DevOps Fundamentals

Secure DevOps Tooling

Agenda

SAST

SCA

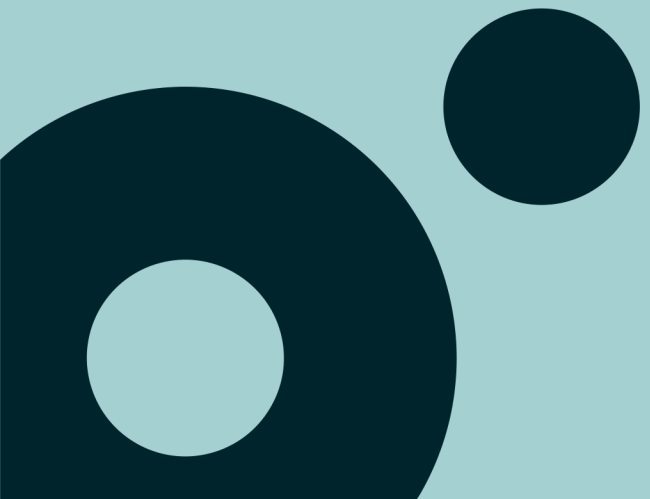
OWASP

CVE, CWE, CVSS

GitHub Advanced Security

moOngy.

SAST



Static Application Security Testing (SAST)

Improve code security and quality on an easy and cost-effective way

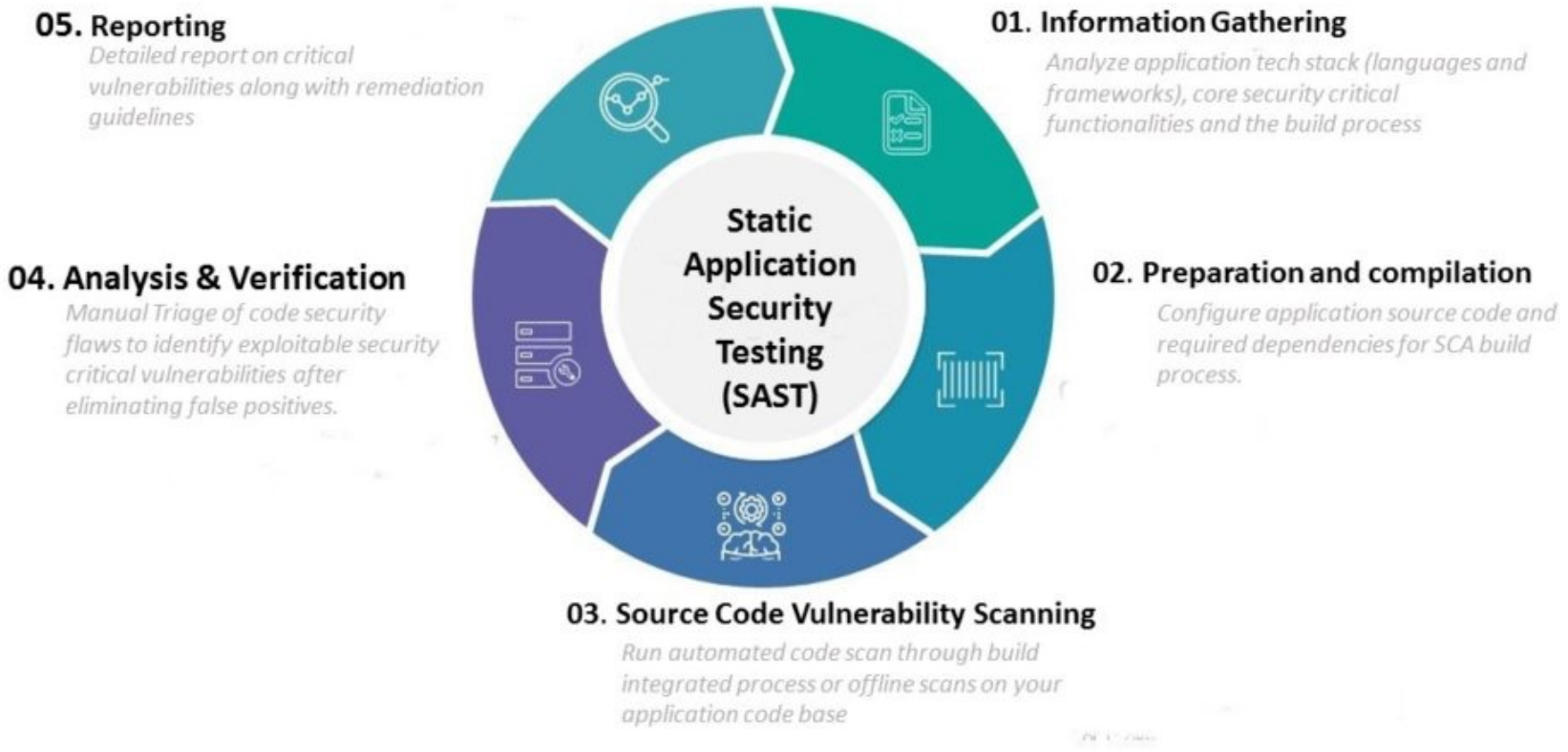
Makes an analysis on your source code and return insights about security, performance, maintainability

Fully automated and can be shift-left for developers IDE

Runs to answer by with the question “Is the code secure?”

- Is it vulnerable to injections (like SQL)?
- Does it use any weak encryption algorithms?
- Are cookies used with the right flags?

Static Application Security Testing (SAST)



SAST Tooling

Semmle/CodeQL

sonarcloud

snyk

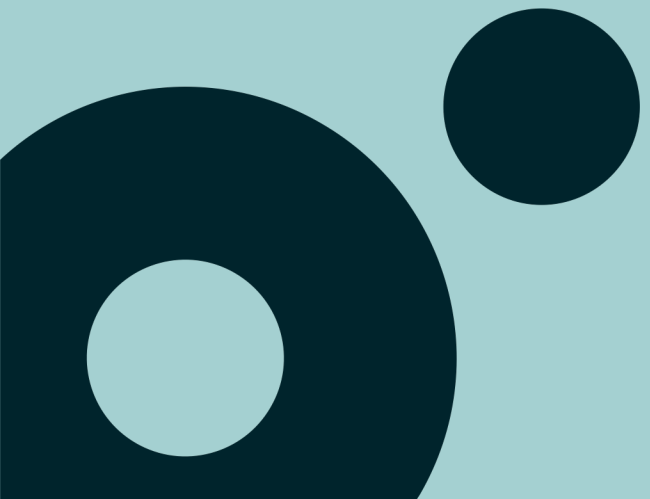
klocwork

Spectral

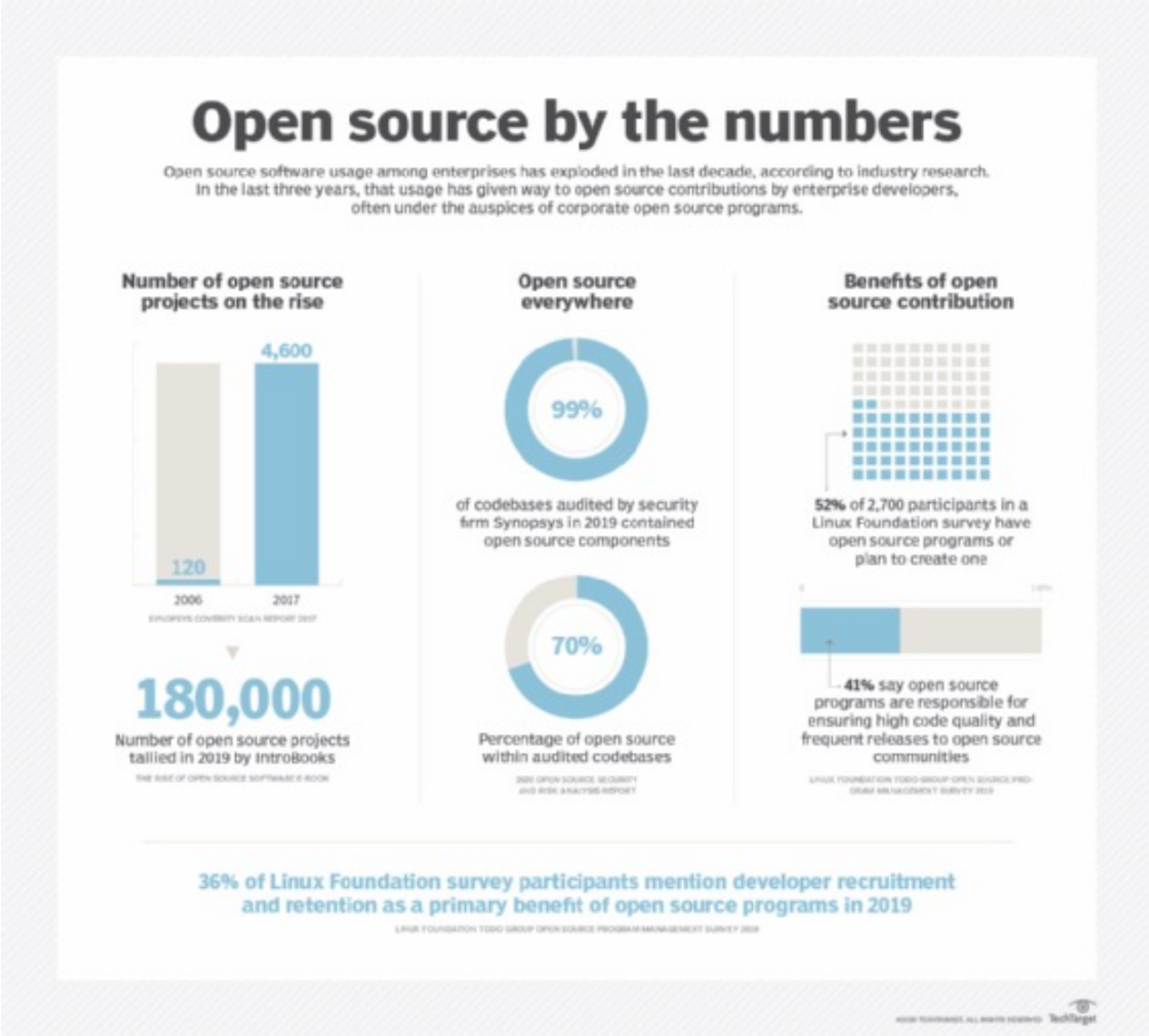
Checkmarx

https://owasp.org/www-community/Source_Code_Analysis_Tools

SCA



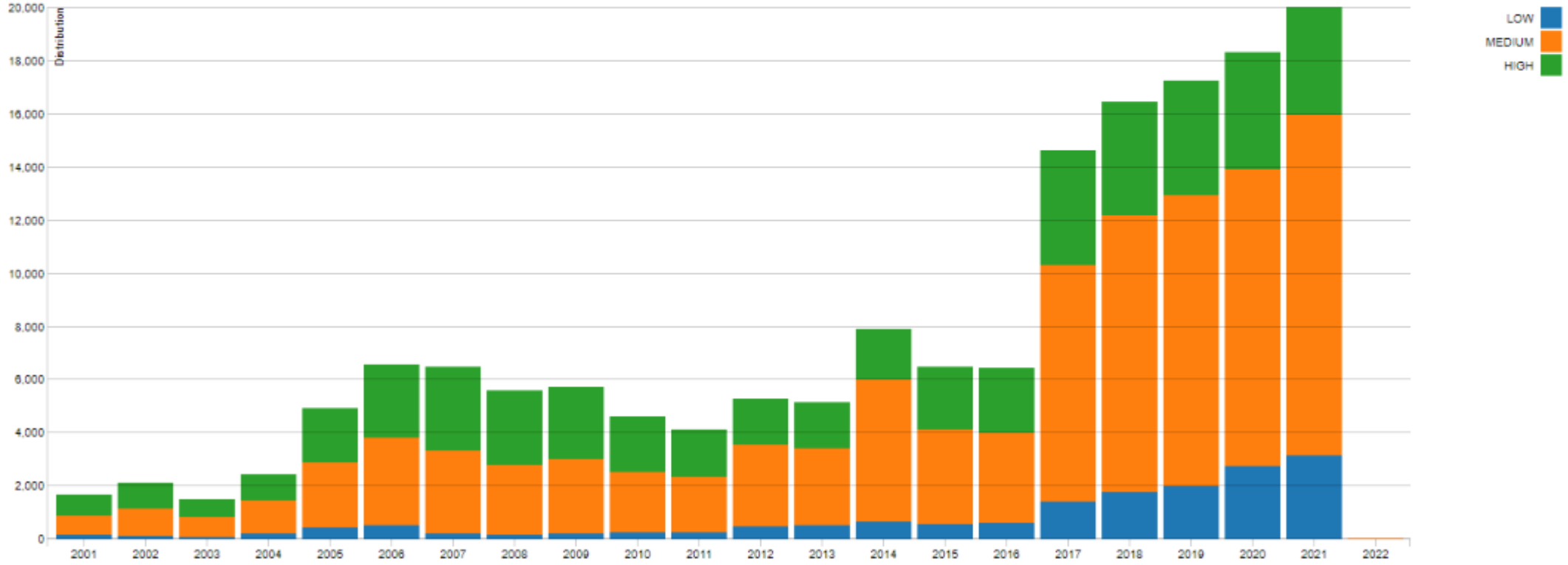
Software Composition Analysis (SCA)



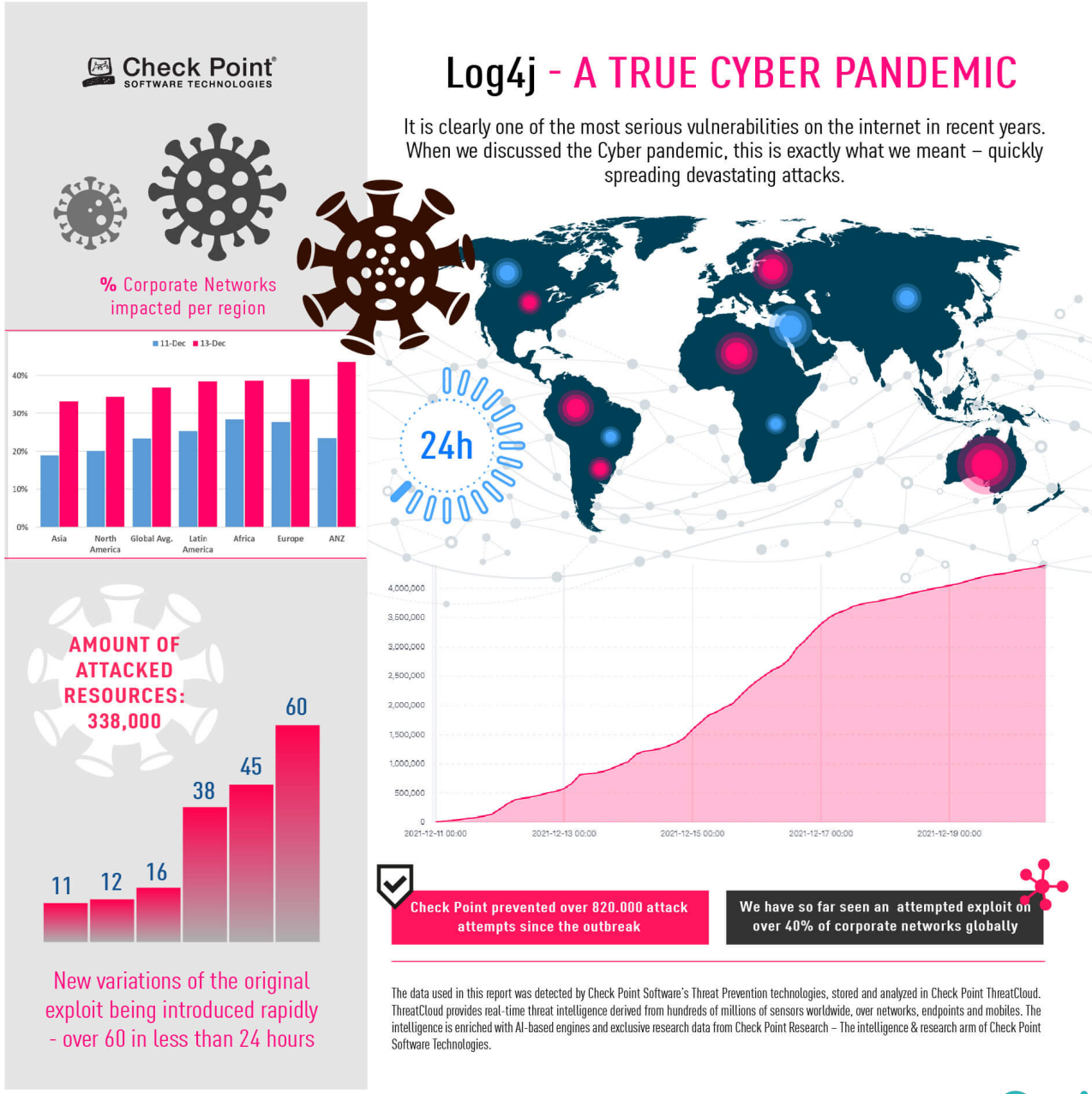
Software Composition Analysis (SCA)

CVSS Severity Distribution Over Time

This visualization is a simple graph which shows the distribution of vulnerabilities by severity over time. The choice of LOW, MEDIUM and HIGH is based upon the CVSS V2 Base score. For more information on how this data was constructed please see the NVD CVSS page .



Log4j Vulnerability



Software Composition Analysis (SCA)

So, Open Source is a bad and dangerous thing? Of course not!

But you need to use it careful and mostly you need to clearly know what are you using!

Constantly run a scan on your dependencies is crucial to understand known vulnerabilities on your supply chain

Knowing the vulnerabilities and their severity you may define you plan to fix them

Know what you're using means knowing your dependency graph! Your direct dependency have its own dependencies. That dependencies have their own dependencies and so on...

Dependency Graph

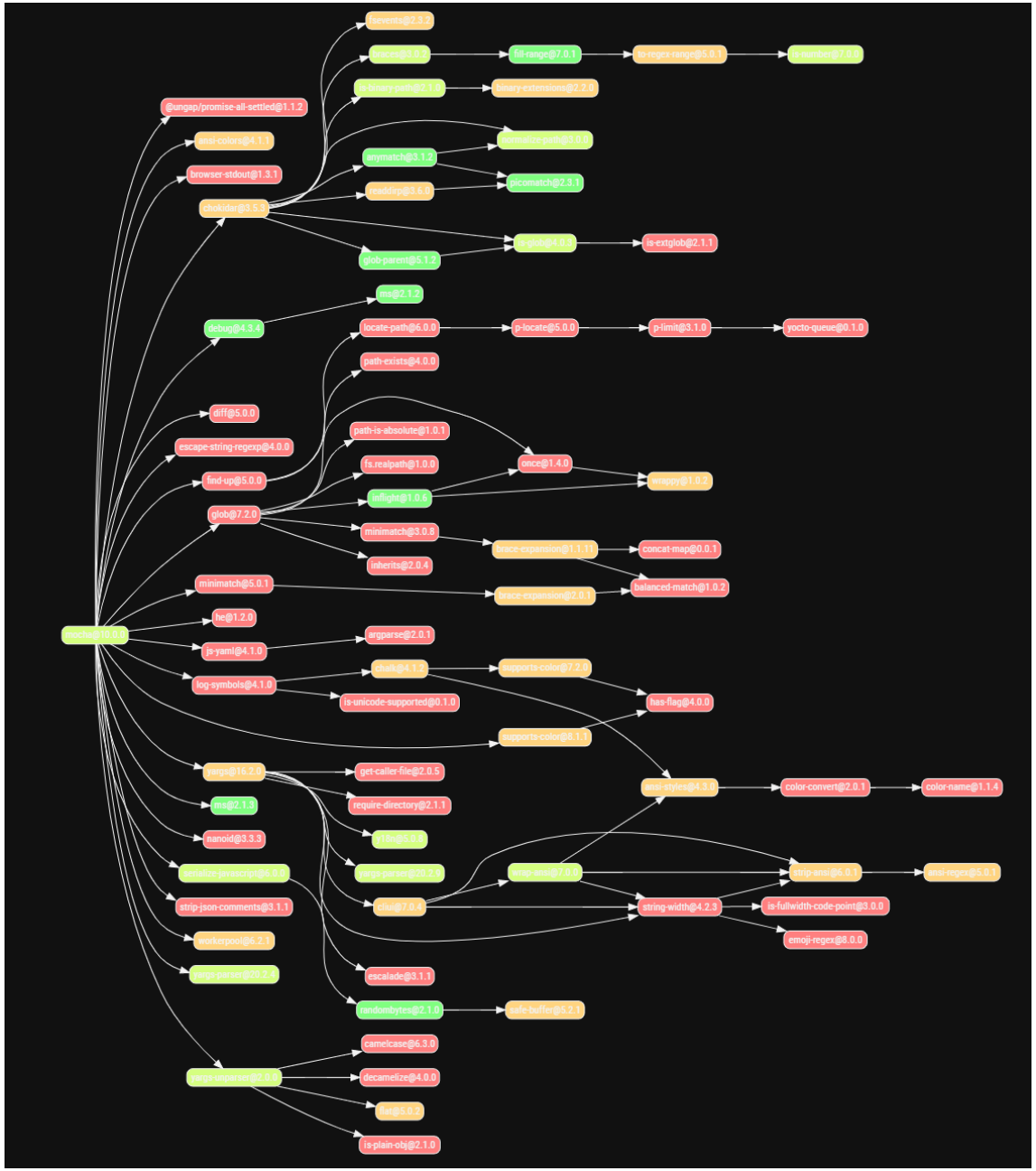
npm package Mocha

You select only one package but look to your attack surface!

Another risk is about how open source project is maintained

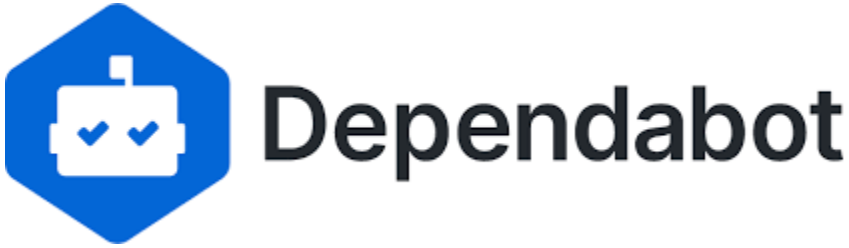
On image, red means only one maintainer.

Is a risk you may want to take, but you have to clearly know it!

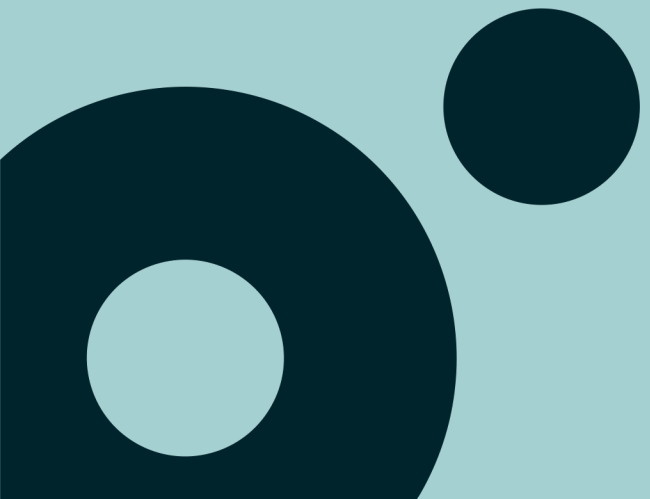


moOngy.

SDL Practices: SCA Tooling



OWASP



OWASP (Open Web Application Security Project)

OWASP is a worldwide nonprofit organization focused on improving the security of software (mainly web applications)

Provides resources and guidance to developers, security professionals, and organizations.

Organizes conferences, training sessions, and other events around the world, as well as maintains a community forum for developers and security professionals to share knowledge and best practices related to web application security

Maintains various open-source projects and tools that help identify and mitigate security vulnerabilities in web applications, including the OWASP Zed Attack Proxy (ZAP) and the OWASP Web Security Testing Guide.

OWASP TOP 10

One of the most well-known contributions of OWASP is the OWASP Top Ten Project

Provides a regularly-updated list of the top 10 web application security risks based on community feedback and research.

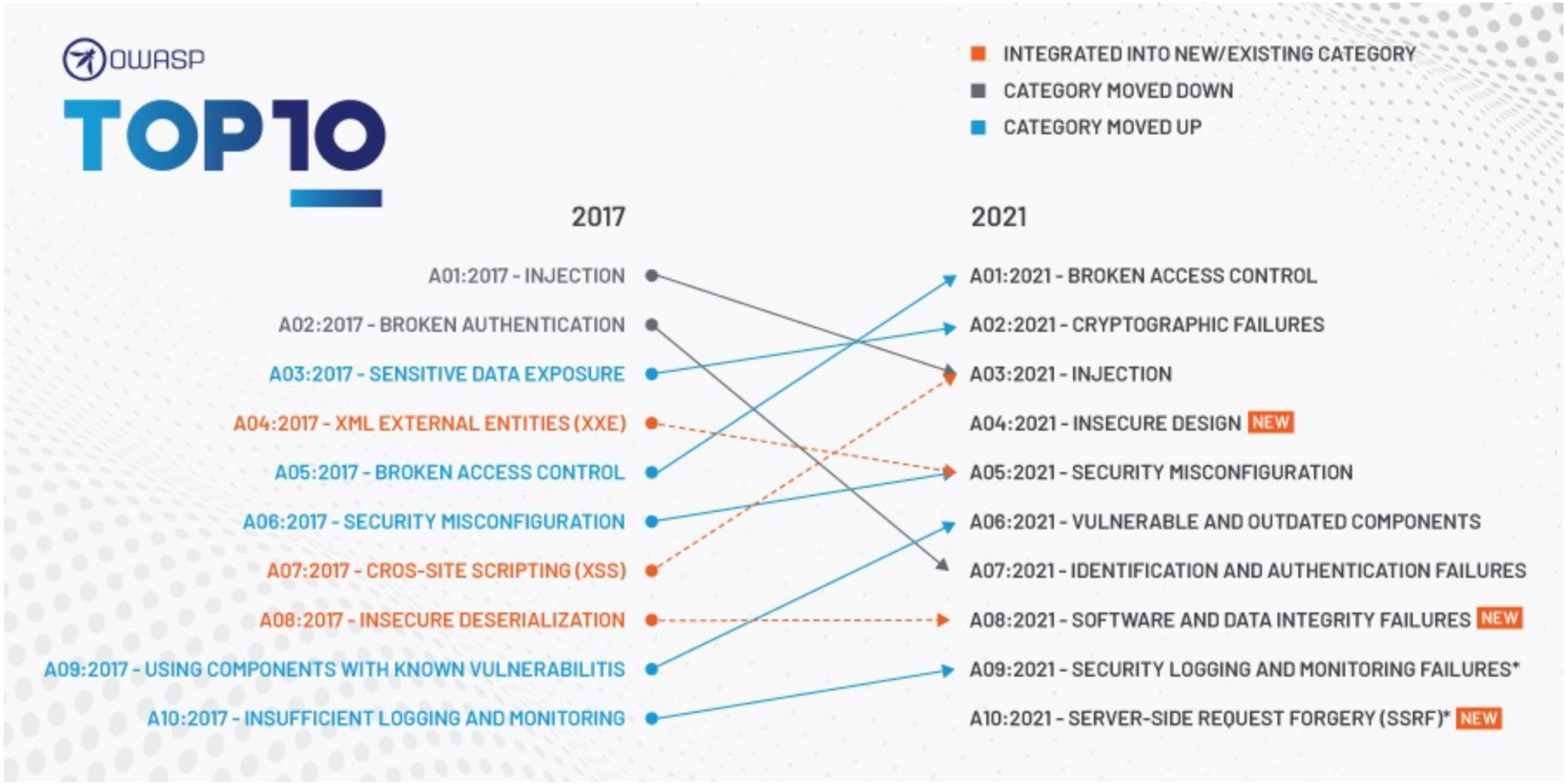
The latest version is the OWASP Top 10 2021, which includes risks such as injection flaws, broken authentication and session management, and security misconfigurations.

This Top 10 is updated regularly but is interesting to observe that the main vulnerabilities don’t change a lot since 2013

OWASP Top 10: 2013 vs 2017

OWASP TOP 10 – 2013	OWASP TOP 10 – 2017
A1 – Injection	A1 – Injection
A2 – Broken Authentication and Session Management	A2 – Broken Authentication
A3 – Cross-Site Scripting (XSS)	A3 – Sensitive Data Exposure
A4 – Insecure Direct Object References **[Merged + A7]**	A4 – XML External Entities (XXE) **[NEW]**
A5 – Security Misconfiguration	A5 – Broken Access Control **[MERGED]**
A6 – Sensitive Data Exposure	A6 – Security Misconfiguration
A7 – Missing Function Level Access Control **[Merged + A4]**	A7 – Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	A8 – Insecure Deserialization **[NEW, COMMUNITY]**
A9 – Using Components with Known Vulnerabilities	A9 – Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards	A10 – Insufficient Logging & Monitoring **[NEW, COMMUNITY]**

OWASP Top 10: 2017 vs 2021



OWASP Top Ten Web Application Security Risks | OWASP

moOngy.

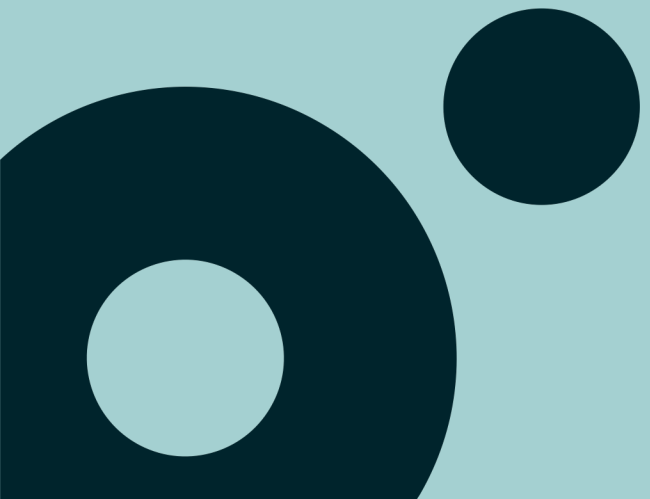
OWASP Top 10

If your application is not affected by OWASP Top 10 vulnerabilities, can you say your application is secure and solid?

OWASP Top 10

If your application is affected by any OWASP Top 10 vulnerabilities, can you say your application is insecure?

CVEs



Common Vulnerability and Exposures (CVE)

CVE is a dictionary of publicly known information security vulnerabilities and exposures that provides a standardized naming scheme for these issues

Each CVE entry contains a unique identifier, a description of the vulnerability, and references to advisories and patches

On this database, each CVE found on open-source software are published with vulnerability description and how to fix

Allow to clearly identify all known vulnerabilities on open-source code to allow you to have a more secure code

https://www.cve.org

Common Vulnerability and Exposures (CVE)

CVE-2021-45046 PUBLISHED

View JSON

Apache Log4j2 Thread Context Message Pattern and Context Lookup Pattern vulnerable to a denial of service attack

Important CVE JSON 5 Information +

Assigner: Apache
Published: 2021-12-14 **Updated:** 2022-07-25

It was found that the fix to address CVE-2021-44228 in Apache Log4j 2.15.0 was incomplete in certain non-default configurations. This could allows attackers with control over Thread Context Map (MDC) input data when the logging configuration uses a non-default Pattern Layout with either a Context Lookup (for example, $${ctx:loginId}) or a Thread Context Map pattern (%X, %mdc, or %MDC) to craft malicious input data using a JNDI Lookup pattern resulting in an information leak and remote code execution in some environments and local code execution in all environments. Log4j 2.16.0 (Java 8) and 2.12.2 (Java 7) fix this issue by removing support for message lookup patterns and disabling JNDI functionality by default.

Product Status

Learn About the Versions Section +

Vendor	Versions
Apache Software Foundation	***Default Status:*** *unknown* • affected from **Apache Log4j2** before **2.16.0**
Product	
Apache Log4j	

moOngy.

Common Weakness Enumeration (CWE)

It is a community-developed list of common software security weaknesses.

Each CWE entry provides a description of the weakness, examples of its occurrence in real-world software, and guidance on how to mitigate or eliminate it.

A CVE is defined on top of CWE (Common Weakness Enumeration) definition, being an effective instance of 1+ CWE exploit

Common Weakness Enumeration (CWE)

699 - Software Development

- API / Function Errors - *(1228)*
- Audit / Logging Errors - *(1210)*
- Authentication Errors - *(1211)*
- Authorization Errors - *(1212)*
- Bad Coding Practices - *(1006)*
- Behavioral Problems - *(438)*
- Business Logic Errors - *(840)*
- Communication Channel Errors - *(417)*
- Complexity Issues - *(1226)*
- Concurrency Issues - *(557)*
- Credentials Management Errors - *(255)*
- Cryptographic Issues - *(310)*
- Key Management Errors - *(320)*
- Data Integrity Issues - *(1214)*
- Data Processing Errors - *(19)*
- Data Neutralization Issues - *(137)*
- Documentation Issues - *(1225)*
- File Handling Issues - *(1219)*
- Encapsulation Issues - *(1227)*
- Error Conditions, Return Values, Status Codes - *(389)*
- Expression Issues - *(569)*
- Handler Errors - *(429)*
- Information Management Errors - *(199)*
- Initialization and Cleanup Errors - *(452)*
- Data Validation Issues - *(1215)*
- Lockout Mechanism Errors - *(1216)*
- Memory Buffer Errors - *(1218)*
- Numeric Errors - *(189)*
- Permission Issues - *(275)*
- Pointer Issues - *(465)*
- Privilege Issues - *(265)*
- Random Number Issues - *(1213)*
- Resource Locking Problems - *(411)*
- Resource Management Errors - *(399)*
- Signal Errors - *(387)*
- State Issues - *(371)*
- String Errors - *(133)*
- Type Errors - *(136)*
- User Interface Security Issues - *(355)*
- User Session Errors - *(1217)*

moOngy.

Common Vulnerability Scoring System (CVSS)

It is a framework for assessing the severity of vulnerabilities in software systems

CVSS assigns a score to each vulnerability based on its impact on the system's confidentiality, integrity, and availability, as well as other factors such as complexity and exploitability

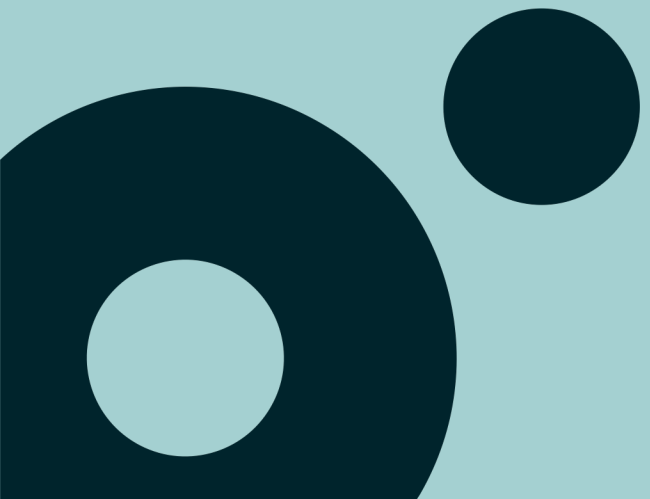
For each CVE a CVSS score is calculated granting a potential risk you're exposed

Uses 3 metrics to make the calculations: Base Metrics, Temporal Metrics and Environmental Metrics

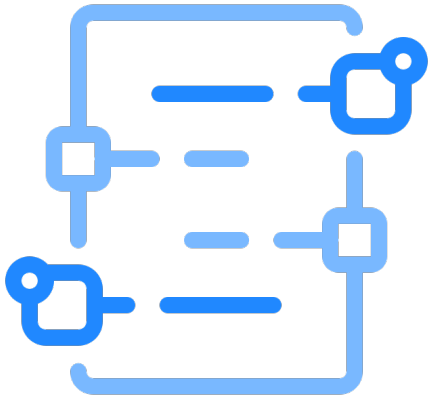
This metrics produces a values between 0-10 to define severity

CVSS Score	Qualitative Rating
0.0	None
0.1 – 3.9	Low
4.0 – 6.9	Medium
7.0 – 8.9	High
9.0 – 10.0	Critical

GitHub Advanced Security



Secure software lifecycle with GitHub



Dependency Scanning

Code Scanning

Secrets Scanning

Dependency Scanning

Dependency graph: See the packages your project depends on, the repositories that depend on them, and any vulnerabilities detected in their dependencies.

Dependabot alerts: Get notified when there are new vulnerabilities affecting your repositories. GitHub detects and alerts users to vulnerable dependencies in public and private repos.

Dependabot security and version updates: Keep your supply chain secure and up-to-date by automatically opening pull requests that update vulnerable or out-of-date dependencies.

About supply chain security - GitHub Docs

Code Scanning

Find and fix vulnerabilities fast, before they are merged into the code base with automated CodeQL scans.

Community of top security experts produce CodeQL queries to empower every project with a world-class security team. You can even create your own custom queries.

Integrated with developer workflow for a frictionless experience and faster development, beginning with IDE integration and automate on GitHub Actions

Extensible as you may plug other SAST tools into the same developer workflow.

Secret Scanning

Identifies secrets as early as possible, since the moment they are pushed to GitHub and immediately notifies developers when they are found. Scan your entire git history

Community of secret scanning partners, for every commit made to your repository, and its full git history, we'll look for secret formats from secret scanning partners

Secret scanning watches both **public and private repos** for potential secret vulnerabilities.

Protection from exposed secrets, automatically disable or suspend secrets from 100+ service providers as soon as they are committed

Demo: GitHub Advanced Security

Lab

Lab 3: Secure DevOps

Learning Objectives

- Create a workflow to build and test your code
- Use Pull Request to validate your code
- Run Continuous Integration workflow
- Manage your local repo

Markdown version: https://github.com/tasb/devops-with-github-training/blob/main/labs/lab03.md

HTML version: https://tasb.github.io/devops-with-github-training/labs/lab03.html

moOngy.
Minds on the move

Rua Sousa Martins, nº 10

1050-218 Lisboa | Portugal