snyk | SECURITY

Deserialization of Untrusted Data

Affecting com.google.code.gson:gson package, versions [2.2.3,2.8.9)

INTRODUCED: 11 OCT 2021 CVE-2022-25647 CWE-502
FIRST ADDED BY SNYK

How to fix?

Upgrade `com.google.code.gson:gson` to version 2.8.9 or higher.

Overview

Affected versions of this package are vulnerable to Deserialization of Untrusted Data via the `writeReplace()` method in internal classes, which may allow a denial of service attack if combined with another exploit.

Details

Serialization is a process of converting an object into a sequence of bytes which can be persisted to a disk or database or can be sent through streams. The reverse process of creating object from sequence of bytes is called deserialization. Serialization is commonly used for communication (sharing objects between multiple hosts) and persistence (store the object state in a file or a database). It is an integral part of popular protocols like *Remote Method Invocation (RMI)*, *Java Management Extension (JMX)*, *Java Messaging System (JMS)*, *Action Message Format (AMF)*, *Java Server Faces (JSF) ViewState*, etc.

Deserialization of untrusted data (CWE-502), is when the application deserializes untrusted data without sufficiently verifying that the resulting data will be valid, letting the attacker to control the state or the flow of the execution.

Java deserialization issues have been known for years. However, interest in the issue intensified greatly in 2015, when classes that could be abused to achieve remote code execution were found in a

Severity

RECOMMENDED

CVSS assessment by Snyk's Security Team. Learn more

Threat Intelligence

EPSS 2.87% (87th percentile)

Do your applications use this vulnerable package?

In a few clicks we can analyze your entire application and see what components are vulnerable in your application, and suggest you quick fixes.

Test your applications

Snyk Learn

Learn about Deserialization of Untrusted Data vulnerabilities in an interactive lesson

popular library (Apache Commons Collection). These classes were used in zero-days affecting IBM WebSphere, Oracle WebLogic and many other products.

An attacker just needs to identify a piece of software that has both a vulnerable class on its path, and performs deserialization on untrusted data. Then all they need to do is send the payload into the deserializer, getting the command executed.

> Developers put too much trust in Java Object Serialization. Some even de-serialize objects pre-authentication. When deserializing an Object in Java you typically cast it to an expected type, and therefore Java's strict type system will ensure you only get valid object trees. Unfortunately, by the time the type checking happens, platform code has already created and executed significant logic. So, before the final type is checked a lot of code is executed from the readObject() methods of various objects, all of which is out of the developer's control. By combining the readObject() methods of various classes which are available on the classpath of the vulnerable application, an attacker can execute functions (including calling Runtime.exec() to execute local OS commands).

References

- GitHub Commit Introducing Vulnerable Method
- GitHub Fix Commit
- GitHub PR

CVSS Base Scores

version 3.1

Snyk 6.5 MEDIUM

Attack Vector (AV)	Network	Scope (S)	Unchanged	Confidentiality (C)	None
Attack Complexity (AC)	High			Integrity (I)	Low
Privileges Required (PR)	None			Availability (A)	High
User Interaction (UI)	None				

NVD 7.5 HIGH

Red Hat 7.5 HIGH

SUSE 7.5 HIGH

an interactive lesson.

Start learning

Snyk ID SNYK-JAVA-COMGOOGLECODEGSON-1730327

Published 2 Nov 2021

Disclosed 11 Oct 2021

Credit Marcono1234

Report a new vulnerability

Found a mistake?