Nexaweb アプリケーションへの署名

日本ネクサウェブ株式会社

免責事項

掲載されている資料の著作権は、日本ネクサウェブ株式会社に帰属しており、目的の如何を問わず、コンテンツの無断複製、無断転載、その他二次利用行為は国内及び国外の著作権法により禁止されています。当資料に記載されている情報は、告知なしに変更されることがあります。

資料内に記載されたソースコードはすべてライセンス下でのみ利用可能であり、その条項の範囲において利用を許諾いたします。日本ネクサウェブ株式会社は、資料に記載されたサブシステムやアプリケーションを含む適用可能である全ての情報に関して著作権を保有します。

日本ネクサウェブ株式会社は、資料内に記載された製品の操作方法や使用方法に関して、一切責任を負わないものとします。また、如何なる場合に於いても、間接的、特殊、偶発的、または必然的な損害(業務上の損害、利益の損失、その他を含む) に関して、たとえ弊社もしくはその利用者がそのような損害が生ずる可能性を認識していたとしても、契約の不履行、不法行為(過失を含む)、無過失責任、製造責任またはその他の原因に基づくと否とを問わず、一切の責任を負わないものとします。

日本ネクサウェブ株式会社

目次

1 Nexaweb アプリケーションへの署名

Nexaweb アプリケーションは Java Applet として提供されており、Java Applet と同様に、通常はサンドボックスセキュリティ制限が課せられています。
これらのセキュリティ制限は、アプリケーションに署名を行う事で解除できますが、Nexaweb アプリケーションでは、ユーザーが作成したプログラム以外にも Nexaweb Platform が提供する各種 Jar ファイルが存在しています。
本ドキュメントでは、署名対象となる Jar ファイル及び、署名時のコンフィグレーションについて説明します。

1.1 Nexaweb Platform が提供する Jar ファイル群

Nexaweb Platform は利用する機能、クライアント側で使用される JVM のバージョンによって複数の Jar ファイルが提供されています。
署名を行ったアプリケーションを正常に動作させるためには、実行時に使用されるプログラムファイル群全てに対して署名されている必要があるため、アプリケーション毎に使用する機能、想定される JVM のバージョンを明確にし、使用される全ての Jar ファイルに署名する必要があります。

1.1.1 クライアント側JVMのバージョンと種類によって異なる使用Jarファイル

各 Jar ファイルはクライアント側の JVM のバージョンと種類によって使用される Jar ファイルが異なる場合があります。JVM バージョンによって使用される Jar ファイルが異なるケースには以下のケースが存在します。

- JVM 1.4 以下(1.4 含む)が使用されている場合
  拡張子「.jar」ファイルが使用されます。
- JVM 1.5 以上が使用されている場合
  拡張子「.jar.pack.gz」ファイルが存在している場合、拡張子「.jar.pack.gz」ファイルが使用されます。
  拡張子「.jar.pack.gz」のファイルは、JVM 1.5 から利用可能になった pack200 圧縮を使用した Jar ファイルです。拡張子「.jar.pack.gz」ファイルが存在しなかった場合は JVM 1.4 以下と同じ Jar ファイルが使用されます。
- MSJVM が使用されている場合
  拡張子「.cab」ファイルが使用されます。

これらの Jar ファイルはクライアントからのアクセス時にクライアント JVM バージョン・種類情報に基づいて Nexaweb によって自動的に使用するファイルが切り替えられます。

1.1.2 コンフィグレーションによって異なる使用Jarファイル

Nexaweb ではテスト時に、 Test API のサポートや、デバッグ情報を含んだテスト用の Jar ファイルを使用することができます。
これは、nexaweb-client.xml によって設定されており、以下の設定項目を有効(true)に設定した場合に使用されます。

/client-app/ launch-configuration/ ui-test

この設定が有効になっている場合、Nexaweb はデバッグバージョンを持つ Jar ファイルとして、拡張子を除くファイル名の末尾に「-UITesting」が付加されている Jar ファイルを使用します。
例.) 「NexawebClient.jar」の場合「NexawebClient-UITesting.jar」が使用されます。
この際、「1.1.1 クライアント側 JVM のバージョンと種類によって異なる使用 Jar ファイル」の JVM バージョン(JVM 1.4 以下、JVM 1.5 以上)による使用 Jar ファイルの違いは生じず、常に「-UITesting」が付加されている拡張子「.jar」ファイルが使用されます。
MSJVM を使用している場合は、拡張子「.cab」のファイルに「-UITesting」が付加されているファイルが使用されます。

2 署名対象のファイル群

この章では Nexaweb Platform が提供する全ての Jar ファイルをリストアップします。
コンフィグレーション等によって使用されるファイル、使用されないファイルが別れていますが、条件に応じて必要なファイル群に対して署名を行ってください。

2.1 detect.jar

起動時に使用される jar ファイルになります。
初期起動時にロード画面を表示しています。

ファイル格納フォルダ	/WEB-INF/Nexaweb/client/lib/
使用条件	必ず使用されます。
JVM 1.4 以下	detect.jar
JVM 1.5 以上	提供されません(JVM 1.4 以下と同じものが使用されます)。
MSJVM	detect.cab
UITesting ファイル	detect-UITesting.jar, detect-UITesting.cab(MSJVM の場合)

2.2 NexawebClient.jar

Nexaweb で使用しているメインの jar ファイルになります。
Nexaweb の主要機能の全てがこのファイル内に納められています。

ファイル格納フォルダ	/WEB-INF/Nexaweb/client/lib/
使用条件	必ず使用されます。
JVM 1.4 以下	NexawebClient.jar
JVM 1.5 以上	NexawebClient.jar.pack.gz
MSJVM	NexawebClient.cab
UITesting ファイル	NexawebClient-UITesting.jar, NexawebClient-UITesting.cab(MSVJM の場合)

2.3 NexawebClient-Common-Resources.jar

リソースファイルが含まれている jar ファイルで、通常は Nexaweb がリソース要求時に動的にダウンロードされます。

ファイル格納フォルダ	/WEB-INF/Nexaweb/client/lib/
使用条件	必要に応じて動的に使用されます。
JVM 1.4 以下	NexawebClient-Common-Resources.jar
JVM 1.5 以上	提供されません(JVM 1.4 以下と同じものが使用されます)。
MSJVM	提供されません(JVM 1.4 以下と同じものが使用されます)。
UITesting ファイル	提供されません。

2.4 NexawebClient-OnDemand.jar

SVG もしくはチャート機能を使用している場合に動的にダウンロードされるファイル群を格納している jar ファイルになります。

SVG もしくはチャート機能を使用しているのであれば、このファイルに対しても署名が必要になります。

ファイル格納フォルダ	/WEB-INF/Nexaweb/client/lib/
使用条件	SVG 機能、チャート機能を使用している場合に使用されます。通常は動的に必要なプログラムファイルだけがダウンロードされます。
JVM 1.4 以下	NexawebClient-OnDemand.jar
JVM 1.5 以上	提供されません(JVM 1.4 以下と同じものが使用されます)。
MSJVM	NexawebClient-OnDemand.cab
UITesting ファイル	提供されません。

2.5 nexaweb-nfc.jar

NFC(Nexaweb Foundation Classes) もしくは XFC(XAL Foundation Classes)を使用している場合にダウンロードされる jar ファイルになります。

NFC は nexaweb-client.xml の中の <nfc> の設定が true の場合に使用されるファイルになります。

NFC/XFC を使用しているのであれば、このファイルに対しても署名が必要になります。

ファイル格納フォルダ	/WEB-INF/Nexaweb/client/lib/
使用条件	NFC/XFC 機能を使用している場合に使用されます。
JVM 1.4 以下	nexaweb-nfc.jar
JVM 1.5 以上	nexaweb-nfc.jar.pack.gz
MSJVM	nexaweb-nfc.cab
UITesting ファイル	提供されません。

2.6 PluginAccessibility-classes.jar

アクセシビリティ機能を使用している場合にダウンロードされる jar ファイルになります。

アクセシビリティは「/WEB-INF/client/plugins」フォルダ直下に「PluginAccessibility.jar」が配置されていた場合に使用されます。

※「 PluginAccessibility.jar 」はクライアント側にダウンロードされるファイルではないため、「PluginAccessibility.jar」への署名は不要です。以下の解凍後ファイルに対する署名が必要になります。

署名は「3.2.2.1 プラグイン情報定義ファイル(plugin.xml)のパッケージング」後に実施してください。

ファイル格納フォルダ	WEB-INF/client/plugins/PluginAccessibility.jarunpacked/
使用条件	アクセシビリティ機能を使用している場合に使用されます。
JVM 1.4 以下	lib/PluginAccessibility-classes.jar
JVM 1.5 以上	提供されません(JVM 1.4 以下と同じものが使用されます)。
MSJVM	サポートされません。
UITesting ファイル	debug/PluginAccessibility-classes-debug.jar

2.7 PluginDataFramework-classes.jar

データフレームワーク機能を使用している場合にダウンロードされる jar ファイルになります。
データフレームワークは「/WEB-INF/client/plugins」フォルダ直下に「PluginDataFramework.jar」が配置されていた場合に使用されます。
※「PluginDataFramework.jar」はクライアント側にダウンロードされるファイルではないため、「PluginDataFramework.jar」への署名は不要です。以下の解凍後ファイルに対する署名が必要になります。
署名は「3.2.2.1 プラグイン情報定義ファイル(plugin.xml)のパッケージング」後に実施してください。

ファイル格納フォルダ	WEB-INF/client/plugins/PluginDataFramework.jarunpacked/
使用条件	データフレームワーク機能を使用している場合に使用されます。
JVM 1.4 以下	lib/PluginDataFramework-classes.jar lib/ognl-2.6.7.jar
JVM 1.5 以上	提供されません(JVM 1.4 以下と同じものが使用されます)。
MSJVM	サポートされません。
UITesting ファイル	提供されません。

2.8 PluginDataRequest-classes.jar

データリクエスト機能を使用している場合にダウンロードされる jar ファイルになります。
データデータリクエストは「/WEB-INF/client/plugins」フォルダ直下に「PluginDataRequest.jar」が配置されていた場合に使用されます。
※「PluginDataRequest.jar」はクライアント側にダウンロードされるファイルではないため、「PluginDataRequest.jar」への署名は不要です。以下の解凍後ファイルに対する署名が必要になります。
署名は「3.2.2.1 プラグイン情報定義ファイル(plugin.xml)のパッケージング」後に実施してください。

ファイル格納フォルダ	WEB-INF/client/plugins/PluginDataRequest.jarunpacked/
使用条件	データリクエスト機能を使用している場合に使用されます。
JVM 1.4 以下	lib/PluginDataFramework-classes.jar lib/ognl-2.6.7.jar
JVM 1.5 以上	提供されません(JVM 1.4 以下と同じものが使用されます)。
MSJVM	サポートされません。
UITesting ファイル	提供されません。

2.9 PluginValidationFramework-classes.jar

バリデーションフレームワーク機能を使用している場合にダウンロードされる jar ファイルになります。
バリデーションフレームワークは「/WEB-INF/client/plugins」フォルダ直下に「PluginValidationFramework.jar」が配置されていた場合に使用されます。
※「PluginValidationFramework.jar」はクライアント側にダウンロードされるファイルではないため、「PluginValidationFramework.jar」への署名は不要です。以下の解凍後ファイルに対する署名が必要になります。
署名は「3.2.2.1 プラグイン情報定義ファイル(plugin.xml)のパッケージング」後に実施してください。

ファイル格納フォルダ	WEB-INF/client/plugins/PluginValidationFramework.jarunpacked/
使用条件	バリデーションフレームワーク機能を使用している場合に使用されます。
JVM 1.4 以下	lib/PluginValidationFramework-classes.jar
JVM 1.5 以上	提供されません(JVM 1.4 以下と同じものが使用されます)。
MSJVM	サポートされません。
UITesting ファイル	提供されません。

2.10 PluginTestScript-classes.jar

Test API 機能を使用している場合にダウンロードされる jar ファイルになります。
Test API は「/WEB-INF/client/plugins」フォルダ直下に「PluginTestScript.jar」が配置されていた場合に使用されます。
※「PluginTestScript.jar」はクライアント側にダウンロードされるファイルではないため、「PluginTestScript.jar」への署名は不要です。以下の解凍後ファイルに対する署名が必要になります。
署名は「3.2.2.1 プラグイン情報定義ファイル(plugin.xml)のパッケージング」後に実施してください。

ファイル格納フォルダ	WEB-INF/client/plugins/PluginTestScript.jarunpacked/
使用条件	バリデーションフレームワーク機能を使用している場合に使用されます。
JVM 1.4 以下	lib/PluginTestScript-classes.jar
JVM 1.5 以上	提供されません(JVM 1.4 以下と同じものが使用されます)。
MSJVM	サポートされません。
UITesting ファイル	debug/PluginTestScript-classes-debug.jar

2.11 ユーザーMCO 及びユーザー使用外部ライブラリ

署名を行った Java Applet は署名されていないプログラムを実行することができません。
クライアント側で動作するプログラムは、Nexaweb Platform が提供するモジュール群だけでは無く、ユーザープログラム(MCO)も含まれます。
このため、ユーザーMCO も Jar ファイルにパッケージングし、署名する必要があります。
また、ユーザーMCO 内で外部ライブラリを使用してた場合、外部ライブラリの Jar ファイルに対しても署名を行う必要があります。
外部ライブラリには、プラグインアーキテクチャを用いて Nexaweb に組み込んだ UI コンポーネントが使用するライブラリも含まれます。

2.11.1 ユーザーMCOのパッケージング

ユーザーMCO で使用されるクラスファイルの実態は以下のパスに出力されます。

/WEB-INF/client/classes

上記フォルダ以下に拡張子「.class 」ファイルが出力されているので、上記フォルダ以下のファイル群を全て Jar ファイルにパッケージングしてください。
Jar ファイルへのパッケージングについては以下のドキュメントを参照ください。
http://docs.oracle.com/javase/jp/7/technotes/guides/jar/index.html

作成された Jar ファイルに対して署名を行う必要があります。

2.11.2 ユーザー使用外部ライブラリ

ユーザーMCO や、プラグインアーキテクチャを用いて新しい UI コンポーネントを組み込んだ際に外部ライブラリを使用している場合、外部ライブラリに対しても署名が必要となります。
外部ライブラリの多くは Jar ファイル形式で提供されているため、外部ライブラリの Jar ファイルに対して署名を行ってください。
外部ライブラリが Jar ファイルで提供されていない場合は、Jar ファイルにパッケージングしてから署名を行ってください。

3 初期ロード設定

署名済みの各 Jar ファイル群はアプリケーションの起動時に一括でダウンロードさせる必要があります。
各ファイルを起動時にダウンロードさせるための手段をここでは説明します。
以下での各ファイルは「2 署名対象のファイル群」で記載されているファイル群になります。
JVM のバージョン・種類によるファイル名称の違い、コンフィグレーションによるファイル名称の違い(UITesting)は手動で設定を行う場合を除いては自動的に適切なファイル名が使用されます。

3.1 デフォルトで起動時にダウンロードされるファイル群

3.1.1 detect.jar, NexawebClient.jar

「detect.jar」「NexawebClient.jar」は JVM のバージョン・種類に応じて自動的に起動時にダウンロードされるようになっているため、特別な設定は不要です。

3.1.2 nfc.jar

「nfc.jar」は<nfc/>オプションが有効な場合、自動的に起動時にダウンロードされるようになっているため特別な設定は不要です。

3.2 手動よる起動時ダウンロードの設定

「3.1 デフォルトで起動時にダウンロードされるファイル群」に記載されているファイル群以外は、自動的に起動時にダウンロードされず、動的にダウンロードする設定となっています。
このため、手動で起動時にダウンロードされるように設定する必要があります。
手動で起動時にダウンロードするための方法はコンフィグレーションによる設定と、ファイルの配置による設定を行う二つのケースが存在しています。

3.2.1 コンフィグレーションによる起動時ダウンロードの設定

nexaweb-client.xml の設定によって各 Jar ファイルを起動時にダウンロードさせるように設定する必要があります。
起動時ダウンロードの設定は、以下の要素の子要素に対して<archive/> 要素を追加し、<archive/>要素の属性値を適切な値に設定する必要があります。

/client-app/client-classpath/preloaded-in-applet-def

<archive/> 要素には以下の属性値を設定可能ですので、 jar ファイルのファイル名、パスに応じて設定を行ってください。

属性名	属性値
name	拡張子「.jar」ファイルのファイル名を文字列で指定(必須)
path	拡張子「.jar」ファイルのファイルパスを WebContent フォルダからの相対パスとして指定(必須)
pack200Path	pack200 圧縮された jar ファイルのファイルパスを WebContent フォルダからの相対パスとして指定(任意:pack200 ファイルが存在している場合)
Cab	path 属性値で指定されているファイルが cab ファイルである場合に true を指定(任意)

設定例.) ユーザーMCO の Jar ファイルを「/WebContent/WEB-INF/client/lib/mco.jar」として配置した場合

```
<archive name="mco.jar" path="/WEB-INF/client/lib/mco.jar"/>
```

3.2.1.1 コンフィグレーションによる起動時ダウンロードを必要とするファイル群

次のファイル群は、コンフィグレーションによる起動時ダウンロードの設定を必要とします。

3.2.1.1.1 NexawebClient-Common-Resources.jar

「NexawebClient-Common-Resources.jar」は通常は、必要に応じて動的にダウンロードされる設定がデフォルトになっているため、このファイルを使用する場合、起動時にダウンロードさせる設定が必要になります。
設定例.)

```
<archive name=" NexawebClient-Common-Resources.jar "
  path="/WEB-INF/Nexaweb /client/lib/NexawebClient-Common-Resources.jar"/>
```

3.2.1.1.2 ユーザーMCO、ユーザー使用外部ライブラリ

ユーザーMCO のファイル群をパッケージ化した Jar ファイルと、ユーザー使用外部ライブラリの Jar ファイルは、起動時にダウンロー路させる設定が必要になります。
設定例.) ユーザーMCO の Jar ファイルを「/WebContent/WEB-INF/client/lib/mco.jar」として配置した場合

```
<archive name="mco.jar" path="/WEB-INF/client/lib/mco.jar"/>
```

3.2.2 プラグインファイル群の起動時ダウンロードの設定

プラグインを使用している場合は、以下の利用しているプラグインファイルを「3.2.1 コンフィグレーションによる起動時ダウンロードの設定」の設定によって起動時にダウンロードさせる必要があります。

- PluginAccessibility-classes.jar
- PluginDataFramework-classes.jar
- PluginDataRequest-classes.jar
- PluginValidationFramework-classes.jar
- PluginTestScript-classes.jar

設定例.)

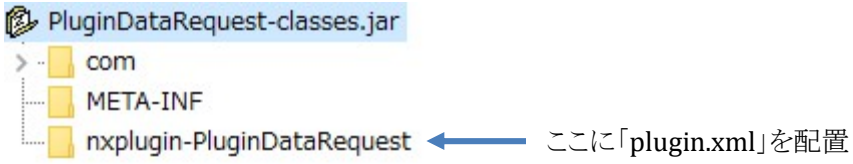
```
<archive name=" PluginAccessibility-classes.jar "
  path="/WEB-INF/Nexaweb /client/plugins/PluginAccessibility.jarunpacked/lib/ PluginAccessibility-classes.jar "/>
```

また、プラグインファイル群はプラグイン情報が定義されている XML ファイルを必要とします。
このため、上記のファイル群を起動時にダウンロードする場合、XML ファイルを JAR ファイルにパッケージングし、署名後のファイルを「3.2.1 コンフィグレーションによる起動時ダウンロードの設定」の設定によって起動時にダウンロードさせる必要があります。

3.2.2.1 プラグイン情報定義ファイル(plugin.xml)のパッケージング

各プラグイン情報ファイルは「/WEB-INF/client/plugins/<*プラグイン名*>.jarunpacked」フォルダ以下に「plugin.xml」ファイルとして保持されています。
この各「plugin.xml」ファイルを各プラグイン JAR ファイル内に「nxplugin-<*プラグイン名*>」パッケージの下に格納します。

例えば、「*PluginDataRequest*-classes.jar」の場合、JAR ファイルを解凍後、「nxplugin-PluginDataRequest」フォルダを作成し、「nxplugin-*PluginDataRequest*」フォルダの下に「/WEB-INF/client/plugins/*PluginDataRequest*.jarunpacked/plugin.xml」ファイルを配置して、JAR ファイルを再パッケージングします。
※*青斜体*の部分はプラグイン名によって変更が必要です。
JAR ファイルは以下のようになります。



プラグインファイル群の署名は、この再パッケージング後に実施してください。

4 マニフェスト

マニフェストの仕様については以下のドキュメントを参照ください。
http://docs.oracle.com/javase/jp/7/technotes/guides/jar/jar.html#JAR_Manifest
マニフェスト情報の変更は、Jar ファイルのパッケージング時に行ってください。
Jar ファイルにマニフェストを含める必要がある場合は各ファイルに対して以下の情報を含めるようにしてください。

4.1.1 Nexaweb Platformファイル群

「 detect.jar 」「 NexawebClient.jar 」「 NexawebClient-OnDemand.jar 」「 nexaweb-nfc.jar 」「PluginAccessibility-classes.jar」「PluginDataFramework-classes.jar」「PluginDataRequest-classes.jar」「PluginValidationFramework-classes.jar」「PluginTestScript-classes.jar」の Nexaweb Platform が提供している各ファイル群に対しては既存のマニフェスト情報に加えて、以下のマニフェスト情報を追加してください。

Permissions	all-permissions
Codebase	アプリケーションが実行されるドメイン URL

※赤字部分はプロジェクトの情報に併せて適切な値を設定してください。

5 署名方法

「2 署名対象のファイル群」に対しての署名方法は通常の Jar ファイルと同じ方法で署名します。
署名方法の詳細については以下のドキュメントを参照ください。

http://docs.oracle.com/javase/jp/7/technotes/guides/jar/jar.html#Signed_JAR_File
http://docs.oracle.com/javase/jp/7/technotes/tools/index.html#security
http://docs.oracle.com/javase/jp/7/technotes/tools/windows/jarsigner.html