IT Biztonság (VIHIAC01)
Házi feladat

Web Biztonság

Szerzők:
Gazdag András, Futóné Papp Dorottya

2021. március 31.

Tartalomjegyzék

1. Általános információk

Ebben a házi feladat kiírásban hat feladat található, melyek a web témakörhöz kapcsolódnak. A feladatok megoldásához szükséges háttér információk a feladatok leírásában találhatók. A feladatok megoldásához a BME Cloudban elérhető virtuális gép használata szükséges. A mérési elrendezés bemutatása a Függelék A részében található.

Megoldásként feladatonként egy fájl beadását várjuk, melyek tartalmazzák külön-külön egy-egy feladat megoldásának leírását. A beadott fájlok elvárt tartalmára és formátumára vonatkozó információk a feladatok leírásában találhatók.

A házi feladat beadási határideje után, egy Moodle kvízt is ki kell majd tölteni, melyben olyan kérdések lesznek, melyeket a feladatok megoldása ismeretében könnyen meg lehet válaszolni. A kvízt fogjuk pontozni, az lesz a házi feladatra kapott pont. Maximum 10 pont szerezhető így. A megszerzett pontszámot azonban 20%-kal csökkentjük, ha a megoldások beadása a határidő után történik.

Az esetleges csalások kiszűrése érdekében a beadott megoldások egy véletlen választott részhalmazát összevetjük a hozzájuk tartozó kvíz eredményével, és ha ellentmondásokat, inkonzisztenciákat tapasztalunk, vagy ha úgy ítéljük meg, hogy a kvízben adott helyes válaszok alátámasztása nem található meg a beadott megoldásban, akkor pontszám levonást alkalmazunk.

2. Feladatok

2.1. Adminisztrátori hozzáférés

2.1.1. A feladat

Találjon egy sérülékeny bemenetet, ahol a hibát kihasználva képes az adminisztrátor nevében bejelentkezni. A feladatot SQL injectionnel kell megoldani.

Tanácsok a megoldáshoz: A bejelentkezést végző szerver oldali kód az alábbi linken elérhető: https://github.com/bkimminich/juice-shop/blob/master/routes/login.js

2.1.2. A beadandó megoldással kapcsolatos elvárások

A beadandó megoldásnak egy `1-feladat.txt` nevű fájlnak kell lenni a következő tartalommal:

- 1. sor: a támadásnál használt felhasználónév
- 2. sor: a támadásnál használt jelszó

Az összes feladat megoldását egyetlen ZIP fájlba csomagolva várjuk.

2.2. Hibás keresés

2.2.1. A feladat

Keressen az alkalmazásban olyan bemenetet, amely tartalma a szerverrel megvalósított kommunikáció során megjelenik a válaszban is! Hajtson végre egy DOM alapú XSS támadást, amely során egy HTML elemet és azzal együtt JavaScript kódot szúr be a weboldalba, amelyet könnyű felismerni (például: `<iframe>` tag)!

Egy sérülékeny kódrészt tartalmaz például ez a része az alkalmazásnak: https://github.com/bkimminich/juice-shop/blob/master/frontend/src/app/search-result/search-result.component.html

Tanácsok a megoldáshoz:

- A gyakran használt XSS támadásokat több helyen is összegyűjtötték már. Egy ilyen lista ennek a feladatnak a megoldásában is segítség lehet, a próbálkozásokhoz innen javasolt ötleteket gyűjteni: https://owasp.org/www-community/xss-filter-evasion-cheatsheet

- A feladat megoldása során a cél mindig a JavaScript kód futtatás demonstrálása, amit a legegyszerűbben az `alert()` függvény meghívásával lehet megtenni.

- A támadásokat gyakran megnehezíti, hogy string beszúrása során figyelni kell, hogy az idézőjelek megfelelően legyen használva. Az eredeti forráskód is rend szerint használja a `'` és a `"` karaktereket. Ilyen esetekben célszerű lehet még a `` ` `` (backtick) karaktert is kipróbálni, amely szintén használható string eleje és vége jelként.

2.2.2. A beadandó megoldással kapcsolatos elvárások

A beadandó megoldásnak egy `2-feladat.txt` nevű fájlnak kell lenni a következő tartalommal:

- a támadásnál használt bemenet

Az összes feladat megoldását egyetlen ZIP fájlba csomagolva várjuk.

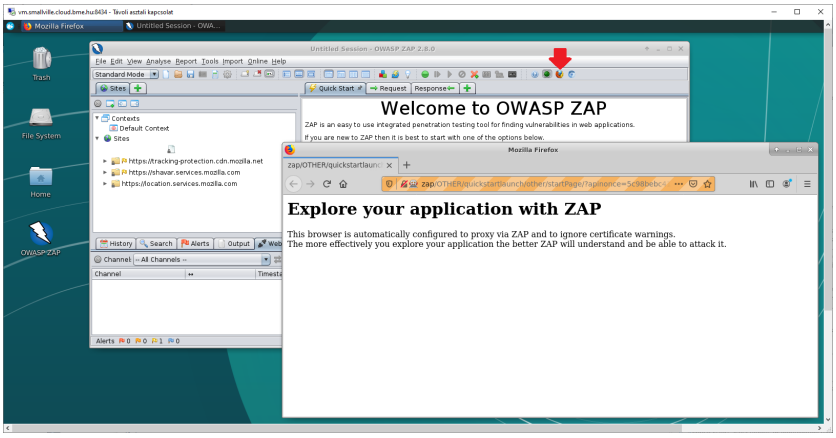
2.3. Bevezető az OWASP ZAP használatába

2.3.1. A feladat

A házi feladat több részéhez érdemes az OWASP Zed Attack Proxy (ZAP) tesztelő eszközt használni. A szoftver megismeréséhez teljesítenie kell a beépített bevezetőt.

1. Indítsa el a ZAP-ot az asztalon található parancsikon segítségével! (Ez eltarthat egy ideig a háttérben futó konfigurációs lépések miatt, 5 percet mindenképp érdemes várni.)

2. A ZAP felületén teljes képernyős módban a menüsor alatt találhatóak apró ikonok. Az ikonsorozat jobb szélén két olyan ikon van, amelyik előre konfigurált böngészőt nyit meg ZAP-pal történő proxy-záshoz.



1. ábra. ZAP-paroxy-val konfigurált Firefox indítása

Indítsa el a Firefox böngészőt az ikonon keresztül! Onnan tudja, hogy ZAP-on keresztül proxy-zik böngészés közben, hogy az URL beviteli mező háttere narancssárga, ahogy azt a 1. ábra is mutatja.

3. Látogassa meg a `localhost:3000`-t és várjon, amíg megjelenik a ZAP HUD. A HUD-ot a 2. ábra mutatja. A HUD-on keresztül indítsa el és teljesítse a bevezetőt!

2.3.2. A beadandó megoldással kapcsolatos elvárások

A beadandó megoldásnak tartalmaznia kell az alábbi képernyőképeket:

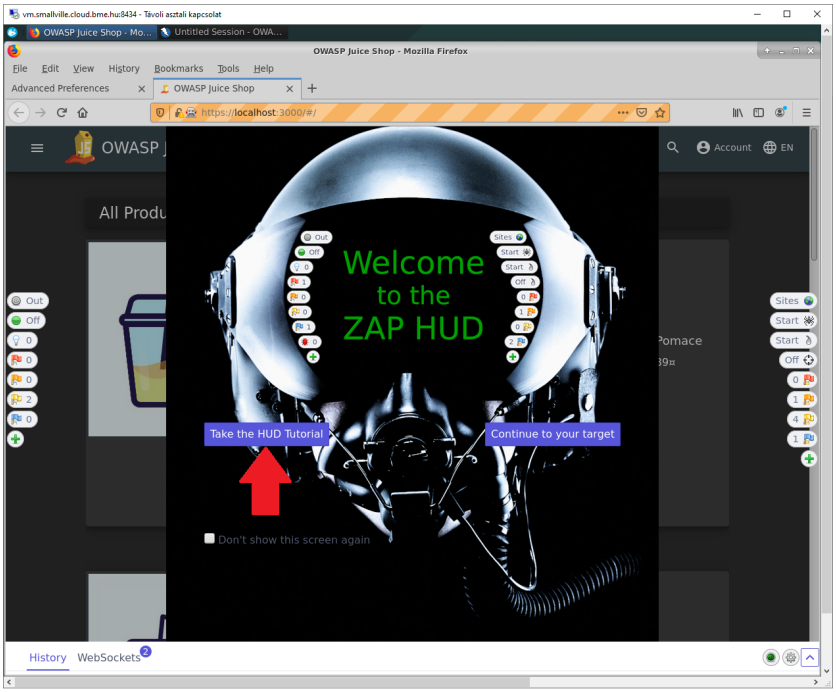
- HUD bevezető feladatainak megoldása

Az összes feladat megoldását egyetlen ZIP fájlba csomagolva várjuk.

2.4. Eredményjelző

2.4.1. A feladat

Ebben a feladatban meg kell találni az OWASP Juice Shopban elrejtett eredményjelző oldalt. Az eredményjelzőn több feladat is fel van sorolva, ezek közül néhány adja a kötelező házi feladatot. A téma iránt érdeklődő hallgatók természetesen a többi feladattal is foglalkozhatnak. Ez az oldal rejtett, vagyis csak akkor látogatható meg, ha ismerjük a rá mutató URL-t és azt közvetlenül látogatjuk meg.



2. ábra. ZAP HUD

Tanácsok a megoldáshoz:

1. Az OWASP Juice-Shop alkalmazás angularjs-t használ. Emiatt egyrészt a weboldalak route-olása kliensoldalon történik, másrészt az URL-ek `<domain név>/#/<oldal name>` formában érhetőek el.

2. Az eredményjelző szót angolra scoreboardként fordítjuk.

2.4.2. A beadandó megoldással kapcsolatos elvárások

A beadandó megoldásnak tartalmaznia kell az alábbi képernyőképeket:

- Az eredményjelző oldal route-olását végző javascript kód kijelölve
- Eredményjelző oldal böngészőben megnyitva

Az összes feladat megoldását egyetlen ZIP fájlba csomagolva várjuk.

2.5. Bizalmas fájlok elérése

2.5.1. A feladat

Az OWASP Juice Shop alkalmazás egyik sérülékenysége, hogy bizalmas fájlokat lehet autentikáció nélkül elérni. A fájlok eléréséhez meg kell találnia a rájuk mutató URL-eket.

Tanácsok a megoldáshoz: Mivel autentikáció nélkül elérhetőek a fájlok, ezért a weboldal linkjeinek letapogatásával könnyen megtalálhatóak.

2.5.2. A beadandó megoldással kapcsolatos elvárások

A beadandó megoldásnak tartalmaznia kell az alábbi képernyőképeket:

- A bizalmas fájlokat tartalmazó mappa böngészőben megnyitva
- A bizalmas fájlokat tartalmazó mappa tartalma az OWASP ZAP által letapogatott oldalak között

Az összes feladat megoldását egyetlen ZIP fájlba csomagolva várjuk.

2.6. A nulla csillagos visszajelzés

2.6.1. A feladat

Az előző feladatok során kiderült, hogy sok probléma van az OWASP Juice Shoppal. Erről természetesen az üzemeltetőknek is tudniuk kell, ezért visszajelzést kell küldenie a webalkalmazáson keresztül. Hogy az üzemeltetők számára is világos legyen a helyzet súlyossága, olyan visszajelzést küldjön, ami egyetlen csillagot sem ad a weboldalnak!

Tanácsok a megoldáshoz: Visszajelzést a `Customer Feedback` menüponton keresztül küldhet. Érdemes megnézni, hogy milyen HTTP kérés-válasz pár eredményeként adható le új visszajelzés.

2.6.2. A beadandó megoldással kapcsolatos elvárások

A beadandó megoldásnak tartalmaznia kell az alábbi képernyőképeket:

- Érvényes visszajelzés elküldéséhez tartozó csomag a History fülön
- Visszajelzés visszajátszásához használható felugróablak a HUD-on

Az összes feladat megoldását egyetlen ZIP fájlba csomagolva várjuk.

Függelék

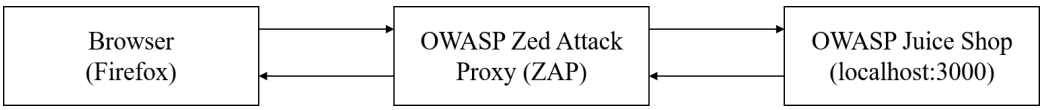
A. Mérési elrendezés a házi feladathoz

A 3. ábra mutatja a házi feladat során használt mérési elrendezést. A virtuális gépre mindegyik komponenst előre telepítettük.

Az OWASP Zed Attack Proxy[1] (ZAP) egy nyílt forráskódú webes biztonsági tesztelő szofter. Képes HTTP forgalmat elkapni és lehetővé teszi a felhasználó számára, hogy tetszőlegesen módosítsa a fejlécek és a body tartalmát. Többféle automatizált tesztelési technikát is implementál, pl. fuzzing, de ezek nem képezik a házi feladat részét. A működéshez a szoftver előkonfigurálja a böngészőt, hogy a HTTP és HTTPS kéréseket is a ZAP-on keresztül küldje a böngésző az elemzett webalkalmazásnak (ezt nevezzük proxy-zásnak).

A feladatok során a OWASP Juice Shop[2] webalkalmazást használjuk. Ez egy nyílt forráskódú webalkalmazás, amit kifejezetten azért fejlesztenek, hogy sérülékenységeket tartalmazzon. A webalkalmazást oktatási célokra lehet használni: webes biztonsági tréningeken lehet vele tipikus webes sérülékenységeket demonstrálni kontrollált környezetben. A virtuális gépben a Juice Shopot közvetlenül a `localhost:3000`-es címen találjuk és tartozik hozzá egy indító szolgáltatás, melynek neve `juice-shop`. A webalkalmazás a frontendhez Angulart használ, aminek eredményeként a weboldalakat egy `#/` kezdetű útvonallal érjük el, pl. `localhost:3000/#/search`. A webalkalmazásban többféle feladat található 6 nehézségi szintre osztva, némelyikhez tartoznak tippek a megoldás elkezdéséhez. A házi feladat megoldása ebből a halmazból néhány feladat megoldását jelenti.

[1] https://owasp.org/www-project-zap/
[2] https://owasp.org/www-project-juice-shop/



3. ábra. A házi feladat során használt mérési elrendezés

B. A virtuális környezet debuggolása

A következő bash parancsokat érdemes használnia a virtuális gép működtetéséhez és hibakezeléséhez:

```
# szolgáltatás indítása
sudo service <service name> start

# szolgáltatás újraindítása (pl. konfiguráció változás miatt)
sudo service <service name> restart

# szolgáltatás állapotának lekérdezése
sudo service <service name> status

# szolgáltatás leállítása
sudo service <service name> stop

# naplóbejegyzések olvasása
sudo journalctl -xe

# nyitott portok és hozzájuk tartozó folyamatok listázása
sudo netstat -tulpn

# keresés egy parancs kimenetében
<command> | grep <search condition>

# futó folyamatok listázása
ps aux
top

# szövegszerkesztő megnyitása parancssorból
mousepad <file name>
nano <file name>
```

C. Háttér

Egy weboldal betöltése a böngészőben több lépésből áll. Az első lépés a Universal Resource Locator (URL) által azonosított erőforrás letöltése. Az

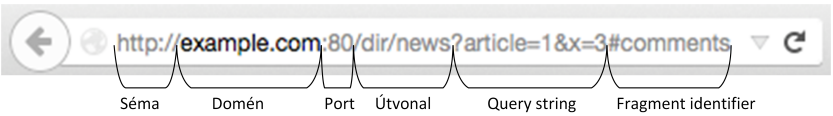
URL-nek több része van, ahogy azt a 4. ábra mutatja. Egyes részekhez a böngészők alapértelmezett értékeket rendelnek: ha nincs külön kiírva, akkor a séma alapértelmezett értéke `http://`, a porté `:80` és az útvonalé `/`. A query sztring és a fragment identifier komponensek opcionálisak.

A böngésző ezután elküldi az útvonalat és a query stringet egy kérésben annak a webszernek, amelyiket a domén azonosít. Ez a lépés általában megköveteli a böngészőt futtató számítógépet, hogy egy vagy több DNS kéréssel megállapítsa a doménhez tartozó IP címet. Az elküldött kérés a HyperText Transfer Protocol (HTTP) nevű protokollt követi, ami a kliens-szerver architektúrára épülő kétirányú kommunikációs protokoll. Egy példa kérés-válasz interakciót mutat be a 5. ábra. A kérésben szerepel egy HTTP `verb` (pl. `GET`, `POST` or `PUT`), a lekérdezett útvonal és a query string, valamint a kérés során használt protokoll verzió. A kérés ezen felül több fejlécet is tartalmaz, amivel többlet információt szolgáltat a szervernek. Néhány széleskörben használt fejléc mező:

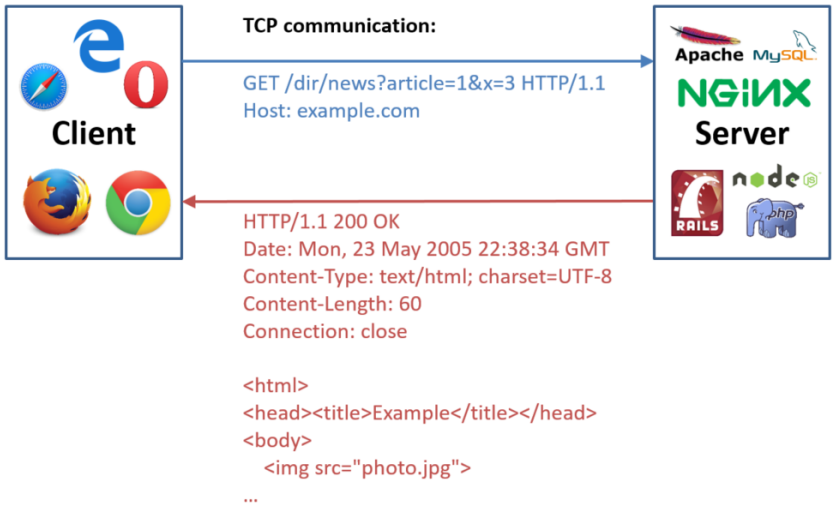
- `Accept`: Válaszként elfogadott média típusok (pl. `text/html`, `application/xml`)
- `Host`: A szerver doménje és opcionálisan portja
- `User-Agent`: a kérést küldő klienst azonosító karaktersorozat (pl. `Mozilla/5.0 (X11; Linux x86_64; rv:12.0)`)
- `Referer`: A legutóbb meglátogatott weboldal címe, ami a böngészőt a most lekért oldalra irányította

A szerver válaszában szerepel a protokoll használt verziója, a HTTP *státusz kód*, további fejléc mezők és, amennyiben a lekérés sikeres, a kért erőforrás. A HTTP státusz kódokat öt csoportra osztjuk:

- Információs (`1xx`), pl. `100 Continue` and `101 Switching Protocols`
- Sikeres (`2xx`), pl. `200 OK`, `201 Created`



4. ábra. Universal Resource Locator



5. ábra. Példa HTTP kérés-válasz pár

- Átirányítás (`3xx`), pl. `301 Moved Permanently`, `307 Temporary Redirect`
- Kliens hiba (`4xx`), pl. `400 Bad Request`, `403 Forbidden`, `404 Not Found`
- Szerver hiba (`5xx`), pl. `500 Internal Server Error`, `502 Bad Gateway`, `503 Service Unavailable`

A letöltött erőforrástól függ a böngészőbeli feldolgozás. Szkriptek esetén a böngésző futtatja a leírt parancsokat. Weboldal (HyperText Markup Language (HTML) formátumú adat) esetén a böngésző memóriába olvassa az adatot, ha szükséges, lekéri az alerőforrásokat (képek, szkriptek, frame-ek, stb.), majd ha minden erőforrás rendelkezésre áll, megjeleníti a tartalmat a felhasználónak. Végezetül a böngésző egy esemény ciklust futtat, amiben egyes eseményekről (pl. űrlapok beküldése) értesíti a Javascript motort, amely lekezeli az eseményeket.