Windows Signing (SignTool) Integration Document

CodeSign Secure lets you sign all types of Windows files — including .exe, .dll, .sys, .cab, .msi, and more — using Microsoft's Signtool utility, while the private key never leaves your HSM. Because the signing key stays under HSM protection and every operation is logged centrally, you get strong key custody and a complete audit trail without changing the way your developers already sign.

In this guide, we will show you how to sign a Windows file using Microsoft's Signtool with Encryption Consulting's Key Storage Provider (KSP) on a Windows machine, and how to verify the resulting signature.

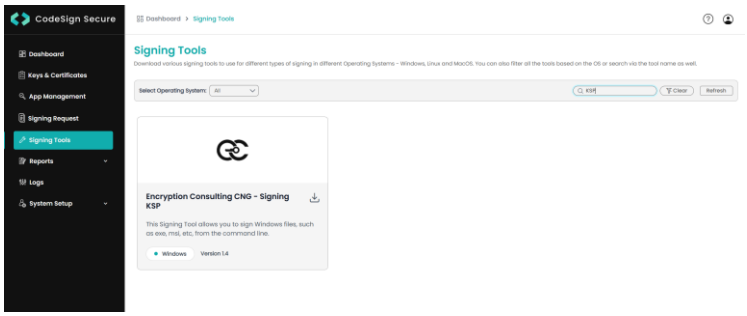
1. Set up CodeSign Secure KSP

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, 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").

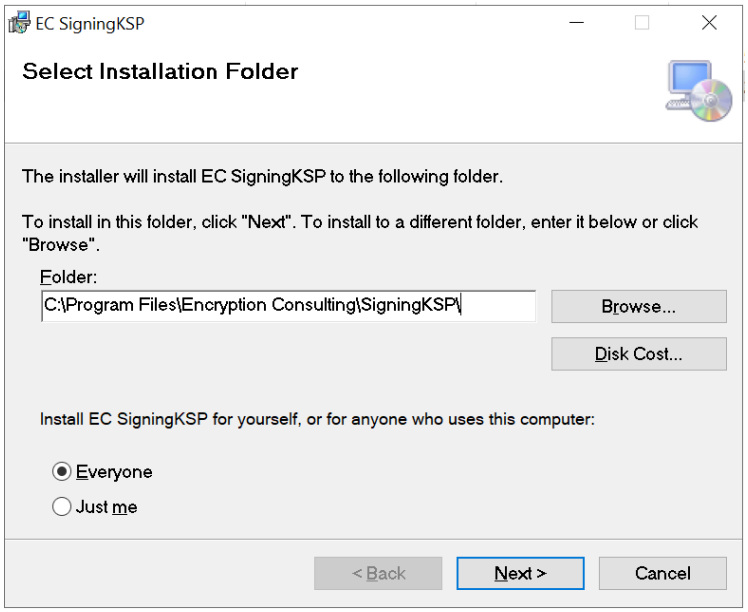


- Extract the zip file to get the "Setup.msi" file.

Step 2: Install the EC KSP

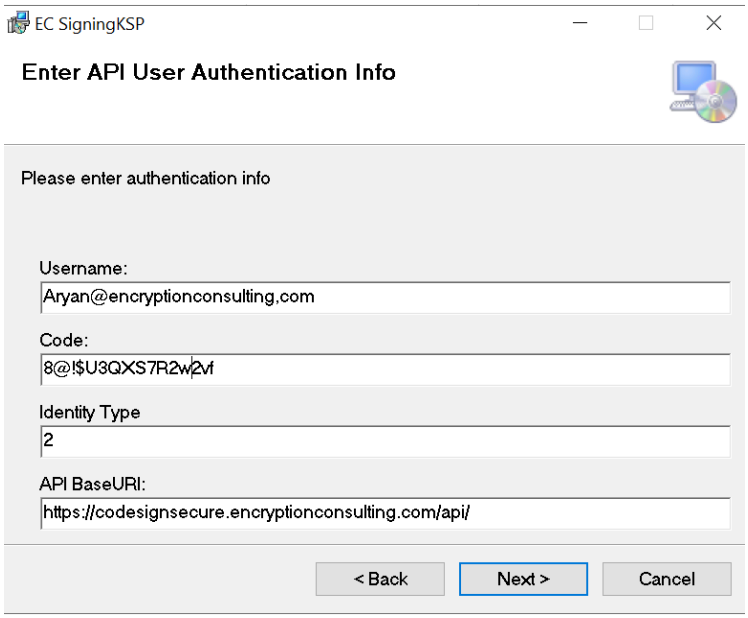
- Run the "Setup.msi" installer with Administrator privileges.

- 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.

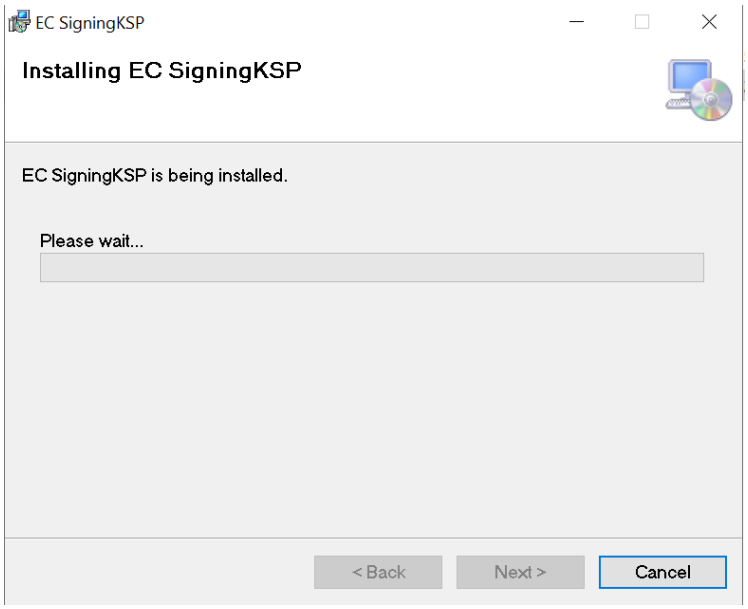


- Enter the prompted details such as:
  - **Username –** The username/email that you use to log in to the CodeSign Secure portal.
  - **Code –** The secret code that you set at the time of setting up the CodeSign Secure solution (this is the code defined in your conf.ini configuration file).

- **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.

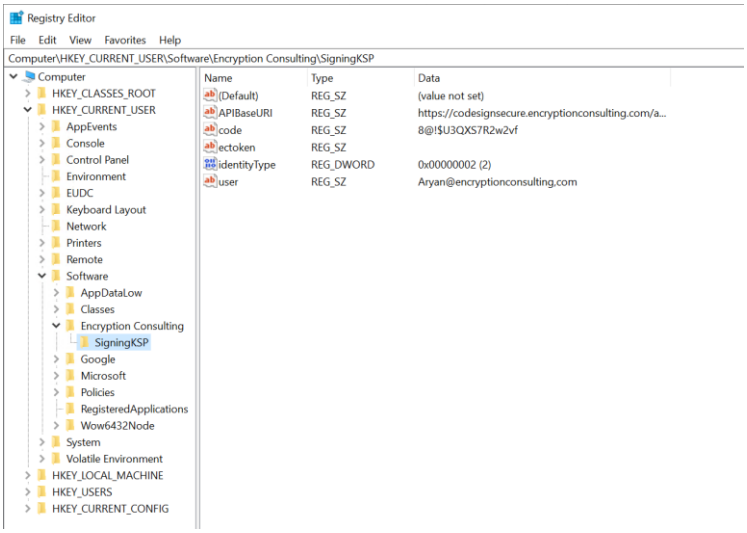


- Click Next and confirm the installation. When Windows asks whether you want to allow this program to make changes to your PC, click Yes.

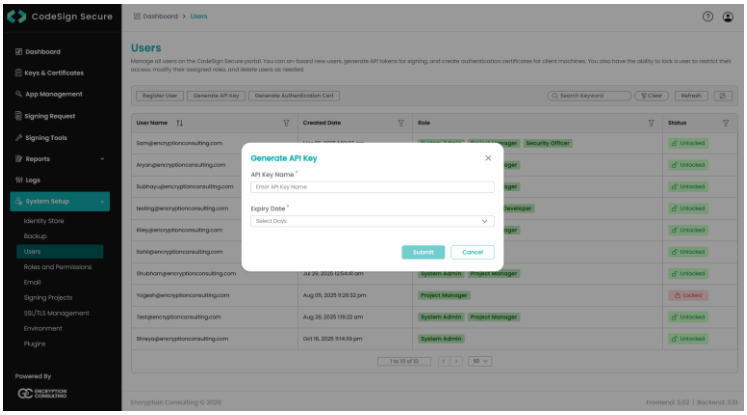


Step 3: Configure the Registry Editor settings

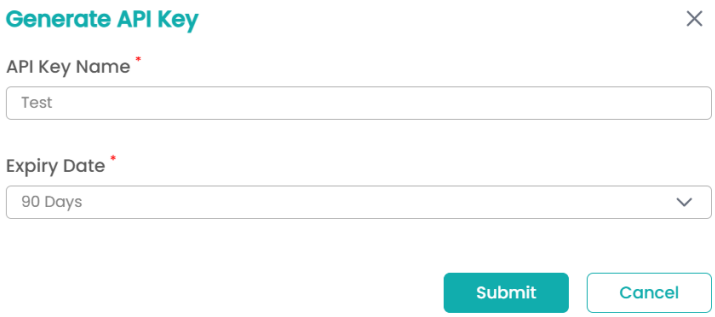
- Open the Registry Editor from the Start menu and navigate to HKEY_CURRENT_USER > Software > Encryption Consulting > SigningKSP.



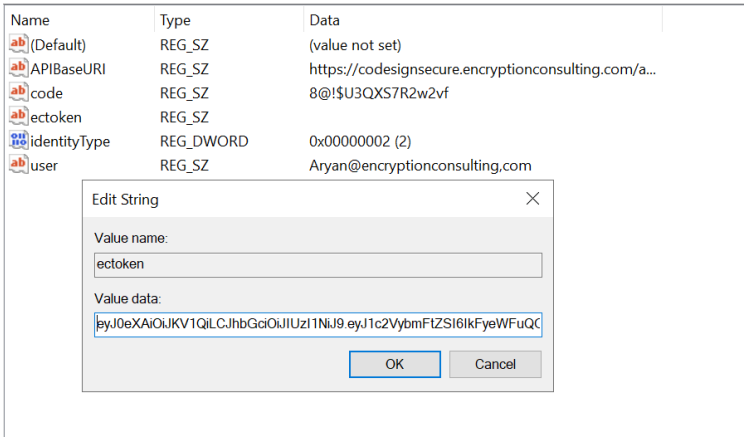
- Now open the CodeSign Secure portal and navigate to System Setup > User. Select the “Generate API Key” option.



- 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.



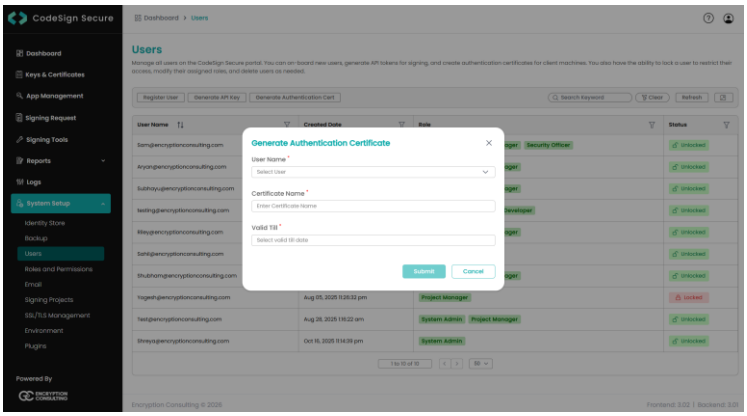
2. Set up P12 Authentication Certificate

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

Steps:

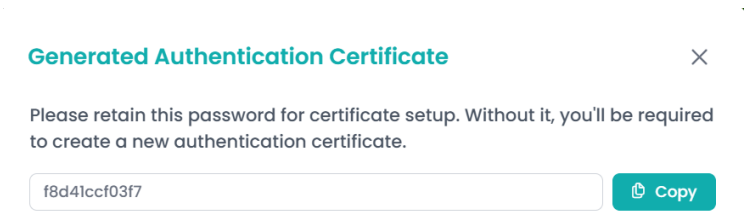
Step 1: Create a Machine Authentication Certificate

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



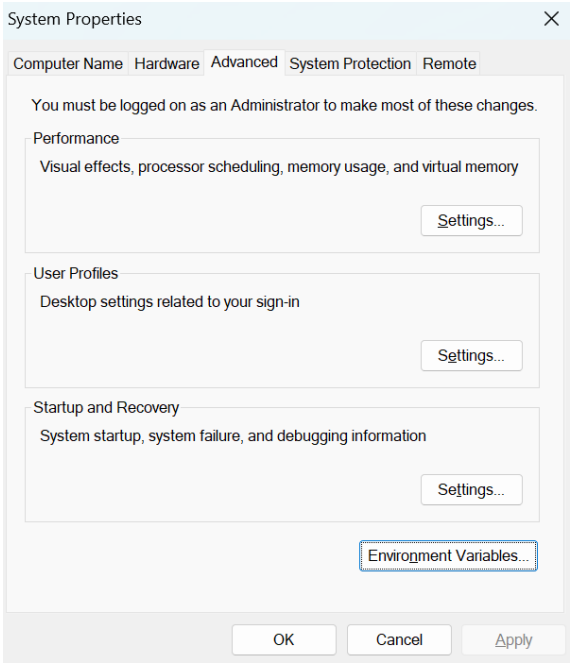
- Select the user name from the drop down and enter the details like certificate name and its expiry date.
- It will then provide you with a .pfx certificate file and also display the password to the certificate file.

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.

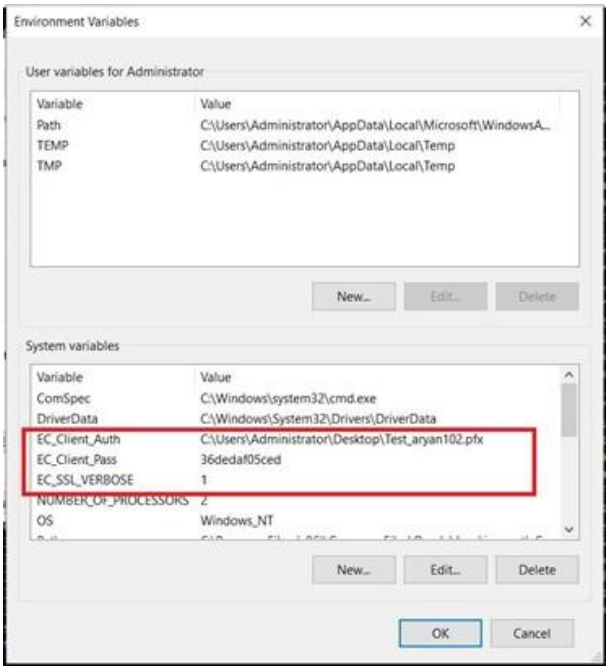


Step 2: Configure the Environment Variables

- 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.



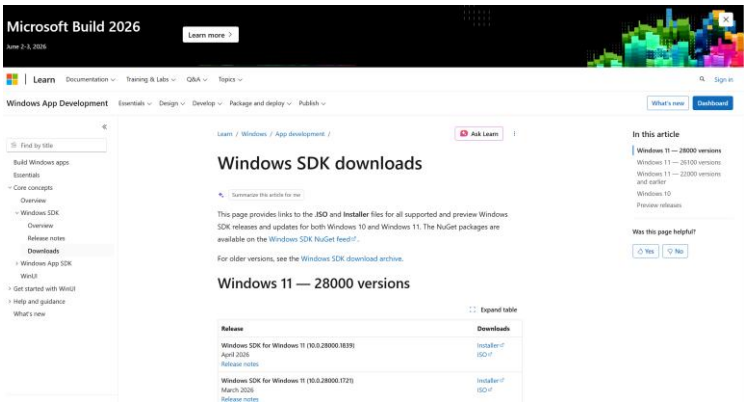
3. Set up Signtool for Signing

Setting up signtool for code signing involves ensuring that the Signtool.exe utility is available on your machine and configured to correctly interact with Encryption Consulting's cryptographic provider that provides access to your code signing certificate's private key.

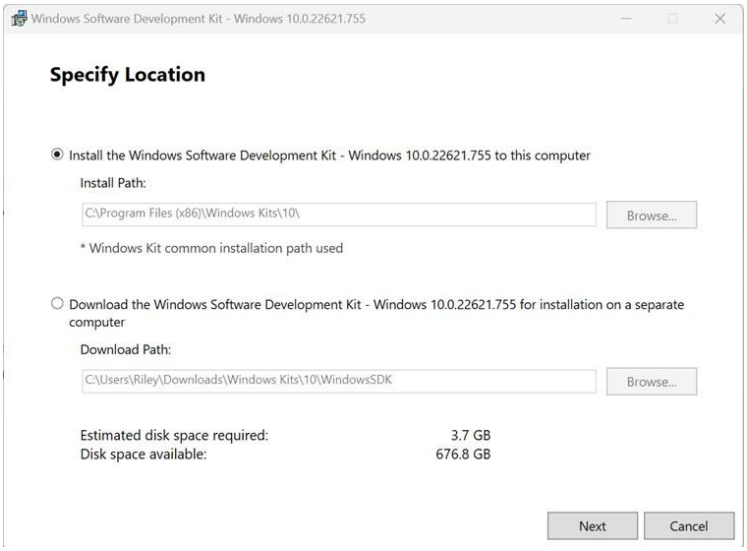
Steps:

Step 1: Download and Install Windows SDK

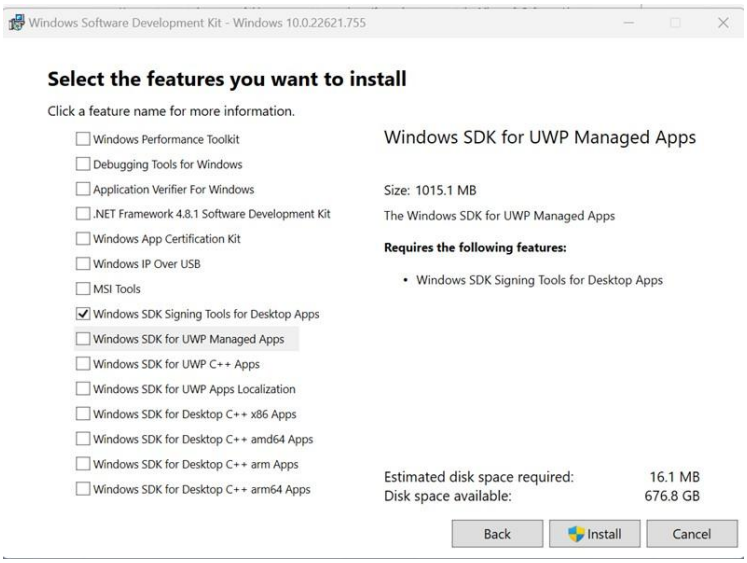
- Using the following download link, download the Windows Software Development Kit: [Windows SDK downloads - Windows apps | Microsoft Learn]



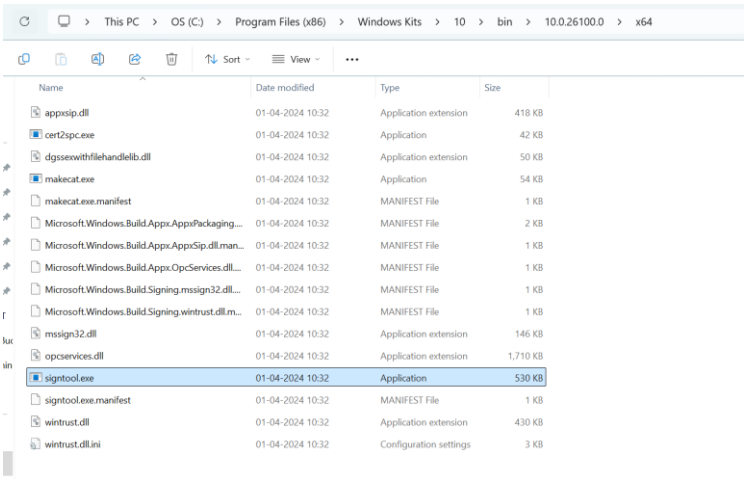
- Open the installer once downloaded and select "Next" on the first screen to keep the default settings.



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



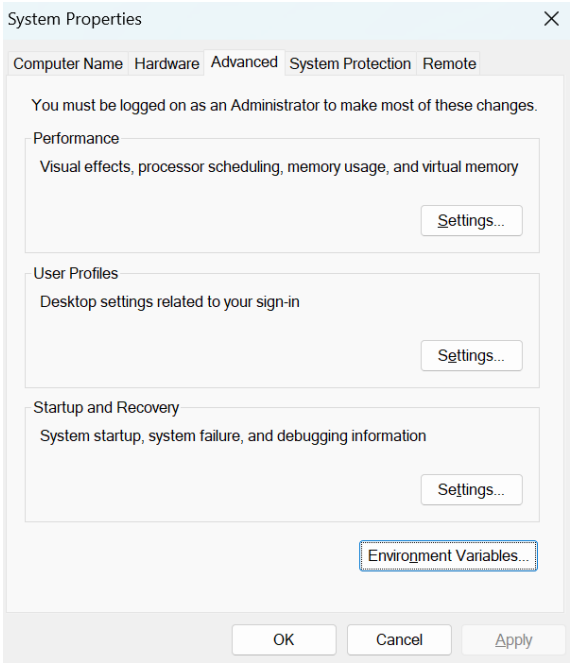
- Go to the following path where the tools should have been downloaded to: “C:\Program Files (x86)\Windows Kits\10\bin”. Select the desired version directory and check whether the “signtool.exe” file is present.



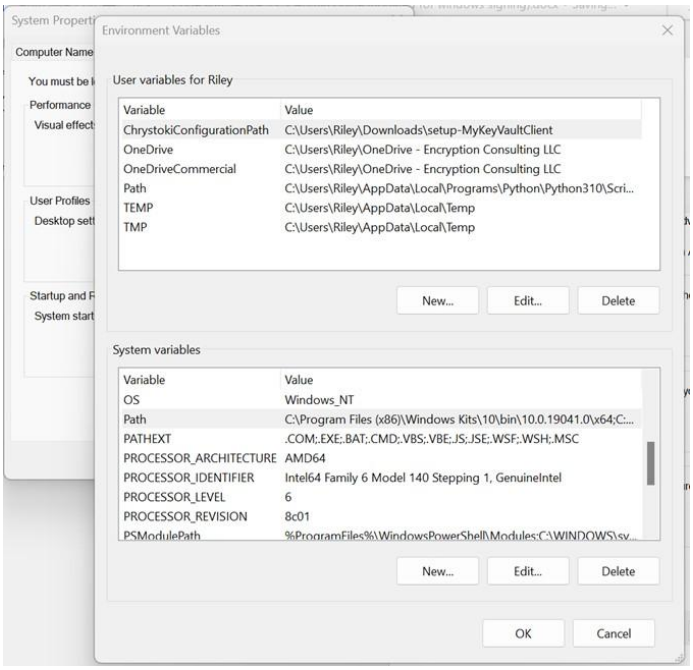
- Ensure you are in the x64 directory and copy this directory path.

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

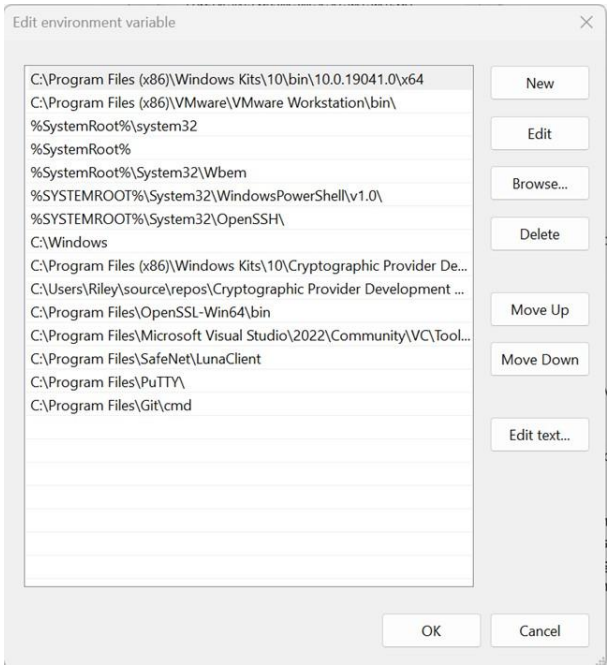
- Open the Environment Variables from the Start Menu.



- 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.

4. Sign a File Using Signtool

With the KSP installed, the authentication certificate in place, and signtool on your PATH, you are ready to sign. There are multiple ways to sign a file with signtool using a certificate; the procedure below signs a file using the public certificate issued from CodeSign Secure while the private key remains in the HSM.

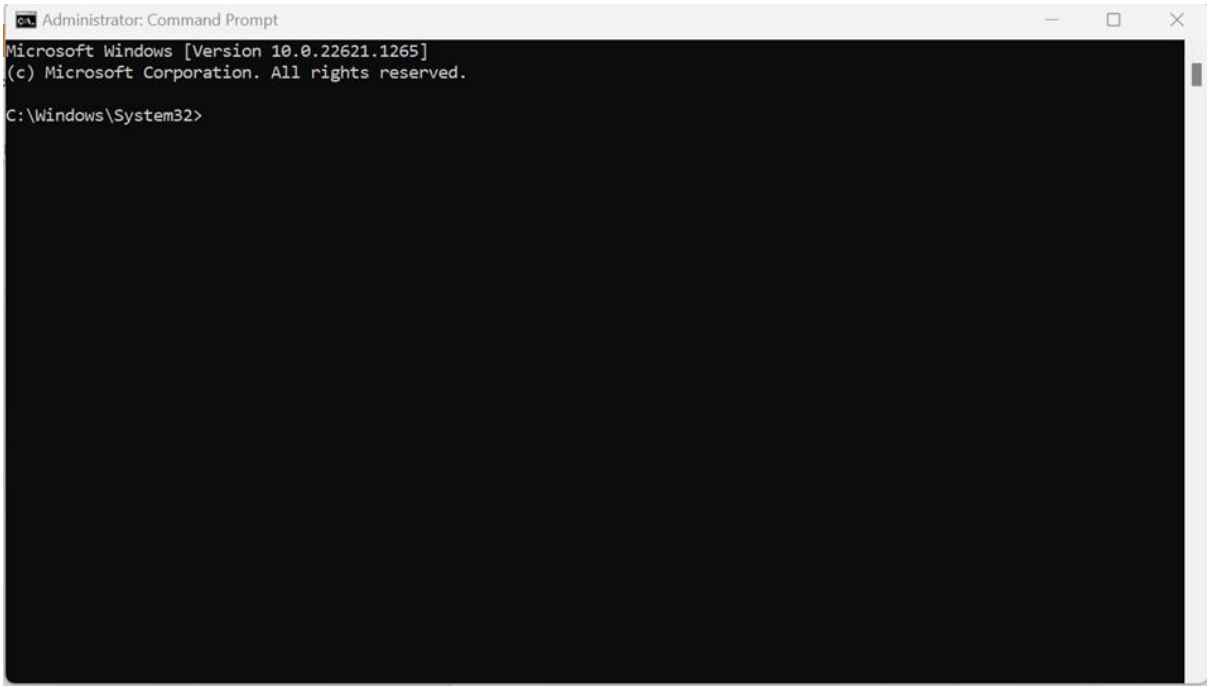
Steps:

Step 1: Obtain the Public Certificate

- You will need a copy of the public certificate. You can download it from the Keys and Certificates section of the CodeSign Secure portal.
- Place the certificate file in the same directory from which you will run the signing command.

Step 2: Open an Administrator Command Prompt

- Search for Command Prompt in the Start menu and select “Run as administrator”.

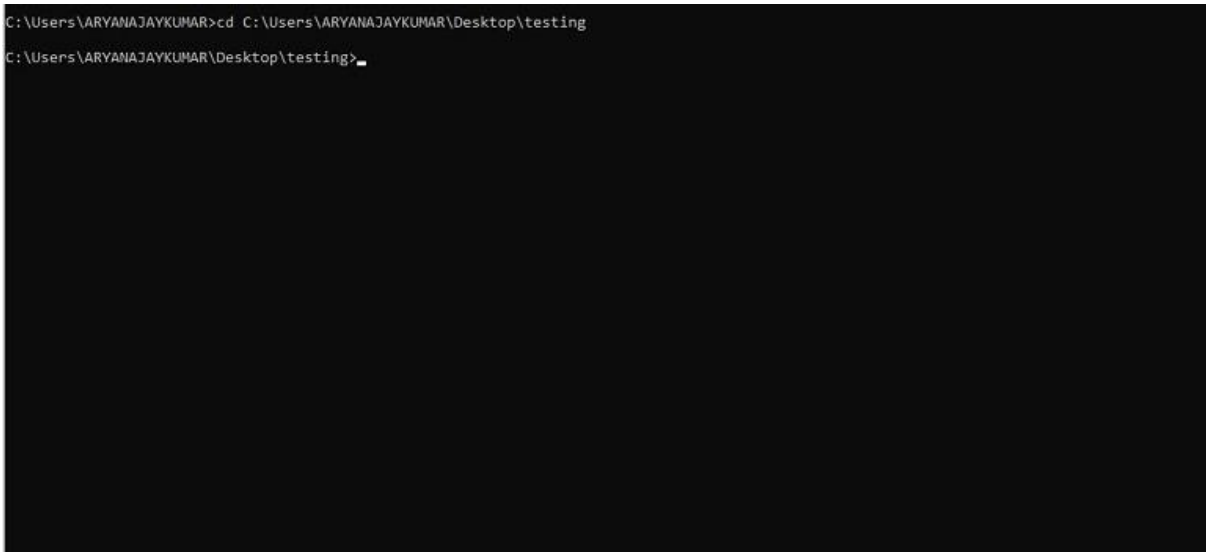


NOTE: The remainder of this guide assumes you are running all Windows commands from an elevated (Administrator) prompt.

Step 3: Navigate to the File to be Signed

- Change into the directory that contains the file you want to sign. This step is necessary if you have not set up the PATH environment variable for signtool.

```
cd <Path to your directory with the file to be signed>
```



Step 4: Run the Signtool Command

- Run the following command to sign your Windows file:

```
signtool sign
  /csp "Encryption Consulting Key Storage Provider"
  /kc  <key name of the private key associated with the certificate>
  /fd  <the desired signing algorithm to be used when signing>
  /f   <the location of the certificate to be used for signing>.pem
  /tr  <the URL of the time stamping server to be used>
  /td  <the algorithm to be used with the time stamping server>
  <the path to the file to be signed>
```

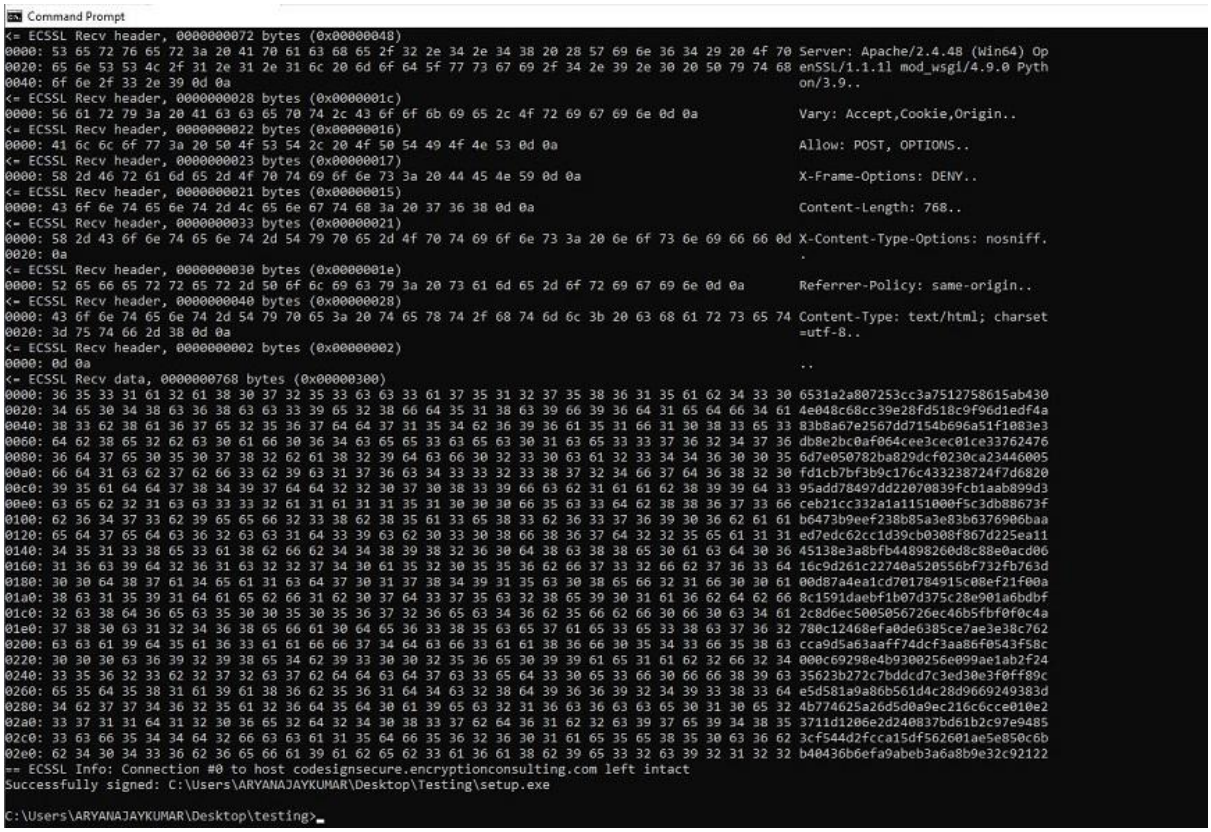
- Here is a working example of the command for your reference, signing setup.exe with a certificate named evcodesigning:

```
signtool sign /csp "Encryption Consulting Key Storage Provider"
  /kc evcodesigning /fd SHA256 /f evcodesigning.pem
  /tr http://timestamp.digicert.com /td SHA256 setup.exe
```

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

Flag	Description
/csp	This specifies the Key Service Provider to be used. This should always be “Encryption Consulting Key Storage Provider”.
/kc	This should be the name of the private key associated with your certificate. This will likely be the same name as the certificate name.
/fd	This is the signing algorithm to be used with the signing function. This can be SHA256, SHA384, or SHA512.
/f	The name of the certificate to be used for signing. This should be the same name as the /kc flag, with a .pem appended to the end. This certificate must be in the same directory this command is run from.
/tr	(Optional) This is the URL of the timestamping authority to be used. If /tr is not in use, and timestamping is not required, then /td should also be left out of this command.
/td	(Optional) This is the signing algorithm to be used with the timestamping server. This can be SHA256, SHA384, or SHA512. This algorithm should be the same as the /fd field. If /tr is not in use, and timestamping is not required, then /td should also be left out of this command.
<file path>	This is the name of the file to be signed. This file must be in the directory this command is being run in.

- On success, signtool reports that the file was successfully signed, along with the timestamp details if a timestamping authority was supplied. Below is an example of a successful output of the command.



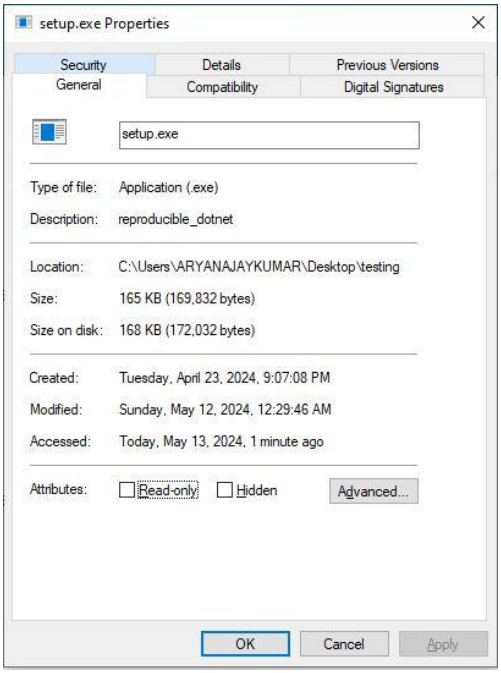
5. **Verify the Signature**

   After the signing operation completes, confirm that a valid digital signature has been applied to the file.

   Steps:

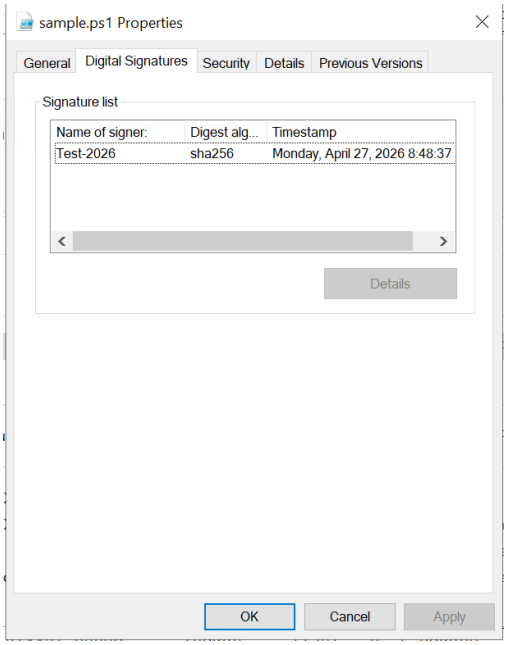
   Step 1: Open the File Properties

   - Right click the signed file and select Properties.

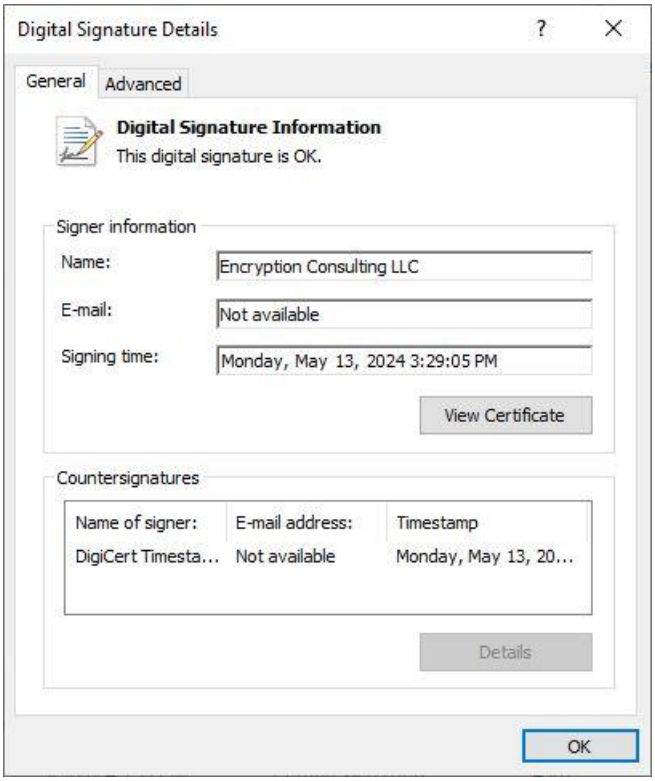


Step 2: Review the Digital Signature

- 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.



- You can also cross-check the operation in the CodeSign Secure portal under Reports > Signing Request Report, where every signing request performed through the KSP is recorded for audit purposes.