Application Authenticity Protection

fork and edit tutorial (https://github.ibm.com/MFPSamples/DevCenter/tree/master/tutorials/en/foundation/6.3/authentication-security/application-authenticity-protection/index.html) | report issue (https://github.ibm.com/MFPSamples/DevCenter/issues/new)

Overview

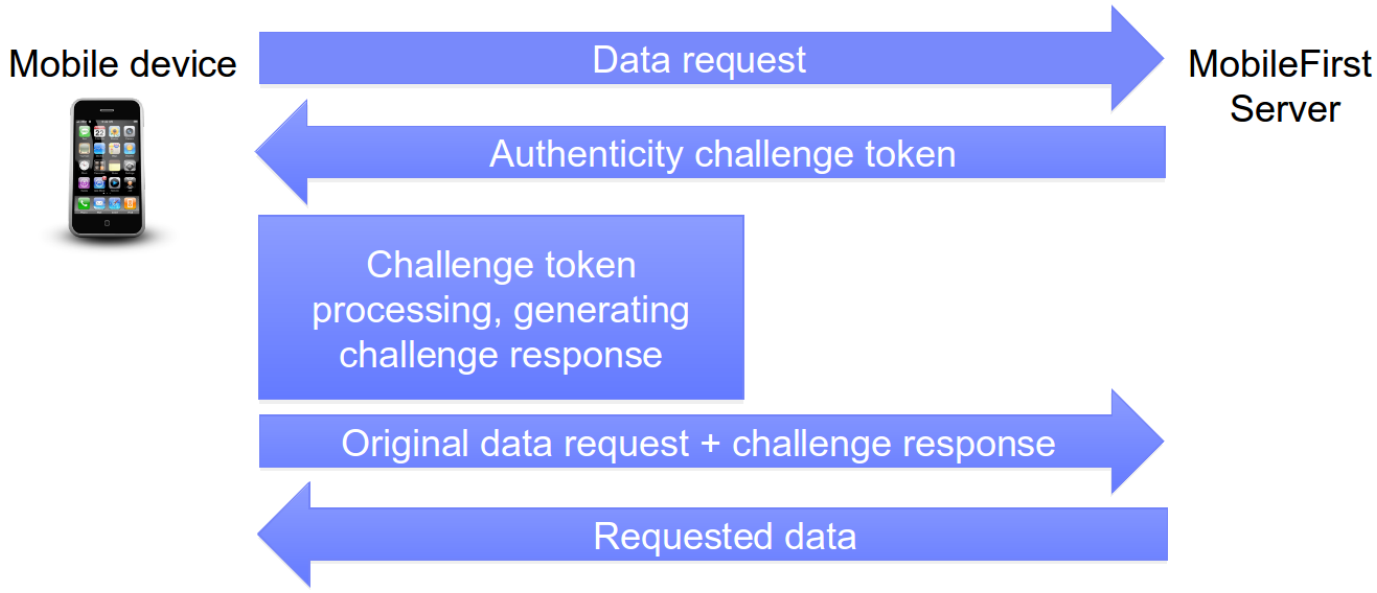
The HTTP services (APIs) that IBM MobileFirst Platform Foundation Server offers can be accessed by any entity by issuing an HTTP request.
As described in previous tutorials, it is possible to protect relevant services with various security tests.

The application authenticity check ensures that the application that tries to connect to a MobileFirst Server is the authentic one and was not tampered with or modified by a third-party attacker.

Application authenticity protection is available for Android, iOS and Windows Phone 8.

Important:
Application authenticity protection is *not available* in the MobileFirst Development Server. To test, deploy the application to a MobileFirst Server on a remote application server.
Application authenticity protection is available only to licensed installations of MobileFirst Server.

Authenticity Protection Check Flow



The challenge token is processed by using a compiled native code, therefore third-party attacker cannot see the logic of this processing.

The application authentication is based on certificate keys that are used to sign application bundle.
Only the developers or the enterprise who have the original private key that was used to create the application are able to modify, repackage, and re-sign the bundle.

Enabling application authenticity protection

authenticationConfig.xml

Add the relevant authentication realm to a security test.

If mobileSecurityTest is used, the testAppAuthenticity child-element must be added to it:

```
<mobileSecurityTest name="mobileTests">
 <testAppAuthenticity/>
 <testDeviceId provisioningType="none" />
 <testUser realm="myMobileLoginForm" />
 <testDirectUpdate mode="perSession" />
</mobileSecurityTest>
```

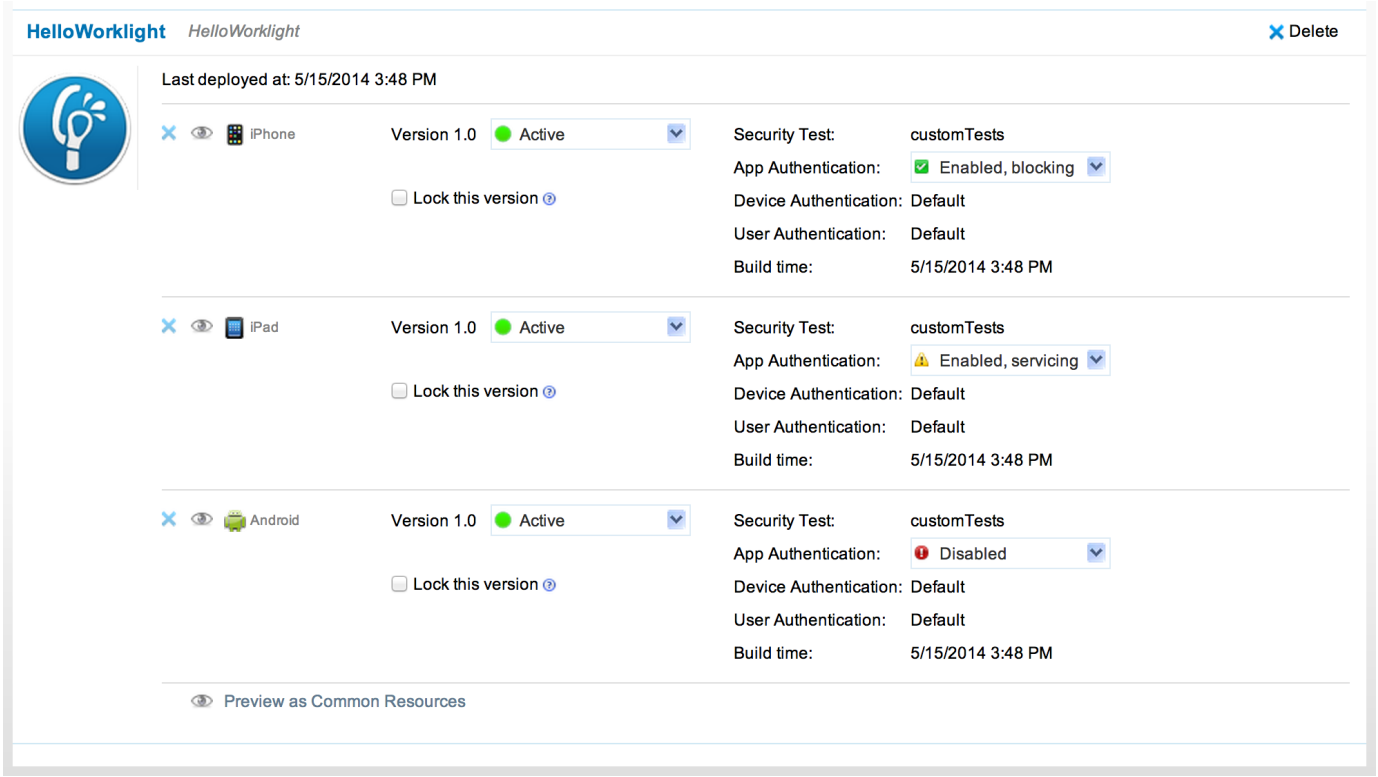
If customSecurityTest is used, the wl_authenticityRealm realm must be added to it:

```
<customSecurityTest name="customTests">
 <test realm="wl_antiXSRFRealm" step="1"/>
 <test realm="wl_authenticityRealm" step="1"/>
 <test realm="wl_remoteDisableRealm" step="1"/>
 <test realm="wl_directUpdateRealm" mode="perSession" step="1"/>
   <test realm="wl_anonymousUserRealm" isInternalUserID="true" step="1"/>
 <test realm="wl_deviceNoProvisioningRealm" isInternalDeviceID="true" step="2"/>
 </customSecurityTest>
```

Controlling application authenticity from MobileFirst Console

IBM MobileFirst Platform Foundation Console provides means for enabling and disabling application authenticity realm.
You can set three modes:

- **Enabled, blocking** – This mode means that the application authenticity check is enabled. If the application fails the check, it will not be served by a MobileFirst Server.
- **Enabled, serving** – This mode means that the application authenticity check is enabled. If the application fails the check, it will still be served by a MobileFirst Server.
- **Disabled** – This mode means that application authenticity check is disabled.


HelloWorklight
HelloWorklight
Delete
Last deployed at: 5/15/2014 3:48 PM
iPhone
Version 1.0
Active
Lock this version
Security Test: customTests
App Authentication: Enabled, blocking
Device Authentication: Default
User Authentication: Default
Build time: 5/15/2014 3:48 PM
iPad
Version 1.0
Active
Lock this version
Security Test: customTests
App Authentication: Enabled, servicing
Device Authentication: Default
User Authentication: Default
Build time: 5/15/2014 3:48 PM
Android
Version 1.0
Active
Lock this version
Security Test: customTests
App Authentication: Disabled
Device Authentication: Default
User Authentication: Default
Build time: 5/15/2014 3:48 PM
Preview as Common Resources