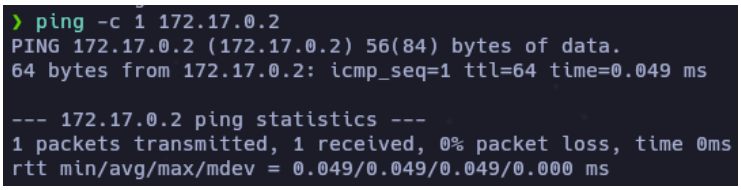
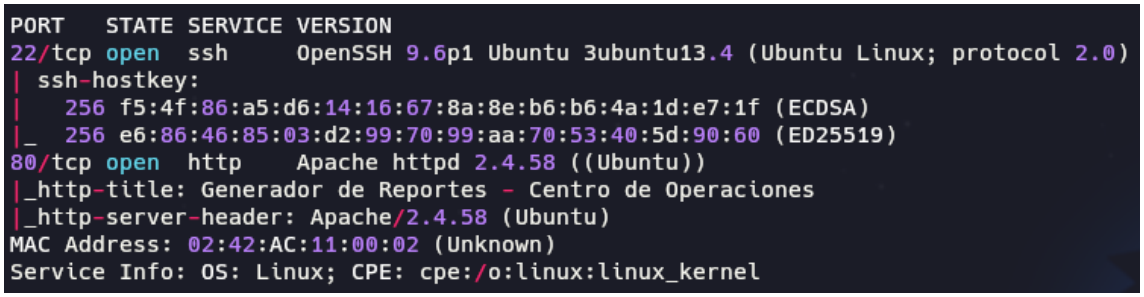
Desplegamos la máquina.

Le aplicamos un ping para comprobar la conectividad y además con el ttl de 64 sabemos que estamos ante una máquina Linux.



Con nmap vemos los puertos que están abiertos y además sus servicios.

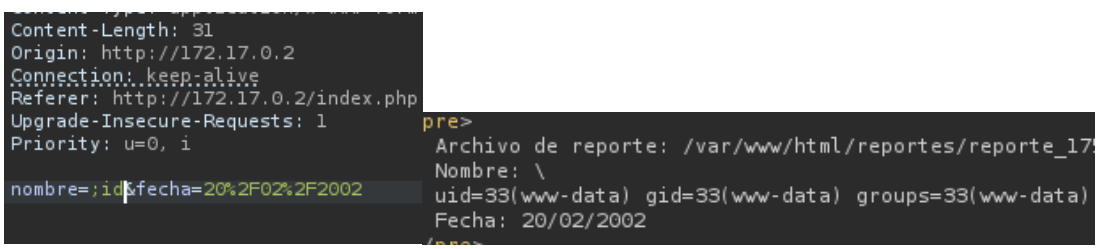


Con gobuster vemos directorios y archivos que están ocultos.

```
❯ gobuster dir -u http://172.17.0.2 -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
 -x php,html,htm,xml,json,css,md,txt,log,conf,ini,js,ts,sh,bak,old,backup,zip,tar,tar.gz,rar,7z,png
,jpg,jpeg,gif,svg,webp,woff,woff2,ttf,eot,exe,bin,py,pl,rb,asp,aspx,pcap,pcapng -t 100
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://172.17.0.2
[+] Method:                  GET
[+] Threads:                 100
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Extensions:              old,exe,py,pl,php,xml,zip,jpg,ttf,md,gif,woff2,eot,aspx,pcapng,png,jpe
g,htm,tar,tar.gz,pcap,ini,sh,backup,webp,rb,ts,css,txt,js,svg,bin,asp,rar,7z,conf,html,json,log,bak
,woff
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.php                 (Status: 403) [Size: 275]
/.htm                 (Status: 403) [Size: 275]
/.html                (Status: 403) [Size: 275]
/index.php            (Status: 200) [Size: 2832]
/scripts.js           (Status: 200) [Size: 1090]
/upload.php           (Status: 200) [Size: 33]
/upload.html          (Status: 200) [Size: 2314]
/upload.js            (Status: 200) [Size: 1645]
/old                  (Status: 301) [Size: 306] [--> http://172.17.0.2/old/]
/styles.css           (Status: 200) [Size: 2693]
```

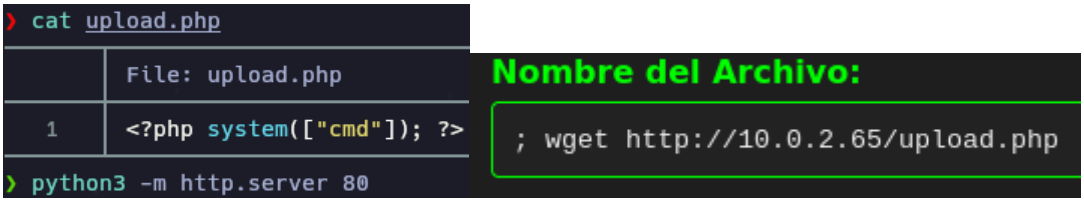
Vemos que hay un generador de reportes por lo que vamos a interceptarlo con burpsuite.

Vemos que podemos injectar código directamente con ‘;’ previamente.



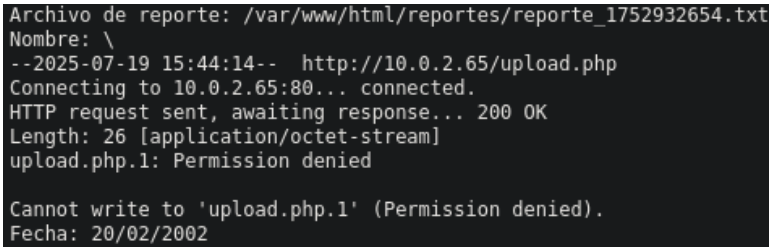
Vamos a comprobar que la máquina tenga el binario wget y de ese modo tratar de subir un archivo malicioso .php para poder hacernos una reverse shell.

Nos generamos un archivo que injectemos código con el parámetro cmd.

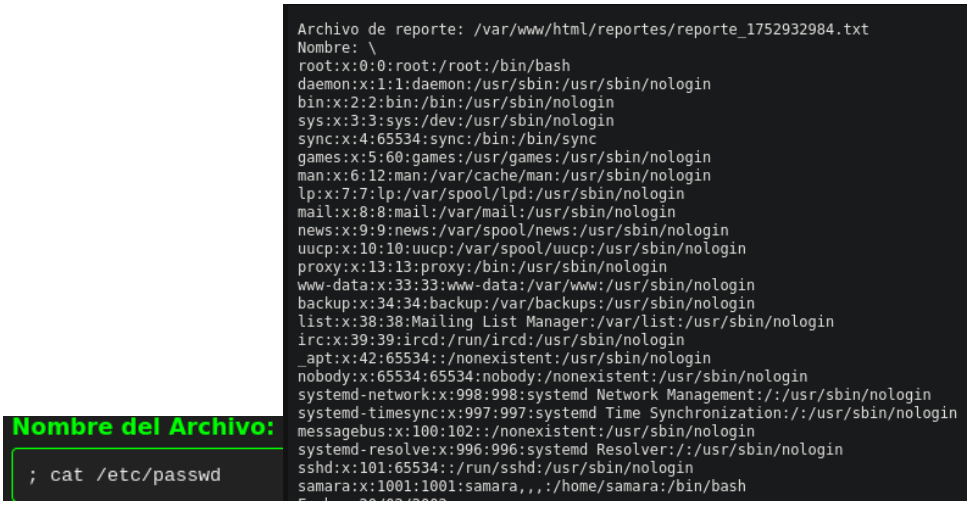


Una vez que comprobamos que se ha subido, previamente levantando un servidor python3.

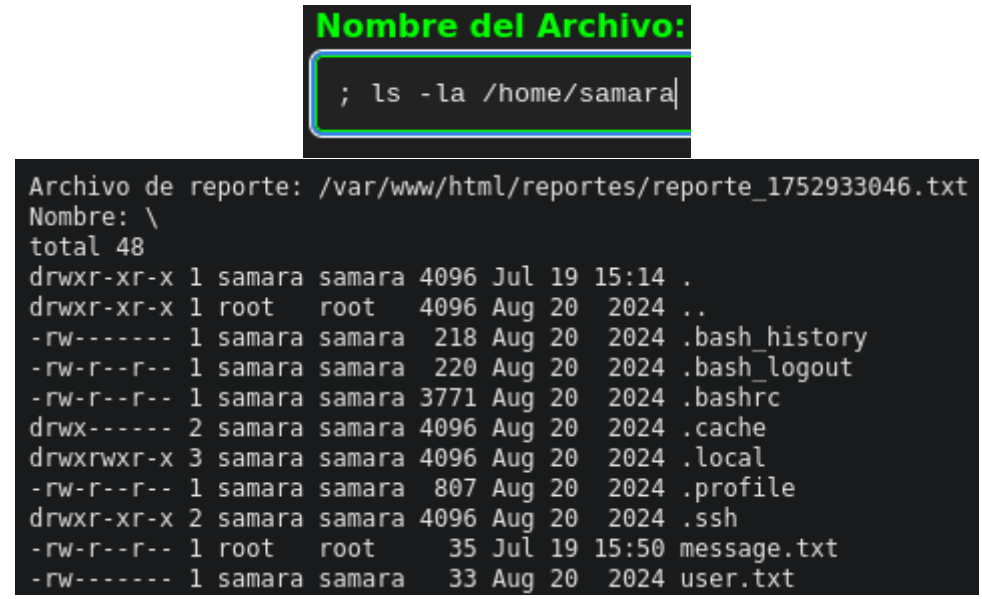
Vemos que nos deniega la opción de escribirlo.



Pero como podemos ejecutar código, vamos a ver los usuarios que existen.

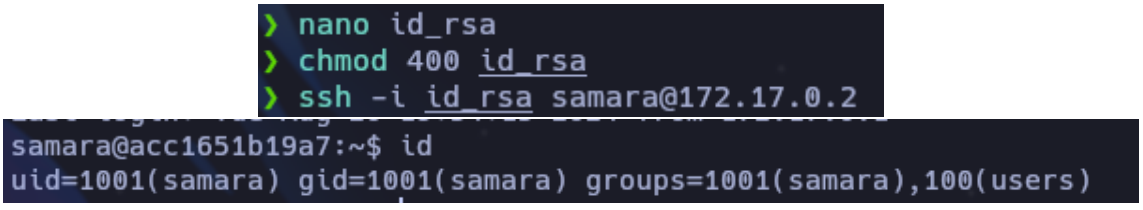


Dentro del usuario samara podemos ver que tiene una carpeta .ssh.

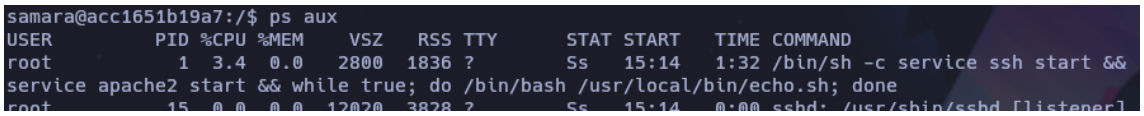


Comprobamos que tiene un id_rsa por lo que podremos conectarnos mediante ssh una vez lo copiemos en nuestra Kali sin necesidad de contraseña.

Lo creamos y le damos los permisos adecuados y nos conectamos como el usuario samara.

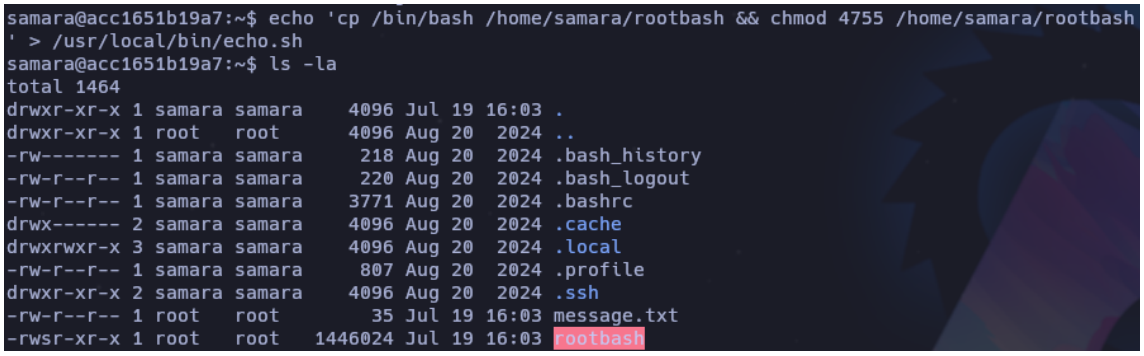


Una vez dentro vemos vemos el estado de los procesos y vemos un archivo .sh que se ejecuta como root.



Lo revisamos y vemos que podemos escribir en dicho archivo, por lo que lo modificaremos.

Vamos a crear una copia de la bash y le vamos a dar permisos SUID de modo que vamos a tener una bash con permisos SUID en el directorio de samara.



Listo ya solo debemos ejecutar la copia de la bash que nos hemos hecho con la flag -p para mantener los permisos de root.