Secure Software Design

Mario Alviano

OWASP

OWASP Top 10 - 2017

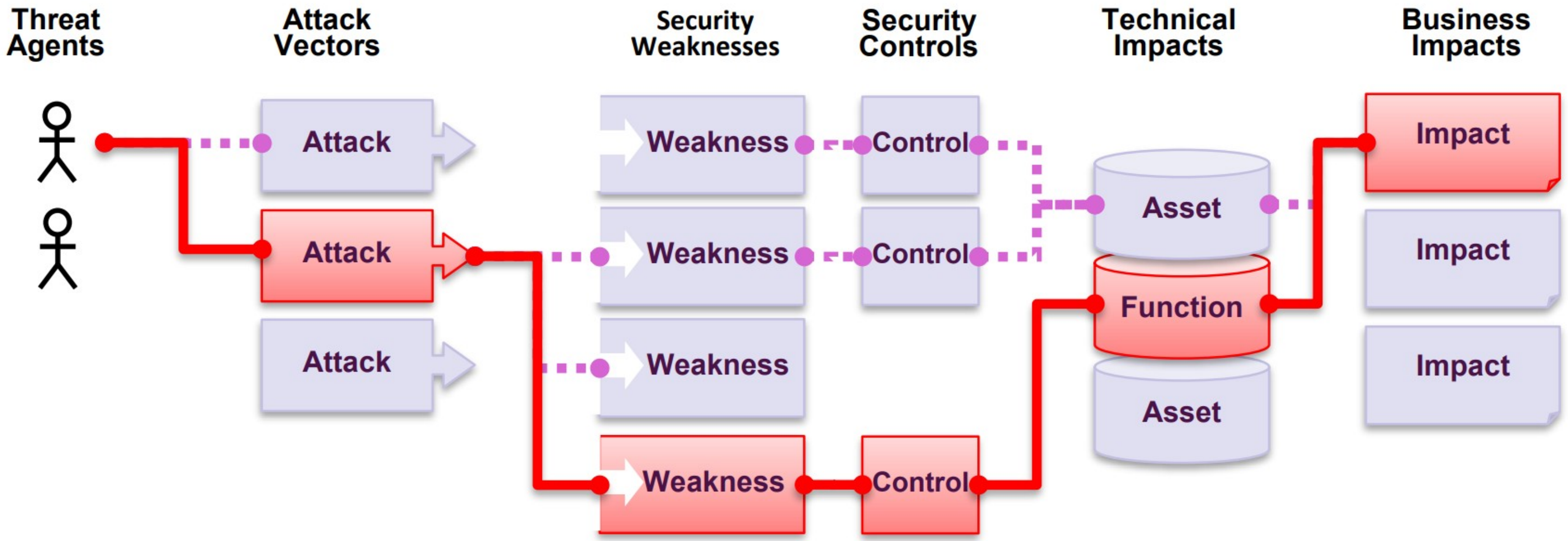
The Ten Most Critical Web Application Security Risks

https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

From the Forward

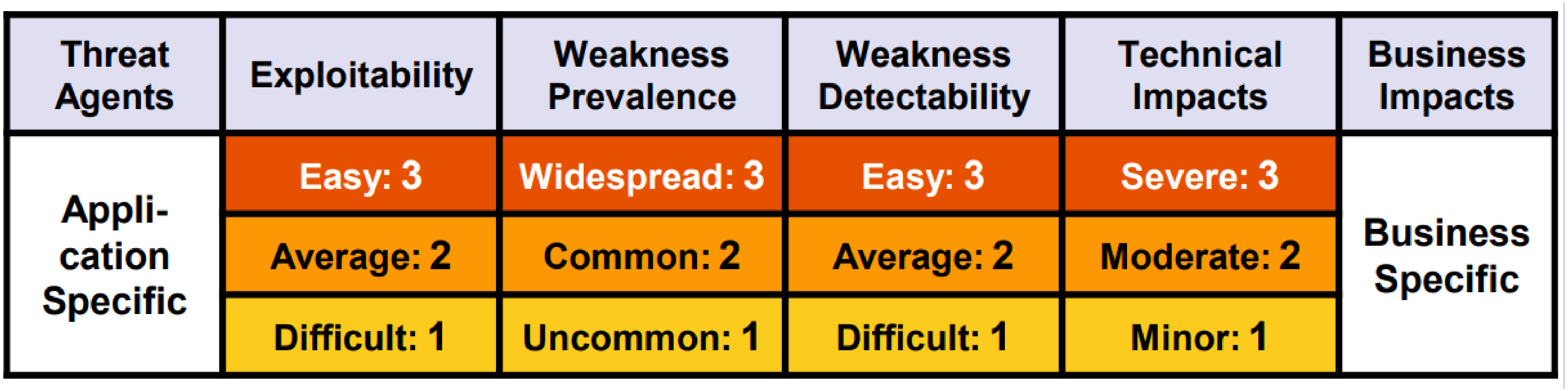
- Insecure software is undermining our financial, healthcare, defense, energy, and other critical infrastructure
- As our software becomes increasingly complex, and connected, the difficulty of achieving application security increases exponentially
- We can no longer afford to tolerate relatively simple security problems like those presented in this OWASP Top 10
- Although the original goal of the OWASP Top 10 project was simply to raise awareness amongst developers and managers, it has become **the** de facto application security standard

Application Security Risks



Every unexpected path represents a risk

OWASP Risk Rating



Threat Agents	Exploitability	Weakness Prevalence	Weakness Detectability	Technical Impacts	Business Impacts
Appli-cation Specific	Easy: 3	Widespread: 3	Easy: 3	Severe: 3	Business Specific
	Average: 2	Common: 2	Average: 2	Moderate: 2	
	Difficult: 1	Uncommon: 1	Difficult: 1	Minor: 1	

A1 – Injection

Injection flaws, such as SQL, NoSQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker’s hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.

- Let’s check it on the OWASP Top 10
- Also have a look at the description and examples in
  - CWE-77: Command Injection
  - CWE-89: SQL Injection
  - https://www.owasp.org/index.php/SQL_Injection
  - https://www.owasp.org/index.php/Blind_SQL_Injection

A2 – Broken Authentication

Application functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users’ identities temporarily or permanently.

- Let’s check it on the OWASP Top 10
- Also have a look at the description and examples in
  - CWE-287: Improper Authentication
  - CWE-384: Session Fixation

A3 – Sensitive Data Exposure

Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.

- Let’s check it on the OWASP Top 10
- Also have a look at the description and examples in
  - CWE-312: Cleartext Storage of Sensitive Information
  - CWE-319: Cleartext Transmission of Sensitive Information

A4 – External Entities (XXE)

Many older or poorly configured XML processors evaluate external entity references within XML documents. External entities can be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution, and denial of service attacks.

- Let’s check it on the OWASP Top 10
- Also have a look at the description and examples in
  - https://www.owasp.org/index.php/XML_External_Entity_(XXE)_Processing
  - https://en.wikipedia.org/wiki/Billion_laughs_attack

A5 – Broken Access Control

Restrictions on what authenticated users are allowed to do are often not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.

- Let's check it on the OWASP Top 10
- Also have a look at the description and examples in
  - https://en.wikipedia.org/wiki/Cross-origin_resource_sharing
  - CWE-22: Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')

A6 – Security Misconfiguration

Security misconfiguration is the most commonly seen issue. This is commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information. Not only must all operating systems, frameworks, libraries, and applications be securely configured, but they must be patched and upgraded in a timely fashion.

- Let’s check it on the OWASP Top 10

A7 – Cross-Site Scripting (XSS)

XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with user-supplied data using a browser API that can create HTML or JavaScript. XSS allows attackers to execute scripts in the victim’s browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.

- Let’s check it on the OWASP Top 10
- Also have a look at the description and examples in
  - CWE-79: Improper neutralization of user supplied input

A8 – Insecure Deserialization

Insecure deserialization often leads to remote code execution. Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks, including replay attacks, injection attacks, and privilege escalation attacks.

A9 – Using Components with Known Vulnerabilities

Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts.

A10 – Insufficient Logging & Monitoring

Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, maintain persistence, pivot to more systems, and tamper, extract, or destroy data. Most breach studies show time to detect a breach is over 200 days, typically detected by external parties rather than internal processes or monitoring.