Coexistence of Default and Custom TLS Certificates

until the custom certificates are added and the default certificates deleted

IBM/HCL Workload Automation

Sajjad M. Kabir
Solutions Architect
IBM/HCL Lab Services

Version: 1.7
Date: Jun 26, 2024

1 Document History

1.1 Document Location

This is a snapshot of an on-line document. Paper copies are valid only on the day they are printed. Refer to the author if you are in any doubt about the currency of this document.

1.2 Revision History

The following is a revision history of the document.

Revision Number	Revision Date	Summary of Changes	Changes Marked
1.0	May 17, 2024	Initial version	
1.1	May 28, 2024	Updated some of the steps	No
1.2	May 30, 2024	Added a new step 6 in section 4.5 Populate the Certificate Depot from the new Certificate	No
1.3	Jun 5, 2024	Reordered and added steps 7 & 8 in section 4.4 Populate the Certificate Depot from the new Certificate	No
1.4	Jun 11, 2024	Added sections 4.8 and 4.9	No
1.5	Jun 12, 2024	Updated section 4.6.6	No
1.6	Jun 24, 2024	Updated various sections	No
1.7	Jun 26, 2024	Added Links to other resources with the same topic in section 5 Other Resources Regarding Replacing Certificates	No

1.3 Approvals

This document requires following approvals.

Name	Title

Contents

2 Introduction

Network communication between each component of IBM Workload Automation (IWA) is secured using TLS v1.2 and v1.3 protocols. This is accomplished by deploying TLS Certificates for each component. IWA deploys IBM Self-Signed certificates at installation time to make it easy to deploy and have a working scheduling environment with little effort. To enhance security, organizations decide to replace the default certificates with commercially recognized CA (Certificate Authority) signed certificate for each server in the environment. This requires all default self-signed certificates used by every component to be replaced by CA signed certificates.

In order to keep the environments intact and all components communicating while the default certificates are replaced, instead of a rip and replace approach, a cap and grow approach is adopted where the default certificates coexist with the new certificates until the new certificates have been imported into the KeyStores and TrustStores of all components. The default certificates can then be deleted from all KeyStores and TrustStores.

The following sections describe the procedure to replace the certificates.

3 Overview of TLS in IWA

IWA utilizes IBM WebSphere Application Server Liberty Profile (WLP) as the frontend web user interface as well as the backend engine. As such, there are two WLP instances that provide the services for DWC and MDM. The TLS implementation in WLP for DWC and MDM requires two stores to be present. The following is a description of the terminologies and concept used.

KeyStore (KeyFile) Is a file to store private keys in the form of Personal Certificates. Some KeyStores also contain trusted or public keys.

TrustStore (TrustFile) Is a file to store public keys in the form of Signer Certificates. Some TrustStores also contain trusted or private keys.

Personal Certificate Represents the identity of a server and contains a private key for signing/encrypting data.

Signer Certificate Represents the identity of a server and contains a public key for decrypting data.

Certificate Chain Is a concatenation of CA signed server certificate and certificates of Intermediate and Root CAs.

Trusted CAs Is a concatenation of trusted CA certificates.

TLS Certificates are imported into the KeyStores for each profile and then the public keys are extracted and imported into the TrustStores.

The stores can have different formats for compatibility and security reasons and are used by different components as shown below.

JKS Java Key Store
Used by the MDM and DWC WLP as well as a component of the Dynamic Agent
File extension is jks

CMS Cryptographic Message Syntax
Used by the Dynamic Agents
File extension is kdb

There are different types of file extensions used to signify the type of certificate or key it contains as described below.

CSR A Personal Certificate Signing Request contains the request to be sent to the CA for signing. It is generated from a KeyStore.

CRT A Personal Certificate that is signed by the CA. It is imported into the KeyStore

CER A Personal Certificate that is signed by the CA or a public key for root CA, Intermediate CA, server exported from a KeyStore and imported into a TrustStore

KEY A Private Key

Alias A label for a signed certificate after being imported into a KeyStore or TrustStore

ARM A Signer Certificate chain. It is exported from a KeyStore and imported in to a TrustStore.

DER Distinguished Encoding Rules to encode any data object into a binary file. A certificate is encoded using these rules and presented in a binary file.

PKCS12 A Public-Key Cryptography Standards (PKCS#12) keystore that contains the private key(s), public certificate(s), and their CA certificate chain(s). The file extension can be either p12 or pfx (Personal Information Exchange).

PEM Privacy-Enhanced Electronic Mail is an encryption format used by certificates. It is a base 64 encoding of the Certificate and is enclosed between

"-----BEGIN CERTIFICATE-----"

and

"-----END CERTIFICATE-----"

4 Replacing the Default Certificates with Custom Certificates

4.1 KeyStore and TrustStore Used by MDM and DWC

The MDM and DWC each have one KeyStore and one TrustStore in JKS format. They are located in the following directories:

MDM <install_dir>/usr/servers/engineServer/resources/security
KeyStore TWSServerKeyFile.jks
TrustStore TWSServerTrustFile.jks

DWC <<install_dir>/usr/servers/dwcServer/resources/security
KeyStore TWSServerKeyFile.jks
TrustStore TWSServerTrustFile.jks

TWSServerKeyFile.jks

This KeyStore contains the private and public certificates for the server, the Root, and Intermediate public certificates of the CA that signed the server certificate. The public certificate of the server is exported and imported into the TWSServerTrustStore.jks and TWSClientKeyStore.

TWSServerTrustFile.jks

This TrustStore contains the public certificates for the server, the Root and Intermediate public certificates of the CA that signed the server certificate.

4.2 Importing new Certificates for MDM and DWC

The following is a procedure to import the new signed certificates. The steps are the same for MDM and DWC.

1. Create a request for a certificate for the DWCs and MDMs and include the DNS names for all other MDMs, BKMs, DDMs, and BDDMs in different environments.
2. Backup the KeyStore and TrustStore

   MDM: cd <install_dir>/usr/servers/engineServer/resources/security
   DWC: cd <install_dir>/usr/servers/dwcServer/resources/security
   cp TWSServerKeyFile.jks TWSServerKeyFile.jks.orig
   cp TWSServerTrustFile.jks TWSServerTrustFile.jks.orig
3. The private key, signed server certificate, intermediate and root CA certificates (forming the CA Chain) can be provided in many formats by the CA. If they are provided in PEM formats, they can be directly imported into the DWC/MDM KeyStore and TrustStore. If they are provided in a PFX (Personal Information Exchange) format, run the following command to import the contents of this file into the **DWC and MDM** KeyStores as shown below:

   cp <dir>/<name_of_file>.pfx <install_dir>/usr/servers/**<component>Server**/resources/security

   keytool -importkeystore -srckeystore <name_of_file>.pfx -srcstoretype PKCS12 -srcstorepass <password> -destkeypass <password> -destkeystore **TWSServerKeyFile.jks** -deststorepass <password>
4. View the contents of the KeyStore

keytool -list -keystore **TWSServerKeyFile.jks** -storepass <password>

Keystore type: jks
Keystore provider: SUN

Your keystore contains 4 entries

server (default certificate)
glowfish (new certificate)
intermediate (CA who signed the new certificate)
rootca (CA who signed the intermediate certificate)

5. Export the public certificate of the new certificate into a file from the KeyStore

   keytool -exportcert -alias **glowfish** -file **glowfish.pub.crt** -keystore **TWSServerKeyFile.jks** -storetype jks -storepass <password>

6. Export the intermediate certificate of the new certificate into a file from the KeyStore

   keytool -exportcert -alias **intermediate** -file **intermediate.pub.crt** -keystore **TWSServerKeyFile.jks** -storetype jks -storepass <password>

7. Export the root CA certificate of the new certificate into a file from the KeyStore

   keytool -exportcert -alias **rootca** -file **rootca.pub.crt** -keystore **TWSServerKeyFile.jks** -storetype jks

8. Export the public certificate of the default certificate into a file from the KeyStore

   keytool -exportcert -alias **server** -file **server.pub.crt** -keystore **TWSServerKeyFile.jks** -storetype jks -storepass <password>

9. Import all the public, intermediate, and root certificates exported in the above steps into the TrustStore

   keytool -importcert -file **<name>**.pub.crt -keystore TWSServerTrustFile.jks -alias <**name**> -storepass <password> -trustcacerts -noprompt

10. Export the certs included in the TrustStore to a file

    keytool -list -keystore TWSServerTrustFile.jks -storepass <password> -v > cert.list

11. Find the following 4 certificates in the file, cert.list

    Alias name: **glowfish** CN=**glowfish**
    Alias name: **intermediate** CN=**Intermediate**
    Alias name: **rootca** CN=**RootCA**
    Alias name: twstrustkey CN=ServerNew
    Alias name: client CN=ClientNew
    Alias name: twstrustkeyold CN=Server (expired in 2014)

12. A trusting relationship already exists between the DWC and MDM since their certificates are signed by the same CA. Therefore, it is not necessary to import the CA Certificate chain of one to the other component. If that is not the case, import the CA certificate chain from one to the other.

13. WebSphere Liberty Profile uses the KeyStore, TWSServerKeyFile.jks, and now that it contains two certificates, which certificate it should use must be indicated by adding the following attribute in **ssl_config.xml** file in the **defaults** directory as shown below on the MDM:

    cd <TWSDATA>/usr/servers/engineServer/configDropins/**defaults**
    cp **ssl_config.xml** ssl_config.xml.orig

vi **ssl_config.xml**
Add the following line in bold:

<ssl id="twaSSLSettings" keyStoreRef="twaKeyStore" trustStoreRef="twaTrustStore" sslProtocol="TLSv1.2" clientAuthenticationSupported="true" **serverKeyAlias="server"**/>

14. The **DWC** doesn't need to keep both certificates in the KeyStore, hence, delete the old certificate

    cd <install_dir>/usr/servers/**dwcServer**/resources/security
    keytool -delete -alias **server** -keystore **TWSServerKeyFile.jks**

15. Restart the **DWC** WLP

    cd <Install_Dir>/appservertools
    ./stopAppServer.sh
    ./startAppServer.sh

16. Go to the DWC URL and verify that it is using the new certificate by clicking on the pad lock on the URL address bar and viewing the certificate.

4.3 Update DWB Workstation Properties

To work with the Dynamic Agents (DA) through the Dynamic Workload Broker (DWB), the CN of the CA signed certificates needs to be added to the Broker.AuthorizedCNs property in BrokerWorkstation.properties file.

Follow the steps below to update this file.

1. Go to the directory where the BrokerWorkstation.properties file is located

   cd <Install_Dir>/TWSDATA/broker/config

2. Backup and edit the BrokerWorkstation.properties file and append the values shown in bold below

   cp BrokerWorkstation.properties BrokerWorkstation.properties.orig
   vi BrokerWorkstation.properties
   Broker.AuthorizedCNs=Server;ServerNew;**glowfish**

   Note:

   The CN=Server has the Alias twstrustkeyold in TWSServerTrustFile.jks, which expired in 2014
   The CN=ServerNew has the Alias twstrustkey in TWSServerTrustFile.jks
   The CN=glowfish has the Alias glowfish in TWSServerTrustFile.jks

3. Restart the **MDM** WLP

   cd <Install_Dir>/appservertools
   ./stopAppServer.sh
   ./startAppServer.sh

4.4 Populate the Certificate Depot from the new Certificate

The utility, AgentCertificateDownloader, downloads the new certificates from the **MDM** to the **DA**, but the certificates need to be in PEM format. Follow the steps below to create the certificates in PEM format.

1. Go to the directory where the KeyStore is located on the **MDM**

cd <Install_Dir>/usr/servers/**engineServer**/resources/security

2. The private key is embedded in PFX KeyStore and is not displayed for security purposes. The private key, public certificate, and the CA chain can be exported from the pfx keystore directly as it is already in PKCS12 format. But if the new certificates have been imported into the JKS KeyStore already then follow the steps below to export the certificates.
3. Convert the JKS KeyStore to PKCS12 (p12) format

   keytool -importkeystore -srckeystore **TWSServerKeyFile.jks** -destkeystore **TWSServerKeyFile.p12** -deststoretype PKCS12 -srcalias **glowfish** -deststorepass <password> -destkeypass <password>
4. Export the private key of the new certificate from the PKCS12 KeyStore

   openssl pkcs12 -in **TWSServerKeyFile.p12** -nodes -nocerts -out **tls.key**
5. Export the public certificate of the new (**client**) certificate from the PKCS12 KeyStore

   openssl pkcs12 -in **TWSServerKeyFile.p12** -nokeys -**clcerts** -out **tls.crt**
6. Export the **CA** certificate chain from the PKCS12 KeyStore

   openssl pkcs12 -in **TWSServerKeyFile.p12** -nokeys -**cacerts** -**chain** -out **tls.crt**
7. Append the ca.crt to the trusted certificates used by the FTA on the MDM. This allows the new certificates to be deployed to the FTA trusted by the MDM.

   cat **ca.crt** >> <Install_Dir>/TWS/ssl/OpenSSL/**TWSTrustCertificates.cer**
8. Open the resulting file, TWSTrustCertificates.cer, and make sure that each certificate begins with, **BEGIN CERTIFICATE**, and ends with, **END CERTIFICATE**, on a line by itself. Any text in between two certificates is ignored.
9. Copy the updated file, **TWSTrustCertificates.cer**, to **ca.crt** overwriting it so that they both have the same certificates. This allows the new certificates to be deployed to the FTAs trusted by the MDM.

   cp <Install_Dir>/TWS/ssl/OpenSSL/**TWSTrustCertificates.cer ca.crt**
10. Create the stash file with the password

    echo -n <password> | base64 > **tls.sth**
11. Copy all the certificates to the Certificate Depot

    cp **tls.key tls.crt tls.sth ca.crt** <Install_Dir>/TWSDATA/**ssl/depot**

4.5 Importing new Certificates for BKMs, DDMs, BDDMs

These components have the same structure as the MDM. As such, copy the **KeyStore**, **TrustStore** from the MDM to these components, update the **BrokerWorkstation.properties** and **ssl_config.xml** files, and restart WLP.

4.6 Importing new Certificates for DAs on MDM, BKMs, DDMs, BDDMs, DMs, & BDMs

The Dynamic Agent uses two KeyStores, one in CMS format (kdb extension) and the other in JKS format (jks extension). Both KeyStores contain the same certificates, which include the private certificate for the server where the agent is installed, the public certificate of the MDM

server and the CA Chain, and all third party trusted CA certificates. They are located in the following directory:

JKS KeyStore: <Install_Dir>/TWSDATA/ssl/TWSClientKeyStoreJKS.jks
CMS KeyStore: <Install_Dir>/TWSDATA/ssl/GSKit/TWSClientKeyStore.kdb

To make things easier and economical, the new certificates used on the MDM can be used for the DAs without compromising security. A new utility, AgentCertificateDownloader, can download the certificates from the Certificate Depot on the MDM and build the KeyStores. Since it only downloads the new certificates and leaves out the default certificates, it breaks the trusting relationship with the MDM and causes the DAs to not link. As such, a few additional steps need to be executed.

Follow the steps below to download the new certificates and import the default certificates.

1. Backup the following KeyStores and files
   a. cd <Install_Dir>/TWSDATA/ssl/
      cp TWSClientKeyStoreJKS.jks TWSClientKeyStoreJKS.jks.orig
   b. cd <Install_Dir>/TWSDATA/ssl/GSKit
      cp TWSClientKeyStore.kdb TWSClientKeyStore.kdb.orig
   c. cd <Install_Dir>/TWSDATA
      cp localopts localopts.orig
2. The following certs are included in the CMS KeyStores before running AgentCertificateDownloader

   export PATH=/usr/Tivoli/TWS/GSKit64/8/bin/:$PATH
   cd <Install_Dir>/TWSDATA/ssl/GSKit
   gsk8capicmd_64 -cert -list -db TWSClientKeyStore.kdb -v -stashed

   Certificates found
   * default, - personal, ! trusted, # secret key
   ! **server** CN=ServerNew,OU=TWS,O=IBM,C=US CN=ServerNew,OU=TWS,O=IBM,C=US
   ! **serverold** CN=Server,OU=TWS,O=IBM,C=US CN=Server,OU=TWS,O=IBM,C=US
   - **client** CN=ClientNew,OU=TWS,O=IBM,C=US CN=ClientNew,OU=TWS,O=IBM,C=US
3. The following certs are included in the JKS KeyStores before running AgentCertificateDownloader

   cd <Install_Dir>/TWSDATA/ssl
   export PATH=<Install_Dir>/TWS/JavaExt/jre/jre/bin:$PATH
   keytool -list -keystore TWSClientKeyStoreJKS.jks -storepass <password> -v > cert.list

   Find the following 2 certificates in the file, cert.list

   vi **cert.list**
   Alias Name: **server** CN=ServerNew
   Alias Name: **client** CN=ClientNew
4. The utility, AgentCertificateDownloader, needs to be launched in a shell where the IWS environmental variables haven't been set. This is due to a conflict with the curl libraries sourced with the OS provided curl binary. Follow the steps below to setup such a shell env and run the utility
   a. vi .profile and comment the line that sources the env setting script, tws_env.sh
   b. Logout and Log back in
   c. cd <Install_Dir>/TWS

d. ./AgentCertificateDownloader.sh --apikey <**Personal API key from DWC**> --tdwbhostname <MDM> --tdwbport 31116 --work_dir /tmp/acd

5. Uncomment the line in .profile to source the env setting script, tws_env.sh
6. Source the environment
   . ./.profile
7. Import the public certificate of the default certificate from the MDM to the updated CMS and JKS KeyStores of the DA
   a. Go to the dir where the JKS KeyStore is located
      cd <Install_Dir>/TWSDATA/ssl
   b. Copy the public certificate of the default certificate, server.pub.crt, from the MDM to DA
   c. Import the public certificate of the default certificate with an alias, **serverold**, because the alias, server, is used by the new certificate

      keytool -importcert -file **server.pub.crt** -keystore **TWSClientKeyStoreJKS.jks** -alias **serverold** -trustcacerts -storepass <password> -noprompt
   d. Go to the dir where the CMS KeyStore is located
      cd <Install_Dir>/TWSDATA/ssl/GSKit
   e. Import the public certificate of the default certificate with an alias, server

      gsk8capicmd_64 -cert -add -db **TWSClientKeyStore.kdb** -file ../**server.pub.crt** -label **server** -trust enable -stashed
8. The following certs are now included in the CMS KeyStores after running AgentCertificateDownloader and importing the default certificate

   gsk8capicmd_64 -cert -list -db **TWSClientKeyStore.kdb** -v -stashed
   Certificates found
   * default, - personal, ! trusted, # secret key
   ! **ca** CN=**glowfish** …
   ! **server** CN=**ServerNew**,OU=TWS,O=IBM,C=US
   - **client** CN=**glowfish** …
9. The following certs are now included in the CMS KeyStores after running AgentCertificateDownloader and importing the default certificate

   cd <Install_Dir>/TWSDATA/ssl
   keytool -list -keystore **TWSClientKeyStoreJKS.jks** -storepass <password> -v > cert.list.new

   Find the following 2 certificates in the file, cert.list.new

   vi **cert.list.new** and search for the following:
   Alias name: **server** CN=**glowfish**
   Alias name: **ca** CN=**RootCA**
   Alias name: **serverold** CN=**ServerNew**
10. Go to the **OpenSSL** dir and view the contents of the **ca.crt** file. All the trusted certificates should be in **ca.crt** now, each separated with a **BEGIN CERTIFICATE** header and **END CERTIFICATE** footer. Any verbiage in between two certificates can be ignored.

    cd /twsutec/ibm/TWA/TWSDATA/ssl/**OpenSSL**
    cat **ca.crt**
    Server

Client
ServerNew
ClientNew
glowfish
Root CA Certificate

11. Restart the DA (ShutDownLwa and StartUpLwa)
12. Restart the FTA, if one exists (conman "stop;wait" and conman "shut;wait")
13. Verify that the DA and FTA are linked
14. Verify that composer and optman commands work (on BKMs, DDMs, and BDDMs)

4.7 Place the New Certificates on DMs and FTAs

Follow the steps below to replace the certificates on the DM and FTA.

1. Copy the following certificates in PEM format from the directory on the MDM as shown below to the FTA. If the DA on the FTA was already updated with running the utility, AgentCertificateDownloader, these certificates would have been already copied.

   <Install_Dir>/TWSDATA/ssl/depot
   tls.key
   tls.crt
   tls.sth
   ca.crt

2. Place the above certificates in the dir shown below on the FTA

   <FTA_Install_Dir>/TWSDATA/ssl/**OpenSSL**

3. Append the public certificate of the default certificate and other trusted CAs into the new ca.crt

   cd <Install_Dir>/TWSDATA/ssl/**OpenSSL**
   cat <Install_Dir>/TWS/ssl/OpenSSL/**TWSTrustCertificates.cer** >> **ca.crt**

4. Go to the following directory and edit localopts

   cd <Install_Dir>/TWSDATA
   vi **localopts**

5. Make the following changes in **bold**

   nm SSL full port =**31113**

   SSL key = "<Install_Dir>/TWS**DATA**/ssl/OpenSSL/**tls.key**"
   SSL certificate = "<Install_Dir>/TWS**DATA**/ssl/OpenSSL/**tls.crt**"
   SSL key pwd = "<Install_Dir>/TWS**DATA**/ssl/OpenSSL/**tls.sth**"
   SSL CA certificate = "<Install_Dir>/TWS**DATA**/ssl/OpenSSL/**ca.crt**"
   SSL random seed = "<Install_Dir>/TWS**DATA**/ssl/OpenSSL/**tls.rnd**"

6. Update the Workstation definition of the FTA with the following settings in **bold**. For example, the definition of the FTA, GLOWFISH, is shownn below.

   composer cr glowfish.txt from ws=@!glowfish

   CPUNAME glowfish
   OS UNIX
   NODE glowfish.raleigh.ibm.com TCPADDR 31111
   SECUREADDR 31113
   TIMEZONE America/New_York
   DOMAIN MASTERDM

```
  FOR MAESTRO
    TYPE FTA
    AUTOLINK ON
    BEHINDFIREWALL OFF
    SECURITYLEVEL FORCE_ENABLED
    FULLSTATUS OFF
  END
```

7. Import the modified FTA definition

   composer replace glowfish.txt

8. Update the plan so that the updated FTA definition is reflected in the plan

   optman chg cf = all
   JnextPlan -for 0000
   optman chg cf = yes

3. Stop the FTA
   conman "stop;wait"
   conman "shut;wait"

4. Start the FTA
   cd <Install_Dir>/TWS
   ./StartUp
   conman start

5. Verify that the FTA linked with the server successfully and it shows all the flags, **LTI JW**

4.8 Delete the Default Certificates

Once the certificates on all components have been updated, the default certificates can be deleted from various KeyStore and TrustStores. Follow the steps below to delete the default certificates.

1. Update the FTA on the MDM to start using the new certificates
   a. cd <Install_Dir>/TWSDATA/ssl/**depot**
   b. cp * ../**OpenSSL**
   c. cd <Install_Dir>/**TWSDATA**
   d. vi **localopts** and update the following parameters:

      SSL key = "<Install_Dir>/TWS**DATA**/ssl/OpenSSL/**tls.key**"
      SSL certificate = "<Install_Dir>/TWS**DATA**/ssl/OpenSSL/**tls.crt**"
      SSL key pwd = "<Install_Dir>/TWS**DATA**/ssl/OpenSSL/**tls.sth**"
      SSL CA certificate = "<Install_Dir>/TWS**DATA**/ssl/OpenSSL/**ca.crt**"
      SSL random seed = "<Install_Dir>/TWS**DATA**/ssl/OpenSSL/**tls.rnd**"

   e. Restart the FTA on the MDM

      cd <Install_Dir>/TWS
      conman "stop;wait"
      conman "shut;wait"
      ./StartUp
      conman start

2. Update the MDM to start using the new certificate by switching the attribute in **ssl_config.xml** from **server** to the alias of the new certificate, **glowfish**

cd <Install_Dir>/usr/servers/engineServer/configDropins/**defaults**
vi **ssl_config.xml**
Change the value of following attribute in bold:

<ssl id="twaSSLSettings" keyStoreRef="twaKeyStore" trustStoreRef="twaTrustStore" sslProtocol="TLSv1.2" clientAuthenticationSupported="true" serverKeyAlias="**glowfish**"/>

3. Delete the default certificates from the KeyStores and TrustStores of the MDM, BKM, DDM, BDDM
   a. cd <install_dir>/usr/servers/engineServer/resources/security
   b. keytool -delete -alias **client** -keystore TWSServerKeyFile.jks -storepass <password>
   c. keytool -delete -alias **clientold** -keystore TWSServerTrustFile.jks -storepass <password>
   d. keytool -delete -alias **twstrustkey** -keystore TWSServerTrustFile.jks -storepass <password>
   e. keytool -delete -alias **twstrustkeyold** -keystore TWSServerTrustFile.jks -storepass <password>
   f. Delete the old CNs from BrokerWorkstation.properties file

      Broker.AuthorizedCNs=~~Server;ServerNew;~~**glowfish**

4. Delete the default certificates from the two KeyStores of the DA.
   a. cd <Install_Dir>/TWSDATA/**ssl**
   b. keytool -delete -alias **serverold** -keystore **TWSClientKeyStoreJKS.jks** -storepass <password>
   c. cd <Install_Dir>/TWSDATA/ssl/**GSKit**
   d. gsk8capicmd_64 -cert -delete -label **server** -db **TWSClientKeyStore.kdb** -stashed
5. Delete the default certificates from the trusted certificates PEM file, ca.crt, on the **FTAs** and **DMs**
   1. cd <Install_Dir>/TWSDATA/ssl/**OpenSSL**
   2. vi **ca.crt**
   3. The default certificates should be at the top of the file. They are easy to identify as they are only 1K in key length and hence are much shorter than the new certificate. Delete the 1st 4 certificates.
   4. Restart the FTAs and DMs
   5. Verify that they link with the MDM
6. Delete the default certificates from the trusted certificates PEM file, ca.crt, from the **FTA** on the **MDM**
   a. cd <Install_Dir>/TWSDATA/ssl/**OpenSSL**
   b. vi **ca.crt**
   c. The default certificates should be at the top of the file. They are easy to identify as they are only 1K in key length and hence are much shorter than the new certificate. Delete the 1st 4 certificates.
   b. Verify that the FTAs are still linked with the MDM

4.9 Renewing the Expired Certificate

Each certificate has an expiration date and it must be renewed ahead of that date in order to keep the various components communicating with each other and not interrupt the execution of the workload automation.

The AgentCertificateDownloader script is also capable of downloading certificates automatically 15 days before the expiration. Place new certificates in PEM format on the MDM's depot directory for the AgentCertificateDownloader script to automatically download and import them.

Follow the steps below to renew the Certificates for the MDM and DWC.

1. Follow the procedure in the organization to request for certificates. If the original CSR created earlier is still available, send it to the CA and have it signed. If it is not available, generate a new CSR according to the process established by the security team.
2. Once the new certificate is received, delete the expired certificate from the KeyStore and TrustStore.
3. Import the new certificate into the KeyStores and TrustStores following the steps described in the previous sections.
4. The CA Certificate chain usually has a long expiration time and may not need to be renewed. If it is approaching expiration or the CA that signs the certificate has changed, import the public certificate chain into the TrustStore.

5 Other Resources Regarding Replacing Certificates

1. Refer to the following blog for a procedure on how to replace the default certificates with **CA signed** certificates. Note that this procedure adopts the **rip and replace** method where certificates on all components must be replaced at the same time to avoid any down time. This may be suitable for smaller and non-complex environments.

   Replacing Default SSL Certificates with CA signed Custom Certificates

2. Refer to the following Technote for a procedure on how to replace the older weak default certificates with **self-signed** certificates using the **same CN** (Common Name) as the default certificates. Note that this procedure adopts the **rip and replace** method where certificates on all components must be replaced at the same time to avoid any down time. This may be suitable for smaller and simpler environments.

   How to Substitute older weak default certificates (1024 key size) with new stronger default certificates (2048 key size)

6 Author's Bio

Sajjad Kabir is a Solutions Architect and a certified Workload Automation consultant with over 30 years of experience. He has a B.Sc. in Computer Systems Engineering from Western Michigan University, Kalamazoo, MI. Sajjad started working with IBM Workload Automation in 1998 while on assignment in IBM Singapore. He has been assisting clients deploy Workload Automation solutions worldwide. He has worked with clients of all sizes and complexities. He has published an IBM Redbook and numerous articles and blogs. He loves to work with people, especially who have just started their endeavors with workload automation. With the transition from IBM to HCL, he has mentored HCL consultants and continued to assist clients with implementing automation solutions on-prem and in the cloud. Sajjad loves to travel, enjoys ethnic cuisines, and SCUBA diving in exotic places around the world.