

Securing 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

Security overview.....	4
Configuring SSL encryption.....	4
Configuring SSL encryption and authentication.....	4
Configuring authentication with Kerberos.....	5
Configuring authentication with Kerberos and encryption with SSL.....	5
Configuring security for clusters with differing security setups.....	6

Security overview

Learn how to set up security for SRM.

SRM can operate in both secure and unsecure environments, as well as a mix of both. In addition, it can also connect to clusters that have differing security setups. Configuring security can be achieved by adding the appropriate security related properties to your `srm.properties` configuration file. The properties set should match the security setup of the cluster or clusters that SRM is connecting to.

Configuring SSL encryption

Learn how to set up SSL encryption for SRM.

About this task

Configuring SSL encryption for SRM can be achieved by adding the appropriate configuration properties to your `srm.properties` configuration file.



Note: Depending on your security environment, other, optional configuration settings may be required.

Procedure

1. Specify the security protocols:

```
security.protocol = SSL
```

2. Specify truststore information:

```
ssl.truststore.location = /var/private/ssl/kafka.client.truststore.jks  
ssl.truststore.password = test1234
```

Configuring SSL encryption and authentication

Learn how to set up SSL encryption and authentication for SRM.

About this task

Configuring SSL encryption and authentication for SRM can be achieved by adding the appropriate configuration properties to your `srm.properties` configuration file.



Note: Depending on your security environment, other, optional configuration settings may be required.

Procedure

1. Specify the security protocols:

```
security.protocol = SSL
```

2. Specify truststore information:

```
ssl.truststore.location = /var/private/ssl/kafka.client.truststore.jks
```

```
ssl.truststore.password = test1234
```

3. Specify keystore information:

```
ssl.keystore.location = /var/private/ssl/kafka.client.keystore.jks  
ssl.keystore.password = test1234  
ssl.key.password = test1234
```

Configuring authentication with Kerberos

Learn how to set up Kerberos authentication for SRM.

About this task

Configuring authentication with Kerberos for SRM can be achieved by adding the appropriate configuration properties to your `srm.properties` configuration file.

Procedure

1. Specify the security protocols:

```
security.protocol = SASL_PLAINTEXT
```

2. Specify the authentication mechanism:

```
sasl.mechanism = GSSAPI
```

3. Specify the service name:

```
sasl.kerberos.service.name = kafka
```

4. Configure the JAAS.

You have two options when configuring the JAAS:

a) Embed it in the `srm.properties` file with the `sasl.jaas.config` property:

```
sasl.jaas.config = \  
com.sun.security.auth.module.Krb5LoginModule required \  
  useKeyTab=true \  
  keyTab="path/to/keytab file" \  
  storeKey=true \  
  useTicketCache=false \  
  principal="streamsrepmgr@STREAMANALYTICS.COM";
```

b) Create a JAAS configuration file containing the properties and pass its location with the `SRM_KERBEROS_OPTS` environment variable:

```
export SRM_KERBEROS_OPTS="-Djavax.security.auth.useSubjectCredsOnly=false  
-Djava.security.auth.login.config=/opt/streams-replication-manager/conf/srm-jaas.conf"
```

For convenience, this and other environment variables can be added to `/opt/streams-replication-manager/config/srm-env.sh`.

Configuring authentication with Kerberos and encryption with SSL

Learn how to set up Kerberos authentication and SSL encryption for SRM.

About this task

Configuring Kerberos authentication and SSL encryption for SRM can be achieved by adding the appropriate configuration properties to your `srm.properties` configuration file.



Note:

Depending on your security environment, other, optional configuration settings may be required.

Procedure

1. Specify the security protocols:

```
security.protocol = SASL_SSL
```

2. Specify truststore information:

```
ssl.truststore.location = /var/private/ssl/kafka.client.truststore.jks
ssl.truststore.password = test1234
```

3. Specify the authentication mechanism:

```
sasl.mechanism = GSSAPI
```

4. Specify the service name:

```
sasl.kerberos.service.name = kafka
```

5. Configure the JAAS.

You have two options when configuring the JAAS:

- a) Embed it in the `srm.properties` file with the `sasl.jaas.config` property:

```
sasl.jaas.config = \
com.sun.security.auth.module.Krb5LoginModule required \
  useKeyTab=true \
  keyTab="path/to/keytab file" \
  storeKey=true \
  useTicketCache=false \
  principal="streamsrepmgr@STREAMANALYTICS.COM";
```

- b) Create a JAAS configuration file containing the properties and pass its location with the `SRM_KERBEROS_OPTS` environment variable:

```
export SRM_KERBEROS_OPTS="-Djavax.security.auth.useSubjectCredsOnly=false \
-Djava.security.auth.login.config=/opt/streams-replication-manager/conf/srm-jaas.conf"
```

For convenience, this and other environment variables can be added to `/opt/streams-replication-manager/config/srm-env.sh`.

Configuring security for clusters with differing security setups

Learn how to set up SRM security for clusters with differing security setups.

About this task

In a complex system you might have clusters that require no encryption or authentication while others do. SRM is capable of operating in environments where the security setup of each cluster SRM is connecting to is different. Configuring security is achieved by adding the appropriate configuration properties to your `srm.properties` configuration file. When connecting SRM to clusters with non-uniform security setups, you have to add each

configuration property separately for each cluster in your system. This can be done by prepending the configuration property with the cluster name. For example, `primary.security.protocol` specifies the security protocol for the cluster named “primary”.

Procedure

1. Specify the security protocols for each cluster:

```
primary.security.protocol = SSL
backup.security.protocol = SASL_SSL
```

2. Add truststore information:

Truststore information is only required if you are using SSL encryption.

```
backup.ssl.truststore.location=/var/private/ssl/kafka.client.truststore.
jks
backup.ssl.truststore.password=test1234
primary.ssl.truststore.location=/var/private/ssl/kafka.client.truststore
.jks
primary.ssl.truststore.password=test1234
```

3. Add keystore information:

Keystore information is only required if you are using SSL authentication.

```
primary.ssl.keystore.location=/var/private/ssl/kafka.client.keystore.jks
primary.ssl.keystore.password=test1234
primary.ssl.key.password=test1234
```

4. Specify the authentication mechanism:

Specifying the authentication mechanism is only required if you are using SASL.

```
backup.sasl.mechanism = GSSAPI
```

5. Specify the service name:

Service name information is only required if you are using Kerberos.

```
primary.sasl.kerberos.service.name = kafka
backup.sasl.kerberos.service.name = srm
```

6. Configure the JAAS.

JAAS configuration is only required if you are using SASL.

If one or more of your clusters operates in a different kerberos realm, you have to specify a unique JAAS configuration for each. Configuring multiple kerberos realms is only possible by embedding the JAAS configuration in the `srm.properties` file. Passing JAAS configuration files with the `SRM_KERBEROS_OPTS` environment variable is not supported in this scenario.

```
primary.sasl.jaas.config = \
com.sun.security.auth.module.Krb5LoginModule required \
  useKeyTab=true \
  keyTab="path/to/keytab file" \
  storeKey=true \
  useTicketCache=false \
  principal="streamsrepmgr@STREAMANALYTICS.COM" ;

backup.sasl.jaas.config = \
com.sun.security.auth.module.Krb5LoginModule required \
  useKeyTab=true \
  keyTab="path/to/keytab file" \
```

```
storeKey=true \  
useTicketCache=false \  
principal="streamsrepmgr@EXAMPLE.COM";
```