Webbiztonság alapjai

XSS, CSRF, CORS, Cookie security, OWASP Top 10

Webbiztonság alapjai

Miért fontos a webbiztonság?

- A web univerzális felhasználói felület lett
- Kritikus rendszerek webes vezérlése
- A támadások többsége nem „hackelés”
- Biztonság = tervezési kérdés is

A webes támadások alapgondolata

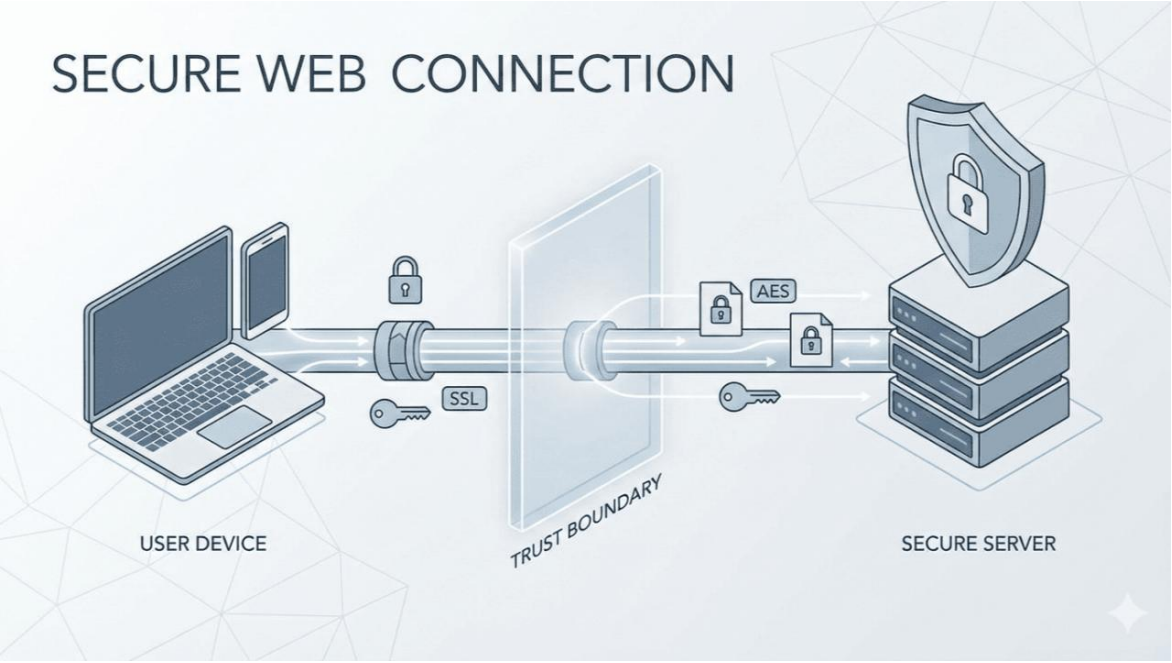
A támadó nem feltör – kihasznál

- Szabványos HTTP kérések
- Böngésző normál működése
- Logikai tervezési hibák

Trust boundary (bizalmi határ)

A webbiztonság egyik legfontosabb fogalma a **trust boundary**, azaz a bizalmi határ

- Böngésző nem megbízható
- Felhasználói input nem megbízható
- Hálózat nem megbízható
- Szerver = ellenőrzési pont, megbízható



XSS: Cross-Site Scripting

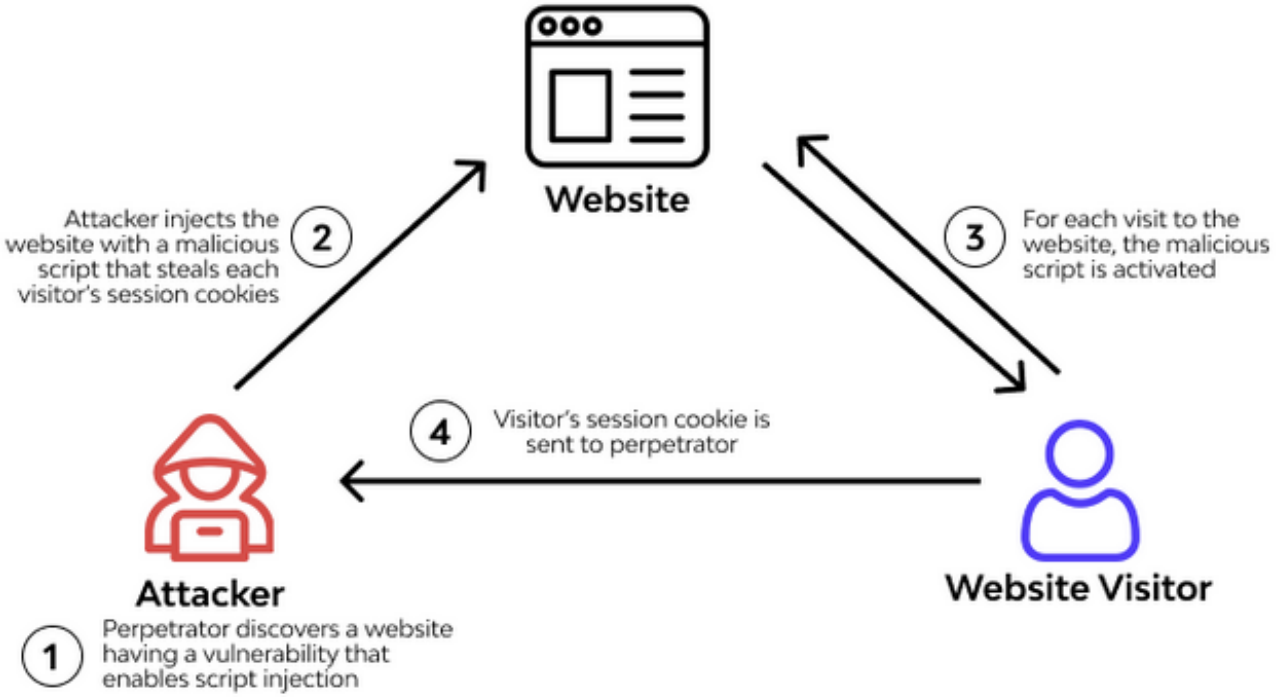
- **Kliensoldali** támadás
- **Idegen JavaScript** kód fut le
- A **felhasználó** böngészőjében
- **Megbízható** weboldal kontextusában
- Nem a szervert támadja

XSS következménye:

- **Session** ellopása
- Felhasználó nevében **műveletek**
- **Adatlopás**
- Felhasználó **megtévesztése**

XSS alapelve

- **Felhasználói input** megjelenik
- **Kontextus** nélküli beillesztés
- HTML / JS / attribútum kontextus
- Hiányzó **escaping és validáció**



XSS típusok

- **Stored XSS** – eltárolt

```
<div class="comment">
  {{comment.text}}
</div>
```

- **Reflected XSS** – „visszapattanó”

```
/search?q=<img src=x onerror=alert('XSS')>
```

- **DOM-based XSS** – frontend oldali

```
document.getElementById("out").innerHTML =
  new URLSearchParams(location.search).get("msg");
```

```
/dashboard?msg=<svg onload=alert('XSS')>
```

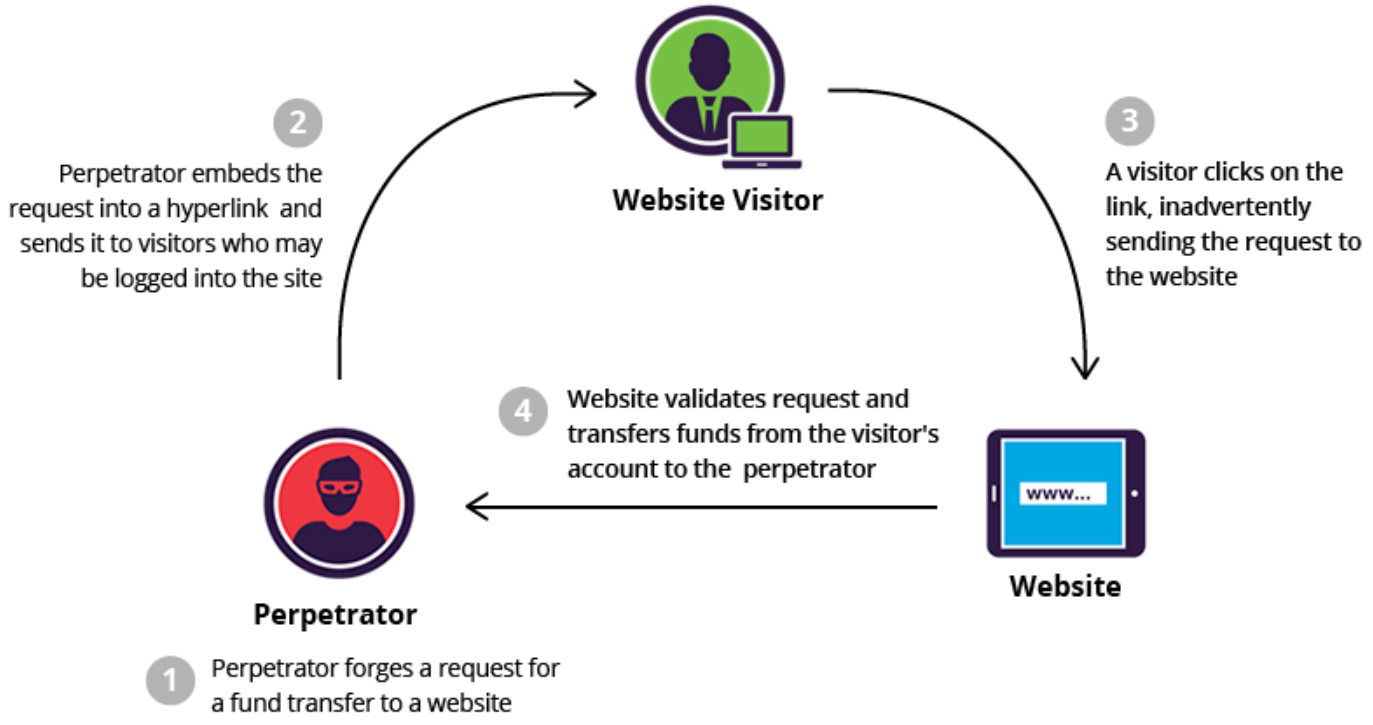
XSS elleni védekezés

- **Output encoding** minden felhasználói adatnál
- **Safe-rendering: Biztonságos** megjelenítés
- **DOM-based XSS** elkerülése
- Content Security Policy (**CSP**)
- **Cookie** és **Session** védelem
- **Input validáció**

CSRF: Cross-Site Request Forgery

- Böngésző automatikus viselkedése
- Cookie-k mindig elküldődnek
- Felhasználó tudta nélkül
- Hiteles, de nem szándékos kérés

CSRF lépései



- Felhasználó bejelentkezik
- Session aktív
- Rosszindulatú oldal betölt
- Böngésző automatikusan küldi a cookie-t
- Webalkalmazás végrehajtja a kérést

```
<img src="https://bank.hu/transfer?to=attacker&amount=100000">
```

CSRF elleni védekezés

- **CSRF token**
  - Egyedi, véletlen érték, amit a szerver generál
  - A kliens elküldi minden kéréshez
  - Kérésenként ellenőrizve
- **SameSite** kiegészítés
- **Same-Origin Policy** (alapértelmezett)

CORS: Cross-Origin Resource Sharing

Cél: megakadályozni az adatolvasást más domainről

- **Böngésző** oldali szabály
- **Same-Origin Policy** (SOP) alapból tiltja a cross-origin adatolvasást
- Nem backend védelem, hanem **böngésző** által nyújtott **védelem**
- Nem véd a nem-browser kliensektől

Cookie security

- **Secure** – a cookie **csak HTTPS kapcsolaton** keresztül kerül elküldésre
- **HttpOnly** – a cookie-hoz **JavaScript nem fér hozzá**
- **SameSite** – a cookie **cross-site** helyzetben **korlátozva** kerül elküldésre
- Együtt alkalmazva hatásosak

Input validation

Aranyszabály: **Soha ne bízz az inputban!**

- Minden input gyanús
- Ellenőrzés kötelező
- Formátum, tartomány, hossz
- Backend felelősség

OWASP Top 10

- **OWASP**: Open Web Application Security Project
- **Leggyakoribb** webes **kockázatok** listája
- **Tapasztalaton** alapúló lista
- Tervezési iránytű
- Közös nyelv

OWASP Top 10 – 2025

- A01: Broken Access Control
- A02: Security Misconfiguration
- A03: Software Supply Chain Failures
- A04: Cryptographic Failures
- A05: Injection
- A06: Insecure Design
- A07: Authentication Failures
- A08: Software or Data Integrity Failures
- A09: Security Logging & Alerting Failures
- A10: Mishandling of Exceptional Conditions

Összefoglalás

- Web = vezérlési felület
- **XSS** és **CSRF** alapfogalmak
- **CORS** és **cookie** szerepe
- OWASP Top 10