HLK/HCK Signing Integration Document

Encryption Consulting LLC's expertise in this domain ensures your code signing process is seamless, secure, and compliant with Microsoft's standards. Digitally signing your HLK package (.hlkx) involves a series of steps to create a secure and verifiable digital signature. These steps ensure that the code meets Microsoft's standards.

We understand that flexibility is key, so we offer two methods for HLK signing, and you can pick the one that best suits your requirements. The first uses our in-house Utility Tool, which streamlines the signing process into a short series of prompts. The second uses our Key Storage Provider (KSP) together with Microsoft's signtool, which offers a higher level of security and is ideal for clients who prioritize security and compatibility. In both cases the private key remains inside the HSM and every signing operation is recorded centrally.

Sections 1 to 5 prepare the client machine and apply to both methods. Choose whichever of Sections 6 or 7 matches your preference, then verify using Section 8.

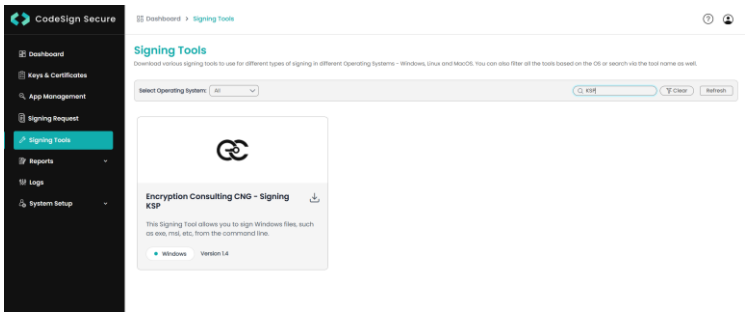
1. Set up CodeSign Secure KSP

The Encryption Consulting Key Storage Provider (KSP) for Windows is a software component that extends the Microsoft Cryptography API: Next Generation (CNG) framework. Its primary purpose is to enable Windows applications, such as signtool.exe and our Utility Tool, to interact seamlessly with the cryptographic keys and certificates stored within an HSM.

Steps:

Step 1: Download the EC KSP

- Log in to the CodeSign Secure portal and navigate to the Signing Tools section to download "Encryption Consulting CNG-SigningKSP" (also listed as "EC KSP for Windows").

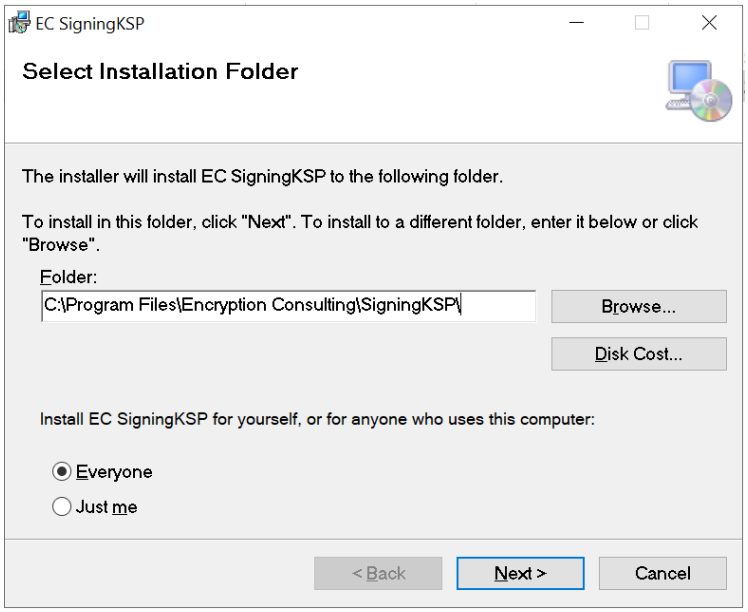


- Extract the zip file to get the "Setup.msi" file.

Step 2: Install the EC KSP

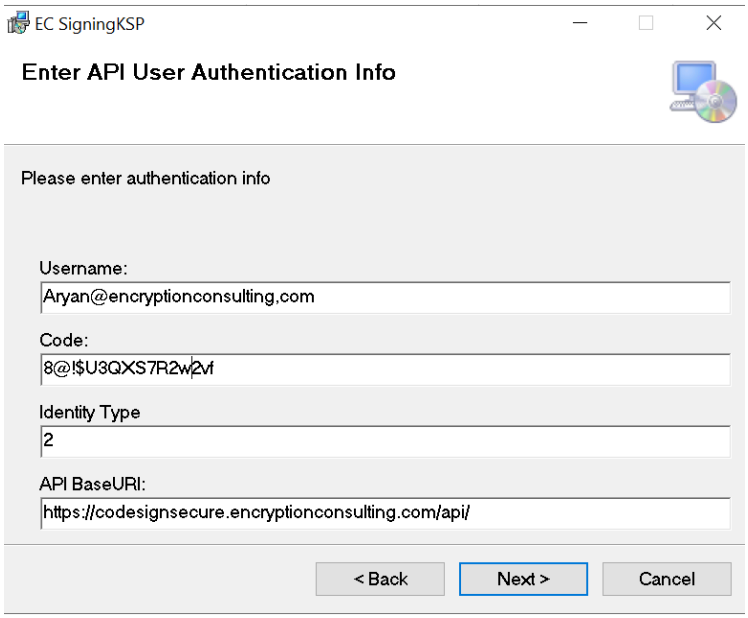
- Run the "Setup.msi" installer with Administrator privileges.

- Follow the on-screen prompts of the installation wizard.
  - Accept the End-User License Agreement.
  - Choose the installation directory (the default is C:\Program Files\Encryption Consulting\SigningKSP).
  - Choose whether you want to install the KSP for Everyone or just for the current user.

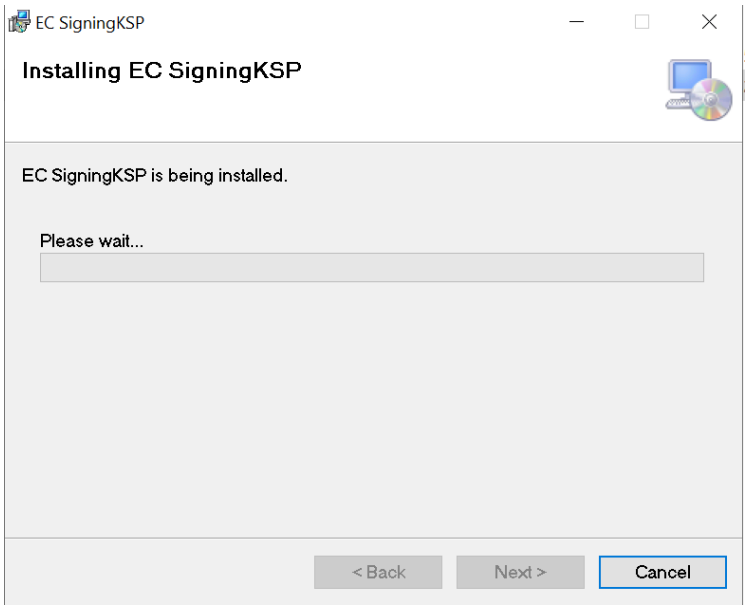


- Enter the prompted details such as:
  - **Username –** The username/email that you use to log in to the CodeSign Secure portal.
  - **Code –** The secret code that you set at the time of setting up the CodeSign Secure solution (this is the code defined in your conf.ini configuration file).

- **IdentityType –** Keep this field as default (2). If your deployment authenticates against a local identity store, set this to 1 as described in the product documentation.
- **CodeSign Secure URL –** The URL to access the portal (remember to add "/api/" at the end of the URL). Leave the API BaseURL unchanged if it is already populated correctly.

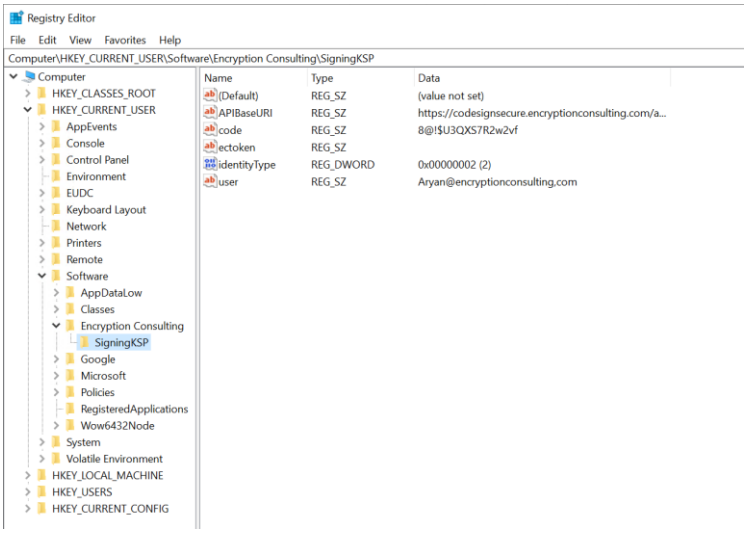


- Click Next and confirm the installation. When Windows asks whether you want to allow this program to make changes to your PC, click Yes.

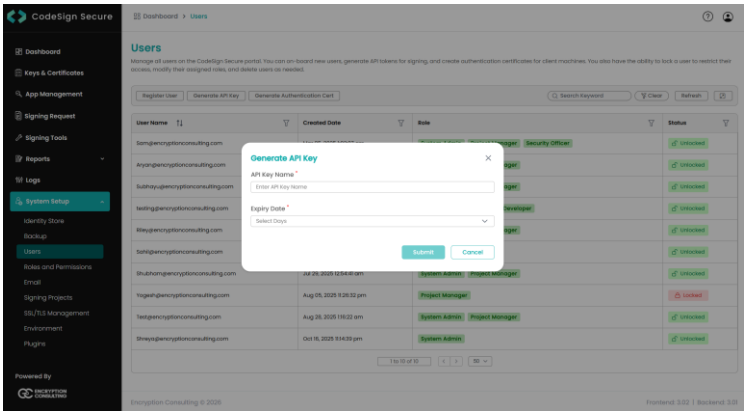


Step 3: Configure the Registry Editor settings

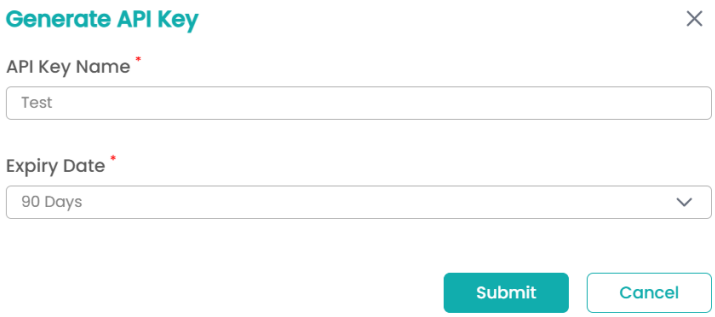
- Open the Registry Editor from the Start menu and navigate to HKEY_CURRENT_USER > Software > Encryption Consulting > SigningKSP.



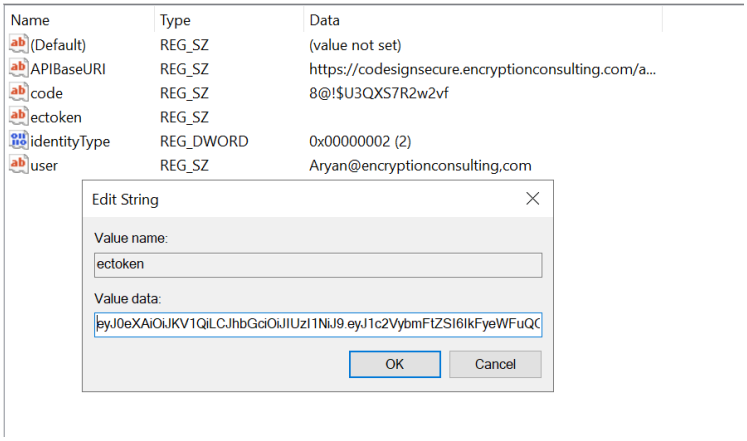
- Now open the CodeSign Secure portal and navigate to System Setup > User. Select the “Generate API Key” option.



- Create a token for your account by providing a name and the validity period. Remember to copy the token as it will be shown only once.



- Add this token to the “ectoken” field in the Registry Editor.



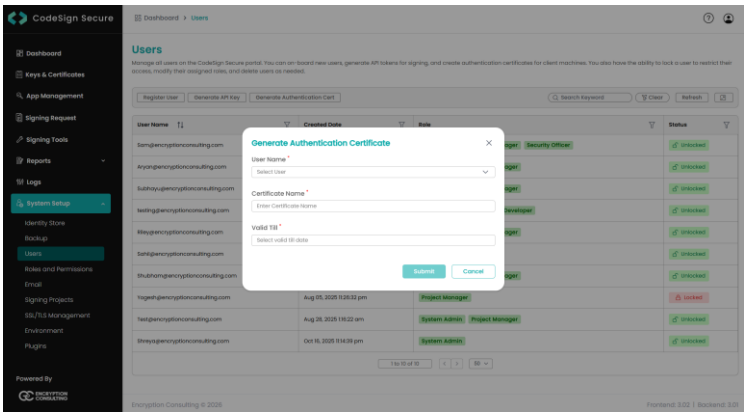
2. Set up P12 Authentication Certificate

Setting up a P12 Certificate involves configuring your environment variables to authenticate your client machine with Encryption Consulting's CodeSign Secure.

Steps:

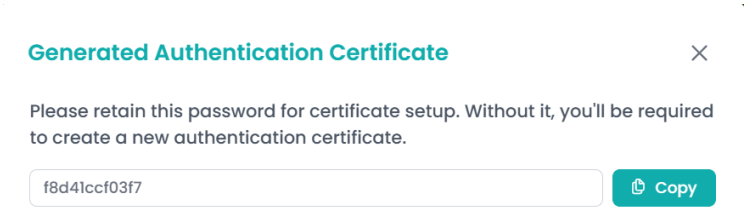
Step 1: Create a Machine Authentication Certificate

- Open the CodeSign Secure portal and navigate to System Setup > User. Select the "Generate Authentication Cert" option.



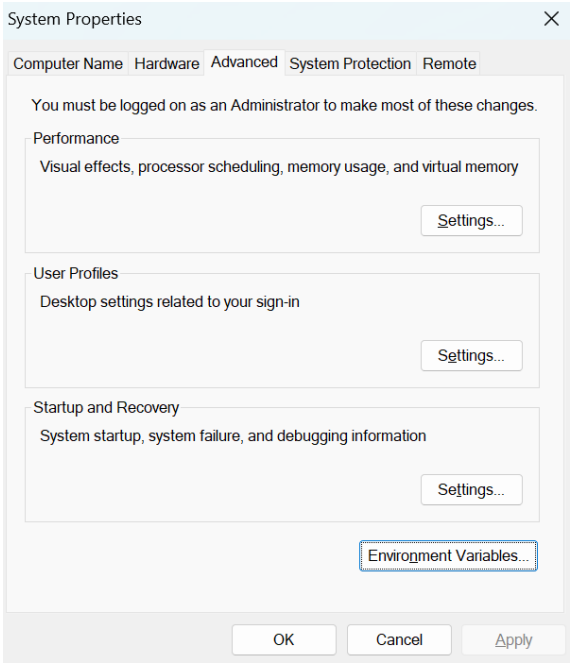
- Select the user name from the drop down and enter the details like certificate name and its expiry date.
- It will then provide you with a .pfx certificate file and also display the password to the certificate file.

NOTE: This password will be displayed only once. So you must copy and store it safely to perform the authentication with the CodeSign Secure server.

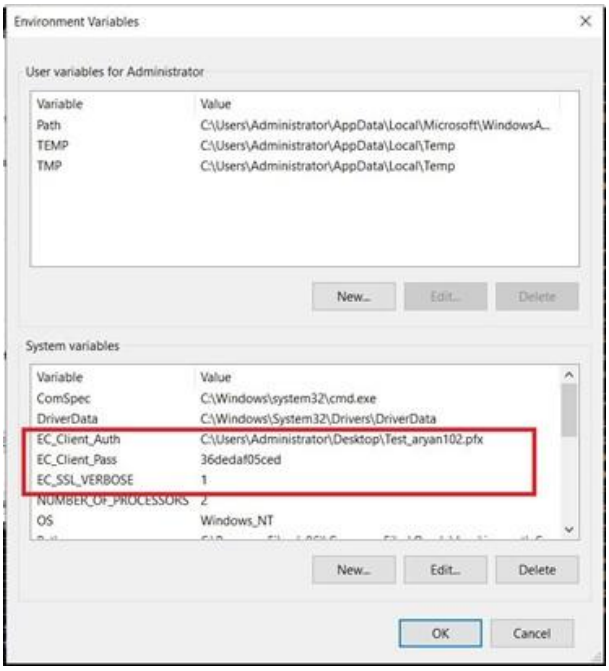


Step 2: Configure the Environment Variables

- Open the Environment Variables from your Start Menu.



- Add new system variables by clicking on the New button. Provide the following variable name and its corresponding details.
  - **EC_Client_Auth:** Corresponds to the path of your SSL Authentication certificate, which can be created from CodeSign Secure.
  - **EC_Client_Pass:** Corresponds to the password of your certificate, which is provided at the time of creation of the certificate.
  - **EC_SSL_VERBOSE:** Corresponds to the setting to either enable (1) or disable (0) the debugging output for EC KSP.



3. Install the Utility Tool and OpenSSL

The following components must be present on the client machine before signing:

- Encryption Consulting's Utility Tool installed in the system.
- OpenSSL configured in the system.
- Sample HLK/HCK file present in the system.

Steps:

Step 1: Download and Install OpenSSL

- Download and install OpenSSL on your system from here.

Step 2: Install the Encryption Consulting Utility Tool

- For installation, you would need to download and run an MSI file from the portal which will install this tool into your machine.
- Along with the tool, it will also prompt you to install various signing softwares such as Windows SDK, Java SDK and Encryption Consulting KSP. These signing softwares will be used by the Utility Tool to perform signing operations on different file types.

NOTE: If you have already installed the Encryption Consulting KSP in Section 1, you can skip that prompt during this installation. Allowing the Windows SDK prompt to run will also satisfy the signtool requirement in Section 4.

- After successful installation, you will find the Utility Tool.exe in the "C:\Program Files\Encryption Consulting\Utility Tool" directory.
- You will need to change the "BASEURL" field in the config file with the URL of your domain.

Step 3: Confirm the HLK/HCK File

- Place the .hlkx (or .hckx) package you intend to sign somewhere convenient on the machine and make a note of its full path. Both signing methods below reference this path.

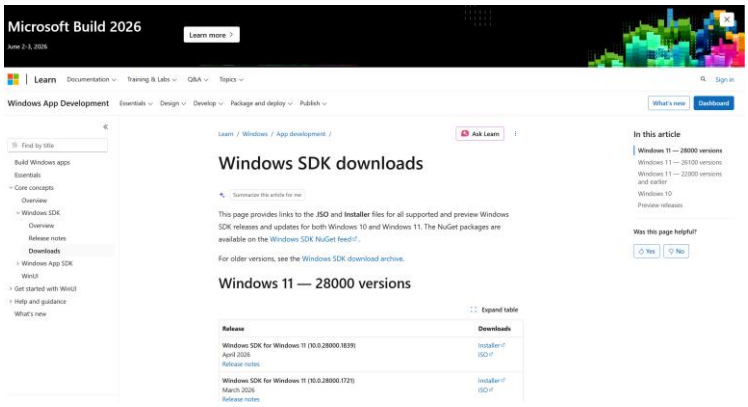
4. Set up Signtool (for the KSP Method)

If you intend to use Section 7, the KSP with signtool, then signtool.exe must be present and reachable. It ships with the Microsoft Windows SDK. Skip this section if you are only using the Utility Tool method in Section 6.

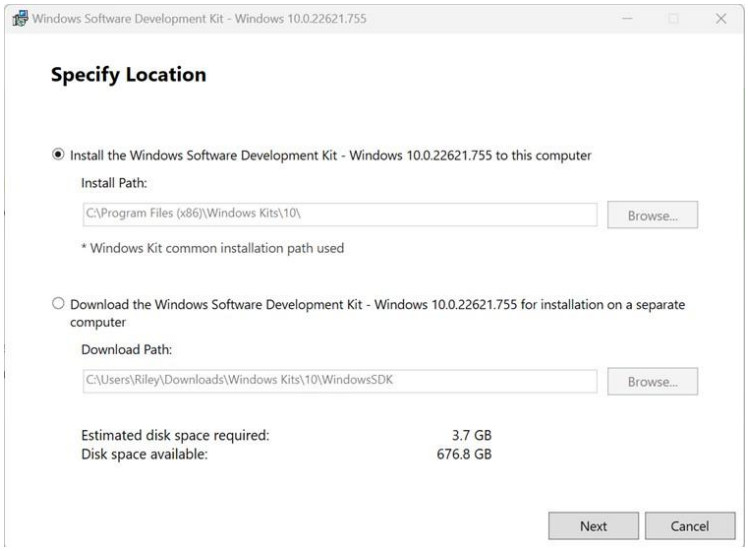
Steps:

Step 1: Download and Install Windows SDK

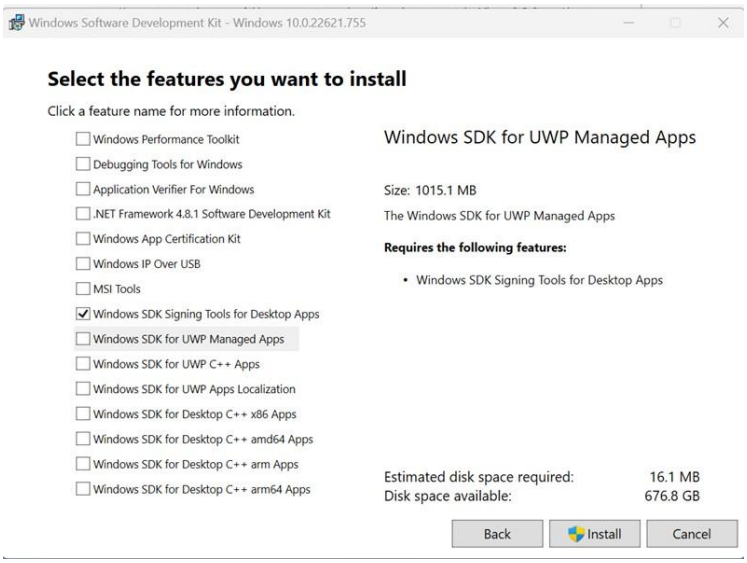
- Using the following download link, download the Windows Software Development Kit: [Windows SDK downloads - Windows apps | Microsoft Learn]



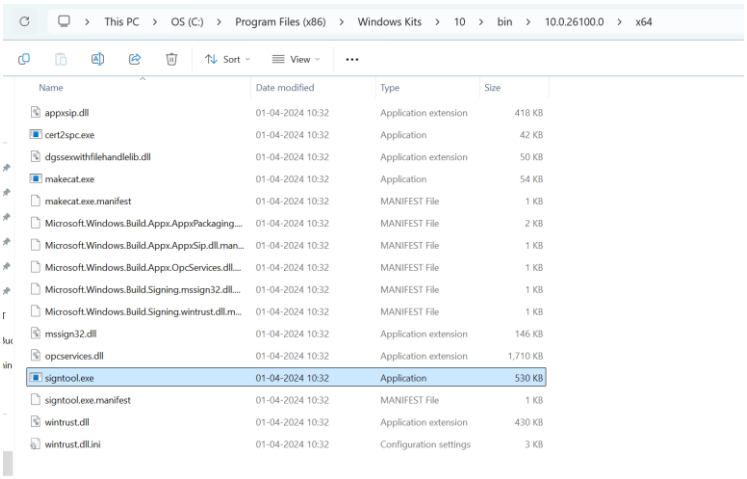
- Open the installer once downloaded and select “Next” on the first screen to keep the default settings.



- Follow the on-screen prompts of the installation wizard.
  - Accept the Windows Kits Privacy notice.
  - Accept the End-User License Agreement.
- Select “Windows SDK Signing Tools for Desktop Apps” and select “Install”.



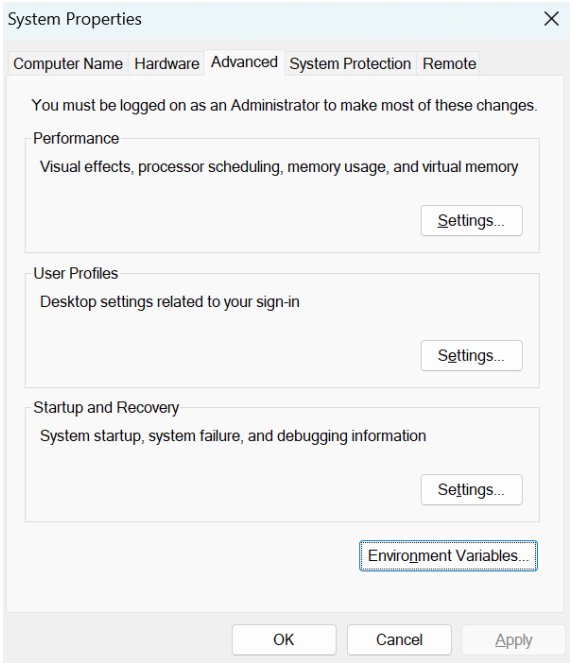
- Go to the following path where the tools should have been downloaded to: “C:\Program Files (x86)\Windows Kits\10\bin”. Select the desired version directory and check whether the “signtool.exe” file is present.



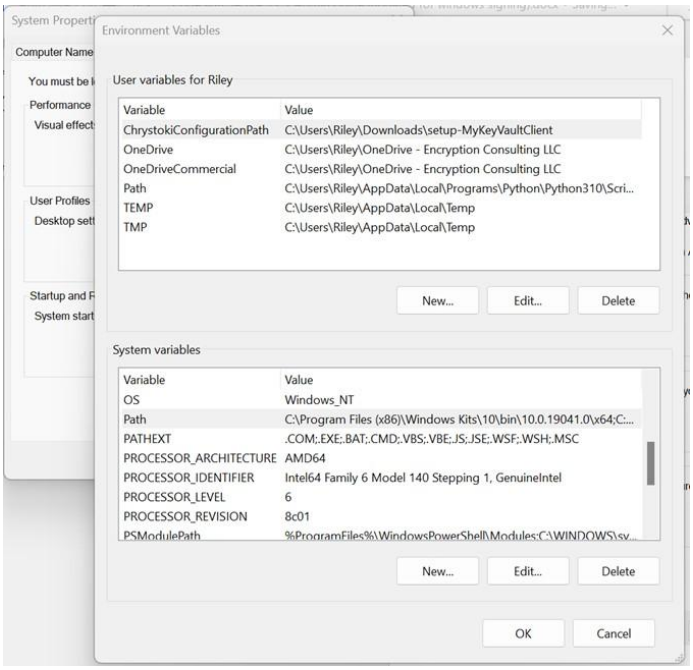
- Ensure you are in the x64 directory and copy this directory path.

Step 2: Add Path to Signtool.exe in Environment Variables

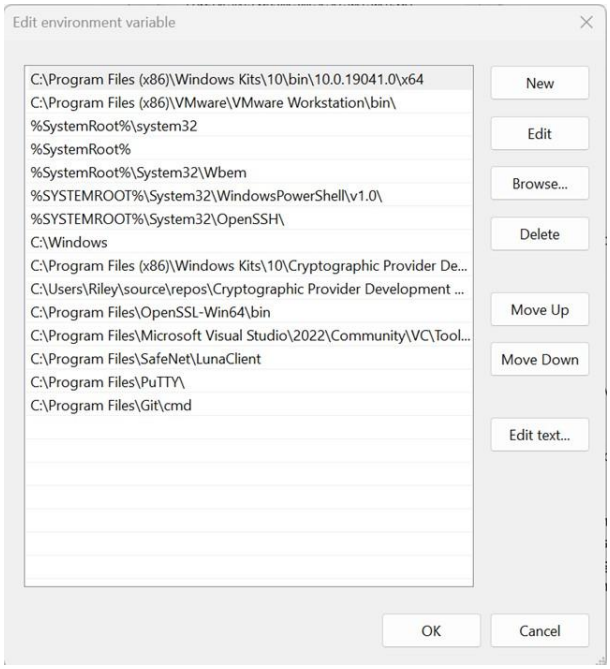
- Open the Environment Variables from the Start Menu.



- Scroll down through the system variables on the bottom table until you find PATH in the variable names.



- Double click on PATH in system variables and select New on the left of the screen. Paste your copied directory path of “signtool.exe” into the new selection.



- Select OK at the bottom of each page to exit the Environment Variables page.

5. Obtain the Signing Certificate

Both signing methods reference a public certificate whose private key is held in the HSM. Signtool requires this certificate in PEM format.

Steps:

Step 1: Download the Public Certificate

- Download the public certificate you wish to use for signing from the Keys and Certificates section of the CodeSign Secure portal, and make a note of the key name (alias) associated with it.
- The two methods expect different formats of the same certificate. Signtool in Section 7 must be given a .pem file, while the Utility Tool in Section 6 is given a .crt file.
- Convert between the two using OpenSSL, which is why it is listed as a prerequisite in Section 3.

```
openssl x509 -outform der -in certificate.pem -out certificate.crt
```

NOTE: The key name you note here becomes the /kc value in Section 7, and the .pem path becomes the /f value.

6. Sign Using the Custom Utility Tool

This method involves leveraging our in-house utility tool designed to perform all kinds of Windows signing (even HLK signing). This tool streamlines the signing process, making it efficient and hassle-free.

Steps:

Step 1: Launch the Utility Tool

- Open the Utility Tool installed in Section 3 from the "C:\Program Files\Encryption Consulting\Utility Tool" directory.

Step 2: Enter the Signing Details

- Enter the appropriate details which will be prompted:
  - **Username –** The username you use to log in to CodeSign Secure.
  - **Application Name –** The signing project the request belongs to.
  - **Environment Name –** The environment configured for that project.
  - **Path of the file –** The .hlkx package to be signed, as noted in Section 3.
  - **Name for the output file –** The signed package to be produced. The tool writes a new file rather than modifying the original.
  - **Path to certificate –** The .crt certificate downloaded in Section 5.

NOTE: If your signing project requires approval, the tool sends an approval request and waits. Signing continues once approval has been granted.

Step 3: Confirm the Result

- The tool embeds the signature into the output file and reports that HLKX signing was successful, along with a verification status.

- Note that the signed package is the output file you named, not the original input file. Verify that file as described in Section 8.

7. Sign Using the KSP with Signtool

This method involves using our KSP (Encryption Consulting Key Storage Provider) with Microsoft's signtool. This method offers a higher level of security and is ideal for clients who prioritize security and compatibility.

Steps:

Step 1: Run the Signtool Command

- From an administrator Command Prompt, run the signing command against your .hlkx package. The command is:

```
signtool sign /csp "Encryption Consulting Key Storage Provider"
  /kc evcodesigning
  /fd SHA256
  /f C:\Users\riley\Desktop\ForTesting\evcodesigning.pem
  /tr http://timestamp.digicert.com
  /td SHA256
  C:\Users\riley\Desktop\CodeSign_UtilityTool\
    f9-879-e0-d5eba97-fa5264.hlkx
```

NOTE: Replace the /kc alias, the /f certificate path, and the final .hlkx path with the values for your own environment as noted in Section 5.

- On success, signtool reports that the package was successfully signed. Unlike the Utility Tool, signtool signs the package in place rather than producing a separate output file.

Step 2: Flag Reference

- The following are the flags and their meaning in the command:

Flag	Description
/csp	To enter our KSP (Encryption Consulting Key Storage Provider). This will not vary and remains constant.
/kc	A key container within the CSP that holds the private key. Rename it to the private key alias you are going to use.
/fd	File digest algorithm to use for creating file signatures.
/f	Indicates the location of the public key file or certificate. Make sure to sign using a .pem file only.
/tr	Specifies the URL of the RFC 3161 timestamp server. This helps provide a timestamp for when the file was signed.
/td	Specifies the digest algorithm used by the RFC 3161 timestamp server.
<file path>	The path to the .hlkx or .hckx package to be signed.

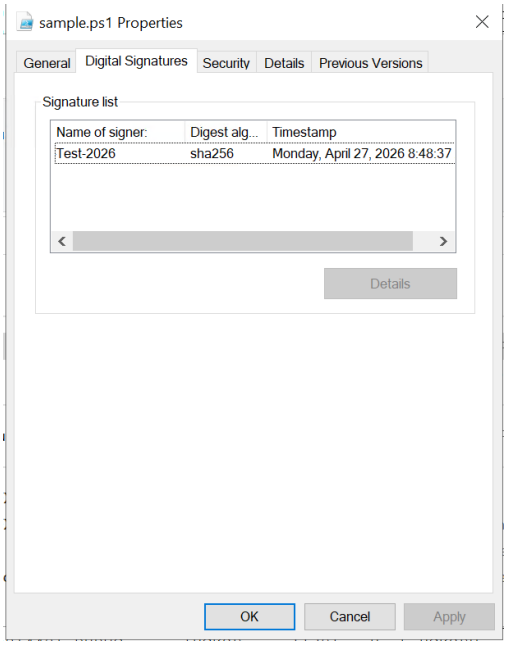
8. Verify the Signature

Whichever method you used, confirm that a valid digital signature has been applied to the package.

Steps:

Step 1: Review the Digital Signature

- Right click the signed package and select Properties.
- Select the Digital Signatures tab at the top of the Properties window.



- Select the name of the signer in the signature list and click Details to confirm that the signature is valid, that it chains to the expected certificate, and that the timestamp is present.

Step 2: Cross-Check in CodeSign Secure

- Open the CodeSign Secure portal and navigate to Reports > Signing Request Report, where every signing request performed through the KSP is recorded for audit purposes. Confirm that a signing request appears for the certificate you used.