MICROSOFT SIGNTOOL

Integration Guide

Applicable Devices:
KMES Series 3

TABLE OF CONTENTS

[1] DOCUMENT INFORMATION

[1.1] DOCUMENT OVERVIEW

The purpose of this document is to provide information regarding the configuration of the Futurex KMES Series 3 with Microsoft SignTool using the Futurex CNG library. For additional questions related to your KMES Series 3 device, see the relevant administrator's guide.

[1.2] APPLICATION DESCRIPTION

From Microsoft's developer documentation website: "SignTool is a command-line tool that digitally signs files, verifies the signatures in files, and timestamps files."

[1.3] PURPOSE OF THE INTEGRATION

Microsoft SignTool uses a code signing certificate to digitally sign files, verify signatures in files, and timestamp files. By integrating with the KMES Series 3, the private key of the code signing certificate is stored on and retrieved from the KMES every time a SignTool command is run.

[2] PREREQUISITES

Supported Hardware:

- KMES Series 3, version 6.1.4.4 and above, with the *PKCS11* license enabled

Supported Operating Systems:

- Windows 7 and above

Other:

- OpenSSL
- Microsoft SignTool (**NOTE:** SignTool is available as part of the Windows SDK, which you can download from https://developer.microsoft.com/windows/downloads/windows-10-sdk/)

[3] INSTALL FUTUREX CNG (FXCNG)

In a Windows environment, the easiest way to install the Futurex CNG (FXCNG) module is through installing **FXTools**. FXTools can be downloaded from the Futurex Portal. Step by step installation instructions are provided below:

NOTE: The Futurex CNG module needs to be installed on the computer/server where Microsoft SignTool will be used.

[3.1] INSTRUCTIONS FOR INSTALLING THE FXCNG MODULE USING FXTOOLS IN WINDOWS

- Run the FXTools installer as an administrator

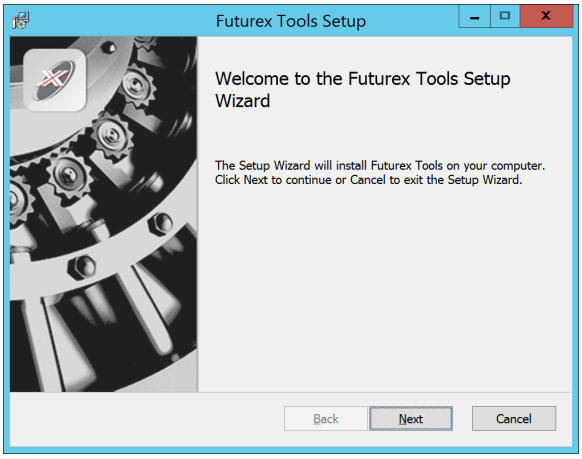


FIGURE: FUTUREX TOOLS SETUP WIZARD

By default, all tools are installed on the system. A user can overwrite and choose not to install certain modules.

- **Futurex Client Tools –**Command Line Interface (CLI) and associated SDK for both Java and C.
- **Futurex CNG Module –**The Microsoft Next Generation Cryptographic Library.
- **Futurex Cryptographic Service Provider (CSP) –**The legacy Microsoft cryptographic library.
- **Futurex EKM Module –**The Microsoft Enterprise Key Management library.
- **Futurex PKCS #11 Module –**The Futurex PKCS #11 library and associated tools.
- **Futurex Secure Access Client –**The client used to connect a Futurex Excrypt Touch to a local laptop, via USB, and a remote Futurex device.

After starting the installation, all noted services are installed. If the Futurex Secure Access Client was selected, the Futurex Excrypt Touch driver will also be installed (Note this sometimes will start minimized or in the background).

After installation is complete, all services are installed in the "*C:\Program Files\Futurex*" directory. The CNG Module, CSP Module, EKM Module, and PKCS #11 Module all require configuration files, located in their corresponding directory with a *.cfg* extension.

NOTE: Only the HSM version of the CNG configuration file is installed. For KMES integrations, the <HSM> section needs to be replaced with a <KMS> section.

[4] KMES SERIES 3 CONFIGURATION

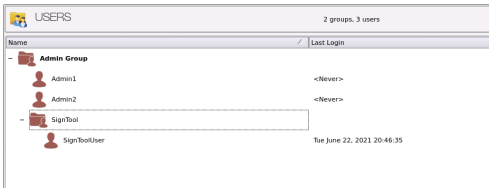
The first six subsections below cover general configurations that must be made on the KMES Series 3 to allow Microsoft SignTool to integrate with the KMES via the Futurex CNG library. The final subsection will cover the steps needed to configure TLS communication between the System/Host API port on the KMES and the Futurex CNG library.

[4.1] CREATE A USER FOR SIGNTOOL WITH THE REQUIRED PERMISSIONS

A new user group and user need to be created for SignTool on the KMES Series 3.

NOTE: In a later section, the name of this user will be configured inside of the Futurex CNG configuration file.

1. Log in to the KMES Series 3 application interface with the default Admin identities.
2. Go to the *Users* menu, expand the Admin user group, and click the **Add Group** button. This will open the *User Group Editor* dialog.
3. Specify a name for the user group, set the number of logins required to "1", and allow group members to authenticate to the Host API port only. All other fields can be left as the default values.
4. Move to the *Permissions* tab and select the following permissions:
    - Manage certificates -> Export
    - Manage certificates -> Upload
    - Manage keys (only the top-level "Manage keys" permission)
5. Click the **OK** button to finish creating the user group. The new user group should now be listed along with the other user groups that exist on the KMES Series 3.
6. Right-click on the newly created SignTool user group and select **Add** -> **User**.
7. In the *New User* dialog:
    a. Specify a user name.
    b. Set a password.
    c. Ensure that the "Allow password login" box is checked.
    d. Leave all other fields as the default values and click the **OK** button to finish creating the user. The new user should now be listed inside of the SignTool user group.



[4.2] ENABLE THE HOST API COMMANDS REQUIRED FOR THE MICROSOFT SIGNTOOL OPERATION

Because the Futurex CNG library will be connecting to the Host API port on the KMES, users must define which Host API commands will be enabled for execution by the FXCNG library. To set the enabled commands, complete the following steps:

1. Log in to the KMES Series 3 application interface with the default Admin identities.
2. Go to *Configuration -> Host API Options*, enable the commands listed below, then click **Save**.

- **ECHO**: Communication Test/Retrieve Version
- **RAFA**: Filter Issuance Policy
- **RAGA**: Retrieve Issuance Policy
- **RAGO**: Retrieve Request (Hash Signing)
- **RAUO**: Upload Request (Hash Signing)
- **RAGZ**: Retrieve Request (Authenticode)
- **RAUZ**: Upload Request (Authenticode)
- **RAGJ**: Retrieve Request (Jar Signing)
- **RAUJ**: Upload Request (Jar Signing)
- **RKCP**: Get Command Permissions
- **RKLN**: Lookup Objects
- **RKLO**: Login User
- **RKRK**: Retrieve Generated Keys

[4.3] CREATE A SIGNING APPROVAL GROUP

1. Select *Signing Approval* in the left menu, then click the **Add Approval Group...** button at the bottom of the page.
2. Set a name for the Approval Group, such as "SignTool", then click **OK** to save.
3. Right-click on the **SignTool** Approval Group, then select **Permission...**
4. Give the **SignTool** User Group the "Use" permission, then click **OK**.

[4.4] CREATE A CODE SIGNING CERTIFICATE

This section will describe three different methods that can be used to issue a code signing certificate.

[4.4.1] Issued by a CA on the KMES

1. Go to the *Certificate Authorities* menu and click the **Add CA...** button at the bottom of the page.
2. In the *Certificate Authority* dialog, enter a name for the Certificate Container, such as "KMES Issued". Set the owner field to the group that contains the user created in section 4.1, then click **OK**. The new Certificate Container will be listed now in the Certificate Authorities menu.
3. Right-click on the **KMES Issued** Certificate Container and select **Add Certificate -> New Certificate...**
4. In the *Subject DN* tab, set a Common Name for the certificate, such as "Root".
5. In the *Basic Info* tab, change the Major key to the **PMK**. All other settings can be left as the default values.
6. In the *V3 Extensions* tab, select the "Example Certificate Authority" profile, then click **OK**. The **Root** CA certificate will be listed now under the "KMES Issued" Certificate Container.
7. Right-click on the **Root** CA certificate that was just created and select **Add Certificate** -> **New Certificate...**.
8. In the *Subject DN* tab, set a Common Name for the certificate, such as "Code Signing".
9. Move straight to the *V3 Extensions* tab, select the "Example Code Signing Certificate" profile, and then click **OK**. The **Code Signing** certificate will be listed now under the **Root** CA certificate inside of the **KMES Issued** Certificate Container.

[4.4.2] Issued by an external CA

For this method, the external CA certificate(s) need to be imported into an empty Certificate Container on the KMES. A Certificate Signing Request (CSR) will then be generated, which the external CA will use to issue a code signing certificate. The code signing certificate will then be imported into the Certificate Container on the KMES that contains the external CA certificate.

1. Go to the *Certificate Authorities* menu and click the **Add CA...** button at the bottom of the page.
2. In the *Certificate Authority* dialog, enter a name for the Certificate Container, such as "Externally Issued". Set the owner field to the group that contains the user created in section 4.1, then click **OK**. The new Certificate Container will be listed now in the Certificate Authorities menu.
3. Right-click on the **Externally Issued** Certificate Container and select **Import** -> **Certificate(s)...**. This will open the *Import Certificates* dialog.
4. Click the **Add...** button in the bottom left-hand portion of the dialog, then find and select the external CA certificate(s) that will be issuing the code signing certificate in the file browser. The CA certificate(s) will populate in the Verified section of the *Import Certificates* dialog.
5. Click **OK** to save. The external CA certificate(s) should be listed now in tree form under the **Externally Issued** Certificate Container.

6. Next, we'll create a placeholder code signing certificate, from which a CSR can be generated. Right-click on the lowest level CA certificate in the tree and select **Add Certificate** -> **Pending...**. This will open the *Create X.509 Certificate* dialog.
7. In the *Subject DN* tab, set a Common Name for the certificate, such as "Code Signing".
8. Leave all other values as the default and click **OK**. The **Code Signing** placeholder certificate will be listed now under the external CA certificate(s).
9. Right-click on placeholder **Code Signing** certificate and select **Export** -> **Signing Request...**. This will open the *Create PKCS #10 Request* dialog.
10. Leave all of the settings in the *Subject DN* tab as the default values.
11. In the *V3 Extensions* tab, select the "Example Code Signing Certificate" profile.
12. In the *PKCS #10 Info* tab, specify a save location for the CSR, then click **OK**. There should be a message stating that the certificate signing request was successfully written to the location you specified.
13. The CSR file then needs to be taken to an external certificate authority. Using the CSR, the external CA will issue a code signing certificate.

    NOTE: After the external CA issues the code signing certificate, it needs to be copied to the storage medium that is configured on the KMES.
14. In the *Certificate Authorities* menu on the KMES, right-click on the placeholder **Code Signing** certificate and select **Replace** -> **With Signed Certificate...**. This will open the *Import Certificates* dialog.
15. Click the **Add...** button in the bottom left-hand portion of the dialog, then find and select the externally signed code signing certificate in the file browser. The code signing certificate will populate under the CA certificate(s) in the Verified section of the *Import Certificates* dialog.
16. Click **OK** to save.

[4.4.3] Importing an existing code signing certificate as a PKCS #12 file

NOTE: To be able to perform the steps in this section you must go to **Configuration** -> **Options** and eanble the "Allow import of certificates using passwords" option.

1. Go to the *Certificate Authorities* menu and click the **Add CA...** button at the bottom of the page.
2. In the *Certificate Authority* dialog, enter a name for the Certificate Container, such as "Imported". Set the owner field to the group that contains the user created in section 4.1, then click **OK**. The new Certificate Container will be listed now in the Certificate Authorities menu.
3. Right-click on the **Imported** Certificate Container and select **Import** -> **PKCS12...**. This will open the *Import PKCS12* dialog.
4. Select the `.p12` file that you wish to import, then click **Next**.
5. Input the file password, then click **Next**.
6. Click **Finish** to initiate the import.

[4.5] EXPORT THE CODE SIGNING CERTIFICATE

NOTE: Regardless of which method was used in section 4.4 for creating a code signing certificate, the steps outlined below are the same.

1. Go to the *Certificate Authorities* menu.
2. Right-click on the code signing certificate that was configured in the previous subsection, then select **Export** -> **Certificate(s)...**.
3. Changing the encoding to **PEM**, then click the **Browse...** button and specify the location where you want to save the file.
4. Click **OK** to initate the export.

[4.6] EXPORT THE CA CERTIFICATE(S) THAT ISSUED THE CODE SIGNING CERTIFICATE

NOTE: Regardless of which method was used in section 4.4 for creating a code signing certificate, the steps outlined below are the same.

1. Go to the *Certificate Authorities* menu.
2. Right-click on the CA certificate that issued the code signing certificate, then select **Export** -> **Certificate (s)...**.
3. Changing the encoding to **PEM**, then click the **Browse...** button and specify the location where you want to save the file.
4. Click **OK** to initate the export.
5. Repeat steps 1-4 for any additional CA certificates that are in the certificate tree (if applicable).

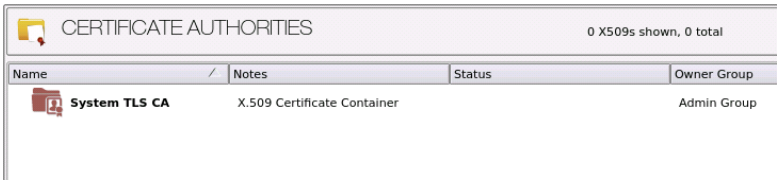
[4.7] APPLY AN ISSUANCE POLICY TO THE CODE SIGNING CERTIFICATE

1. Go to the *Certificate Authorities* menu.
2. Right-click on the code signing certificate that was configured, then select **Issuance Policy** -> **Add...**
3. In the *Basic Info* tab, set Approvals to **0**.
4. In the *X.509* tab, set the Default approval group to the **SignTool** approval group.
5. In the *Object Signing* tab, select the **Allow object signing** checkbox.
6. Click **OK** to apply the Issuance Policy to the code signing certificate.

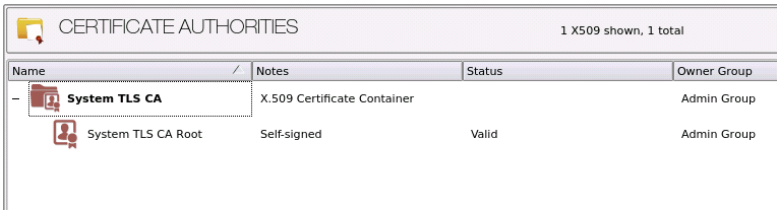
[4.8] CONFIGURE TLS COMMUNICATION BETWEEN THE KMES SERIES 3 AND THE FUTUREX CNG LIBRARY

[4.8.1] Create a Certificate Authority (CA)

1. Log in to the KMES Series 3 application interface with the default Admin identities.
2. Select *Certificate Authorities* in the left menu, then click the **Add CA...** button at the bottom of the page.
3. In the *Certificate Authority* dialog, enter a name for the Certificate Container, leave all other fields as the default values, then click **OK**.
4. The new Certificate Container will be listed now in the Certificate Authorities menu.



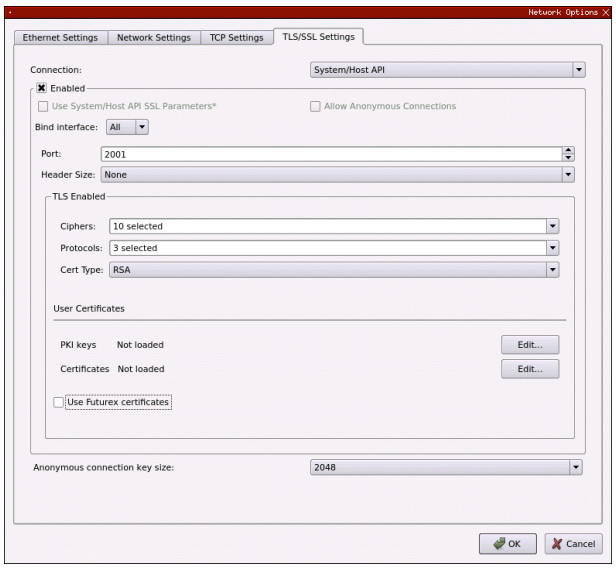
5. Right-click on the Certificate Container and select **Add Certificate -> New Certificate...**
6. In the *Subject DN* tab, set a Common Name for the certificate, such as "System TLS CA Root".
7. In the *Basic Info* tab, change the Major key to the **PMK**. All other settings can be left as the default values.
8. In the *V3 Extensions* tab, select the "Example Certificate Authority" profile, then click **OK**.
9. The root CA certificate will be listed now under the previously created Certificate Container.



[4.8.2] Generate a CSR for the System/Host API connection pair

1. Go to *Configuration -> Network Options*.
2. In the *Network Options* dialog, select the *TLS/SSL Settings* tab.

3. Under the **System/Host API** connection pair, uncheck "Use Futurex certificates", then click **Edit...** next to PKI keys in the User Certificates section.

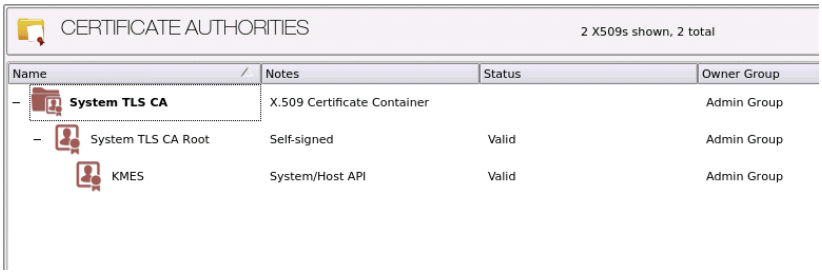


4. In the *Application Public Keys* dialog, click **Generate...**
5. There will be a warning stating that SSL will not be functional until new certificates are imported. Select **Yes** if you wish to continue.
6. In the *PKI Parameters* dialog, change the Encrypting key to the **PMK**, then change the Key Size to **2048** and click OK.
7. It should show that a PKI Key Pair is loaded now in the *Application Public Keys* dialog. If this is the case, click **Request...**
8. In the *Subject DN* tab, set a Common Name for the certificate, such as "KMES".
9. In the *V3 Extensions* tab, select the "Example TLS Server Certificate" profile.
10. In the *PKCS #10 Info* tab, select a save location for the CSR, then click **OK**.
11. There should be a message stating that the certificate signing request was successfully written to the file location that was selected. Click **OK**.
12. Click **OK** again to save the *Application Public Keys* settings.
13. In the main *Network Options* dialog, it should now say "Loaded" next to **PKI keys** for the System/Host API connection pair.

[4.8.3] Sign the System/Host API CSR

1. Go to the *Certificate Authorities* menu.
2. Right-click on the root CA certificate created in section 2.1.1, then select **Add Certificate** -> **From Request...**

3. In the file browser, find and select the CSR that was generated for the System/Host API connection pair.
4. Once loaded, none of the settings need to be modified for the certificate. Click **OK**.
5. The signed System/Host API certificate should now show under the root CA certificate on the *Certificate Authorities* page.



[4.8.4] Export the Root CA certificate

1. Go to the *Certificate Authorities* menu.
2. Right-click on the "System TLS CA Root" certificate, then select **Export** -> **Certificate(s)...**
3. In the *Export Certificate* dialog, change the encoding to "PEM", then click **Browse...**
4. In the file browser, navigate to the location where you want to save the Root CA certificate. Specify "tls_ca.pem" as the name for the file, then click **Open**.
5. Click **OK**. A message box will pop up stating that the PEM file was successfully written to the location that you specified.

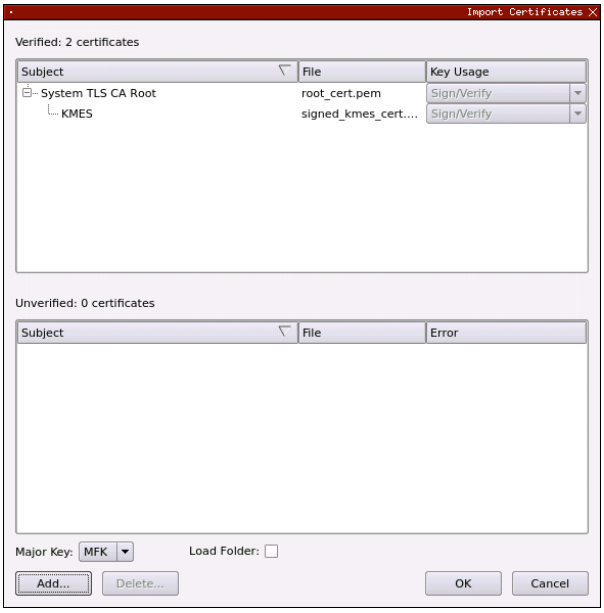
[4.8.5] Export the signed System/Host API certificate

1. Go to the *Certificate Authorities* menu.
2. Right-click on the "KMES" certificate, then select **Export** -> **Certificate(s)...**
3. In the *Export Certificate* dialog, change the encoding to "PEM", then click **Browse...**
4. In the file browser, navigate to the location where you want to save the signed System/Host API certificate. Specify `tls_ca.pem` as the name for the file, then click **Open**.
5. Click **OK**. A message box will pop up stating that the PEM file was successfully written to the location that you specified.

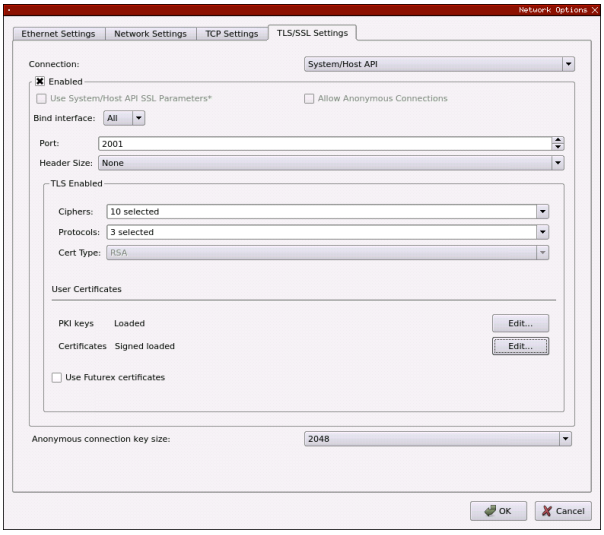
[4.8.6] Load the exported certificates into the System/Host API connection pair

1. Go to *Configuration -> Network Options*.
2. In the *Network Options* dialog, select the *TLS/SSL Settings* tab.
3. Click **Edit...** next to Certificates in the User Certificates section.

4. Right-click on the **System/Host API SSL CA** X.509 Certificate Container, then select **Import...**
5. Click **Add...** at the bottom of the *Import Certificates* dialog.
6. In the file browser, find and select both the root CA certificate and the signed System/Host API certificate, then click **Open**. The certificate chain should appear as shown below:



7. Click **OK** to save the changes. In the *Network Options* dialog, the System/Host API connection pair should show "Signed loaded" next to Certificates in the User Certificates section, as shown below:



8. Click **OK** to save and exit the Network Options dialog.

[4.8.7] Issue a client certificate for SignTool

NOTE: The client certificate that is being created for SignTool will be configured inside of the Futurex CNG configuration file.

1. Go to the *Certificate Authorities* menu.
2. Right-click on the **System TLS CA Root** certificate and select **Add Certificate** -> **New Certificate...**.
3. In the *Subject DN* tab, set a Common Name for the certificate, such as "SignTool".
4. All settings in the *Basic Info* tab should be left as the default values.
5. In the *V3 Extensions* tab, select the "Example TLS Client Certificate" profile, then click **OK**.
6. The **SignTool** certificate will be listed now under the **System TLS CA Root** certificate.

[4.8.8] Export the SignTool certificate as PKCS #12 file

NOTE: To be able to perform the steps below you must go to **Configuration** -> **Options** and enable the "Allow export of certificates using passwords" option.

1. Go to the *Certificate Authorities* menu.
2. Right-click on the **SignTool** certificate, then select **Export** -> **PKCS12...**
3. Make sure that the "Export Tree" option is selected, specify a unique name for the export file, then click **Next**.
4. Input a file password of your choosing, then click **Next**.
5. Click **Finish** to initate the export.

NOTE: The **SignTool** certificate and the **System TLS CA Root** certificate that was exported in section 4.4.4 both need to be moved to the computer that will be using the Microsoft SignTool application. In a later section, they will be configured inside of the Futurex CNG configuration file and used for TLS communication with the KMES Series 3.

[5] EDIT THE FUTUREX CNG (FXCNG) CONFIGURATION FILE

[5.1] DEFINE CONNECTION INFORMATION

The *fxcng.cfg* file allows the user to set the FXCNG library to connect to the KMES Series 3. To edit, run a text editor as an Administrator and edit the configuration file accordingly. Most notably, the fields shown below must be set inside the **<KMS>** section (note that the full *fxcng.cfg* file is not included).

NOTE: Our CNG library expects the CNG config file to be in a certain location (*C:\Program Files\Futurex\fxcng\fxcng.cfg*).

```
<KMS>
    # Which PKCS11 slot
    <SLOT>                  0                       </SLOT>

    # Login username
    <CRYPTO-OPR>            SignToolUser                </CRYPTO-OPR>
       <CRYPTO-OPR-PASS>       safest                  </CRYPTO-OPR-PASS>

    # Connection information
    <ADDRESS>               10.0.8.20 </ADDRESS>
    <PROD-PORT>             2001                    </PROD-PORT>
    <PROD-TLS-ENABLED>      YES                     </PROD-TLS-ENABLED>
    <PROD-TLS-ANONYMOUS>    NO                      </PROD-TLS-ANONYMOUS>
    <PROD-TLS-CA>           /home/bbarrows/tls/root.pem        </PROD-TLS-CA>
    <PROD-TLS-CERT>         /home/bbarrows/tls/signed-client-cert.pem      </PROD-TLS-CERT>
    <PROD-TLS-KEY>          /home/bbarrows/tls/PKI.p12    </PROD-TLS-KEY>
    <PROD-TLS-KEY-PASS>     safest                 </PROD-TLS-KEY-PASS>

    # YES = This is communicating through a Guardian
    <FX-LOAD-BALANCE>       NO                      </FX-LOAD-BALANCE>
</KMS>
```

The **<SLOT>** field can be left as the default value of 0.

In the **<CRYPTO-OPR>** field, specify the name of user that was created on the KMES in section 4.1.

In the **<CRYPTO-OPR-PASS>** field, specify the password of the user configured in the **<CRYPTO-OPR>** field.

In the **<ADDRESS>** field, specify the IP of the KMES that the CNG library should connect to.

In the **<LOG-FILE>** field, set the path to the CNG log file.

In the **<PROD-PORT>** field, set the CNG library to connect to the default Host API port on the KMES, port 2001.

The **<PROD-TLS-ENABLED>** field needs to be set to "YES" because the only way to connect to the Host API port on the KMES is over TLS.

The **<PROD-TLS-ANONYMOUS>** field defines whether the PKCS #11 library will be authenticating to the KMES or not. Since we're connecting to the Host API port, this value must be set to "NO".

The location of the CA certificate/s needs to be defined with one or more instances of the **<PROD-TLS-CA>** tag.

The location of the signed client certificate needs to be defined with the **<PROD-TLS-CERT>** tag.

The **<PROD-TLS-KEY>** tag defines the location of the client private key. Supported formats for the TLS private key are PKCS #1 clear private keys, PKCS #8 encrypted private keys, or a PKCS #12 file that contains the private key and certificates encrypted under the password specified in the **<PROD-TLS-KEY-PASS>** field.

Because a PKCS #12 file is defined in the **<PROD-TLS-KEY>** field in this example, it is not necessary to define the signed client cert with the **<PROD-TLS-CERT>** tag, or the CA cert/s with one or more instances of the **<PROD-TLS-CA>** tag.

If a Guardian is being used to manage KMES Series 3 devices in a cluster, the **<FX-LOAD-BALANCE>** field must be defined as "YES". If a Guardian is not being used it should be set to "NO".

For additional details, reference the Futurex CNG technical reference found on the Futurex Portal.

[6] IMPORTING CERTIFICATES INTO WINDOWS CERTIFICATE STORE

This section will explain how to import the code signing certificate (which Microsoft SignTool will use to sign files) and the CA certificate(s) that issued the code signing certificate into the Windows certificate store.

[6.1] IMPORT THE CA CERTIFICATE(S)

1. In Windows, open the "Manage computer certificates" application.
2. In the top menu, select *Action -> All Tasks -> Import*.
3. The Local Machine store should be selected. Click **Next**.
4. Click the **Browse** button, then find and select the CA certificate(s) that signed the code signing certificate that SignTool will be using. Click **Next**.
5. Select the option that will place all certificates in the "Trusted Root Certificate Authorities" store, then click **Next**.
6. Click **Finish**.

[6.2] IMPORT THE CODE SIGNING CERTIFICATE

1. In Windows, open the "Manage computer certificates" application.
2. In the top menu, select *Action -> All Tasks -> Import*.
3. The Local Machine store should be selected. Click **Next**.
4. Click the **Browse** button, then find and select the code signing certificate that SignTool will be using. Click **Next**.
5. Select the option that will place all certificates in the "Personal" store, then click **Next**.
6. Click **Finish**.

[7] ASSOCIATING A PRIVATE KEY WITH A CERTIFICATE

This section will explain how to associate a private key (stored on the KMES) with it's corresponding code signing certificate (stored in the Local Machine Windows certificate store).

The primary method of associating a private key with a certificate is via a tool called CertUtil. The primary resource for using the CertUtil command past any advanced case is this manual page. However, generally your use-case won't go past the command demonstrated below.

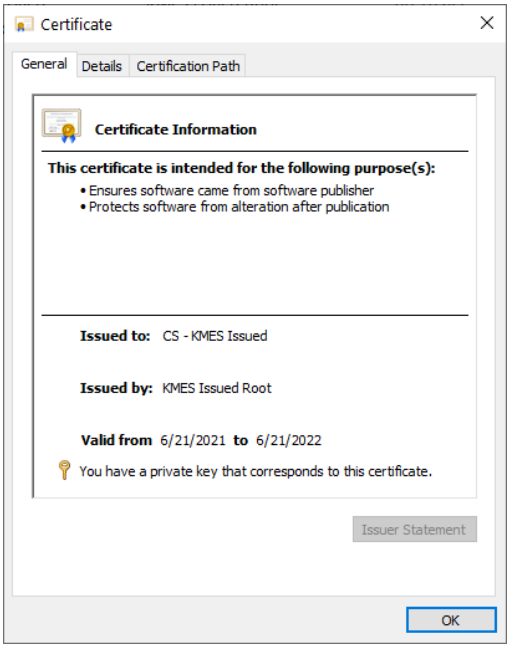
To associate a private key held in the KMES with a code signing certificate held in the Local Machine Windows certificate store, open the Command Prompt application and run the following command (replacing the fields surrounded in < and > symbols with the actual values that are required):

```
certutil -f -csp <human name of csp> -repairstore <store name> <serial number of cert in store>
```

As an example, the command could look like this:

```
certutil -f -csp "Futurex CNG" -repairstore My 00f204e900000070
```

For this integration, the CSP should be "Futurex CNG" and the store name should be "My". The "My" value tells the CertUtil command to look for the certificate in the X.509 certificate store for personal certificates, which is where we imported the code signing certificate in the previous section. The only field that should be changed is the serial number field. To find the serial number of your certificate, locate it in the Personal certificate store and double click on it. This will pull up the following dialog:



Select the *Details* tab, then note down the serial number that is listed for the certificate to use in the CertUtil command.

[8] TESTING MICROSOFT SIGNTOOL COMMANDS

In this section we'll run two Microsoft SignTool commands (i.e. `signtool sign` and `signtool verify`).

NOTE: The `signtool sign` command pertains more specifically to this integration, as it is the only SignTool command that initiates communication with the KMES Series 3. SignTool must be able to access the private key that is stored on the KMES in order to complete the code signing operation successfully.

[8.1] SIGN A FILE USING THE CONFIGURED CODE SIGNING CERTIFICATE

NOTE: In the following example an `.exe` file is being signed, but several other types of files can be signed using SignTool. Please reference the following url for details: https://docs.microsoft.com/en-us/windows/win32/seccrypto/cryptography-tools

Open the Windows Command Prompt application and run the following command:

NOTE: Replace "MyCertificate" with the Subject Name of your certificate and "example.exe" with the name of the file that you are signing.

```
signtool sign /sm /fd sha256 /s My /n "MyCertificate" example.exe
```

If the command is successful you should receive the following message:

```
Done Adding Additional Store
Successfully signed: example.exe
```

[8.2] VERIFY THE FILE THAT WAS SIGNED

To verify the file that was signed run the following command:

```
signtool verify /pa example.exe
```

If the command is successful, you should see output similar to the following:

```
File: example.exe
Index  Algorithm  Timestamp
========================================
0      sha1       None

Successfully verified: example.exe
```

APPENDIX A: XCEPTIONAL SUPPORT

In today's high-paced environment, we know you are looking for timely and effective resolutions for your mission-critical needs. That is why our Xceptional Support Team will help do whatever it takes to ensure you have the best experience and support possible. Every time. Guaranteed.

- 24x7x365 mission critical support
- Level 1 to level 3 support
- Extremely knowledgeable subject matter experts

At Futurex, we strive to supply you with the latest data encryption innovations as well as our best-in-class support services. Our Xceptional Support Team goes above and beyond to meet your needs and provide you with exclusive services that cannot be found anywhere else in the industry.

- Technical Services
- Onsite Training
- Virtual Training
- Customized Consulting
- Customized Software Solutions
- Secure Key Generation, Printing, and Mailing
- Remote Key Injection
- Certificate Authority Services

Toll-Free: 1-800-251-5112

E-mail: support@futurex.com

ENGINEERING CAMPUS

864 Old Boerne Road

Bulverde, Texas, USA 78163

Phone: +1 830-980-9782

+1 830-438-8782

E-mail: info@futurex.com

XCEPTIONAL SUPPORT

24x7x365

Toll-Free: 1-800-251-5112

E-mail: support@futurex.com

SOLUTIONS ARCHITECT

E-mail: solutions@futurex.com