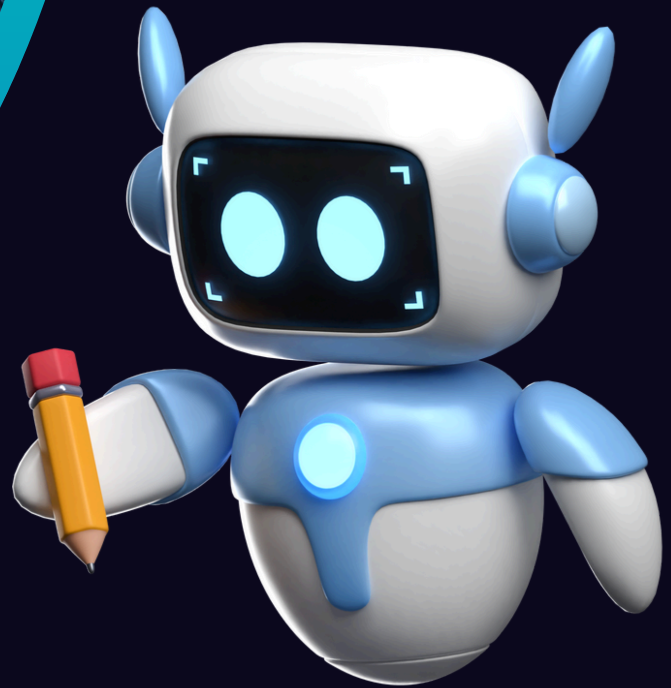
TALENTO TECH

REGIÓN 3
CAUCA - NARIÑO
LESSON 3



Lesson 3. Reading
OWASP Top 10 Vulnerabilities

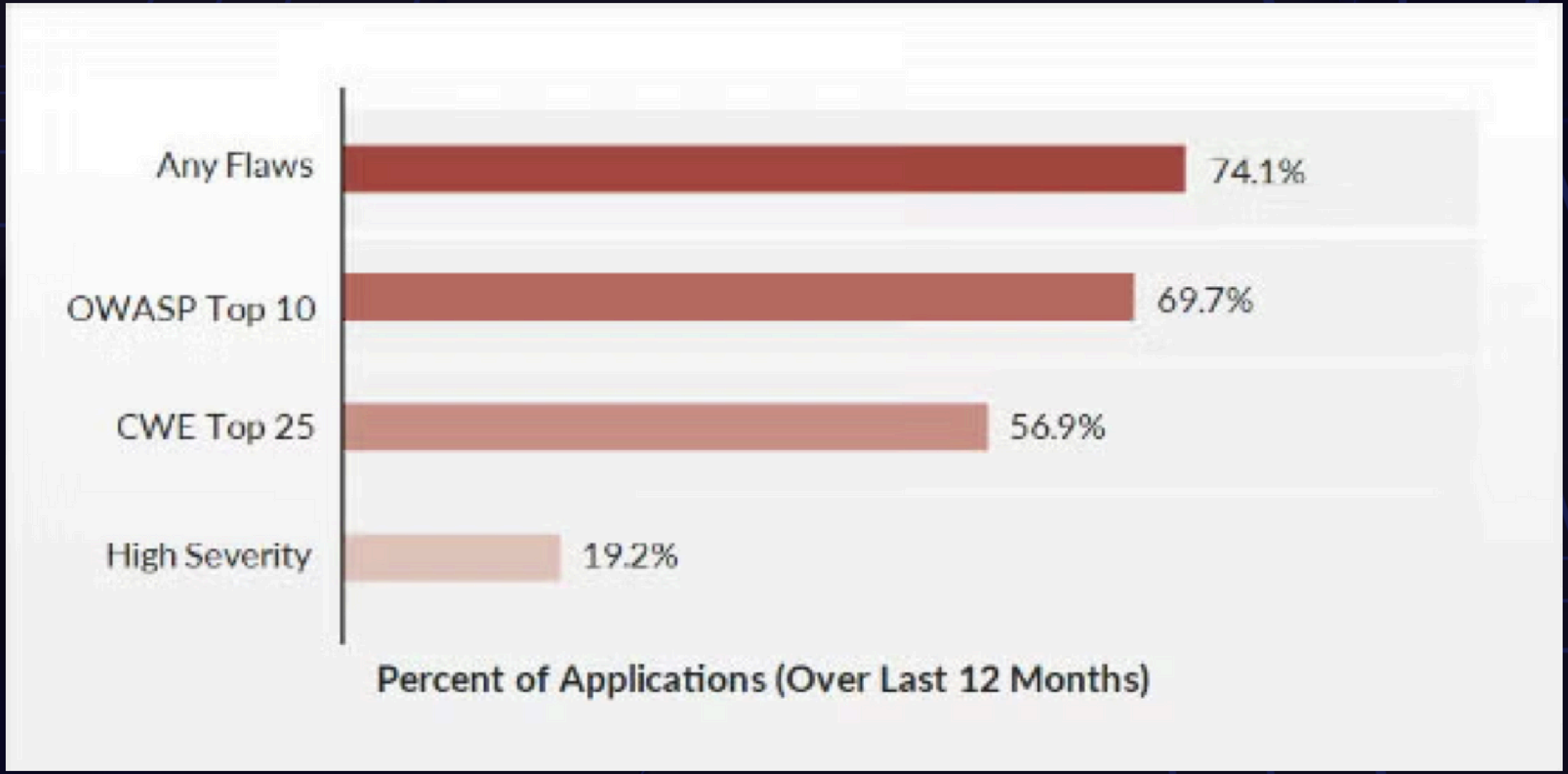
The OWASP Top 10 is a list of the 10 most common web application security risks. By writing code and performing robust testing with these risks in mind, developers can create secure applications that keep their users' confidential data safe from attackers.

What is OWASP?

OWASP, or the Open Web Application Security Project, is a nonprofit organization focused on software security. Their projects include a number of open-source software development programs and toolkits, local chapters and conferences, among other things. One of their projects is the maintenance of the OWASP Top 10, a list of the top 10 security risks faced by web applications.

Meeting OWASP Compliance to Ensure Secure Code

The OWASP Top 10 is a great foundational resource when you're developing secure code. In our State of Software Security 2023, a scan of 759,445 applications found that nearly 70% of apps had a security flaw that fell into the OWASP Top 10.



The OWASP Top 10 isn't just a list. It assesses each flaw class using the OWASP Risk Rating methodology and provides guidelines, examples, best practices for preventing attacks, and references for each risk. By learning the flaws on the OWASP Top 10 chart and how to resolve them, application developers can take concrete steps toward a more secure application that helps keep users safe when it comes to malicious attacks.

Of course, the vulnerabilities listed by OWASP aren't the only things developers need to look at. Check our guide on Application Security Fallacies and Realities to learn about common misconceptions, errors, and best practices for application security testing and production.

A Guide to OWASP Top 10 Testing

Testing for OWASP vulnerabilities is a crucial part of secure application development. The sheer number of risks and potential fixes can seem overwhelming but are easy to manage if you follow a few simple steps:

Build security into your development process, rather than making it an afterthoughtTest your code against security standards repeatedly throughout developmentUse IDE and CI Pipeline integrations to automate testingIdentify known vulnerabilities in third-party code to ensure your program does not rely on insecure dependencies

OWASP Top 10 Vulnerabilities

¿So, what are the top 10 risks according to OWASP? We break down each item, its risk level, how to test for them, and how to resolve each.

A01. Broken Access Control

If authentication and access restrictions are not properly implemented, it's easy for attackers to take whatever they want. With broken access control flaws, unauthenticated or unauthorized users may have access to sensitive files and systems, or even user privilege settings.

Penetration testing can detect missing authentication but cannot determine the misconfigurations that lead to the exposure. One of the benefits of the increasing use of Infrastructure as Code (IaC) tools is the ability to use scanning tools to detect configuration errors leading to access control failures.

Weak access controls and issues with credentials management in applications are preventable with secure coding practices, as well as preventative measures like locking down administrative accounts and controls and using multi-factor authentication.

A02: Cryptographic Failures

Common errors such as using hardcoded passwords, outdated cryptographic algorithms, or weak cryptographic keys can result in the exposure of sensitive data (the previous name for this category).
Scanning images for hard coded secrets, and ensuring that data is properly encrypted at rest and in transit can help mitigate exposing sensitive data to attackers.

A03: Injection

Injection attacks occur when attackers exploit vulnerabilities in web applications that accept untrusted data. Common types include SQL injection and OS command injection. This category now also includes Cross Site Scripting (XSS). By inserting malicious code into input fields, attackers can execute unauthorized commands, access sensitive databases, and potentially gain control over systems.
Application security testing can reveal injection flaws and suggest remediation techniques such as stripping special characters from user input or writing parameterized SQL queries.

A04: Insecure Design

Insecure design is a new category in the 2021 OWASP Top Ten which focusses on fundamental design flaws and ineffective controls as opposed to weak or flawed implementations.
Creating secure designs and secure software development lifecycles requires a combination of culture, methodologies and tools. Developer training, robust threat modelling, and an organizational library of secure design patterns should all be implemented to reduce the risks of insecure designs creating critical vulnerabilities.

A05: Security Misconfiguration

Application servers, frameworks, and cloud infrastructure are highly configurable, and security misconfigurations such as too broad permissions, insecure default values left unchanged, or too revealing error messages can provide attackers easy paths to compromise applications.

The 2023 Veracode State of Software Security reported that misconfiguration errors were reported in 70% or more applications that had introduced a new vulnerability in the last year.

To reduce misconfiguration risks organizations should routinely harden deployed application and infrastructure configurations and should scan all infrastructure as code components as part of a secure SDLC.

A06: Vulnerable and Outdated Components

Modern applications are built using a large number of third-party libraries (which themselves are dependent on other libraries), and frequently run on open-source frameworks. In a modern application there may be orders of magnitude more code from libraries and components than written by an organization's developers.

As might be expected with any software, vulnerabilities in libraries and components will routinely be discovered, patched, and new versions released. The challenges of identifying all the components in use, keeping track of their vulnerability status, and routinely rebuilding and testing deployed software is both essential and onerous. Perhaps this is why so many organizations are still running vulnerable software in production.

A critical mitigation step is to build a Software Bill of Materials (SBoM) for all the software deployed or supplied to customers. Veracode Software Composition analysis and Container Scanning tools can produce SBoMs in standardized formats to give organizations a view of their exposure to vulnerabilities in third-party components.

A07 Identification and Authentication Failures

Identifying and authorizing users and non-human clients is a fundamental security practice. It goes without saying that weaknesses in a way an application allows access or identifies users is a critical vulnerability. While mitigation starts with secure coding practices, tools to detect and prevent credential stuffing and brute force attacks are also useful protections.

A08: Software and Data Integrity Failures

The tools used to build, manage, or deploy software are increasingly common vectors of attack. A CI'CD pipeline that routinely builds, tests and deploys software can also be used to inject malicious code (or libraires), create insecure deployments, or steal secrets.

As discussed above in 'Vulnerable and Outdated Components' modern applications use many third-party components, often pulling them from third party repositories.

Organizations can mitigate this threat by ensuring both the security of the build process, and the components pulled into the build. Adding in code scanning and software component analysis steps into a software build pipeline can identify malicious code or libraries. Ensuring the build and

A09: Security Logging and Monitoring Failures

Having adequate logging and monitoring in place is essential in both detecting a breach early, hopefully limiting the damage, and in incident forensics to establish the scope of the breach, and to determine the method of compromise.

Simply generating the data is obviously insufficient, organizations must have adequate collection, storage, alerting and escalation processes. Organizations should also verify that these processes are working correctly – using Dynamic Application Security Testing (DAST) tools like Veracode DAST, for instance, should produce significant logging and alerting events.

A10: Server-Side Request Forgery (SSRF)

Modern web applications commonly fetch additional content or data from a remote resource. If an attacker can influence the destination resource, and the application does not validate the supplied URL, then a crafted request may be sent to a target destination.

Mitigating SSRF attacks is done using familiar methods such as sanitizing user input, using explicit allow lists, and inspecting request responses before they are returned to clients.

Comprehensive AppSec Guides and Services

Veracode offers comprehensive guides for training developers in application security, along with scalable web-based tools to make developing secure applications easy. Download one of our guides or contact our team to learn more about our demo today.