Authenticity in Native Android applications

fork and edit tutorial (https://github.ibm.com/MFPSamples/DevCenter/tree/master/tutorials/en/foundation/7.0/authentication-security/application-authenticity-protection/application-authenticity-protection-native-android-applications.html) | report issue (https://github.ibm.com/MFPSamples/DevCenter/issues/new)

This is a continuation of the Application Authenticity Protection (../) tutorial.

Adding required files

From the MobileFirst project's Native API folder, copy the following folders to your native's project `lib` folder: `armabi`, `armabi-v7a`, `mips`, `x86`.

The application-descriptor.xml file

You modify the `application-descriptor.xml` file of your application by adding a security test and a public signing key.

Adding the security test

Add the `securityTest` attribute to the Android or iPhone/iPad environment element. For example:

```
<iphone bundleId="com.worklight.MyBankApp" applicationId="MyBankApp" securityTest="customTests" version="1.0">
```

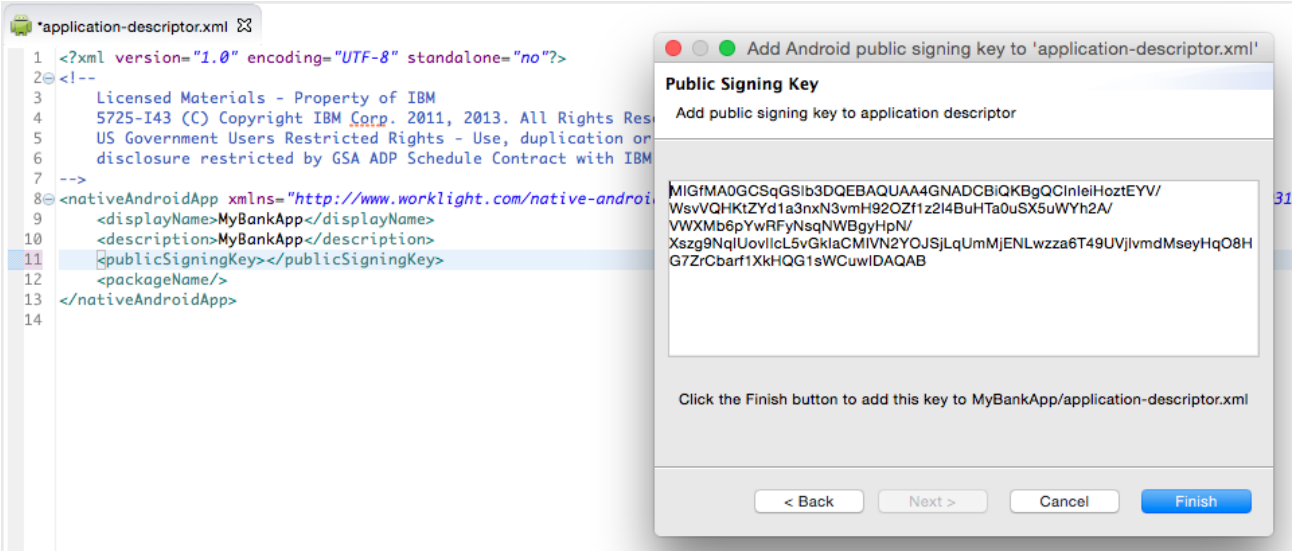
Adding the public signing key

1. Extract the public signing key of the certificate that is used to sign application bundle (`.apk` file).
   - If the application is built for distribution (production), extract the public key from the certificate that is used to sign the production-ready application.
   - If the application is built in the development environment, you can use the default public key that is supplied by the Android SDK. You can find the development certificate in a keystore that is in a `{user-home}/.android/debug.keystore` file.

   You can extract the public signing key either manually or by using the wizard that MobileFirst Studio provides.

   #### Extracting the public signing key by using the wizard

   1. Right-click the Android NativeAPI folder and select **Extract public signing key**.
   2. Specify the location and the password of a keystore file and click **Load Keystore**. The default password for `debug.keystore` is `android`.
   3. Set the **Key alias** and click **Next**.
      A dialog displays the public key.
   4. Click **Finish** to automatically paste the public signing key to the relevant section of the `application-descriptor.xml` file.



- Add the Application package name by using the Application Descriptor Editor (design view).

- Take the Application package name value from the package attribute of the *manifest* node in the `AndroidManifest.xml`.

If you decide to change the value, make sure that you change it in both locations.

You can also edit the `application-descriptor.xml` file directly to add the package name:

```
<android version="1.0" securityTest="customTests">
  <worklightSettings include="false"/>
  <security>
    <encryptWebResources enabled="false"/>
    <testWebResourcesChecksum enabled="false" ignoreFileExtensions="png, jpg, jpeg, gif, mp4, m
p3"/>
    <publicSigningKey>MIGfM …</publicSigningKey>
    <packageName>com.MyBankApp</packageName>
  </security>
</android>
```

- From the Worklight project Native API folder, copy the following folders to your native project `lib` folder: `armabi`, `armabi-v7a`, `mips`, `x86`.