Webbiztonság – Gyakorlati feladatok

Ez a dokumentum a Webbiztonság alapjai című előadáshoz kapcsolódó gyakorlati anyagot tartalmazza. A gyakorlat célja nem a támadások implementálása, hanem a böngésző viselkedésének megfigyelése és a biztonsági kockázatok felismerése.

Gyors technikai feltételek

- Internetkapcsolattal rendelkező számítógép
- Modern böngésző (Chrome vagy Firefox)
- Böngésző DevTools használata (F12)
- Nincs szükség telepítésre vagy buildelésre

Demo oldal

A gyakorlat egy Python demo programot használ, amely két helyi (localhost) szervert indít:
• **App**: http://localhost:8000
• **Evil**: http://127.0.0.1:8001

Indítás (Windows):
1) Töltsd le a **server.py** Python scriptet a "*webbiztonsag-demo*" könyvtárból.
2) Nyiss PowerShell-t a mappában.
3) Futtasd: **python.exe server.py**
4) Nyisd meg a böngészőben: **http://localhost:8000**

Megjegyzés: a demo HTTP-n fut, ezért a Secure cookie flag viselkedése eltérhet (a böngésző eldobhatja).

1. Gyakorlat – XSS és input viselkedés

Cél: Megérteni, hogyan jelenik meg a felhasználói input a weboldalon, és miért vezethet ez XSS sebezhetőséghez.

Feladat 1 – Input megfigyelése

1. Nyisd meg az helyi demo oldalt (http://localhost:8000).
   Kattints az XSS demo "**unsafe megjelenítés**" linkre.
2. Írj be egy egyszerű szöveget az űrlap mezőbe (pl. 'Hello világ').
3. Figyeld meg, hogyan jelenik meg az oldalon.

Kérdések:
- A böngésző honnan tudja, hogy ez szöveg?
- Ki felelős a biztonságos megjelenítésért?

Feladat 2 – Nem várt input

1. Írj be speciális karaktereket tartalmazó szöveget, például egy script kódot:

```
<script>alert('XSS demo')</script>
```

2. Figyeld meg a megjelenítést.

Kérdések:
- A böngésző megkülönbözteti-e a szöveget és az utasítást?
- Hol kellett volna ezt kezelni?

Most ismételd meg a feladatokat az XSS demo "**safe megjelenítés**" linkre kattintva.

Tanulság

Az XSS alapvető oka, hogy a felhasználói input ellenőrzés és "output enconding" nélkül kerül vissza az oldalra.

2. Gyakorlat – Cookie, CSRF és CORS

Cél: Megérteni a böngésző automatikus viselkedését és annak biztonsági következményeit.

Feladat 3 – Cookie-k megfigyelése

1. Nyisd meg a demo oldalt. Válaszd az "**Account**" példát, és nézd meg a különböző Cookie (SameSite) beállításokat.
2. Nyisd meg a böngészőben: DevTools → Application/Storage → Cookies.
3. Vizsgáld meg a cookie flag-eket.

Kérdések:
- Mit jelent a Secure?
- Miért fontos a HttpOnly?
- Mit csinál a SameSite?

Feladat 4 – CORS hiba értelmezése

1. Nyisd meg a demo oldalt. Válaszd a "**CORS demo**" példát, amely külső API-t próbál hívni.
2. Próbáld ki a megadott CORS példákat.
3. Nyisd meg a böngészőben: DevTools → Console vagy DevTools → Network.
4. Vizsgáld meg a CORS hibát.

Kérdések:
- Ki dobta a hibát: a szerver vagy a böngésző?
- Mit NEM véd meg a CORS?

Feladat 5 – CSRF viselkedése

1. Nyisd meg a demo oldalt. Válaszd a "**Account**" példát, és jelentkezz be.
2. Állítsd be a session cookie-hoz egy SameSite paramétert.
3. Válaszd a "**CSRF demo**" példát.
3. Nyisd meg a böngészőben: DevTools → Network.
4. Gondold végig, miért küldi el a böngésző automatikusan a cookie-t.

A feladatot ismételd meg különböző SameSite beállításokkal. Figyeld meg, hogy mikor melyik esetben küldi el a session cookie-t.

Kérdések:
- A szerver mit lát ebből?
- Hogyan lehetne megkülönböztetni a szándékos és nem szándékos kérést?
- Melyik esetben küldi a böngésző a cookie-t, és mikor nem küldi?