

Java Verified™ Program

Technical Topics

Topic 1: The Java Verified Program and MIDP 2.0 Security

The MIDP 2.0 specification (JSR 118) describes an enhanced security model for MIDlet suites. Because the Java Verified™ Program uses aspects of this specification and because more MIDP 2.0 compliant devices are appearing on the market, it is important that MIDP developers have an understanding of MIDP 2.0 security, in general and how the Java Verified Program uses it, in particular. Note that this topic does not attempt to be a description of security in MIDP 2.0. Please consult the resources listed at the end of this section for more information.

MIDP 2.0 Security for Applications

On MIDP1.0 devices, the security model constrains MIDlet suites to operate in a sandbox or untrusted domain that gives very limited access to sensitive API's and device functions. In contrast, the MIDP 2.0 security model provides device manufacturers, operators and developers much more flexibility and control over what applications can or cannot do on a device.

The GSM/UMTS Recommended Security Policy – Chapter 15 of the specification - defines three trusted domains to augment an untrusted domain: the manufacturer, operator and trusted third party domains. The trusted domains grant to applications permission to use certain sensitive API's and device functions. First, an application must prove that it can be allowed to operate in one of these trusted domains. This is done by presenting a certificate (see the next section, please) . If an application cannot do this then it is either not loaded onto the device or it is loaded but considered untrusted. As an untrusted application, it will have the most restrictive access to the operating environment (APIs and device functions). MIDP1.0 applications run in the untrusted domain by default.

Applications that have passed through the Java Verified Program successfully are allowed to operate in the trusted third party domain. Note, however, that this does not automatically grant the application any specific permissions.

The JavaVerified Signing Process

The way that an application proves to a device that they have been Java Verified is by presenting an X.509 Public Key Infrastructure (PKI) certificate. When a

testing house declares that an application has met all the criteria defined by the Unified Testing Initiative, the Java Verified Program gives that application to a Certificate Authority (CA) - a trusted entity. The CA creates a unique certificate for the developer: an identity certificate. This certificate is then used to "sign" an application. This signature is a unique series of numbers that are generated from the actual content of the Jar file. The certificate is signed, in turn, by another certificate: the UTI root certificate that the CA securely holds and is embedded in the device. This chain of trust serves to prove that the certificate is authentic. Finally, the identity certificate and the signature are placed in the Jad file of the suite. Finally, this Jad file and its Jar file are returned to the developer.

Integrity

Having a certificate accompanying a MIDlet suite also allows the device to check the integrity of the application. Since the application signature is based on the value of every byte in the Jar file, any tampering – intentional or not – of those bytes will invalidate that signature.

For developers this means that you should not change the contents of a Jar file, even if it is non-code (images and other resources). Doing so will invalidate the application's signature and prevent the application from being loaded onto a MIDP2.0 compliant device.

Behavior

What happens when a Java Verified application is delivered to a device? It depends.

If the device is a MIDP 1.0 device then the application runs in the untrusted sandbox. In this particular case, it doesn't matter that the application is signed. The signature is, in fact, ignored.

If the device is MIDP 2.0 compliant then a series of steps (more details in the specification) are performed. One of those steps is to authenticate the signature on the application's certificate. Recall that this certificate was signed by the UTI root certificate. If the **device** does **not** have a UTI root certificate stored in it then there is no way to authenticate the application certificate. The application will not be installed in the device. There are MIDP 2.0 devices now being manufactured with the UTI root certificate embedded in them.

In the above case, if the signature and certificate are removed from the Jad file then the application can be installed in the device. But it could only run in the untrusted domain. Any specific API's and functions that the application might have needed would not be made available.

Finally, there can be multiple certificates in the JAD – certificates other than those issued by the UTI. They may or may not take precedence over the UTI certificate. For instance, there could be a certificate issued by the network

operator. How multiple certificates are handled is described in the specification (chapter 3).

Resources

1. <http://jcp.org/aboutJava/communityprocess/final/jsr118/index.html> The MIDP 2.0 specification. Chapters 3,4 and 15 concentrate on security.
2. <http://developers.sun.com/techtasks/mobility/reference/techart/index.html> A good place start. There are many articles, papers and tips on MIDP 2.0 security available.