ENTRUST

Signing Automation Service

User Guide

December 10, 2021

Contents

1 Service Overview

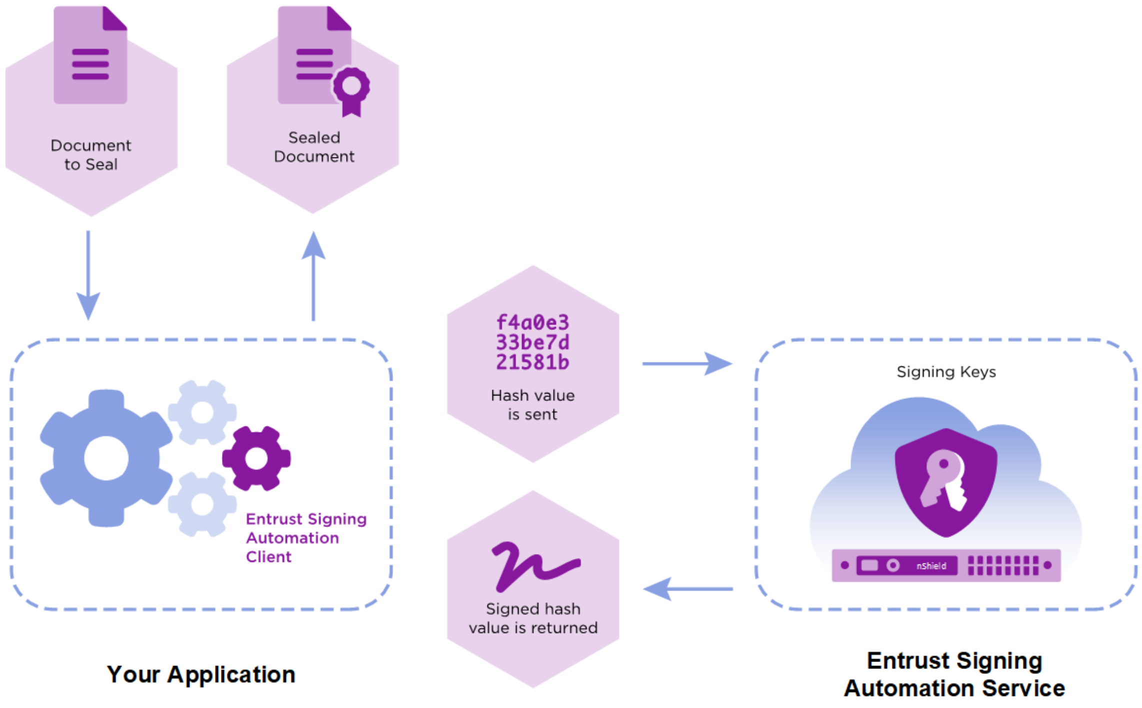
The Entrust Signing Automation Service is a cloud-based hash signing service, with private keys securely generated and protected by the service. When deployed:

1 Using the Entrust Signing Automation Client, your application sends a document hash to the Signing Automation Service.

2 The Signing Automation Service generates the digital signature using the private key of the certificate issued by Entrust and returns the signed document hash.

3 Your application embeds the signature back into the document to create a valid signature.

The Entrust Signing Automation Service includes timestamping and OCSP validation services.



1.1 Entrust Signing Automation Client

The Entrust Signing Automation Client is an on-premise software that leverages the PKCS #11 Cryptographic Token Interface (Cryptoki) by RSA Security Inc. to authenticate into the Signing Automation Service and request cryptographic operations.

1.2 Entrust Signing Automation Certificate

To use the Entrust Signing Automation Service, you also need an Entrust Signing Automation certificate issued by Entrust Certificate Services. The Signing Automation Service generates a CSR (Certificate Signing Request) to apply for the Signing Automation certificate within the

Entrust Certificate Services portal. Once the certificate is issued and installed by a User, it is used with the Signing Automation Service and your signing application to generate document signatures.

1.3 Signing Service users

Four users can consume the Signing Automation Service from the computers configured for that purpose. All service users have the same functionality; however, it will be usual to separate the following roles.

Role	Actions
Administrator	Create signing keys and import certificates in the token, as explained in Creating the signing key and the certificate.
Software application	Select a token's key and sign documents, as explained in the technical integration guides.

1.4 Logical token

The Signing Automation Service uses "logical tokens" as per the "physical" tokens described in the PKCS #11 standard. When your signing application connects to the server, the Signing Automation Client provides a logical token with the following standard operations.

PKCS #11 Mechanism	Supported operations
CKM_ECDSA	Sign and verify with NIST P256, NIST P384, or NIST P521 curves.
CKM_ECDSA_KEY_PAIR_GEN	Generate ECDSA key pairs with NIST P256, NIST P384, or NIST P521 curves.
CKM_RSA_PKCS	Sign and verify with PKCS1.5 padding (and software-based hashing) with 2048, 3072, or 4096 key sizes.
CKM_RSA_PKCS_KEY_PAIR_GEN	Generate RSA key pairs with 2048, 3072, or 4096 key sizes.
CKM_SHA256_RSA_PKCS	Sign and verify with PKCS1.5 and SHA 256 hashing with 2048, 3072, or 4096 key sizes.
CKM_SHA384_RSA_PKCS	Sign and verify with PKCS1.5 and SHA 384 hashing with 2048, 3072, or 4096 key sizes.
CKM_SHA512_RSA_PKCS	Sign and verify with PKCS1.5 and SHA 512 hashing with 2048, 3072, or 4096 key sizes.

1.5 Signing Automation Service & Client licensing

This section defines the licensing terms and permitted uses for the Signing Automation Service, including the Signing Automation Client software.

In this document, when we refer to “you”, we mean the customer who has purchased the Signing Automation Service or one of that customer’s Users. In general, the Signing Automation Service is for the customer’s internal use only. However, the customer is permitted to appoint outside organizations and/or their software applications as Users and distribute the Signing Automation Client software to such Users, solely to enable those Users to use the Signing Automation Service on the customer’s behalf (and not for the outside User’s benefit). The customer is responsible for overseeing and controlling how the Users use the Signing Automation Service.

You may deploy the Entrust Signing Automation Client software on your company’s infrastructure and/or commercial cloud accounts. You are strongly encouraged to keep your deployments up to date with our latest product release.

You may use the Signing Automation Service only to apply for and in connection with an Entrust Signing Automation certificate that identifies you (Entrust’s customer) as the Subject by your organization name (e.g., “ABC Ltd”) or one of your corporate affiliates if that affiliate authorizes you to sign documents on its behalf. You are expressly prohibited from using the Signing Automation Service together with signing certificates that identify as the Subject any person other than yourself or an affiliate that has authorized you to sign documents on its behalf. You are expressly prohibited from using the Signing Automation Service in conjunction with any certificate not issued by Entrust.

For each subscription to the Signing Automation Service, we will provide a license key that determines the number of digital signatures that you may generate using the service, the number of Signing Automation Certificates with which you may use the service, and/or for otherwise enabling or controlling certain functions within the service. You may not copy or alter your license key. You may not circumvent or attempt to circumvent the license key mechanism. You may only use a license key provided by Entrust and only in conjunction with the Signing Automation Service and Signing Automation Client software for which it was delivered. Unless otherwise specified in your Order(s), each Entrust Signing Automation Service license key is capped to one Signing Automation certificate based on RSA 2048-bit key pairs for 10,000 signatures to be consumed within one year. Additional Entrust Signing Automation certificates or signatures can be purchased separately.

1.6 Software and documentation download

Log in to Entrust TrustedCare at https://trustedcare.entrust.com for downloading the Signing Automation Client software, the software updates, and the service documentation.

1.7 Service activation

With the license key, you can activate the Entrust Signing Automation Service and configure both the Signing Service Users and the Signing Automation Clients

Entrust delivers the license key in two separate emails: one with a password-protected file and a second one with the file password.

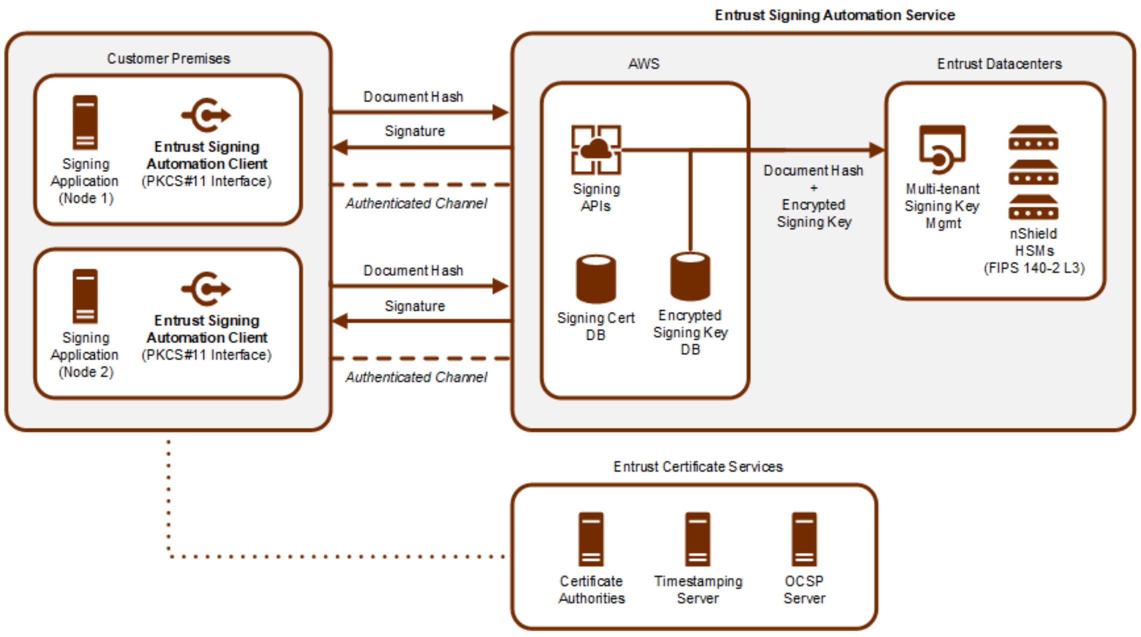
1.8 Third-party licenses

The Entrust Signing Automation Client provides an interface compatible with the PKCS #11 Cryptographic Token Interface (Cryptoki) by RSA Security Inc.

The Entrust Signing Automation Client includes software developed by the OpenSSL Software Foundation.

2 Architecture

The Entrust Signing Automation Service provides a multi-tenant and cloud-based service for document signing. When integrated into a signing workflow, this service comprises the following components.


Entrust Signing Automation Service
Customer Premises
Signing Application (Node 1)
Entrust Signing Automation Client (PKCS#11 Interface)
Signing Application (Node 2)
Entrust Signing Automation Client (PKCS#11 Interface)
Document Hash
Signature
Authenticated Channel
Document Hash
Signature
Authenticated Channel
AWS
Signing APIs
Signing Cert DB
Encrypted Signing Key DB
Document Hash + Encrypted Signing Key
Entrust Datacenters
Multi-tenant Signing Key Mgmt
nShield HSMs (FIPS 140-2 L3)
Entrust Certificate Services
Certificate Authorities
Timestamping Server
OCSP Server

See below for a description of each component.

2.1 Customer premises

The customer premises run the Entrust Signing Automation Client and the signing application. A signing application is a third-party product with signing capabilities — for example:

- iText library
- Oracle JDK
- OpenJDK
- Entrust Java Toolkits

2.2 Amazon Web Services

The AWS cloud hosts the following components.

2.2.1 Encrypted signing key database

The encrypted signing key database stores the wrapped signing keys. Thus, this database provides multi-tenancy by supporting an indefinite number of signing keys for the Signing Application.

NOTE: The HSM keys in the Entrust datacenters protect the wrapped keys. Wrapped keys are never unwrapped or used outside an HSM.

2.2.2 Signing certificate database

The signing certificate database stores the signing certificates. Again, this database provides multi-tenancy by supporting an indefinite number of signing certificates for the Signing Application

2.2.3 Signing APIs

The signing APIs authenticate the Entrust Signing Automation Client and process the request. For example, when processing a document signing request:

1. Queries the signing key database for the client's signing key.
2. Sends the key and the document hash to the Entrust datacenters for signing.
3. Queries the signing certificate database for the client's signing certificate.
4. Returns the signed document hash and the signing certificate to the Entrust Signing Automation Client.

2.3 Entrust datacenters

The cloud-based Entrust datacenters host the following services.

2.3.1 Multi-tenant signing key management

The multi-tenant signing key management selects the nShield HSM that will:

- Generate and wrap the keys when processing key generation requests.
- Unwrap the signing key of the signing key database and sign a document hash when processing a signature request.

2.3.2 nShield HSMs

The nShield HSMs are FIPS 140-2 L3 compliant devices for:

- Generating and wrapping the keys stored in the signing key database.
- Unwrapping the signing keys.

- Signing document hashes with the unwrapped keys.

2.4 Service rate

Entrust Signing Automation Service can deliver a minimum of 10 signatures per second and customer.

3 Signing Automation Client requirements

To install and run the Signing Automation Client, you need the following environment.

3.1 Operating system

See the table below for the supported operating systems.

Platform	Operating System
Windows	Windows 10, version 1909 (64-bit)
Linux	Ubuntu 20.04.x (64-bit)

3.2 Oracle/Open JDK

In the supported operating systems, you can configure the Signing Automation Client into the Java platform for Java applications integrated with the Sun PKCS #11 provider. With this provider, applications using the Java Cryptography Architecture (JCA) and the Java Cryptography Extension (JCE) APIs can access native PKCS #11 tokens/HSMs as our PKCS #11 Signing Automation Client.

- For more information about Sun PKCS #11 in Java 8 LTS, see:

  https://docs.oracle.com/javase/8/docs/technotes/guides/security/p11guide.html

- Starting with Java 9, there were slight changes in the Sun PKCS #11 integration in applications. You can find more details in the latest LTS version documentation, currently Java 11 LTS.

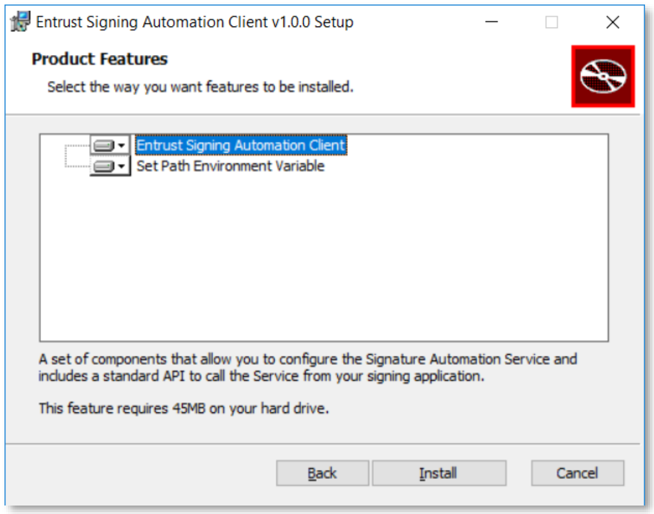
  https://docs.oracle.com/en/java/javase/11/security/pkcs11-reference-guide1.html

4 Installing the Signing Automation Client

Follow the steps described below to install and uninstall the Entrust Signing Automation Client in Windows or Linux.

4.1 Installing the Signing Automation Client in Windows

To install the Signing Automation Client in Windows, run the `SigningClient64.msi` installer as a user with administrator privileges.


Entrust Signing Automation Client v1.0.0 Setup
Product Features
Select the way you want features to be installed.
Entrust Signing Automation Client
Set Path Environment Variable
Back
Install
Cancel

A set of components that allow you to configure the Signature Automation Service and includes a standard API to call the Service from your signing application.

This feature requires 45MB on your hard drive.

The installer will add the application folder to the PATH variable.

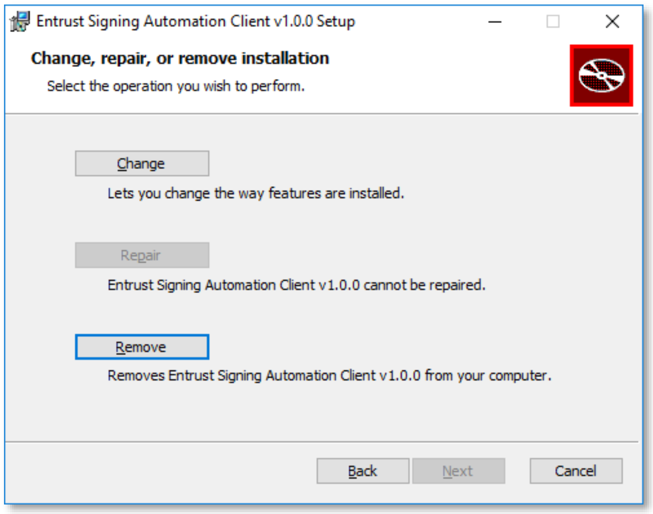
```
C:\Program Files\Entrust\SigningClient
```

See the following table for the files in this folder.

File	Description
P11SigningClient64.dll	The application library for 64-bit Windows systems
SigningClient.exe	The application executable

4.2 Uninstalling the Signing Automation Client in Windows

To uninstall the Signing Automation Client in Windows, run the `EPKCS11KSP.msi` installer, and select **Remove**.


Entrust Signing Automation Client v1.0.0 Setup

Change, repair, or remove installation

Select the operation you wish to perform.

Lets you change the way features are installed.

Entrust Signing Automation Client v1.0.0 cannot be repaired.

Removes Entrust Signing Automation Client v1.0.0 from your computer.

Alternatively, you can also uninstall the application using the Windows **Add or remove programs** dialog

4.3 Installing the Signing Automation Client in Linux

To install the Signing Automation Client in Linux platforms, extract the contents of the distribution TAR file.

4.4 Uninstalling the Signing Automation Client in Linux

To uninstall the Signing Automation Client in Linux platforms, remove the installation files.

5 Installing the License Key

In the machine where the Entrust Signing Automation Client runs, install the license key for each of the four service users.

WARNING: You cannot re-run the license key installation. Contact the Entrust Support if you want to reinstall the license key, for example, with a different set of service users. You can also contact Support if you lose the license key.

To install the license key in a machine

1 If required, set the P11PKIHUB_CONFIG environment variable to modify the location for the user settings. In windows, the default location for these settings is:

```
C:\Users\%USERNAME%\AppData\Roaming\Entrust\SigningClient
```

In Linux:

```
/home/$USERNAME/.signingclient
```

2 Run the following command.

```
signingclient process license <license>
```

Where <license> is the path of the license key file — for example:

```
signingclient process license C:\Users\sysadmin\Documents\license.zip
```

3 When prompted, enter the password you received by email to unzip the license key file.

4 Select the user— for example, type "1" to select the `admin1` user.

```
Choose the user to configure:
  1) admin1
  2) admin2
  3) application1
  4) application2
Select number [1-4]: 1
```

5 When prompted, enter, and confirm the user password.

NOTE: See Installing a new license key to reset a lost password.

6 Select the logical token. By default, the license key will contain a single token — for example:

```
Choose the token to configure:
  1) token1
Select number [1-1]: 1
```

5.1 Credentials file

With the password entered by the administrator, the command-line tool encrypts the user secret in the following file.

WARNING: We strongly recommend backing up this file.

OS	Default path
Windows	C:\Users\%USERNAME%\AppData\Roaming\Entrust\SigningClient\credentials
Linux	$HOME/.signingclient/credentials

5.2 Configuration file

When completing the license key installation, the command-line tool saves the user settings in the following file.

OS	Default path
Windows	C:\Users\%USERNAME%\AppData\Roaming\Entrust\SigningClient\config
Linux	$HOME/.signingclient/config

As explained in the Command-line reference, you can manage these configuration settings with the config list and config set commands. For example, to update the library path:

```
signingclient config set --library /home/john/your_library_path/libp11pkihub.so
```

6 Creating the signing key and the certificate

To perform cryptographic operations with the Signing Automation Client, you must create a signing key and obtain the corresponding certificate.

NOTE: To renew your certificate, generate a new key and issue the corresponding certificate as explained in this section.

To create a signing key with the certificate

1 Log in to your cloud.entrust.net account.

2 Follow the steps described in the Entrust Certificate Services online documentation, section **Creating certificates > Requesting a Document Signing Certificate**, to issue a "Signing Automation" certificate.

   NOTE: When completed, this requesting process automatically installs the key and the certificate in your logical token.

3 Save the PEM encoding of the intermediate CA certificate in a file with the `cer` extension.

```
-----BEGIN CERTIFICATE-----
MIIFOTCCBCGgAwIBAgIMVRYVFQAAAABRzhYOMA0GCSqGSIb3DQEBCwUAMIG0MRQw
EgYDVQQKEwtFbnRydXN0Lm5ldDFAMD4GA1UECxQ3d3d3LmVudHJ1c3QubmV0L0NQ
U18yMDQ4IGluY29ycC4gYnkgcmVmLiAobGltaXRzIGxpYWIuKTElMCMGA1UECxMc
KGMpIDE5OTkgRW50cnVzdC5uZXQgTGltaXRlZDEzMDEGA1UEAxMqRW50cnVzdC5u
ZXQgQ2VydGlmaWNhdGlvbiBBdXRob3JpdHkgKDIwNDgpMB4XDTE2MDIyNTE4MDgx
NloXDTI5MDYyNTE4MzgxNlowgbcxCzAJBgNVBAYTAlVTMRYwFAYDVQQKEw1FbnRy
dXN0LCBJbmMuMSgwJgYDVQQLEx9TZWUgd3d3LmVudHJ1c3QubmV0L2xlZ2FsLXRl
cm1zMTkwNwYDVQQLEzAoYykgMjAxNSBFbnRydXN0LCBJbmMuIC0gZm9yIGF1dGhv
cml6ZWQgdXNlIG9ubHkxKzApBgNVBAMTIkVudHJ1c3QgQ2xhc3MgMyBDbGllbnQg
Q0EgLSBTSEEyNTYwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDGnEvB
T0qd2X3TO1eRq83pdhUtwCAvLDGGxQk9sB+RhJhDlS7UnqraVeLgYOi7B+/Lg+0u
Xxny0CjtOmQ/y64wYCHmZqtYTmJndk5SjNx7mEQODi2QULUh+42xza8hByWXz7oP
GEcZTnHLabj6I20aBhE1wVa6n2Ih8bDxAY9ez/EiosFCDvXNMugrJ/SSbwsVXvz6
aVKwjn6ky3W5RYS1kwMLcitAs25DQqETGRhkRNSmIAlFsDpkD1b95IUojrjUOCPH
LuKw+5r7GjiBkzLnLR+ujjcXzvzCFD993yTsseygqo4jBIEce68pztTn1OFm6W5k
6eEFsiqRmHBY2PILAgMBAAGjggFEMIIBQDAOBgNVHQ8BAf8EBAMCAQYwNAYDVR0l
BC0wKwYIKwYBBQUHAwIGCCsGAQUFBwMEBgorBgEEAYI3CgMMBglghkgBhvprKAsw
OwYDVR0gBDQwMjAwBgRVHSAAMCgwJgYIKwYBBQUHAgEWGmh0dHA6Ly93d3cuZW50
cnVzdC5uZXQvcnBhMBIGA1UdEwEB/wQIMAYBAf8CAQAwMwYIKwYBBQUHAQEEJzAl
MCMGCCsGAQUFBzABhhdodHRwOi8vb2NzcC5lbnRydXN0Lm5ldDAyBgNVHR8EKzAp
MCegJaAjhiFodHRwOi8vY3JsLmVudHJ1c3QubmV0LzIwNDhjYS5jcmwwHQYDVR0O
BBYEFAafb06iKU4PDK4Xv7aYRu+tuDtyMB8GA1UdIwQYMBaAFFXkgdERgL7YibkI
ozH5oSQJFrlwMA0GCSqGSIb3DQEBCwUAA4IBAQB8eBvEzfG7ciGMiBdPtSqio/2d
h+DXHDyC2Z6Vkzd305spuLwA0olAKJKZgKFM804XffTDY4zCTvY3sX9gMvHUk1ut
lt2Kt8KPDfFLrfxL21sNyj79WG99p7vrzVlsO+8AFZU2AdTLPLVjz9/Tmqr5RRKy
q4IPZg0uaAM4+m6VIOceWnYEI2A9S+XpEHWqF9vbCevuF0iLnZalaqPdTBkfYkAu
D/T6AOZabkbolo2bjssLzYsHOZExFCFu37kJZTw/JaDlC7o6A0r0QaZojaXqYM0j
SfppwIWH58keRNVFyBIApO0GmIpBSieh8hZlo1X6K0yukH+M53cikOr4IS/F
-----END CERTIFICATE-----
```

4 Run the following command to import the intermediate CA certificate.

```
signingclient import certificate <cer> --key-label "Entrust Class 3 Client CA
- SHA256"
```

Where `<cer>` is the path of the certificate file.

NOTE: You don't need to import the "Entrust.net Certificate Authority (2048)" root certificate because this certificate is most likely already in your operating system. In case you need it, you can download this certificate from https://www.entrust.com/resources/certificate-solutions/tools/root-certificate-downloads

3 Verify the keys in your token with the following command.

```
signingclient list keys
```

Verify also the certificates in your token.

```
signingclient list certificates
```

Both commands will request the user password selected when Installing the License Key.

7 Generating digital signatures

Once installed, you can integrate the Signing Automation Client to sign documents with different tools. For example, a Java project using a signing library to automate signature generation.

7.1 Integrating signing applications

See the integration guides in the `interoperability` folder for the supported tools. If you are using a signing tool not formally approved by Entrust, it is still good to work if it conforms to the standard PKCS #11 operations.

7.2 Adding timestamps to signatures

Entrust provides a timestamp server at the following URL.

http://timestamp.entrust.net/TSS/RFC3161sha2TS

When adding timestamps to signatures, the date is accurate and backed by a trusted entity. Timestamps also allow creating Long Term Validation (LTV) signatures.

7.3 Checking the revocation status of the signing certificate

Entrust provides an OCSP server at the following URL.

http://ocsp.entrust.net

OCSP is the most efficient way to check the revocation status of your certificates.

8 Sharing certificates between several applications

See the sections below for sharing the signing certificates between different applications.

8.1 Sharing a token between applications on the same node

Applications installed in the same node can share the same token, as explained below.

To share a token between applications on the same node

1. Install a single client, as explained in Installing the Signing Automation Client or Installing the Signing Automation Client in Linux.
2. Install the license key, as explained in Installing the License Key.
3. Generate the key and the certificate, as explained in Creating the signing key and the certificate.
4. In each application, use the same `application` user and password.

8.2 Sharing a token between applications on different nodes

Applications installed on different nodes can share the same token, as explained below.

8.2.1 Generating the token in the main node

Before sharing the token certificates between applications installed on different nodes, you must select a node to install the Signing Application Client and generate a token.

To generate the token in the main node

1. Install the software, as explained in Installing the Signing Automation Client.
2. Install the license key, as explained in Installing the License Key.
3. Generate the key and the certificate, as explained in Creating the signing key and the certificate.
4. Check the access to the signing certificate.

   ```
   signingclient list certificates
   ```

   When prompted, authenticate as the `application` user.

8.2.2 Replicating the token in the other nodes

In the other nodes of the high availability cluster, replicate the token as follows.

To replicate the token in the other nodes

1 Install the software, as explained in Installing the Signing Automation Client or Installing the Signing Automation Client in Linux.

2 Copy the Credentials file and the Configuration file generated when Installing the License Key in the main node.

3 Check the access to the signing certificate.

```
signingclient list certificates
```

When prompted, authenticate as the `application` user.

9 Recovering lost credentials

When losing a user password or key file, follow the steps described in the following sections.

9.1 Getting a new license key

Contact with Entrust to obtain a new license key. As the initial Service activation, you will get the new license key in two separate emails: one with a password-protected file and a second one with the file password.

9.2 Installing a new license key

To install the new license key, run the same command described in Installing the License Key. For example:

```
>signingclient process license dsaas397582_update.zip
Enter the password to decrypt the zip file:
Organization ID: dsaas397582
Choose the user to configure:
  1) application1
Select number [1-1]: 1
User ID: application1
Create a password to protect your credentials
New password:
Confirm password:
Writing credentials to:
C:\Users\johnd\AppData\Roaming\Entrust\SigningClient\credentials
Writing config to: C:\Users\johnd\AppData\Roaming\Entrust\SigningClient\config
Choose the token to configure:
  1) token1
Select number [1-1]: 1
Token ID: token1
Writing config to: C:\Users\johnd\AppData\Roaming\Entrust\SigningClient\config
Process license OK
```

When updating a user's license key, the user selector only displays the name of this user. To display all the user names, add the `--all-available-users` flag. For example:

```
>signingclient process license dsaas397582_update.zip --all-available-users
Enter the password to decrypt the zip file:
Organization ID: dsaas397582
Choose the user to configure:
  1) admin1
  2) admin2
  3) application1
  4) application2
```

10 Debugging with the pkcs11-logger library

The out-of-the-box distribution of the Signing Automation Client includes the `pkcs11-logger` library. In debugging mode, this library logs PKCS #11 operations by sitting between the Signing Application Client and the `P11SigningClient64` library.

1 The Signing Application Client sends the PKCS #11 operation calls to the `pkcs11-logger` library.

2 The `pkcs11-logger` library redirects the calls to the `P11SigningClient64` library.

3 The `pkcs11-logger` library returns the call results to the Signing Application Client.

See the project readme for a sample debugging output.

https://github.com/Pkcs11Interop/pkcs11-logger/blob/master/README.md

To enable the debugging mode, configure the following environment variables.

PKCS11_LOGGER_LIBRARY_PATH

This variable specifies the path to the `P11SigningClient64` library, without the enclosing quotes. Run the version command to get this path.

PKCS11_LOGGER_LOG_FILE_PATH

This variable specifies the path to the log file, without the enclosing quotes.

PKCS11_LOGGER_FLAGS

This variable specifies the sum of requested logging flags. For example, a value of 6 disables logging the process id and the thread id. When this variable is not set, the value defaults to 0.

See the following table for all the supported flags.

Hex	Dec	Flag
0x01	1	Disables logging into the log file
0x02	2	Disables logging of process id.
0x04	4	Disables logging of thread id.
0x08	8	Enables logging of PINs.
0x10	16	Enables logging to the stdout.
0x20	32	Enables logging to the stderr.
0x40	64	Enables reopening of the log file. This feature decreases performance but allows deleting the log file when needed.

11 Command-line reference

The `signingclient` command-line tool executed when Installing the License Key and Creating the signing key and the certificate supports the commands described in this section. See the table below for the options shared by these commands.

Option	Description	Commands
--help, -h	Print the command help.	All
--log <log>	Generate the <log> log file.	All
--password <pwd>	Authenticate in the token with the <pwd> password. When omitting this option, the command prompts for the password.	Key and certificate management
--verbose	Print more information about the errors.	All

11.1 config list

After Installing the License Key, you can list the user settings with the following command.

```
signingclient config list
```

This command prints information like the following.

```
User ID:            application1
Subject ID:         0123456789abcdef0123456789abcdef01234567
Organization ID:    dsaas999999
Token ID:           token1
PKCS #11 library:   P11SigningClient64.dll
Signing server:     https://rawsigner.dev.pkihub.com
IdP server:         https://idp.dev.pkihub.com
Proxy server:       <not set>
Proxy auth:         <not set>
```

11.2 config set

After Installing the License Key, you can modify the user settings with the following command.

```
signingclient config set [options]
```

Where [options] are the options described in the following table.

Option	Description
--library <library>	Select the <library> PKCS #11 library.
--proxy-auth <usr>:<pwd>	Authenticate in the proxy as the <usr> user with the <pwd> password.
--proxy-host <host>:<port>	Select the proxy in the <host> host and the <port> port.

Option	Description
--token	List the available tokens and select the one to store in the user configuration.

To delete the proxy configuration, run:

```
signingclient config set --proxy-host "" --proxy-auth ""
```

11.3 create key

To generate a key pair with the corresponding CSR (Certificate Signing Request):

```
signingclient create key --key-type <key-type> [options]
```

Where the supported values for <key-type> are:

- RSA2048
- RSA3072
- RSA4096
- ECDSAP256
- ECDSAP384
- ECDSAP521

And [options] are the options described in the following table.

Option	Value	Default
--csr-out	The path of the generated CSR.	The command skips the CSR generation
--key-id	The hexadecimal key identifier	The identifier is the public key's SHA1
--key-label	The key label	The label is the key identifier
--password	The token password	The command prompts for the password value.

11.4 credentials

To change the user password, run the following command.

```
signingclient credentials --change-password
```

When prompted, enter the current password and the new password.

```
Password:
Enter the new password to protect your credentials
New password:
Confirm password:
Writing credentials to: /home/user/.signingclient/credentials
```

```
Password changed
```

11.5 delete certificate

To delete a certificate from a token, run the following command.

WARNING: Deleting the certificates obtained in Creating the signing key and the certificate makes the Signing Automation Service unusable.

```
signingclient delete certificate --cert-id <cert_id> [options]
```

Where <cert_id> is the certificate identifier, and [options] are the options described in the following table.

Option	Description
--force	Do not prompt for confirmation before deleting the certificate.

11.6 delete key

To delete a key from a token, run the following command.

WARNING: Deleting the key obtained in Creating the signing key and the certificate makes the Signing Automation Service unusable.

```
signingclient delete key --key-id <key_id> [options]
```

Where <key_id> is the key identifier, and [options] are the options described in the following table.

Option	Description
--force	Do not prompt for confirmation before deleting the key.

11.7 import certificate

To import a certificate:

```
signingclient import certificate <cert_file> [options]
```

Where <cert_file> is the certificate file's path, and [options] the options described in the following table.

Option	Value	Default
--cert-id	The hexadecimal identifier of the certificate.	The identifier is: the public key identifier if the certificate public key is in the token; the public key's SHA1 otherwise.

Option	Value	Default
--cert-label	The certificate label.	The certificate label is the certificate subject.
--no-trusted	Set CKA_TRUSTED to `false` in the certificate flags. **WARNING**: Setting this value to `false` can prevent Java signing applications from working.	CKA_TRUSTED is `true`.

11.8 list certificates

To list the certificates imported in the token, run the following command.

```
signingclient list certificates
```

11.9 list keys

To list the keys in the token, run the following command.

```
signingclient list keys
```

11.10 version

To check the version of the Entrust Signing Automation Client, run the following command.

```
signingclient version
```

This command will print information like the following.

```
CLI Version:         v202103090152
Library:             C:\Program Files\Entrust\SigningClient\P11SigningClient64.dll
Library Version:     1.0
Library Description: Signing Client v202102181732
```

11.11 process license

When Installing the License Key or Installing a new license key, you must import the license key with the following command.

```
signingclient process license <license> [options]
```

Where <license> is the path of the license key compressed file, and [options] are the options described in the following table.

Option	Description
--force	Overwrite the existing credentials file (if any). As explained in Installing the License Key, you can use the P11PKIHUB_CONFIG environment variable to select the path of this file.
--all-available-users	Display all user names in the user selector when Installing a new license key. When first Installing the License Key, the user selector displays all the user names, with or without the flag.