OWASP

OWASP

- **[Open Web Application Security Project](#)**

  - Free and open software security community

  - 501(c)(3) Nonprofit organization

- Core purpose

  - Be the thriving global community that drives visibility and evolution in the safety and security of the world's software

Core Values

- **OPEN** Everything at OWASP is radically transparent from our finances to our code.

- **INNOVATION** OWASP encourages and supports innovation and experiments for solutions to software security challenges.

- **GLOBAL** Anyone around the world is encouraged to participate in the OWASP community.

- **INTEGRITY** OWASP is an honest and truthful, vendor neutral, global community.

Principles

- Free & Open

- Governed by rough consensus & running code

- Abide by a [code of ethics](#)

- Not-for-profit

- Not driven by commercial interests

- Risk based approach

OWASP Projects

OWASP Projects

An OWASP project is a collection of related tasks that have a defined roadmap and team members.

Project Type	Examples
Tool	Zed Attack Proxy, Dependency Check, DefectDojo, Juice Shop
Code	ModSecurity Core Rule Set, Java HTML Sanitizer, Security Logging Project, AppSensor
Documentation	OWASP Top 10, Application Security Verification Standard (ASVS), OWASP 24/7 Podcast, Cornucopia

OWASP Communities

Community	Advancing the state of security in the area of...
Builders	...application development
Breakers	...security testing
Defenders	...application defense, including the tools and techniques that enable the detection and response to application layer attacks

Project Lifecycle

Level	Icon	Description
Incubator		OWASP Incubator projects represent the experimental playground where projects are still being fleshed out, ideas are still being proven, and development is still underway.
Labs		OWASP Labs projects represent projects that have produced an OWASP reviewed deliverable of value.
Flagship		The OWASP Flagship designation is given to projects that have demonstrated strategic value to OWASP and application security as a whole.

Determining Lab Status

It is essential for an OWASP Labs project to have:

- A version number with a clear release schedule

- GitHub source control and a public issue tracking system

- Stable build and release

- Instructions on how to use and build the project properly

Determining Flagship Status

It is essential for an OWASP Flagship project to have:

- Considerable number of users and contributors

- Considerable number of commits and improvements in a time span of at least two years

- A unique approach or proposition in application security

- Exposure through security conferences

- Use and acceptance by the community

- Being used as reference in books and other resources

10

OWASP Chapters

OWASP Chapters

OWASP Chapters exist to raise awareness of the OWASP mission, making application security visible, at the local level.

OWASP German Chapter

There is one Chapter for Germany in total which is complemented by a so-called OWASP Stammtisch each in several metropolitan areas such as Hamburg, Munich, Frankfurt, Stuttgart or Karlsruhe.

OWASP
German Chapter

Mandatory Chapter Rules

- Organize free and open meetings

- Hold a minimum of 4 chapter meetings or events each year

- Give official meeting notice through the wiki, chapter mailing list, and OWASP Calendar

- Abide by OWASP principles and the code of ethics

- Protect the privacy of the chapter's local contacts

- Maintain vendor neutrality (act independently)

- Spend any chapter funds in accordance with the OWASP goals, code of ethics, and principles

OWASP Top 10

OWASP Top 10 `Builders` `Defenders`

1	Injection	6	Security Misconfiguration
2	Broken Authorization	7	Cross-Site-Scripting (XSS)
3	Sensitive Data Exposure	8	Insecure Deserialization
4	XML External Entities	9	Using Components with Known Vulnerabilities
5	Broken Access Control	10	Insufficient Logging & Monitoring

Application Security Risks
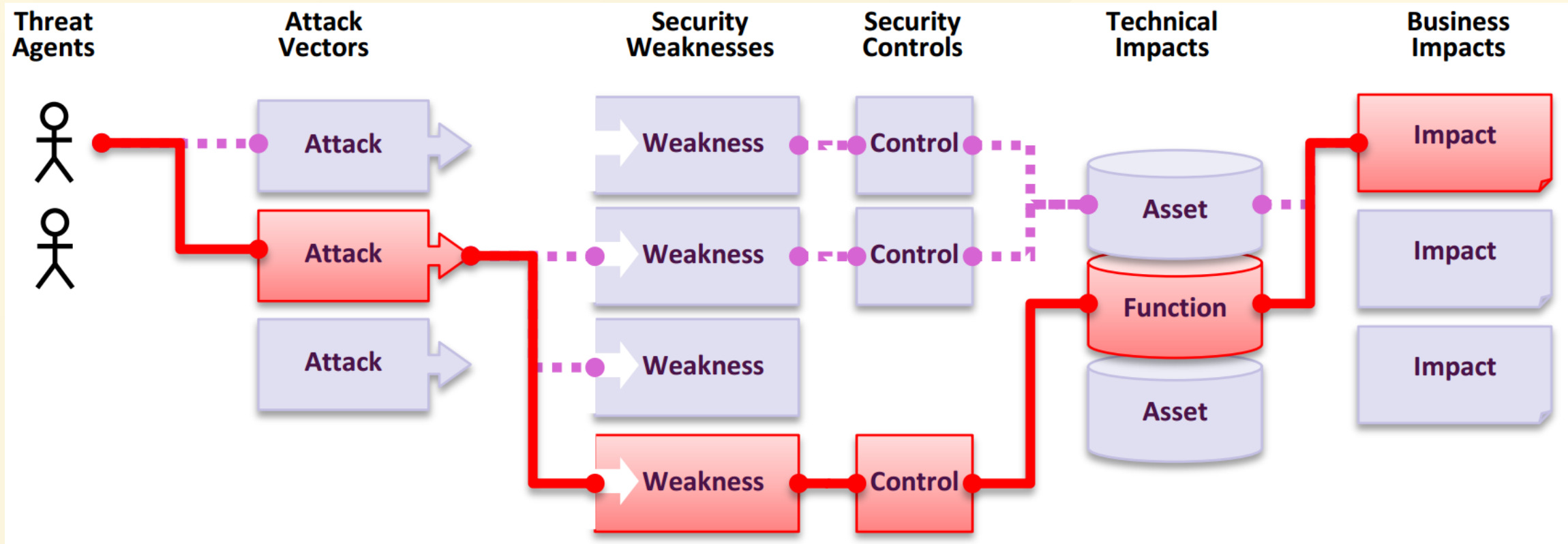
Risk Rating Table

Threat Agents	Exploitability	Weakness Prevalence	Weakness Detectability	Technical Impacts	Business Impacts
App Specific	EASY: 3	WIDESPREAD: 3	EASY: 3	SEVERE: 3	App / Business Specific
	AVERAGE: 2	COMMON: 2	AVERAGE: 2	MODERATE: 2	
	DIFFICULT: 1	UNCOMMON: 1	DIFFICULT: 1	MINOR: 1	

ℹ️ *Based on the OWASP Risk Rating Methodology*

Risk Calculation Example

Threat Agents / Attack Vectors	Security Weakness		Impacts	
App Specific	Exploitability: 3	Prevalence: 3	Detectability: 3	Technical: 2	Business ?
	3	3	3		
	Likelihood Rating: 3.0 (Average of Exploitability, Prevalence and Detectability)			* 2	
	Risk Ranking: 6.0 (Likelihood * Impact)				

Top 10 Risk Factor Summary

Risk	Threat Agents	Attack Vectors (Exploitability)	Security Weakness (Prevalence)	Security Weakness (Detectability)	Impacts (Technical)	Impacts (Business)	Score
A1:2017-Injection	App Specific	EASY: 3	COMMON: 2	EASY: 3	SEVERE: 3	App Specific	8.0
A2:2017-Broken Authentication	App Specific	EASY: 3	COMMON: 2	AVERAGE: 2	SEVERE: 3	App Specific	7.0
A3:2017-Sensitive Data Exposure	App Specific	AVERAGE: 2	WIDESPREAD: 3	AVERAGE: 2	SEVERE: 3	App Specific	7.0
A4:2017-XML External Entities (XXE)	App Specific	AVERAGE: 2	COMMON: 2	EASY: 3	SEVERE: 3	App Specific	7.0
A5:2017-Broken Access Control	App Specific	AVERAGE: 2	COMMON: 2	AVERAGE: 2	SEVERE: 3	App Specific	6.0
A6:2017-Security Misconfiguration	App Specific	EASY: 3	WIDESPREAD: 3	EASY: 3	MODERATE: 2	App Specific	6.0
A7:2017-Cross-Site Scripting (XSS)	App Specific	EASY: 3	WIDESPREAD: 3	EASY: 3	MODERATE: 2	App Specific	6.0
A8:2017-Insecure Deserialization	App Specific	DIFFICULT: 1	COMMON: 2	AVERAGE: 2	SEVERE: 3	App Specific	5.0
A9:2017-Vulnerable Components	App Specific	AVERAGE: 2	WIDESPREAD: 3	AVERAGE: 2	MODERATE: 2	App Specific	4.7
A10:2017-Insufficient Logging&Monitoring	App Specific	AVERAGE: 2	WIDESPREAD: 3	DIFFICULT: 1	MODERATE: 2	App Specific	4.0

Some(!) Additional Risks to Consider

Cross-Site Request Forgery (CSRF)	Unvalidated Forward and Redirects
Uncontrolled Resource Consumption ('Resource Exhaustion', 'AppDoS')	Improper Control of Interaction Frequency (Anti-Automation)
Unrestricted Upload of File with Dangerous Type	Inclusion of Functionality from Untrusted Control Sphere (3rd Party Content)
User Interface (UI) Misrepresentation of Critical Information (Clickjacking etc.)	Server-Side Request Forgery (SSRF)

Other Resources on AppSec

- **SANS** Software Security Community

  - CWE/SANS TOP 25 Most Dangerous Software Errors

  - Securing Web Application Technologies [SWAT] Checklist

- **CWE** Common Weakness Enumeration

  - Community-developed list of common software security weaknesses

OWASP Juice Shop

OWASP Juice Shop Builders Breakers Defenders

OWASP Juice Shop is an intentionally insecure webapp for security trainings written entirely in JavaScript which encompasses the entire OWASP Top Ten and other severe security flaws.

Installation

- Individual local instance per student

- Runs on node.js, Docker or Vagrant

⚠ Hacking Rules

- Do **not** look at the source code on GitHub

- Do **not** look at GitHub issues, PRs etc.

- Do **not** read online tutorials or walkthroughs

- Report problems during exercises immediately

Exercise 1.1

Install the OWASP Juice Shop `v7.x`

1. Open https://github.com/bkimminich/juice-shop#setup

2. Follow the instructions for one method out of

   - From Sources

   - Packaged Distributions (💙 on university PCs)

   - Docker Container

   - Vagrant

Exercise 1.2

Happy path shopping tour

1. Register a user account at your local Juice Shop

2. Browse the inventory and purchase some products

3. Try out all other functionality you find in the application

Exercise 1.3

Score Board

1. Find the hidden Score Board in the Juice Shop (⭐)

Exercise 1.4 *(optional)*

Transfer your hacking progress

1. Open your browser's developer tools (`F12` in Chrome/Firefox)

2. Find the cookie `continueCode` and copy its value to your other computer

3. Install OWASP Juice Shop on your other computer and launch it

4. `F12` into the developer tools and create the cookie `continueCode` with the value from your first computer

5. Restart the Juice Shop server