# Authenticity Protection in Hybrid applications

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

## The application-descriptor.xml file

Add the securityTest attribute to the relevant environment element. For example:

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

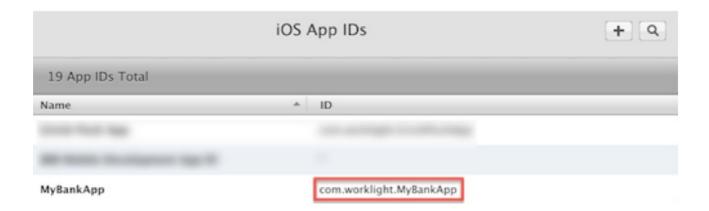
Next, you must make modifications that are specific to each environment.

#### iOS

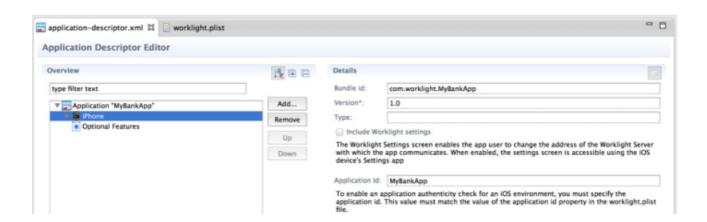
To enable application authenticity protection check for the iPhone/iPad environment, specify the following identifiers in the application-descriptor.xml file.

#### Specifying the bundleId and applicationId identifiers

1. Specify the bundleId value of the application exactly as it is defined in the **Apple Developer portal**.



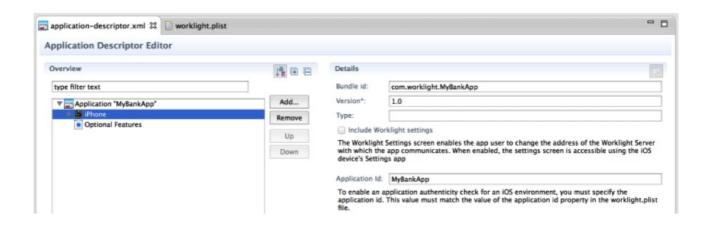
You can add the value either in the Application Description Design view:



Or in the Application Descriptor Source view:

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

2. Specify the applicationId value. The application identifier must match the value of the application id property, which is located in the native\worklight.plist file. You can add the value either in the Application Description Design view:



Or in the Application Descriptor Source view:

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

3. In Xcode, verify that the following value exists in the Other Linker Flags field: -0bjC

#### **Android**

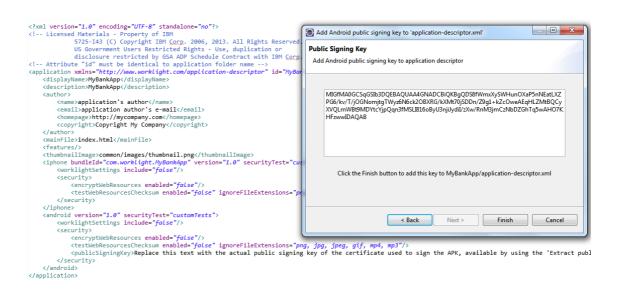
To enable application authenticity protection check for the Android hybrid environment, follow these steps:

- 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 environment folder and select Extract public signing key.
- Specify the location and the password of a keystore file and click Load Keystore.
   The default password for debug.keystore is and roid.
- Set the **Key alias** and click **Next**.A dialog displays the public key.

4. Click **Finish**. The public key is automatically pasted to the relevant section of the application-descriptor.xml file.



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

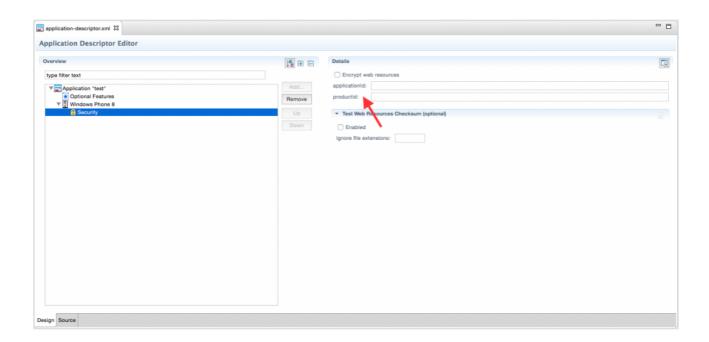


- 6. Take the value of the application package name from the package attribute of the *manifest* node in the AndroidManifest.xml file.
  - If you decide to change the value, make sure that you change it in both locations. You can also directly edit the application-descriptor.xml file and add a package name.

## **Windows Phone 8**

To enable application authenticity check for the Windows Phone 8 hybrid environment, modify the application-descriptor.xml file as follows.

• In the Application Descriptor Design view, supply the applicationId and productId in the Windows Phone 8 Security section:



You can find the productId value in the native\PropertiesWMAppManifest.xml file. The applicationId value must match the value of the wlAppId property, which you can find in the native\wlclient.properties file.

• You can also supply these values in the Application Descriptor Source view. For example: