Signing Process

An executable for any application must be "signed" before it can be executed by anyone who does not have an NX Open Author license. The signing process is typically performed when an application is distributed to the user base and is used to verify that the application has been developed using a valid NX Open Author license.

The following two steps define the general signing process.

1. A resource file must be added to the source files. The resource must be compiled and linked with the executable. This step is not required for Java applications.
2. Run the signing utility to add an encrypted string to the executable.

When running without an NX Open Author license, NX checks for this encrypted string when the application is loaded. If NX does not find an NX Open Author license or signature, it will not load the executable. If you are running a batch file, the Common API will fail to initialize.

- The signing utility may only be executed if an NX Open Author license is available.
- The signing utility also provides a verify option that displays a message confirming whether the file has been correctly signed or not.
- Running the signing utility multiple times on the same executable does no harm. The results are the same as running it once.
- Journals and GRIP programs do not need to be signed. All other NX Open applications must be signed.

For programs run outside of NX, it is allowable for an unsigned application to call into a signed application. For example, if you are running a Java program outside of NX, an unsigned jar file could call into a signed jar file. For programs run inside of NX, the starting point must be signed.

The following indicates when the signature check or license check happens for NX Open programs:

.Net	Signature check happens
Run inside of NX	When the first session object is obtained
Run outside of NX	When the first session object is obtained

C++	License check happens
Run inside of NX	When the shared object is loaded by NX
Run outside of NX	When the first session object is obtained.

Java	License check happens
Run inside of NX	When the class or jar file is loaded by NX
Run outside of NX	When the first session object is obtained

Signing Process - Language Specific Details

NX Open for C++

NX Open for .NET

NX Open for Java

NX Open for C++

The C++ resource file and signing utility are found in *<NX install directory>\UGOPEN*

Resource File	NXSigningResource.cpp
Signing Utility	signcpp

Note NXSigningResource.cpp does not require a C++ compiler. You may need to change the file extension to match the requirements of your compiler.

To *embed* the resource file compile and link it with the executable.

To *sign* an executable run *signcpp* at a command line prompt and provide the name of the executable. For example:

signcpp myApplication.exe

To *verify* that an executable has been signed use the *-verify* option. For example:

signcpp -verify myApplication.exe

Valid file extensions are: dll, so, sl and exe

NX Open for .NET

The .NET resource file is found in <*NX install directory*>*UGOPEN*\

The .NET signing utility is found in <*NX install directory*>*UGII*\

Resource File	NXSigningResource.res
Signing Utility	SignDotNet

To *embed* the resource file from a command line use the /resource compiler switch. For example:

VB Example

vbc /libpath:<NX install dir>\UGII\managed /t:library /r:NXOpen.dll /r:NXOpen.Utilities.dll /resource:<NX install dir>\UGOPEN\NXSigningResource.res myApplication.vb.

C# Example

csc /resource:<NX install dir>\UGOPEN\NXSigningResource.res /t:library myApplication.cs

For *Visual Studio* add NXSigningResouce.res as a resource to the project and set the Compile property on resource to: Embedded Resource.

To *sign* an executable run *SignDotNet* at a command line prompt and provide the name of the executable. For example:

SignDotNet myApplication.dll

To *verify* that an executable has been signed use the *-verify* option. For example:

SignDotNet -verify myApplication.dll

Valid file extensions are: dll and .exe

NX Open for Java

The Java signing utility is found in <*NX install directory*>*UGOPEN*\

Resource File	not required
Signing Utility	SignJar

To sign a java application, a jar file must be used. Class files cannot be signed. This means that class files cannot be distributed to users who do not have NX Open Author licenses.

No resource file is required for Java applications.

To *sign* a jar file run *SignJar* at a command line prompt and provide the name of the jar file. For example:

SignJar myApplication.jar

To *verify* that a jar file has been signed use the *-verify* option. For example:

SignJar –verify myApplication.jar

Valid file extensions are: jar