



# Build secure software with OWASP tools and guides



**OWASP**

The Open Web Application Security Project



# OWASP

The Open Web Application Security Project



## Martin Knobloch

+10 years developer experience

+15 years information security experience

Dutch OWASP Chapter Leader since 2007

OWASP Foundation Board of Directors since 2017

Global AppSec Strategist @ Micro Focus / Fortify

### Contact:

[martin.knobloch@owasp.org](mailto:martin.knobloch@owasp.org)

<https://twitter.com/knoblochmartin>

<https://www.linkedin.com/in/martin-knobloch>

<https://xing.to/knoblochmartin>



# OWASP

The Open Web Application Security Project

Please support the OWASP mission to improve software security through open source initiatives and community education. [Donate Now!](#)

SEARCH OWASP  [Donate](#) [Join](#)

PROJECTS CHAPTERS EVENTS ABOUT



**Who is the OWASP® Foundation?**

The Open Web Application Security Project® (OWASP) is a nonprofit foundation that works to improve the security of software. Through community-led open source software projects, hundreds of local chapters worldwide, tens of thousands of members, and leading educational and training conferences, the OWASP Foundation is the source for developers and technologists to secure the web.

- Tools and Resources
- Community and Networking
- Education & Training

For nearly two decades corporations, foundations, developers, and volunteers have supported the OWASP Foundation and its work. [Donate](#), [Join](#), or become a [Corporate Member](#) today.

**Project Spotlight: Mobile Security Testing Guide**



The OWASP Mobile Security Testing Guide (MSTG) is a comprehensive manual for mobile app security testing and reverse engineering for the iOS and Android platforms, describing technical processes for verifying the controls listed in the MSTG's co-project

Mobile Application Verification Standard (MASVS). The MASVS defines a mobile app security model and lists generic security requirements for mobile apps, while the MSTG serves as a baseline for manual security testing and as a template for automated security tests during or after development. Included with the MSTG, the Mobile Security Hacking Playground is a collection of iOS and Android mobile apps that are intentionally built insecure. These apps are used as examples to demonstrate different vulnerabilities explained in the

**Featured Chapter: Bay Area**



Hosted at some of most iconic technology companies in the world, the Bay Area chapter is one of the Foundation's largest and most active. This month they are hosting a Hacker Day and monthly meetups in San Francisco at Insight Engines and in South Bay at eBay. Usually the agenda includes three proactive and interesting talks, lots of interesting people to meet, and great food. The Bay Area Chapter also participates in planning AppSec California.



# OWASP

The Open Web Application Security Project

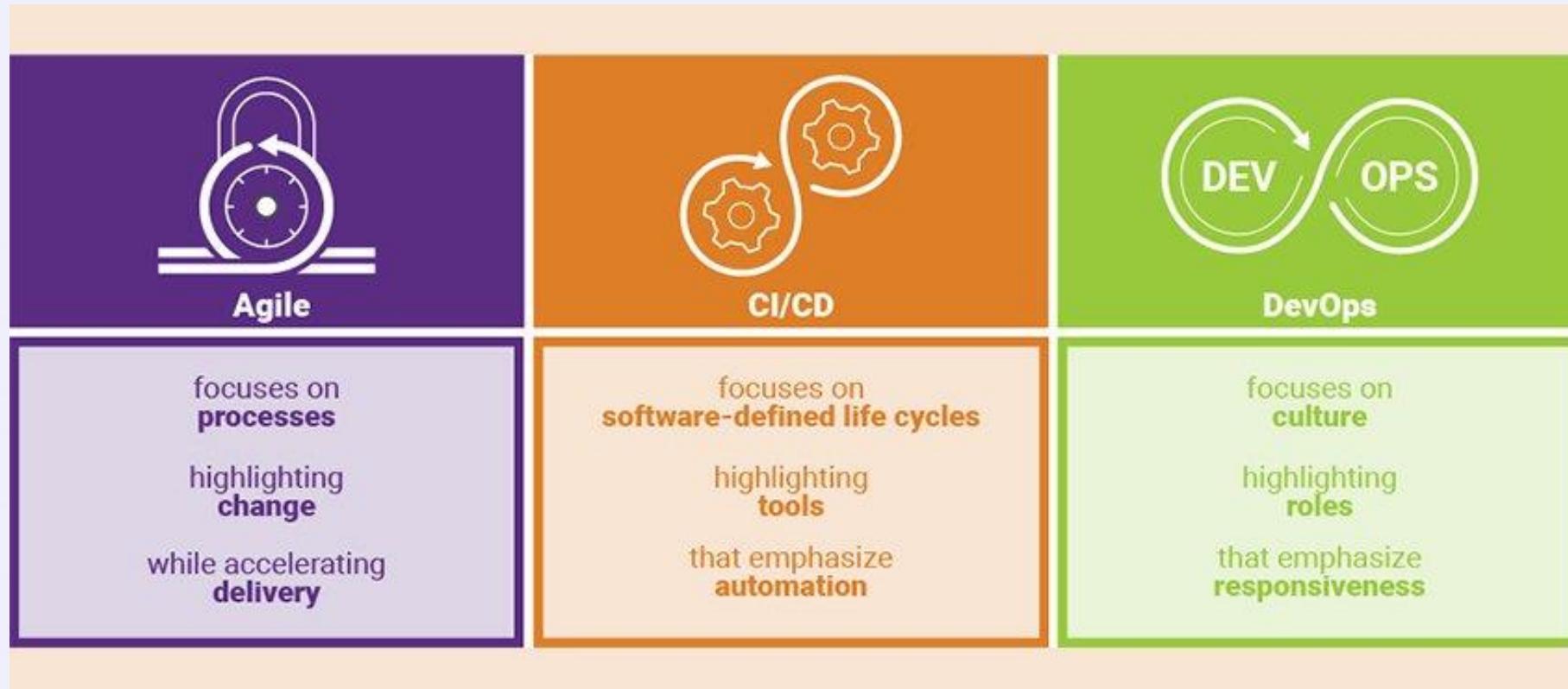
- to make application security "visible," so that people and organizations can make informed decisions about application security risks





# OWASP

The Open Web Application Security Project



# OWASP Guide for CISOs



# OWASP

The Open Web Application Security Project



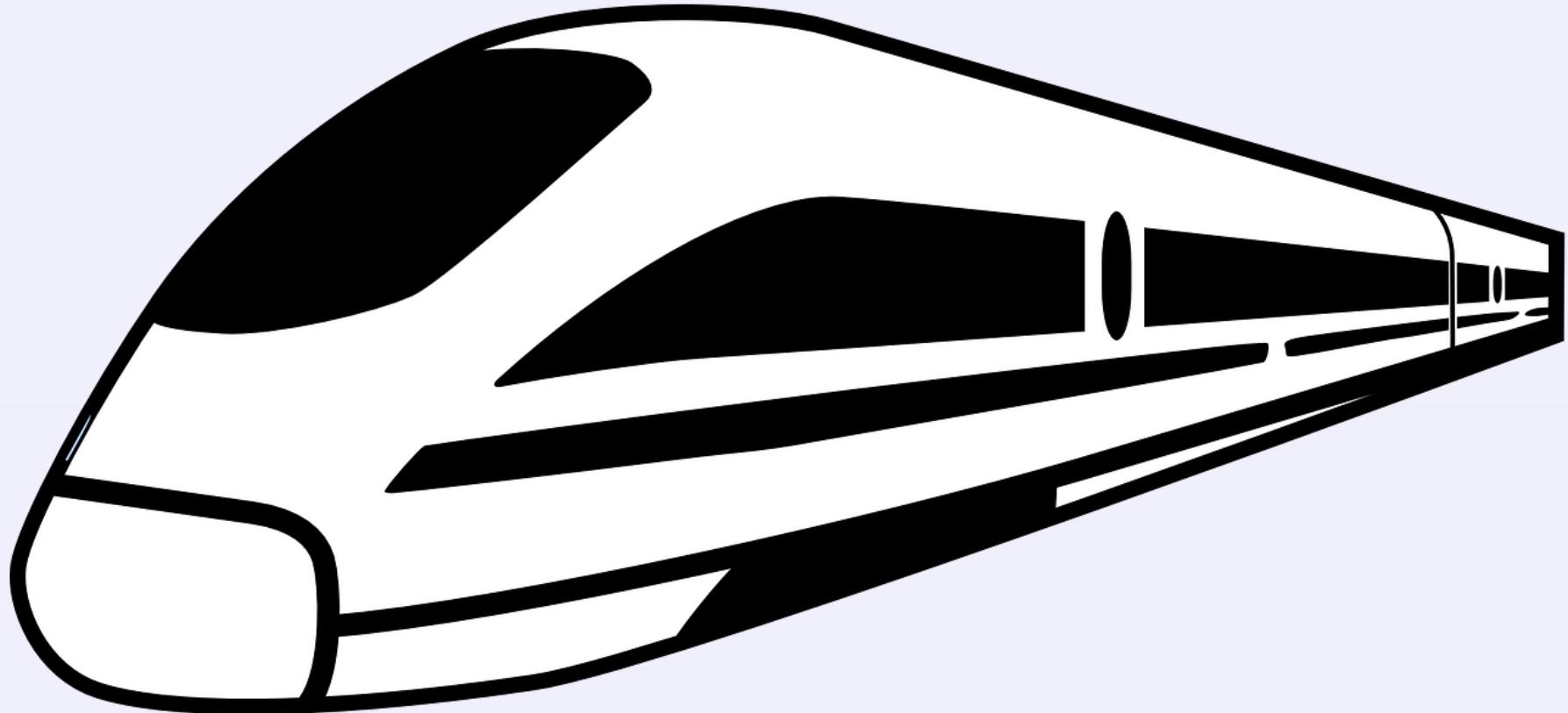
[https://www.owasp.org/index.php/Application\\_Security\\_Guide\\_For\\_CISOs](https://www.owasp.org/index.php/Application_Security_Guide_For_CISOs)

Faster towards production



**OWASP**

The Open Web Application Security Project





# OWASP

The Open Web Application Security Project

OWASP Top 10 – 2013 (Previous)	OWASP Top 10 – 2017 (New)
A1 – Injection	A1 – Injection
A2 – Broken Authentication and Session Management	A2 – Broken Authentication and Session Management
A3 – Cross-Site Scripting (XSS)	A3 – Cross-Site Scripting (XSS)
A4 – Insecure Direct Object References - Merged with A7	A4 – Broken Access Control (Original category in 2003/2004)
A5 – Security Misconfiguration	A5 – Security Misconfiguration
A6 – Sensitive Data Exposure	A6 – Sensitive Data Exposure
A7 – Missing Function Level Access Control - Merged with A4	A7 – Insufficient Attack Protection (NEW)
A8 – Cross-Site Request Forgery (CSRF)	A8 – Cross-Site Request Forgery (CSRF)
A9 – Using Components with Known Vulnerabilities	A9 – Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards - Dropped	A10 – Underprotected APIs (NEW)

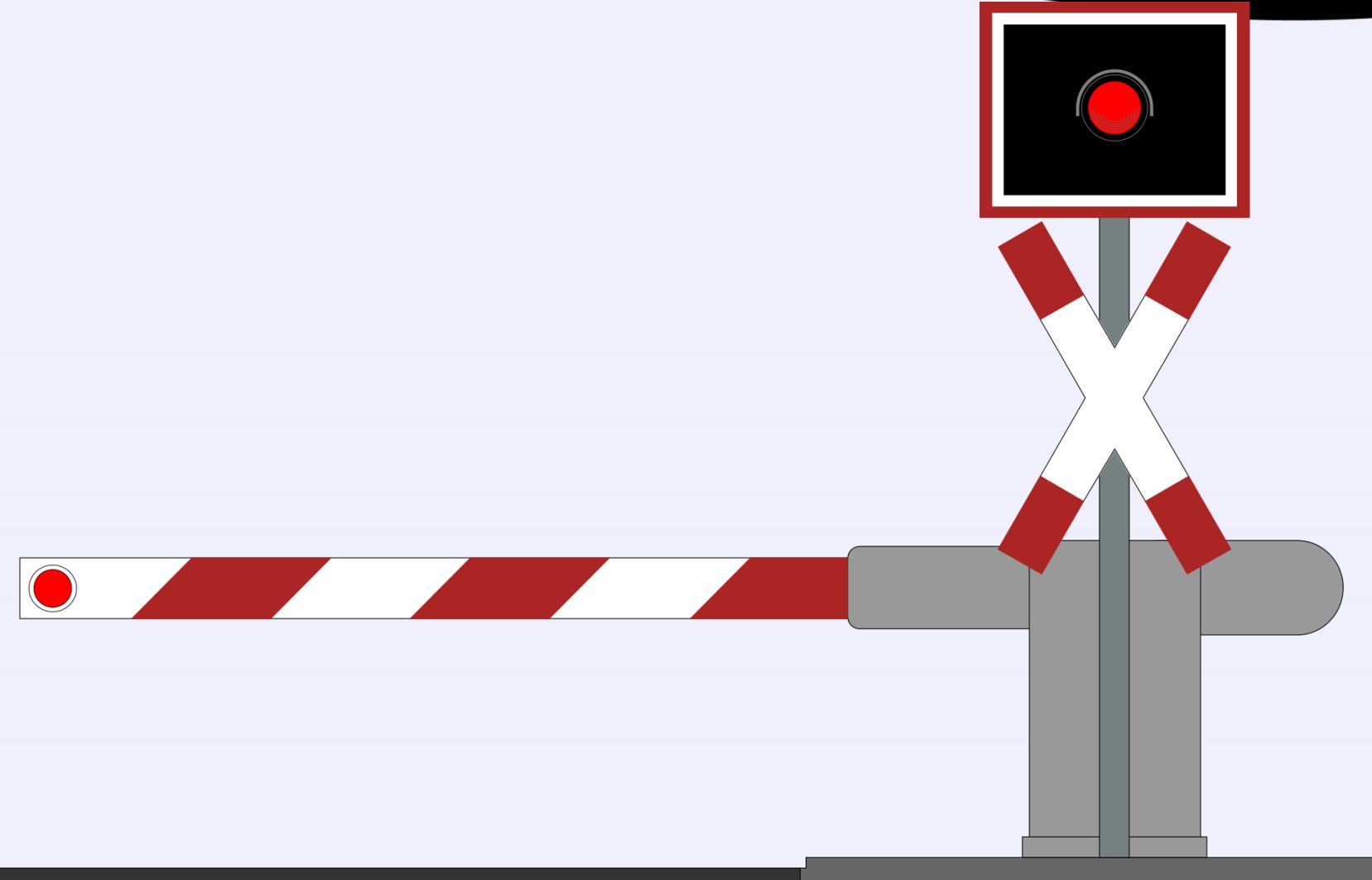
<https://owasp.org/www-project-top-ten/>

..but



# OWASP

The Open Web Application Security Project



..Fail?



# OWASP

The Open Web Application Security Project

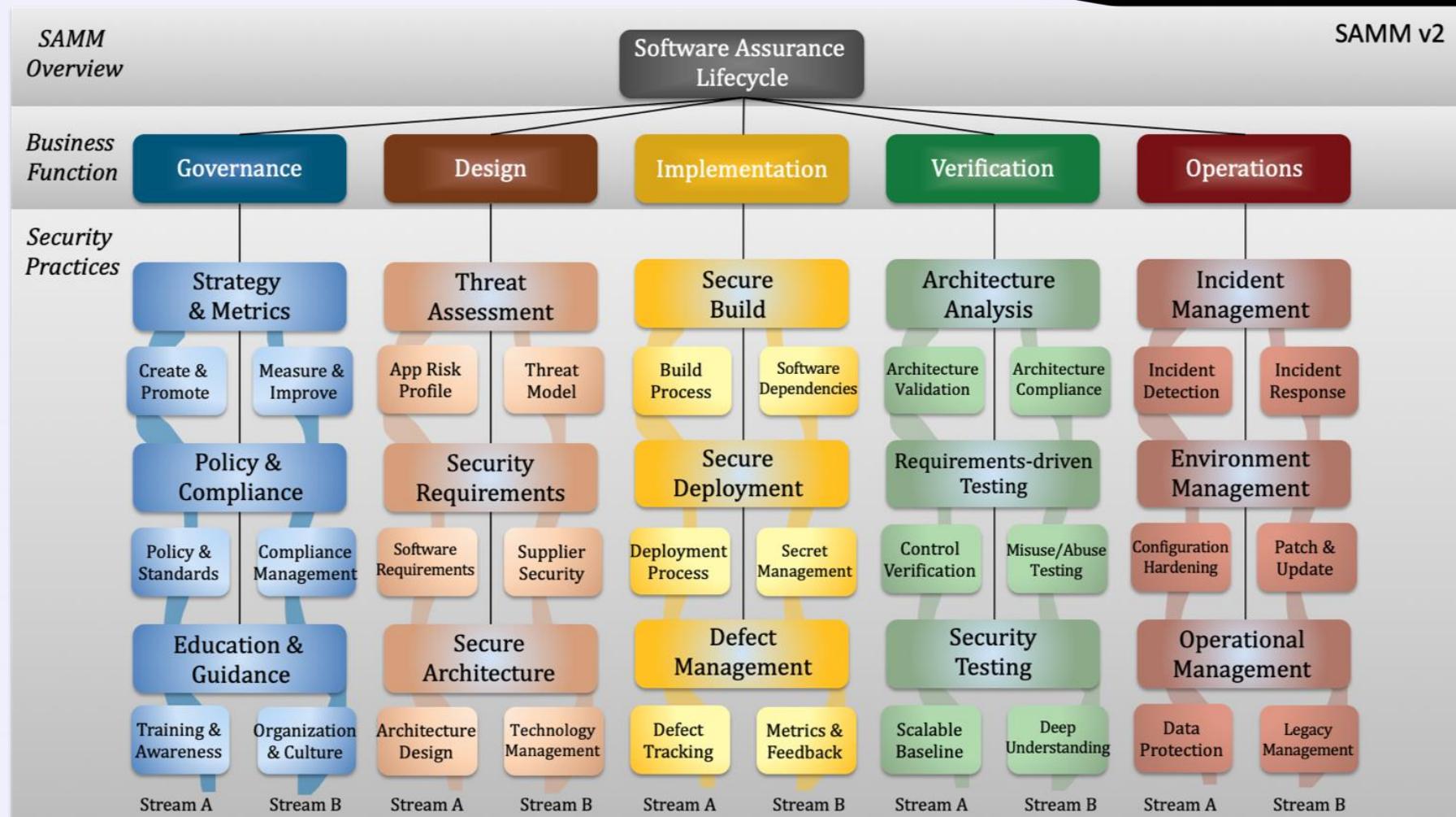




# OWASP

The Open Web Application Security Project







# OWASP

The Open Web Application Security Project



C1: Define Security Requirements

C2: Leverage Security Frameworks and Libraries

C3: Secure Database Access

C4: Encode and Escape Data

C5: Validate All Input

C6: Implement Digital Identity

C7: Enforce Access Controls

C8: Protect Data Everywhere

C9: Implement Security Logging and Monitoring

C10: Handle All Errors and Exceptions

<https://owasp.org/www-project-proactive-controls/>

# Cheat Sheet Series



# OWASP

The Open Web Application Security Project



Life is too short • AppSec is tough • Cheat!

The **OWASP Cheat Sheet Series** was created to provide a concise collection of high value information on specific application security topics. These cheat sheets were created by various application security professionals who have expertise in specific topics.

<https://cheatsheetseries.owasp.org/>

<https://owasp.org/www-project-cheat-sheets/>



# OWASP

The Open Web Application Security Project



V1: Architecture,  
Design and Threat  
Modeling  
Requirements

V2: Authentication  
Verification  
Requirements

V3: Session  
Management  
Verification  
Requirements

V4: Access Control  
Verification  
Requirements

V5: Validation,  
Sanitization and  
Encoding Verification  
Requirements

V6: Stored  
Cryptography  
Verification  
Requirements

V7: Error Handling and  
Logging Verification  
Requirements

V8: Data Protection  
Verification  
Requirements

V9: Communication  
Verification  
Requirements

V10: Malicious Code  
Verification  
Requirements

V11: Business Logic  
Verification  
Requirements

V12: File and  
Resources Verification  
Requirements

V13: API and Web  
Service Verification  
Requirements

V14: Configuration  
Verification  
Requirements

<https://owasp.org/www-project-application-security-verification-standard/>



## Requirements

#	Description	1	2	3	Since
4.1	Verify that the principle of least privilege exists - users should only be able to access functions, data files, URLs, controllers, services, and other resources, for which they possess specific authorization. This implies protection against spoofing and elevation of privilege.	✓	✓	✓	1.0
4.4	Verify that access to sensitive records is protected, such that only authorized objects or data is accessible to each user (for example, protect against users tampering with a parameter to see or alter another user's account).	✓	✓	✓	1.0
4.5	Verify that directory browsing is disabled unless deliberately desired. Additionally, applications should not allow discovery or disclosure of file or directory metadata, such as Thumbs.db, .DS_Store, .git or .svn folders.	✓	✓	✓	1.0
4.8	Verify that access controls fail securely.	✓	✓	✓	1.0
4.9	Verify that the same access control rules implied by the presentation layer are enforced on the server side.	✓	✓	✓	1.0
4.10	Verify that all user and data attributes and policy information used by access controls cannot be manipulated by end users unless specifically authorized.		✓	✓	1.0
4.11	Verify that there is a centralized mechanism (including libraries that call external authorization services) for protecting access to each type of protected resource.			✓	1.0
4.12	Verify that all access control decisions can be logged and all failed decisions are logged.	✓	✓		2.0

<https://owasp.org/www-project-application-security-verification-standard/>

Cornucopia



# OWASP

The Open Web Application Security Project



<https://owasp.org/www-project-cornucopia/>

# Security RAT



# OWASP

The Open Web Application Security Project

Security  
RAT

The screenshot shows the GitHub organization page for "SecurityRAT". The header features the OWASP logo and the text "SecurityRAT". Below the header, there are three repository cards:

- SecurityRAT**: A tool for handling security requirements in development, written in JavaScript, updated 21 days ago.
- securityrat.github.io**: Documentation for the SecurityRAT project, written in CSS, updated on 31 Aug.
- Security-Requirements**: Initial set of security requirements provided as a MySQL dump, written in PLSQL, updated on 9 May.

On the right side of the page, there's a "Top languages" section showing JavaScript, CSS, and PLSQL, and a "People" section which notes that the organization has no public members.

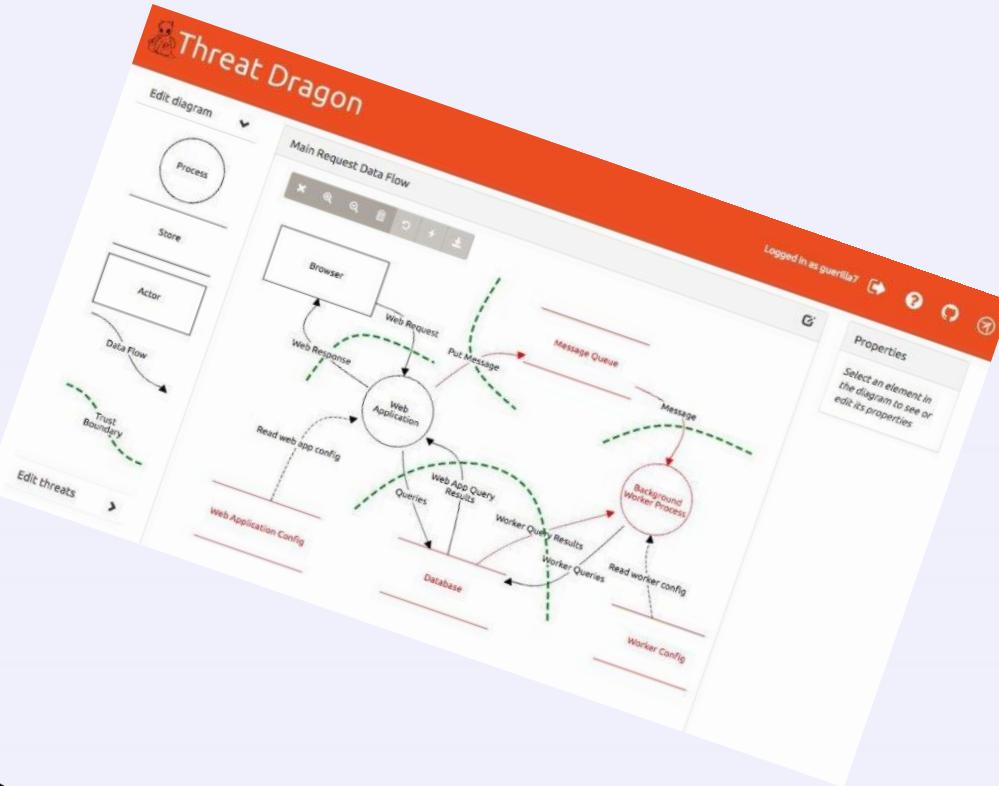
<https://owasp.org/www-project-securityrat/>

# Threat Dragon



# OWASP

The Open Web Application Security Project



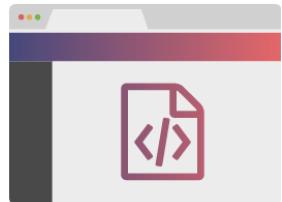
<https://owasp.org/www-project-threat-dragon/>

# Security Knowledge Framework



# OWASP

The Open Web Application Security Project



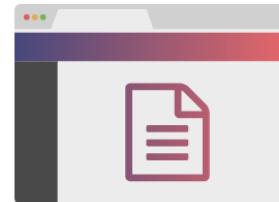
## Detect possible threats in your application

In pre-development detect possible threats based on the processing functions on your application.



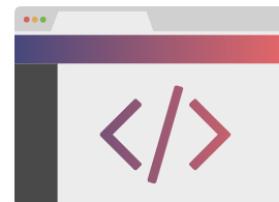
## Learn about threats and vulnerabilities in the SKF knowledge base

An extensive library of common hacks and exploits, learn the hacker mindset and keep your project secure.



## Run OWASP ASVS Checklists

Harden your application functions in post-development by running OWASP ASVS checklists, complete with feedback and solutions.



## Learn to code secure from best practice code examples

An extensive library of code examples for a wide range of functions, beautifully commented.



<https://www.securityknowledgeframework.org/>  
<https://owasp.org/www-project-security-knowledge-framework/>

# Dependency Check



# OWASP

The Open Web Application Security Project



<https://owasp.org/www-project-dependency-check/>

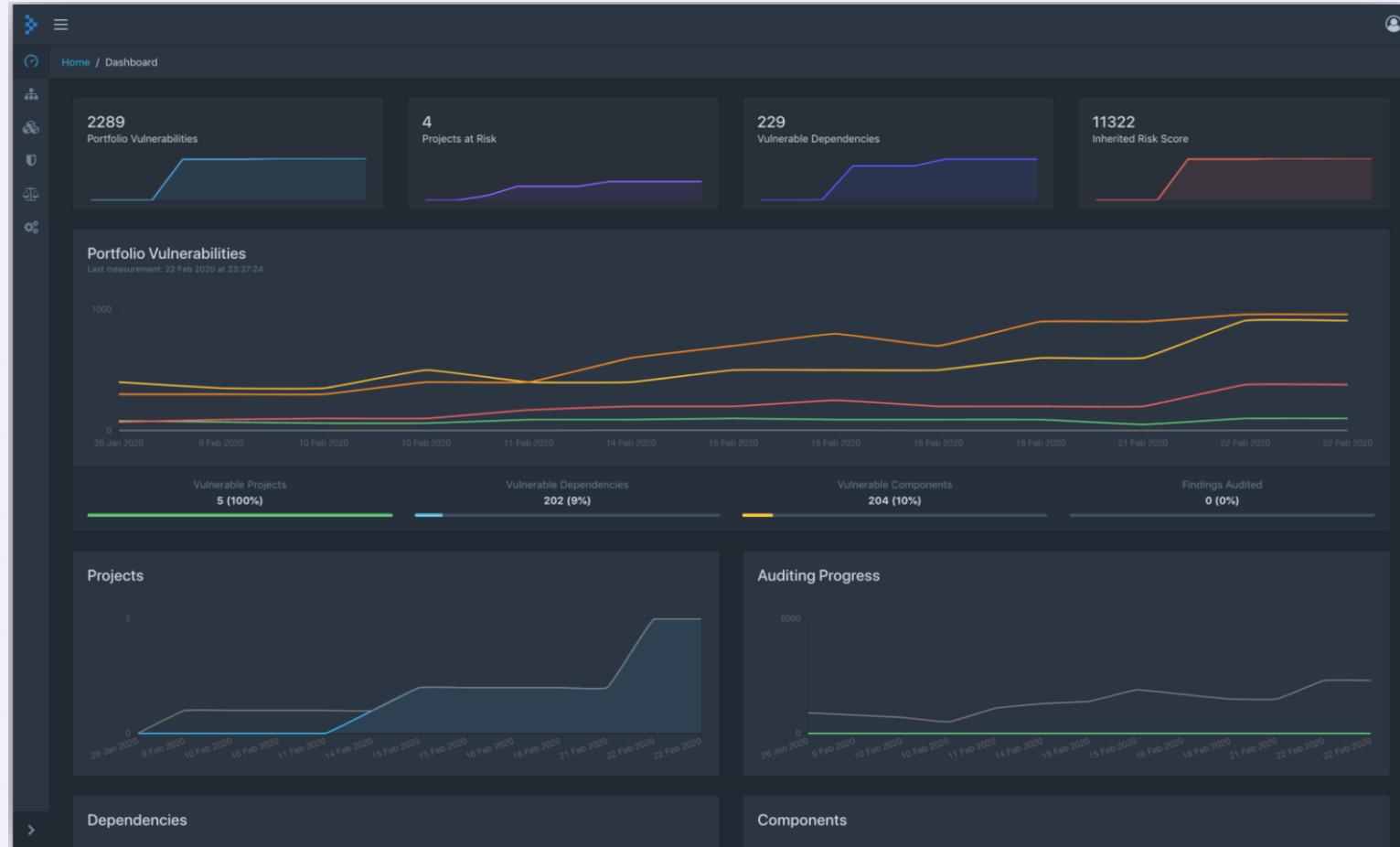
<https://owasp.org/www-project-dependency-track/> - <https://docs.dependencytrack.org/>

# Dependency Track



# OWASP

The Open Web Application Security Project



<https://owasp.org/www-project-dependency-check/>

<https://owasp.org/www-project-dependency-track/> - <https://docs.dependencytrack.org/>

# OWASP Testing Guide



# OWASP

The Open Web Application Security Project



<https://owasp.org/www-project-web-security-testing-guide/>

<https://owasp.org/www-project-mobile-security-testing-guide/>



# OWASP

The Open Web Application Security Project

**Information  
Gathering**

**Configuration and  
Deploy Management  
Testing**

**Identity Management  
Testing**

**Authentication  
Testing**

**Authorization Testing**

**Session Management  
Testing**

**Input Validation  
Testing**

**Testing for Error  
Handling**

**Testing for Weak  
Cryptography**

**Business Logic Testing**

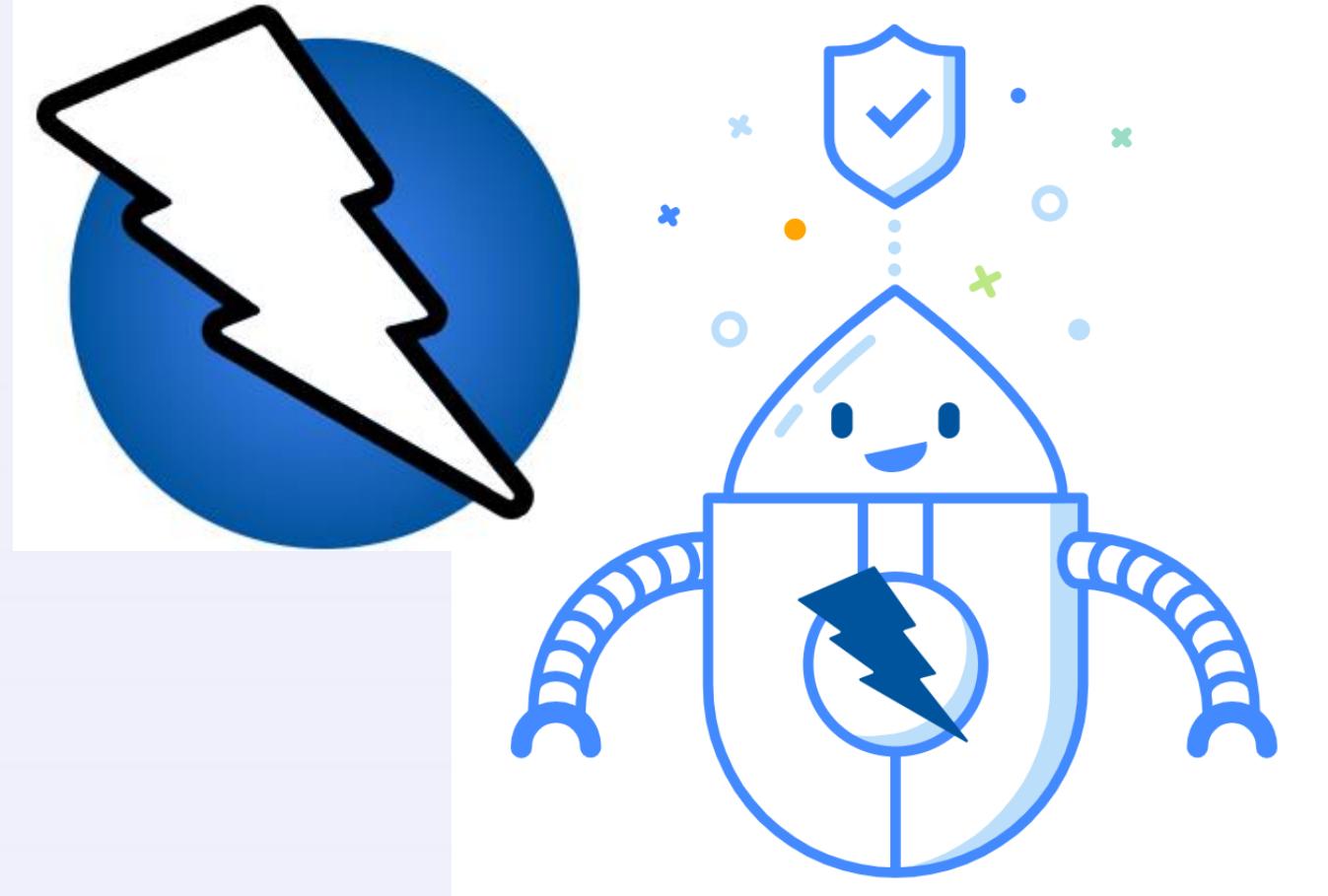
**Client-Side Testing**

**API Testing**



# OWASP

The Open Web Application Security Project



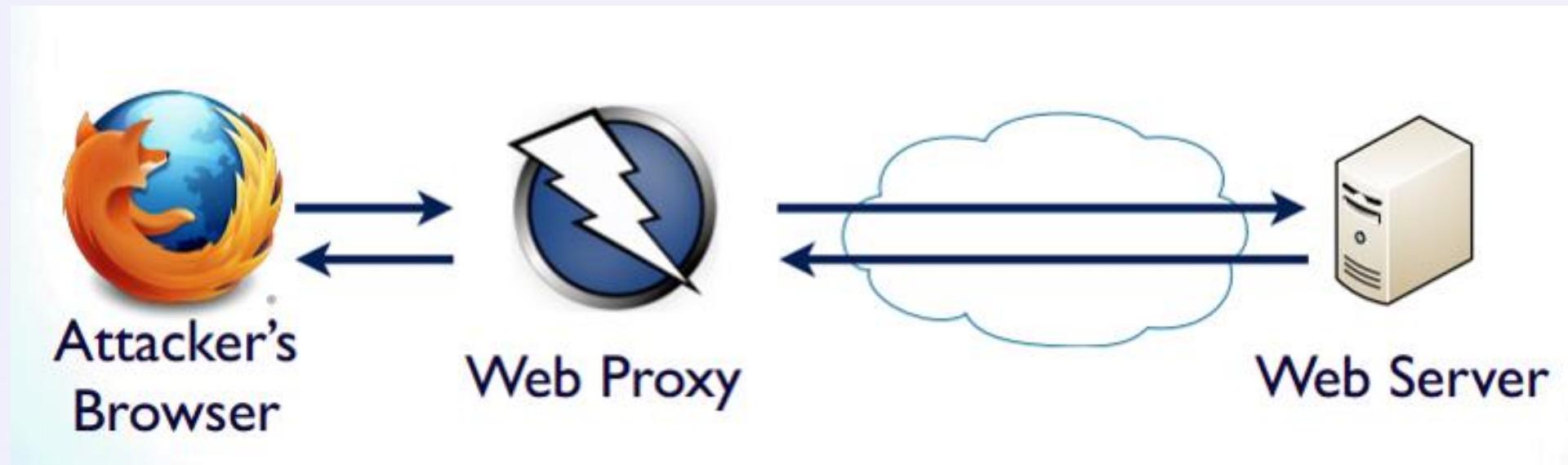
<https://owasp.org/www-project-zap/>

<https://www.zaproxy.org/>



# OWASP

The Open Web Application Security Project



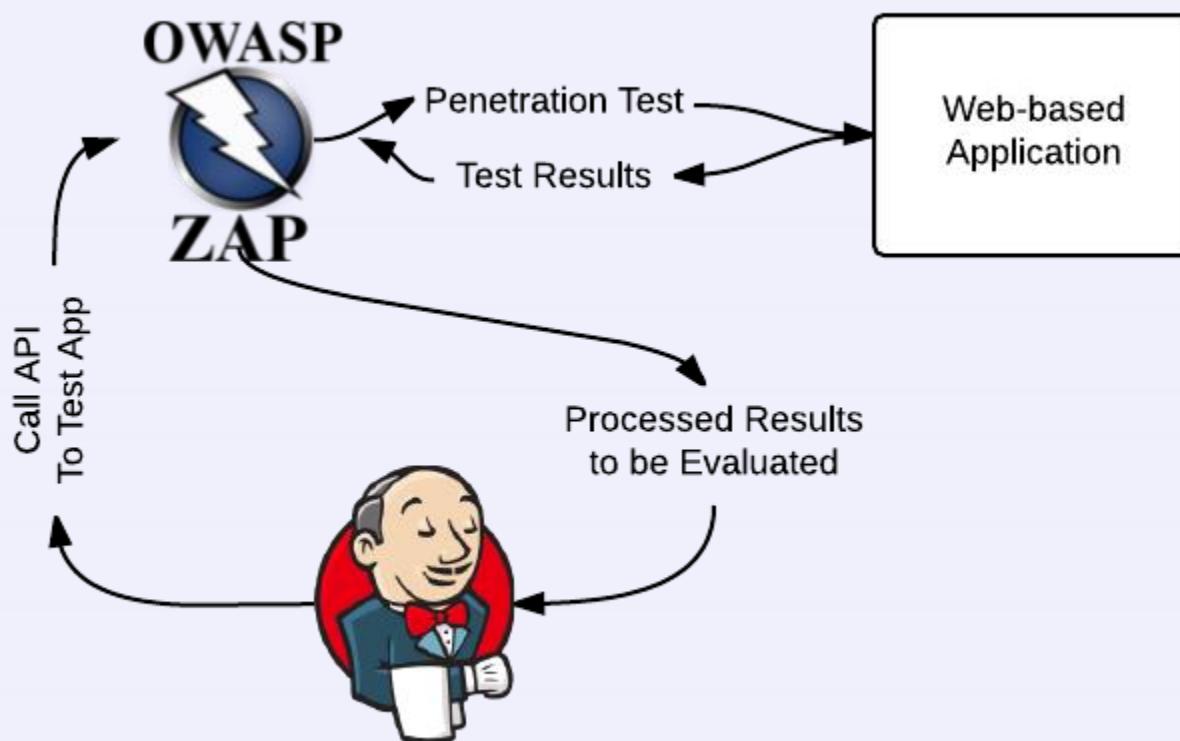
<https://owasp.org/www-project-zap/>

<https://www.zaproxy.org/>



# OWASP

The Open Web Application Security Project



<https://owasp.org/www-project-zap/>

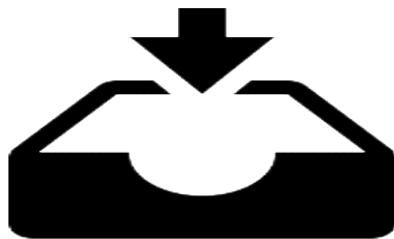
<https://www.zaproxy.org/>



# OWASP

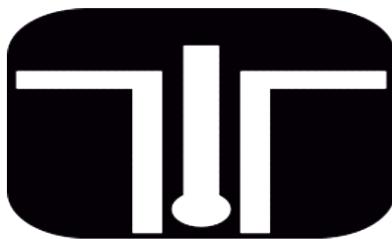
The Open Web Application Security Project

## Stages of an AppSec Pipeline



The first stage of an AppSec Pipeline which handles inbound requests of the AppSec program. These can be new apps, existing apps that have never been assessed, apps which have been assessed before or retesting of previous security findings. These tools aim to tame the inflow of work into the AppSec Pipeline.

INTAKE TOOLS



The second stage of an AppSec Pipeline which prioritizes inbound requests and assesses their testing needs based on the risk level. The more risky the app, the more activities are assigned. These tools aim to provide automation and orchestration to reduce the startup time of the testing stage.

TRIAGE TOOLS



The third stage of an AppSec Pipeline which runs one or more tests in parallel to assess the security posture of an application. Ideally, these testing or at least their setup should be automated. Priority should be given to tools that can be run programmatically and produce results with few false positives.

TEST TOOLS



The forth and final stage of an AppSec Pipeline which collects and normalizes the data created during testing. Any duplicate findings should be removed so that the same issue found by multiple tools is only reported once. Here we link to issue tracking systems, produce reports, and otherwise provide data for stakeholders.

DELIVERY TOOLS

<https://owasp.org/www-project-appsec-pipeline/>

<https://www.appsecpipeline.org/>

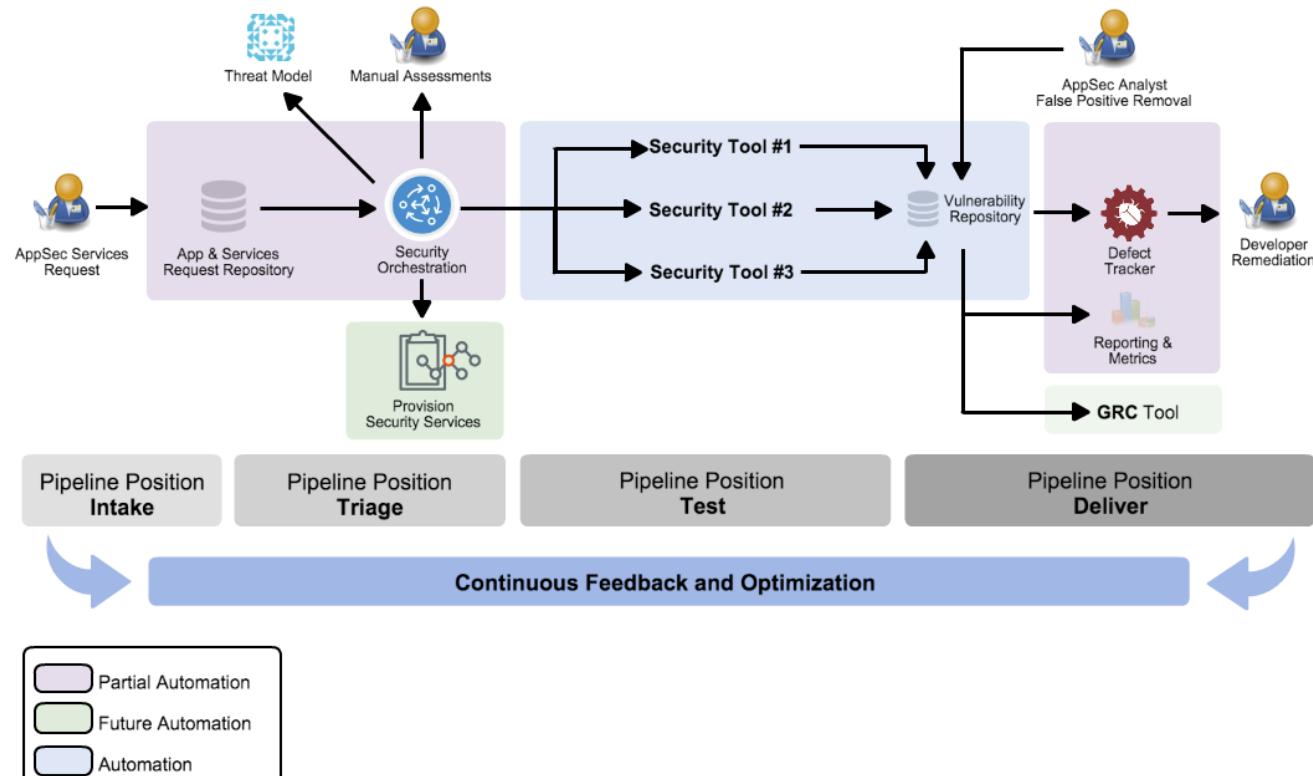
# AppSec Pipeline



# OWASP

The Open Web Application Security Project

## Rugged Devops - AppSec Pipeline Template



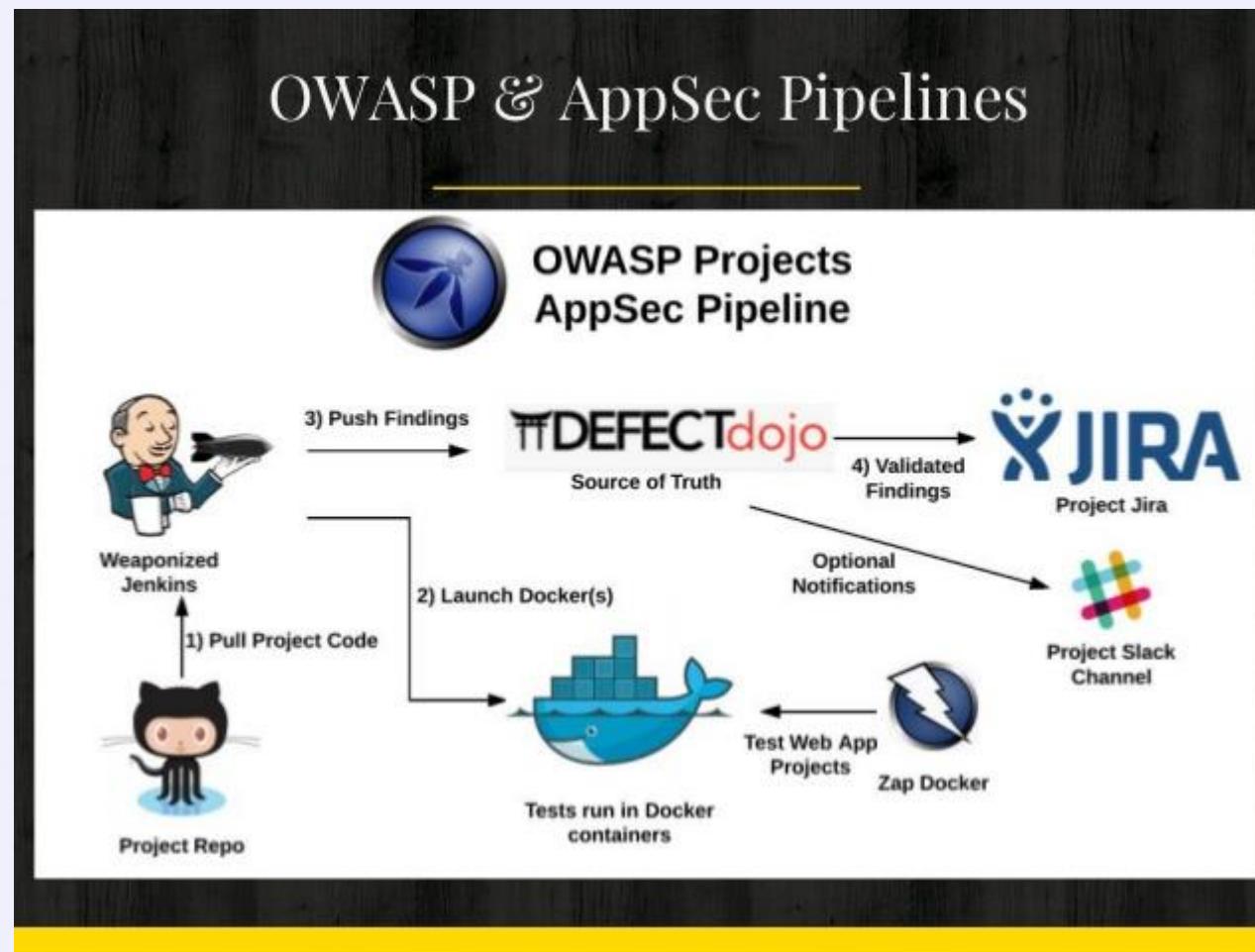
<https://owasp.org/www-project-appsec-pipeline/>

<https://www.appsecpipeline.org/>



# OWASP

The Open Web Application Security Project

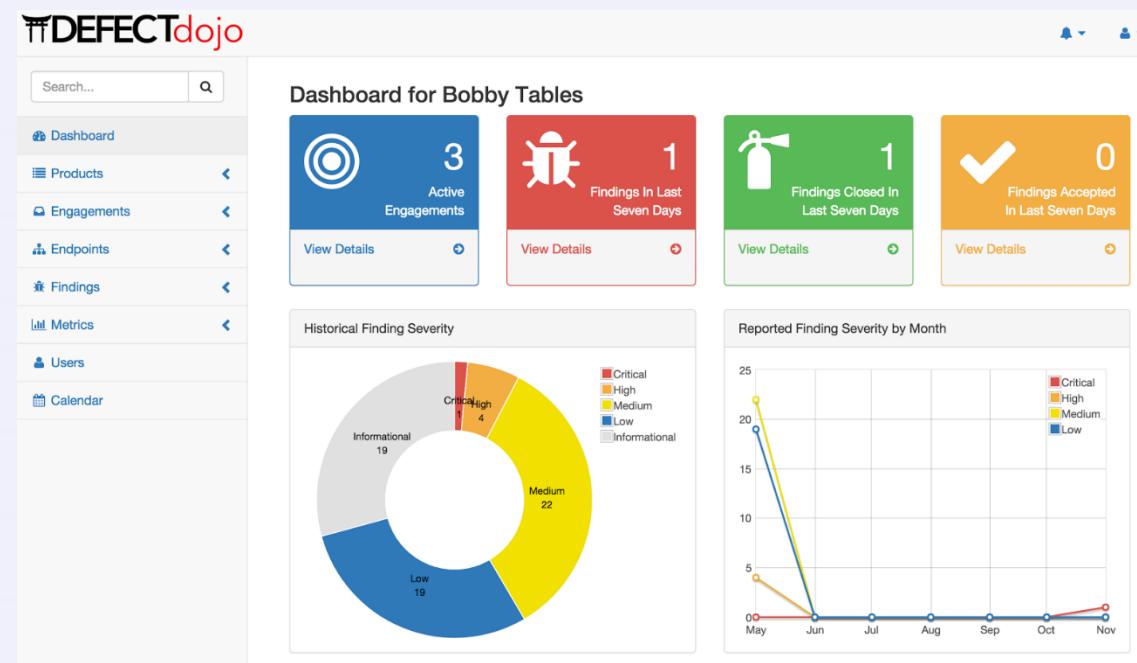




# OWASP

The Open Web Application Security Project

# DEFECTdojo



<https://owasp.org/www-project-defectdojo/>

<https://www.defectdojo.org/>



# OWASP

The Open Web Application Security Project



<https://owasp.org/www-project-webgoat/>



# OWASP

The Open Web Application Security Project



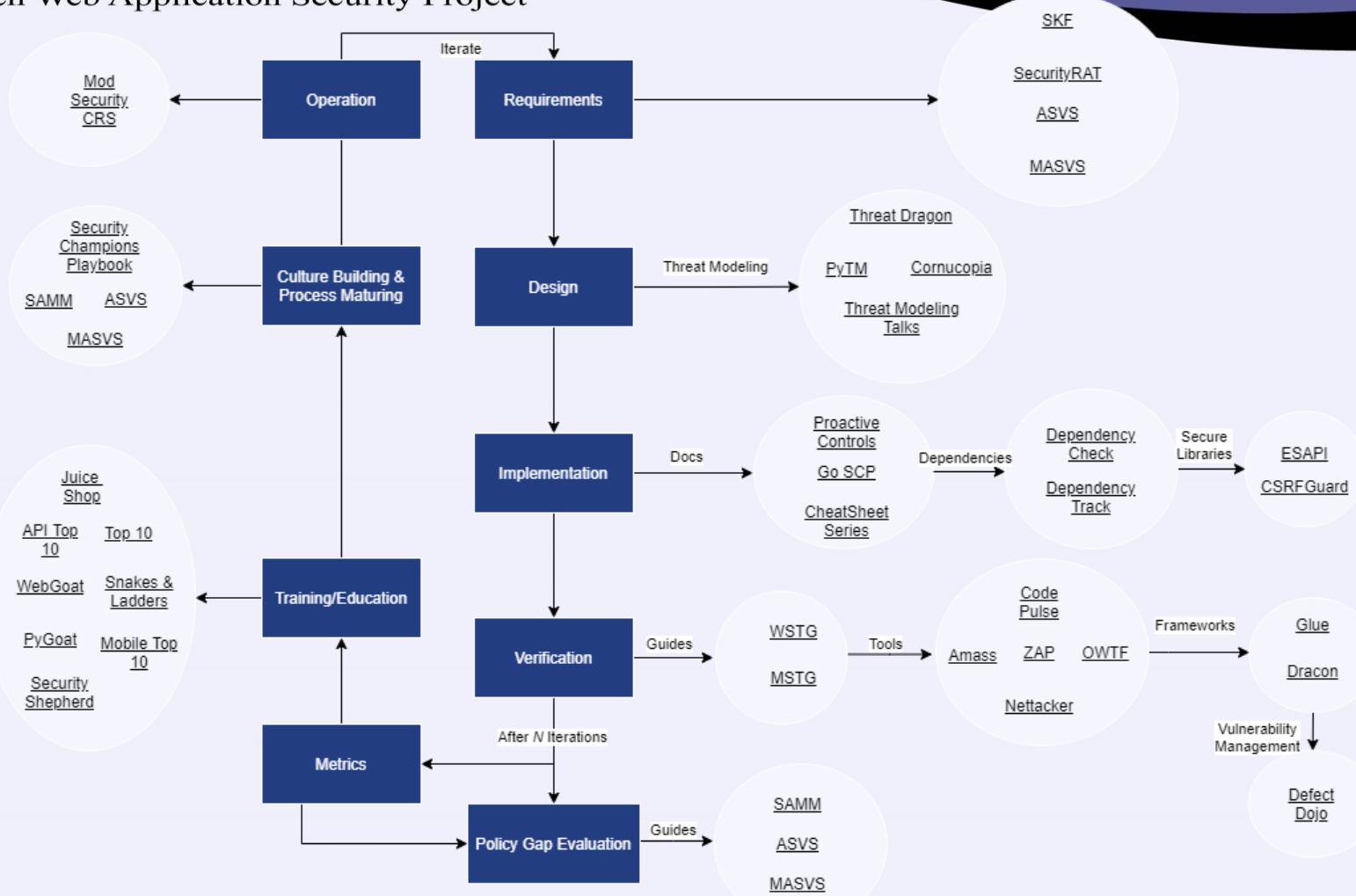
<https://owasp.org/www-project-juice-shop/>

# Application Security Wayfinder



# OWASP

# The Open Web Application Security Project



<https://owasp.org/www-project-integration-standards/>



# OWASP

The Open Web Application Security Project

