



Instruction – Certum Code Signing - Using Signtool and Jarsigner

Tools for Certum Code Signing

version 2.3



Spis treści

1.	Product description	2
2.	Signtool.....	3
2.1	Tool description	3
2.2	Signing	3
2.3	Verification.....	4
2.4	Batch signing	5
2.5	Dual signature	6
3	Jarsigner.....	6
3.1	Tool description	6
3.2	Configuration	6
3.2.1	Create a configuration file provider.cfg	6
3.2.2	Create certificate path file bundle.pem	7
3.2.3	Change the user alias (label) on a card (only for users with diacritical marks in the Common Name (CN) field in a certificate)	11
3.3	Signing	16
3.4	Verification.....	16
3.5	Batch signing	17
4.	Most common problems	18

1. Product description

Certum Code Signing certificates protect software against unauthorized modification or tampering by third parties. It is often the case that software downloaded from the web is treated as malware by the computer. This is due to the fact that it does not have a certificate issued by an authorised certification authority such as Certum.

By securing your software with **Certum Code Signing** certificates, you can protect your software from unauthorized access and theft and minimize the risk of warning messages from SmartScreen ® Application Reputation.

Software secured with **Certum Code Signing** certificates will result in increased security and customer trust and therefore more software downloads. Code Signing is used to sign code and pre-built code using well-known tools such as signtool.exe and jarsigner.

2. Signtool

2.1 Tool description

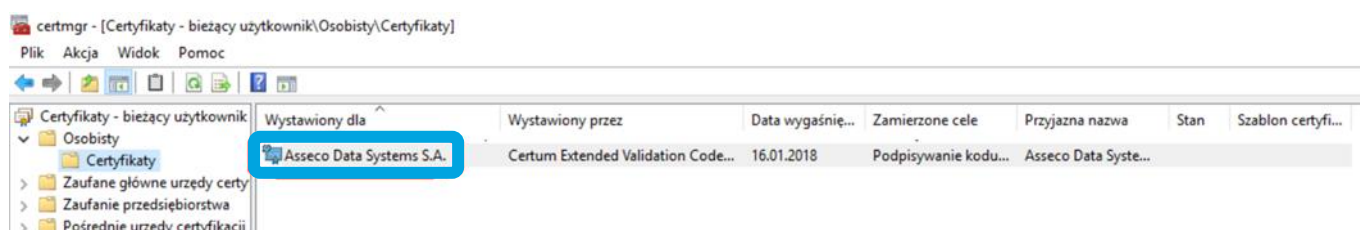
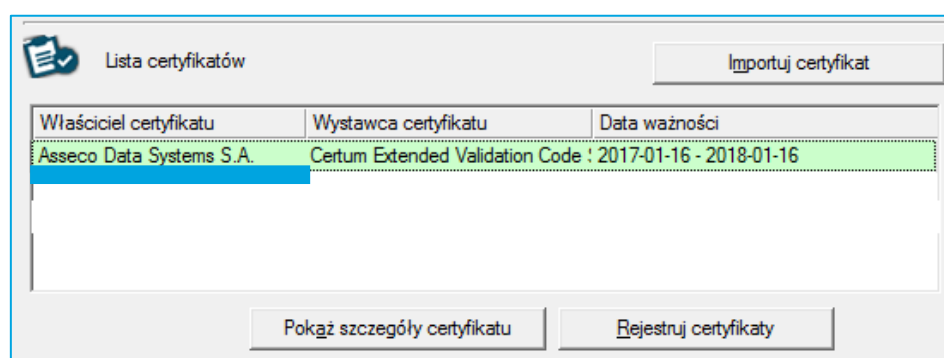
Signtool is a command line tool that digitally signs files, verifies file signatures and timestamps files. This tool can be found in the Windows development package (Windows SDK[Software Development Kit]). All operations performed with Code Signing require a connected reader together with a card on which there is a Code Signing certificate.

2.2 Signing

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

signtool sign /n "[1]" /t [2] /fd [3] /v [4]

[1] – Name or part of the name of the certificate's owner, which can be checked in the proCertum CardManager application or in the system tool certmgr.msc:



[2] – Timestamp Address. For Certum <http://time.certum.pl>,

[3] – The name of the signature algorithm. Available sha1 and sha256,

[4] – The path to the file to be signed.

Examples of correct commands:

```
signtool sign /n "Asseco Data Systems S.A." /t http://time.certum.pl/ /fd sha1 /v aplikacja.exe
```

As a result, cmd.exe console should return a message about the correctness of the file signature

```
The following certificate was selected:
  Issued to: Asseco Data Systems S.A.
  Issued by: Certum Code Signing CA SHA2
  Expires:   Fri Jul 06 10:16:38 2018
  SHA1 hash: E0828DF9D71C4CD87A349460027F0D9CB802BF31
```

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

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

```
signtool sign /n "Asseco Data Systems S.A." /t http://time.certum.pl/ /fd sha256 /v aplikacja.exe
```

As a result, the cmd.exe console should return a valid file signature message:

```
The following certificate was selected:
  Issued to: Asseco Data Systems S.A.
  Issued by: Certum Code Signing CA SHA2
  Expires:   Fri Jul 06 10:16:38 2018
  SHA1 hash: E0828DF9D71C4CD87A349460027F0D9CB802BF31
```

```
Done Adding Additional Store
Successfully signed and timestamped: aplikacja.exe
```

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

2.3 Verification

To verify the file, use the following command on the command line (cmd.exe):

```
signtool verify /pa [1]
```

[1] – The name of the signed file

Example of correct command:

signtool verify /pa aplikacja.exe

As a result, the cmd.exe console returns a message about the correctness of the file signature, for example:

```
File: aplikacja.exe
Index Algorithm Timestamp|
=====
0      sha1      Authenticode

Successfully verified: aplikacja.exe
```

```
File: aplikacja.exe
Index Algorithm Timestamp
=====
0      sha256     Authenticode

Successfully verified: aplikacja.exe
```

Or lack of a signature:

```
File: aplikacja.exe
Index Algorithm Timestamp
=====
SignTool Error: No signature found.

Number of errors: 1
```

2.4 Batch signing

In order to batch sign multiple files during a single session, they should be given as consecutive command parameters. This eliminates the need to run the command each time in the console and to enter the PIN code when signing subsequent files.

Example command:

```
signtool sign /n "Asseco Data Systems S.A." /t http://time.certum.pl/ /fd sha1 /v aplikacja1.exe aplikacja2.exe aplikacja3.exe
```

As a result, the cmd.exe console returns a message about the correctness of the file signature:

```

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

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

```

2.5 Dual signature

In order to create a dual signature (using both algorithms: SHA-1 and SHA-2 the following procedure should be carried out:

1. Perform an application signature using the SHA-1 algorithm with the example command:

```
signtool sign /n "Asseco Data Systems S.A." /t http://time.certum.pl/ /fd sha1 /v aplikacja.exe
```

2. Then perform a signature on the same application using the SHA-2 algorithm and the /as switch:

```
signtool sign /n "Asseco Data Systems S.A." /t http://time.certum.pl/ /fd sha256 /as /v aplikacja.exe
```

The result of the verification of the dual-signed file should be the following console message:

```

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

Successfully verified: aplikacja.exe

```

Windows 8 or higher is required to perform and verify a dual signature. To perform or verify a dual signature on Windows 7 systems, please refer to this article published by Microsoft: <https://technet.microsoft.com/en-us/library/security/2949927>.

3 Jarsigner

3.1 Tool description

Jarsigner is a command line tool that digitally signs files and verifies signatures. This tool can be found in the Oracle development package (JDK [Java Development Kit]). All operations performed with Code Signing require a connected reader with a card on which there is a certificate.

3.2 Configuration

3.2.1 Create a configuration file provider.cfg

Before using jarsigner, additional configuration is needed. The first step is to create a provider configuration file for PKCS#11. To do so, create a new file with the extension *.cfg (example: provider.cfg). Its content looks like this:

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

[1] – Provider Name. Preferably Crypto3PKCS.

[2] – The path to the PKCS library. If you have installed proCertum CardManager the default path is: C:\Windows\System32\crypto3PKCS.dll

[3] – The slot number in which the card is located. The default value is -1 which automatically detects the first available slot.

Example configuration for ordinary profile of cryptoCertum card:

name=Crypto3PKCS
library=C:\Windows\System32\crypto3PKCS.dll
slot=-1

Example configuration for Certum virtual card:

name=SimplySignPKCS.dll
library=C:\Windows\System32\SimplySignPKCS.dll
slot=-1

3.2.2 Create certificate path file bundle.pem

The next step is to create a certificate path file with extension*.pem (example: bundle.pem). Its contents looks like this:

1. "Above": User Certificate
2. "Below": intermediate certificate for user certificate
3. "Bottom": cross certificate*

***Note:** cross certificate must be downloaded from:

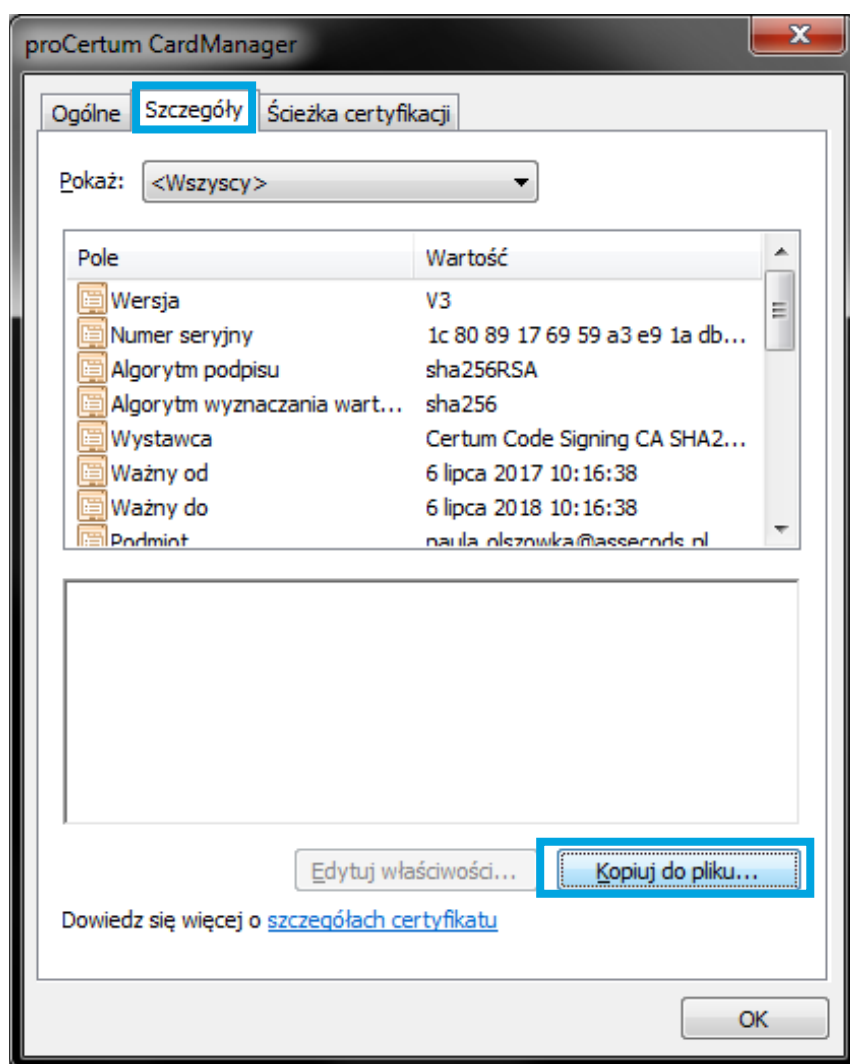
https://www.certum.eu/en/cert_expertise_root_certificates/(certificate serial number: 1bb58f252adf23004928c9ae3d7eed27)

Note: The contents of the bundle.pem file must necessarily be in the order listed above.

Obtaining a user Certificate

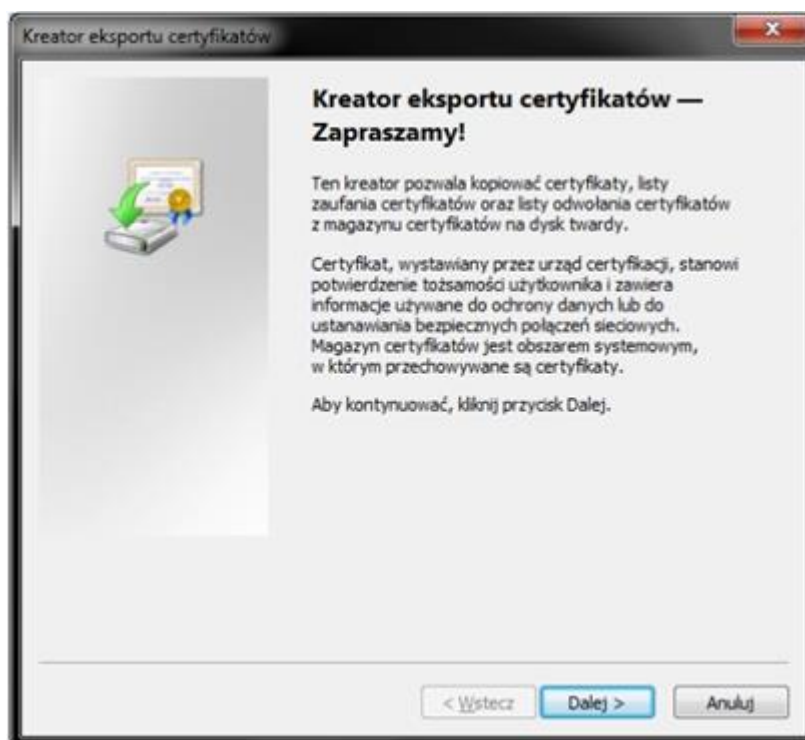
The user's certificate can be obtained by starting the proCertum CardManager program, clicking on the Read Card button and going to the Common Profile tab.

Then select the certificate you want to save from the list and use the "Show certificate details" button. The certificate will be displayed, using the "Copy to file" button on the "Details" tab you can save the certificate:



It is a good idea to write down the contents of the [Issuer](#) field in this step. This will help you later in selecting the intermediate certificate

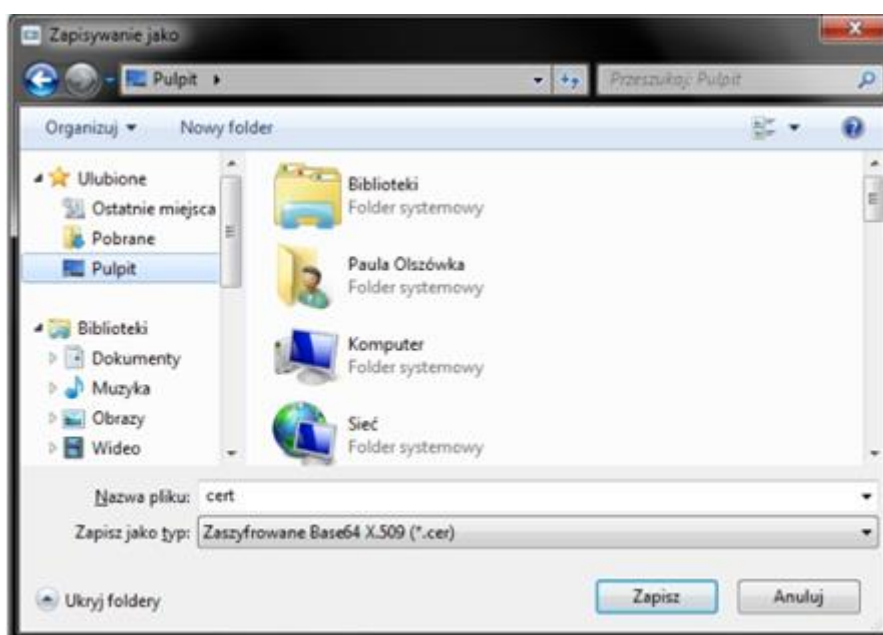
When you click "Copy to File", the Save Wizard is launched:



Click [Next](#). In the next step, select "X.509 encrypted with Base-64 algorithm (.CER)" and click [Next](#):



Next, select where to save the file and name it. To do this, click [Browse](#), select a location and enter a name for the file, then click [Save](#), and in the wizard window go to the next step by clicking [Next](#) and then [Finish](#). The wizard will confirm the export of the file.



Obtaining an Intermediate Certificate

Intermediate certificates should be downloaded from Certum's website:

https://www.certum.eu/en/cert_expertise_root_certificates/

The name of the Issuer in the "Issuer" field of the user's certificate can be used to select the correct intermediate certificate(s). Please find the issuer of your certificate on the Certum website and save his certificate in the PEM text format..

Then, having two certificate files, create a new text file. Paste the content of the two files you got earlier (the user certificate and intermediate certificate) into one text file in the order mentioned above:

1. „Above”: User Certificate
2. "Below": intermediate certificate for user certificate
3. "Bottom": cross certificate*

***Note:** cross certificate must be downloaded from:

https://www.certum.eu/en/cert_expertise_root_certificates/ (certificate serial number: 1bb58f252adf23004928c9ae3d7eed27)

Save the file and change its extension to *.pem.

Below is a sample bundle.pem file:

```

-----BEGIN CERTIFICATE-----
MIIHIDCCBQigAwIBAgIQaYbDkbg4OMOF7bNN4bc/QDANBgkqhkiG9w0BAQsFADBW
MQswCQYDVQQGEwJQTDEHMB8GALUEChMYQXNzZWNVIERhdGEgU3lzdGltcyBTLkEu
MSQwIgwYDVQQDEXTDZlXj0dW0gQ29kZSBTaWduaW5nIDIwMjEwMjEgQ0EwHhcNMjEw
MDAwMDAwWWhcNMjEwMDAwMDAwMDAwWjCBxTELMakGALUEBhMCUEwxFtATBgNVBAGM
DHBvZGthcnBhY2tpZTERMA8GALUEBwwIUUnplc3rDs3cxJTAjBgNVBAsMHFBpb24g
QWRtaW5pc3RyYWNqaSBSESfZG93ZWoxGzAZBgNVBAOMekFzc2VjbyBQb2xhbmQg
Uy5BLjEibmBkGALUEAwSQQXNzZWNVIFBvbGZuZCBTLkEuMSswKQYJKoZIhvcNAQk
BhxxvCHJvZ3JhbW93YW5pZS55YXBAYXNzZWNVLnBsMIIICjANBgkqhkiG9w0BAQE
AAOCAG8AMIICCGKCAgEArpPvPT1KRJWFgHB5cgK9FKiYaKB7OnvS8zMVYP6rUP7C
F10jRxe9k+fAM+TmtW+34fQm5nq0eB4d15WrZhuAcOcrWz9K7UxMFlalf6peXkSq
hpZuc8fQLqnTCR0Vtng+mch+hviKcUyzeUk3PSKCzW3IbYvU4wXUPglugE6tKuL+
/dQWsfTNkPOZEikaDR/5oRdWUQrHCmjtCTQAcDhPHbvU57iKQK+IId4k0r5Fv4LW
j8ylw5dRQ2010x5Q1oEQrDoug0/p6JYiNXJEKEbjkaXTVUKtkjFdjgtsRRdQ0teA
qiUMxgCqedf55PyMQu55EU4MRkutDdvWh+vEgr56EtuEo4Ci0QDuv/mPuynN4yX6
BLVSpX78VwGk+zoNw8bUJqcvhH8r3J2B13BOPLUPHwEQdckT1P9D2pDdJ6vYt5MI
-----END CERTIFICATE-----

-----BEGIN CERTIFICATE-----
MIIGuTCCBKGGAwIBAgIRAJmJgAomVTt1q9xuhKaz6jkwDQYJKoZIhvcNAQEMBQAw
gYACzAJBgNVBAYTAlBMMSiWIAyDVQQKExlVbml6ZXRVIFRlY2hub2xvZ2llcyBT
LkEuMSswCQYDVQQLEX5DZXJ0dW0gQ2VydG1maWNhdGlvbiBBdXR0b3JpdHkxJDAi
BgNVBAMTG0N1cnRlbSBUCnVzdGVKIE5ldHdvcm5gQ0EgMjEwMDAwMDAwMTkNTMy
MThaFw0zNjAlMTgwNTMyMThaMFYxCzAJBgNVBAYTAlBMMSEwHwYDVQKEExhBc3Nl
Y28gRGF0YSBTeXN0ZW1zIFMuQS4xJDAiBgNVBAMTG0N1cnRlbSBDb2RlIFNpZ25p
bmcmGjAyMSBBDQTCcAIIwDQYJKoZIhvcNAQEBBQADggIPADCCAgCGgIBAJ0jzwQw
IzvBRiznM3M+Y1l6dbq+XE26vest+L7k5n5TeJkgH4Cyk74IL9uP61olRxsxU/WB
AE1TMNQI/HsE0uCJ3VPL0UufnY0qDHG7yCnJOvoSNbIbMPT+Cci75scCx7UsKK1
fcJo4TXetu4du2vEXa09Tx/bndCBfp47zJNsamzUyD7JlrcNxoW5g6FJg0ImIv7n
CeNn3B6gZG28WAwE0mDqLrvU49chyKIc7gvCjan3GH+2eP4mYJASf1BTQ3H0s6JG
driSMVoD1lzbJobtYDF4L/GhlLEXWgrVQ9m0pW37KuwYqpY42grp/kSYE4BUQrBL
gBMNKRvfhQPskDfZ/5GbTCyvlqPN+0OEDmYgK1VkoMenDO/xtMrMINRJS5SY+jWC
i8PRHAVx00xdx8m2bWL4/ZQldp0/JhUpHEpABMc3eKax8GI1F03mSJVV6o/nmmKq
-----END CERTIFICATE-----

-----BEGIN CERTIFICATE-----
MIIFyTCCBLGgAwIBAgIQG7WPJSrfIwBJKmmuPX7tJzANBgkqhkiG9w0BAQwFADB+
MQswCQYDVQQGEwJQTDEIMCAGALUEChMZVW5pemV0byBUZWNobm9sb2dpZXMGUy5B
LjEnMCUGALUECXMqQ2VydHVTIEN1cnRpb2MlYXRpb24gQXV0aG9yaXR5MSIWIAYD
VQDExlDZXJ0dW0gVHJlc3RlZCB0ZXR3b3JrIENBMB4XDTIxMDUzMTA2NDMwN1oX
DTI5MDkxNzA2NDMwN1owGyYACzAJBgNVBAYTAlBMMSiWIAyDVQQKExlVbml6ZXRV
IFRlY2hub2xvZ2llcyBTLkEuMSswCQYDVQQLEX5DZXJ0dW0gQ2VydG1maWNhdGlv
biBBdXR0b3JpdHkxJDAiBgNVBAMTG0N1cnRlbSBUCnVzdGVKIE5ldHdvcm5gQ0Eg
MjEwMDAwMDAwMTkNTMyMThaFw0zNjAlMTgwNTMyMThaMFYxCzAJBgNVBAYTAlBMMSEw
HwYDVQKEExhBc3NlY28gRGF0YSBTeXN0ZW1zIFMuQS4xJDAiBgNVBAMTG0N1cnRlb
SBDb2RlIFNpZ25pbmcmGjAyMSBBDQTCcAIIwDQYJKoZIhvcNAQEBBQADggIPADCCAgCGg
IBAL35ePjmlYAMZJ2GG5ZK
Zz8iOh51AX3v+lxnjMnMXGupkea5QuUgS5vam3u5mV32m4BL14RAKyft6Lowuz4J
GqdJle8rQCTCl8en7psl76gKAJeFWqqd3CnJ4jUH63BNSbtBsla4oUE4m9H7MX+P
4F/hsT8PjhZJYNcGjRj5qiYQyrtONFnjRtGvkcwLS5y0cVj2udjeUR+S2MkiYYu
ND8pTFKLKqfA4pEoibnAW/kd2ecnrf+aApfBx1CSmwIsvam5NFkKv4RK/9/+s5/r
2Z7gmCPspmt3FirbzK07HKSH3EzZxhliAEVX5JCCQrtClvBh4MGjFWajXfQY7ojJ
jRdFKZkydQix7ikmyG5C5rV1RX83FVojAInUPt5OJ7DwQAY8TrfLTaKzHtAGWt32
-----END CERTIFICATE-----

```

User Certificate

Intermediate Certificate

Cross Certificate

3.2.3 Change the user alias (label) on a card (only for users with diacritical marks in the Common Name (CN) field in a certificate)

A user certificate alias is used to indicate the certificate to be used to sign the application using the Jarsigner tool.

By default, the certificate alias is created based on the contents of the Common Name field from the Entity field of the certificate. If the Common Name field contains a string containing diacritics, it must be replaced with a string that does not contain diacritics.

Example:

Alias (label) before the change: Urzqd

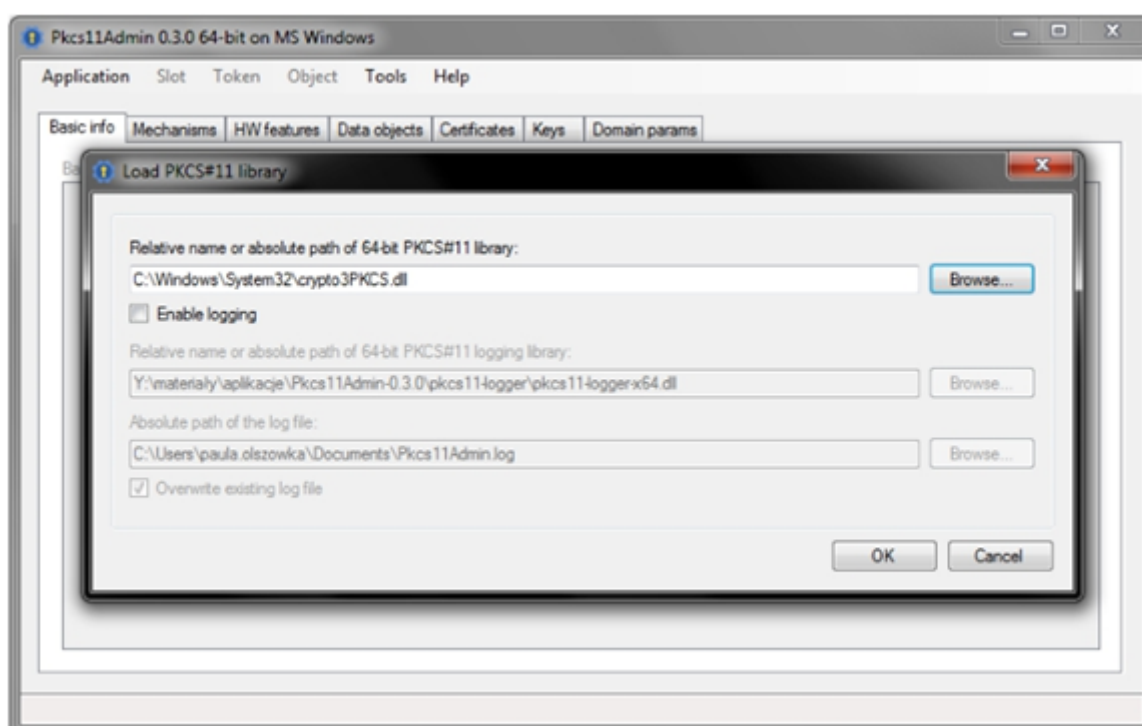
Alias (label) after the change: Urzad

Note: Changing the alias does not change the certificate. Only the identifier of the certificate on the card is subject to editing. The signed application in the signature field will still contain the Subscriber's data containing diacritical marks.

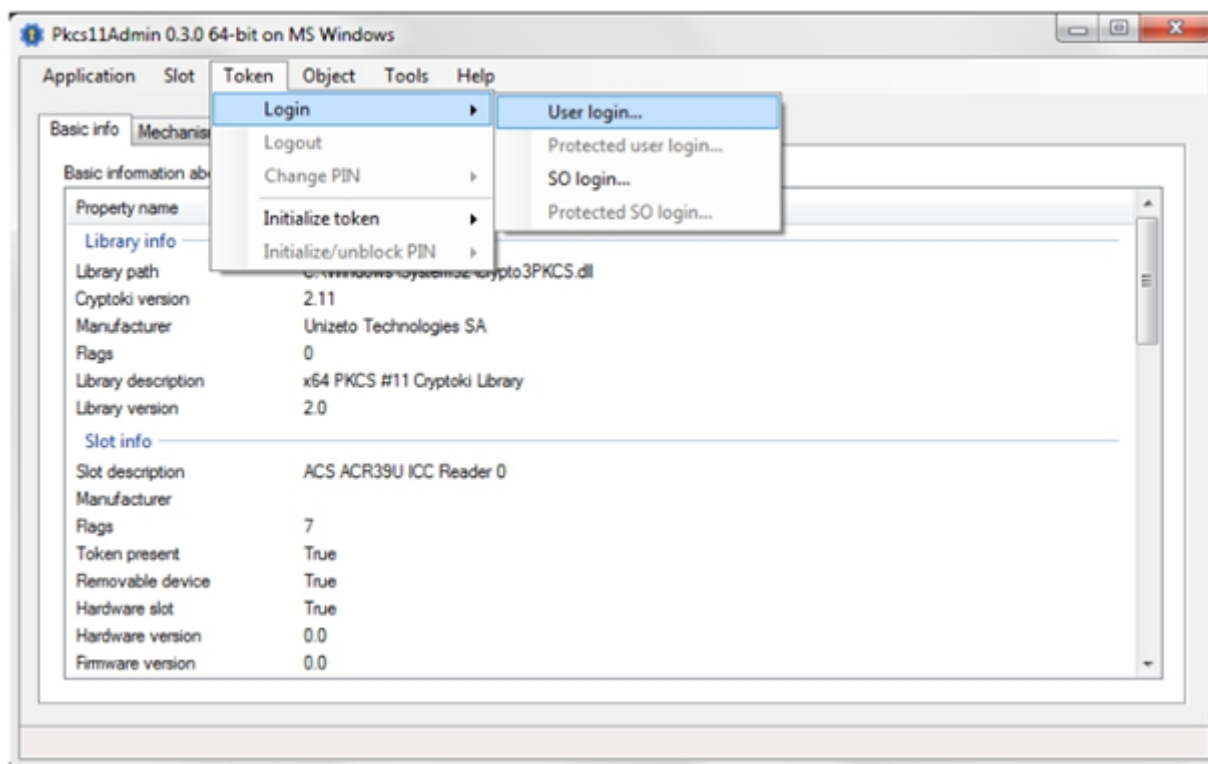
Changes to the alias (label) can be made using software provided free of charge PKCS11Admin, downloadable here: <http://www.pkcs11admin.net/>

The presented method of changing the label has been implemented in PKCS11Admin software version 0.3.0. Procedure of changing label looks as follows:

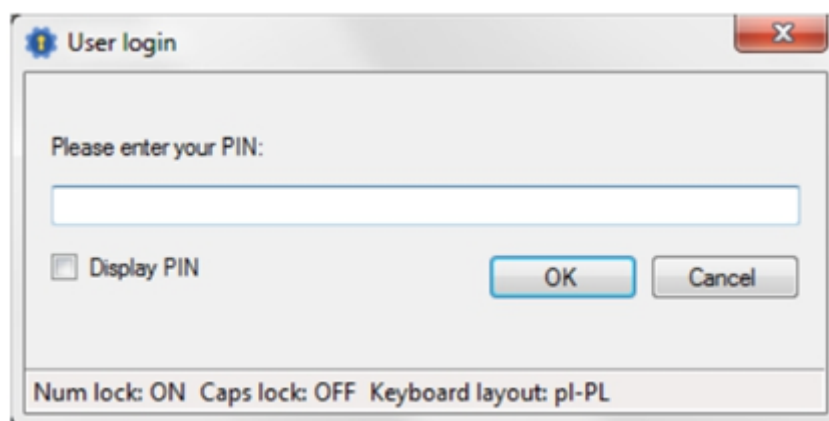
1. Download and extract PKCS11Admin to a directory of your choice.
2. Then enter the directory containing the extracted PKCS11Admin software and run the PKCS11Admin-x86 or PKCS11Admin-x64 application, depending on your operating system version. The launch results in the opening of the program and the selection window of the library supporting the card. For Certum cards choose the [crypto3PKCS.dll](#) library located in the [C:\Windows\System32](#) directory and click **OK**:



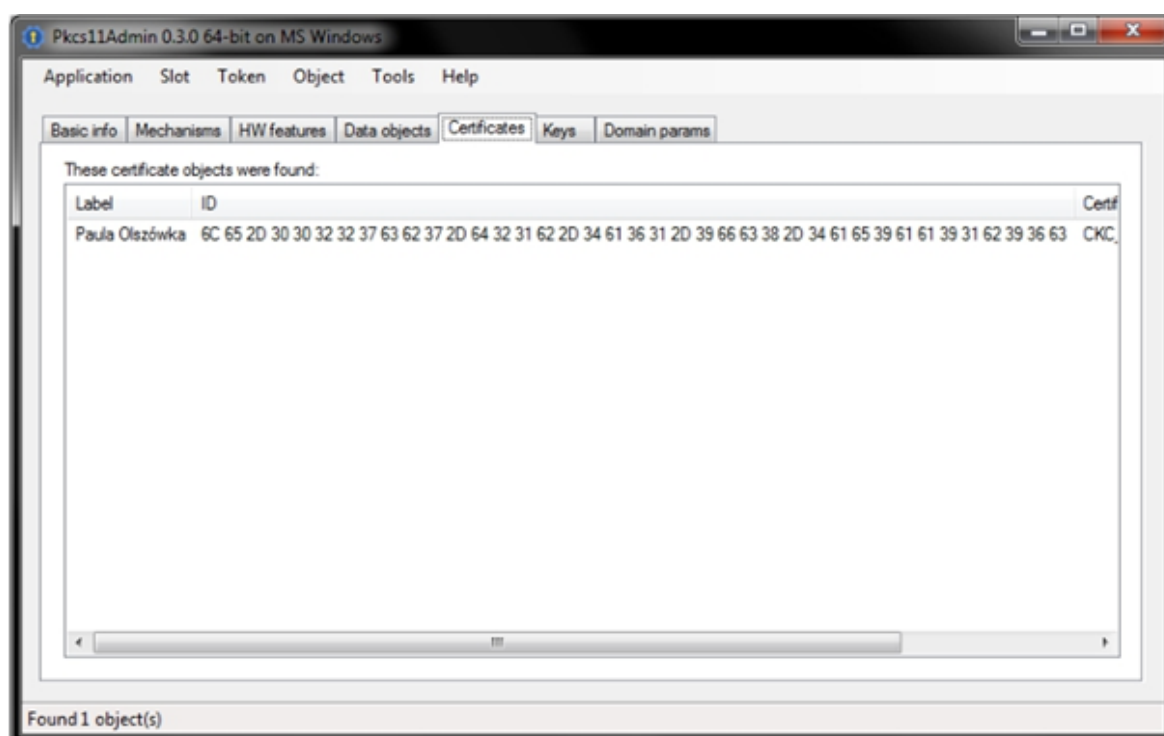
3. When the program loads the contents of the cryptographic card, select **Token > Login > User login** from the bar:



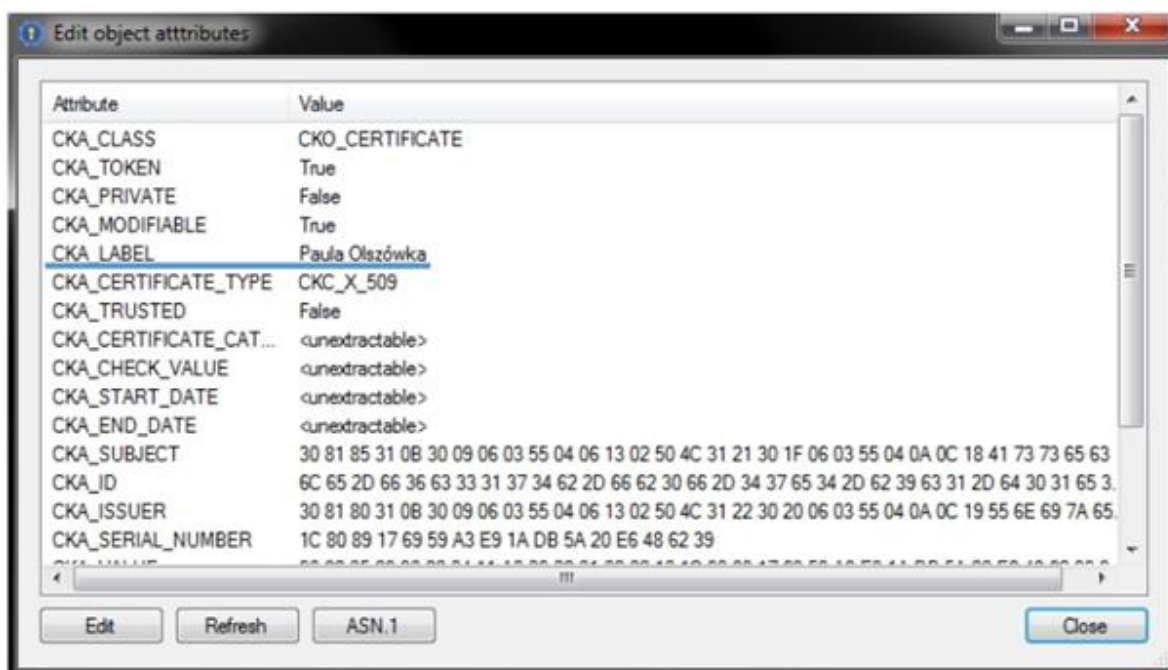
4. The program will prompt you to enter the PIN code for the regular card profile. Enter it and confirm with **OK** button:

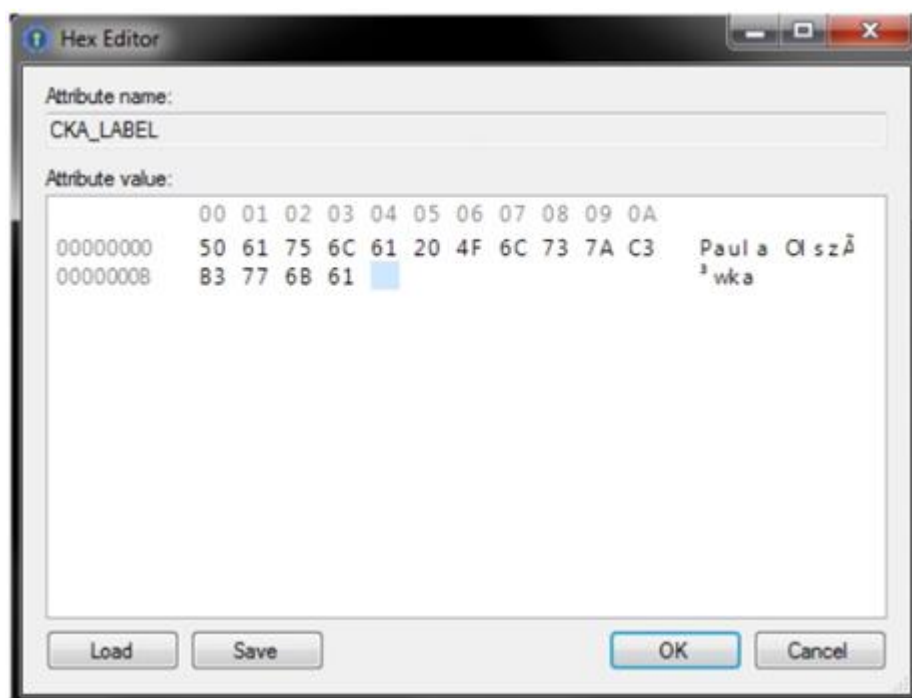


5. Then go to the **Certificates** tab. The list will display the labels of certificates for which the change will be made (here, for example, the user label containing a diacritical mark „ó”):

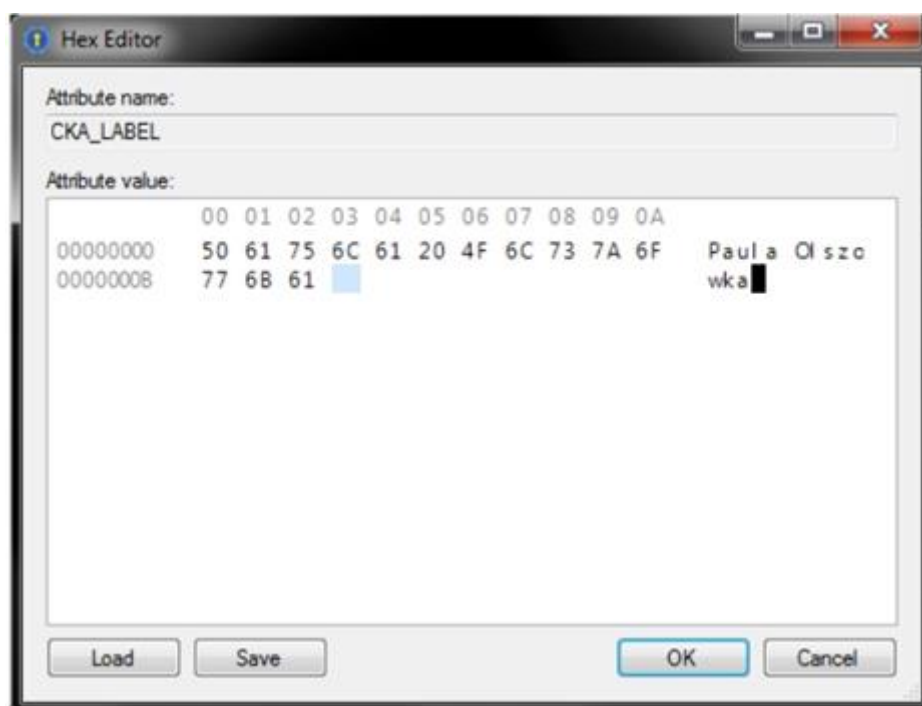


6. Right-clicking on the label calls up a menu from which you can select the [Edit attributes...](#)
7. In the window that appears, presenting the attributes of an entry, find the entry [CKA_LABEL](#). Then select the [CKA_LABEL](#) entry and use the [Edit](#) button to change the incorrect content of the entry:





8. The label field is now editable. Simply correct the label by removing the diacritical marks and confirm with **OK**:



9. The change process is now complete. Thanks to this when selecting a certificate to sign you will be able to provide an alias without diacritics and thus the correct use of the certificate.

Before signing, the result of changing the user alias can be checked with the command:

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

As a result, the instruction returns the contents of the key store:

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

Keystore type: PKCS11
Keystore provider: SunPKCS11-Crypto3CSP

Your keystore contains 1 entry

Paula Olszowka, PrivateKeyEntry,
Certificate fingerprint (SHA1):
E0:82:8D:F9:D7:1C:4C:D8:7A:34:94:60:02:7F:0D:9C:B8:02:BF:31
```

3.3 Signing

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

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

- [1]** – Timestamp Address. For Certum <http://time.certum.pl>,
- [2]** – The path to the certificate path file (Section "Configuration"),
- [3]** – The path to the provider's configuration file (Section "Configuration"),
- [4]** – Password for common card profile,
- [5]** – The path to the file to be signed,
- [6]** – The name of the certificate owner which can be checked in the proCertum CardManager or with the Keytool too

Example of correct command:

```
jarsigner -keystore NONE -certchain "bundle.pem" -tsa "http://time.certum.pl" - storetype PKCS11 -  
providerClass sun.security.pkcs11.SunPKCS11 -providerArg "provider.cfg" -storepass "123456" "aplikacja.jar"  
"Asseco Data Systems S.A."
```

If the signature operation was successful, the console will display the following result:

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

3.4 Verification

To verify the file, use the following command on the command line (cmd.exe):

```
jarsigner -verify "[1]"
```


[1] – The path to the file to be signed,

Example of correct command:

```
jarsigner -verify "aplikacja.jar"
```

If the file is verified correctly, the console will display:

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

If there is no signature, the result is as follows:

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

3.5 Batch signing

In order to batch sign multiple files in a single session, it is necessary to create a *.bat file containing as many entries as the number of files to be signed during one signing process. This eliminates the need to call the command in the console each time and to enter the PIN code when signing subsequent files..

To create a file, create a new text file *.txt, paste the file signing entries, save the file, and change its extension from *.txt to *.bat.

The following example shows the contents of the *.bat file for signing three applications simultaneously:

```
jarsigner -keystore NONE -certchain "bundle.pem" -tsa "http://time.certum.pl" -storetype PKCS11 -  
providerClass sun.security.pkcs11.SunPKCS11 -providerArg "provider.cfg" -storepass "123456"  
"aplikacja1.jar" "Asseco Data Systems S.A."
```

```
jarsigner -keystore NONE -certchain "bundle.pem" -tsa "http://time.certum.pl" -storetype PKCS11 -  
providerClass sun.security.pkcs11.SunPKCS11 -providerArg "provider.cfg" -storepass "123456"  
"aplikacja2.jar" "Asseco Data Systems S.A."
```

```
jarsigner -keystore NONE -certchain "bundle.pem" -tsa "http://time.certum.pl" -storetype PKCS11 -  
providerClass sun.security.pkcs11.SunPKCS11 -providerArg "provider.cfg" -storepass "123456"  
"aplikacja3.jar" "Asseco Data Systems S.A."
```

You can run this file in cmd.exe console or by double-clicking it, and the result will be starting the signing of subsequent files contained in the *.bat file.

The result of running the *.bat file in the console will be information about the next command invocation and file signature:

```
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 "123456" "aplikacja1.jar" "Asseco Data
Systems S.A"
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 "123456" "aplikacja2.jar" "Asseco Data
Systems S.A"
```

```
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 "123456" "aplikacja3.jar" "Asseco Data
Systems S.A"
Picked up _JAVA_OPTIONS: -Xms256m -Xmx1024m
jar signed.
```

4. Most common problems

1. During signing with Signtool tool using SHA-2 algorithm there is a problem with signing:

```
Done Adding Additional Store
SignTool Error: An unexpected internal error has occurred.
Error information: "Error: SignerSign() failed." (-
2146893784/0x80090028)
```

Solution: In the proCertum CardManager software select the [Options](#) button > check the option [EV Code Signing - replace CSP with minidriver library](#). Then restart the system and try to sign again.

2. When signing with Jarsigner, the following message appears:

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

```
Warning:
The signer's certificate chain is not validated.
```

Solution: Verify the contents of the [bundle.pem](#) file. The file probably contains invalid certificates or certificates in the wrong order.

The [bundle.pem](#) file should contain certificates:

1. Subscriber's Certificate,
2. Appropriate intermediate certificate.

More about the bundle.pem file in section 3.2.2.

3. When verifying the signature with the Jarsigner tool, a message appears:

```
Picked up _JAVA_OPTIONS: -Xms256m -Xmx1024m
jarsigner: java.lang.SecurityException: cannot verify signature
block file META-INF/PAULA_OL
```

PAULA_OL is a sample signature that depends on the user's alias. For more on aliases, see section 3.2.3.

Solution: Verify the contents of the [bundle.pem](#) file. The file probably contains invalid certificates or certificates in the wrong order.

The [bundle.pem](#) file should contain certificates:

1. Subscriber's Certificate,
2. Appropriate intermediate certificate.

4. When signing with the Jarsigner tool, a message appears:

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

```
Warning:
The signer certificate's KeyUsage extension doesn't allow code
signing.
```

Solution: Verify the contents of the [bundle.pem](#) file. The file probably contains invalid certificates or certificates in the wrong order.

The [bundle.pem](#) file should contain certificates:

1. Subscriber's Certificate,
2. Appropriate intermediate certificate.

More about the bundle.pem file in section 3.2.2.