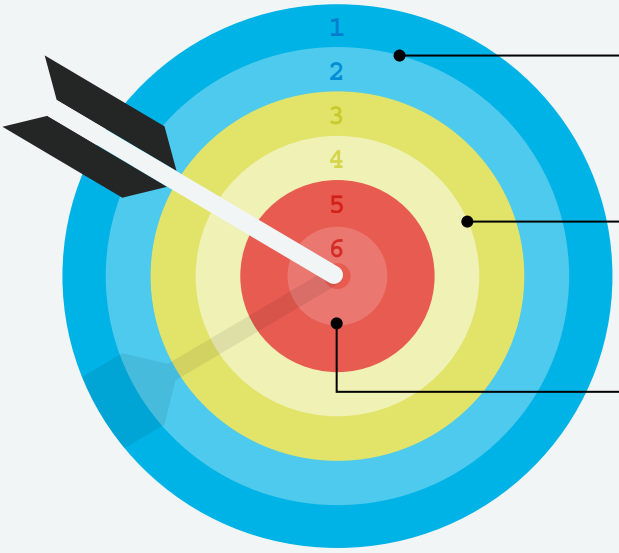
VERACODE

Red Zone Vulnerabilities

Some of the riskiest modern software flaws—and how to go about preventing them



LOW RISK

Code Quality: Commonly seen in Java and .NET

Information Leakage: Commonly seen in .NET, PHP, and Java

MEDIUM RISK

CRLF Injection: Commonly seen in Java and JavaScript

Directory Traversal: Commonly seen in PHP and Python

DANGER ZONE

Cryptographic Issues: Commonly seen in Python and PHP

Cross-Site Scripting: Commonly seen in PHP, JavaScript, and Python

1

Code Quality

Seen in 60.4% of applications with flaws

Problems with code quality arise when there are weaknesses in your code that indicate to an attacker the app has not been carefully developed or maintained. Code quality issues do not directly introduce flaws into your applications, but they can cause unpredictable behavior that hackers easily abuse.

We see instances of code quality problems in 60.4% of applications with security flaws, especially in Java and .NET. While it is one of the most common problems, you can get ahead of it with good coding methodologies. It's also important to use consistent coding patterns and automate security testing in your SDLC to discover and remediate flaws more efficiently while you code.

LEARN MORE →

2

Information Leakage

Seen in 63% of applications with flaws

Information leakage is a coding issue that can appear in many forms, but at a high level, it typically boils down to users having access to information and data that they should not have access to. Attackers can use this access to obtain leaked information and then exploit the app, or simply steal the data.

Information leakage flaws typically show up in .NET, PHP, and Java. It's the most common flaw type in PHP and .NET, showing up in around 63% of applications with flaws in both languages. Your best line of defense is to implement secure coding practices through developer training tools and to automate security testing procedures right in the SDLC while you work.

LEARN MORE →

3

CRLF Injection

Seen in 65.4% of applications with flaws

Carriage Return or Line Feed exploits, also known as CRLF injections, enable threat actors to maliciously manipulate a web application's functions after injecting a CRLF sequence into an HTTP stream.

We know that CRLF injections impact Java, JavaScript, and Python more frequently, appearing in a hefty 65.4% of applications with a flaw on initial scan. But the good news is that you can prevent these flaws by never trusting user input, and by properly sanitizing user-supplied data with validation and coding. Additionally, make sure you're encoding output in your HTTP headers.

LEARN MORE →

4

Directory Traversal

Seen in 47.8% of applications with flaws

Sometimes called a path traversal, directory traversals are a type of HTTP exploit that attackers use to gain unauthorized access to files and restricted directories. This flaw enables threat actors to exploit inadequate security mechanisms, accessing directories and files that are stored outside of the web root folder.

Directory traversal vulnerabilities are found in nearly half (47.8%) of applications with a flaw, impacting PHP, C++, .NET, and Java most frequently. You can prevent them by following secure coding practices and running frequent static analysis (SAST) scans. It's also important to validate user input from browsers, keep your web server software up to date with the latest patches, and use filters to block specific user input.

LEARN MORE →

5

Cryptographic Issues

Seen in 63.7% of applications with flaws

Cryptographic errors in code include using broken crypto algorithms, employing inadequate encryption strength, storing sensitive information in cleartext, and improperly validating certificates. It's a big problem: nearly two-thirds (63.7%) of applications have cryptographic issues on first scan, most often plaguing Python, PHP, Java, JavaScript, and C++ languages.

These issues can lead to stolen or destroyed data, including sensitive information like Social Security numbers and bank details. Fortunately, you can prevent issues with cryptographic flaws through secure coding best practices. Concerns over improper implementation usually come up on a case by case basis and may require a remediation plan that fits your specific application's needs.

LEARN MORE →

6

Cross-Site Scripting

Seen in 74.6% of applications with flaws

Impacting several common languages, cross-site scripting (XSS) is a dangerous flaw that allows attackers to inject client-side scripts into an application. This creates a large attack surface that hackers can use to spread worms and Trojans, hijack user accounts, access browser history, exploit online applications, and even control a browser remotely.

Though it shows up in PHP most frequently (74.6%), XSS also makes regular appearances in JavaScript, Python, Java, and .NET. It's the most common type of vulnerability in open source code across nearly every language, too. To prevent XSS attacks, implement secure coding practices and always sanitize input from search fields or forms by validating that it's the expected content, and then properly encoding it for the "endpoint."

LEARN MORE →

Secure coding skills are critical, but you can build them over time with real-world practice. Veracode Security Labs Community Edition is a complimentary version of our training platform that will help you build secure coding know-how to reduce pressure, save time, and squash risky flaws before they become headaches.

LEARN MORE →