Australian Government
Australian Signals Directorate

ACSC Australian Cyber Security Centre

Mitigating Java-based Intrusions

JUNE 2020

Introduction

Java applications are widely deployed by organisations. As such, exploiting security vulnerabilities in the Java platform is particularly attractive to adversaries seeking unauthorised access to organisations' networks.

What is Java?

Java is a software platform owned and supported by Oracle. The Java platform consists of the Java Virtual Machine (JVM) and Java applications that are written using the Java programming language. The JVM is powerful, flexible and easily deployed to a wide range of devices and provides a bridge between Java applications and the devices they are run on.

Java applications give a consistent user experience independent of the underlying devices used. This makes Java highly desirable for many organisations. Unlike most applications, Java applications require the JVM to be used, and will not run natively on devices. Java applications can be presented in a web browser as an applet, or launched outside of the web browser as a Java Web Start application. Java applications may run in either privileged or sandbox modes.

What are the security issues?

Java is regularly scrutinised for security flaws and has a history of exploitable security vulnerabilities. Two methods of Java-based intrusion are:

- exploits that target security vulnerabilities in the JVM via drive-by downloads
- malicious Java applications that run outside the sandbox as privileged applications.

Once an adversary executes malicious code using either method, the compromised device could be used to conduct activities such as stealing valuable information from an organisation or gaining access to other devices on an organisation's network.

Exploitation of security vulnerabilities in the JVM

Exploitation of the JVM is mostly associated with malicious or compromised websites, but can also occur when opening an email or email attachment. This type of exploitation allows an adversary to run malicious non-Java code outside of the JVM thereby compromising a device. In doing so, the adversary can gain the same level of access as the user, or possibly even higher. Java exploits are valued as they can grant access to a device without the knowledge of users.

JVM security vulnerabilities are generally exploitable up until the time that a patch has been applied. This is a highly attractive window of opportunity for adversaries. In the time between a patch becoming available and being applied,

both the number and quality of exploits increase. As such, patching is strongly recommended as part of a defence-in-depth approach to Java security; however, security controls can be used to minimise harm and protect devices until patches can be deployed.

Malicious Java applications

Malicious Java applets will often request permission from a user to run via a pop up dialogue box. If the user trusts that the applet is safe, and accepts the certificate, the malicious applet can then run in privileged mode. Once running in privileged mode, an adversary can access parts of a device that were previously protected, such as files and network connections.

By default, Java applets running in a web browser that require privileged mode will do so using a pop up dialogue box; however, this requirement can be disabled in the Java security control panel. Changing this setting to be more permissive will leave devices at greater risk and should not be done. If a user declines a certificate, a Java applet can still run in sandbox mode, and can still gather information that may be useful to the adversary, but it will not have as much freedom to cause harm compared to if it was run in privileged mode.

How can Java be used securely?

Organisations should determine their business requirements and use cases for Java. Use cases should address which Java applications need to be run and the degree of trust associated with each. For example, a Java user interface for an internal database might be necessary for database access and have a high level of trust. Conversely, a Java application for viewing video files on the web may have both a low business requirement and a low trust level.

Business requirements can be used to determine which of the mitigation strategies below to implement. These mitigation strategies are not mutually exclusive, and should be combined to implement a strong defence-in-depth. If there are no business requirements to use Java then it should not be installed on devices (i.e. devices that can't run Java can't be compromised by Java).

Recommended mitigation strategies include, but are not limited to:

- applying security patches
- using Deployment Rule Sets to control the use of Java applications
- allowing use of Java applications from trusted sources only
- content filtering at the gateway
- configuring separate browsers for internal and external use
- host-based detection using Enhanced Mitigation Experience Toolkit (EMET).

Applying security patches

Security patches will protect devices against the exploitation of known JVM security vulnerabilities. Many adversaries rely on unpatched software, and can be stopped by installing patches for the JVM. Note however, Java applications running with privileges can still compromise a patched JVM as they rely on social engineering and do not need to exploit an unpatched JVM security vulnerability.

Using Deployment Rule Sets to control the use of Java applications

A security feature added in Oracle Java 7 Update 40 was 'Deployment Rule Sets'. This feature allows administrators to control the use of Java applications based on attributes such as location, file hash or signature hash. Notably, signature hash is a powerful way to identify and verify Java applications from trusted sources.

Deployment Rule Sets also allow administrators to specify the version of Java that is required to run a Java application. By default, Java applications will attempt to run in the newest version of the JVM available. However, legacy applications that do not work under newer versions of the JVM can be run in an older version. Deployment Rule Sets are highly recommended for organisations that rely on legacy Java applications.

Allowing use of Java applications from trusted sources only

Allowing Java applications to run only from trusted sources, such as an organisation's intranet, can reduce the security risk associated with using Java applications. Further, as signed Java applications can run with a high privilege level by asking for user consent, only trusted signing certificates should be entitled to this level of privilege.

Trusted domains can be configured in the web browser, at the gateway or using Deployment Rule Sets. Web browser-based security controls work effectively when used with separate browsers. This can be implemented in conjunction with content filtering at the gateway level. Deployment Rule Sets are the most flexible way of configuring both trusted sources and trusted code signing certificates.

Configuring separate web browsers for internal and external use

Different web browsers can be configured for use with internal (intranet) and external (internet) websites. The use of separate web browsers is simple, but can effectively control the use of Java applications.

External web browsers should be configured to block or heavily restrict Java applications, noting this may inhibit the legitimate use of Java applications from the internet. If Java from external websites is required, then the web browser should be configured to only allow Java from trusted sources. Java applications from untrusted websites should still be controlled or blocked.

The internal web browser in comparison may be more permissive in running Java applications as intranet websites are inherently more trustworthy. However, the web browser used for intranet traffic should be blocked at the organisation's internet gateway to prevent it from being used for accessing external websites.

Multiple web browsers can be used with different versions of Java if necessary. For example, an internal web browser may need to run Java 6 for compatibility with a legacy business application, but the external browser could be updated to the latest version for security reasons.

Content filtering at the gateway

Delivery of Java content to users can be controlled using a web proxy or web content filter. Such devices can be configured to refuse outgoing requests for Java content based on file extension or MIME type. Java file extensions include *.class*, *.jar* and *.jnlp*. Java MIME types include *application/.*-java-.** and *application/java-archive*. In addition, restriction of trusted sources can be implemented by creating exceptions for trusted domains. For example, configuring a web proxy to drop all requests for Java file extensions, unless the request is for a local intranet address range, in which case the request will be allowed.

Further information

The ***Australian Government Information Security Manual*** (ISM) assists in the protection of information that is processed, stored or communicated by organisations' systems. It can be found at https://www.cyber.gov.au/acsc/view-all-content/ism.

The ***Strategies to Mitigate Cyber Security Incidents*** complements the advice in the ISM. The complete list of strategies can be found at https://www.cyber.gov.au/acsc/view-all-content/publications/strategies-mitigate-cyber-security-incidents.

Further information on Deployment Rule Sets can be found at https://docs.oracle.com/javase/10/deploy/deployment-rule-set.htm#JSDPG926.

Contact details

If you have any questions regarding this guidance you can contact us via 1300 CYBER1 (1300 292 371) or https://www.cyber.gov.au/acsc/contact.