --export > out.csv
```

Alternatively, you can use the `--props` option together with the `bootstrap.servers` Kafka property to define the bootstrap server.

```
srm-control --props bootstrap.servers=localhost:9092 offsets --source [SOURCE_CLUSTER] --group [GROUP] --export > out.csv
```

Related Information

[How to Set up Failover and Failback](#)

[Migrating Consumer Groups Between Clusters](#)

Options reference

A collection of all options and their descriptions for the `srm-control` command line tool.

Table 1: Topics and groups subcommand properties

Options	Description
<code>-h, --help</code>	Shows the help message.
<code>--source</code>	Specifies the source cluster
<code>--target</code>	Specifies the target cluster
<code>--config</code>	Specifies the SRM configuration file to use.
<code>--add</code>	Specifies topics or groups to add to the allowlist
<code>--remove</code>	Specifies topics or groups to remove from the allowlist
<code>--add-blacklist</code>	Specifies topics or groups to add to the denylist
<code>--remove-blacklist</code>	Specifies topics or groups to remove from the denylist
<code>--list</code>	Lists current allowlist and denylist
<code>--bootstrap-servers</code>	Specifies the bootstraps servers
<code>--producer-props</code>	Specifies producer configuration properties
<code>--consumer-props</code>	Specifies consumer configuration properties
<code>--props</code>	Specifies client configuration properties

Table 2: Offsets subcommand properties

Option	Description
-h, --help	Shows the help message.
--source	Specifies the source cluster
--target	Specifies the target cluster
--config	Specifies the SRM configuration file to use.
--export	Export translated offsets
--group	Specifies the groups translated offsets should be exported for
--bootstrap-servers	Specifies the bootstraps servers
--props	Specifies client configuration properties

srm-service

Learn how to use the `srm-service` command line tool which is used to launch the SRM REST service and Kafka streams application.

The `srm-service` tool launches a REST service and a Kafka streams application, which enable users to monitor replications.

Each instance of the service is associated with a specific target cluster. The tool uses a properties file to acquire connection information for the target cluster. When not specified otherwise, it uses the default configuration file. If required, you can use the `--config` option to specify a different properties file.

Additionally, a second configuration file, `srm-service.yaml`, is required for settings specific to the service, for example which HTTP ports to use. Use the `--yaml` option to specify which `srm-service.yaml` file to use. As with `--config`, the `--yaml` option is only required if you are storing the file in a non-default location. For .rpm or .deb based installations, the yaml file is read from `/opt/streams-replication-manager/config/srm-service.yaml`. For ZIP or TAR based installations, the service will default to `$SRM_CONF_DIR/srm-service.yaml`.

Run the SRM services with following command:

```
srm-service --target [TARGET_CLUSTER]
```

The yaml file supports Dropwizard properties as well as changing the rpc port used by `srm-service` nodes for communication. The default port is 6669. To change the default port, add the following to the yaml configuration file:

```
rpcPort: 8081
server:
  applicationConnectors:
    - type: http
      port: 8081
```

Related Information

[Dropwizard Configuration Reference](#)

Options Reference

A collection of all options and their descriptions for the `srm-service` command line tool.

Option	Description
-h, --help	Shows the help message.
--clusters	Specifies the target clusters for this node.

Option	Description
--config	Specifies the SRM configuration file to use.
--yaml	Specifies the Dropwizard configuration file to use.
--target	Target cluster aliases.

Monitoring Replication with Streams Messaging Manager

Learn about monitoring SRM replication with Streams Messaging Manager.

Users have the ability to connect SRM with Streams Messaging Manager (SMM) and monitor replications through the SMM UI. This is achieved with the Kafka Streams application and the REST API that come bundled with SRM. The Kafka Streams application calculates and aggregates replication metrics, the REST API exposes these metrics. SMM uses the REST API to display aggregated metrics to the end users, enabling monitoring as a result. Monitoring replication flows in SMM is available starting with version 2.0.0.

For more information regarding the requirements and setup of SRM with SMM, see [Monitoring Kafka Cluster Replication using SMM](#) in the SMM guide.

Related Information

[Monitoring Cluster Replications Overview](#)

Replicating Data

A step by step guide on how to start replicating data between Kafka clusters with SRM.

About this task

In SRM a replication can be set up with the help of the `srm-driver` and `srm-control` tools.

Before you begin

Verify that SRM is configured correctly. Make sure that connection information for each Kafka cluster is added as well as at least one `source->target` replication is specified and enabled.

Procedure

1. Launch SRM with the `srm-driver` tool.

```
srm-driver
```

Alternatively, if you want to use a properties file other than the default, you can do so with the `--config` option.

```
srm-driver --config [path/to/srm.properties]
```

The driver connects to the clusters defined in your configuration. However, data replication is not yet started.

2. Update the topics whitelist to start data replication.

```
srm-control topics --source [SOURCE_CLUSTER] --target [TARGET_CLUSTER] --add [TOPIC1],[TOPIC2]
```

Alternatively, if you want to use a properties file other than the default, you can do so with the `--config` option.

```
srm-control --config [path/to/srm.properties] topics --source [SOURCE_CLUSTER] --target [TARGET_CLUSTER] --add [TOPIC1],[TOPIC2]
```



Note: If required, instead of listing the topics that you want to add, you can also use regular expressions to add multiple topics with one command.

Results

The topics you specify with the `--add` option are added to the topic whitelist and are replicated to the specified target cluster.

How to Set up Failover and Failback

Learn how to prepare for failover and failback scenarios with SRM.

If a primary Kafka cluster is temporarily unavailable, you can migrate mission-critical workloads to a backup Kafka cluster (failover). When the primary cluster is restored, you can migrate back (failback). To prepare for this scenario, ensure SRM is configured with bidirectional replication of mission-critical consumer groups and topics. Then in the case of a disaster scenario you can migrate consumer groups between clusters.

Related Information

[Offsets Subcommand](#)

Configure SRM for Failover and Failback

Learn how to configure SRM for failover and failback.

About this task

To prepare for a failover or failback scenario you have to set up SRM with bidirectional replication. Additionally, you have to make sure that all mission critical topics and consumer groups are whitelisted on both the primary and backup clusters.

Procedure

1. Set up bidirectional replication between clusters:



Note:

The following example contains the minimum required properties only. For a more in-depth configuration example for a cluster setup with bidirectional replication, see [Configuration Examples](#).

Configure the following configuration properties based on your environment.

```
# Specify clusters and add connection information

clusters = primary, secondary
primary.bootstrap.servers = primary_host1:9092, primary_host2:9092, primary_host3:9092
```

```
secondary.bootstrap.servers = secondary_host1:9092, secondary_host2:9092,
secondary_host3:9092

# Enable bidirectional replication
primary->secondary.enabled = true
secondary->primary.enabled = true
```

2. Whitelist required consumer groups and topics on the primary cluster.

a) Whitelist groups:

```
srm-control groups --source [PRIMARY_CLUSTER] --target
t [SECONDARY_CLUSTER] --add [GROUP1],[GROUP2]
```

a) Whitelist topics:

```
srm-control topics --source [PRIMARY_CLUSTER] --target
t [SECONDARY_CLUSTER] --add [TOPIC1],[TOPIC2]
```

3. Whitelist required remote topics and consumer groups on the secondary cluster.



Important:

If remote topics and consumer groups are not whitelisted on the secondary cluster, a failback operation will be impossible to carry out.

a) Whitelist remote groups:

```
srm-control groups --source [SECONDARY_CLUSTER] --target
t [PRIMARY_CLUSTER] --add [GROUP1],[GROUP2]
```

b) Whitelist remote topics:

```
srm-control topics --source [SECONDARY_CLUSTER] --target
t [PRIMARY_CLUSTER] --add
dd [PRIMARY_CLUSTER.TOPIC1],[PRIMARY_CLUSTER.TOPIC2]
```

4. Verify that all required topics and consumer groups are whitelisted.

a) Verify consumer groups:

```
srm-control groups --source [PRIMARY_CLUSTER] --target
t [SECONDARY_CLUSTER] --list
```

```
srm-control groups --source [SECONDARY_CLUSTER] --target
t [PRIMARY_CLUSTER] --list
```

b) Verify topics:

```
srm-control topics --source [PRIMARY_CLUSTER] --target
t [SECONDARY_CLUSTER] --list
```

```
srm-control topics --source [SECONDARY_CLUSTER] --target
t [PRIMARY_CLUSTER] --list
```

Results

SRM is set up with bidirectional replication and all mission critical topics and consumer groups are whitelisted on both the primary and secondary clusters.

Related Information

[Configuration Examples](#)

Migrating Consumer Groups Between Clusters

Learn how to migrate consumers between clusters.

About this task

If a primary Kafka cluster is temporarily unavailable, you can migrate mission-critical workloads to a secondary Kafka cluster (failover). When the primary cluster is restored, you can migrate back (failback). The steps for migrating consumers in a failover or failback scenario are identical. However, depending on the scenario, your source and target clusters will be different. During failover you migrate consumers from primary to secondary, while during failback you migrate consumers from secondary to primary.

Before you begin

- Make sure that the clusters that you are migrating consumers between are set up with bidirectional replication.
- Verify that all mission critical consumer groups and topics, including the ones on the secondary cluster are whitelisted.

Procedure

1. Export the translated consumer group offsets of the source cluster:

```
srm-control offsets --source [SOURCE_CLUSTER] --target [TARGET_CLUSTER] --group [GROUP1] --export > out.csv
```

2. Reset consumer offsets on the target cluster:

```
kafka-consumer-groups.sh --bootstrap-server [TARGET_BROKER:PORT] --reset-offsets --group [GROUP1] --execute --from-file out.csv
```

3. Start consumers on the target cluster.

Results

Consumers automatically resume processing messages on the target cluster where they left off on the source cluster.

Related Information

[Offsets Subcommand](#)