HOW TO CONFIGURE SSL ON MQ APPLIANCE QUEUE MANAGERS

https://www.ibm.com/support/pages/node/7009177

Date last updated: June-30-2023

Mariana Perez
Christian Montes Tirado
IBM MQ Support
https://www.ibm.com/products/mq/support
Find all the support you need for IBM MQ

+++ Objective +++

The Objective of this technote is to provide steps to configure TLS on a MQ HA Appliance queue manager connecting to a MQ Linux queue manager.

++ Overall Steps ++

1. Create a certificate request (CSR)
2. Send it to the CA to be signed.
3. Add the CA signer certificates returned by the CA into the key repository.
4. Receive the CSR signed into the key reposiroty
5. Validate the certificate chain.
6. Exchnage the CA signers between Qmgrs
7. Verify the cert label is correct on the Qmgrs
8. Refresh security
9. Create the SDR/RCVR channels.
10. Test without SSL
11. Enable encryption on the channels and test.

Note: The process described on this tutorial will be using CA signed personal certificates.

++ MQ Configuration ++

APMQ – Appliance Qmgr
PIKES - Linux Qmgr
APMQ.TO.PIKES – SDR/RCVR channel pair

Command use to create the MQ HA Appliance Qmgr:

```
crtmqm -fs 1 -sx APMQ
```

-fs file system size. Default is 64 GB
-sx specifies this is HA Qmgr

To view the status of the HA Qmgr:

```
status APMQ
```

Command to list the certificate files contained in the mqpubcert directory

```
mqa(config)# dir mqpubcert:
```

++ Procedure on MQ Appliance Qmgr ++

NOTES:

- When a queue manager is created on the MQ appliance, a key repository is automatically created for that queue manager. The key repository is deleted when the queue manager is deleted.
- Certificate files and certificate request files for the MQ Appliance are stored in mqpubcert:
- Documentation reference on certificate commands. https://www.ibm.com/docs/en/mq-appliance/9.3?topic=security-tls-certificate-management

1) Create a CSR

Use the createcertrequest command:

```
mqa# mqcli
mqa(mqcli)#createcertrequest -m APMQ -dn "CN=APMQ,O=IBM,C=US,OU=MQ
Support,ST=NorthCarolina" -label apmq-cert
```

- Can use the “listcertrequest” to list the certificate request that exist in the key repository:

```
mqa(mqcli)#listcertrequest -m APMQ
```

- Can see the CSR details with “detailcertrequest” command:

```
mqa(mqcli)#detailcertrequest -m APMQ -label apmq-cert
```

2) Send the CSR to the Certificate Authority (CA) to be signed.

Use the copy command or MQ Console WebUI to download the CSR from the Appliance.
For this test, the copy command was used to send it to the Linux VM to be signed with some internal CAs.

mqa(config)# copy mqpubcert://csr_filename scp://username@ipaddress/[/]directorypath

```
mqa(config)# copy mqpubcert:// APMQ -2023  scp://root@9.46.111.71//var/mqm/qmgrs/PIKES/ssl
        Password: ***************
        File copy success
```

The CA will sign the certificate request and send back the certificate signed, along with the CA root and intermediate(s) used to sign this certificate.

3) Upload the signed certificate and CA certificates back to MQ Appliance

Using the copy command:
mqa(config)# copy scp://username@ipaddress/[/]directorypath/cert_filename mqpubcert://

```
mqa(config)# copy scp://root@9.46.111.71//var/mqm/qmgrs/PIKES/ssl/ signed-APMQ -2023.csr
mqpubcert://
mqa(config)# copy scp://root@9.46.111.71//home/mqm/ca-certs/rootca.arm mqpubcert://
```

4) Add the CA certificates with the "addcert" command:

```
mqa(mqcli) # addcert -m APMQ -label root_cert -file rootca.arm
mqa(mqcli) # addcert -m APMQ -label ca-int-cert -file intermediate-1-ca.arm
```

5) Received the signed certificate using the “receivecert” command:

```
mqa(mqcli) # receivecert -m APMQ -file filename
```

NEW: On MQ 9.3, a new command was added to validate the certificate chain:
mqa(mqcli)# validatecert -m QmgrName -label cert-label

```
mqa(mqcli)# validatecert -m APMQ -label apmq-cert
5724-H72 (C) Copyright IBM Corp. 1994, 2022.
OK
```

++ Procedure on MQ Linux Qmgr ++

NOTE: This tutorial uses the runmqakm tool shipped with MQ.
Another option is to use runmqckm or the iKeyman GUI, both also shipped with MQ.

https://www.ibm.com/docs/en/ibm-mq/9.3?topic=securing-managing-keys-certificates-aix-linux-windows
Title: Managing keys and certificates on AIX, Linux, and Windows

1) Create a key repository

```
runmqakm -keydb -create -db key.kdb -type cms -pw passw0rd -stash
```

2) Create a CSR -

```
runmqakm -certreq -create -db key.kdb. -stashed -label ibmwebspheremqpikes -dn
"CN=LINMQ2,O=IBM,C=US,OU=MQ Support,ST=NorthCarolina" -size 2048 -sig_alg SHA256WithRSA -
file certreq-PIKES.csr
```

- Can list the certificate request with:

```
runmqakm -certreq -list -db key.kdb -stashed
```

3) Send the CSR to the CA to be signed.

This tutorial uses an internal CA to sign the certificate:

```
runmqakm -cert -sign -db /home/mqm/ca-certs/cert-auth.kdb -stashed -label Intermediate-1 -file
certreq-PIKES.csr -target signed-certreq-PIKES.crt
```

The CA will send the certificate signed and the root, intermediate(s) certificates.

4) Add the CA signer certificates.

```
runmqakm -cert -add -db key.kdb -stashed -label root-ca-cert -file /home/mqm/ca-certs/rootca.arm
runmqakm -cert -add -db key.kdb -stashed -label int1-ca-cert -file /home/mqm/ca-certs/intermediate-
1-ca.arm
```

5) Receive the CSR signed

```
runmqakm -cert -receive -file signed-certreq-PIKES.crt -db key.kdb -stashed
```

- validate the certificate chain

```
runmqakm -cert -validate -db key.kdb -stashed
```

++ Exchange CA certificates ++

For one way TLS, only the TLS client needs to validate the TLS certificate.

For mutual TLS (mTLS), meaning that the TLS server needs to authenticate the TLS client, both parties need to have the CA root and intermediate(s) certificates used to sign its corresponding personal certificate.

In MQ, mutual TLS is determined by the channel attribute SSLCAUTH set to REQUIRED.

https://www.ibm.com/docs/en/ibm-mq/9.3?topic=keywords-sslcauth-ssl-client-authentication
Title: SSLCAUTH (SSL Client Authentication)

This tutorial implements mutual TLS, thus we need to exchange the certificates from both parties.

From the MQ Appliance, we can use the copy command to:

a) upload the TLS client CA certificates to the MQ Appliance
b) to send the MQ Appliance CA certs to the TLS peer.

1) Send the CA signer certs from Linux Qmgr to MQ appliance

mqa(mqconfig)# copy scp://username@ipaddress:port//path/ mqpubcert:///certFileName

```
mqa(config)# copy scp://root@9.46.111.71//home/mqm/ca-certs/rootca.arm mqpubcert://
```

2) Send the CA signer certs from MQ appliance to Linux Qmgr.

mqa(mqconfig)# copy mqpubcert://csr_filename scp://username@ipaddress/[/]directorypath

```
mqa(config)# copy mqpubcert://rootca.crt scp://root@9.46.111.71//var/mqm/qmgrs/PIKES/ssl
```

Documentation Reference:
https://www.ibm.com/docs/en/mq-appliance/9.3?topic=appliance-uploading-certificates
Uploading certificates to the appliance

++ Refresh Security ++

At this point we have all the certificate in the corresponding key repositories.

Next, we need to verify that the queue manager attributes CERTLABL and SSLKEYR (on non-Appliance systems) have the correct values.

Then need to issue “refresh security type(ssl)” to pick up the new changes made to the key repositories.

```
#runmqsc QmgrName
REFRESH SECURITY TYPE(SSL)
END
```

++ Enable TLS on the channels ++

It is good practice to test the connection first without TLS.

To enable TLS, the channels need to be altered to have a cipher:

```
ALTER CHANNEL(APMQ.TO.PIKES) SSLCIPH(TLS_RSA_WITH_AES_128_GCM_SHA256)
```

If want to restrict the connection more, for example by only accepting specific certificates provided by the peer TLS, then can make use of SSLPEER.

https://www.ibm.com/docs/en/ibm-mq/9.3?topic=keywords-sslpeer-ssl-peer
Title: SSLPEER (SSL Peer)

This tutorial does not implement that attribute.

++ Test ++

Hopefully is a successful connection.
If it fails, review the Qmgr error logs on both sides to have some guidance about what went wrong.

On MQ Appliance, we can test a managed failover to verify the TLS related data and configuration was replicated and available to use in the other MQ Appliance.

Display the status to verify where the Qmgr is running

```
mqa(mqcli)# status APMQ
```

On the Appliance where we want to move/failover the Qmgr, run the sethapreferred command:

```
mqa(mqcli)# sethapreferred APMQ
```

Verify the key repository information:

```
mqa(mqcli)#listcert -m APMQ
```

++ Key repository Backup and Restore++

On MQ Appliance we can use the “keybackup” and “keyrestore” to take a backup of the Qmgr key repository and to restore it back.

The keybackup command will generate a password and will place the backup file in the mqbackup command. It is very important to save this password as it would be needed to restore.

mqa(mqcli)# keybackup -m Qmgrname

```
mqa(mqcli)# keybackup -m APMQ
5724-H72 (C) Copyright IBM Corp. 1994, 2022.
This operation will generate a copy of your queue manager key repository, which
may include private keys. Although encrypted, you should take appropriate
security precautions in handling this file.  The password required if you ever
need to modify or restore this file will be displayed after the copy has been
created.
Do you wish to continue? [Y/N]
Y
Key repository has been backed up to 'mqbackup://APMQ_keyrepos.tar.gz'.
Password for key repository is:
K@:}INjc"rrqr\
```

To restore the key repository we need to provide the above generated password.

You must enclose the password in double quotes if it includes special characters. You must also escape any backslash or double quote characters that are part of the password with a backslash character.

mqa(mqcli)# keyrestore -m QmgrName -file filename -password password

```
mqa(mqcli)# keyrestore -m APMQ -file APMQ_keyrepos.tar.gz -password "K@:}INjc\"rrqr\\"
5724-H72 (C) Copyright IBM Corp. 1994, 2022.
mqa(mqcli)#
```

++ Appliance Commands Reference ++

https://www.ibm.com/docs/en/mq-appliance/9.3?topic=security-tls-certificate-management
TLS certificate management

- List the certificates in the Qmgr key repository
  mqa(mqcli)#listcert -m QmgrName

- List the certificate request
  mqa(mqcli)#listcertrequest -m QmgrName

- Show details of a certificate
- mqa(mqcli)# detailcert -m QMgrName -label Label

- Show details of a certificate request
  mqa(mqcli)# detailcertrequest -m QMgrName -label Label

- Create a certificate request
  mqa(mqcli)# createcertrequest -m QMgrName -dn DistinguishedName -label LabelName

- Add a public certificate, like CA cert
  mqcli# addcert -m QmgrName -label Label – file filename

- Receive a certificate request
  mqa(mqcli)# receivecert -m QmgrName -file filename

- Validate a certificate
  mqa(mqcli)# validatecert -m QmgrName -label label-name

- Rename a certificate
  mqa(mqcli)# renamecert -m QMgrName -label CurrentLabel -new_label NewLabel

- Take a backup of the key repository
  mqa(mqcli)# keybackup -m Qmgrname

- Restore the key reposiory from backup
  mqa(mqcli)# keyrestore -m QmgrName -file filename -password password

+++ end +++