DIGITAL SUNDHED

SAMMENHÆNGENDE DIGITAL SUNDHED I DANMARK

Deployment Notes

Revision History

Version	Date	By	Comments
1	06/01/10	Brian Graversen	Initial release of document
2	01-20-10	Søren Mors	Added notes about the java wrapper

Introduction

The target audience for this document is vendors that wants to use the SignOn Library in their respective clients. This document will describe the steps necessary to use the library, and deploy the DLL's to the end-users.

Files to deploy

The SignOnLibrary framework uses the SignOnLibrary.dll assembly file as the entrypoint both for COM and .NET access. Accessing the library from Java clients requires an additional JAR file, that wraps the functionality in the SignOnLibrary.dll file.

Besides the framework DLL, various plugins will have to be deployed as well. In the examples below, we will assume that the vendor needs to deploy the SOSI GW plugin and the CAPI PKI Plugin – deploying other plugins will likely be a similar experience, but the documentation bundled with that 3rd party plugin should be consulted before deploying.

Framework location
The framework file(s), namely SignOnLibrary.dll (and SignOnLibrary.jar for Java clients) are bundled and deployed with the vendors application, and should merely be available on the classpath so the vendors application can find the file. A simple approach would be to place the SignOnLibrary.dll in the same folder as the vendor application.

Plugin location
Plugins can be installed either together with vendor application, and just referenced using a relative path in the configuration (example below), or the plugins can be installed in the Global Assembly Cache (GAC) on the end-users machine. Installing the plugins in the GAC is slightly more cumbersome (without adding any real benifit), and for the rest of the document, we will assume that the plugins are installed together with the vendor application.

The only difference from the sample configuration below, and the GAC installation is the AddPlugin() method call, where the plugin is simple referenced by name, instead of the relative path to the DLL.

Assuming the following file layout for a simple vendor application, where the vendor application is represented by a single EXE file:

```
|-- VendorApp.exe
|-- SignOnLibrary.dll
`-- Plugins
    |-- GWPlugin.dll
    `-- CAPIPKIPlugin.dll
```

Then the following code samples are enough to reference the plugins correctly (the path to the plugins is relative to the location of the vendor application – VendorApp.exe).

The sole goal of the code samples below are to highlight how plugins are referenced in the code, relative to where the files are located on the harddrive. The code does not contain error handling, nor does it work without adding more calls to the framework...

C# Code Sample

```
static void Main(string[] args)
{
     SignOnRequest request = new SignOnRequest();

     request.AddPlugin(@"Plugins\CAPIPKIPlugin.dll", -1, false);
     request.AddPlugin(@"Plugins\GWPlugin.dll", -1, false);
}
```

Visual C (COM) Code Sample

```
int main(int argc, char **argv)
{
     SignOnLibrary::ISignOnRequestPtr
           pRequest(__uuidof(SignOnLibrary::SignOnRequest));

     BSTR capiDll = SysAllocString(L"Plugins\\CAPIPKIPlugin.dll");
     BSTR gwDll = SysAllocString(L"Plugins\\GWPlugin.dll");

     pRequest->AddPlugin(capiDll, -1, false);
     pRequest->AddPlugin(gwDll, -1, false);

     SysFreeString(capiDll);
     SysFreeString(gwDll);
}
```

Special COM notes

Deploying the SignOn Library with a .NET vendor application does not require any additional registration on the end-users machine, since the vendor application accesses the SignOnLibrary.dll file directly. For COM application this is not the case, since all calls goes through COM. Here we need to register the SignOnLibrary.dll as a COM component before it can be used on the end-users machine. Ordinarily regsvr32.exe is used to register DLL's, but for .NET components, the regasm.exe file is used instead. For manual testing, the regasm.exe file is located at

```
C:\Windows\Microsoft.NET\Framework\v2.0.50727\regasm.exe
```

where "C:\Windows" is the location of the Windows installation on the end-users machine.

Before the SignOn Library component has been registered, the vendor application will not be able to call the SignOn Library through COM. Most modern MSI tools can be configured to automatically register the SignOnLibrary.dll file using regasm.exe, and the documentation for the MSI tool used by the vendor should be consulted for the exact details.

Special Java notes

The Java wrapper around the SignOn Library uses the COM interface to access the SignOn Library. The notes about the COM interface above therefore also applies to the java wrapper. The wrapper uses a native .dll containing JNI functionality to access the COM interface.

The SignOnJNI.dll file must be placed in the same directory as the SignOnWrapper.jar file, since

the java code looks for it there. There are no other restrictions on how to deploy the java library.