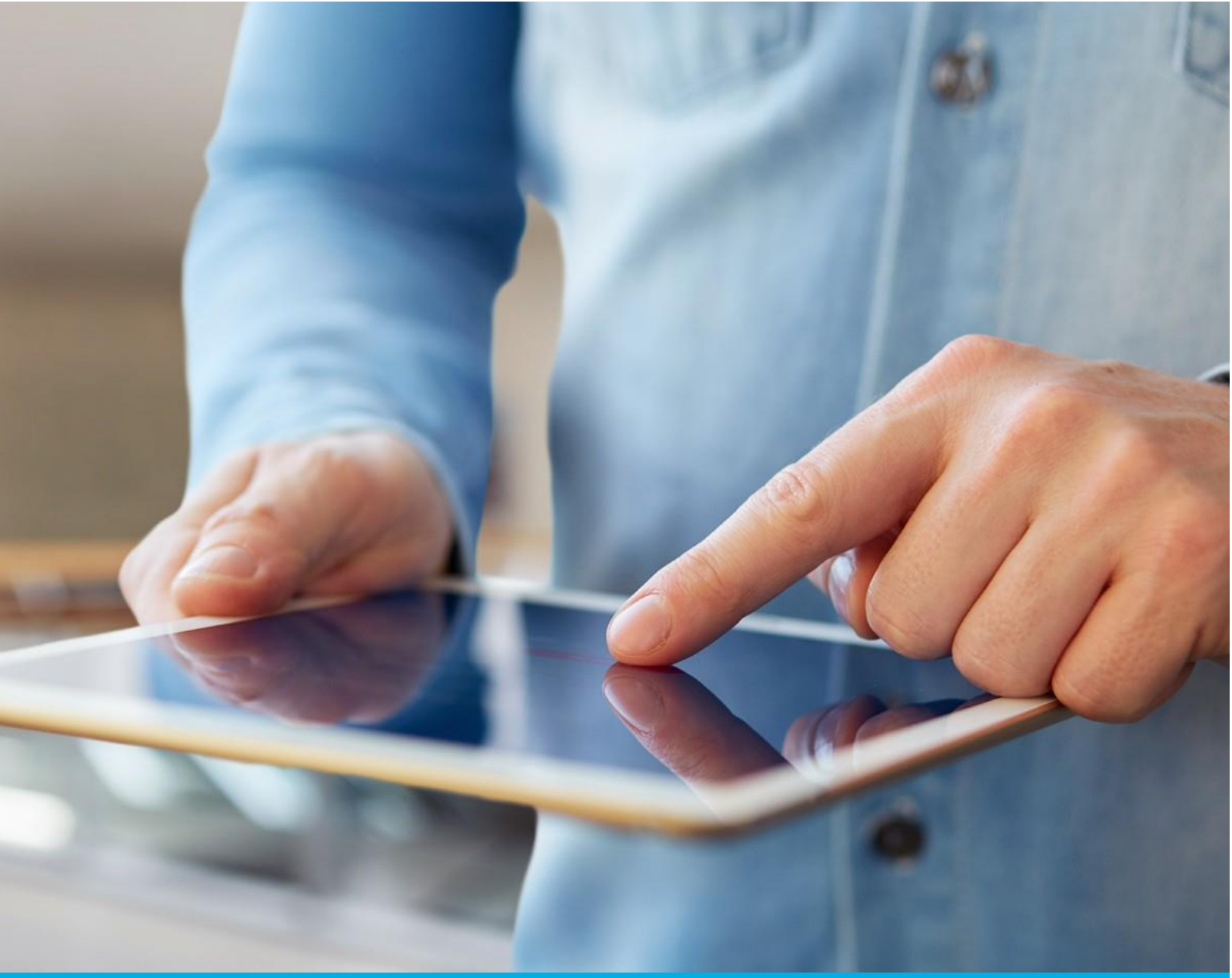


Code Signing in the cloud – Signing in Signtool and Jarsigner

Ver. 1.1

asseco

Certum by asseco

Table of contents

certum.pl
infolinia@certum.pl

1. Product description

The Code Signing certificate allows you to digitally sign applications and drivers, certifying their authenticity and security. Thanks to this, users of your software can be sure that it has not been modified, infected or damaged by third parties.

Signing the application with Code Signing eliminates the problem of code anonymity on the internet. With a digital signature you can be sure that users will not see an "unknown publisher" warning when installing or running your program and they will be ensured about its security. Signing your app helps protect both: your users and your brand's reputation.

Digital code signing makes using the application safe, which translates into greater trust in your brand and an expansion of your group of users.

2. Signing in a Windows environment

Checking the access to the service and display of certificates

After regaining the access to the service, on a portable device, the so-called token allowing for logging into the SimplySign account is generated in the **SimplySign** application.

Figure 1: The SimplySign application - the generated token

In the Windows environment, with the use of **SimplySign Desktop**, you can check the correctness of token generation and the content of the SimplySign account.

In order to install the **SimplySign Desktop** application for Windows, go to the website:

https://support.certum.eu/en/cert-offer-software-and-libraries/

Download the correct package and install the application from the website.

After installing the application, turn it on - the application's icon will appear in the so-called tray - next to the system clock.

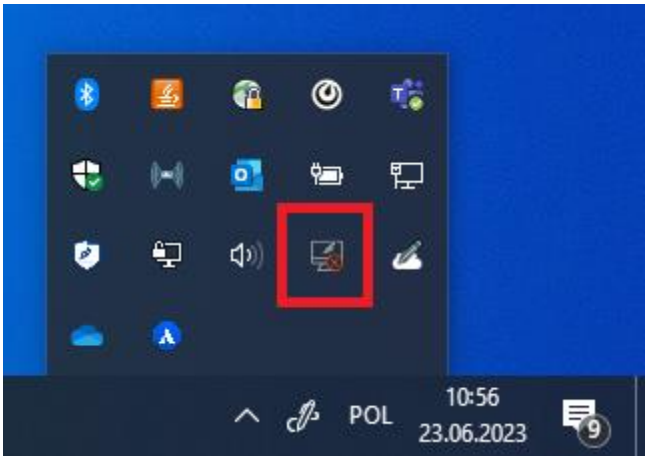


Figure 2: The SimplySign Desktop application - the icon

Then, right-click the application icon - a menu will appear.

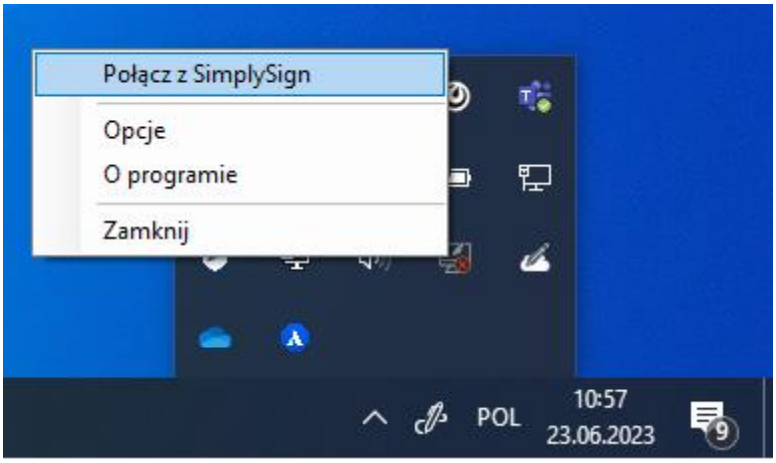


Figure 3: The SimplySign Desktop Application - the menu

certum.pl
infolinia@certum.pl

Select the **Connect to SimplySign command.** A window for logging into the service will appear.

Figure 4: The SimplySign Desktop Application - logging into the service

Enter the Username and the token generated on the mobile device and press the **Ok** button. If correct data are entered, you will be logged into the service - a relevant notification with information on the number of cards and certificates will be displayed.

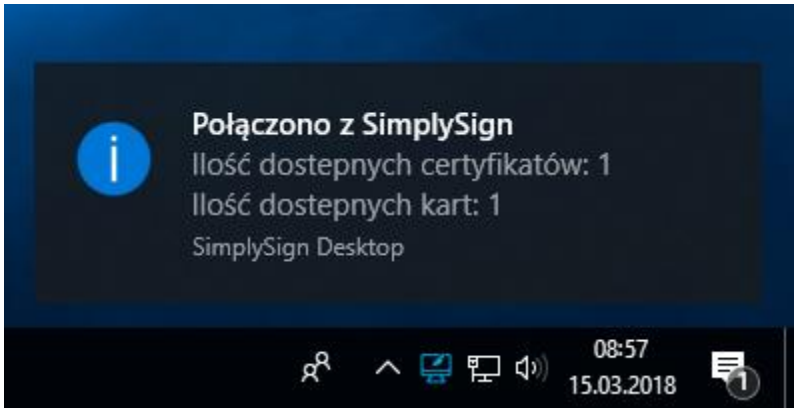


Figure 5: The SimplySign Desktop Application - information after logging into the service

certum.pl
infolinia@certum.pl

After logging in, view the list of certificates. For this purpose, right-click on the **SimplySign Desktop** application icon and select **Manage certificates → Certificate list** from the menu.

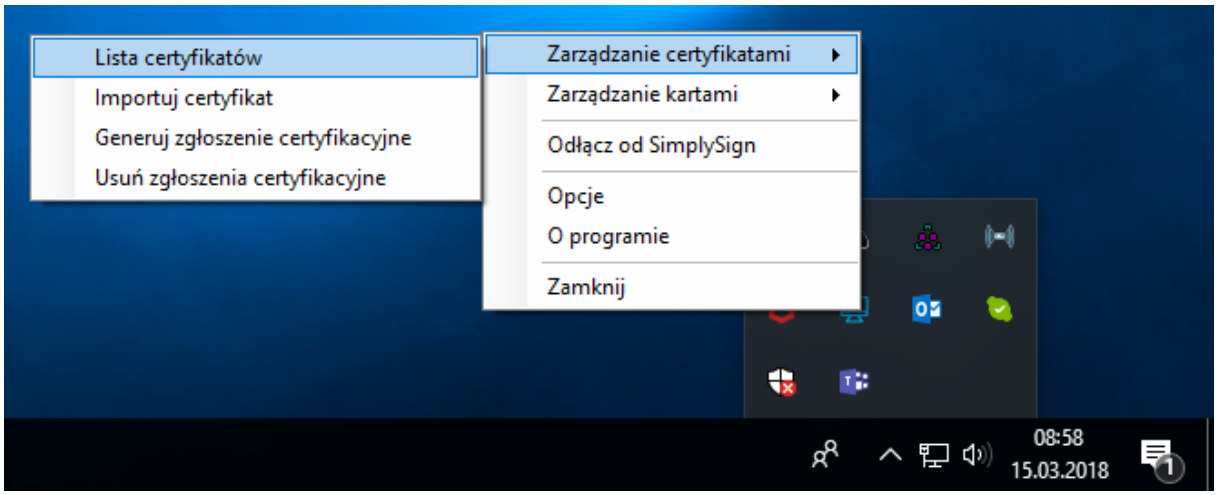


Figure 6: The SimplySign Desktop Application – the menu allowing for displaying the certificate list

A list of certificates will be displayed.

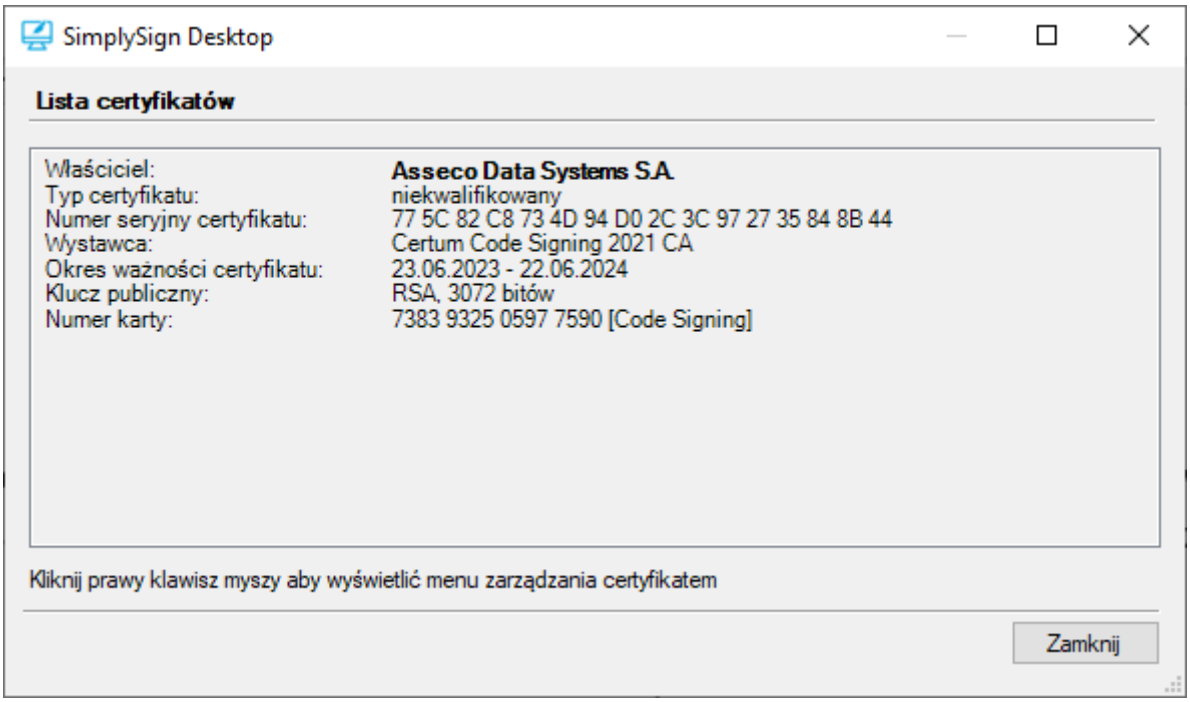


Figure 7: The SimplySign Desktop Application – the list of certificates

In order to display a certificate you need to double click within its field - this will display the details of this certificate.

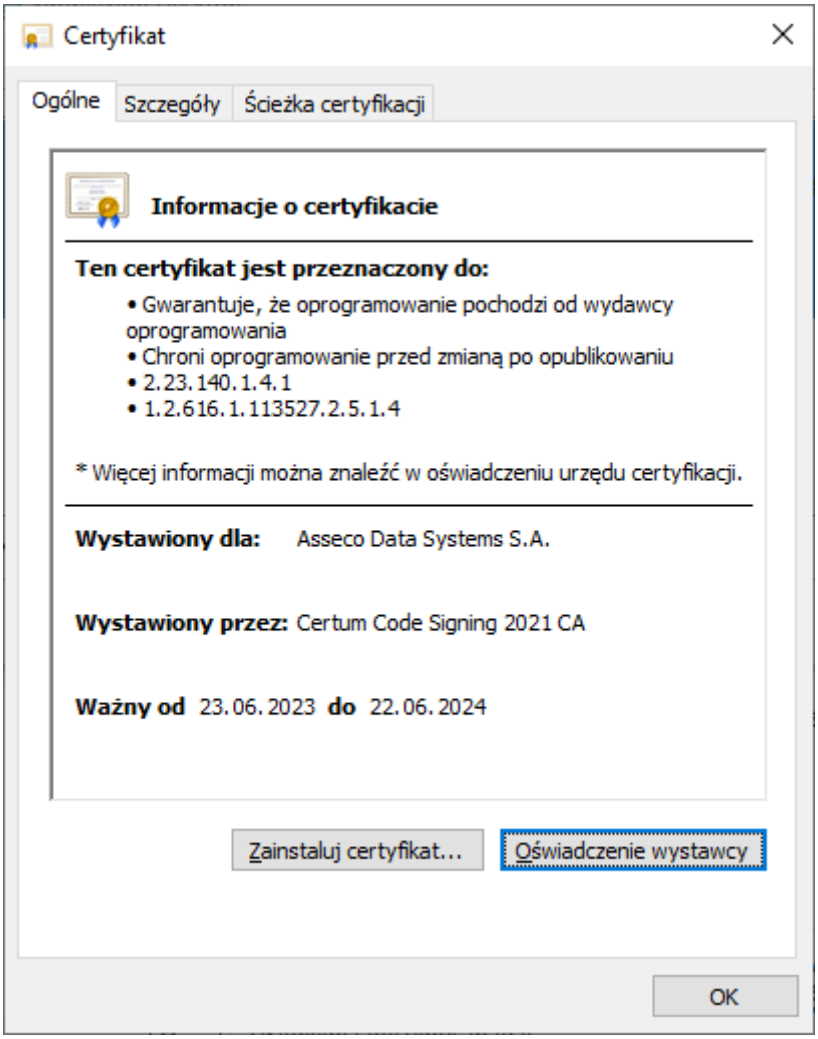


Figure 8: The SimplySign Desktop Application – the list of certificates

Signing with signtool

An individual signature

In order to make a signature with the use of signtool, it is necessary to set the so-called "**thumbprint from the certificate**". For this purpose, display the certificate and go to **Details** tab and then go to **Thumbprint** field.

certum.pl
infolinia@certum.pl

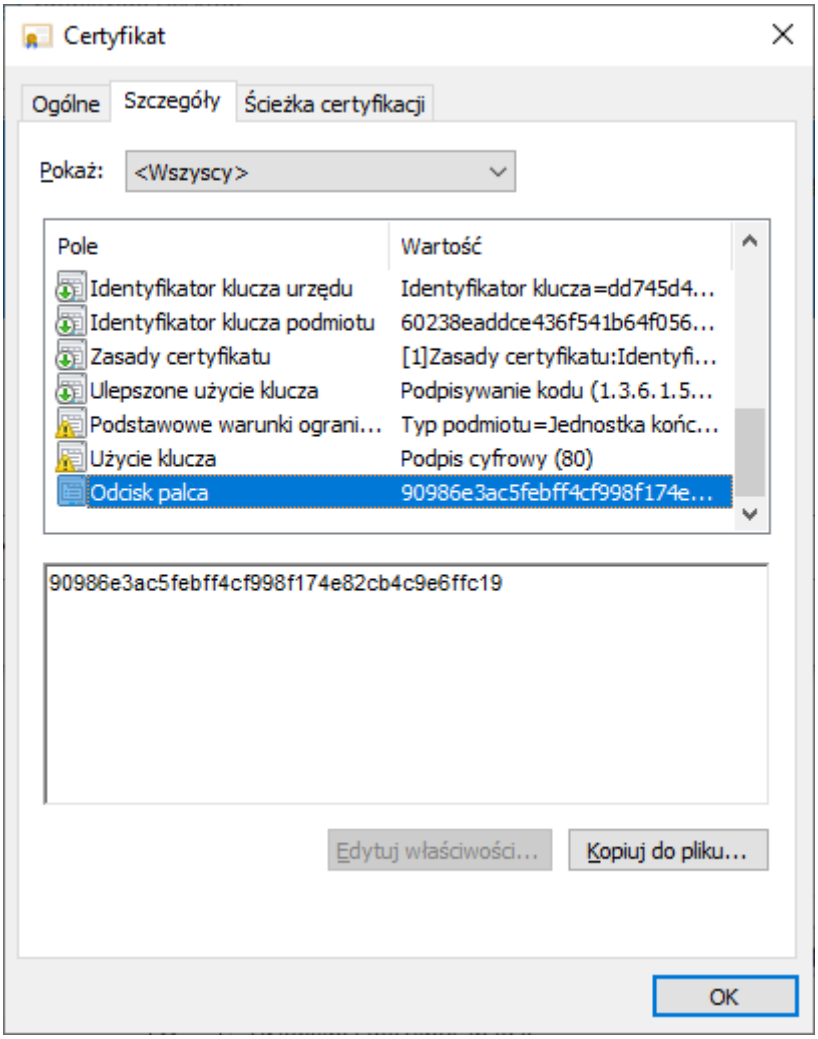


Figure 9: Certificate details - thumbprint value

After obtaining a thumbprint, you can prepare a command allowing for signing a file. The command's syntax is as follows:

signtool sign /sha1 "[1]" /tr [2] /td [3] /fd [4]/v "[5]"

[1] – the so-called thumbprint of the certificate – in the following example it is a value of 90986e3ac5febff4cf998f174e82cb4c9e6ffc19

[2] – time stamp address – in the example below it is a value of http://time.certum.pl

[3] – an abbreviation which will be used for time stamp – in the following example SHA-256

[4] – an abbreviation which will be used for signature – in the following example SHA-256

[5] – a path to the file which is to be signed;

An exemplary command:

signtool sign /sha1 "90986e3ac5febff4cf998f174e82cb4c9e6ffc19" /tr http://time.certum.pl /td sha256 /fd sha256 /v "plik.exe"

In case of pin cards, a window where you have to enter PIN code to SimplySign card where the indicated certificate is located will appear after giving the first command.

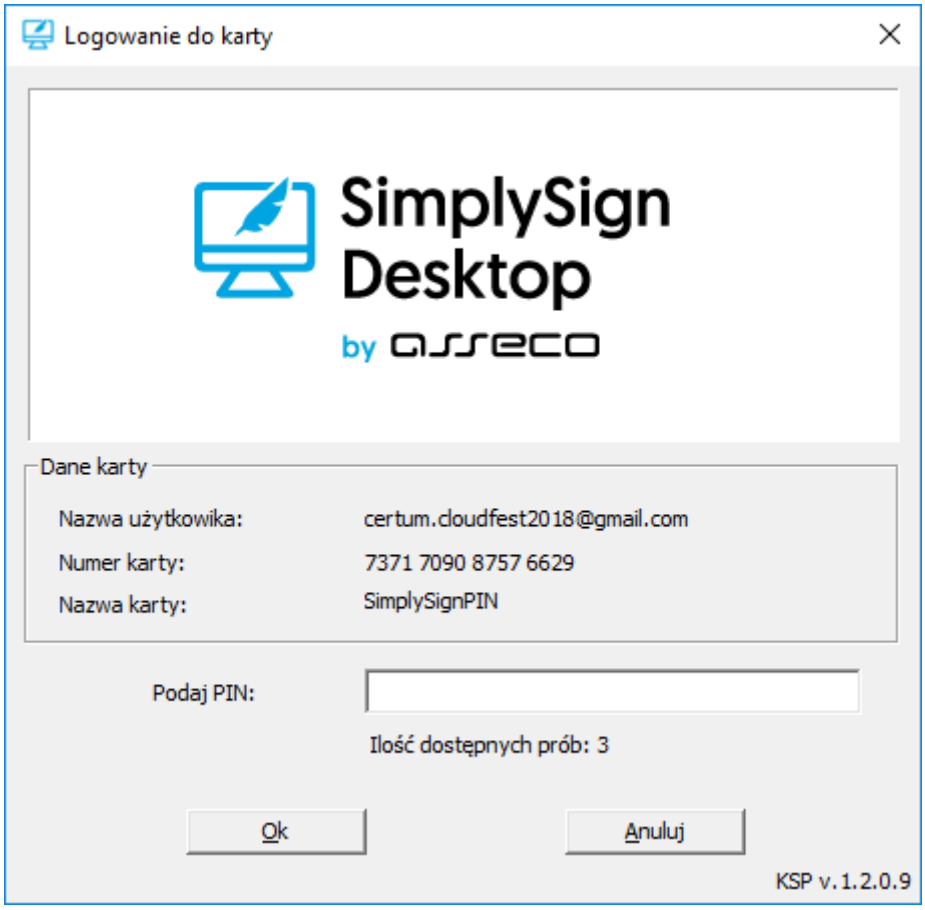


Figure 10: SimplySign Desktop Application - entering the PIN code to the card

In case of pinless cards, signature of the file will be performed immediately without entering PIN code.

In both cases, the following information will be displayed:

```
The following certificate was selected:
    Issued to: Asseco Data Systems S.A.
    Issued by: Certum Code Signing 2021 CA
    Expires:   Sat Jun 22 09:47:15 2024
    SHA1 hash: 90986E3AC5FEBFF4CF998F174E82CB4C9E6FFC19
```

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

Number of files successfully Signed: 1
Number of warnings: 0
Number of errors: 0
```

Batch signing

In order to perform a batch signature of many files during one session, enter the files which are to be signed in the command of signature for the attribute **/v**. Such an action eliminates the necessity to run the command in the console each time and enter the PIN code when signing subsequent files.

An exemplary command:

signtool sign /sha1 "90986e3ac5febff4cf998f174e82cb4c9e6ffc19" /tr http://time.certum.pl /td sha256 /fd sha256 /v "aplikacja1.exe" "aplikacja2.exe" "aplikacja3.exe"

As a result, the cmd.exe console indicates the correctness of the file signature:

```
The following certificate was selected:
    Issued to: Asseco Data Systems S.A.
    Issued by: Certum Code Signing 2021 CA
    Expires:   Sat Jun 22 09:47:15 2024
    SHA1 hash: 90986E3AC5FEBFF4CF998F174E82CB4C9E6FFC19

Done Adding Additional Store
Successfully signed: aplikacja1.exe
Successfully signed: aplikacja2.exe
Successfully signed: aplikacja3.exe

Number of files successfully Signed: 3
Number of warnings: 0
Number of errors: 0
```

Dual signature

In order to make a dual signature (using both algorithms: SHA-1 and SHA-2, you have to perform the following procedure:

1. Perform the signature of the application with the use of SHA-1 with the exemplary command:

signtool sign /sha1 "90986e3ac5febff4cf998f174e82cb4c9e6ffc19" /tr http://time.certum.pl/ /td sha256 /fd sha1 /v aplikacja.exe

2. Then, perform a signature of the same application with the use of SHA-2 algorithm and the switch **/as**:

signtool sign /sha1 "90986e3ac5febff4cf998f174e82cb4c9e6ffc19" /tr http://time.certum.pl/ /td sha256 /fd sha256 /as /v aplikacja.exe

The result of verification of the file signed dually should be the following message from the console:

```
File: aplikacja.exe
Index  Algorithm  Timestamp
========================================
0      sha1       RFC3161
1      sha256     RFC3161

Successfully verified: aplikacja.exe
```

Windows 8 or higher is required to perform and verify the dual signature. In order to perform or verify the dual signature on Windows 7 systems, please read the article published by Microsoft: https://technet.microsoft.com/en-us/library/security/2949927.

Signature verification with signtool

A signature made using signtool can be verified with the use of the same tool. A syntax of such a command is as follows:

signtool verify /pa /all [1]

[1] – name of the file being verified - in the example file.exe

An exemplary command:

signtool verify /pa /all plik.exe

After running the exemplary command, the following information will be visible on the console:

```
File: plik.exe
Index  Algorithm  Timestamp
========================================
0      sha256     RFC3161

Successfully verified: plik.exe
```

Signing with jarsigner tool

Before starting using the jarsigner tool, an additional configuration is required.

Creation of a configuration file provider.cfg

As the first step, create a provider configuration file for PKCS#11. For this purpose, create a new file with extension *.cfg (an example: provider.cfg). Its content is as follows:

name=[1]
library=[2]
slotListIndex=[3]

[1] – Provider name. Preferably SimplySignPKCS.

[2] – Path to PKCS library. Default path: C:\Windows\System32\crypto3PKCS.dll

[3] – The number of the slot in which the card is located. The first slot number is 0, the second 1, etc. In case when there is one card on the **SimplySign** account, set to 0. In case if there are more cards on the **SimplySign** account, then the slot numbers correspond to the list of cards presented by the **SimplySign Desktop** application - the slot number of a card located at the "highest" place is 0. The next one below has a slot with a number 1, etc.

Attention!!!

Due to the possibility of adding and removing cards from the SimplySign account, which influences the sequence of slots, it is recommended to verify the correctness of the slot number each time prior to signature.

An exemplary configuration:

name=SimplySignPKCS.dll
library=C:\Windows\System32\SimplySignPKCS.dll
slotListIndex=0

Creation of a file of the certificate path bundle.pem

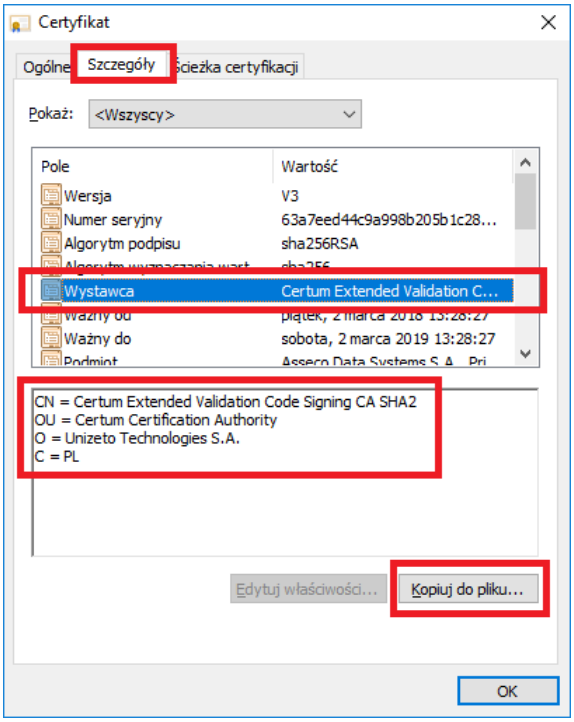
The next step is to create a file of the certificate path with the extension *.pem (an example: bundle.pem). Its content is as follows:

1. "On the top": The user's certificate
2. "Below": an intermediate certificate for the user's certificate

NOTE. The content of bundle.pem file must be in the above mentioned order.

Obtaining the User's certificate

To obtain the user's certificate, you have to simply view it and go to the **Details** tab.



At this step, it is worth to save the content of the "Issuer" field. It will be later helpful in the selection of the intermediate certificate.

Figure 11: Certificate details

Then press the **Copy to File** button. A certificate export wizard will be launched.

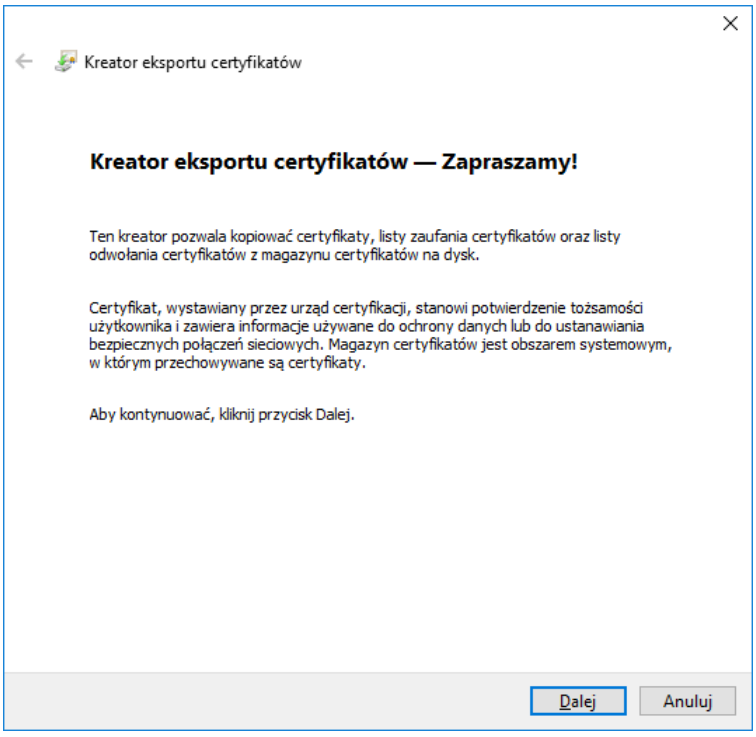


Figure 12: The Certificate Export Wizard

Then, press the **Next** button. A window allowing for the selection of the format in which the certificate is to be exported will be displayed.

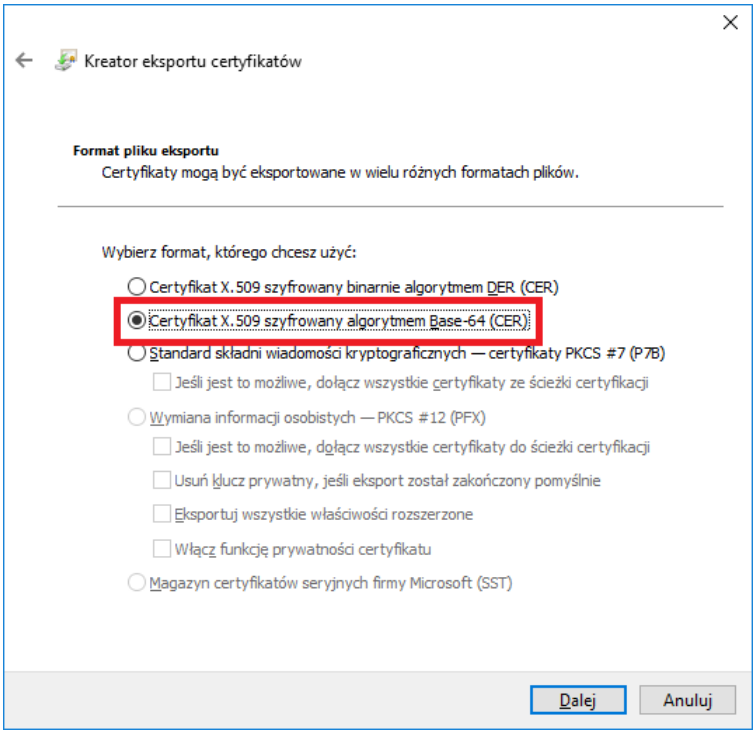


Figure 13: The Certificate Export Wizard - selection of the certificate format

Select the Base-64 format and press the **Next** button. A window allowing to define the location of the exported certificate file will be displayed.

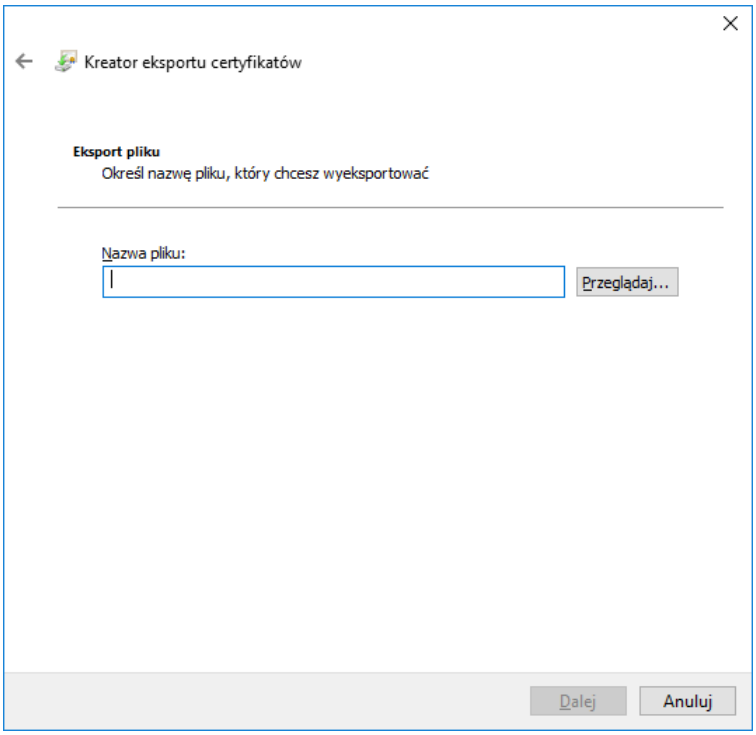


Figure 14: The Certificate Export Wizard - specifying the path

Press the **Browse** button. A window allowing for setting a name of the exported file of the certificate will be displayed.

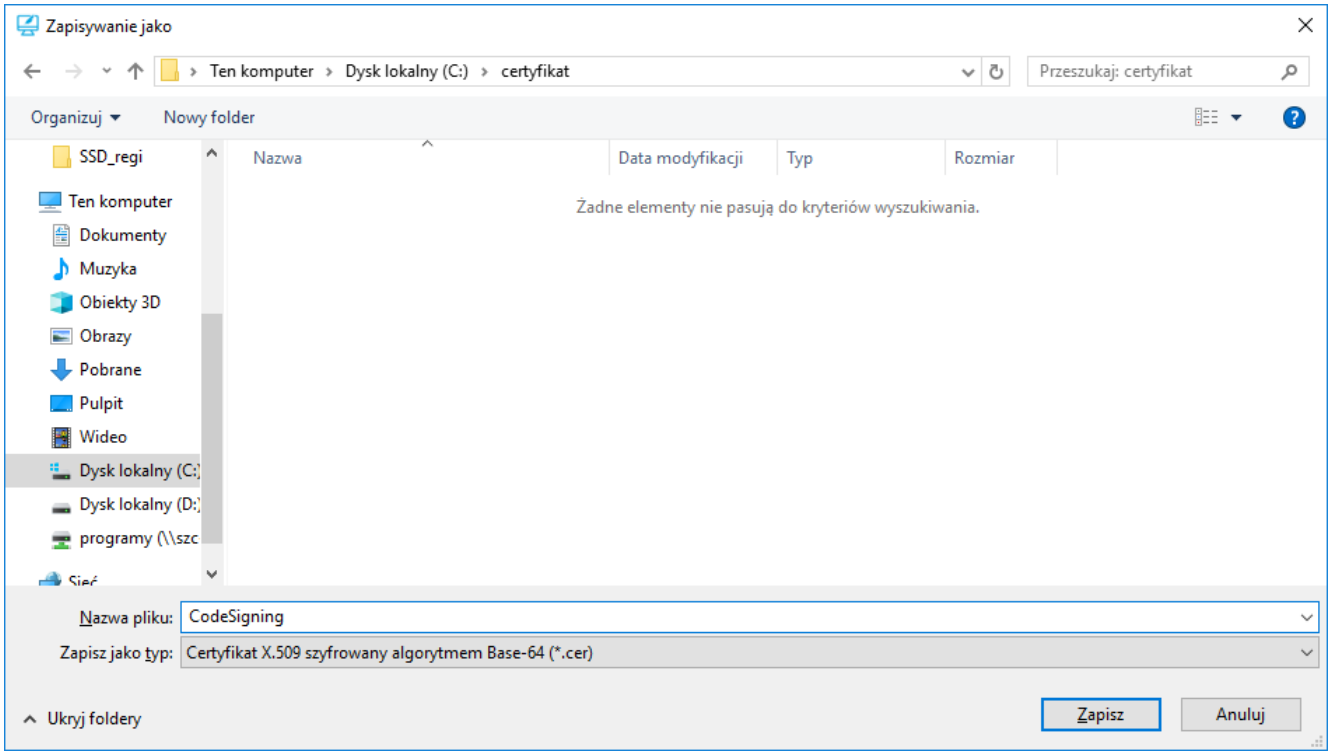


Figure 15: The Certificate Export Wizard - specifying the file name

After specifying the destination folder and defining the file name, press the **Save** button. You will then return to the export wizard. The indicated path will be visible in the wizard.

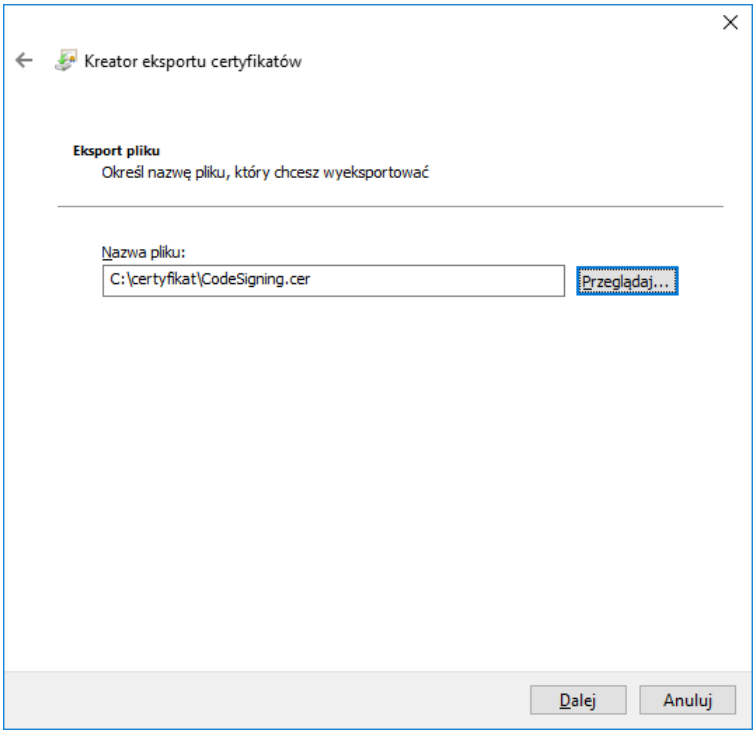


Figure 16: The Certificate Export Wizard – defined location of the certificate

Press the **Next** button. The export wizard window will be displayed.

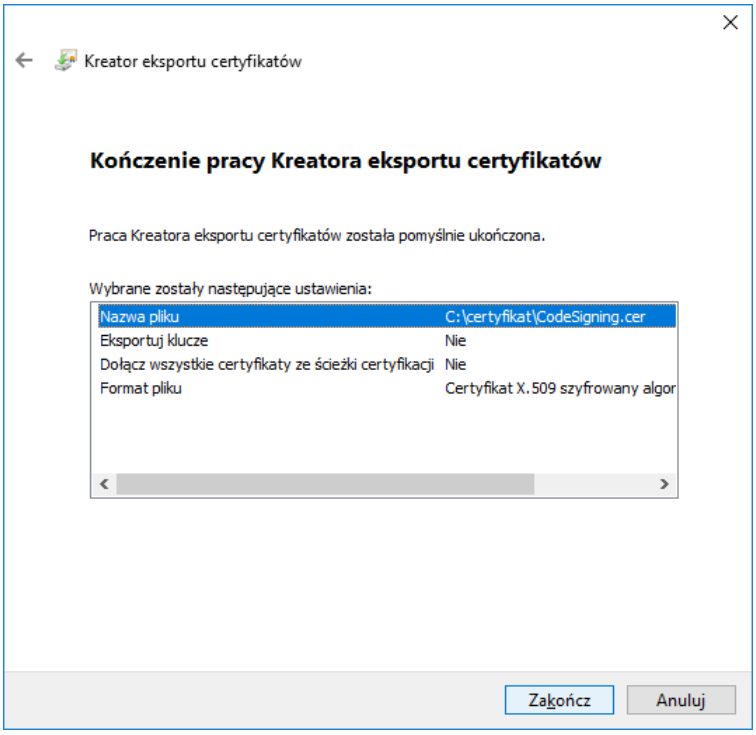


Figure 17: The Certificate Export Wizard - the final window

Press the **Finish** button. The certificate will be exported to a file and an appropriate message will be displayed.

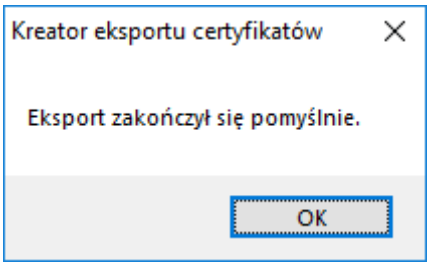


Figure 18: The Certificate Export Wizard – information on the correct export of a file

Obtaining the intermediate certificate

Intermediate certificates should be downloaded from the Certum website:

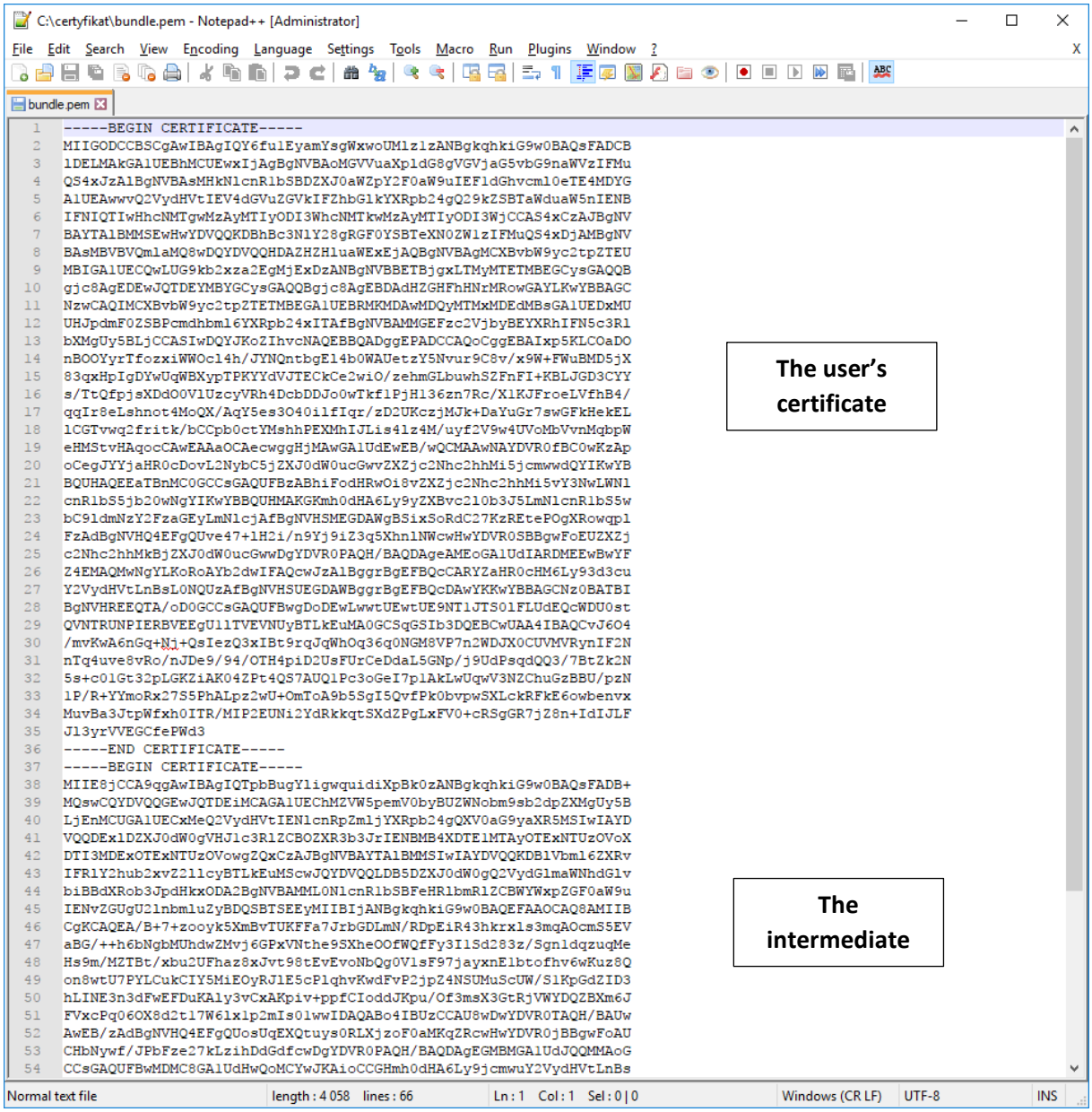
https://www.certum.pl/pl/wsparcie/cert_wiedza_zaswiadczenia_klucze_certum/

The selection of the appropriate intermediate certificate(s) will be facilitated by the previously saved Issuer name from the "Issuer" field of the user's certificate. Find on the Certum website the issuer of your certificate and save its certificate in the PEM text format.

Then, if you have two files with certificates, create a new text file. The content of the both previously obtained files (the user's certificate and the intermediate certificate) should be pasted to one text file in the above mentioned order:

1. "On the top": The user's certificate
2. "Below": an intermediate certificate for the user's certificate

The file has to be saved and its extension changed to *.pem.



The user's certificate

The intermediate

Figure 19: bundle file

Obtaining the certificate alias

Prior to signing, you have to obtain the so-called certificate alias. For this purpose, the following command should be used:

keytool -list -keystore NONE -storetype PKCS11 –providerclass sun.security.pkcs11.SunPKCS11 -providerArg provider.cfg

As a result, the instruction provides the content of the key store:

certum.pl
infolinia@certum.pl

```
Picked up _JAVA_OPTIONS:  -Xms256m -Xmx1024m
Enter keystore password:

Keystore type: PKCS11
Keystore provider: SunPKCS11-SimplySignPKCS

Your keystore contains 1 entry

63A7EED44C9A998B205B1C2850C973D7, PrivateKeyEntry,
Certificate                    fingerprint                    (SHA1):
F5:91:5E:3F:D2:00:F7:BA:57:43:F9:8A:E8:CE:09:A9:83:2F:A9:F7
```

In this case, the alias is:

```
63A7EED44C9A998B205B1C2850C973D7
```

Signing

To sign a file, use the following command in the command line (cmd.exe):

jarsigner -keystore NONE -tsa "[1]" -certchain "[2]" –sigalg [3] -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11 -providerArg "[4]" -storepass "[5]" "[6]" "[7]"

[1] – Time stamp address. For Certum http://time.certum.pl,
[2] – Path to the certificate path file [bundle.pem],
[3] – indication of the signature algorithm [SHA1withRSA or SHA256withRSA],
[4] – Path to the provider configuration file,
[5] – PIN code to the virtual card [for pinless cards enter any PIN code - it cannot be skipped in the command],
[6] – Path to the file being signed,
[7] – Alias of the certificate in which the file will be signed.

An exemplary correct command:

jarsigner -keystore NONE -certchain "bundle.pem" -sigalg SHA256withRSA -tsa "http://time.certum.pl" -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11 -providerArg "provider.cfg" -storepass "12341234" "plik.jar" "63A7EED44C9A998B205B1C2850C973D7"

If the signing procedure is completed successfully, the console will display the following result:

```
Picked up _JAVA_OPTIONS: -Xms256m -Xmx1024m
jar signed.
```

Batch signing

In order to perform a batch signature of many files during one session, create a *.bat file, containing the number of entires equal to the number of files to be signed during one signing process. Such an action eliminates the necessity to run the command in the console each time and enter the PIN code when signing subsequent files.

In order to create a file, create a new *.txt text file, paste the entries for file signature, save the file and change its extension from *.txt to *.bat.

The following example presents the *.bat file content for signature of three applications at the same time:

```
jarsigner -keystore NONE -certchain "bundle.pem" -tsa "http://time.certum.pl" -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11 -providerArg "provider.cfg" -storepass "12341234" "aplikacja1.jar" "63A7EED44C9A998B205B1C2850C973D7
jarsigner -keystore NONE -certchain "bundle.pem" -tsa "http://time.certum.pl" -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11 -providerArg "provider.cfg" -storepass "12341234" "aplikacja2.jar" "63A7EED44C9A998B205B1C2850C973D7
jarsigner -keystore NONE -certchain "bundle.pem" -tsa "http://time.certum.pl" -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11 -providerArg "provider.cfg" -storepass "12341234" "aplikacja3.jar" "63A7EED44C9A998B205B1C2850C973D7"
```

A file saved in such a manner can be opened in the console cmd.exe or by double click, and as a result, signing of the next files contained in the *.bat file will be started.

The result of running the *.bat file in the console will be the information on the subsequent call up of commands and signature of files:

```
C:\Users\user\Desktop\jarsigner>jarsigner -keystore NONE -certchain "bundle.pem" -tsa http://time.certum.pl -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11 -providerArg "provider.cfg" -storepass "12341234" "aplikacja1.jar" "63A7EED44C9A998B205B1C2850C973D7"
Picked up _JAVA_OPTIONS: -Xms256m -Xmx1024m
jar signed.

C:\Users\user\Desktop\jarsigner>jarsigner -keystore NONE -certchain "bundle.pem" -tsa http://time.certum.pl -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11 -providerArg "provider.cfg" -storepass "12341234" "aplikacja2.jar" "63A7EED44C9A998B205B1C2850C973D7"
Picked up _JAVA_OPTIONS: -Xms256m -Xmx1024m
jar signed.
```


certum.pl
infolinia@certum.pl

```
C:\Users\user\Desktop\jarsigner>jarsigner -keystore NONE -certchain
"bundle.pem"  -tsa  http://time.certum.pl  -storetype  PKCS11  -
providerClass       sun.security.pkcs11.SunPKCS11       -providerArg
"provider.cfg"      -storepass      "12341234"      "aplikacja3.jar"
"63A7EED44C9A998B205B1C2850C973D7"
Picked up _JAVA_OPTIONS: -Xms256m -Xmx1024m
jar signed.
```

File verification with jarsigner tool

Verification of the signed file with the use of the jarsigner tool is performed with the following command:

jarsigner -verify "[1]"

[1] – Path to the file being signed,

An exemplary correct command:

jarsigner -verify "plik.jar"

In case of correct verification of the file, the console will display:

```
Picked up _JAVA_OPTIONS: -Xms256m -Xmx1024m
jar verified.
```

In case of the lack of the signature, the result is as follows:

```
Picked up _JAVA_OPTIONS: -Xms256m -Xmx1024m
jar is unsigned.
```


certum.pl
infolinia@certum.pl