Library/Plugin Signing Setup

Revision History

Version	Date	By	Comments
1	08/01/10	Brian Graversen	Initial release of document

Introduction

When a client application makes use of the SignOn Library, it is important that it verifies the integrity of the library before using it. The end-users credentials (username, password) is supplied to the SignOn Library, so to prevent a malicious 3rd party from injecting a modified plugin or library dll, all components in the SignOn Library framework has been signed.

This document describe the proposed setup.

A self-signed certificate, issued by SDSD

To avoid unnecessary complexity, we propose that a self-signed certificate, with a lifespan of 20 years is used to sign both the library and the plugins. The certificate should be issued by SDSD (or an operator assigned by SDSD) and the private part should be stored securely. The public certificate should be bundled with the SignOn Library, so 3rd party plugin developers and client developers has easy access to the trust anchor.

How to create a self-signed certificate

Creating a self-signed certificate can be done by various tools, like OpenSSL or Java Keytool. Below we will outline the steps needed to use Java Keytool, but other tools can be used instead.

Start by generating the keystore

```
$> keytool -genkey -alias signer -keyalg RSA -validity 7305 -keystore keystore.pfx
           -storetype PKCS12 -dname "CN=SignOn Library Signing Certificate, O=SDSD, C=DK"
```

The above command will prompt for a password. This password is reused throughout the steps below, and has to be entered each time a DLL is signed – the password should be stored securely.

The certificate generated will be valid for 20 years.

Validate the keystore

```
$> keytool -list -v -storetype PKCS12 -keystore keystore.pfx
```

The output from the above command displays the content of the keystore – check that the certificate contains the data that was requested.

Extract the public certificate

```
$> keytool -export -alias signer -keystore keystore.pfx -rfc -file root.crt -storetype PKCS12
```

The root.crt file outputted from this command should be bundled with the SignOn Library, the pfx file should be stored securely by the signing operator.

Signing DLLs

Signing a DLL requires SignTool.exe, a program by Microsoft, which allows a certificate holder to sign binaries using Microsofts Authenticode scheme. The following command can be used to sign a binary (DLL).

```
$> SignTool.exe sign /f [KEYSTORE] /p [PASSWORD] [BINARY]
```

fx: `SignTool.exe sign /f keystore.pfox /p Test1234 plugin.dll`

The SignTool binary is part of the SignOn Library bundle (and can be downloaded from Microsoft)