securosys

SWISS SECURITY TECHNOLOGIES
FOR COMMUNICATIONS SYSTEMS

Securing Microsoft SignTool

Using Primus Hardware Security Module
Application Note

Primus HSM Integration Guide for
Microsoft Windows Server 2016/2012R2 and Windows 10 Clients

Securosys SA, Förrlibuckstrasse 70
CH-8005 Zürich, Switzerland
Tel. +41 44 552 31 00 • www.securosys.ch
info@securosys.ch

Microsoft Partner

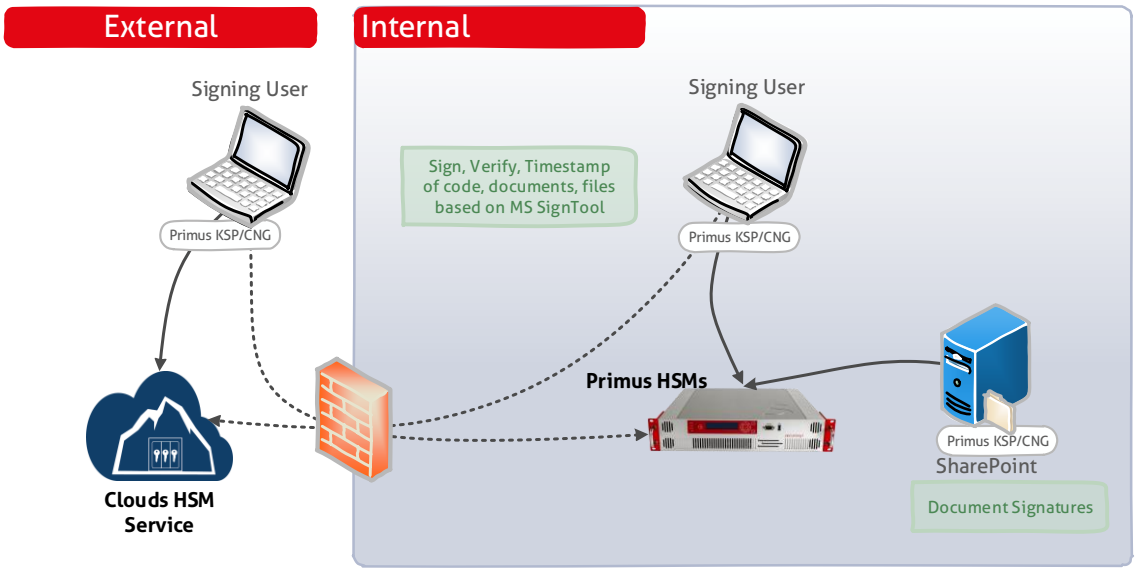
securosys

Table of Contents

1 Introduction

This document describes how to secure private keys used by the Microsoft SignTool by using the Securosys Primus HSM or Securosys Clouds HSM service.

The Microsoft SignTool is a command-line CryptoAPI tool, used to digitally-sign, verify or timestamp files, codes, libraries and documents. It is part of the Microsoft Windows Software Development Kit (SDK) or of the Microsoft Visual Studio developer environment.



The Primus HSMs are built to securely generate and store true random cryptographic keys, providing a central, certified secure storage. They also control and regulate access to the keys and the related cryptographic functionality. The Primus HSM combined with SignTool meets or exceeds the best practice security requirements and is one step ahead of fulfilling your compliance demands by providing:

- Hardware-based secure generation of true random cryptographic keys
- Central and highly secure storage of cryptographic keys
- Load balancing and fail-over by clustering the HSMs
- Controlled and regulated access to the keys
- Hardware acceleration of cryptographic operations such as encryption, authentication, and digital signatures, relieving the host server of processor intensive computations
- Scalable performance at manageable cost

All certificate issuance and validation processes occur within the protected confines of the HSM. Private keys are never accessible outside the HSM.

The Primus HSM can easily be integrated in a Microsoft Windows system by installing the Primus CNG Provider. This enables all Windows servers and clients to generate and store their private keys and certificates securely in the HSMs, and perform all related cryptographic functionality, like signing or certificate validation, hardware accelerated on the Primus HSM.

1.1 Requirements

- Windows operating system with Crypto API Next Generation (CNG)
  (Server 2008 or higher, Windows Vista or higher).
- Windows integrated tools (certreq.exe, certutil.exe)
- Microsoft Signtool.exe (part of Microsoft Windows SDK or Microsoft Visual Studio)
- Microsoft .NET framework 4
- Securosys Primus HSM or Clouds HSM with CNG/KSP Provider

1.1.1 Software and Hardware Used for this Setup

- Windows Server 2016/x64, Windows 10/x64 with latest patches installed.
- Virtualization Software (VMWare Workstation 14.1.1)
- Microsoft Signtool 10.0
- Securosys Primus-X HSM V2.5.1
- Securosys KSP Provider V1.21.1

1.2 More Information...

For more information about HSM administration, refer to the Primus HSM User Guide or contact Securosys support.

For more information about OS support, contact your Microsoft sales representative or integration partner.

The following references provide further information:

- Search on the web "Introduction to Code Signing" for information about why signing files is important
- https://docs.microsoft.com/en-us/windows-hardware/drivers/devtest/signtool
- https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-R2-and-2012/dn296456%28v%3dws.11%29
- See also "Secure Boot Key Generation and Signing Using HSM (Example)" from Microsoft: https://docs.microsoft.com/en-us/windows-hardware/manufacture/desktop/secure-boot-key-generation-and-signing-using-hsm--example

2 Procedures

This procedure provides a straightforward integration process, which has been tested. Please take notice that there may be other ways to achieve interoperability. This guide assumes that you are familiar with the Primus HSM and the Microsoft Windows SDK signing tool and does not cover every step of the hardware and software setup process.

Note: Throughout this guide, the term HSM refers to Securosys Primus HSM products.

2.1 Setup the HSM

Setting-up the Primus HSM hardware or your Clouds HSM partition is not part of this application note. Please refer to the corresponding Quick Start Guides and User Manuals.

2.1.1 Installing the Primus HSM KSP/CNG Provider

Download and install the latest KSP/CNG provider from the Securosys support portal (e.g. SecurosysPrimusKsp_V1.21.1.zip). Configure and test the Primus or Clouds HSM connections within the Securosys Key Storage Provider.

For further details regarding installation and configuration of the CNG provider consult the application note "PrimusHSM_CNGInstallation_AN-E01".

2.2 Install a New Signing Key and Certificate

2.2.1 Prepare the Signing Key Request File

Prepare the input file for the certreq.exe tool (sample request.inf file):

```
[Version]
Signature= "$Windows NT$"
[NewRequest]
ValidityPeriod = Years
ValidityPeriodUnits = 1
Subject = "CN=signtooldemo.hsmdemo.test"
FriendlyName = "Signtool Demo Key"
MachineKeySet = True
RequestType = Cert
Exportable = FALSE
HashAlgorithm = SHA256
KeyAlgorithm = RSA
KeyLength = 4096
KeyUsage = 0xf0
ProviderName = "Securosys Primus HSM Key Storage Provider"
KeyContainer = "signtooldemokey"
;[EnhancedKeyUsageExtension]
;OID=1.3.6.1.5.5.7.3.1
;[RequestAttributes]
;CertificateTemplateName = <yourADCSTemplate>
```

Update the following values:

- **Subject**, replace with your real data
- **ValidityPeriod, ValidityPeriodUnits** – replace according your requirements
- **KeyContainer** – define your key container ID or remove the line for default values
- Adapt the algorithms according to your requirements
- Define additional extensions and CertificateTemplateName if required
- **RequestType** – adapt e.g. if you want to use a public signed certificate (PKCS10)

2.2.2 Generate the Signing Key and Self-Signed Certificate

```
certreq -new request.inf SigningCertificate.cer
Installed Certificate:
  Serial Number: 1aaa8bbcd7c98cb74c9dff9f9e40cf10
  Subject: CN=signtooldemo.hsmdemo.test
  NotBefore: 14/02/2018 16:21
  NotAfter: 14/02/2019 16:31
  Thumbprint: cb1a55f6ab8ccacedab3fcd9de48bd69be16b88d
  Friendly Name: Signtool Demo Key
  Securosys Primus HSM Key Storage Provider
  signtooldemokey


CertReq: Certificate Created and Installed
```

2.2.3 Generate the Signing Key and Public Signed Certificate

When a public signed certificate is required, adapt the *RequestType* to "PKCS10" and generate the Certificate Signing Request (CSR) to be signed by a public Certification Authority (CA):

```
certreq -new request.inf SigningCertificate.csr
```

To make the public signed certificate available for use, execute the following command:

```
certreq.exe –accept -machine <certificateFilename.cer>
```

Where <certificateFilename.cer> is the binary signed certificate received from the CA/SubCA.

2.2.4 Validate the Certificate

Verify your signing certificate:

```
certutil -store -v my "<Certificate_serial_number_or_thumbprint>"
```

Example of the self-signed certificate:

```
certutil -store -v my "1aaa8bbcd7c98cb74c9dff9f9e40cf10"
my "Personal"
================ Certificate 0 ================
X509 Certificate:
Version: 3
Serial Number: 1aaa8bbcd7c98cb74c9dff9f9e40cf10
Signature Algorithm:
    Algorithm ObjectId: 1.2.840.113549.1.1.11 sha256RSA
    Algorithm Parameters:
    05 00
Issuer:
    CN=signtooldemo.hsmdemo.test
  Name Hash(sha1): e0f35912a4d53f78e8443f219d77c57b79c65759
  Name Hash(md5): e066872e7bef87abba6bb1f1a4512d94

 NotBefore: 14/02/2018 16:21
 NotAfter: 14/02/2019 16:31

Subject:
    CN=signtooldemo.hsmdemo.test
  Name Hash(sha1): e0f35912a4d53f78e8443f219d77c57b79c65759
  Name Hash(md5): e066872e7bef87abba6bb1f1a4512d94

Public Key Algorithm:
…
```

2.3 Sign your Files with Signtool.exe

The signtool.exe is used to sign or timestamp codes, certificates, etc. Depending on the OS version the tool may be located in different folders (e.g. Windows 10/x64: C:\Program Files (x86)\Windows Kits\10\bin\x64\signtool.exe).

Example – with certificate reference by subject name:

```
signtool.exe sign /v /fd sha256 /sm /n "signtooldemo.hsmdemo.test" ApplicationToSign.exe

The following certificate was selected:
    Issued to: signtooldemo.hsmdemo.test
    Issued by: signtooldemo.hsmdemo.test
    Expires:   Thu Feb 14 16:31:30 2019
    SHA1 hash: CB1A55F6AB8CCACEDAB3FCD9DE48BD69BE16B88D

Done Adding Additional Store
Successfully signed: ApplicationToSign.exe

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

Example – with certificate reference by SHA1 hash:

```
signtool.exe sign /v /fd sha256 /sm /sha1 "cb1a55f6ab8ccacedab3fcd9de48bd69be16b88d" ApplicationToSign.exe
```

Take care, that the certificate references are unique, otherwise the sign process could fail (subject name, or sha1 value of the certificate). Referencing the certificate by CSP (/csp) and key container (/kc) seems to fail (reason yet not known).

Refer to signtool.exe help and Internet for the bunch of signing and verification options.

2.3.1 Verify the Signed File

The signature can be verified either on command-line or using the file properties dialog (Digital Signatures):

```
signtool.exe verify /pa /v ApplicationToSign.exe
Verifying: ApplicationToSign.exe
Signature Index: 0 (Primary Signature)
Hash of file (sha256):
02B5134F2AB32A9638284AC016BDFA528F963016D17885F9C00
9167108A1BC85

Signing Certificate Chain:
    Issued to: signtooldemo.hsmdemo.test
    Issued by: signtooldemo.hsmdemo.test
    Expires:   Thu Feb 14 16:31:30 2019
    SHA1 hash:
CB1A55F6AB8CCACEDAB3FCD9DE48BD69BE16B88D

File is not timestamped.

Successfully verified: ApplicationToSign.exe

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