Integrated Capture Points Guide

OpenMQ—SSL for JMS CP

5/10/2025

OpenMQ—SSL for JMS CP

This section provides an example of enabling SSL with the OpenMQ provider.

Outline

In general, configuration of an SSL connection consists of the following major steps:

1. Prepare the certificates.
2. Configure the JMS provider to operate in SSL mode.
3. Configure the options in Interaction Server's **jvm-options** section and add required JARs to the class path.
4. Configure the JMS Capture Point.

Configure Capture Point to use SSL (OpenMQ example)

This example assumes that an instance of Open MQ is configured and operating with a JMS Capture Point, without SSL.

The first several steps involve configuring the OpenMQ broker.

1. Generate a self-signed broker certificate:
   a. Run keytool to generate a key store (if one does not already exist) to generate a self-signed certificate:

      <OpenMQ installation dir>\mq\bin>imqkeytool

   b. Answer all the prompts and remember the chosen passwords. By default, the keystore will be called **keystore** and will be located in **<OpenMQ installation dir>\etc\mq**.
2. Add **ssljms** to active broker services:
   a. Locate the file **<OpenMQinstallation>\var\mq\instances\imqbroker\props\config.properties**.
   b. At the end of the file, add the following line: `imq.service.activelist=ssljms,admin,httpjms`
   c. Set the SSL port by adding the following line: `imq.ssljms.tls.port=1756`
   d. Restart the broker.

   The broker will prompt the user for a keystore password.
3. Update the connection factory properties: In the **.bindings** file, find the line
   {Your connection factory lookup name}/RefAddr/44/Content=
   and change it to
   `{Your connection factory lookup name}/RefAddr/44/Content=mqssl\://{your broker`

`host}\:1756`
where 1756 is the same port as that set in the broker properties. This operation can be done using the OpenMQ Administration Console by selecting the corresponding connection factory and adding `mqssl://{your broker host}:1756` to the Message Server Address List properties on its **Connection Handing** tab.

The next steps involve configuring Interaction Server.

4. Export the broker certificate to a trust store:
   a. Export the broker certificate with the following command:

      ```
      keytool -export -alias imq -keystore keystore -file openmqbroker.cer
      ```

   b. Copy the **.cer** file to Interaction Server's host and import it to a local trust store:

      ```
      keytool -import -keystore truststore.jks -file openmqbroker.cer -alias
       openmqbroker
      ```

3. Add the following to the Interaction Server **jvm-options** section:

   ```
   -Djavax.net.ssl.trustStore= {Path to the local trust store}/truststore.jks
   -Djavax.net.ssl.trustStorePassword={your local trust store password}
   -Djavax.net.ssl.trustStoreType=jks
   ```

   For debugging purposes, you can also add the following option, which prints debug information to the console:
   `-Djavax.net.debug=ssl:handshake,data,trustmanager,record`

4. Finally, configure the JMS Capture Point by adding the following to the **jms-additional-context-attributes** section:

   ```
   java.naming.security.protocol=ssl java.naming.security.authentication=simple
   ```

It should be noted that in this example, the JNDI naming service used has all of the relevant context stored in a **.bindings** file and does not have any mechanism of authorization and authentication. With other JNDI services, the user accessing JNDI may have to provide a username and a password, which can be different from the JMS connection credentials. If this is the case, the JMS Connection credentials must be specified in the JMS Capture Point **settings** section as `username` and `password`, while the JNDI username and password must be specified in the **jms-additional-context-attributes** section as `java.naming.security.principal` and `java.naming.security.credentials`, respectively.