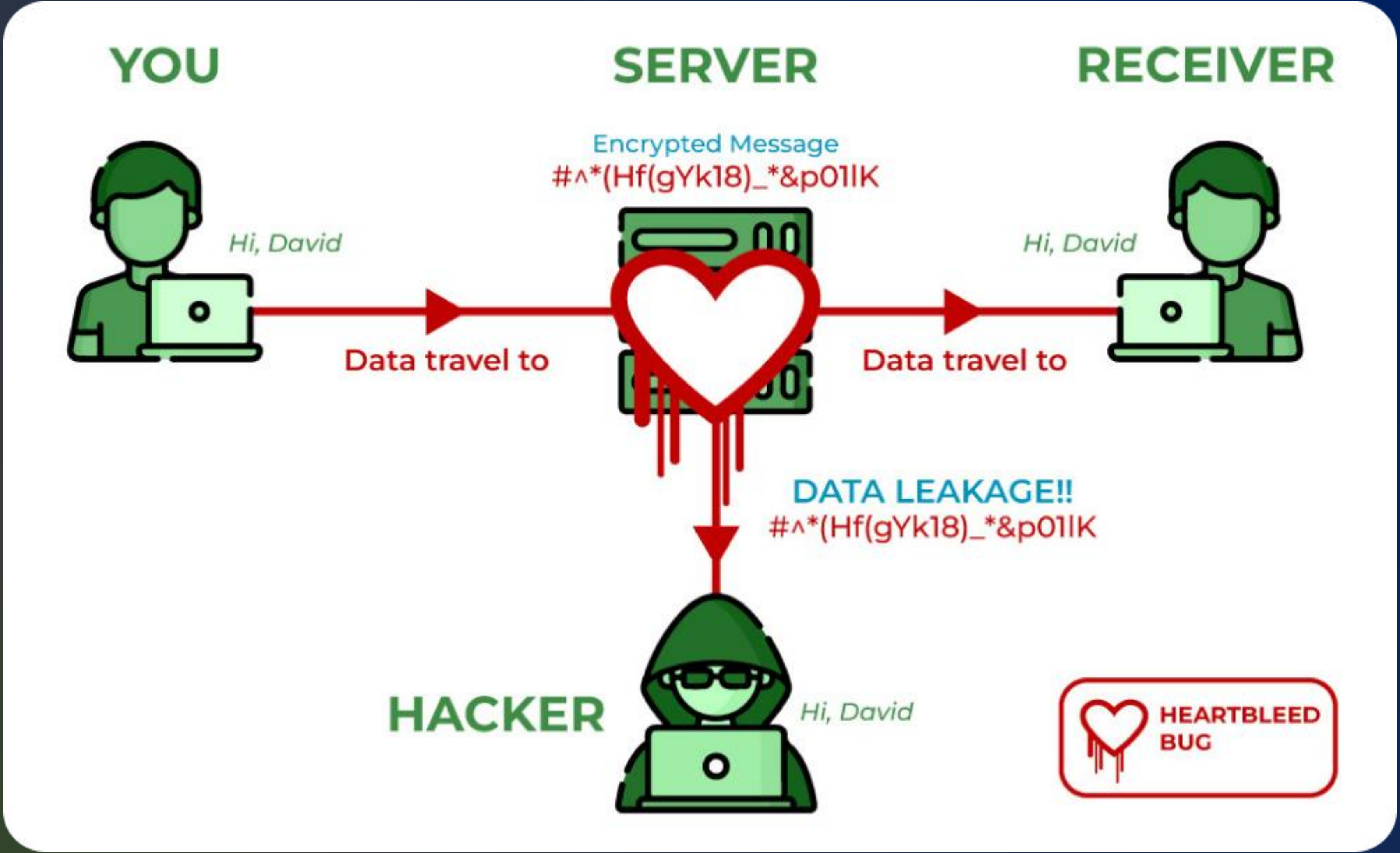
OWASP TOP 10 IN 10 MINUTES

THE WEB'S MOST CRITICAL RISKS

LEARNING OBJECTIVES

- ✅ Understand what OWASP Top 10 is
- ✅ Explore key vulnerabilities through a real case
- ✅ Learn practical mitigations
- ✅ End with a quick quiz

HEARTBLEED (OPENSSL 2014)



HEARTBLEED (OPENSSL 2014)

🕒 Discovered: April 8, 2014, by researchers at Google & Codenomicon.

🧩 Component: OpenSSL — the encryption library securing ~66% of the Internet.

🔐 Purpose: Handles SSL/TLS encryption — the “secret language” of HTTPS.

💔 The Bug: Allowed attackers to read server memory → leak private keys, passwords, sensitive data.

🧠 Hidden Danger: Undetectable — no logs, no signs of attack.

⏳ Exposure Window: Active since Dec 2011 → over 2 years of silent risk.

🌍 Impact: Millions of websites, apps, banks, and services vulnerable.

⚠️ Lesson: Even trusted open-source components can introduce massive risk if not regularly audited or updated.

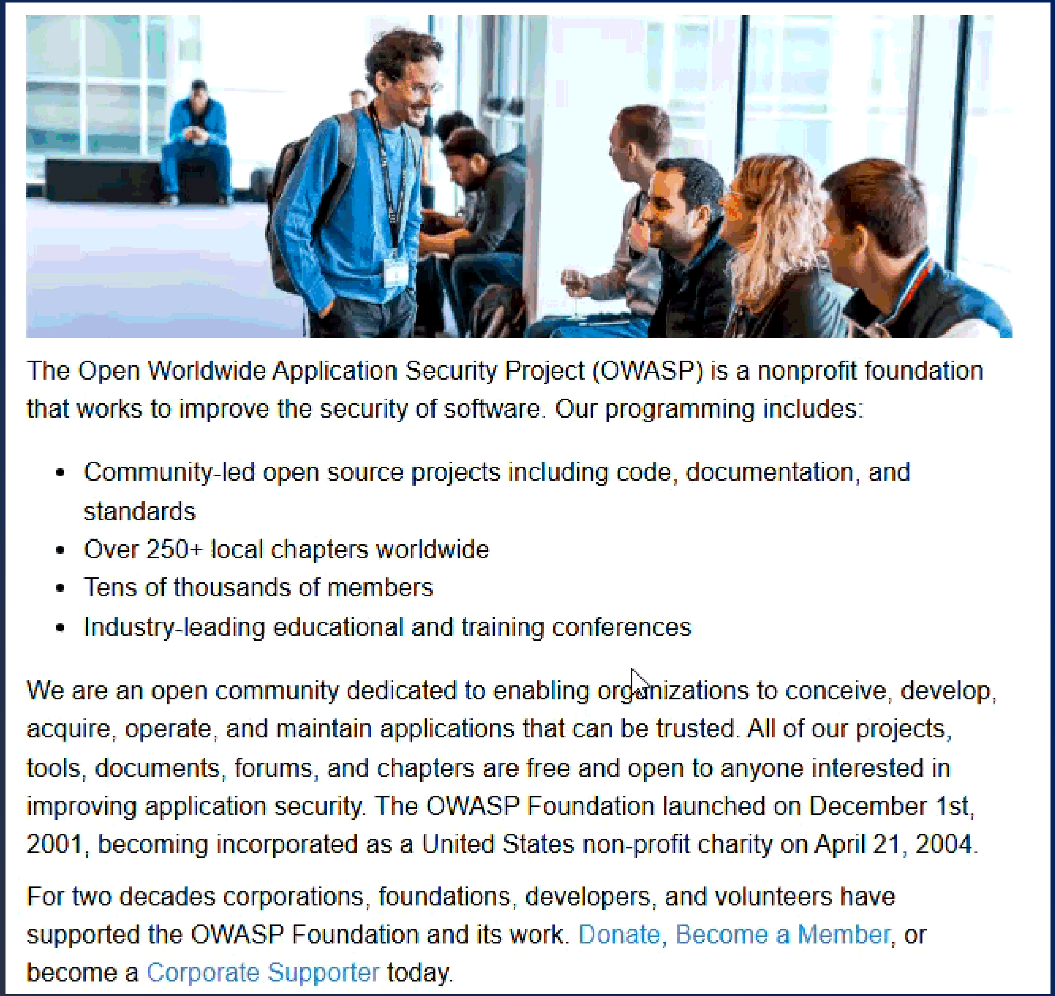
WHAT IS OWASP?

OWASP stands for the Open Web Application Security Project.

🌐 Founded: 2003 — non-profit organization improving web application security.

OWASP is best known for its Top 10, which is a regularly updated list of the most critical security risks to web applications. This list is widely regarded as the standard starting point for web application security—used by developers, security professionals, and organizations around the world.

- [TryHackMe – OWASP Top 10 – 2021]
- [https://owasp.org/about/#]



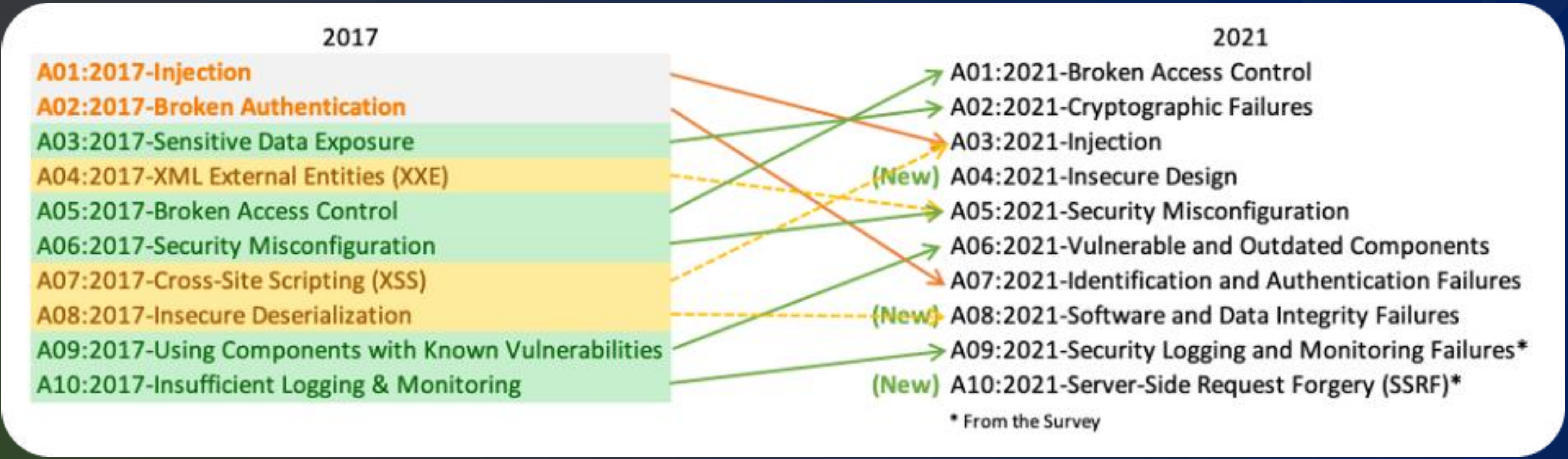
The Open Worldwide Application Security Project (OWASP) is a nonprofit foundation that works to improve the security of software. Our programming includes:

- Community-led open source projects including code, documentation, and standards
- Over 250+ local chapters worldwide
- Tens of thousands of members
- Industry-leading educational and training conferences

We are an open community dedicated to enabling organizations to conceive, develop, acquire, operate, and maintain applications that can be trusted. All of our projects, tools, documents, forums, and chapters are free and open to anyone interested in improving application security. The OWASP Foundation launched on December 1st, 2001, becoming incorporated as a United States non-profit charity on April 21, 2004.

For two decades corporations, foundations, developers, and volunteers have supported the OWASP Foundation and its work. Donate, Become a Member, or become a Corporate Supporter today.

OVERVIEW OF THE TOP 10



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

2021 - #1

BROKEN ACCESS CONTROL

UP

BROKEN ACCESS CONTROL

2017 - #5

“Occurs when users can act outside their intended permissions—accessing data or performing actions they shouldn’t be allowed to.”

2021 - #2

CRYPTOGRAPHIC ERRORS

UP

SENSITIVE DATA EXPOSURE

2017 - #3

"Happens when sensitive data is exposed due to weak or missing encryption, insecure key management, or outdated cryptographic algorithms."

2021 - #3

INJECTION

DOWN

CROSS-SITE SCRIPTING

2017 - #1

"When untrusted input is sent to an interpreter (like SQL, OS, or LDAP), causing unintended commands or queries to be executed."

2021 - #4
INSECURE DESIGN
NEW

“Refers to flaws in the system’s architecture or logic that make it vulnerable, even before coding begins—security wasn’t built into the design.”

2021 - #5

SECURITY MISCONFIGURATION

UP

SECURITY MISCONFIGURATION

2017 - #6

"When systems, frameworks, or applications are not securely configured—like leaving default settings, unused features, or open cloud storage."

2021 - #6

VULNERABLE & OUTDATED COMPONENTS

UP

USING COMPONETS WITH KNOWN VULNERABILITES

2017 - #9

“Using libraries, frameworks, or software that have known vulnerabilities or are no longer supported, exposing the application to known exploits.”

2021 - #7

IDENTIFICATION & AUTHENTICATION FAILURES

DOWN

BROKEN AUTHENTICATION

2017 - #2

“Weak or broken authentication mechanisms that let attackers compromise passwords, session tokens, or impersonate users.”

2021 - #8

SOFTWARE & DATA INTEGRITY FAILURES

NEW

“Occur when code or infrastructure doesn’t verify integrity—like using untrusted updates, insecure CI/CD pipelines, or unsigned code.”

2021 - #9

SECURITY LOGGING & MONITORING FAILURES

UP

INSUFFICIENT LOGGING & MONITORING

2017 - #10

“Inadequate logging or alerting prevents detection and response to breaches, allowing attackers to stay undetected.”

2021 - #10

SERVER-SIDE REQUEST FORGERY

NEW

“When a server fetches a remote resource based on unvalidated user input, allowing attackers to make requests to internal or external systems.”

Category	Description / Risk	Examples	Prevention / Mitigation
Broken Access Control	Users can act outside their intended permissions.	Accessing another user's data, modifying roles via URL tampering.	Enforce least privilege, use server-side access checks, deny by default.
Cryptographic Failures	Sensitive data exposed due to weak or missing encryption.	Using HTTP instead of HTTPS, weak hashing (MD5/SHA1), hardcoded keys.	Use TLS 1.2+, modern algorithms (AES, SHA-256), proper key management.
Injection	Untrusted input alters commands or queries.	SQL Injection, OS Command Injection, LDAP Injection.	Use prepared statements, parameterized queries, and input validation.
Insecure Design	Security not considered during system design.	Lack of threat modeling, no validation workflow, insecure architecture.	Apply secure design patterns, threat modeling, and defense-in-depth.
Security Misconfiguration	Improperly set security headers, permissions, or defaults.	Default credentials, directory listing, missing headers.	Harden configurations, disable defaults, automate secure setup.
Vulnerable and Outdated Components	Using components with known vulnerabilities.	Unpatched libraries, outdated frameworks (like old OpenSSL → Heartbleed).	Regularly update dependencies, use SCA tools, monitor CVEs.
Identification and Authentication Failures	Flaws in login, session, or identity handling.	Weak passwords, session fixation, credential stuffing.	Implement MFA, secure session management, strong password policies.
Software and Data Integrity Failures	Code or data integrity not verified.	Untrusted software updates, dependency confusion, CI/CD tampering.	Verify digital signatures, use signed packages, implement integrity checks.
Security Logging and Monitoring Failures	Insufficient detection or response to attacks.	Missing audit trails, no alerting, silent failures.	Enable centralized logging, alert on anomalies, retain logs securely.
Server-Side Request Forgery (SSRF)	Server makes requests to unintended locations.	Fetching URLs from user input → internal network exposure.	Validate URLs, use allow-lists, restrict network access for outbound calls.

REFLECTION QUESTION

Spot the Vulnerability:

Which OWASP Top 10 category does the Heartbleed vulnerability best represent, and why?

Cryptographic Failures.

Reasoning: Heartbleed was a vulnerability in the OpenSSL library, which handles encryption for SSL/TLS connections. The bug didn’t break the encryption algorithm itself — AES, RSA, etc. were still mathematically sound. Instead, it bypassed encryption entirely by allowing attackers to read the server’s private memory, which could include encryption keys, usernames, passwords, and sensitive data in plaintext. That means it caused exposure of sensitive data due to poor implementation of cryptography, not because the cryptography was weak — a textbook case of Cryptographic Failures under OWASP Top 10.

RECAP

- ✅ OWASP definition: a regularly updated list of the most critical security risks to web applications
- ✅ Explored key vulnerabilities through a real case
- ✅ Answered the Heartbleed question

ADDITIONAL RESOURCES

- https://tryhackme.com/room/owasptop102021
- https://owasp.org/www-project-top-ten/
- https://www.geeksforgeeks.org/ethical-hacking/owasp-top-10-vulnerabilities-and-preventions/

THE END