Virtual Machine: Debian Wheezy 64bit

Felhasználónév: secu

Jelszó: csicskalangos

Webgoat: sérülékenyre tervezett applikáció, amelynek célja az it security gyakorlatias elsajátítása

Letölthető: https://github.com/WebGoat/WebGoat-Legacy/releases

Megoldások: http://webappsecmovies.sourceforge.net/webgoat/

Webgoat indítása a virtuális gépen:

Belépünk a /home/secu/Downloads mappába:

cd /home/secu/Downloads

Webgoat futtatása:

java –jar WebGoat-6.0.1-war-exec.jar

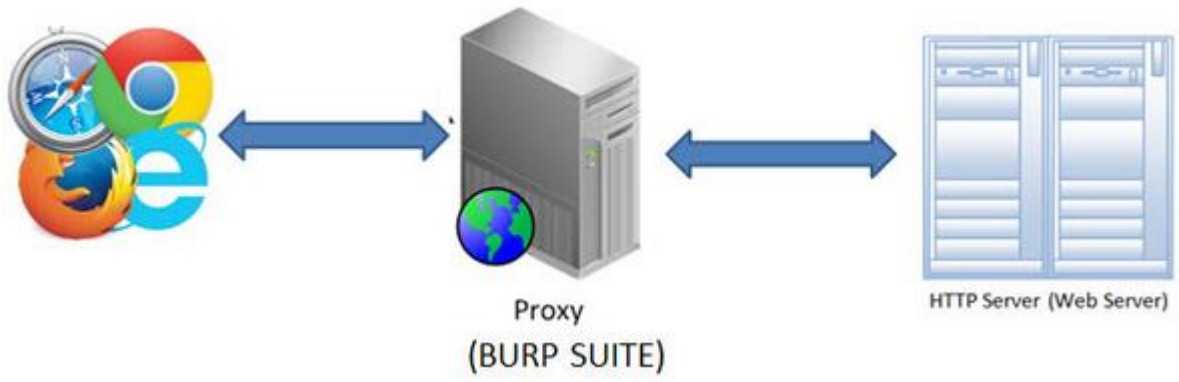
A webgoat ezek után a 8080-as porton fog figyelni

A feladatok könnyebb megoldása érdekében egy proxyt fogunk használni. Ez minden kiküldött és bejövő adatot elkap és lehetőséget ad rá, hogy azokat manuálisan módosítsuk.

Ez a proxy a burp, amit a következő paranccsal indíthatunk (lehetőleg új terminált nyitva):

java –jar burpsuite_free_v1.6.jar

A konfigurálandó architektúra:

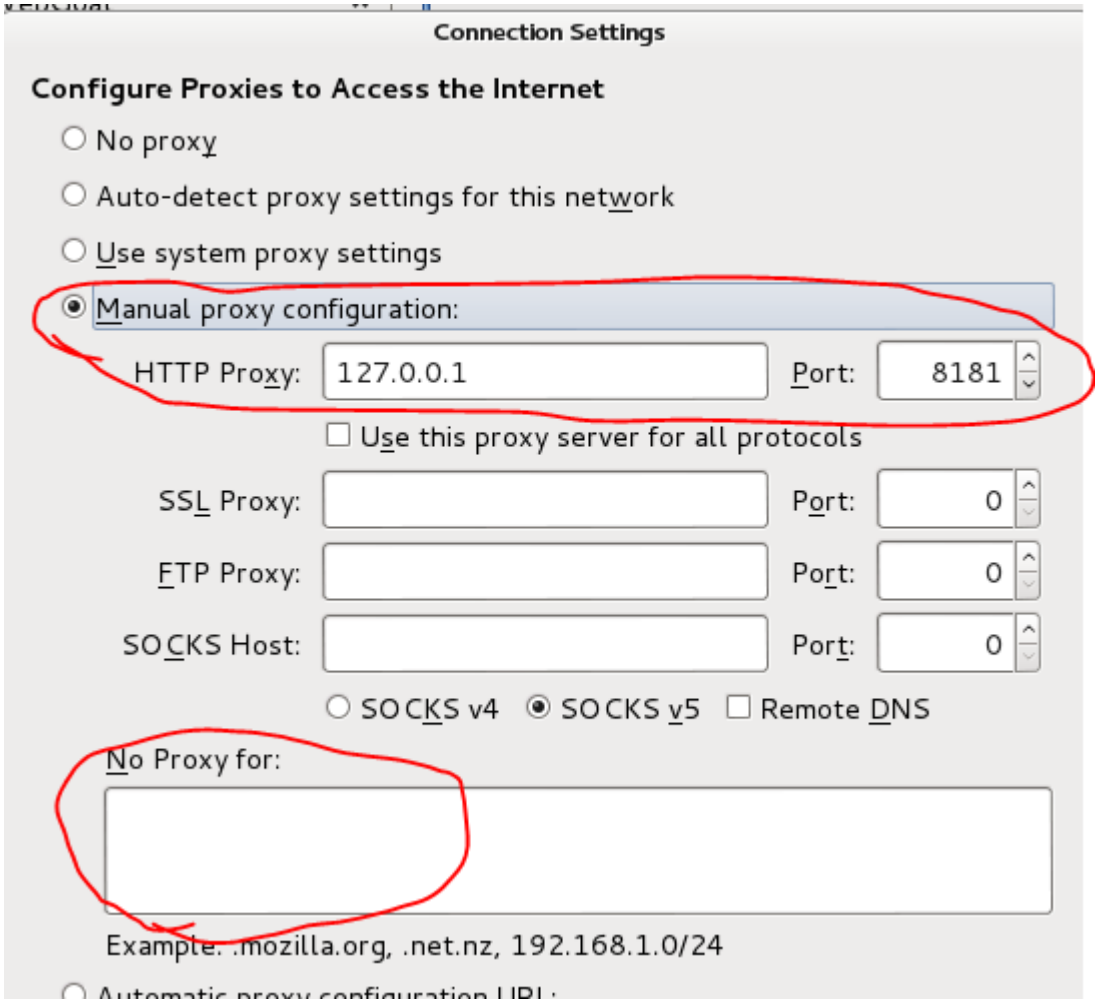


Látható a fenti képről, hogy a böngésző minden kérést a proxyn keresztül küld, és onnan is kap minden választ. Ehhez a használt Iceweasel böngészőben be kell állítani, hogy proxyt használjon, és az a 127.0.0.1-es címen a 8181-es porton figyel. (Fontos kikapcsolni azt a funkciót, hogy localhostra ne használjon proxyt, hiszen most pont ezt akarjuk elérni).

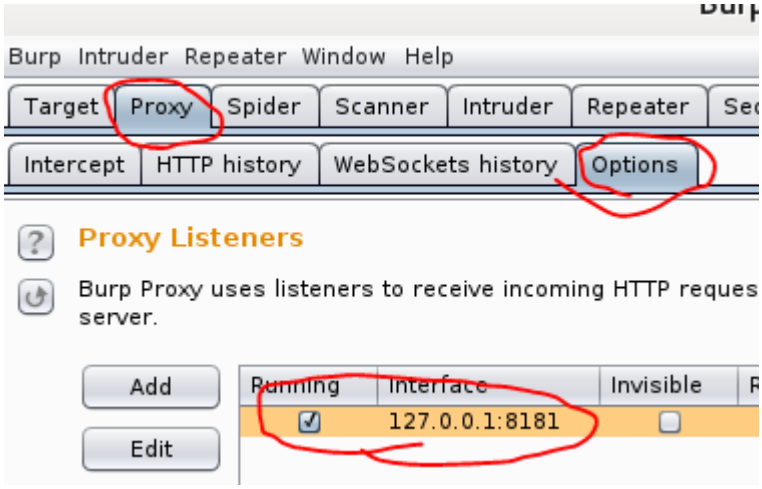
Ezek után a Burp-ben is be kell állítanunk egyrészt, hogy a adatokat a 8181-es portra küldje, és ott figylejen, valamint, hogy a webszerverünk, a webgoat a 8080-as porton van.

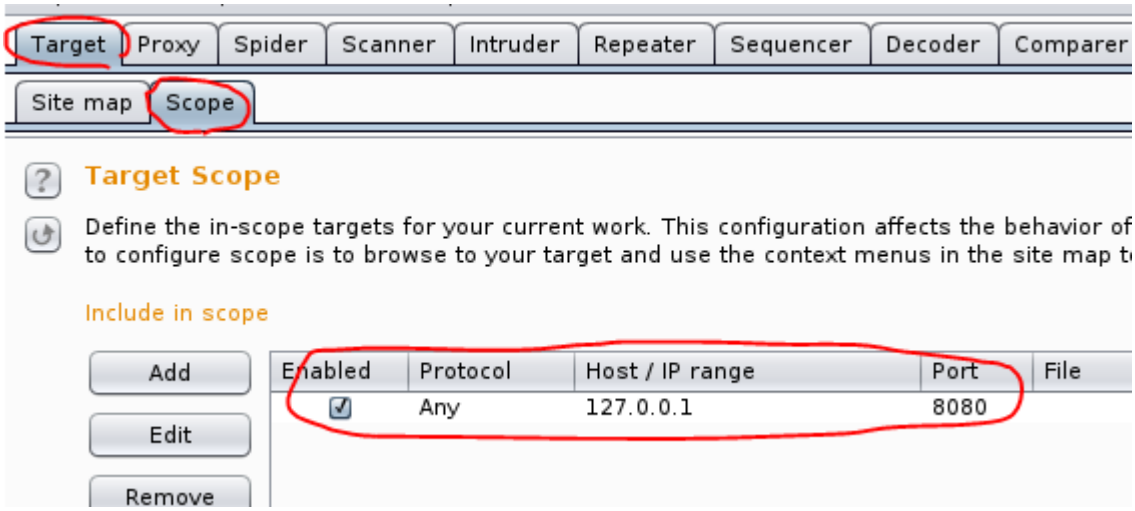
Képregény a beállításokhoz

(A böngésző be van állítva a laborban, csak a Burpöt kell átkonfigurálni)



Burp beállításai:





Ezek után a fenti architektúra működőképes a WebGoat szerverrel való kommunikáció lehallgatásához, módosításához.

A webgoat-ot a 127.0.0.1:8080/WebGoat linken érjük el.

Figyeljünk rá, hogy ezek után a Burp elkapja a csomagokat, úgyhogy vagy egyesével engedélyeznünk kell mindent, vagy ideiglenesen kapcsoljuk ki a funkciót.

SQL cheatsheet: http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/

Injection flaws:

1: String sql injection:

Burpben kapcsoljuk be az intercepted, és itt vizsgáljuk a csomagokat ezentúl.

A feladatnak megfelelően kiválasztjuk Larryt, és tetszőleges jelszóval megpróbálunk belépni.

Burpbe belépve látjuk az elküldött adatokat, ahol minket az employee_id=101&password=ourpass&action=Login

A megadott jelszóval valószínű nem less jó a belépés, de itt kiaknázhatjuk az sql injectionben rejlő lehetőséget. Írunk egy olyan kódot, ami azt mondja, hogy bármilyen esetben tudjunk belépni.

Az sql lekérdezés, amibe injektálunk az a feladata, hogy igazzal térjen vissza, ha helyes felhasználó-jelszó párost adtunk meg, és hamissal, ha nem. Emiatt az a célunk, hogy elérjük, hogy mindig igaz legyen.

Ennek megfelelően a password attribútumot átírjuk 2'+or+'1'='1 –re.

!!!Stage 3: Numeric SQL injection

Larry jelszava 'larry'.

Blindg sql

101 AND 1=((SELECT count(*) FROM pins WHERE cc_number = '1111222233334444' AND pins.pin>=0 AND pins.pin<=10000)

XSS: Cross site scripting

Stage 1: Stored Xss

Belépünk Tom fiókjába, akinek a jelszava tom.

Ezek után frissítjük a profilban az utcanév adatot a következők szerint:

"><script>alert('Hacked!!')</script>

MI történt? Az első rálaciósjel lezárta a megkezdett HTML taget, az előtte lévő idézőjel pedig a benne lévő sztinget fejezte be. Aztán egy javascript kódot injektáltunk, amit aztán el is mentettünk a szerverre. Így bárki megnézi a Tom által szerkeszthető, saját profilját annál lefut a szkript.

Stage 5: Reflected XSS

Reflected XSS attack