

Code Signing w chmurze – Podpisywanie w Signtool oraz Jarsigner

Wer. 1.1

assecO

 **Certum**
by assecO

Spis treści

1. Opis produktu	3
2. Podpisywanie w środowisku Windows	3
Sprawdzenie dostępu do usługi i wyświetlenie certyfikatów	3
Podpisywanie narzędziem signtool – podpis pojedynczy	8
Podpisywanie narzędziem signtool – Podpisywanie wsadowe	11
Podpis dualny	11
Podpis dualny	12
Weryfikacja podpisu narzędziem signtool	13
Podpisywanie narzędziem jarsigner.....	13
Utworzenie pliku konfiguracyjnego provider.cfg.....	13
Utworzenie pliku ścieżki certyfikatu bundle.pem.....	14
Uzyskanie aliasu certyfikatu.....	20
Podpisywanie	21
Podpisywanie wsadowe.....	21
Weryfikacja pliku narzędziem jarsigner	23

1. Opis produktu

Certyfikat Standard Code Signing w chmurze to certyfikat przechowywany na karcie wirtualnej w usłudze SimplySign.

Certyfikat Code Signing umożliwia cyfrowe podpisanie aplikacji, sterowników, poświadczając ich autentyczność i bezpieczeństwo. Dzięki temu użytkownicy Twojego oprogramowania zyskują pewność, że nie zostało ono zmodyfikowane, zainfekowane lub uszkodzone przez osoby trzecie.

Podpisanie aplikacji z pomocą Code Signing eliminuje problem anonimowości kodu w sieci. Dzięki cyfrowemu podpisowi zyskasz pewność, że użytkownicy nie zobaczą ostrzeżenia o "nieznanym wydawcy" w trakcie instalacji lub uruchamiania Twojego programu i upewnią się o jego bezpieczeństwie. Podpisanie aplikacji pozwala chronić zarówno użytkowników, jak i reputację Twojej marki.

Cyfrowe podpisywanie kodu sprawia, że korzystanie z aplikacji jest bezpieczne, co przekłada się na większe zaufanie do Twojej marki i poszerzenie grona klientów.

2. Podpisywanie w środowisku Windows

Sprawdzenie dostępu do usługi i wyświetlenie certyfikatów

Po odzyskaniu dostępu do usługi, na urządzeniu przenośnym, w aplikacji **SimplySign** generowany jest tzw. token pozwalający na logowanie się do konta SimplySign.



Rysunek 1: Aplikacja SimplySign – wygenerowany token

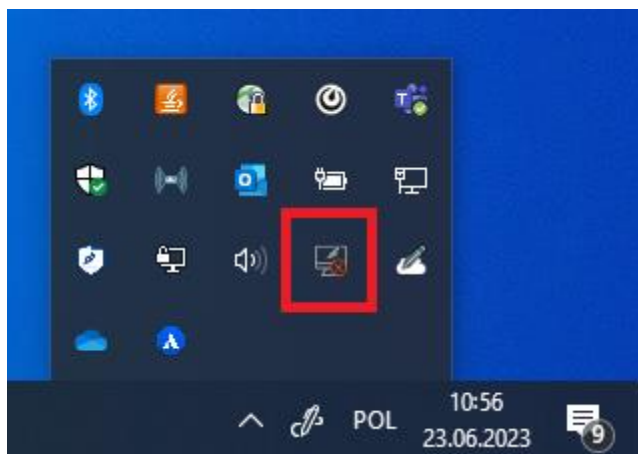
W środowisku Windows, przy pomocy aplikacji **SimplySign Desktop** można sprawdzić poprawność generowania tokenów i zawartość konta SimplySign.

W celu instalacji aplikacji **SimplySign Desktop** dla Windows należy przejść do strony:

<https://pomoc.certum.pl/pl/oprogramowanie/procertum-simplysign/>

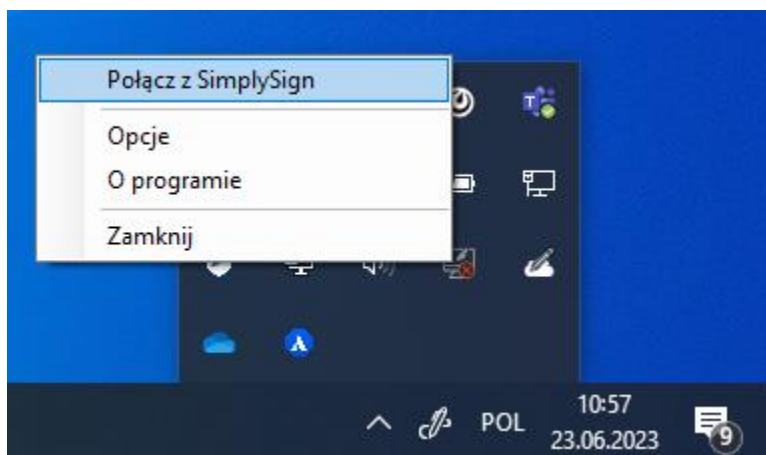
Ze strony należy pobrać odpowiedni pakiet i zainstalować aplikację.

Po zainstalowaniu aplikacji należy ją włączyć – w tzw. tray'u – obok zegara systemowego pojawi się ikona aplikacji.



Rysunek 2: Aplikacja SimplySign Desktop – ikona

Następnie należy nacisnąć prawym klawiszem myszy na ikonę aplikacji – pojawi się menu.




Rysunek 3: Aplikacja SimplySign Desktop – menu

Należy wybrać polecenie **Połącz z SimplySign**. Pojawi się okno logowania do usługi.

SimplySign Desktop ver. 1.0.0.52

Logowanie

 **Certum**
Identity Provider
by ASSECO

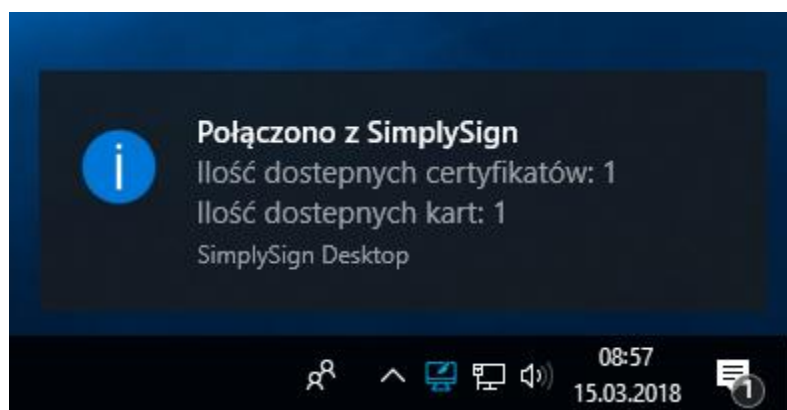
Identyfikator:
Wprowadź swój adres e-mail

Token:
Wprowadź token z SimplySign

Ok Anuluj

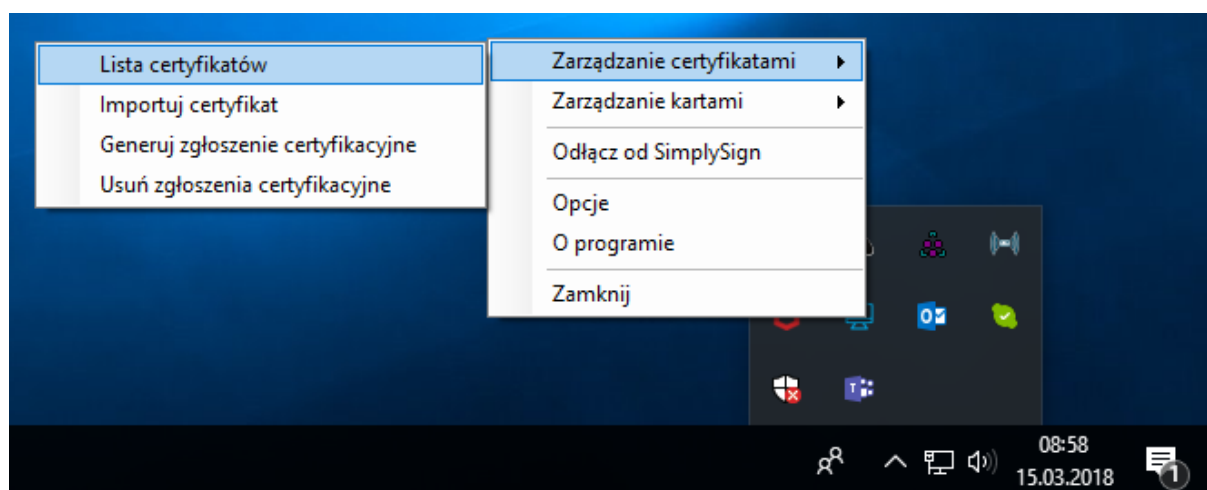
Rysunek 4: Aplikacja SimplySign Desktop – logowanie do usługi

Należy wprowadzić nazwę użytkownika i token generowany na urządzeniu mobilnym i nacisnąć przycisk **Ok**. Jeżeli wprowadzone zostaną poprawne dane to nastąpi zalogowanie do usługi – wyświetlone zostanie stosowne powiadomienie z informacją o ilości kart i certyfikatów.



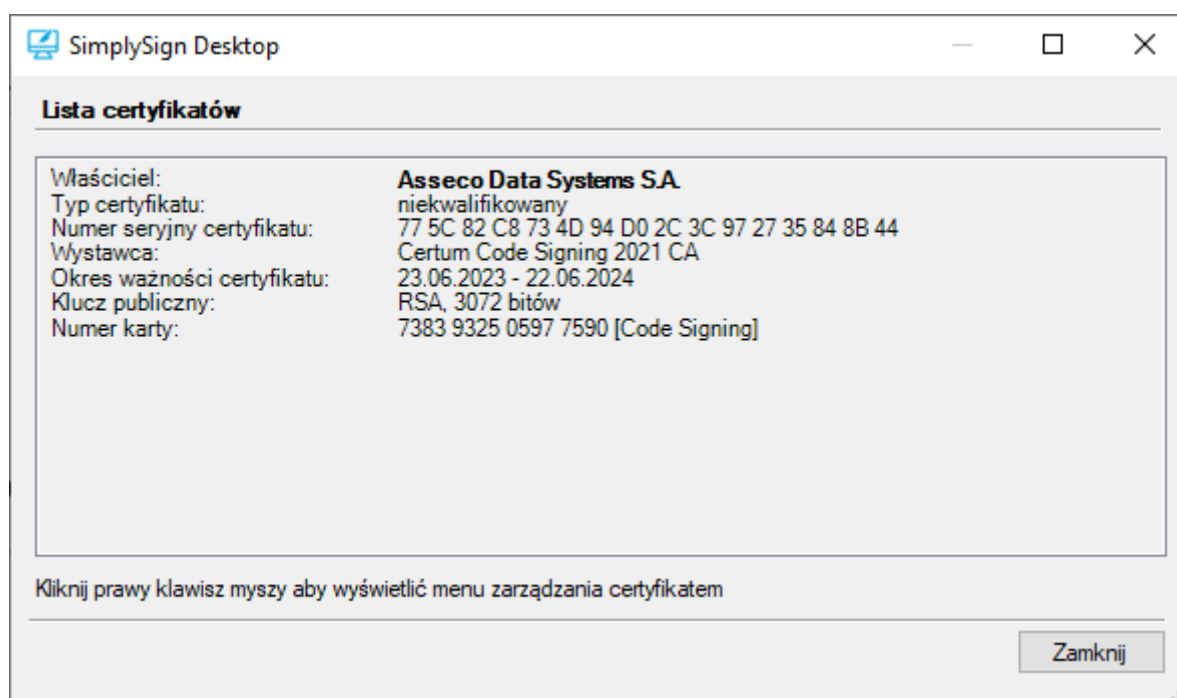
Rysunek 5: Aplikacja SimplySign Desktop – informacja po zalogowaniu się do usługi

Po zalogowaniu należy wyświetlić listę certyfikatów. W tym celu należy kliknąć prawym klawiszem myszy na ikonę aplikacji **SimplySign Desktop** i z menu wybrać **Zarządzanie certyfikatami** → **Lista certyfikatów**.



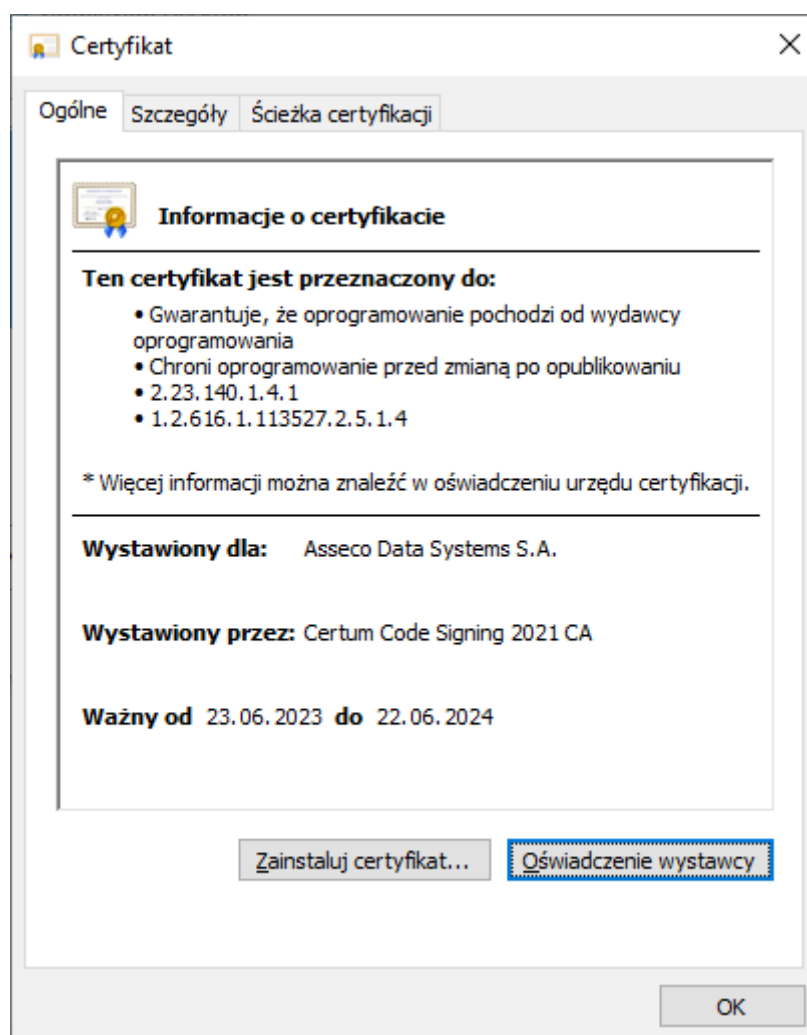
Rysunek 6: Aplikacja SimplySign Desktop – menu umożliwiające wyświetlenie listy certyfikatów

Nastąpi wyświetlenie listy certyfikatów.



Rysunek 7: Aplikacja SimplySign Desktop – lista certyfikatów

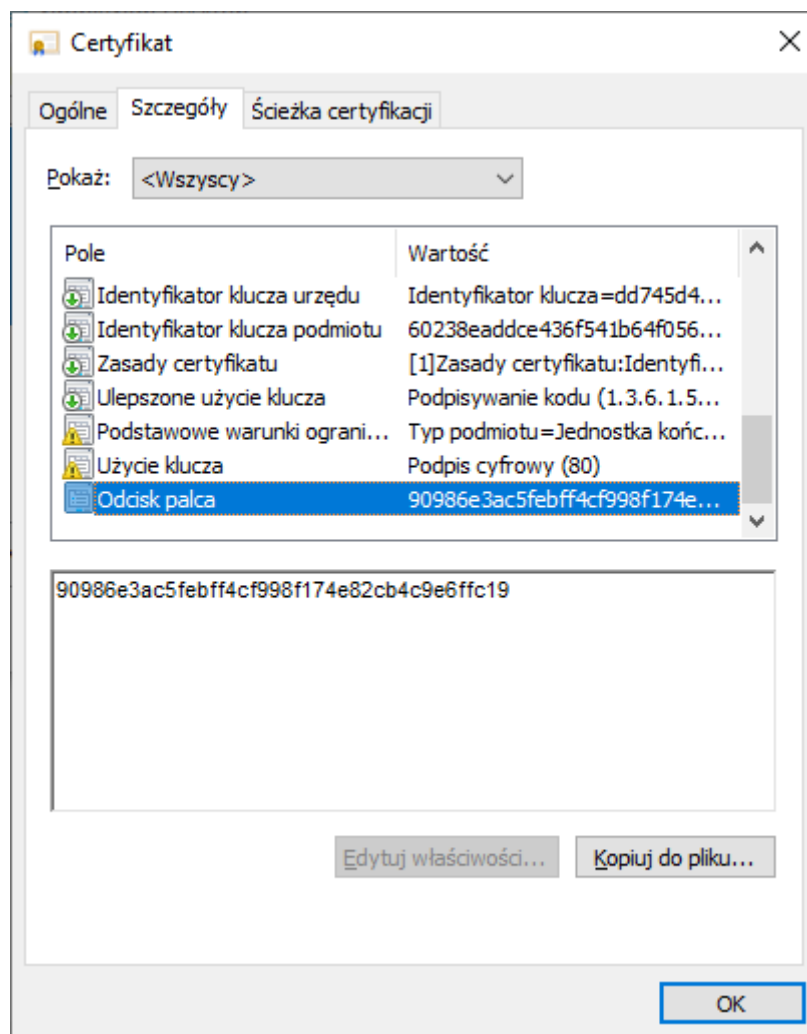
W celu wyświetlenia certyfikatu należy po prostu dwukrotnie kliknąć w jego obrębie - wyświetlone zostaną szczegóły tego certyfikatu.



Rysunek 8: Aplikacja SimplySign Desktop – lista certyfikatów

Podpisywanie narzędziem signtool – podpis pojedynczy

W celu wykonania podpisu narzędziem signtool konieczne jest ustalenie tzw. „**odcisku palca z certyfikatu**”. W tym celu należy wyświetlić certyfikat i przejść do zakładki **Szczegóły** i następnie przejść do pola **Odcisk palca**.



Rysunek 9: Szczegóły certyfikatu – wartość odcisku palca

Po uzyskaniu odcisku palca można przygotować polecenie pozwalające na podpisanie pliku. Składnia polecenia jest następująca:

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

[1] – tzw. odcisk palca certyfikatu – w poniższym przykładzie to wartość 90986e3ac5febff4cf998f174e82cb4c9e6ffc19

[2] – adres znacznika czasu – w poniższym przykładzie to wartość <http://time.certum.pl>

[3] – skrót jaki zostanie użyty do znacznika czasu – w poniższym przykładzie SHA-256

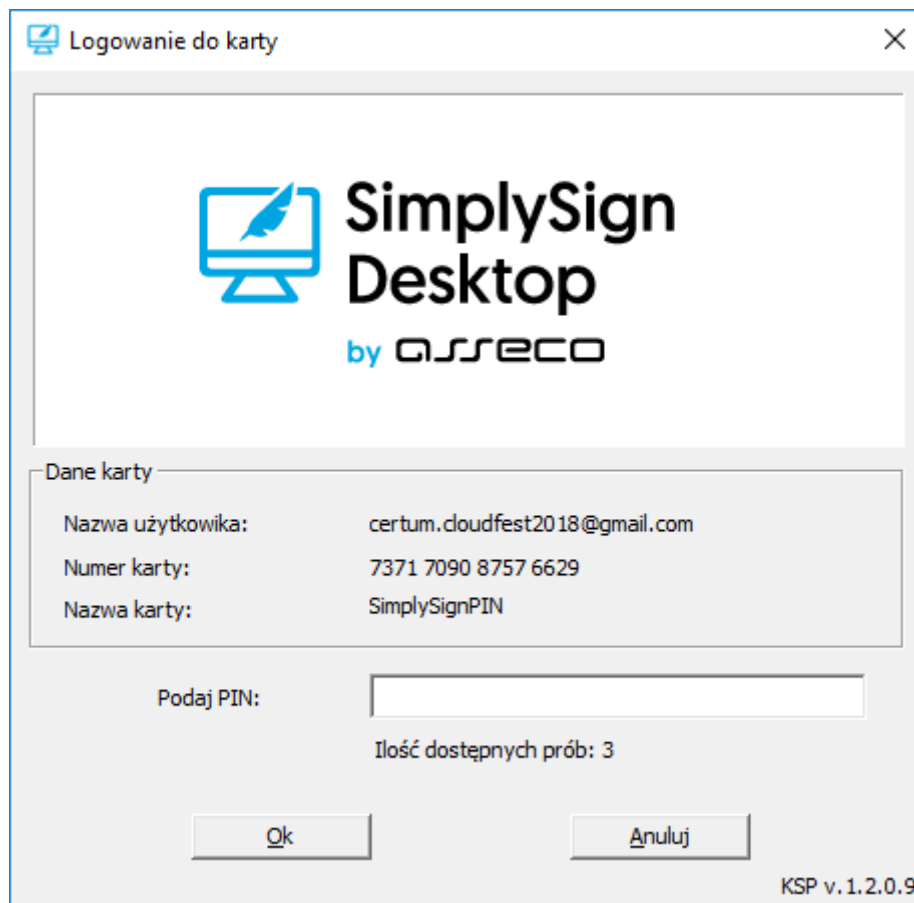
[4] – skrót jaki zostanie użyty do podpisu – w poniższym przykładzie SHA-256

[5] – ścieżka do pliku, który zostanie podpisany;

Przykładowe polecenie:

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

W przypadku kart pinowych, po wydaniu powyższego polecenia pojawi się okno, w którym należy wprowadzić kod PIN do karty SimplySign, na której znajduje się wskazany certyfikat.



Rysunek 10: Aplikacja SimplySign Desktop – wprowadzanie kodu PIN do karty

W przypadku kart bezpinowych, od razu nastąpi podpisanie pliku bez podawania kodu PIN.

W obydwu przypadkach na konsoli będzie widoczny następujący stan:

```
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
```

Podpisywanie narzędziem signtool – podpisywanie wsadowe

W celu wsadowego podpisania wielu plików podczas jednej sesji należy w poleceniu podpisu dla atrybutu `/v` podać kolejno pliki, które mają zostać podpisane. Działanie takie eliminuje konieczność każdorazowego wywoływania komendy w konsoli oraz wpisywania kodu PIN przy podpisie kolejnych plików.

Przykładowe polecenie:

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

W rezultacie konsola cmd.exe zwraca komunikat o poprawności podpisu plików:

```
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
```

Podpis dualny

W celu złożenia podpisu dualnego (wykorzystującego oba algorytmy: SHA-1 oraz SHA-2 należy przeprowadzić następującą procedurę:

1. Wykonać podpis aplikacji z wykorzystaniem algorytmu SHA-1 przykładowym poleceniem:

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

2. Następnie wykonać podpis tej samej aplikacji wykorzystując algorytm SHA-2 oraz przełącznik **/as**:

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

Wynikiem weryfikacji pliku podpisanego dualnie powinien być następujący komunikat z konsoli:

```
File: aplikacja.exe  
Index Algorithm Timestamp  
=====
```

0	sha1	RFC3161
1	sha256	RFC3161

```
Successfully verified: aplikacja.exe
```

Do wykonania i weryfikacji podpisu dualnego wymagany jest Windows 8 lub wyższy. W celu wykonania lub weryfikacji podpisu dualnego na systemach Windows 7 należy zapoznać się z artykułem opublikowanym przez Microsoft: <https://technet.microsoft.com/en-us/library/security/2949927>.

Podpis dualny

W celu złożenia podpisu dualnego (wykorzystującego oba algorytmy: SHA-1 oraz SHA-2 należy przeprowadzić następującą procedurę:

1. Wykonać podpis aplikacji z wykorzystaniem algorytmu SHA-1 przykładowym poleceniem:

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

2. Następnie wykonać podpis tej samej aplikacji wykorzystując algorytm SHA-2 oraz przełącznik **/as**:

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

Wynikiem weryfikacji pliku podpisanego dualnie powinien być następujący komunikat z konsoli:

```
File: aplikacja.exe  
Index Algorithm Timestamp  
=====
```

0	sha1	RFC3161
1	sha256	RFC3161

```
Successfully verified: aplikacja.exe
```

Do wykonania i weryfikacji podpisu dualnego wymagany jest Windows 8 lub wyższy. W celu wykonania lub weryfikacji podpisu dualnego na systemach Windows 7 należy zapoznać się z artykułem opublikowanym przez Microsoft: <https://technet.microsoft.com/en-us/library/security/2949927>.

Weryfikacja podpisu narzędziem signtool

Podpis wykonany narzędziem signtool można zweryfikować przy pomocy tego samego narzędzia. Składania takiego polecenia jest następująca:

signtool verify /pa /all [1]

[1] – nazwa weryfikowanego pliku – w przykładzie plik.exe

Przykładowe polecenie:

signtool verify /pa /all plik.exe

Po uruchomieniu przykładowego polecenia, na konsoli będzie poniższy stan:

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

Successfully verified: plik.exe
```

Podpisywanie narzędziem jarsigner

Przed rozpoczęciem używania jarsigner potrzebna jest dodatkowa konfiguracja.

Utworzenie pliku konfiguracyjnego provider.cfg

W pierwszym kroku należy utworzyć plik konfiguracyjny providera dla PKCS#11. W tym celu tworzymy nowy plik o rozszerzeniu *.cfg (przykład: provider.cfg). Jego zawartość wygląda następująco:

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

[1] – Nazwa providera. Najlepiej SimplySignPKCS.

[2] – Ścieżka do biblioteki PKCS. Ścieżka domyślna to: C:\Windows\System32\crypto3PKCS.dll

[3] – Numer slotu w którym znajduje się karta. Pierwszy slot ma numer 0, drugi numer 1 itd. W przypadku, gdy na koncie **SimplySign** jest jedna karta to należy ustawić 0. W przypadku, gdy na koncie **SimplySign** jest więcej kart, to numery slotów odpowiadają kolejno zgodnie z listą kart prezentowaną przez aplikację **SimplySign Desktop** – karta „najwyżej” ma numer slotu 0. Kolejna poniżej ma numer slotu 1 itd.

Uwaga!!!

Ze względu na możliwość dodawania i usuwania kart z konta SimplySign, która wpływa na kolejność slotów, przed każdym podpisem zaleca się zweryfikowanie poprawności numeru slotu.

Przykładowa konfiguracja:

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

Utworzenie pliku ścieżki certyfikatu bundle.pem

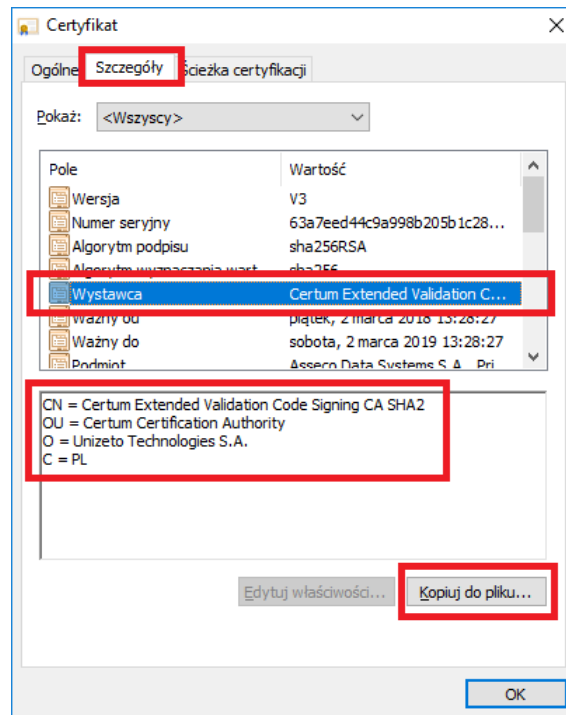
Kolejnym krokiem jest utworzenie pliku ścieżki certyfikatu o rozszerzeniu*.pem (przykład: bundle.pem). Jego zawartość wygląda następująco:

1. „Na górze”: Certyfikat użytkownika
2. „Poniżej”: certyfikat pośredni dla certyfikatu użytkownika

UWAGA. Zawartość pliku bundle.pem musi być koniecznie we wspomnianej wyżej kolejności.

Uzyskiwanie certyfikatu Użytkownika

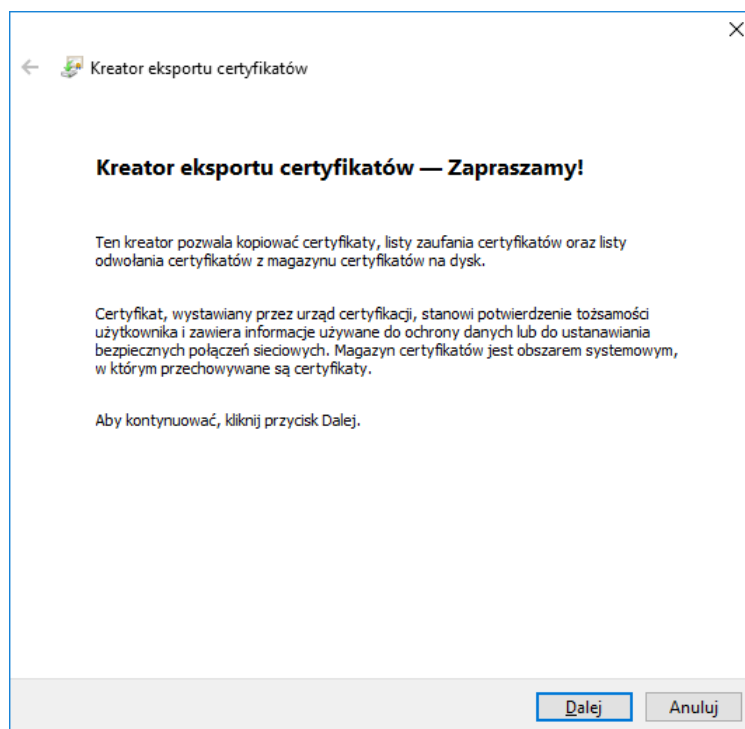
Aby uzyskać certyfikat użytkownika, należy po prostu go wyświetlić i przejść do zakładki **Szczegóły**.



Warto w tym kroku zapisać sobie zawartość pola „Wystawca”. Pomoże to w późniejszym doborze certyfikatu pośredniego.

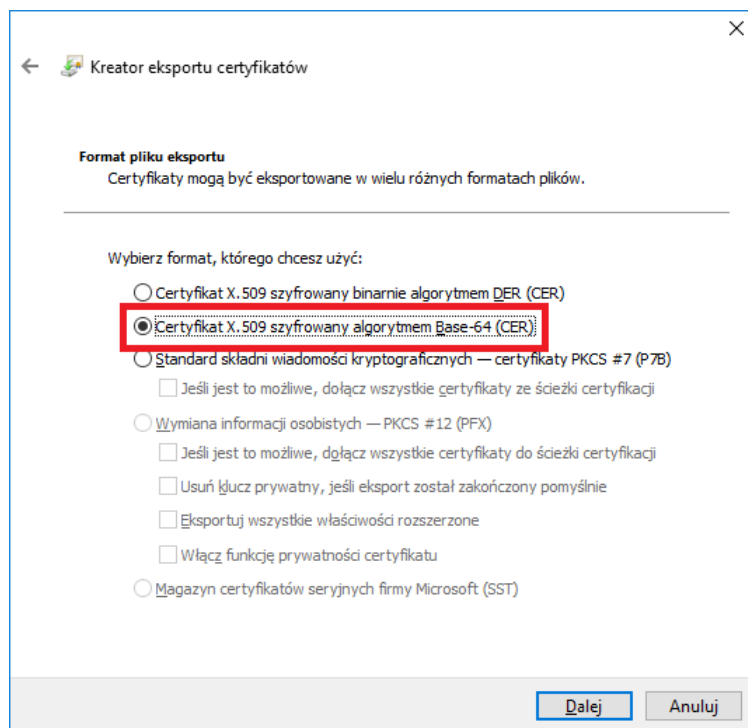
Rysunek 11: Szczegóły certyfikatu

Następnie należy nacisnąć przycisk **Kopuj do pliku**. Uruchomiony zostanie kreator eksportu certyfikatu.



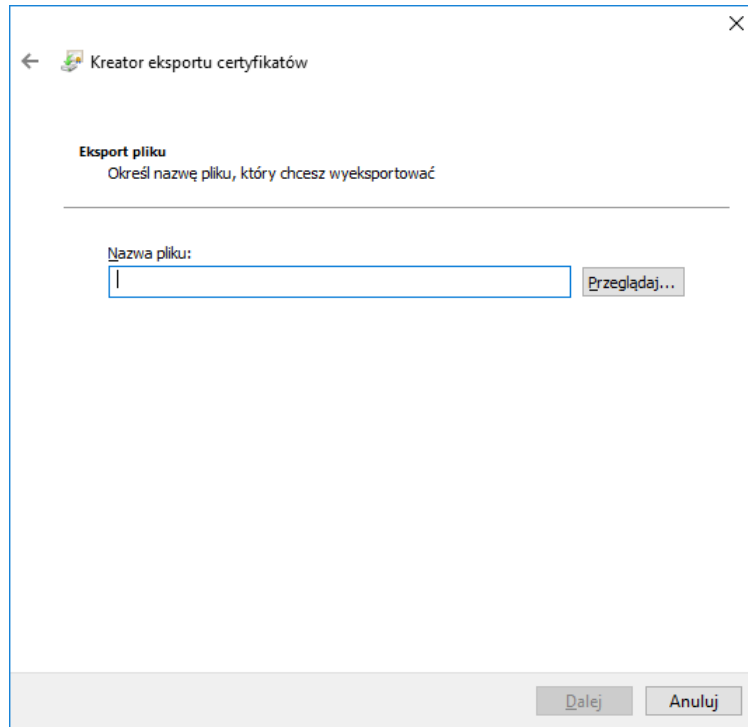
Rysunek 12: kreator eksportu certyfikatu

Następnie należy nacisnąć przycisk **Dalej**. Wyświetlone zostanie okno umożliwiające wybranie formatu w jakim wyeksportowany zostanie certyfikat.



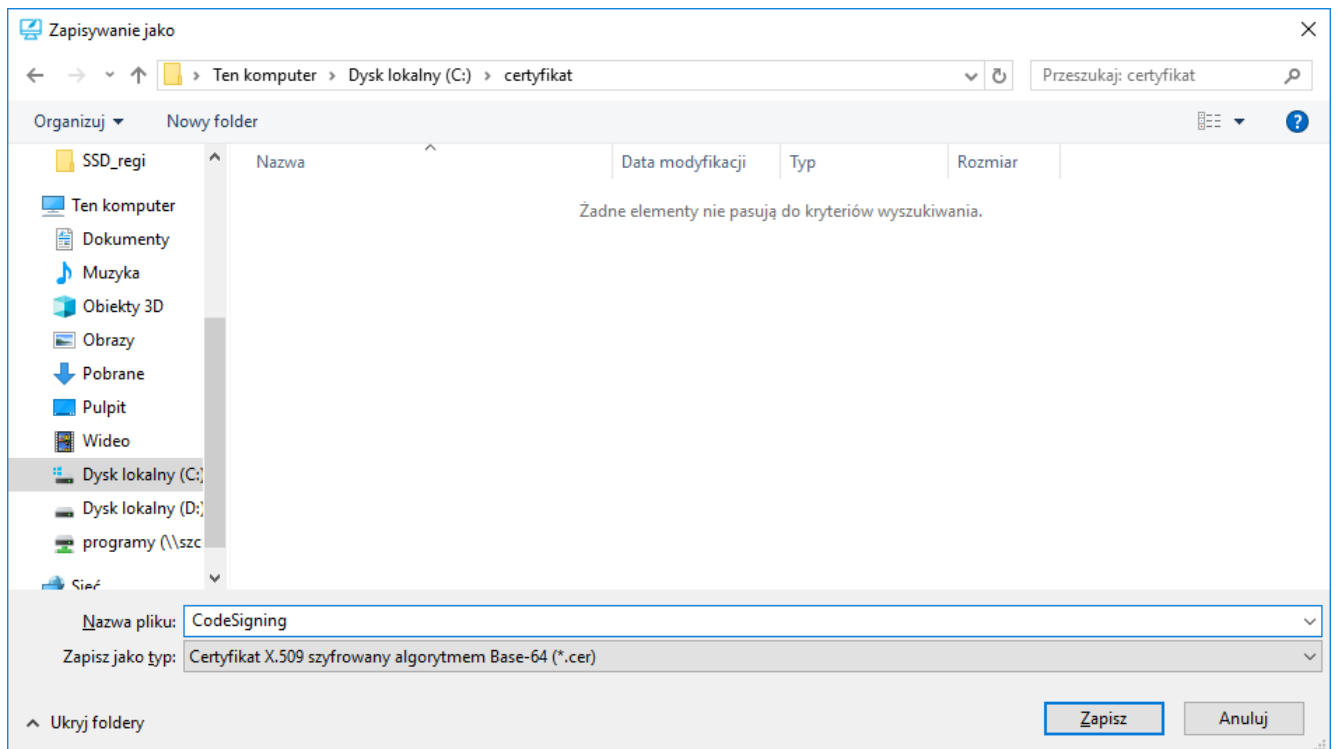
Rysunek 13: kreator eksportu certyfikatu – wybór formatu certyfikatu

Należy wybrać format Base-64 i nacisnąć przycisk **Dalej**. Wyświetlone zostanie okno umożliwiające zdefiniowanie położenia wyeksportowanego pliku certyfikatu.



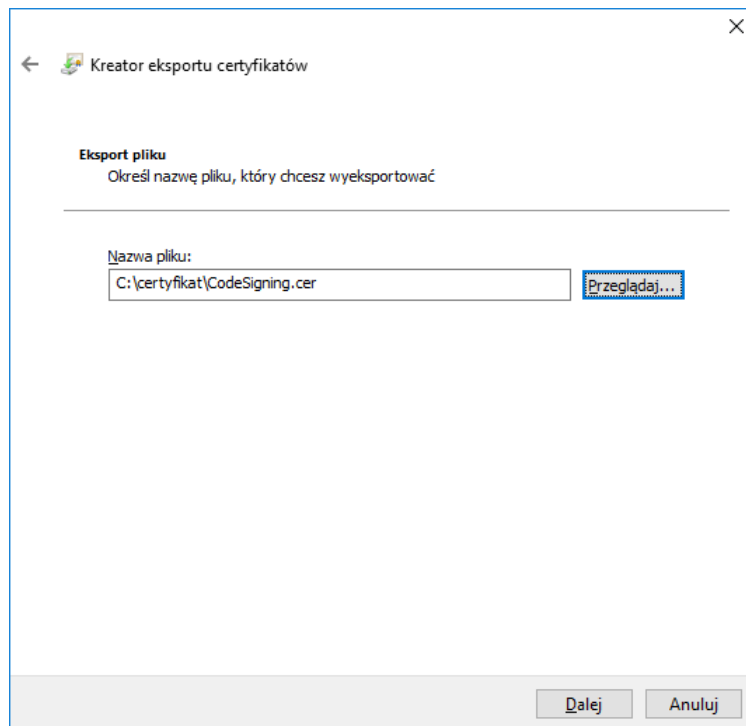
Rysunek 14: kreator eksportu certyfikatu – wskazanie ścieżki

Należy nacisnąć przycisk **Przełączaj**. Wyświetlone zostanie okno umożliwiające nadanie nazwy eksportowanemu plikowi certyfikatu.



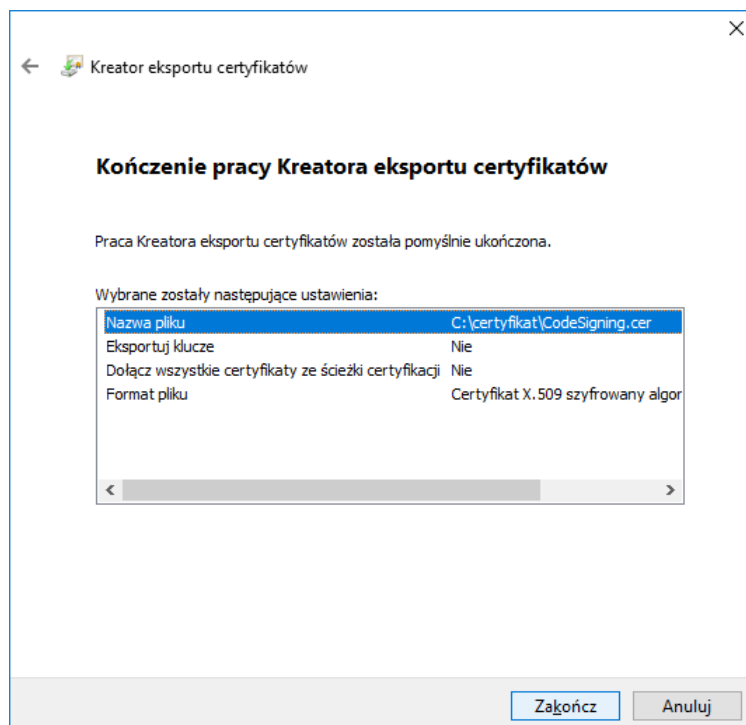
Rysunek 15: kreator eksportu certyfikatu – wskazanie nazwy pliku

Po wskazaniu folderu docelowego i zdefiniowaniu nazwy pliku należy nacisnąć przycisk **Zapisz**. Nastąpi powrót do kreatora eksportu. Wskazana ścieżka będzie widoczna w kreatorze.



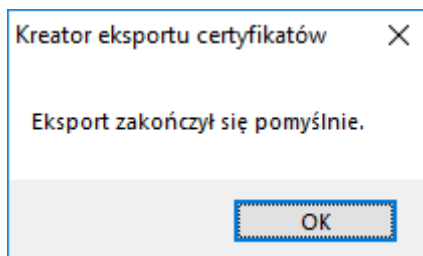
Rysunek 16: kreator eksportu certyfikatu – zdefiniowana lokalizacja certyfikatu

Należy nacisnąć przycisk **Dalej**. Wyświetlone zostanie ostatnie okno kreatora eksportu.



Rysunek 17: kreator eksportu certyfikatu – okno końcowe

Należy nacisnąć przycisk **Zakończ**. Certyfikat zostanie wyeksportowany do pliku i wyświetlony zostanie stosowny komunikat.



Rysunek 18: kreator eksportu certyfikatu – informacja o poprawnym wyeksportowaniu certyfikatu

Uzyskiwanie certyfikatu pośredniego

Certyfikaty pośrednie należy pobierać ze strony Certum:

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

W doborze odpowiedniego certyfikatu (certyfikatów) pośrednich pomoże zapisana wcześniej nazwa Wystawcy z pola „Wystawca” certyfikatu użytkownika. Należy odszukać na stronie Certum wystawcę swojego certyfikatu i zapisać jego certyfikat w formacie tekstowym PEM.

Następnie mając dwa pliki z certyfikatami, należy utworzyć nowy plik tekstowy. Zawartość obu uzyskanych wcześniej plików (Certyfikat użytkownika oraz certyfikat pośredni) należy wkleić do jednego pliku tekstowego we wspomnianej wyżej kolejności:

1. „Na górze”: Certyfikat użytkownika
2. „Poniżej”: certyfikat pośredni dla certyfikatu użytkownika

Plik należy zapisać i zmienić jego rozszerzenie na *.pem.

```

1 -----BEGIN CERTIFICATE-----
2 MIIIGODCCBSCGzAwIBAgIQY6fulEYamYsgWxwoUmlz1zANBgkqhkiG9w0BAQsFADCB
3 IDELMAKGA1UEBMCUEwxiAgBgNVBAAcMGVvuaXpldG8gVGVjaG5vbG9naWVzIFZlbnVz
4 QS4xJzAlBgNVBAsMHkNlcnR1bSBkZlZ0aWZpY2F0aW9uIEFlZGhvcml0eTE4MDYy
5 A1UEAwVQ2VydHVtIEV4dGVuZGVkIFZhbG1kYXRpb24gQ29kZSBTaWduaW5nIENB
6 IFNIQTlWbHcNMTgwMzAyMTIyODI3WmcNMTkwMzAyMTIyODI3WjCCAS4xZCzAJBgNV
7 BAYTAlBMMSEWwHYDQQKDBHc3N1Y28gRGF0YSBTeXN0ZW1zIFMuQS4xZDjAMBGNV
8 BAsMBVBMQ1aMQ9wDQYDZDQWZDZDZDZDZDZDZDZDZDZDZDZDZDZDZDZDZDZDZDZDZD
9 MBIGALUECQW9kb2xza2EgMjExDzANBgNVBETBjgkLTM5MTETMBEGCysGAQQB
10 gjc8AgEDEwJQDEYMBYGCysGAQQBggj8AgEBDAAGZDZDZDZDZDZDZDZDZDZDZDZDZD
11 NzwCAQIMCXBvbw9yc2tp2TETMBEGALUEBRMKMDAwMDQyMTMxMDEGMBEgALUEDXMu
12 UHJpdmF0ZSBPcmhhbml6YXRpb24xITAfBgNVBAMMGFEzc2VjbyBEYXRhIFN5c3R1
13 bXMgUy5BLjCCAS1wDQYJKoZIhvcNAQEBBQADGgEPADCCAQoCggEBAIxp5KLCOAoD
14 nBOOYyTf0zi1WwOcl4h/JYNQntbqE14b0WAUetzy5Nvur9C8v/x9W+FWuBMD5jX
15 83qkHpIgdYwUwBEXypTPKYyVJTECkCe2wi0/zehmGLbwhSzfNF+KBLJGD3CY
16 s/TtQfjjsXdd00V1UzcyVRh4DcbDDJo0wTkl1PjH136zn7Rc/X1KJFroelVfhB4/
17 qqIreLshnot4MoQX/AqY5es304011fIqr/zD2UKczjMjK+DaYuGr7swGFkHeKEL
18 LCGTvwg2fritk/bCCpb0ctYmshhPEXMhIjLis4l24M/uyf2V9w4Uv0bVvnMqbpW
19 eHMSvtvHaqocCAwAAOCaEcgwGjHjMAwGALUdEwEB/wQCMAAwNAYDVR0fBC0wKzAp
20 oCegJYYjaHR0cDovL2Nybc5jZXJ0dW0ucGwvZXZjc2Nhc2hhMi5jcmwwdQYIKwYB
21 BQUHAQEETBnMCOGCCsGAQUFBzABhiFodHRwOi8vZXZjc2Nhc2hhMi5vY3NlLWN1
22 cnR1bS5jb20wNgYIKwYBBQUHMAKGMh0dHA6Ly9yZXZvc210b3J5LmN1cnR1bS5S
23 bc91dmZyZFAzAgEYLnN1cnR1bS5SbG9wZDZkZDZkZDZkZDZkZDZkZDZkZDZkZD
24 FzAdBgNVHQ4EFgQUve47+1H2i/n9Yj9i23q5XhnlNWcWwHYDVR0SBGwFoEUZXXj
25 c2Nhc2hhMkRjZXJ0dW0ucGwvZXZjc2Nhc2hhMkRjZXZjc2Nhc2hhMkRjZXZjc2Nhc2
26 h2E4MAQHWNgYLKRoABy2cdwIFAQcwJzAlBgggrBgEFBQcCARYZaHR0cHM6Ly93d3cu
27 Y2VydHVtLnBsL0NQUzAFBgNVHSUEGDAwBggrBgEFBQcDAwYKKwYBBAGCNz0BATABI
28 BgNVHREEQTA/0D0GCCsGAQUFBwDoDEwLWwtUEwntUE9NT1JTS01FLUdEQCWDU0et
29 QVNTRUNPIERBVEEGU11TVEVNUyBTLkEuMA0GCSqGSIb3DQEBCwUAA4IBAQCvJ604
30 /mvKwA6nGg+Nj+QsIezQ3xIBt9rqJgWhOq36q0NGM8VP7n2WDJXOCUVMVryNIF2N
31 nT4u4ve8vRo/nJDe9/94/OTH4piD2UsFURCeDdaL5GNp/j9UdPsdqQ3/7Bt2k2N
32 Ss+co1Gt32pLGKziAK04ZPt4Q57AUQLPc3oGeI7p1AkLwUqV3N3ZChuGzBBU/pzN
33 1P/R+YYmoRx27S5PhALpz2wU+Omt0A9b55GI5QvFpk0bvpwSXLckRfK66owbenvx
34 MuvBa3JtpWfxh0ITR/MIP2EUNi2YdRkkktSXZPgLxVFO+cRSgGR7jZ8n+IGIJLJ
35 J13yzVVEGcfeFwD3
36 -----END CERTIFICATE-----
37 -----BEGIN CERTIFICATE-----
38 MIIIE3jCCA9qgAwIBAgIQTpbBugY1lgwquidiXpBk0zANBgkqhkiG9w0BAQsFADB+
39 MQswCQYDQQEwJQDEiMCAgALUEChMzVW5pemV0byBUZWNobm9sb2dpZDZkZDZkZDZk
40 LjEnMCAgALUECkMeQ2VydHVtIENlcnRpb24gQXV0aG9yaXR5MSIwIAZD
41 VQDEx1DZkZ0dW0gVHJlc3R1ZCB0ZXR3b3JrIENBMB4XDTE1MTAyOTExNTUzOVc0X
42 DTI3MDExOTExNTUzOVc0XzAlBgggrBgNVBAYTAlBMMSEWwHYDQQKDB1Vbml6ZXRv
43 IFR1Y2hub2xvZ211cyBTLkEuMScwJQYDZDQWZDZDZDZDZDZDZDZDZDZDZDZDZD
44 b1BBDXR0b3JpdHkxODAzBGNVBAAML0N1cnR1bS5SbG9wZDZkZDZkZDZkZDZkZDZk
45 IENvZGVuZ211bnhmluZyBDQSBTSEEyMlIBIjANBgkqhkiG9w0BAQEFAAOCQAQ8AMI
46 CgKCAQEA/B+7+zooyk5XmBvTUKFFa7JrbGDLmN/RDpEiR43hkrx1s3mqA0cmS5EV
47 aBG/+h6bNgbMuhdwZmj6GPxVnthe9SXhe00fWQfFy3I1Sd283z/Sgn1dqzuqMe
48 He9m/MZTbt/xbu2UFhaz8xJvt98tEvEvoNbQg0V1sF97jaxyN1lbt0fhv6wKuz8Q
49 on8wtU7PYLCukCIY5MiEOyRjLE5cP1qhvKwdFvP2jpZ4NSUMUScUW/S1KpGdZID3
50 hLINE3n3dFwEFDuKAlY3vCAKpiv+ppfCIoddJKpu/Of3msX3GtRjVWYDQZBXm6J
51 FVxcPq06X8d2t17W61xlp2mIs0lwwIDAQAB04IBUzCCAUsDwYDVR0TAAQH/BAUw
52 AwEB/zAdBgNVHQ4EFgQUoosUgEXQtuys0RLXjzoF0aMKqZ2RcwHwYDVR0jBBgwFoAU
53 CHbNywF/JpBFz27kLzihdGdfcwDgYDVR0TAAQH/BAQDAgEGMBEMGALUdJQUMMAoG
54 CCsGAQUFBwMDMCAgALUdHwQoMcywKAIoCCGHmh0dHA6Ly9jcmwwY2VydHVtLnBs

```

Rysunek 19: plik bundle

Uzyskanie aliasu certyfikatu

Przed przystąpieniem do podpisywania należy najpierw pozyskać tzw. alias certyfikatu. Służy do tego poniższe polecenie :

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

W rezultacie instrukcja zwraca zawartość magazynu kluczy:

```
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
```

W tym przypadku alias to:

```
63A7EED44C9A998B205B1C2850C973D7
```

Podpisywanie

Aby podpisać plik, w wierszu poleceń (cmd.exe) należy użyć następującego polecenia:

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

- [1] – Adres znacznika czasu. Dla Certum <http://time.certum.pl>,
- [2] – Ścieżka do pliku ścieżki certyfikatu [bundle.pem],
- [3] – wskazanie algorytmu podpisu [SHA1withRSA lub SHA256withRSA],
- [4] – Ścieżka do pliku konfiguracyjnego providera,
- [5] – kod PIN do wirtualnej karty [dla kart bezipnowych należy podać dowolny kod PIN – nie można go pominąć w poleceniu],
- [6] – Ścieżka do pliku podpisywanego,
- [7] – Alias certyfikatu, którym nastąpi podpisanie pliku.

Przykładowe, poprawne polecenie:

```
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"
```

Jeśli operacja podpisu przebiegła prawidłowo, konsola wyświetli następujący wynik:

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

Podpisywanie wsadowe

W celu wsadowego podpisania wielu plików podczas jednej sesji należy utworzyć plik *.bat, zawierający tyle wpisów, ile plików ma zostać podpisane podczas jednego procesu podpisu. Działanie takie eliminuje konieczność każdorazowego wywoływania komendy w konsoli oraz wpisywania kodu PIN przy podpisie kolejnych plików.

W celu utworzenia pliku, należy utworzyć nowy plik tekstowy *.txt, wkleić wpisy do podpisywania plików, zapisać plik oraz zmienić jego rozszerzenie z *.txt na *.bat.

Poniższy przykład prezentuje zawartość pliku *.bat dla podpisu trzech aplikacji jednocześnie:

```
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"
```

Tak zapisany plik można uruchomić w konsoli cmd.exe lub dwuklikiem, a rezultatem będzie rozpoczęcie podpisywania kolejnych plików, zawartych w pliku *.bat.

Rezultatem uruchomienia pliku *.bat w konsoli będzie informacja o kolejnym wywołaniu komend i podpisie plików:

```
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.
```

```
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.
```

Weryfikacja pliku narzędziem jarsigner

Weryfikacja podpisanego pliku, przy użyciu narzędzia jarsigner odbywa się następującym poleceniem:

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

[1] – Ścieżka do pliku podpisywanego,

Przykładowe, poprawne polecenie:

```
jarsigner -verify "plik.jar"
```

W przypadku poprawnej weryfikacji pliku konsola wyświetli:

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

W przypadku braku podpisu wynik jest następujący:

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