Web Application Attack and Audit Framework
W3AF

Bevezetés

A Web Application Security Consortium (WASC) felmérését készített, melyben azt vizsgálták, hogy az interneten található web alkalmazások milyen fajta sebezhetőségektől szenvednek. Az eredmény a következő cégek által végzett automatizált, white illetve black box módszerrel végzett sérülékenység felderítéseken alapult:

- Booz Allen Hamilton
- BT
- Cenzic Hailstorm és ClickToSecure
- dblogic.it
- HP Application Security Center WebInspecttel
- Positive Technologies MaxPatrol
- Veracode Security Review
- WhiteHat Sentinel

Az eredmény a következő: a vizsgált webalkalmazások több mint 7%-a automatikusan feltörhető, kb. 7.72%-uk szenvedtt magas besorolású sebezhetőségtől. A részletes kézi white és black box módszerekkel történő sebezhetőség keresésnél a webalkalmazások 98,65%-ban találtak magas besorolású sérülékenységet. A leggyakoribb sebezhetőségek a következők voltak:

- Cross-Site Scripting
- Információszivárgás
- SQL Injection
- Predictable Resource Location.

Egy másik felmérés, melyet az Open Web Application Security Project (OWASP) készített, a tíz leggyakoribb sebezhetőséget kereste a 2006-os MITRE Sebezhetőségi Trendek című felmérése alapján.
Ennek eredményeként a következő sorrendet állították fel:

1. Cross-Site Scripting
2. Beszúrásos sebezhetőségek
3. Kártékony fájlfuttatás
4. Insecure Direct Object Reference
5. Cross-Site Request Forgery
6. Információszivárgás és nem megfelelő hibakezelés
7. Rossz autentikáció és munkamenet-kezelés
8. Insecure Cryptographic Storage
9. Biztosítatlan kommunikáció
10. URL hozzáférés letiltásának elmulasztása

A w3af projekt célja egy olyan nyílt forrású keretrendszer megalkotása volt, amely amellett, hogy alkalmas a fenti, elterjedt sebezhetőségek felderítésére és kiaknázására, tetszés szerint bővíthető az egyéni igények és a jövő kihívásainak kielégítése érdekében.

Mi is ez a w3af?

A w3af a Web Application Attack and Audit Framework rövidítése. A program magja és a hozzá illeszthető pluginek - amelyekkel a program funkcionalitása tetszés szerint kiegészíthető - Python nyelven íródtak, így biztosított a platform-függetlenség. A szkenner konzolos és grafikus felhasználói felülettel is használható. A w3af segítségével a fentebb említett minden sebezhetőségre

megvizsgálhatjuk webalkalmazásainkat. Mivel a program több mint 130 pluginnel rendelkezik, így a kiegészítések csoportosítva vannak. Ezek a csoportok a következők:

- discovery (új injektálási pontok keresése)
- audit (a discovery plugin által felfedezett pontokat használja fel az új hibák felderítéséhez)
- bruteforce (form és basic)
- evasion (IDS kijátszás)
- grep (minta keresés a HTTP válaszokban)
- mangle (kérések módosítása szabványos kifejezésekkel)
- output (az eredmény mentése)

A w3af és függőségek telepítése

Linux

Töltsük le a honlapról a programot tartalmazó archívumot és tömörítsük ki azt. A következő programcsomagokra van szükség a futtatás előtt:

- A Mag és a konzol függőségei:
  - fpconst-0.7.2
  - pygoogle
  - pywordnet
  - SOAPpy
  - pyPdf
  - Beautiful Soup
  - Python OpenSSL
  - json.py
  - scapy
- Grafikus felület függőségei:
  - python sqlite3
  - graphviz
  - pygtk 2.0
  - gtk 2.12

Az extlib mappában megtalálható a legtöbb függőség, ami hiányzik, azt magunknak kell letölteni és feltelepíteni. Ha kész vagyunk, használhatjuk is a programot. Itt egy kis segítség Python programok telepítéséhez:

```
w3af@pentester:/home/w3af/#  tar -xvf w3af-beta7-r1813.tar.bz2
w3af@pentester:/home/w3af/w3af/# cd w3af
w3af@pentester:/home/w3af/w3af/# cd trunk
w3af@pentester:/home/w3af/w3af/trunk/# cd extlib
w3af@pentester:/home/w3af/w3af/trunk/extlib/# cd fpconst-0.7.2
w3af@pentester:/home/w3af/w3af/trunk/extlib/fpconst-0.7.2/#  python setup.py install
```

A w3af frissítéséhez adjuk ki a következő parancsot a trunk könyvtárban.

```
w3af@pentester:/home/w3af/w3af/trunk# svn update
```

Ezzel letölthetjük a projekt Subversion tárolójában található legfrissebb verziót.

Windows

Windowsosra elérhető egy előre összeállított telepítőkészlet, amely elvégzi az alapvető beállításokat, azonban erősen támaszkodik a Python 2.5-ös értelmezőre, amelyet külön kell telepíteni (www.python.org). Amennyiben az egyes függvénykönyvtárak telepítése mégis meghiúsul, próbálkozzunk más verziójú Python értelmezővel!

A w3af használata

Választhatunk konzolos és grafikus felület közül. A konzolos felület indításához a w3af_console, a grafikus felület indításához pedig a w3af_gui parancsot írd be. Nézzük milyen argumentumokat adhatunk meg: írjuk be a következőt: w3af_gui -h

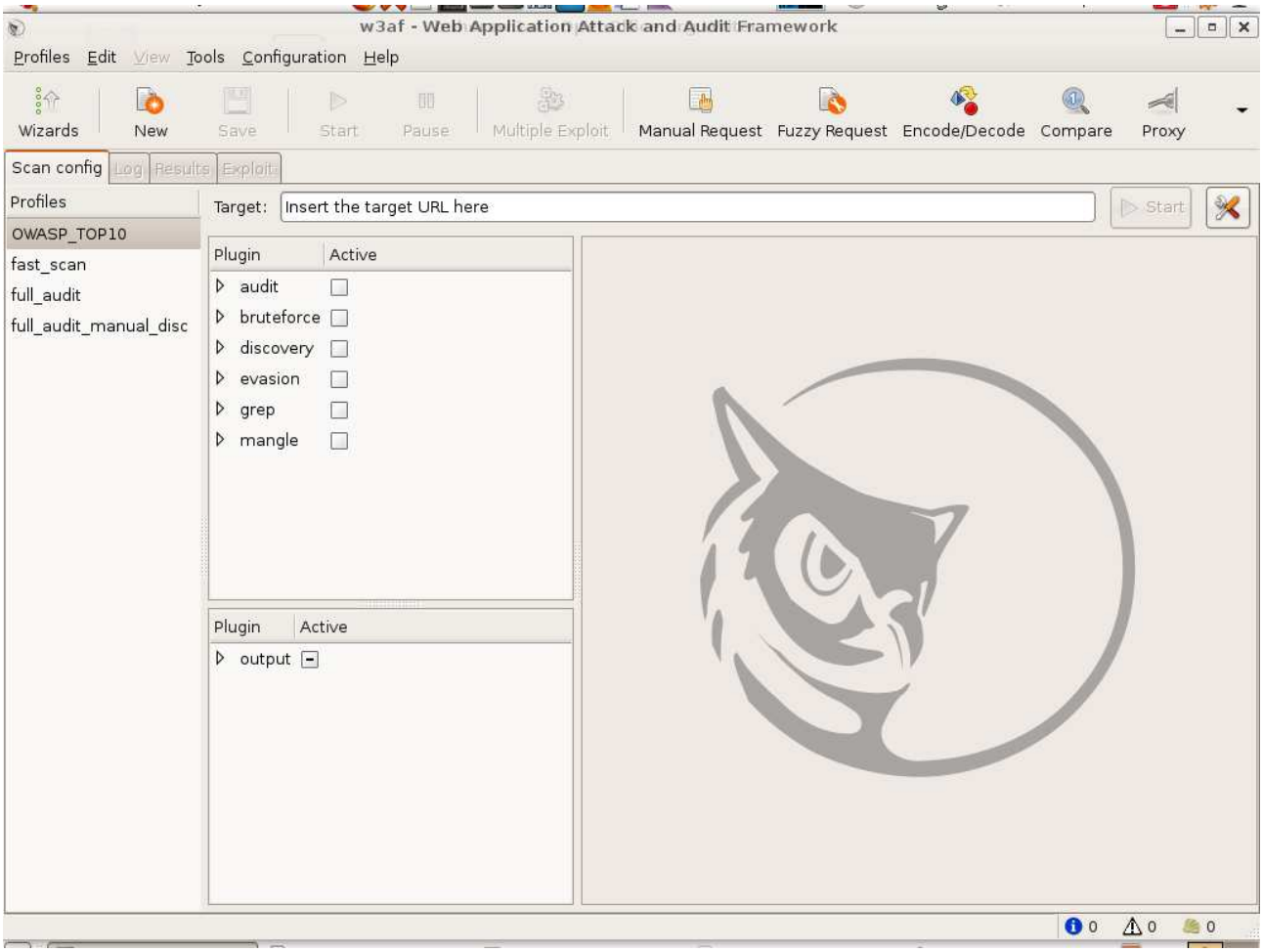
```
w3af@pentester:/home/w3af/w3af/trunk#  ./w3af_gui -h
w3af - Web Application Attack and Audit Framework

Options:
   -h            Print this help message.
   -s <file>       Execute a script file.
   -i <dir>        Directory where MSF is installed (only used to install the virtual daemon).
   -p <profile>    Run with the selected profile

http://w3af.sourceforge.net/
```

- -s<file>, itt megadhatjuk a script fájlt, ami tartalmazza a célpontot és a rajta futtatandó plugineket. Automatikusan lefut, nekünk nem kell semmit csinálni.
- -i<dir>, ezt megadva együtt használhatjuk a Metasploit Frameworkkel.
- -p<profile>, itt pedig azt a profilt adhatjuk meg, amit majd használni fogunk. Mi indítjuk el a szkennelést.

A konzolos felület használatához olvasd el a linkek között megtalálható UsersGuideot.

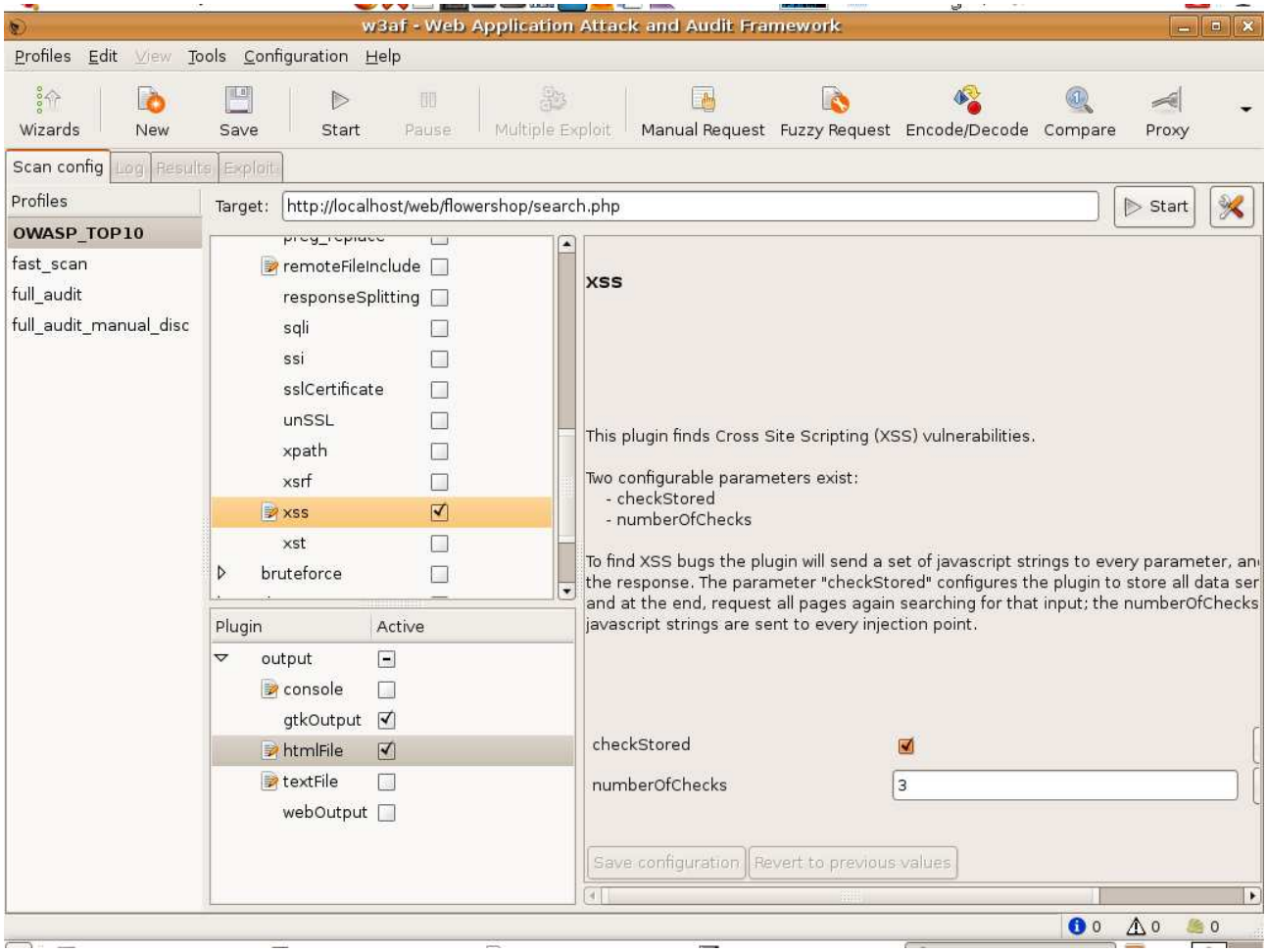
Indítsuk el a grafikus felületet:

```
w3af@pentester:/home/w3af/w3af/trunk #  ./w3af_gui
```

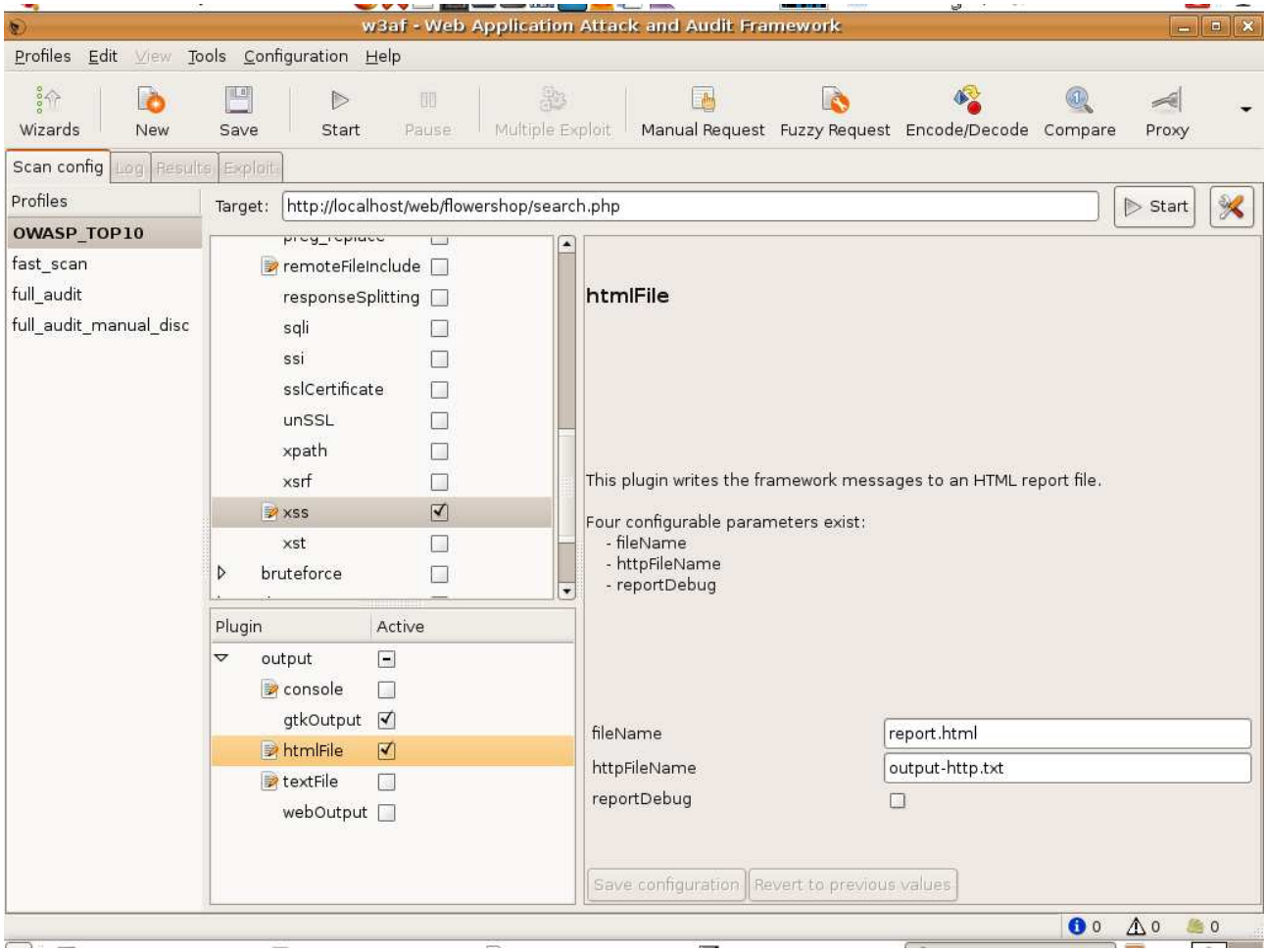


Ha nem történt hiba, akkor ennek a képnek kell fogadnia minket. Létrehozhatunk profilt, amely lényegében a részünkről előnyben részesített *pluginek* gyűjteményeként tekinthető. Lehetőségünk van egyénileg kialakított és *"fuzzy kéréseket"* létrehozni, utóbbiak segítségével megvizsgálhatjuk, hogy egy alkalmazás hogyan reagál nagy mennyiségű véletlenszerűen generált bemenetre. Rendelkezésünkre állnak beépített, a kiküldött adatok kódolására, összezavarására alkalmas algoritmusok, amelyekkel első sorban webalkalmazás-tűzfalak hatékonyságát vizsgálhatjuk. Van beépített profil is, mely valamilyen szempont alapján tartalmazza a kiválasztott plugineket, így csak a címet kell megadni és már szkennelhetünk is. A Pluginek alatt pedig megadhatjuk, hogy az eredményt milyen formában szeretnénk menteni: konzolra, grafikusan, HTML formátumban, vagy egyszerű szövegként.

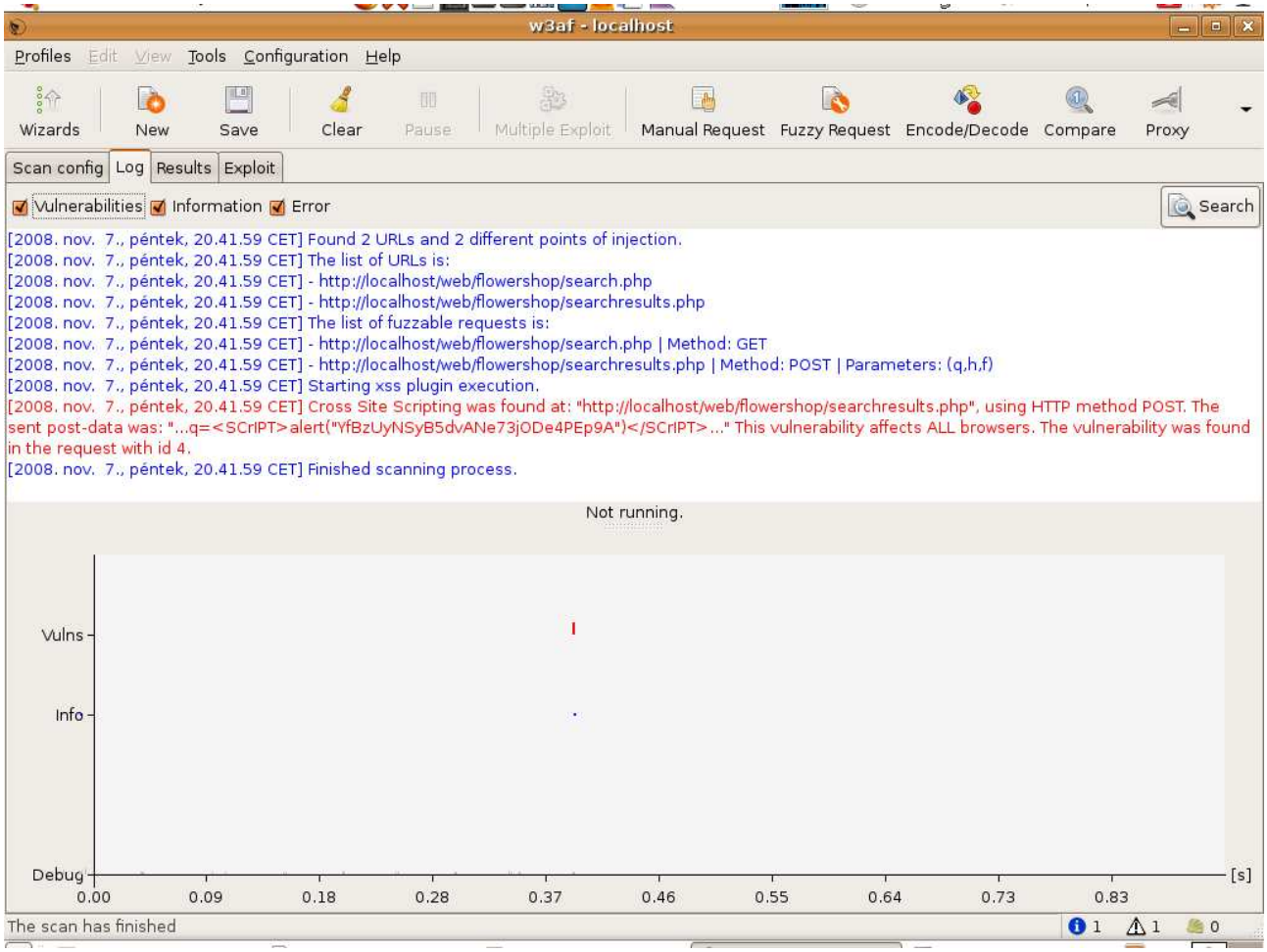
Nézzünk egy egyszerű példát, a Cross-Site Scripting-et!



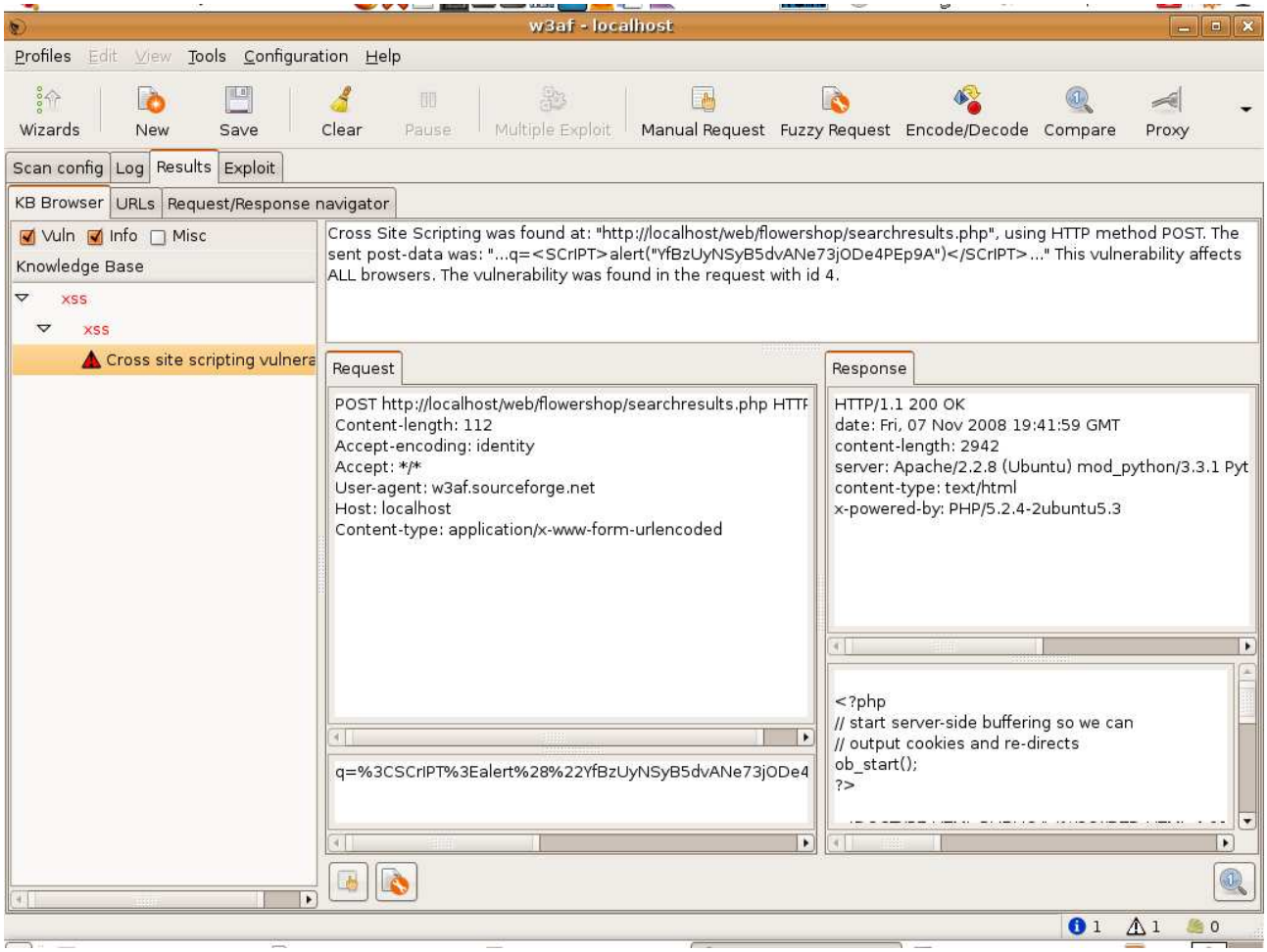
Az *audit pluginek* közül kiválasztjuk az *XSS*-t (azaz Cross-Site Scriptinget), amit be is kell állítanunk. Azt, hogy melyik *plugint* kell beállítani, azt az előtte levő papiron ceruzát ábrázoló kis ikon jelzi. Ha kijelölünk egy *plugint*, akkor láthatunk egy rövid leírás magáról a kiegészítésről és a beállításról (ha van neki).



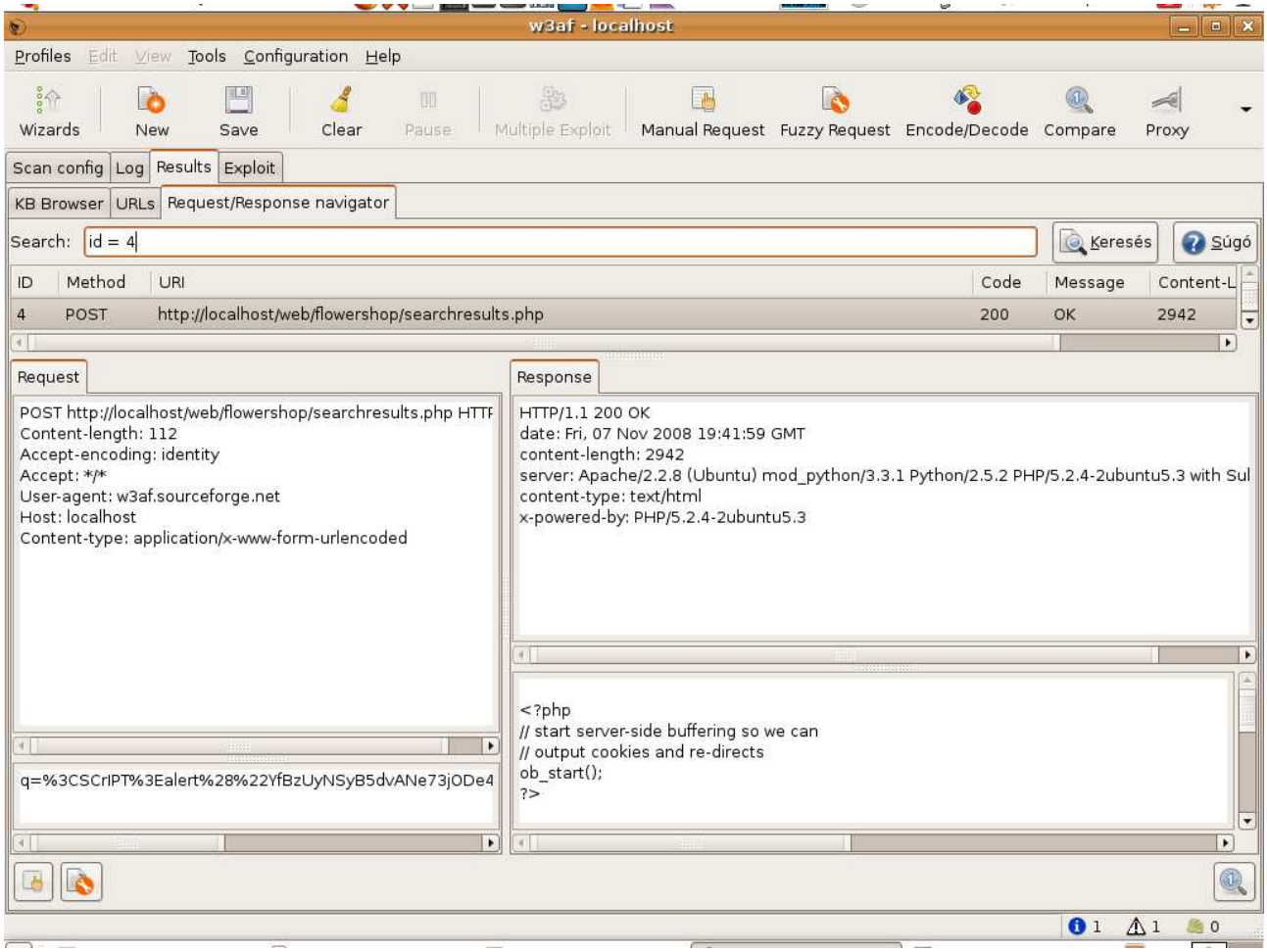
Állítsuk be a kimenetet is: a gtkOutput kimenet grafikus felületnél mindig adott. Ebben az esetben kiválasztjuk a htmlFile kimenetet is, ahol megadhatjuk a HTML fájl és a kimenet nevét. Ha megadtuk a Target mezőben a vizsgálni kívánt oldal címét, akkor el is kezdhetjük a szkennelést a Start gombra kattintva.



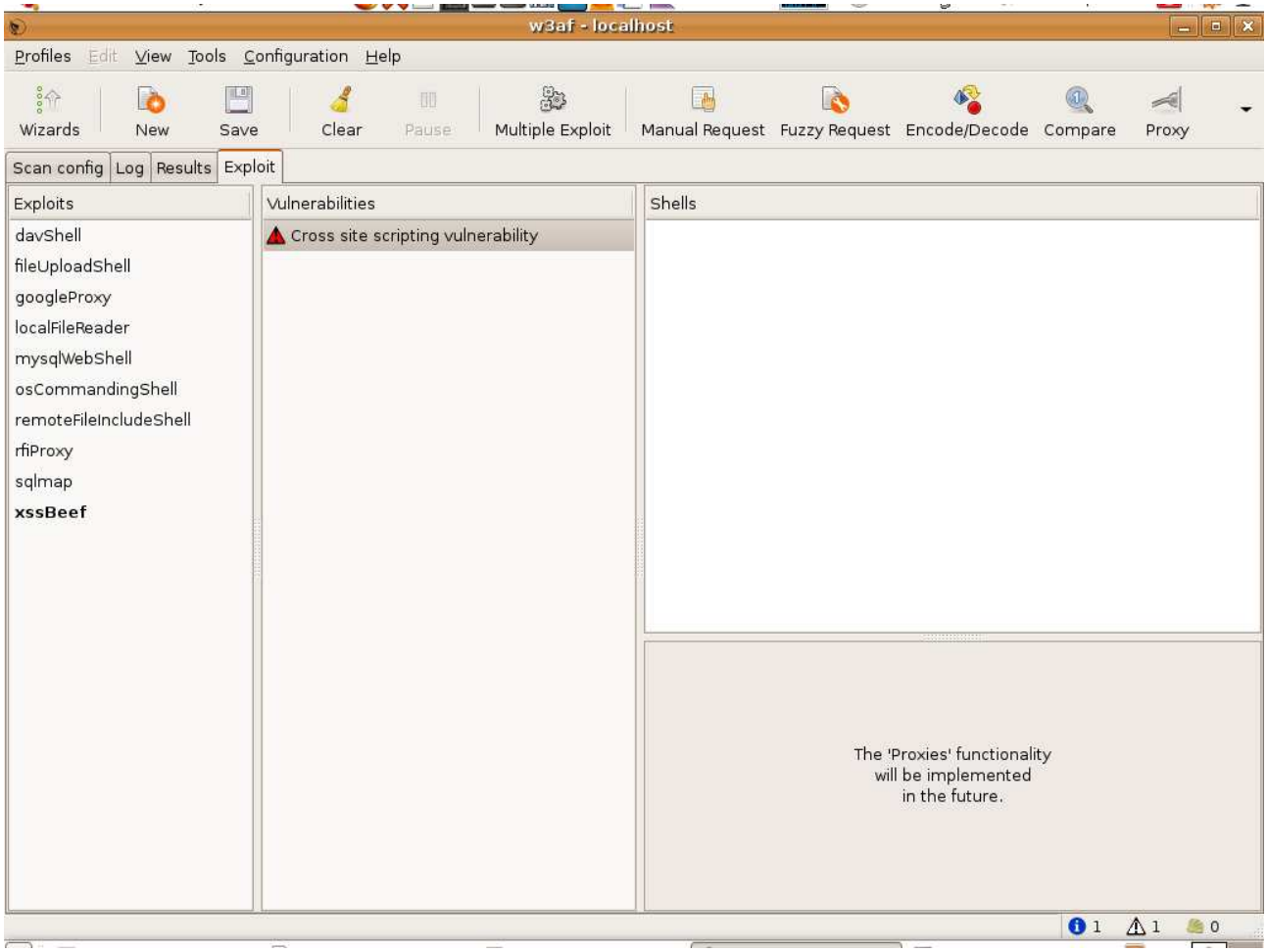
Ahogy elindítottuk a vizsgálatot, a Log ablakban láthatjuk a program által elvégzett műveleteket, illetve azok eredményét. Kiválaszthatjuk, hogy mit akarunk látni: sebezhetőséget, információt vagy a hibákat, esetleg mindhármat. Van keresésre is lehetőség a Search gombra kattintva. Ha a program végzett a felderítéssel, a Results, KB Browser ablakban csak a megtalált sebezhetőségeket láthatjuk kategorizálva, grafikusan is ábrázolva.



Itt, ha kiválasztjuk az egyes sebezhetőségeket (ebben az esetben csak egy van), és láthatjuk, hogy a rendszer milyen HTTP kérést küldött a szervernek és mi volt a válasz. Látható még a forráskód és a paraméterben átadott érték. Szűrhetjük, hogy mit szeretnénk látni: a sebezhetőségeket, az információkat, az általános dolgokat vagy mindhármat.

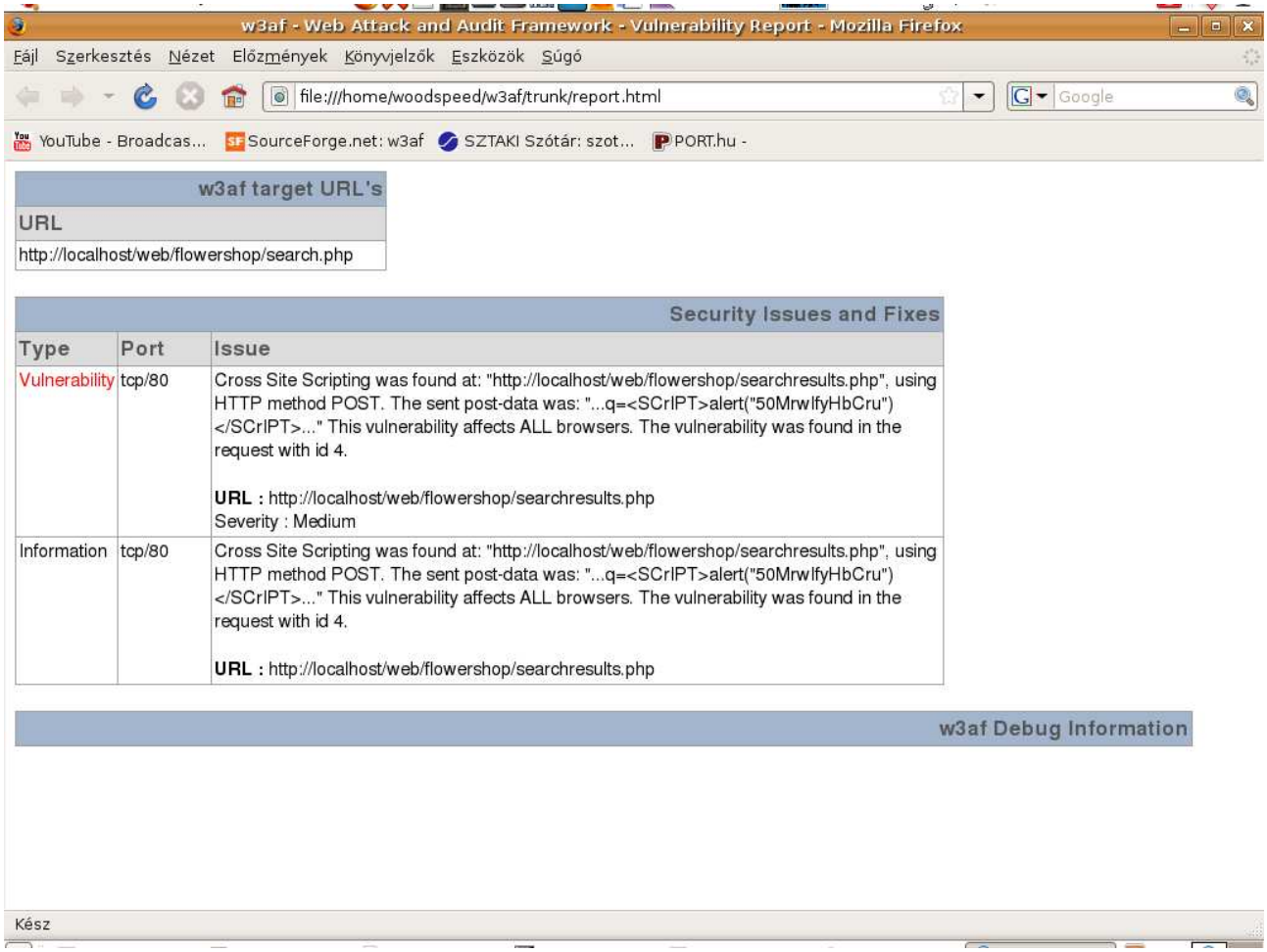


A *Response/Request navigator* ablakban kilistázhatjuk azokat a sebezhetőségeket, amikre kíváncsiak vagyunk. Itt további olyan információk jelennek meg, mint a HTTP kód, metódus, hossz, vagyis a teljes HTTP fejléc.



A felfedezett sebezhetőségek kihasználásához *exploitokat* készíthetünk. Ezek a szkriptek pl. hátsó kaput nyithatnak a célponton, letölthetnek érzékeny adatokat tartalmazó fájlokat, vagy létrehozhatnak új bejegyzéseket a célpont adatbázisában illetve fájlrendszerén, tehát lényegében bizonyos sebezhetőség-típusokon keresztül gyakran végrehajtható akciókat írhatnak le. Ha rendelkezünk a sebezhetőséghez *exploittal*, akkor azt az Exploit ablakban fel is használhatjuk. Azt, hogy rendelkezünk e megfelelő *exploittal*, a feketén kiemelt *exploit* neve jelzi. Az exploitokat konfigurálni kell, utána csak ráhúzzuk a középen lévő sebezhetőségre és automatikusan végrehajtódnak.

A w3af képes együttműködni a népszerű Metasploit keretrendszer 3.0-ás vagy újabb verzióival.



A felderítés végeztével eredményeinket az Output pluginek segítségével összefoglalva elmenthetjük. A fenti ábrán az előzőekben végrehajtott felderítés HTML kimenete látható.

Linkek

http://www.net-security.org/secworld.php?id=6501
http://www.owasp.org/images/e/e8/OWASP_Top_10_2007.pdf
http://www.owasp.org/index.php/Category:OWASP_Guide_Project
http://w3af.sourceforge.net/
http://w3af.sourceforge.net/documentation/user/w3afUsersGuide.pdf
http://nukeit.org/2008/01/17/howto-install-w3af-on-windows-svn-style/
http://nukeit.org/2008/07/26/w3af-updated-prerequisites-win32-howto/
http://www.metasploit.org