Owasp

Pierwszy kontakt

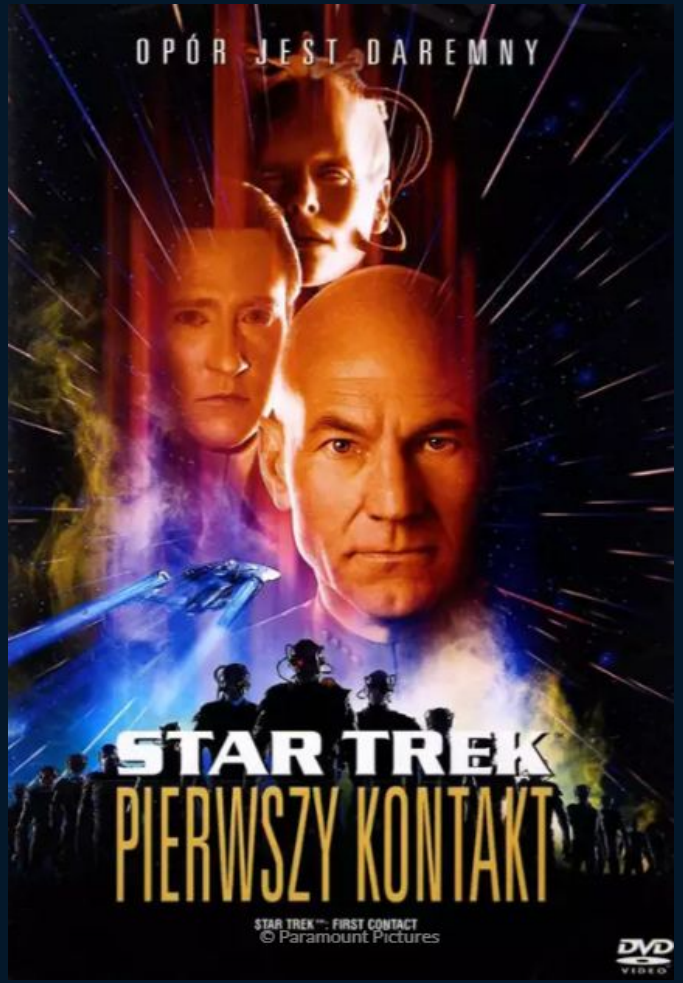
O mnie

- Z wykształcenia psycholog
- Pracuję jako programistka
- Rozwijam się w cybersecurity

- Fanka F1 i szydełkowania

https://www.linkedin.com/in/lena-sedkiewicz/
https://lenasedkiewicz.com/

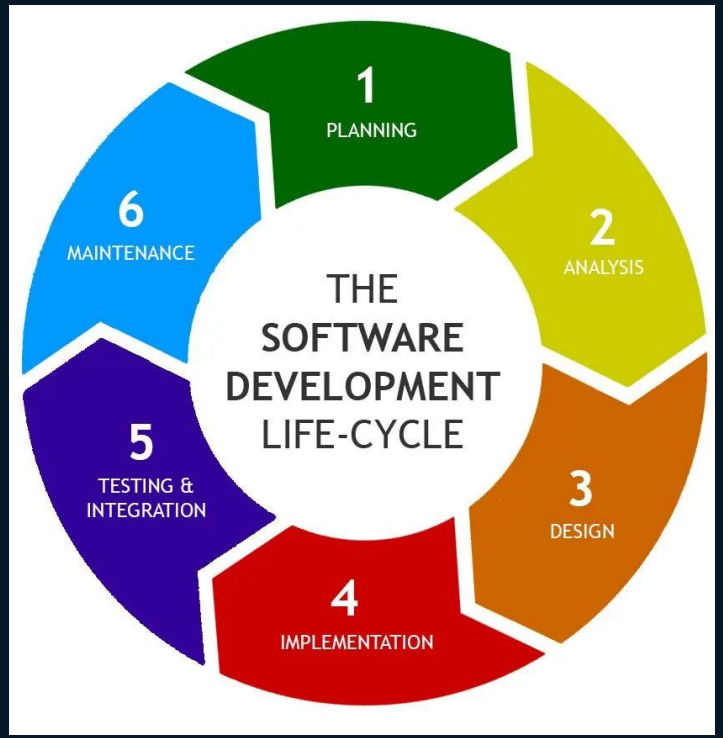
Pierwszy kontakt



Pierwszy kontakt

- Teraźniejszość - na ziemi dominuje cywilizacja Borg
- Powrót w przeszłość - by naprawić teraźniejszość
- [SPOILER ALERT]

SDLC - Cykl życia oprogramowania



https://medium.com/@artjoms/software-development-life-cycle-sdlc-6155dbfe3cbc

1. Planowanie

- Cel projektu
- Wymagania biznesowe
- Grupa docelowa

2. Analiza

- Analiza informacji z etapu planowania
- Wybór architektury / rozwiązań technicznych
- Specyfikacja techniczna

3. Projektowanie

- Opis modułów
- Przepływ danych
- Integracje od zewnętrznych dostawców

4. Implementacja

- Tworzenie produktu

5. Testowanie i integracja

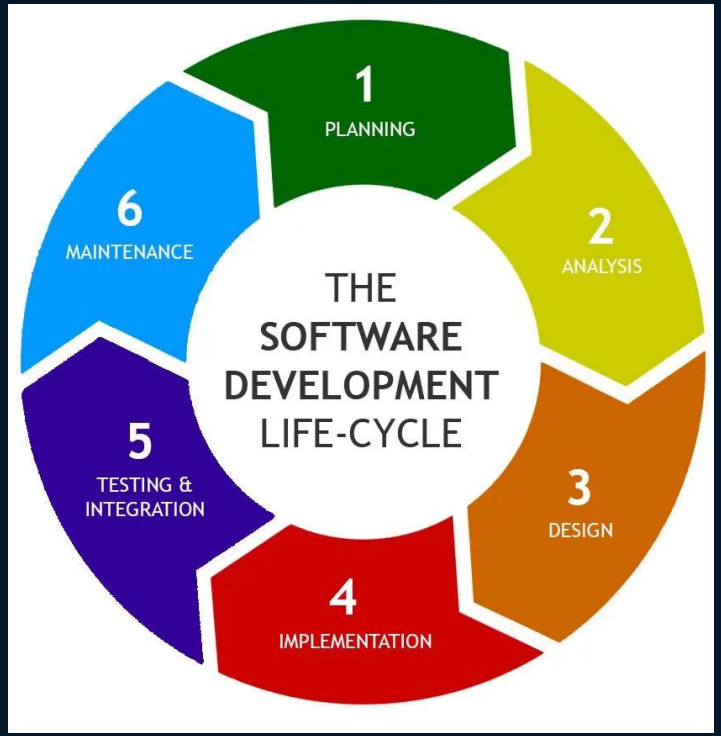
- Testy automatyczne i manualne

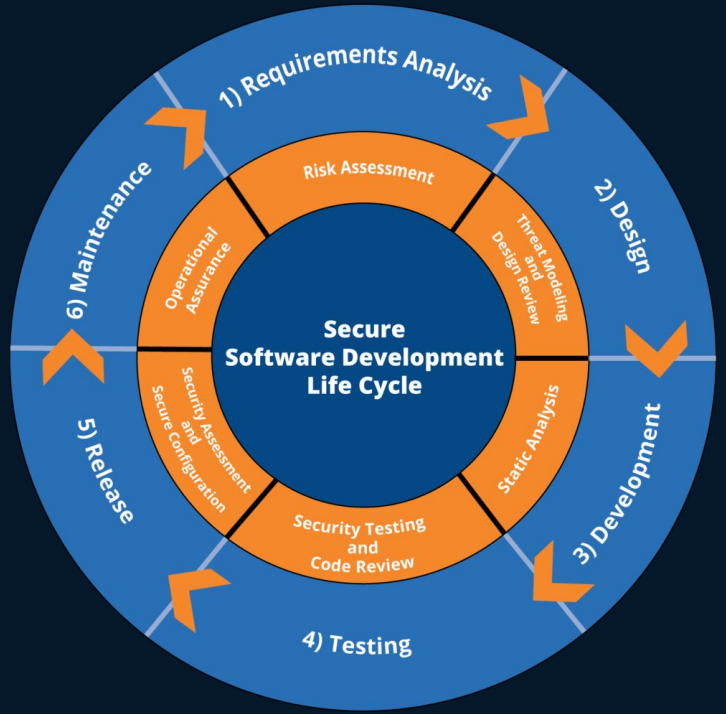
6. Utrzymanie

- Wersja alpha / beta aplikacji
- Wersja produkcyjna
- Utrzymanie

Star Trek VIII: Pierwszy kontakt (1996) © Paramount Pictures

SSDLC - Cykl życia bezpiecznego oprogramowania





https://www.digitalmaelstrom.net/it-security-services/secure-software-development-lifecycle-ssdlc/

1. Planowanie

- Identyfikacja ryzyka
- Ocena ryzyka
  - np. jakie dane będą przetwarzane i jak zapewnić bezpieczeństwo w ich przetwarzaniu
  - jakie zagrożenia niesie wybrany stack technologiczny / wybrane rozwiązania

2. Analiza

- Zdefiniowanie wymogów bezpieczeństwa dla rozwoju aplikacji

3. Projektowanie

- Modelowanie zagrożeń
  - identyfikacja zagrożeń i określenie sposobów ich zabezpieczenia (mitygacja)

4. Implementacja

- Tworzenie produktu zgodnie z wymogami z zakresu bezpieczeństwa
- Code review z perspektywy bezpieczeństwa
- SAST (analiza statyczna kodu)
  - Snyk
  - Semgrep
  - SonarQube

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

5. Testowanie i integracja

- Testy bezpieczeństwa
- Ocena bezpieczeństwa aplikacji i konfiguracji
- DAST?

6. Utrzymanie

- DAST
  - (OWASP) ZAP (ZED Attack Proxy)
  - Burp
  - Nuclei

https://owasp.org/www-project-devsecops-guideline/latest/02b-Dynamic-Application-Security-Testing

Star Trek VIII: Pierwszy kontakt (1996) © Paramount Pictures

OWASP

owasp.org/projects/

Please support the OWASP mission to improve software security through Open Source initiatives and community education. Donate Now!

OWASP

PROJECTS CHAPTERS EVENTS ABOUT

Store Donate Join

Projects

Projects for Good

We are a community of developers, technologists and evangelists improving the security of software. The OWASP Foundation gives aspiring open source projects a platform to improve the security of software with:

- Visibility: Our website gets more than six million visitors a year
- Credibility: OWASP is well known in the AppSec community
- Resources: Funding and Project Summits are available for qualifying Programs
- Community: Our Conferences and Local Chapters connect Projects with users

OWASP Projects are a collection of related tasks that have a defined roadmap and team members. Our projects are open source and are built by our community of volunteers - people just like you! OWASP project leaders are responsible for defining the vision, roadmap, and tasks for the project. The project leader also promotes the project and builds the team. OWASP currently has over 100 active projects, and new project applications are submitted every week.

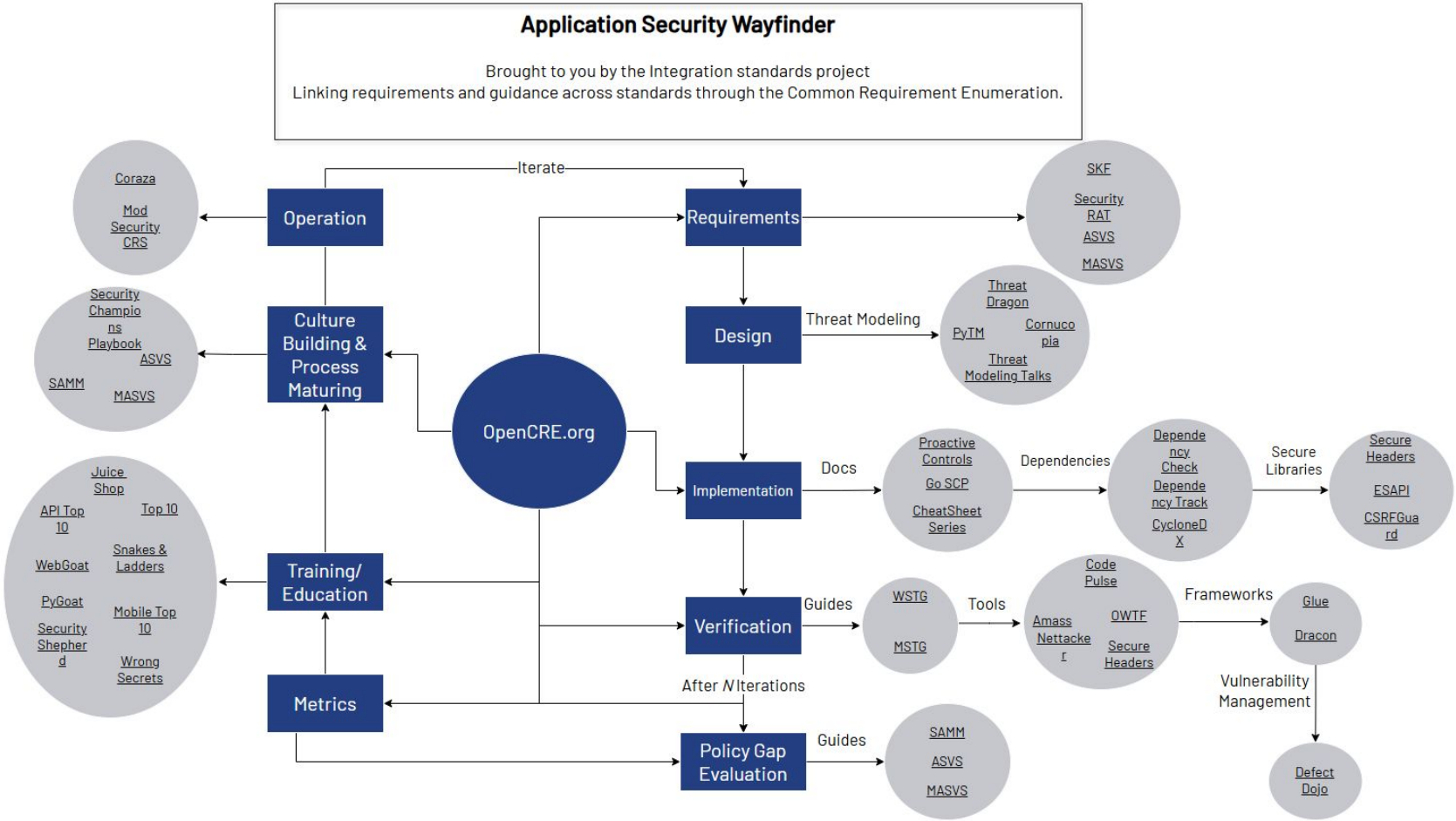
Open Worldwide Application Security Project

- Społeczność działająca na rzecz cyberbezpieczeństwa
- narzędzia i standardy bezpieczeństwa
- badania
- zasoby związane z cyberbezpieczeństwem różnych aspektów aplikacji (i nie tylko)

Przykładowe projekty

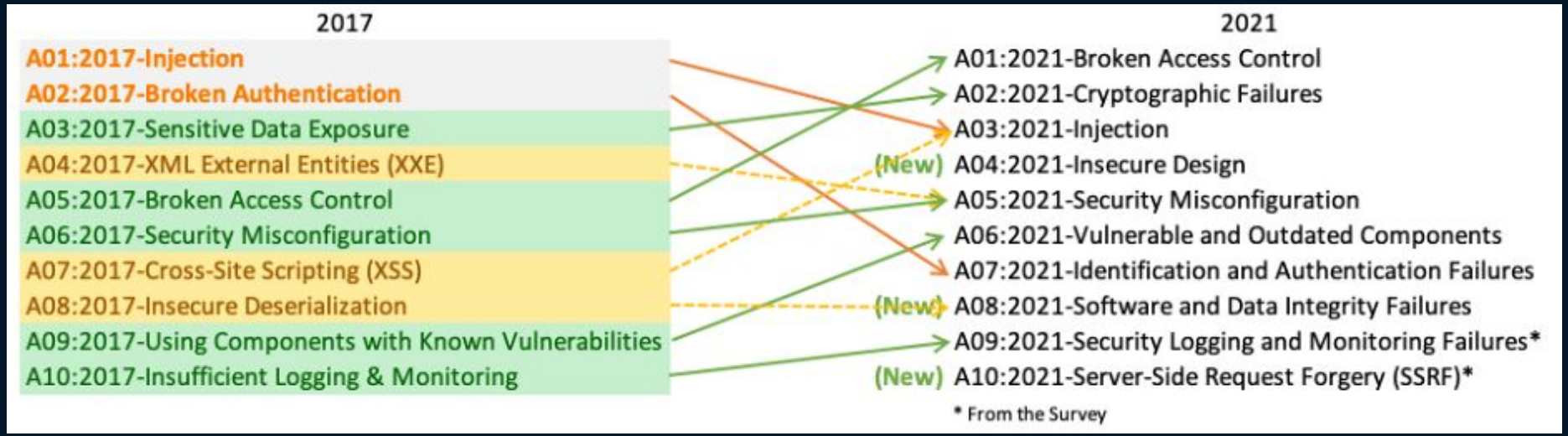
- https://owasp.org/projects/

- https://owasp.org/www-project-top-10-drone-security-risks/
- https://owasp.org/www-project-top-10-low-code-no-code-security-risks/
- https://owasp.org/www-project-security-culture/



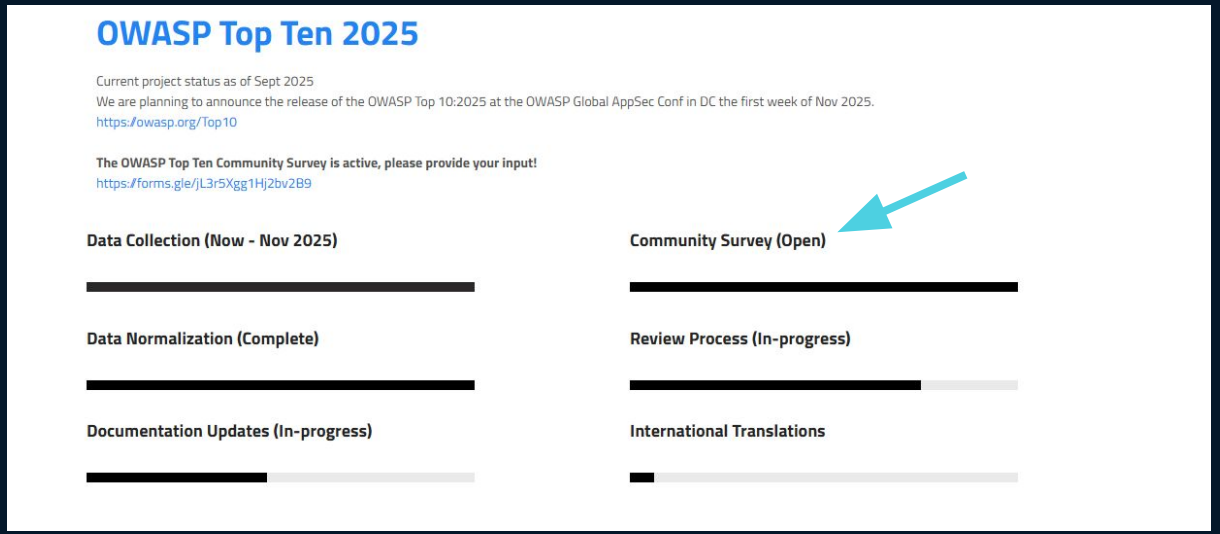
https://owasp.org/projects/

Edukacja / ćwiczenie umiejętności


2017
A01:2017-Injection
A02:2017-Broken Authentication
A03:2017-Sensitive Data Exposure
A04:2017-XML External Entities (XXE)
A05:2017-Broken Access Control
A06:2017-Security Misconfiguration
A07:2017-Cross-Site Scripting (XSS)
A08:2017-Insecure Deserialization
A09:2017-Using Components with Known Vulnerabilities
A10:2017-Insufficient Logging & Monitoring
2021
A01:2021-Broken Access Control
A02:2021-Cryptographic Failures
A03:2021-Injection
(New) A04:2021-Insecure Design
A05:2021-Security Misconfiguration
A06:2021-Vulnerable and Outdated Components
A07:2021-Identification and Authentication Failures
(New) A08:2021-Software and Data Integrity Failures
A09:2021-Security Logging and Monitoring Failures*
(New) A10:2021-Server-Side Request Forgery (SSRF)*
* From the Survey

Top Ten

- https://owasp.org/www-project-top-ten/
- https://www.owasptopten.org/
- Wersja OWASP Top Ten 2025 - pierwszy tydzień listopada 2025! 🥳



How It Works

1. Initial Planning/Data Call

Core team gets together and plans a rough schedule, a data call is released.

2. Industry Survey

We determine content in the survey and release for industry participation.

3. Data Analysis

After the data is collected, it is normalized and analyzed.

4. Draft Top Ten

Once we determine the eight risks from the data and the two from the survey, we draft a new list. The Draft is publicly released for review. All issues raised and decisions made are recorded in GitHub issues.

5. Release

Once we have reached a consensus and the core team agrees, we release the new OWASP Top Ten.

Top Ten

- od 2003
- Cel - budowanie świadomości
- wykorzystanie do programowania / testowania? “bare minimum and starting point”

https://owasp.org/Top10/A00_2021_How_to_use_the_OWASP_Top_10_as_a_standard/

Use Case	OWASP Top 10 2021	OWASP Application Security Verification Standard
Awareness	Yes	
Training	Entry level	Comprehensive
Design and architecture	Occasionally	Yes
Coding standard	Bare minimum	Yes
Secure Code review	Bare minimum	Yes
Peer review checklist	Bare minimum	Yes
Unit testing	Occasionally	Yes
Integration testing	Occasionally	Yes
Penetration testing	Bare minimum	Yes
Tool support	Bare minimum	Yes
Secure Supply Chain	Occasionally	Yes

Use Case	OWASP Top 10 2021	OWASP Application Security Verification Standard
Awareness	Yes	
Training	Entry level	Comprehensive
Design and architecture	Occasionally	Yes
Coding standard	Bare minimum	Yes
Secure Code review	Bare minimum	Yes
Peer review checklist	Bare minimum	Yes
Unit testing	Occasionally	Yes
Integration testing	Occasionally	Yes
Penetration testing	Bare minimum	Yes
Tool support	Bare minimum	Yes
Secure Supply Chain	Occasionally	Yes

Use Case	OWASP Top 10 2021	OWASP Application Security Verification Standard
Awareness	Yes	
Training	Entry level	Comprehensive
Design and architecture	Occasionally	Yes
Coding standard	Bare minimum	Yes
Secure Code review	Bare minimum	Yes
Peer review checklist	Bare minimum	Yes
Unit testing	Occasionally	Yes
Integration testing	Occasionally	Yes
Penetration testing	Bare minimum	Yes
Tool support	Bare minimum	Yes
Secure Supply Chain	Occasionally	Yes

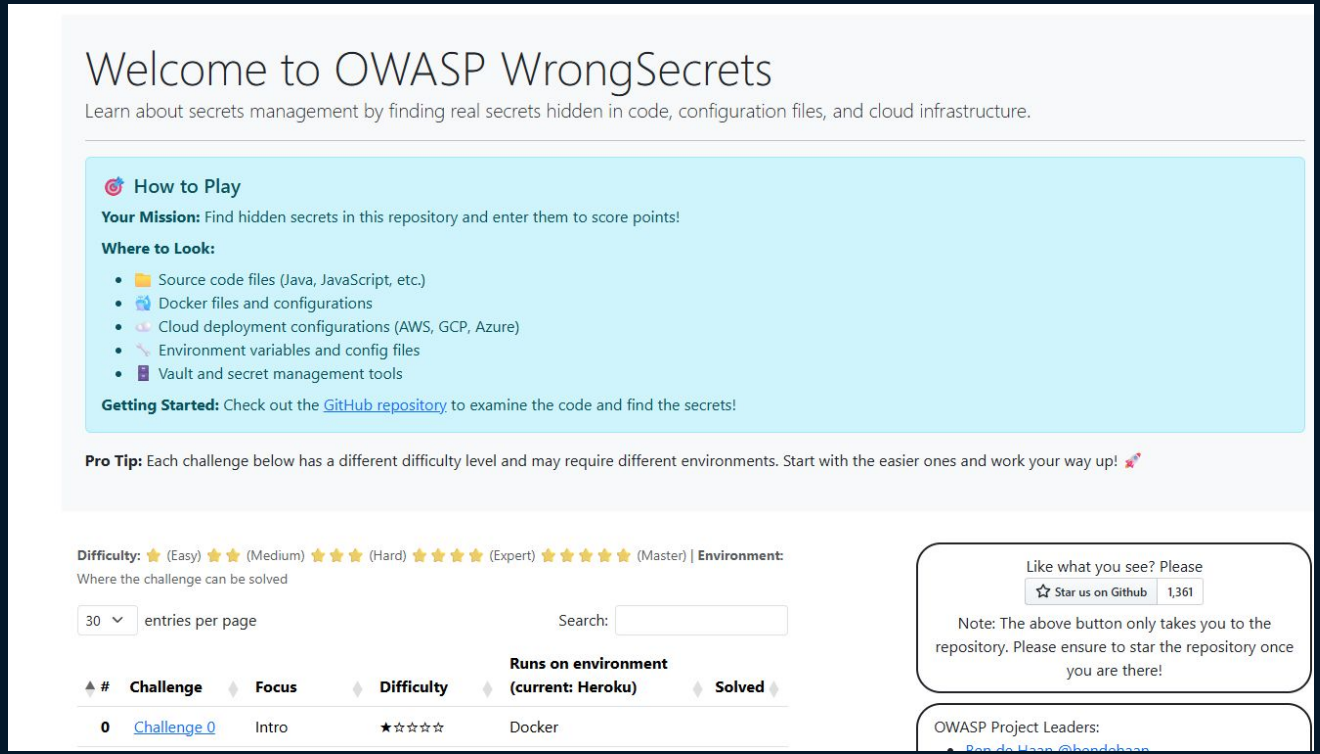
Nie tylko ten “tradycyjny” Top Ten...

- Bezpieczeństwo API: https://owasp.org/www-project-api-security/ (2023)
- Urządzenia mobilne: https://owasp.org/www-project-mobile-top-10/ (2024)
- Aplikacje desktopowe: https://owasp.org/www-project-desktop-app-security-top-10/
- Bezpieczeństwo danych: https://owasp.org/www-project-data-security-top-10/ (2023)
- Work in progress: DevSecOps: https://owasp.org/www-project-devsecops-top-10/ , ale np. https://owasp.org/www-project-devsecops-guideline/

Nie tylko ten “tradycyjny” Top Ten...

- Zdaje się, że zawieszony - Docker: https://owasp.org/www-project-docker-top-10/ -> https://github.com/OWASP/Docker-Security
- Oprogramowanie Open Source: https://owasp.org/www-project-open-source-software-top-10/
- Machine Learning: https://owasp.org/www-project-machine-learning-security-top-10/ (2023), ale np. https://owasp.org/www-project-ai-testing-guide/

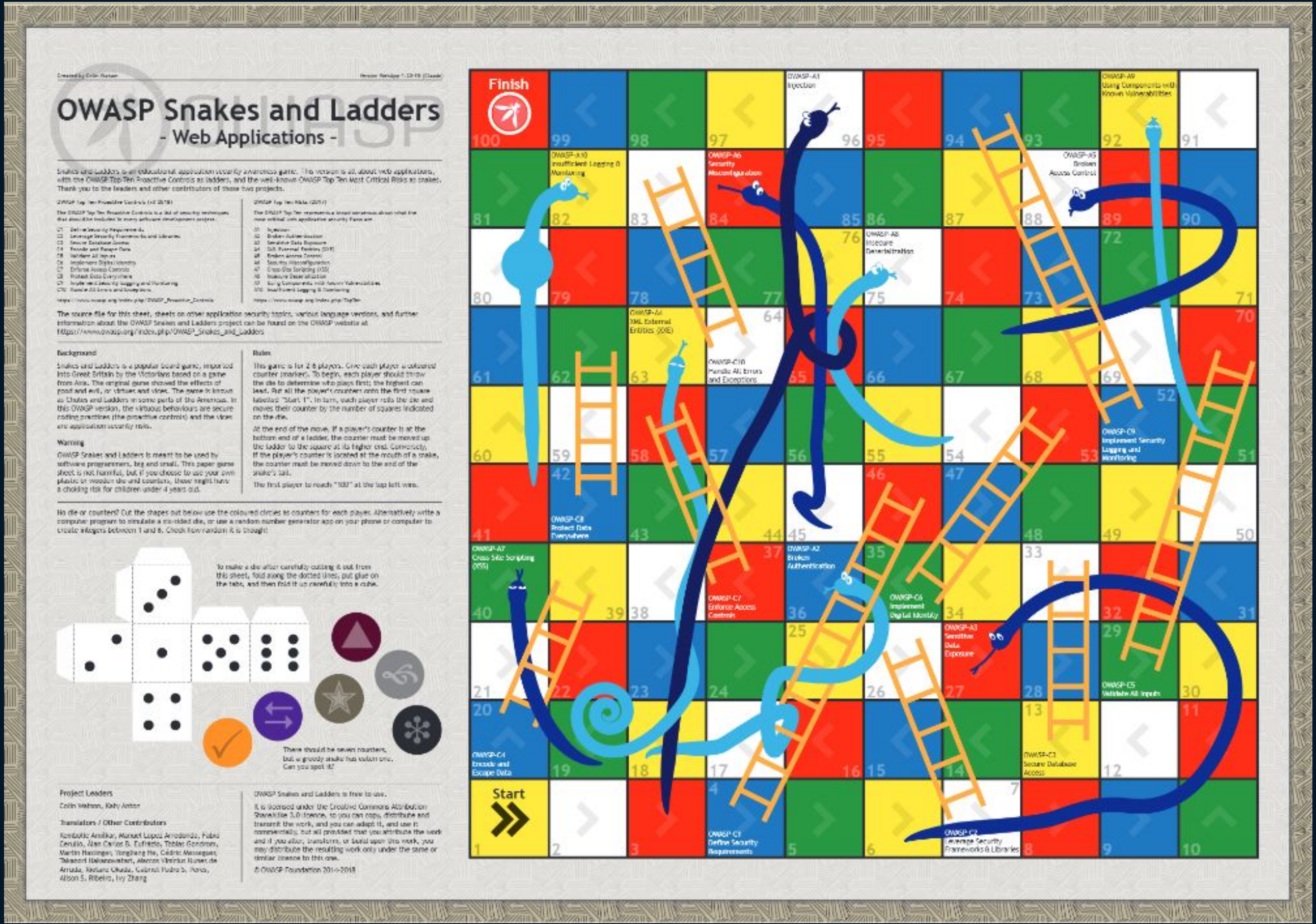
WrongSecrets



https://github.com/OWASP/wrongsecrets?tab=readme-ov-file || https://wrongsecrets.herokuapp.com/

Snakes and Ladders

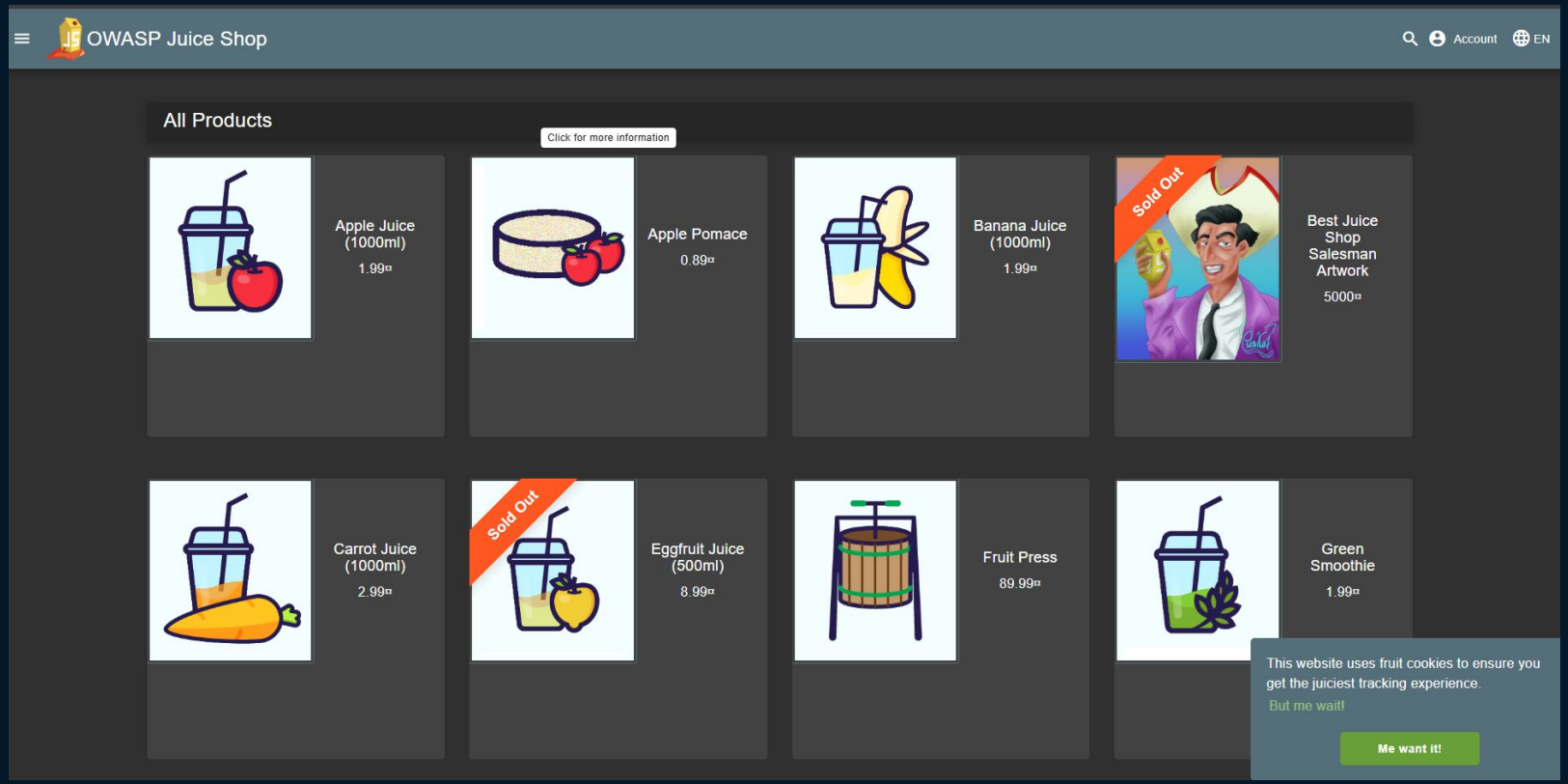
- węże - zagrożenia (2017)
- drabiny - techniki bezpieczeństwa wspierające rozwój bezpiecznego oprogramowania (2018)
- https://github.com/OWASP/www-project-snakes-and-ladders/blob/master/assets/files/web/EN/OWASP-SnakesAndLadders-WebApplications-EN-1v20.pdf


OWASP Snakes and Ladders
- Web Applications -
Snakes and Ladders is an educational application security awareness game. This version is all about web applications, with the OWASP Top Ten Proactive Controls as ladders, and the well-known OWASP Top Ten Most Critical Risks as snakes. Thank you to the leaders and other contributors of those two projects.
The source file for this sheet, sheets on other application security topics, various language versions, and further information about the OWASP Snakes and Ladders project can be found on the OWASP website at https://www.owasp.org/index.php/OWASP_Snakes_and_Ladders
Background
Snakes and Ladders is a popular board game, imported into Great Britain by the Victorians based on a game from Asia. The original game showed the effects of good and evil, or virtues and vices. The game is known as Chutes and Ladders in some parts of the Americas. In this OWASP version, the virtuous behaviours are secure coding practices (the proactive controls) and the vices are application security risks.
Warning
OWASP Snakes and Ladders is meant to be used by software programmers, big and small. This paper game sheet is not harmful, but if you choose to use your own plastic or wooden die and counters, these might have a choking risk for children under 4 years old.
Rules
This game is for 2-6 players. Give each player a coloured counter (marker). To begin, each player should throw the die to determine who plays first; the highest can lead. Put all the player's counters onto the first square labelled "Start 1". In turn, each player rolls the die and moves their counter by the number of squares indicated on the die.
At the end of the move, if a player's counter is at the bottom end of a ladder, the counter must be moved up the ladder to the square at its higher end. Conversely, if the player's counter is located at the mouth of a snake, the counter must be moved down to the end of the snake's tail.
The first player to reach "100" at the top left wins.
No die or counters? Cut the shapes out below use the coloured circles as counters for each player. Alternatively write a computer program to simulate a six-sided die, or use a random number generator app on your phone or computer to create integers between 1 and 6. Check how random it is though!
To make a die after carefully cutting it out from this sheet, fold along the dotted lines, put glue on the tabs, and then fold it up carefully into a cube.
There should be seven counters, but a greedy snake has eaten one. Can you spot it?
Project Leaders
Colin Watson, Katy Anton
Translators / Other Contributors
OWASP Snakes and Ladders is free to use.
It is licensed under the Creative Commons Attribution ShareAlike 3.0 licence, so you can copy, distribute and transmit the work, and you can adapt it, and use it commercially, but all provided that you attribute the work and if you alter, transform, or build upon this work, you may distribute the resulting work only under the same or similar licence to this one.
© OWASP Foundation 2014-2018
Finish
100
Start
1
OWASP-A1 Injection
OWASP-A2 Broken Authentication
OWASP-A3 Sensitive Data Exposure
OWASP-A4 XML External Entities (XXE)
OWASP-A5 Broken Access Control
OWASP-A6 Security Misconfiguration
OWASP-A7 Cross Site Scripting (XSS)
OWASP-A8 Insecure Deserialization
OWASP-A9 Using Components with Known Vulnerabilities
OWASP-A10 Insufficient Logging & Monitoring
OWASP-C1 Define Security Requirements
OWASP-C2 Leverage Security Frameworks & Libraries
OWASP-C3 Secure Database Access
OWASP-C4 Encode and Escape Data
OWASP-C5 Validate All Inputs
OWASP-C6 Implement Digital Identity
OWASP-C7 Enforce Access Controls
OWASP-C8 Protect Data Everywhere
OWASP-C9 Implement Security Logging and Monitoring
OWASP-C10 Handle All Errors and Exceptions

Juice Shop

- Aplikacja dziurawa jak szwajcarski ser
- Podstawy podstaw
- https://owasp.org/www-project-juice-shop/
- https://demo.owasp-juice.shop/#/

Juice Shop


OWASP Juice Shop
Account
EN
All Products
Click for more information
Apple Juice (1000ml)
1.99¤
Apple Pomace
0.89¤
Banana Juice (1000ml)
1.99¤
Sold Out
Best Juice Shop Salesman Artwork
5000¤
Carrot Juice (1000ml)
2.99¤
Sold Out
Eggfruit Juice (500ml)
8.99¤
Fruit Press
89.99¤
Green Smoothie
1.99¤
This website uses fruit cookies to ensure you get the juiciest tracking experience.
But me wait!
Me want it!

Analiza wymagań || Ocena rozbieżności

ASVS

- Application Security Verification Standard
- testowanie webaplikacji
- wsparcie w bezpiecznym rozwijaniu aplikacji dla developerów
- trzy poziomy
  - L1 - krytyczne
  - L2 - dobrze zaimplementować
  - L3 - tylko dla orłów
- https://owasp.org/www-project-application-security-verification-standard/

ASVS 4.03 vs. 5.0.0

V1.11 Business Logic Architecture

#	Description	L1	L2	L3	CWE
1.11.1	Verify the definition and documentation of all application components in terms of the business or security functions they provide.		✓	✓	1059
1.11.2	Verify that all high-value business logic flows, including authentication, session management and access control, do not share unsynchronized state.		✓	✓	362
1.11.3	Verify that all high-value business logic flows, including authentication, session management and access control are thread safe and resistant to time-of-check and time-of-use race conditions.			✓	367

V1.12 Secure File Upload Architecture

#	Description	L1	L2	L3	CWE
1.12.1	[DELETED, DUPLICATE OF 12.4.1]				
1.12.2	Verify that user-uploaded files - if required to be displayed or downloaded from the application - are served by either octet stream downloads, or from an unrelated domain, such as a cloud file storage bucket. Implement a suitable Content Security Policy (CSP) to reduce the risk from XSS vectors or other attacks from the uploaded file.		✓	✓	646

V2.3 Business Logic Security

This section considers key requirements to ensure that the application enforces business logic processes in the correct way and is not vulnerable to attacks that exploit the logic and flow of the application.

#	Description	Level
2.3.1	Verify that the application will only process business logic flows for the same user in the expected sequential step order and without skipping steps.	1
2.3.2	Verify that business logic limits are implemented per the application's documentation to avoid business logic flaws being exploited.	2
2.3.3	Verify that transactions are being used at the business logic level such that either a business logic operation succeeds in its entirety or it is rolled back to the previous correct state.	2

#	Description	Level
2.3.4	Verify that business logic level locking mechanisms are used to ensure that limited quantity resources (such as theater seats or delivery slots) cannot be double-booked by manipulating the application's logic.	2
2.3.5	Verify that high-value business logic flows require multi-user approval to prevent unauthorized or accidental actions. This could include but is not limited to large monetary transfers, contract approvals, access to classified information, or safety overrides in manufacturing.	3

OWASP Top Ten vs OWASP ASVS

2021

A01:2021-Broken Access Control
A02:2021-Cryptographic Failures
A03:2021-Injection
A04:2021-Insecure Design
A05:2021-Security Misconfiguration
A06:2021-Vulnerable and Outdated Components
A07:2021-Identification and Authentication Failures
A08:2021-Software and Data Integrity Failures
A09:2021-Security Logging and Monitoring Failures*
A10:2021-Server-Side Request Forgery (SSRF)*

* From the Survey

- V1 Encoding and Sanitization
- V2 Validation and Business Logic
- V3 Web Frontend Security
- V4 API and Web Service
- V5 File Handling
- V6 Authentication
- V7 Session Management
- V8 Authorization
- V9 Self-contained Tokens
- V10 OAuth and OIDC
- V11 Cryptography
- V12 Secure Communication
- V13 Configuration
- V14 Data Protection
- V15 Secure Coding and Architecture
- V16 Security Logging and Error Handling
- V17 WebRTC

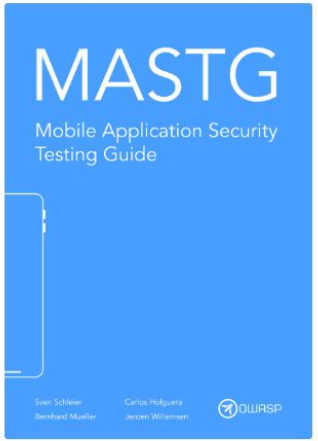
MASVS

- https://mas.owasp.org/MASVS/ - ASVS dla urządzeń mobilnych
  - poziomy weryfikacji: L1, L2, Resilience (reverse engineering, manipulowanie danymi)
- https://mas.owasp.org/#our-mission

OWASP MASVS

OWASP MASWE

OWASP MASTG



OWASP MAS Checklist

Wdrażanie bezpieczeństwa

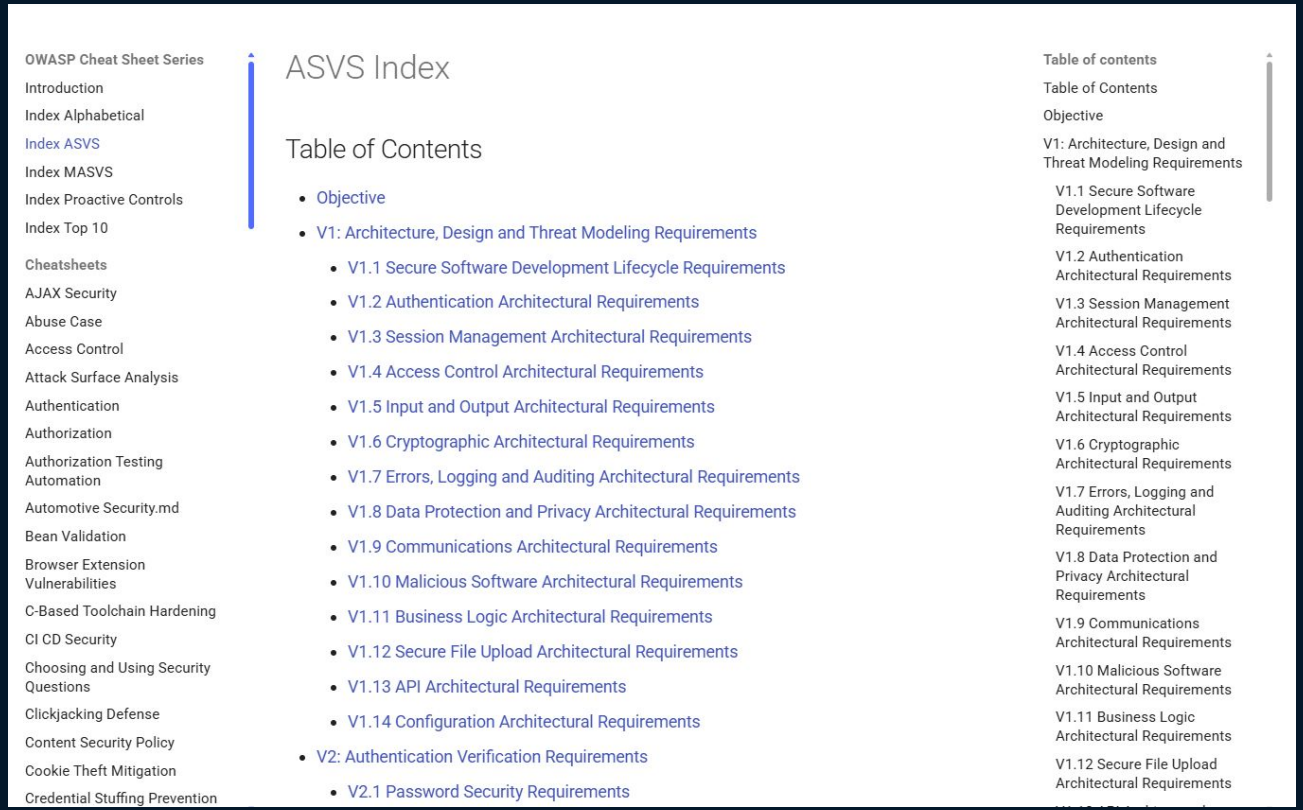
Life is too short • AppSec is tough • Cheat!

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.

Cheat Sheet Series

- Pigułki wiedzy dla programistów
  - https://cheatsheetseries.owasp.org/
  - https://owasp.org/www-project-cheat-sheets/
- Mapowanie wg projektów
  - https://cheatsheetseries.owasp.org/IndexTopTen.html
  - https://cheatsheetseries.owasp.org/IndexASVS.html
  - https://cheatsheetseries.owasp.org/IndexMASVS.html

ASVS 4.0.x



OWASP Cheat Sheet Series

Introduction
Index Alphabetical
Index ASVS
Index MASVS
Index Proactive Controls
Index Top 10

Cheatsheets

AJAX Security
Abuse Case
Access Control
Attack Surface Analysis
Authentication
Authorization
Authorization Testing Automation
Automotive Security.md
Bean Validation
Browser Extension Vulnerabilities
C-Based Toolchain Hardening
CI CD Security
Choosing and Using Security Questions
Clickjacking Defense
Content Security Policy
Cookie Theft Mitigation
Credential Stuffing Prevention

ASVS Index

Table of Contents

Table of contents

Weryfikacja (testy)

WSTG - Web Security Testing Guide

- testowanie aplikacji i usług webowych
- https://owasp.org/www-project-web-security-testing-guide/

Stable

View the always-current stable version at stable.

[Unreleased 4.3]

[Version 4.2] - 2020-12-03

Version 4.2 introduces new testing scenarios, updates existing chapters, and offers an improved writing style and chapter layout.

wstg / document /

Add file

mbiesiad Update 01-Testing_for_Weak_Transport_Layer_Security.md (#1242) 37d6202 · 10 hours ago History

Name	Last commit message	Last commit date
..		
0-Foreword	OWASP W: web > worldwide (#1219)	4 months ago
1-Frontispiece	Mrojz credit (#1224)	3 months ago
2-Introduction	Move to HTTPS links (#1174)	10 months ago
3-The_OWASP_Testing_Framework	Remove any www-pdf-archive links which no longer work (#1206)	7 months ago
4-Web_Application_Security_Testing	Update 01-Testing_for_Weak_Transport_Layer_Security.md (#1242)	10 hours ago
5-Reporting	Move to HTTPS links (#1174)	10 months ago
6-Appendix	OWASP W: web > worldwide (#1219)	4 months ago
README.md	Create 02-API_Broken_Object_Level_Authorization.md (#1190)	7 months ago

README.md

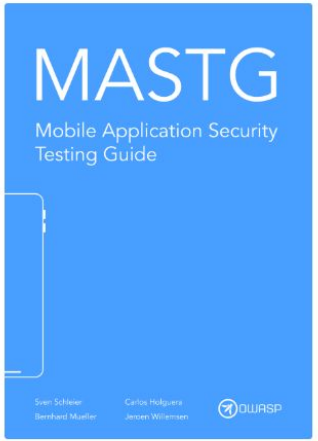
WSTG - Web Security Testing Guide

Web Application Security Testing

MASTG

- https://owasp.org/www-project-mobile-app-security/

OWASP MASTG

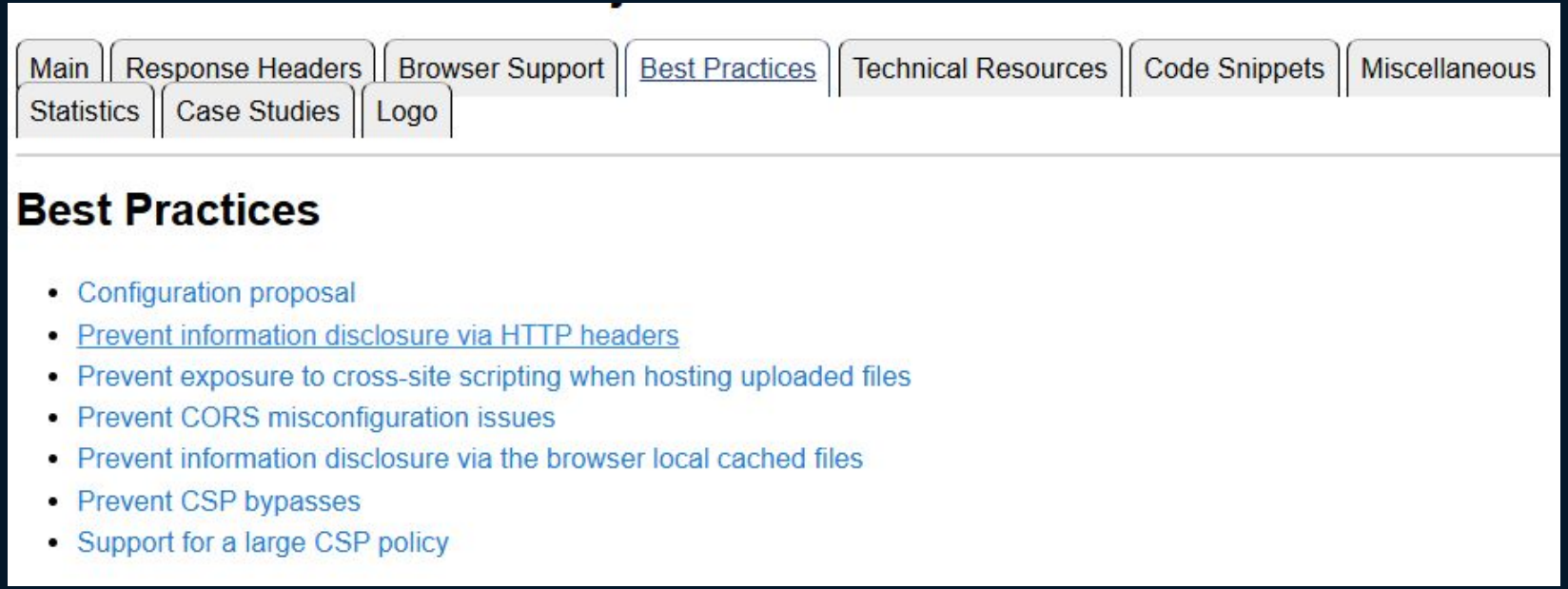


OWASP MAS Checklist

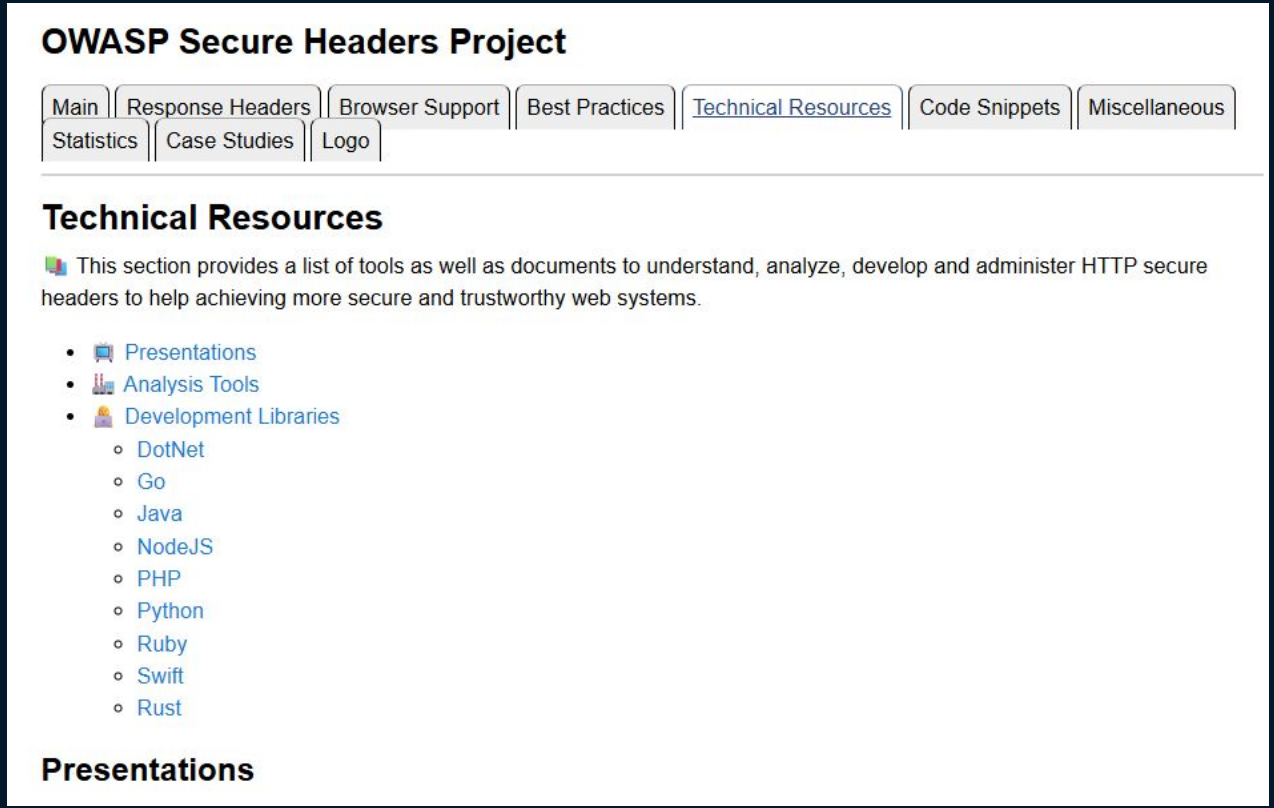
OWASP Secure Headers Project

https://owasp.org/www-project-secure-headers/

https://github.com/OWASP/www-project-secure-headers



OWASP Secure Headers Project



OWASP Secure Headers Project

Main | Response Headers | Browser Support | Best Practices | Technical Resources | Code Snippets | Miscellaneous | Statistics | Case Studies | Logo

Technical Resources

This section provides a list of tools as well as documents to understand, analyze, develop and administer HTTP secure headers to help achieving more secure and trustworthy web systems.

- Presentations
- Analysis Tools
- Development Libraries
  - DotNet
  - Go
  - Java
  - NodeJS
  - PHP
  - Python
  - Ruby
  - Swift
  - Rust

Presentations

OWASP Secure Headers Project

NodeJS

Library	Description	Ref
helmet	Module to help secure Express apps with various HTTP headers.	
ember-cli-content-security-policy	This addon makes it easy to use Content Security Policy (CSP) in your project. It can be deployed either via a Content-Security-Policy header sent from the Ember CLI Express server, or as a meta tag in the index.html file.	

Python

Library	Description	Ref
django-csp and django-security	Content Security Policy for Django. A collection of models, views, middlewares, and forms to help secure a Django project.	/
Secweb	Secweb is a pack of security middlewares for fastApi and starlette server it includes CSP, HSTS, and many more.	
secure	Lightweight library to add security headers to Django, Flask, FastAPI, and more.	

https://owasp.org/www-project-security-culture/stable/2-Why_Add_Security_In_Development_Teams/


BINGE
SOCIETY
© 1996 Paramount Pictures