REPORT

Snyk Top 10: Open Source Vulnerabilities in 2022

Every three to four years, OWASP puts out their Top 10 list of critical vulnerabilities. Their list is "a standard awareness document for developers and web application security... represent[ing] a broad consensus about the most critical security risks to web applications."

At Snyk, we think that every developer that's serious about security should be familiar with the OWASP Top 10 to ensure that their applications are safe from bad actors. And to help those security-conscious developers further, we will be releasing a series of Snyk Top 10 lists based on the data available to us from scans run by our users. In this first piece, we'll be taking a look at the **Snyk Top 10 Open Source Vulnerabilities** in 2022. It's important to note that we're looking at specific vulnerability types, not grouping them as OWASP does.

Everyday, tens of thousands of scans are run by our customers to find vulnerable open source libraries in their applications (and in the transitive dependencies of those libraries). Here are what we found to be the most prevalent **critical** and **high** vulnerabilities from **Jan. 1–Sept. 30** of this year. It's important to note that these vulnerabilities are skewed in favor of Java, as that was the ecosystem most scanned by our user base.

1. Denial of Service (DoS)

Denial of service (DoS) attacks are used to shut down access to a network or server by bombarding the target with so many requests that it's unable to process the load. Both Google and AWS experienced large scale DoS attacks in 2020, but this type of attack isn't just reserved for major cloud providers.

Two common types of DoS vulnerabilities:

- **High CPU/memory consumption**: An attacker sending crafted requests that could cause the system to take a disproportionate amount of time to process. Example: commons-fileupload:commons-fileupload.
- **Crash** - An attacker sending crafted requests that could cause the system to crash. Example: npm ws package.

The top high/critical vulnerability in Java, .NET, and Ruby, let's look at the top DoS vuln for each ecosystem. It's important to note that each of these below vulnerabilities has **low complexity and high availability** (making them easy to exploit), but can all be **remediated with a simple version upgrade**.

TOP OF 2022

Java

CVE-2020-36518 - `com.fasterxml.jackson.core:jackson-databind` is a library which contains the general-purpose data-binding functionality and tree-model for Jackson Data Processor. Affected versions of this package are vulnerable to Denial of Service (DoS) via a large depth of nested objects.

.NET

CVE-2022-29117 - .NET and Visual Studio denial of service vulnerability (resource exhaustion)

Ruby

CVE-2021-22569 - An issue in `protobuf-java` allowed the interleaving of `com.google.protobuf.UnknownFieldSet` fields in such a way that would be processed out of order. A small malicious payload can occupy the parser for several minutes by creating large numbers of short-lived objects that cause frequent, repeated pauses.

2. Remote Code Execution (RCE)

Remote code execution (RCE) attacks occur when a bad actor is able to run commands from a remote system that they shouldn't have access to. RCEs can occur a few ways, including through the use of malware or by exploiting a vulnerable library that hasn't been patched. This vulnerability allows attackers to exploit a server or an application using their code on the server or application. For this list, we're focused on the latter, as this vulnerability can lie dormant in a seemingly safe library for years — as was the case with Log4Shell.

IN THE NEWS

In late 2021, Log4j — the widely-used open source Java logging library that has been around since 2001 — was found to have a RCE that impacted systems around the globe. Named Log4Shell, this exploit sent teams into a frenzied weekend of patching. In the case of this exploit, Log4j was often a transitive dependency of another open source package or a part of a container image, so remediation was time consuming if the proper tools (like Snyk) weren't used. Snyk team did an extensive work around creating a publicly available repo — awesome-log4shell — covering the information available on the internet.

Log4Shell was promptly followed by Spring4Shell, an RCE discovered in the Spring Framework. To learn more about this RCE, we recommend taking our free Spring4Shell: Exploiting a remote code execution vulnerability hands-on lesson.

3. Deserialization of Untrusted Data

Deserialization of untrusted data is when an application deserializes untrusted data without sufficiently verifying that the resulting data will be valid, thus allowing the attacker to control the state or the flow of the execution. A Java deserialization vulnerability occurs when a malicious user tries to insert a modified serialized object into the system in order to compromise the system or it's data. If you want to learn more about this vulnerability type, we recommend reading Serialization and deserialization in Java: Explaining the Java deserialize vulnerability.

TOP INSTANCE

This type of vulnerability can occur in many different languages, but the most prevalent occurrence we found was CVE-2022-23307 — another Log4j vulnerability. This is a high complexity vulnerability, but remains high severity due to the level of impact it can have and the package 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.

4. SQL Injection

SQL Injection is a type of vulnerability when an application takes inputs from the user without validating and passes on the database to process. This allows an attacker to add untrusted data to a database query. For instance, when filling in a web form, a SQL injection could allow an attacker to create user input to steal valuable data, bypass authentication, or corrupt records. For more information on SQL injection, we recommend reading our SQL injection cheat sheet or taking our interactive SQL injection lesson.

TOP PYTHON VULN

This was the most frequently found vulnerability for Python this year, with CVE-2022-28347 coming in at the top of the list. This flaw in Django is vulnerable to SQL injection via `QuerySet.explain(**options)` in option names, using a suitably crafted dictionary (with dictionary expansion) as the `**options argument` on PostgreSQL. It can be resolved by upgrading Django to 2.2.28, 3.2.13, 4.0.4 or higher.

5. Prototype Pollution

Prototype pollution is a vulnerability affecting JavaScript, and it refers to the ability to inject properties into existing JavaScript language construct prototypes, such as objects. JavaScript allows all `Object` attributes to be altered, including their magical attributes such as `__proto__`, `constructor`, and `prototype`. An attacker manipulates these attributes to overwrite (pollute) a JavaScript application object prototype of the base object by injecting other values. Properties on the `Object.prototype` are then inherited by all the JavaScript objects through the prototype chain. When that happens, this most commonly leads to a denial of service by triggering JavaScript exceptions, or if the code evaluates the specific attribute of the JavaScript object controled by the attacker it is able to tamper with the application source code to force the code path that the attacker injects, thereby leading to remote code execution.

There are two main ways in which the pollution of prototypes occurs:

- Unsafe object recursive merge
- Property definition by path

To learn more about prototype pollution, check out this Capture the Flag walkthrough from SnykCon 2021, try this interactive tutorial, or check out this Git repository about exploiting prototype pollution.

TOP JS VULN

This is a common vulnerability type for the JS ecosystem, with CVE-2021-43138 at the top. This prototype pollution vulnerability in `async`. Affected versions of this package are vulnerable via the `mapValues()` method, due to improper check in `createObjectIterator function`. It can be resolved with a simple upgrade to 2.6.4, 3.2.2 or higher.

6. Insecure Temporary File

Insecure temporary files are, well, exactly what they sound like. With this type of vulnerability, temporary files containing sensitive information are created with incorrect permissions or in folders that lack proper permissions. This would fall under the class of vulnerability that OWASP refers to as Broken Access Control.

MOST COMMMON INSTANCE

The most frequently occuring instance of this vulnerability is CVE-2022-27772 in `org.springframework.boot:spring-boot`. Affected versions are vulnerable via the `org.springframework.boot.web.server.AbstractConfigurableWebServerFactory.createTempDir` method, allowing for temporary directory hijacking and privilege escalation. Fortunately, this can be remediated with a simple upgrade to 2.2.11 or higher.

7. Directory/Path Traversal

A directory traversal (a.k.a. path traversal) attack aims to access files and directories that are stored outside the intended folder. By manipulating files with "dot-dot-slash (../)" sequences and its variations, or by using absolute file paths, it may be possible to access arbitrary files and directories stored on the filesystem; including application source code, configuration, and other critical system files. To learn more about directory traversal vulnerabilities, try this interactive lesson, watch this helpful exploit video, or read this blog that explores 3 different types of directory traversal exploits in C/C++.

TOP DIRECTORY TRAVERSAL

The top directory traversal we saw this year was CVE-2022-24785, appearing in the `moment` JS date package. Affected versions of this package are vulnerable when a user provides a locale string which is directly used to switch moment locale, but can be remediated by upgrading to 2.29.2 or higher.

8. Privilege Escalation

Privilege escalation is another type of OWASP's Broken Access Control vulnerability. In this case, access controls are bypassed by an attacker to gain increased permissions to a resource. If you'd like to learn more (and are a fan of DevSecOps), we recommend reading the blog Kernel privilege escalation: how Kubernetes container isolation impacts privilege escalation attacks.

TOP INSTANCE

The top instance of privilege escalation this year was CVE-2022-23181 in `tomcat-catalina`. Affected versions are vulnerable via a `time of check, time of use` vulnerability that allows a local attacker to perform actions with the privileges of the user that the Tomcat process is using. This issue is only exploitable when Tomcat is configured to persist sessions using the FileStore, and can be remediated by upgrading `org.apache.tomcat:tomcat-catalina` to version 8.5.74, 9.0.57, 10.0.15, 10.1.0-M9 or higher.

9. Regular Expression Denial of Service (ReDoS)

Regular expression denial of service (ReDoS) vulnerabilities are a type of DoS attack. Regular expressions (regex) are incredibly powerful, but they aren't very intuitive and can ultimately end up making it easy for attackers to take your site down.

The goal of the attack is bog the target server by having it process a bulk, invalid regex statement. The regex engine will match the first possible way to accept the current character and proceed to the next one. If it then fails to match the next one, it will backtrack and see if there was another way to digest the previous character. If it goes too far down the rabbit hole only to find out the string doesn't match in the end, and if many characters have multiple valid regex paths, the number of backtracking steps can become very large, resulting in what is known as **catastrophic backtracking**. If you'd like to learn more, check out this interactive ReDoS tutorial.

TOP 2022 VULN

Just like with directory traversals, the top ReDoS vulnerability in 2022 was in the `moment` package as CVE-2022-31129. Affected versions of this package are vulnerable via the `preprocessRFC2822()` function in `from-string.js`, when processing a very long crafted string (over 10k characters). The simple fix is to upgrade `moment` to version 2.29.4 or higher.

10. NULL Pointer Dereference

And finally, the 10th most frequently found high/critical vulnerability found during Snyk scans was NULL pointer dereference. Additionally, this was the top high/critical vulnerability in Go as well as being one of the most common vulnerabilities in C and C++. This vulnerability occurs when an application attempts to dereference a point that it expects to be valid, but finds a NULL value instead, which creates a crash. This crash then creates a denial of service flaw.

TOP OCCURANCE

The top occurrence this year was CVE-2020-29652 in `golang.org/x/crypto/ssh`. In this case, a NULL pointer dereference through `v0.0.0-20201203163018-be400aefbc4c` allows remote attackers to cause a denial of service against SSH servers. To remediate, upgrade the crypto/ssh package to `0.0.0-20201216223049-8b5274cf687f` or higher.

Stay security aware

These were just the top ten high/critical vulnerability types found by Snyk scans from Jan. 1–Oct 31, 2022, with the total growing considerably once all vulnerability types are included (medium/low). In order to keep these vulnerabilities out of your code, we recommend the following:

- **Shift security left** - The sooner you catch a vulnerability, the easier it is to fix.
- **Implement developer security tooling** - Implement a tool (like Snyk) that can find vulnerabilities in real time and then offer simple fixes to developers (not just a link off to documentation), as well as contextual prioritization.
- **Cover your full application** - Vulnerabilities in open source and proprietary code are just as dangerous as misconfigurations in your IaC/cloud configs. Be sure to secure all aspects of your application.
- **Automate, automate, automate** - Automate scans and fixes as much as you can to make security a paved path.
- **Keep learning** - Stay ahead of attackers by keeping your security skills fresh. The Snyk Learn lessons we've linked throughout are one way to do so, as security is an ever evolving field.

Keep vulnerabilities out

Find out how simple Snyk makes it to find and fix vulnerabilities in your open source dependencies and across your entire SDLC.

Book a Demo