Configuring Transport Level Security (TLS) between two MQ Queue Managers using self-signed certificates

https://www.ibm.com/support/pages/node/6986363

Date last updated: 26-Apr-2023

Miguel Rodriguez
IBM MQ Support
https://www.ibm.com/products/mq/support
Find all the support you need for IBM MQ

PURPOSE:

This document is intended to demonstrate how to configure two-way channel encryption between two MQ queue managers, that is each queue manager will authenticate the remote party before allowing the connection to succeed.

AUDIENCE:

The document is written for IBM MQ administrators that have an understanding of how to create queue managers, channels, listeners, and queues. This foundation is necessary and is the starting point in understanding how to configure channel encryption.

PREQUISITES:

You need to have 2 queue managers that have sender-receiver channels between them.
The following tutorial can be helpful:

https://www.ibm.com/support/pages/node/152555
MQ Commands to setup two-way communication between two queue managers on Sender and Receiver channels

The following is the list of steps:

1. Create two queue managers, TEST1 and TEST2.
2. Create a sender channel on TEST1 called TEST1_TO_TEST2.
3. Create a transmit queue for the sender channel called TEST2.QX
4. Create a local queue on TEST1 called TEST1.LQ
5. Create a receiver channel on TEST2 called TEST1_TO_TEST2.
6. Create a sender channel on TEST2 called TEST2_TO_TEST1.
7. Create a transmit queue for the sender channel called TEST1.QX
8. Create a local queue on TEST2 called TEST2.LQ
7. Create a receiver channel on TEST1 called TEST2_TO_TEST1.
8. Create a remote queue definition on TEST1 called TEST2.RQ
9. Create a listener for TEST1.

10. Create a remote queue definition on TEST2 called TEST1.RQ
11. Create a listener for TEST2.
12. Start each sender channel.
13. Verify channel starts and runs successfully.

DISCLAIMER:

Before beginning the following procedure, make sure that you have used the MQ command "setmqenv" set the MQ environment variables for the installation that is associated with your queue manager. Setting these environment variables will enable you to run MQ commands without providing a fully qualified path for each command.
For example:

Enter the command dspmqver. If you receive the following output, your installation is not set.

'dspmqver' is not recognized as an internal or external command

Set the installation by running the following command. Notice that you must use the leading dot, followed by a space.

For Linux/AIX:

```
. /opt/mqm/bin/setmqenv -s
```

For Windows:
<MQInstallDir>\path\bin\settmqenv -s
Such as:

```
"C:\Program Files\IBM\MQ\bin\setmqenv" -s
```

After setting environment reimplement dspmqver and now the MQ version should be reported.

Name: IBM MQ
Version: 9.3.0.0

Secondly, my queue managers are running on a Windows 11 computer. The instructions provided will work for Windows or Linux/Unix, however, the paths will be different.

Thirdly, the echo commands for Linux/Unix require quotes around the commands.
For example:

echo **"display qmgr all"** | runmqsc TEST1

<u>Since a Windows MQ Server is being used, the MQ commands WILL NOT have double quotations around them.</u>
For example:

echo **display qmgr all** | runmqsc TEST1

SUMMARY:

Once your MQ environment is created, following are the steps to configure TLS (channel encryption) for queue managers TEST1 and TEST2.

1. Create a key repository for each queue manager.
2. Confirm the SSLKEYR attribute of the queue manager points to the key repository created.
3. Create a personal self-signed certificate for each queue manager.
4. Extract the root signer certificate from the personal self-signed certificates.
5. Exchange the signer certificates for each queue manager and add each to the queue manager's key repository.
6. Select the channel encryption to be used and alter the SSLCIPH attribute of each channel.
7. Start each sender channel and confirm the channel starts successfully.
8. Use an MQ sample utility to put a message to remote queue definition. (OPTIONAL)

DETAILED PROCEDURE:

1. Create a key repository for each queue manager.

- Command line:

a. runmqakm -keydb -create -db C:\MQData\qmgrs\TEST1\ssl\key.kdb -pw p@ssw0rd -stash

b. runmqakm -keydb -create -db C:\MQData\qmgrs\TEST2\ssl\key.kdb -pw p@ssw0rd -stash

NOTE:
You can add the -expire option to allow the key repository password to expire after a number of days.
You can name the key repository whatever you would like, however, it MUST HAVE a .kdb extension

- Key Management Utility:

a. In the Windows Start Menu, Type "IBM Key" in the search bar at the top and enter "IBM Key Management".
Select the Key Management tool from your installation.
OBJ

ATTENTION:
Depending on your Windows Administration settings, you may need to explicitly run this program as an Administrator:

In Windows 11, from the Start menu select "IBM MQ" then "IBM Key Management" then "More >" and finally:
  Run as administrator

OBJ

b. In the "IBM Key Management" GUI, click on the White Page Icon:

OBJ

Select and type accordingly and press the OK button.

Key database type -	CMS
File Name -	key.kdb
Location:	C:\MQData\qmgrs\TEST1\ssl\

NOTE:
Location is the path to the queue manager's TEST1 ssl directory.
Your path to this directory may be different.
Secondly if you are not sure of the path to the directory, click on the Browse button to search and find the desired location.

OBJ

c. When the OK button is pressed, the Key Management utility will request a password and confirmation, an expiration for the key repository password and a selection to Stash the password to a file. Enter these accordingly.

NOTE:
If you do not choose an expiration time (expressed in days), the key repository password will not expire.
In order for MQ to open the key repository, it requires that you stash the password to a file.

OBJ

d. Once the OK Button is pressed the key repository will be presented to you.

Notice that at this point, there are no certificates and therefore, it will be empty.

OBJ

e. Re-implement the same steps, only this time, use the instructions for the queue manager TEST2 key repository.

NOTE:
The paths will be different due to the names of the queue managers.

2. Confirm that the SSLKEYR attribute of the queue manager points to the key repository created.

- Command line:

a. echo display qmgr sslkeyr | runmqsc TEST1

b. echo display qmgr sslkeyr | runmqsc TEST2

Starting MQSC for queue manager TEST2.

```
    1 : display qmgr sslkeyr
AMQ8408I: Display Queue Manager details.
   QMNAME(TEST2)
   SSLKEYR(C:\MQData\qmgrs\TEST1\ssl\key)
One MQSC command read.
No commands have a syntax error.
All valid MQSC commands were processed.
```

Notice that the path looks like it ends with a directory but this is not the case.
The word "key" is the stem name of the key repository files which are distinguished by their extension.
For example, on my system they are called:

```
.
04/03/2023  02:38 PM            88 key.kdb
04/03/2023  02:38 PM            80 key.rdb
04/03/2023  02:38 PM           193 key.sth
```

If the path to the key repository is not correct, use the alter command to modify it.
For example, if my key repository was called test1.kdb, I would run the following command:

echo alter qmgr sslkeyr ('C:\MQData\qmgrs\TEST1\ssl\test1') | runmqsc TEST2

NOTE:
The path for the sslkeyr attribute is within single quotes which are necessary because this is a string containing mixed case.

- MQ Explorer GUI:

a. Open the MQ Explorer App.

OBJ

b. Open the queue managers folder, right click on TEST1, and select properties.

OBJ

c. In the TEST1 Properties window, on the left-hand side, select "SSL" from the list.

OBJ

d. When the window opens review the value for the "SSL Key repository" field. In our example the path/stem did not change. If you are using a new path or a new key repository name, modify accordingly.

OBJ

3. Create a personal self-signed certificate for each queue manager.

- Command Line:

```
runmqakm -cert -create -db C:\MQData\qmgrs\TEST1\ssl\key.kdb -stashed -label
ibmwebspheremqtest1 -dn "CN=TEST1,O=My Company,C=US,OU=MQ Administration,ST=Washington" -
size 2048 -sig_alg SHA256WithRSA

runmqakm -cert -create -db C:\MQData\qmgrs\TEST2\ssl\key.kdb -stashed -label
ibmwebspheremqtest2 -dn "CN=TEST2,O=My Other Company,C=US,OU=MQ
Administration,ST=Montana" -size 2048 -sig_alg SHA256WithRSA
```

NOTE:
When your key repository's password is stashed, you will be able to use the -stashed option so that you do not have to use the -pw option and retype your password.

The label of the certificate can be anything you like, however, you will need to modify the queue manager's CERTLABL attribute with the chosen value.
The label:
ibmwebspheremqtest1
… is the default value created when your queue manager was created.

For example, if my label is:
Miguel
… then I would use these commands:

```
echo alter qmgr certlabl ('Miguel') | runmqsc TEST1
echo display qmgr certlabl | runmqsc TEST1
```

The distinguished name or dn are the unique property values for your organization and there are more then what has been defined in the commands above:

Distinguished Names
https://www.ibm.com/docs/en/ibm-mq/9.3?topic=certificates-distinguished-names

- Key Management Utility:

a. Open the IBM Key Management Utility for the TEST1 queue manager and press the OK button.

OBJ

b. Under the Key database content drop down menu select "Personal Certificates".

OBJ

c. In the "Personal Certificates" click on the button: New Self-Signed …
It is located at the bottom on the right side.

OBJ

d. In the "Create New Self-Signed Certificate" window enter the values for the following attributes and press the OK button.

Key Label : The name of the certificate, to be displayed when you list the personal certificates.
Default is ibmwebspheremqqueuemanagername
Version: Use default
Key Size: 2048 or larger
Signature Algorithm: SHA256withRSA
Common Name: Whatever you would like, ip address, hostname, queue manager name, etc.
Organization: Company Name
Validity Period: Usually a year but it can be extended by modifying the property value in days.

NOTE: You can fill out a subset or the rest of the certificate optional properties

OBJ

e. When completed the IBM Key Management Utility will reference the label name configured in the "Personal Certificates" key ring.

OBJ

f. Re-implement the same steps, only this time, use the instructions for the queue manager TEST2 key repository.

NOTE:
The paths will be different due to the names of the queue managers.

4. Extract the root signer certificate from the personal self-signed certificates.

- Command line.

Note: The result will be a file with suffix "arm" in the current directory from where you are executing the command.

a. runmqakm -cert -extract -db C:\MQData\qmgrs\TEST1\ssl\key.kdb -stashed -label ibmwebspheremqtest1 -target ibmwebspheremqtest1_root.arm

The command above extracts the public key from my personal certificate label/named ibmwebspheremqtest1 to a file I called ibmwebspheremqtest1_root.arm.

b. runmqakm -cert -extract -db C:\MQData\qmgrs\TEST2\ssl\key.kdb -stashed -label ibmwebspheremqtest2 -target ibmwebspheremqtest2_root.arm

The command above takes the file extracted in step a, ibmwebspheremqtest1_root.arm, and adds that certificate into the TEST2 key repository. The label is the name the certificate will have when it is listed in TEST2 key repository.

- Key Management Utility:

a. While the IBM Key Management Utility is still open and the "Personal Certificates" key ring is showing, press the Extract Certificate... button.

OBJ

b. When the "New" window opens, make sure of the following parameters/values and press the OK button.

Data type: Base63-encoded ASCII data
Certificate file name: ibmwebspheremqtest1_root.arm
(or what ever you would like to call it.)
Location: C:\MQData\qmgrs\TEST1\ssl (modify this according to your selections and paths)

OBJ

c. The root certificate file will be created with the name and the path chosen above.

5. Exchange the signer certificates for each queue manager AND add each certificate to the queue manager's key repository.

In Step 4. each queue manager root certificates was extracted to a file.
Now the task is to add the root certificate from the TEST1 queue manager into the key repository of the TEST 2 queue manager and add the root certificate from the TEST2 queue manager into the key repository of the TEST1 queue manager.

- Command Line:

runmqakm -cert -add -db C:\MQData\qmgrs\TEST2\ssl\key.kdb -label ibmwebspheremqtest1_root -file ibmwebspheremqtest1_root.arm -stashed

runmqakm -cert -add -db C:\MQData\qmgrs\TEST1\ssl\key.kdb -label ibmwebspheremqtest2_root -file ibmwebspheremqtest2_root.arm -stashed

- Key Management Utility:

a. Open the IBM Key Management Utility, select the key repository for the TEST1 queue manager and press the OK button.

OBJ

b. From the key database content drop down menu select Signer Certificates.

OBJ

c. On the right hand side of the Signer Certificates key ring, press on the "Add..." button. and browse for the root certificate file from the TEST2 queue manager.

OBJ

d. Browse for the root certificate file from the TEST2 queue manager, select it and press the Open button.

OBJ

e. Press the OK button.

OBJ

f. Now type a certificate label(name) of your choosing and press the OK button.

OBJ

g. The root certificate for the TEST2 queue manager has been successfully added to the key repository of TEST1.

OBJ

h. Using the same instructions add the root certificate of the TEST1 queue manager in step 5 to the key repository of the TEST2 queue manager. Be mindful of the key repository you have opened and browse for the root certificate of the TEST1 queue manager.

6. Select the channel encryption to be used and alter the SSLCIPH attribute of each channel.

Today's standard is to use a TLS 1.2 or TLS 1.3 Cipher Specifications to encrypt the transmission of your messages.
The IBM MQ Documentation provides what Cipher Specifications are supported.

Enabling CipherSpecs - MQ 9.3
https://www.ibm.com/docs/en/ibm-mq/9.3?topic=messages-enabling-cipherspecs

Enabling CipherSpecs - MQ 9.2
https://www.ibm.com/docs/en/ibm-mq/9.2?topic=messages-enabling-cipherspecs

Enabling CipherSpecs - MQ 9.1
https://www.ibm.com/docs/en/ibm-mq/9.1?topic=messages-enabling-cipherspecs

NOTE: Please make sure that you review the available Cpher Specifications for the version of MQ that you are using, some CipherSpecs are not supported by all MQ product versions.

- Command line:

For queue manager TEST1:

```
echo alter channel (TEST1_TO_TEST2) chltype(sdr) sslciph(TLS_RSA_WITH_AES_128_CBC_SHA256) |
runmqsc TEST1

echo alter channel (TEST2_TO_TEST1) chltype(rcvr) sslciph(TLS_RSA_WITH_AES_128_CBC_SHA256) |
runmqsc TEST1
```

For queue manager TEST2:

```
echo alter channel (TEST2_TO_TEST1) chltype(sdr) sslciph(TLS_RSA_WITH_AES_128_CBC_SHA256) |
runmqsc TEST2

echo alter channel (TEST1_TO_TEST2) chltype(rcvr) sslciph(TLS_RSA_WITH_AES_128_CBC_SHA256) |
runmqsc TEST2
```

- MQ Explorer GUI:

a. Open the MQ Explorer App.

OBJ

b. Expand the TEST1 queue manager and select Channels.

OBJ

c. Right click on TEST1_TO_TEST2 sender channel and select properties.

OBJ

d. On the left hand panel, select SSL.

OBJ

e. In the SSL window use the drop down menu for the SSL Cipher Spec attribute, select TLS_RSA_WITH_AES_128_CBC_SHA256 and press the OK button.

OBJ

f. On the TEST1 queue manager expand Channels and right click on receiver channel TEST2_TO_TEST1 and select Properties.

OBJ

g. On the Properties window, on the left side panel, click on SSL.

OBJ

h. On the SSL window use the drop down menu for the SSL Cipher Spec attribute, select TLS_RSA_WITH_AES_128_CBC_SHA256 and press the OK button.

OBJ

i. Repeat the steps for queue manager TEST2, modifying the SSL Cipher Spec attribute for sender channel TEST2_TO_TEST1 and receiver channel TEST1_TO_TEST2.

7. Start or restart (if your channel was running before the SSL CipherSpec was added) each sender channel and confirm the channel starts successfully.

- Command line:

```
echo start channel(TEST1_TO_TEST2) | runmqsc TEST1
```

```
echo display channel(TEST1_TO_TEST2) status | runmqsc TEST1

echo start channel(TEST2_TO_TEST1) | runmqsc TEST2

echo display channel(TEST2_TO_TEST1) status | runmqsc TEST2
```

The channel status will be running if the encryption handshake succeeded.

- MQ Explorer GUI:

a. Open the MQ Explorer App.

OBJ

b. Expand the TEST1 queue manager and select Channels.

OBJ

c. Right click on TEST1_TO_TEST2 sender channel and select Start.

OBJ

d. The channel has started successfully when the downward red arrow becomes a green arrow pointing upwards.

OBJ

8. Use the MQ sample utility to put a message to remote queue definition. (OPTIONAL)

As stated above, this step is optional because if the channel starts and runs successfully, the TLS encryption handshake was performed successfully and all messages will be transmitted with exception. However, you may want to verify the successful transmission of messages between the queue managers.

If the sample utilities are installed, the MQ product comes with a "putting" utility called amqsput.
The amqsput utility is found at the following locations:

Windows: Volume:\MQInstallDirectory\tools\c\Samples\Bin64\

Linux/Unix: /opt/mqm/samp/bin/

AIX: /usr/mqm/samp/bin/

- Command line:

a. From TEST1, issue the following command from the samples directory and press enter.

```
amqsput TEST2.RQ TEST1
```

where

TEST2.RQ is the remote queue definition to send messages to TEST2.
TEST1 is the name of the sending queue manager.

b. The amqsput utility will wait for you to type a message.

```
amqsput TEST2.RQ TEST1
Sample AMQSPUT0 start
target queue is TEST2.RQ
```

c. Type a message and press the enter key, once completed the utility will wait for you to type another message. If you only want to send one message, then press the enter key twice.

```
amqsput TEST2.RQ TEST1
Sample AMQSPUT0 start
target queue is TEST2.RQ
test message
```

d. When completed, the amqsput utility will end.

```
amqsput TEST2.RQ TEST1
Sample AMQSPUT0 start
target queue is TEST2.RQ
test message
Sample AMQSPUT0 end
```

e. To verify that the message has been placed in the TEST2.LQ, use the browse sample utility called amqsbcg.

```
amqsbcg TEST2.LQ TEST2
```

where

TEST2.LQ is the local queue hosted on the TEST2 queue manager.
TEST2 is the name of the receiving queue manager.

The browse output will show the MQ Message Descriptor contents as well as the message sent. Below is an example of the message content provided by the browsing utility.

```
****   Message      ****

 length - 12 of 12 bytes
```

00000000: 7465 7374 206D 6573 7361 6765 'test message '

- MQ Explorer GUI:

a. Open the MQ Explorer App.

OBJ

b. Expand the TEST1 queue manager and select Queues.

OBJ

c. Right click on TEST2.RQ and select "Put Test Message..."

OBJ

d. Type a test message in the "Message data:" field, press the "Put message" button, and press the "Close" button.

OBJ

e. Navigate to the TEST2 queue manager, expand the TEST2 queue manager and select Queues.

OBJ

f. Right click on local queue TEST2.LQ and select "Browse Messages...".

OBJ

g. When the "Message browser" window opens, select the message and use the scroll bar to search for the column heading called "Message data". Under the column heading you will see your test message. If the message is longer than 14 or so characters, you will not be able to see the message in its entirety. You will need to either expand the column width or right click on the message and choose "Properties...".

h. In the "Properties" window, on the left side panel, click on Data. On the right hand side, view the contents of the "Message data:" field. Press the "Close" button when you have confirmed your message content.

OBJ

+++ end +++