MY FIRST CYBERSECURITY CAPTURE THE FLAG(CTF) ACTIVITY AT MASTERSCHOOL

The CTF challenge covers the following aspects:

1. User and File Management

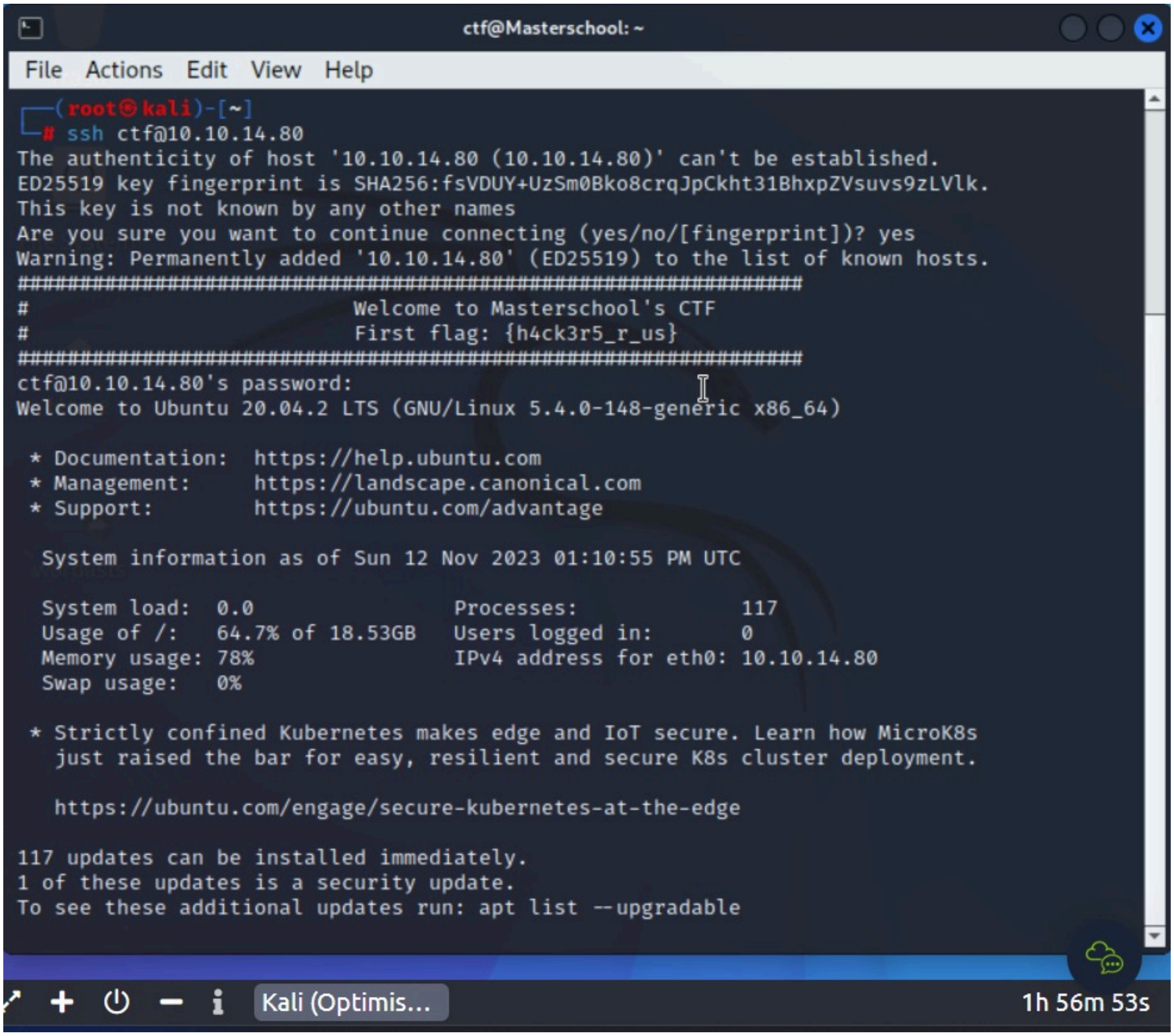
▼ **Task Details:**

- **User Creation:** Start by creating a new user in the Linux system. Remember to assign the user a password for subsequent login purposes.
- **User Switch:** Once the new user has been created, switch your session to that user. This step will involve using command-line authentication.
- **Folder and File Creation:** As the new user, create a new directory. Create a file and write a simple message inside this directory. Your message could be something like "Hello from [your username]!". Remember to save the file before proceeding.
- **Switch Back to Original User:** After successfully writing the message, switch back to your original user session, which for this exercise, is 'ctf'.

2. File System Flags
3. Webpage Flags
4. Hidden Flags Challenge
5. Hash Cracking

STEP 1:

Connecting to the target machine using Secure Shell (SSH) with the syntax `ssh user@targetip`.



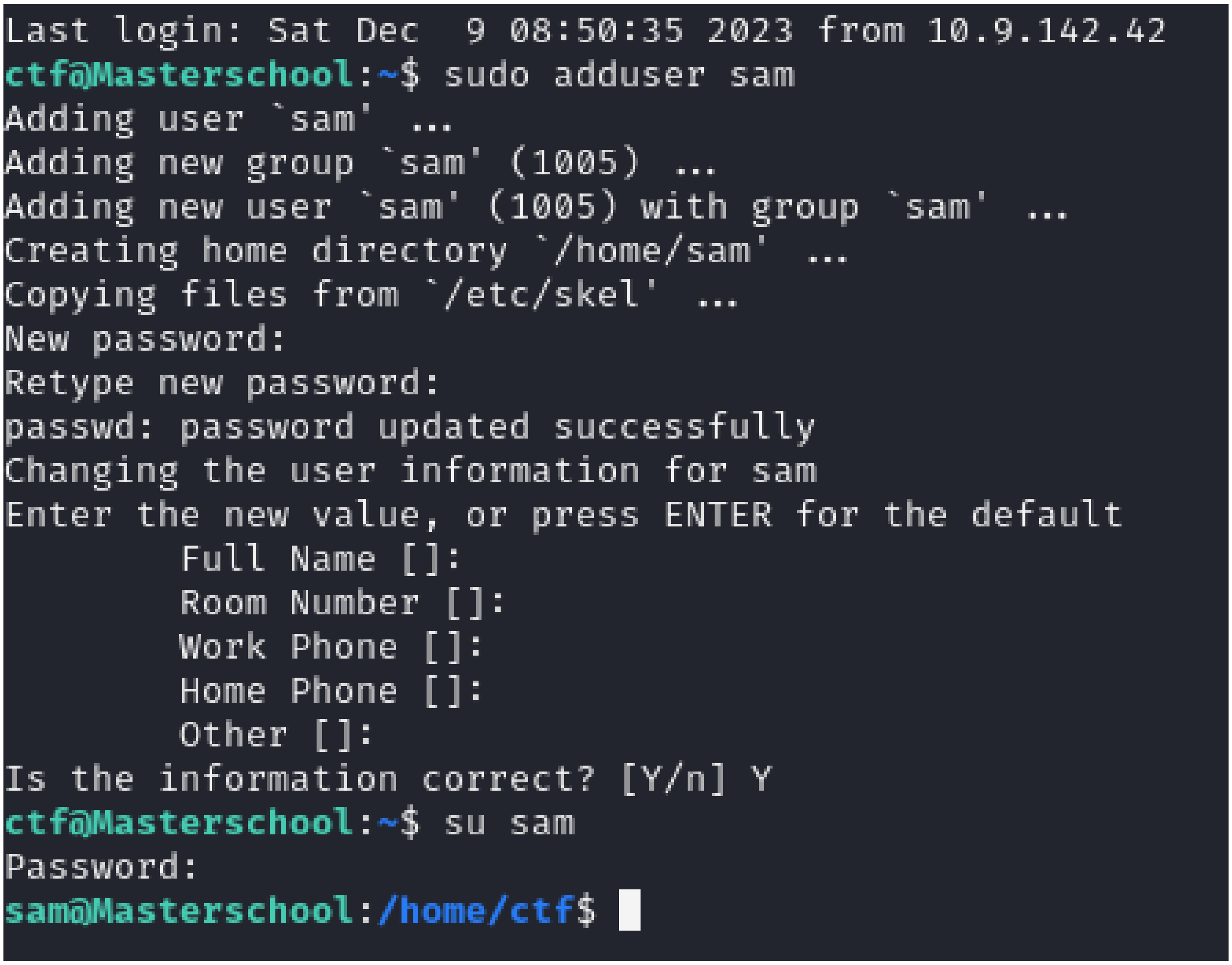
```
┌──(root㉿kali)-[~]
└─# ssh ctf@10.10.14.80
The authenticity of host '10.10.14.80 (10.10.14.80)' can't be established.
ED25519 key fingerprint is SHA256:fsVDUY+UzSm0Bko8crqJpCkht31BhxpZVsuvs9zLVlk.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.14.80' (ED25519) to the list of known hosts.
###############################################################
#                          Welcome to Masterschool's CTF
#                          First flag: {h4ck3r5_r_us}
###############################################################
ctf@10.10.14.80's password:
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-148-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Sun 12 Nov 2023 01:10:55 PM UTC

  System load:  0.0               Processes:             117
  Usage of /:   64.7% of 18.53GB   Users logged in:       0
  Memory usage: 78%                IPv4 address for eth0: 10.10.14.80
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

117 updates can be installed immediately.
1 of these updates is a security update.
To see these additional updates run: apt list --upgradable
```

When i connected to the machine i was welcomed with the **First Flag {h4ck3r5_r_us}.**

STEP 2: User creation & User Switch

This stage encompasses different tasks and for each a different syntax.

syntax `sudo adduser sam` & `su sam`

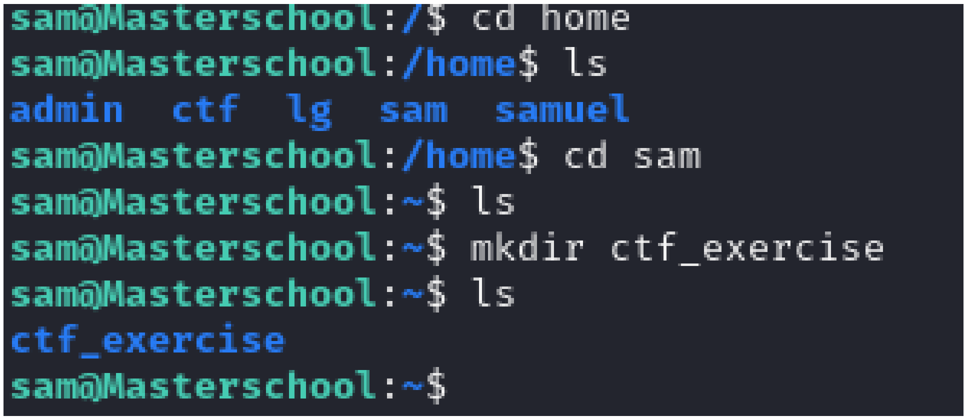


```
Last login: Sat Dec  9 08:50:35 2023 from 10.9.142.42
ctf@Masterschool:~$ sudo adduser sam
Adding user `sam' ...
Adding new group `sam' (1005) ...
Adding new user `sam' (1005) with group `sam' ...
Creating home directory `/home/sam' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for sam
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
ctf@Masterschool:~$ su sam
Password:
sam@Masterschool:/home/ctf$ 
```

From the image above we were able to create the user “Sam” and also switched to the user.

STEP 3: Folder creation

syntax `mkdir ctf_exercise`

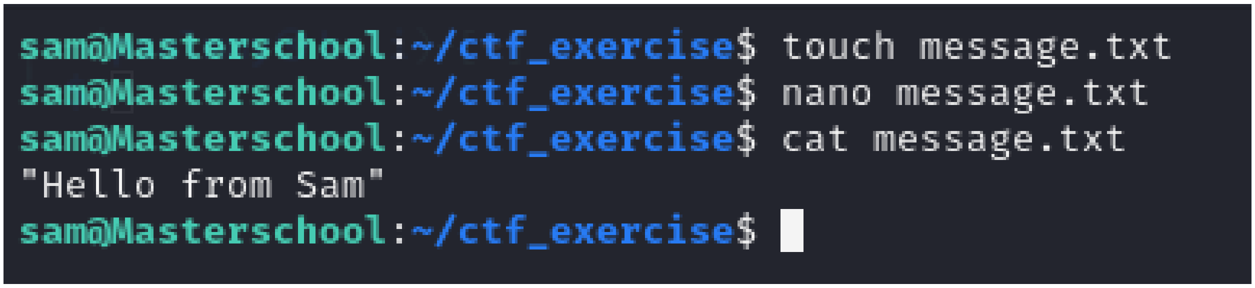


```
sam@Masterschool:/$ cd home
sam@Masterschool:/home$ ls
admin  ctf  lg  sam  samuel
sam@Masterschool:/home$ cd sam
sam@Masterschool:~$ ls
sam@Masterschool:~$ mkdir ctf_exercise
sam@Masterschool:~$ ls
ctf_exercise
sam@Masterschool:~$
```

from the image in Step 2, we created the user SAM, but the user was still in the directory of the user CTF where i had no permission to create a file in. so i navigated to the home directory using `cd home` then listed the items in the home directory with `ls` and navigated into Sam directory with `cd sam` after which i used `mkdir ctf_exercise` this created a folder called ctf_exercise.

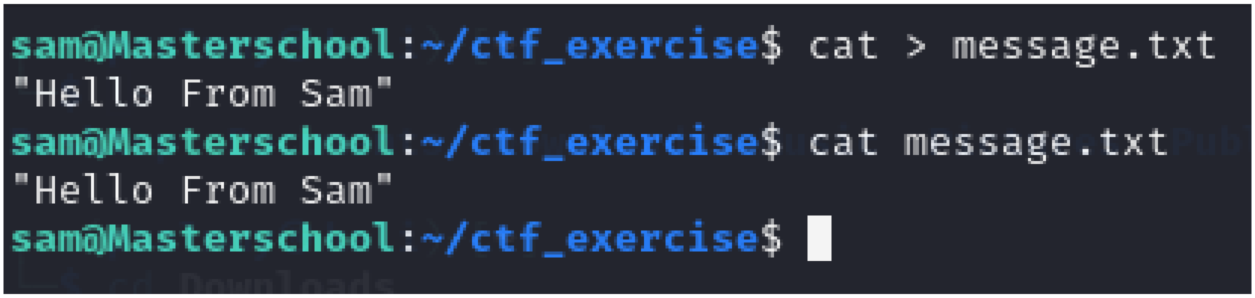
STEP 4: File Creation

Syntax: `touch message.txt` , `nano message.txt` , `cat > message.txt`



```
sam@Masterschool:~/ctf_exercise$ touch message.txt
sam@Masterschool:~/ctf_exercise$ nano message.txt
sam@Masterschool:~/ctf_exercise$ cat message.txt
"Hello from Sam"
sam@Masterschool:~/ctf_exercise$ 
```

i used the command `touch message.txt` to create a file called “message.txt” after which i used the `nano message.txt` command to write/edit my message “Hello from Sam” into the created file. there is another method to this which is by using the `cat > message.txt` command to create the file and also append the text into the file.

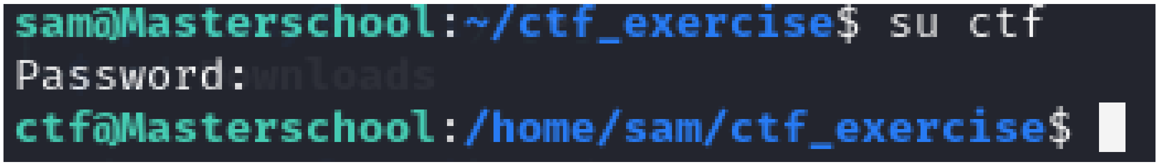


```
sam@Masterschool:~/ctf_exercise$ cat > message.txt
"Hello From Sam"
sam@Masterschool:~/ctf_exercise$ cat message.txt
"Hello From Sam"
sam@Masterschool:~/ctf_exercise$ 
```

This tasks made me understand the usage of nano and also how to use the cat command adequately.

STEP 5: Switch back to Original User

syntax: `su ctf`



```
sam@Masterschool:~/ctf_exercise$ su ctf
Password:
ctf@Masterschool:/home/sam/ctf_exercise$ 
```

STEP 6: File System Flags

This is the second challenge in the exercise and we were told that there are four(4) file system flags and one of them is in a file called “find_flag.txt”. this is actually a good hint and prompted me to use the find command `find / -name "find_flag.txt" -type f 2>/dev/null`

let me break this command down;

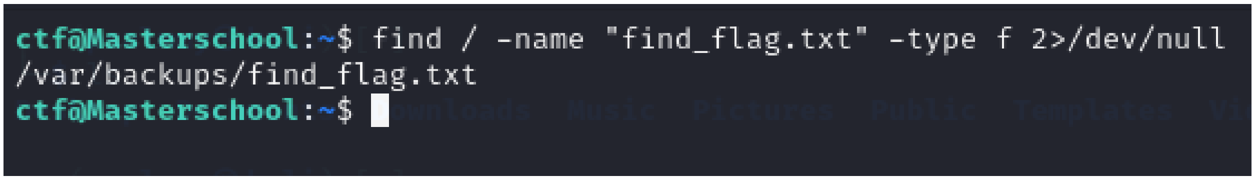
i. find: The command-line utility used for searching files and directories in a Unix-like operating system.

ii. /: Specifies the starting point for the search. it is the root directory, which is the top-level directory in the file system.

iii. -name "find_flag.txt": This option specifies the name of the file to search for. In this case, the file name is "find_flag.txt".

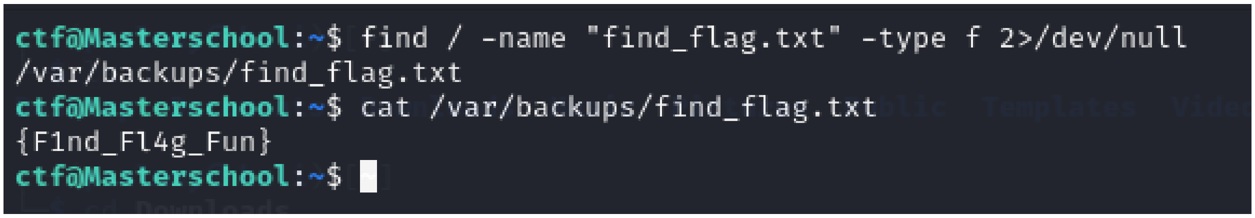
iv. -type f: This option restricts the search to only files. The argument "f" stands for file.

v. 2>/dev/null: This part of the command is used to redirect error messages to /dev/null, which essentially discards them. The number "2" refers to the standard error (stderr), and /dev/null is a special file that represents nothingness. By doing this, any error messages produced during the search (such as permission denied messages for directories the user doesn't have access to) will not be displayed on the terminal.



```
ctf@Masterschool:~$ find / -name "find_flag.txt" -type f 2>/dev/null
/var/backups/find_flag.txt
ctf@Masterschool:~$ 
```

the find command returned the file path to us so we would use the `cat` command to open the file.

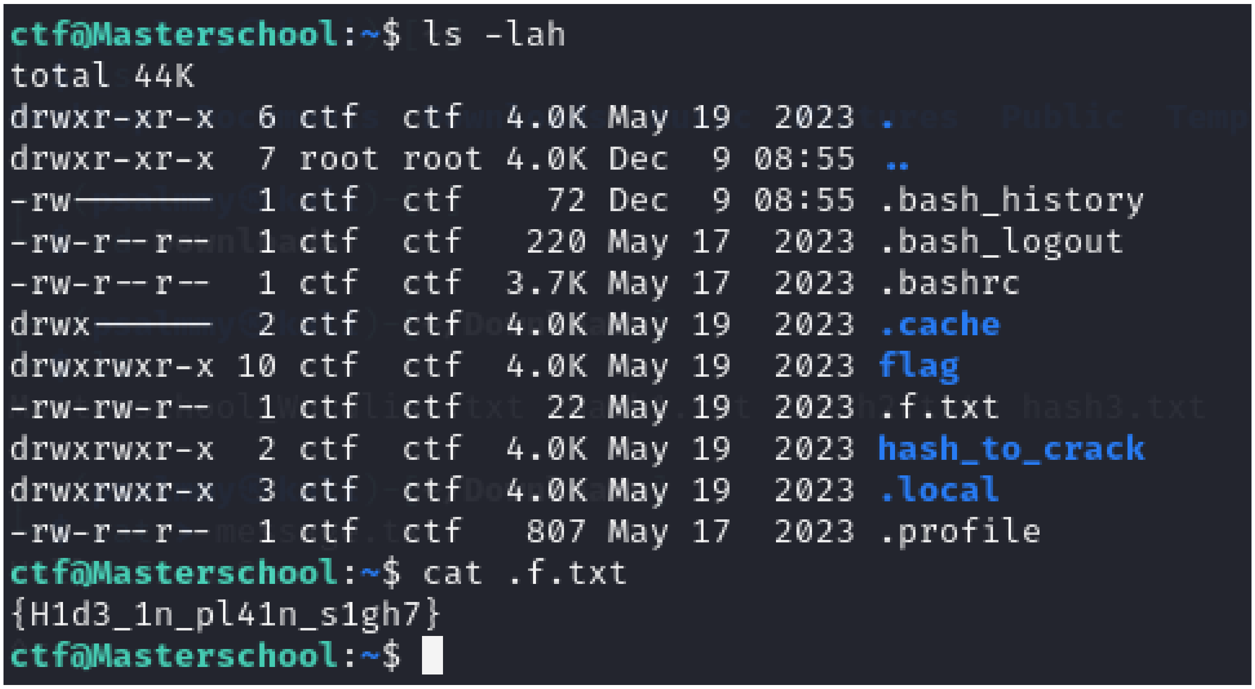


```
ctf@Masterschool:~$ find / -name "find_flag.txt" -type f 2>/dev/null
/var/backups/find_flag.txt
ctf@Masterschool:~$ cat /var/backups/find_flag.txt
{F1nd_Fl4g_Fun}
ctf@Masterschool:~$ 
```

There it is our first file system flag is {F1nd_FL4g_Fun}.

Second File System Flag:

i listed all folders and files in ctf directory using `ls -lah` and voila i found a hidden file .f.txt then i used the command `cat.f.txt` , i found another flag {H1d3_1n_pl41n_s1gh7}

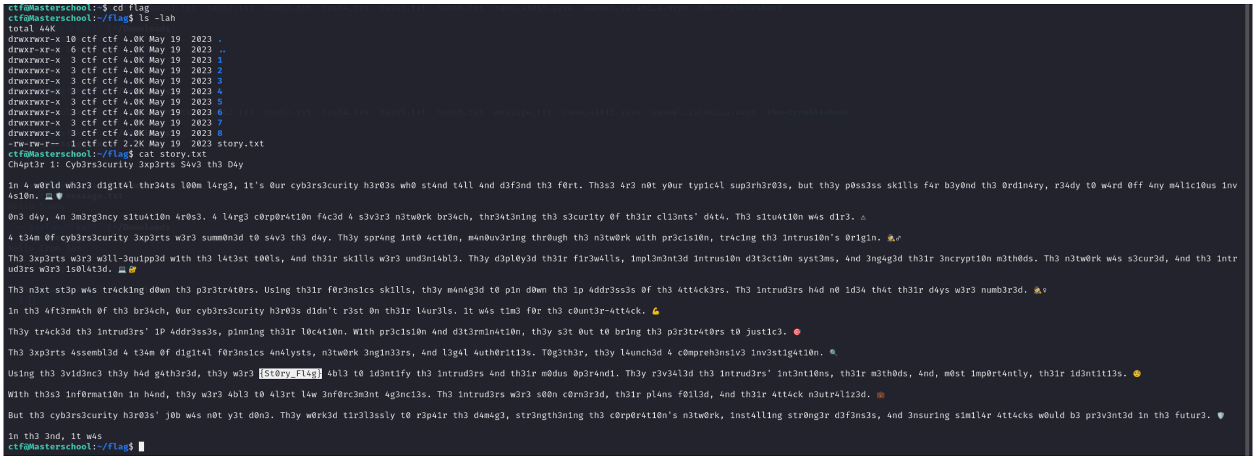


```
ctf@Masterschool:~$ ls -lah
total 44K
drwxr-xr-x  6 ctf  ctf  4.0K May 19  2023 .
drwxr-xr-x  7 root root 4.0K Dec  9 08:55 ..
-rw-------  1 ctf  ctf    72 Dec  9 08:55 .bash_history
-rw-r--r--  1 ctf  ctf   220 May 17  2023 .bash_logout
-rw-r--r--  1 ctf  ctf  3.7K May 17  2023 .bashrc
drwx------  2 ctf  ctf  4.0K May 19  2023 .cache
drwxrwxr-x 10 ctf  ctf  4.0K May 19  2023 flag
-rw-rw-r--  1 ctf  ctf    22 May 19  2023 .f.txt
drwxrwxr-x  2 ctf  ctf  4.0K May 19  2023 hash_to_crack
drwxrwxr-x  3 ctf  ctf  4.0K May 19  2023 .local
-rw-r--r--  1 ctf  ctf   807 May 17  2023 .profile
ctf@Masterschool:~$ cat .f.txt
{H1d3_1n_pl41n_s1gh7}
ctf@Masterschool:~$ 
```

Third File System Flag :

i navigated into another folder “flag” using `cd flag` then listed all items in the folder with `ls -lah`

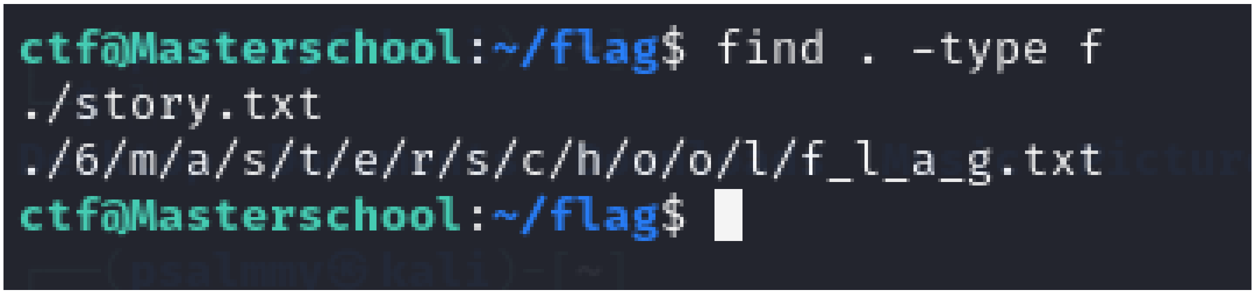
i found an interesting file “story.txt”. i opened it with `cat story.txt` and there was an interesting story of how cybersecurity experts saved the day. whilst reading this, i found another flag {St0ry_FL4g}.



```
ctf@Masterschool:~$ cd flag
ctf@Masterschool:~/flag$ ls -lah
total 44K
drwxrwxr-x 10 ctf ctf 4.0K May 19  2023 .
drwxr-xr-x  6 ctf ctf 4.0K May 19  2023 ..
drwxrwxr-x  3 ctf ctf 4.0K May 19  2023 1
drwxrwxr-x  3 ctf ctf 4.0K May 19  2023 2
drwxrwxr-x  3 ctf ctf 4.0K May 19  2023 3
drwxrwxr-x  3 ctf ctf 4.0K May 19  2023 4
drwxrwxr-x  3 ctf ctf 4.0K May 19  2023 5
drwxrwxr-x  3 ctf ctf 4.0K May 19  2023 6
drwxrwxr-x  3 ctf ctf 4.0K May 19  2023 7
drwxrwxr-x  3 ctf ctf 4.0K May 19  2023 8
-rw-rw-r--  1 ctf ctf 2.2K May 19  2023 story.txt
ctf@Masterschool:~/flag$ cat story.txt
Ch4pt3r 1: Cyb3rs3curity 3xp3rts S4v3 th3 D4y

1n 4 w0rld wh3r3 d1g1t4l thr34ts l00m l4rg3, 1t's 0ur cyb3rs3curity h3r03s wh0 st4nd t4ll 4nd d3f3nd th3 f0rt. Th3s3 4r3 n0t y0ur typ1c4l sup3rh3r03s, but th3y p0ss3ss sk1lls f4r b3y0nd th3 0rd1n4ry, r34dy t0 w4rd 0ff 4ny m4l1c10us 1nv
4s10n. 💻🛡️

0n3 d4y, 4n 3m3rg3ncy s1tu4t10n 4r0s3. 4 l4rg3 c0rp0r4t10n f4c3d 4 s3v3r3 n3tw0rk br34ch, thr34t3n1ng th3 s3cur1ty 0f th31r cl13nts' d4t4. Th3 s1tu4t10n w4s d1r3. ⚠

4 t34m 0f cyb3rs3curity 3xp3rts w3r3 summ0n3d t0 s4v3 th3 d4y. Th3y spr4ng 1nt0 4ct10n, m4n0uv3r1ng thr0ugh th3 n3tw0rk w1th pr3c1s10n, tr4c1ng th3 1ntrus10n's 0r1g1n. 🕵️‍♂️

Th3 3xp3rts w3r3 w3ll-3qu1pp3d w1th th3 l4t3st t00ls, 4nd th31r sk1lls w3r3 und3n14bl3. Th3y d3pl0y3d th31r f1r3w4lls, 1mpl3m3nt3d 1ntrus10n d3t3ct10n syst3ms, 4nd 3ng4g3d th31r 3ncrypt10n m3th0ds. Th3 n3tw0rk w4s s3cur3d, 4nd th3 1ntr
ud3rs w3r3 1s0l4t3d. 💻🔐

Th3 n3xt st3p w4s tr4ck1ng d0wn th3 p3r3tr4t0rs. Us1ng th31r f0r3ns1cs sk1lls, th3y m4n4g3d t0 p1n d0wn th3 1p 4ddr3ss3s 0f th3 4tt4ck3rs. Th3 1ntrud3rs h4d n0 1d34 th4t th31r d4ys w3r3 numb3r3d. 🕵️‍♀️

1n th3 4ft3rm4th 0f th3 br34ch, 0ur cyb3rs3curity h3r03s d1dn't r3st 0n th31r l4ur3ls. 1t w4s t1m3 f0r th3 c0unt3r-4tt4ck. 💪

Th3y tr4ck3d th3 1ntrud3rs' 1P 4ddr3ss3s, p1nn1ng th31r l0c4t10n. W1th pr3c1s10n 4nd d3t3rm1n4t10n, th3y s3t 0ut t0 br1ng th3 p3r3tr4t0rs t0 just1c3. 🎯

Th3 3xp3rts 4ss3mbl3d 4 t34m 0f d1g1t4l f0r3ns1cs 4n4lysts, n3tw0rk 3ng1n33rs, 4nd l3g4l 4uth0r1t13s. T0g3th3r, th3y l4unch3d 4 c0mpreh3ns1v3 1nv3st1g4t10n. 🔍

Us1ng th3 3v1d3nc3 th3y h4d g4th3r3d, th3y w3r3 {St0ry_Fl4g} 4bl3 t0 1d3nt1fy th3 1ntrud3rs 4nd th31r m0dus 0p3r4nd1. Th3y r3v34l3d th3 1ntrud3rs' 1nt3nt10ns, th31r m3th0ds, 4nd, m0st 1mp0rt4ntly, th31r 1d3nt1t13s. 🧐

W1th th3s3 1nf0rmat10n 1n h4nd, th3y w3r3 4bl3 t0 4l3rt l4w 3nf0rc3m3nt 4g3nc13s. Th3 1ntrud3rs w3r3 s00n c0rn3r3d, th31r pl4ns f01l3d, 4nd th31r 4tt4ck n3utr4l1z3d. 💼

But th3 cyb3rs3curity h3r03s' j0b w4s n0t y3t d0n3. Th3y w0rk3d t1r3l3ssly t0 r3p41r th3 d4m4g3, str3ngth3n1ng th3 c0rp0r4t10n's n3tw0rk, 1nst4ll1ng str0ng3r d3f3ns3s, 4nd 3nsur1ng s1m1l4r 4tt4cks w0uld b3 pr3v3nt3d 1n th3 futur3. 🛡️

1n th3 3nd, 1t w4s
ctf@Masterschool:~/flag$ 
```

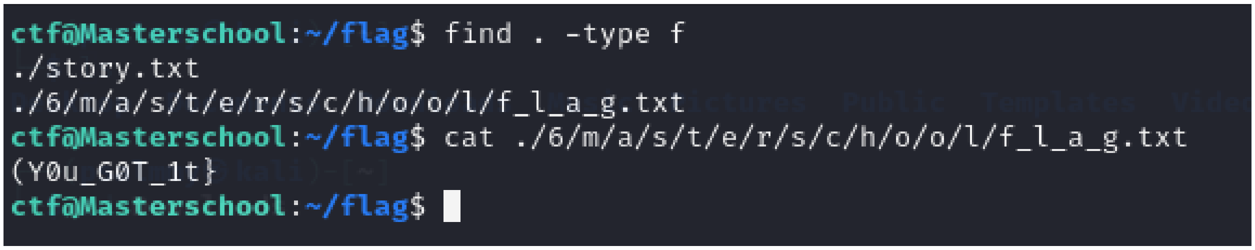
Fourth File System Flag:

this wasn't as easy as i expected, thanks to one of my colleagues Tom, he was the one who guided me on locating the last file system flag. it was just a simple find command to show all hidden files in the flag directory using `find . -type f` . this brought two results. the first which we had viewed earlier and the second which hasnt been viewed.



```
ctf@Masterschool:~/flag$ find . -type f
./story.txt
./6/m/a/s/t/e/r/s/c/h/o/o/l/f_l_a_g.txt
ctf@Masterschool:~/flag$
```

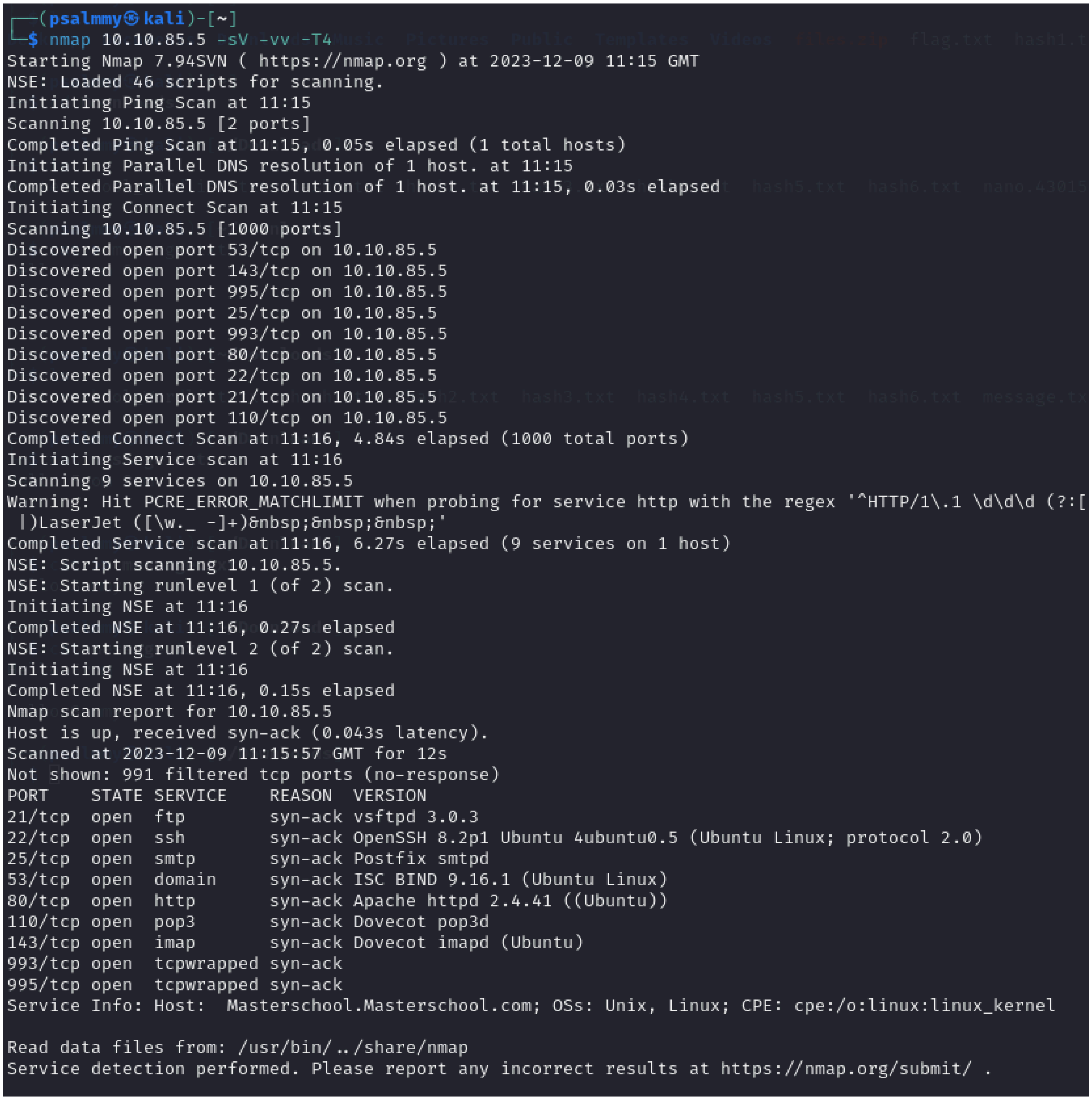
i opened the second using `cat ./6/m/a/s/t/e/r/s/c/h/o/o/l/f_l_a_g.txt` and i found the last flag which was (Y0u_G0T_1t} shown in the image below.



```
ctf@Masterschool:~/flag$ find . -type f
./story.txt
./6/m/a/s/t/e/r/s/c/h/o/o/l/f_l_a_g.txt
ctf@Masterschool:~/flag$ cat ./6/m/a/s/t/e/r/s/c/h/o/o/l/f_l_a_g.txt
(Y0u_G0T_1t}
ctf@Masterschool:~/flag$
```

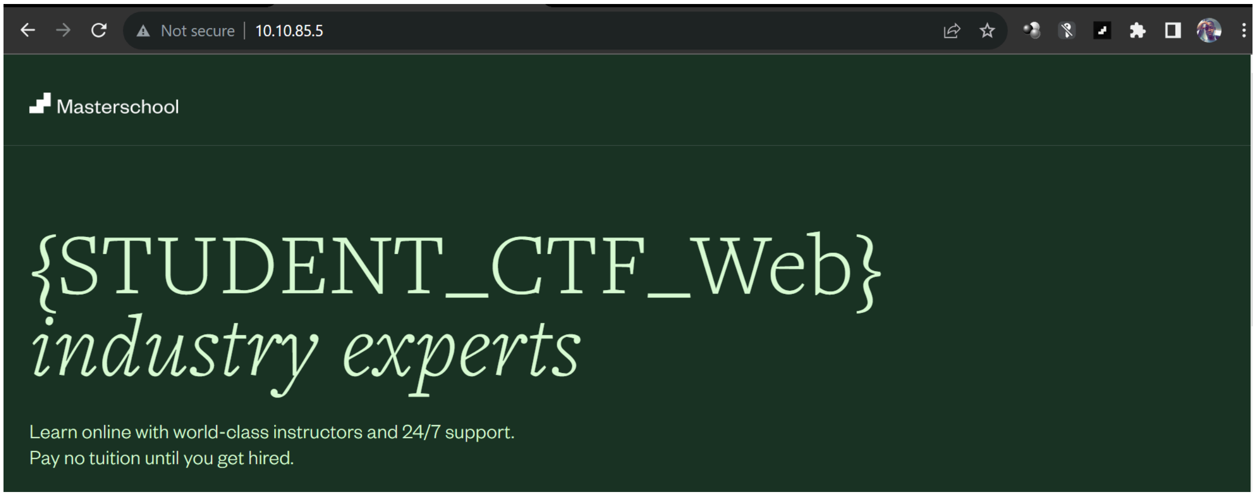
STEP 7: Webpage Flags

i was tasked to find flags hidden within webpages. what i did first was to scan the Ip address with the nmap which is a very powerful network scanning tool used to discover hosts and services on computer networks. the command used was `nmap targetipaddress -sV -vv -T4`



```
┌──(psalmmy㉿kali)-[~]
└─$ nmap 10.10.85.5 -sV -vv -T4
Starting Nmap 7.94SVN ( https://nmap.org ) at 2023-12-09 11:15 GMT
NSE: Loaded 46 scripts for scanning.
Initiating Ping Scan at 11:15
Scanning 10.10.85.5 [2 ports]
Completed Ping Scan at 11:15, 0.05s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 11:15
Completed Parallel DNS resolution of 1 host. at 11:15, 0.03s elapsed
Initiating Connect Scan at 11:15
Scanning 10.10.85.5 [1000 ports]
Discovered open port 53/tcp on 10.10.85.5
Discovered open port 143/tcp on 10.10.85.5
Discovered open port 995/tcp on 10.10.85.5
Discovered open port 25/tcp on 10.10.85.5
Discovered open port 993/tcp on 10.10.85.5
Discovered open port 80/tcp on 10.10.85.5
Discovered open port 22/tcp on 10.10.85.5
Discovered open port 21/tcp on 10.10.85.5
Discovered open port 110/tcp on 10.10.85.5
Completed Connect Scan at 11:16, 4.84s elapsed (1000 total ports)
Initiating Service scan at 11:16
Scanning 9 services on 10.10.85.5
Warning: Hit PCRE_ERROR_MATCHLIMIT when probing for service http with the regex '^HTTP/1\.1 \d\d\d (?:[
 |)LaserJet ([\w._ -]+)&nbsp;&nbsp;&nbsp;'
Completed Service scan at 11:16, 6.27s elapsed (9 services on 1 host)
NSE: Script scanning 10.10.85.5.
NSE: Starting runlevel 1 (of 2) scan.
Initiating NSE at 11:16
Completed NSE at 11:16, 0.27s elapsed
NSE: Starting runlevel 2 (of 2) scan.
Initiating NSE at 11:16
Completed NSE at 11:16, 0.15s elapsed
Nmap scan report for 10.10.85.5
Host is up, received syn-ack (0.043s latency).
Scanned at 2023-12-09 11:15:57 GMT for 12s
Not shown: 991 filtered tcp ports (no-response)
PORT    STATE SERVICE    REASON  VERSION
21/tcp  open  ftp        syn-ack vsftpd 3.0.3
22/tcp  open  ssh        syn-ack OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
25/tcp  open  smtp       syn-ack Postfix smtpd
53/tcp  open  domain     syn-ack ISC BIND 9.16.1 (Ubuntu Linux)
80/tcp  open  http       syn-ack Apache httpd 2.4.41 ((Ubuntu))
110/tcp open  pop3       syn-ack Dovecot pop3d
143/tcp open  imap       syn-ack Dovecot imapd (Ubuntu)
993/tcp open  tcpwrapped syn-ack
995/tcp open  tcpwrapped syn-ack
Service Info: Host:  Masterschool.Masterschool.com; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
```

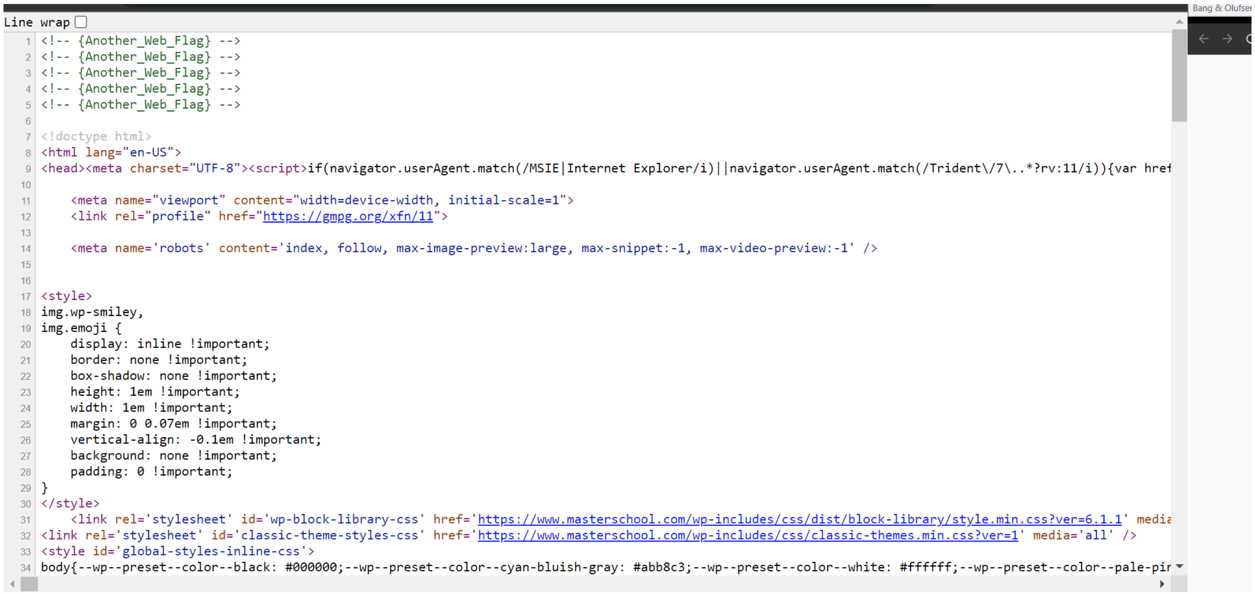
I found an http port open, so i pasted the Ip address of my target machine on my web browser and it showed me my first Webpage flag {STUDENT_CTF_WEB}



{STUDENT_CTF_Web}
industry experts

Learn online with world-class instructors and 24/7 support.
Pay no tuition until you get hired.

at this point i was stuck but Tom came through again and believe me i learnt a lot from his teachings and it was an eye opener for me. The second flag was hiding in plain sight and that leads us to the source page of the above webpage. and the flag found was **{Another_Web_Flag}**



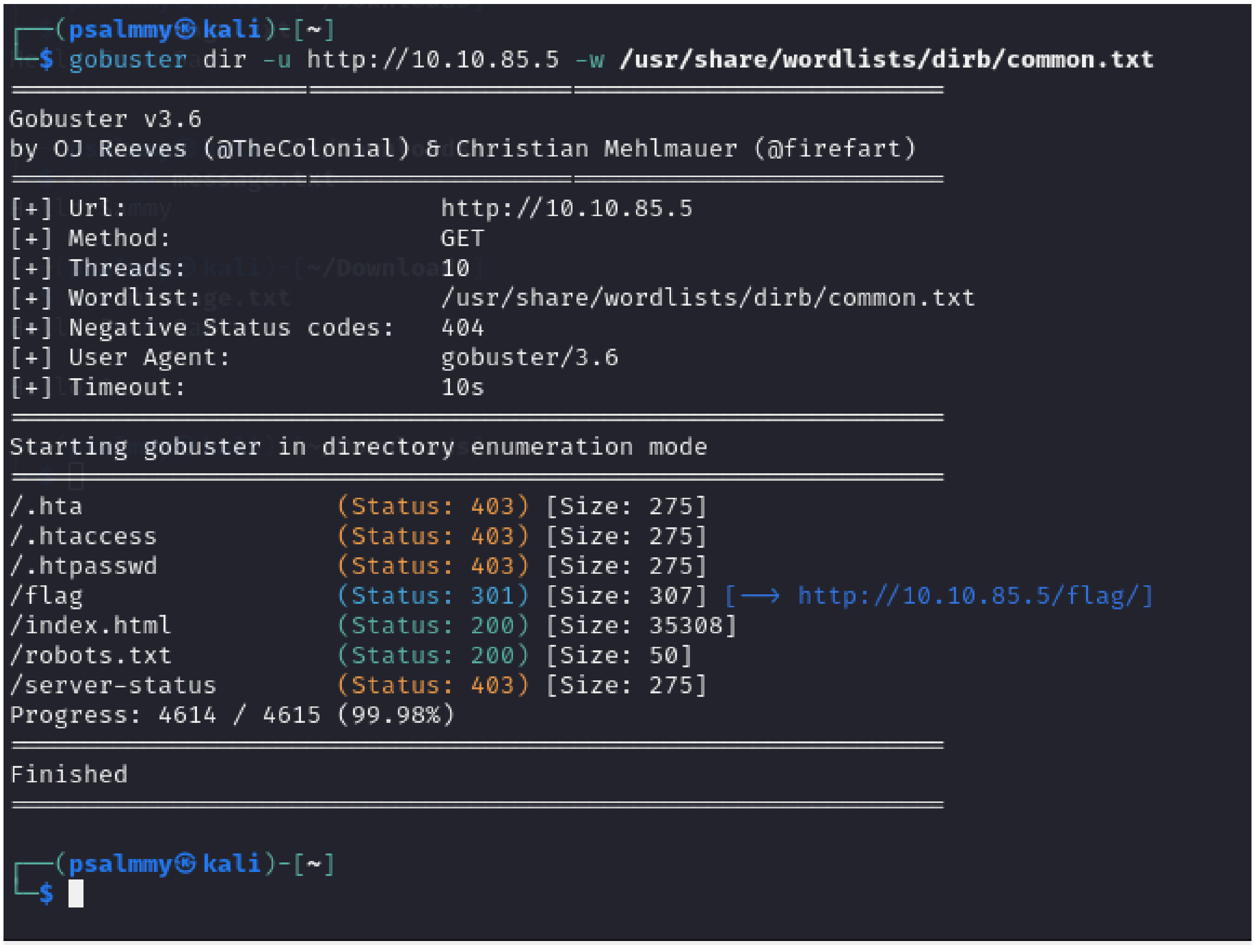
```
<!-- {Another_Web_Flag} -->
<!-- {Another_Web_Flag} -->
<!-- {Another_Web_Flag} -->
<!-- {Another_Web_Flag} -->
<!-- {Another_Web_Flag} -->

<!doctype html>
<html lang="en-US">
<head><meta charset="UTF-8"><script>if(navigator.userAgent.match(/MSIE|Internet Explorer/i)||navigator.userAgent.match(/Trident\/7\..*?rv:11/i)){var href

    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="profile" href="https://gmpg.org/xfn/11">

    <meta name='robots' content='index, follow, max-image-preview:large, max-snippet:-1, max-video-preview:-1' />


<style>
img.wp-smiley,
img.emoji {
    display: inline !important;
    border: none !important;
    box-shadow: none !important;
    height: 1em !important;
    width: 1em !important;
    margin: 0 0.07em !important;
    vertical-align: -0.1em !important;
    background: none !important;
    padding: 0 !important;
}
</style>
    <link rel='stylesheet' id='wp-block-library-css' href='https://www.masterschool.com/wp-includes/css/dist/block-library/style.min.css?ver=6.1.1' media
<link rel='stylesheet' id='classic-theme-styles-css' href='https://www.masterschool.com/wp-includes/css/classic-themes.min.css?ver=1' media='all' />
<style id='global-styles-inline-css'>
body{--wp--preset--color--black: #000000;--wp--preset--color--cyan-bluish-gray: #abb8c3;--wp--preset--color--white: #ffffff;--wp--preset--color--pale-pir
```

in the quest for the third flag, Tom introduced me to a tool which shows hidden pages on a webpage and the tool is called Gobuster(a tool used for directory and file brute forcing)

`gobuster dir -u targetmachineipaddress -w /usr/share/wordlists/dirb/common.txt`

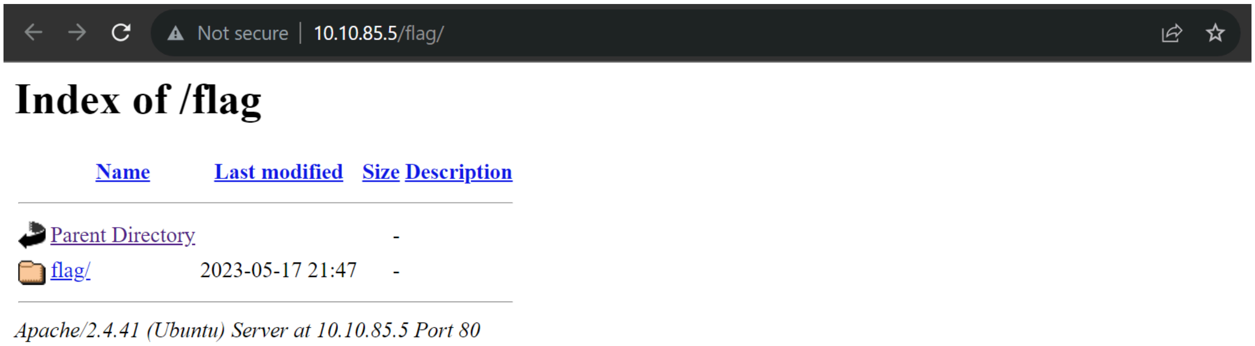
- `dir` : Specifies the mode of operation. In this case, it indicates that you i am performing directory brute-forcing.

- `-u`: Specifies the target URL.
- `-w`: Specifies the wordlist to be used for brute-forcing.
- `/usr/share/wordlists/dirb/common.txt`: This is an inbuilt wordlists that's always on linux machines.



```
┌──(psalmmy㉿kali)-[~]
└─$ gobuster dir -u http://10.10.85.5 -w /usr/share/wordlists/dirb/common.txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.85.5
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.hta                 (Status: 403) [Size: 275]
/.htaccess            (Status: 403) [Size: 275]
/.htpasswd            (Status: 403) [Size: 275]
/flag                 (Status: 301) [Size: 307] [--> http://10.10.85.5/flag/]
/index.html           (Status: 200) [Size: 35308]
/robots.txt           (Status: 200) [Size: 50]
/server-status        (Status: 403) [Size: 275]
Progress: 4614 / 4615 (99.98%)
===============================================================
Finished
===============================================================

┌──(psalmmy㉿kali)-[~]
└─$ 
```

from the above i found another webpage which is http://10.10.85.5/flag

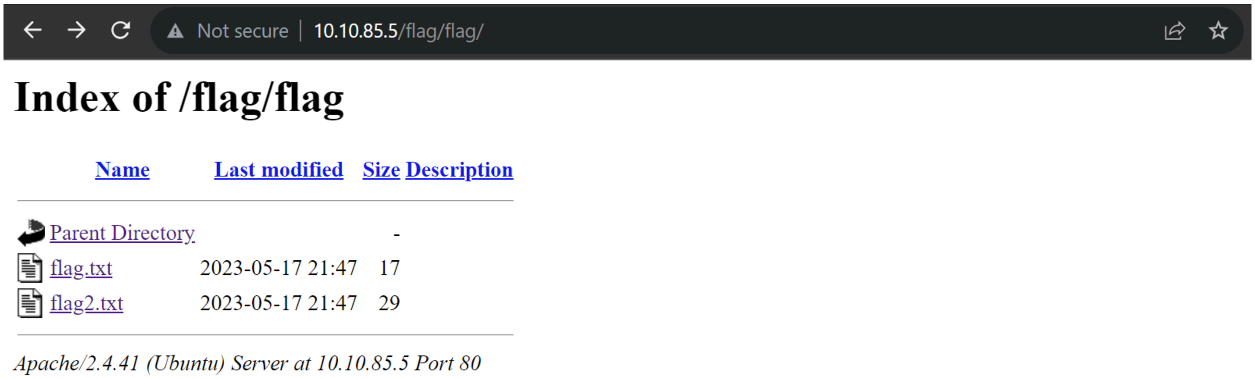


Index of /flag

Name	Last modified	Size	Description
Parent Directory		-	
flag/	2023-05-17 21:47	-	

Apache/2.4.41 (Ubuntu) Server at 10.10.85.5 Port 80

i opened the flag directory and found two text file inside

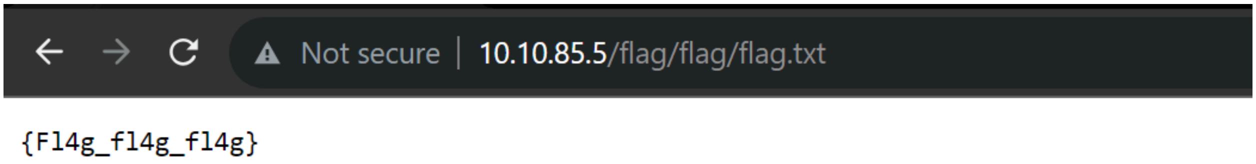


Index of /flag/flag

Name	Last modified	Size	Description
Parent Directory		-	
flag.txt	2023-05-17 21:47	17	
flag2.txt	2023-05-17 21:47	29	

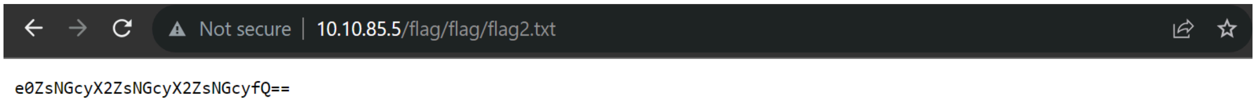
Apache/2.4.41 (Ubuntu) Server at 10.10.85.5 Port 80

flag.txt showed me my third flag which is **{Fl4g_Fl4g_Fl4g}**



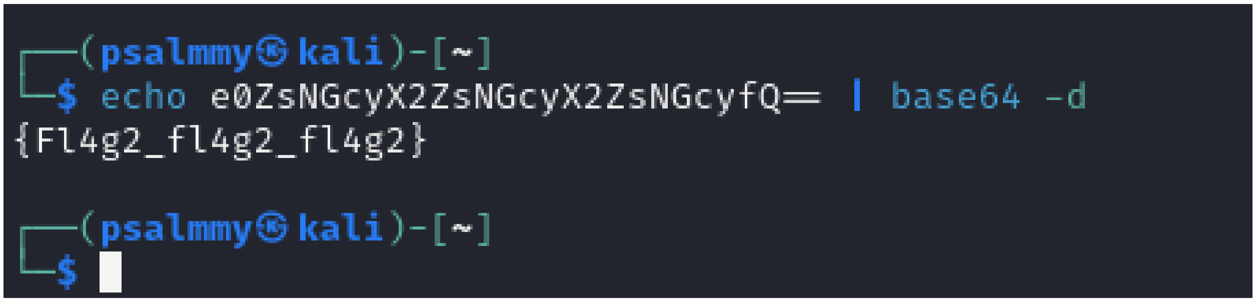
```
{Fl4g_fl4g_fl4g}
```

flag2.txt showed an encoded text in Base64



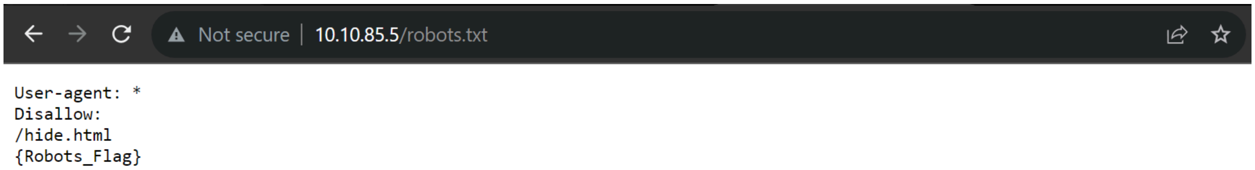
```
e0ZsNGcyX2ZsNGcyX2ZsNGcyfQ==
```

i then used the command `echo e0ZsNGcyX2ZsNGcyX2ZsNGcyfQ== | base64 -d` and my Fourth flag is **{Fl4g2_fl4g2_fl4g2}**



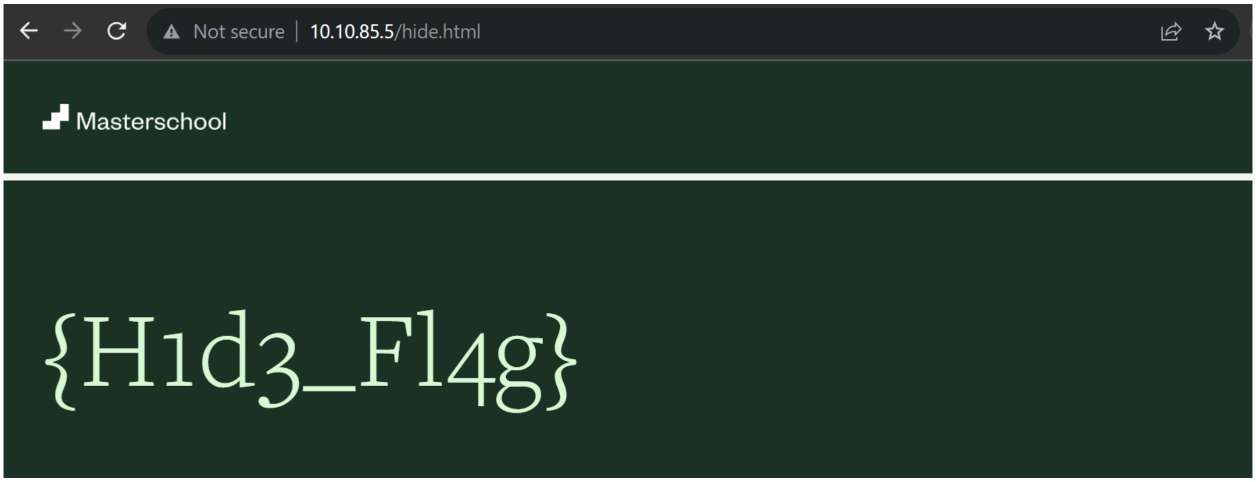
```
┌──(psalmmy㉿kali)-[~]
└─$ echo e0ZsNGcyX2ZsNGcyX2ZsNGcyfQ== | base64 -d
{Fl4g2_fl4g2_fl4g2}

┌──(psalmmy㉿kali)-[~]
└─$ 
```

i went back to my gobuster result and found one more accessible webpage which is 10.10.85.5/robots.txt. and behold when i opened the page, i found my fifth webpage flag **{Robots_Flag}**



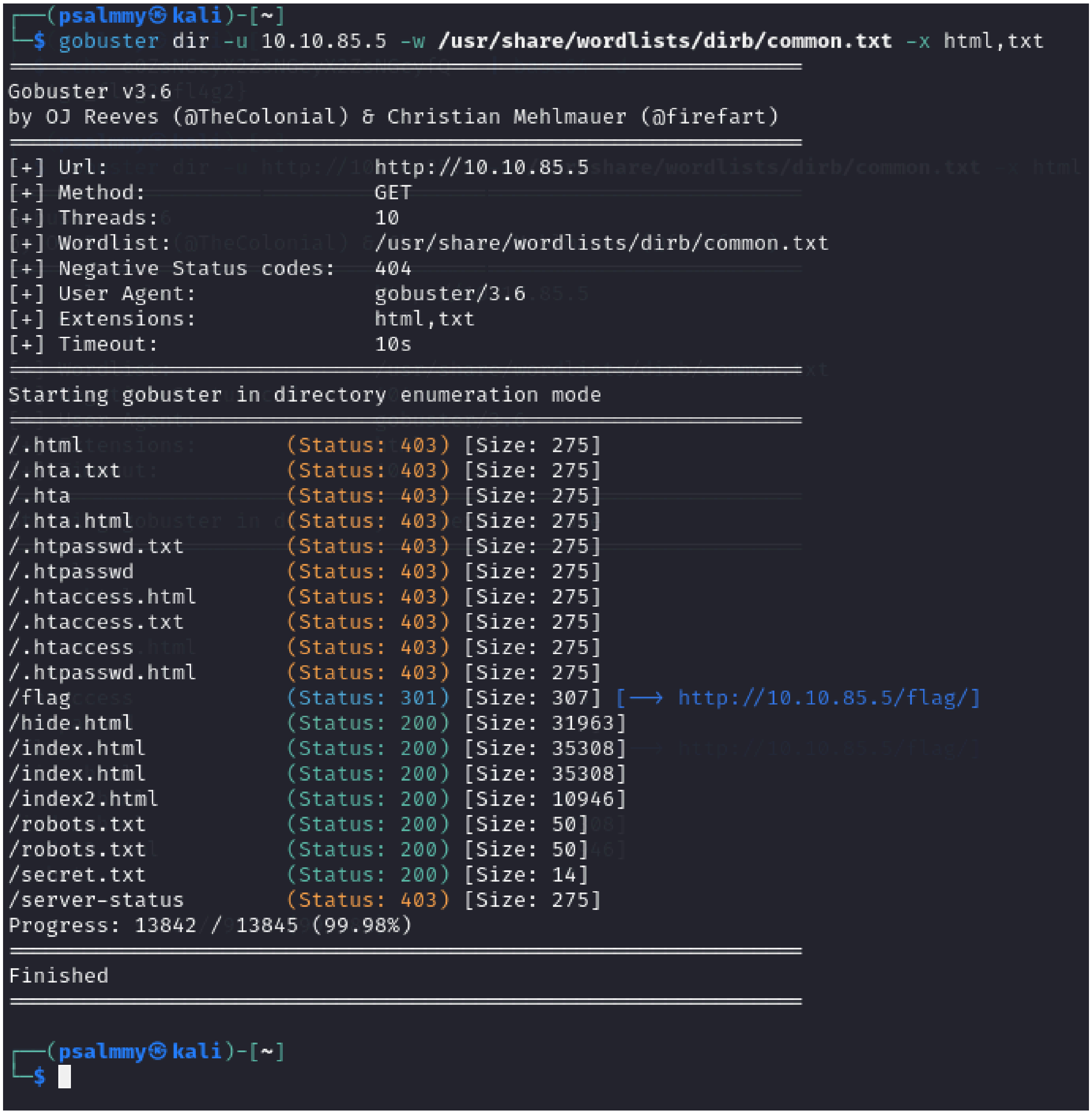
```
User-agent: *
Disallow:
/hide.html
{Robots_Flag}
```

looking at the result from the robots.txt page i found the /hide.html so i proceeded to the page and i found my Sixth webpage flag **{H1d3_Fl4g}**



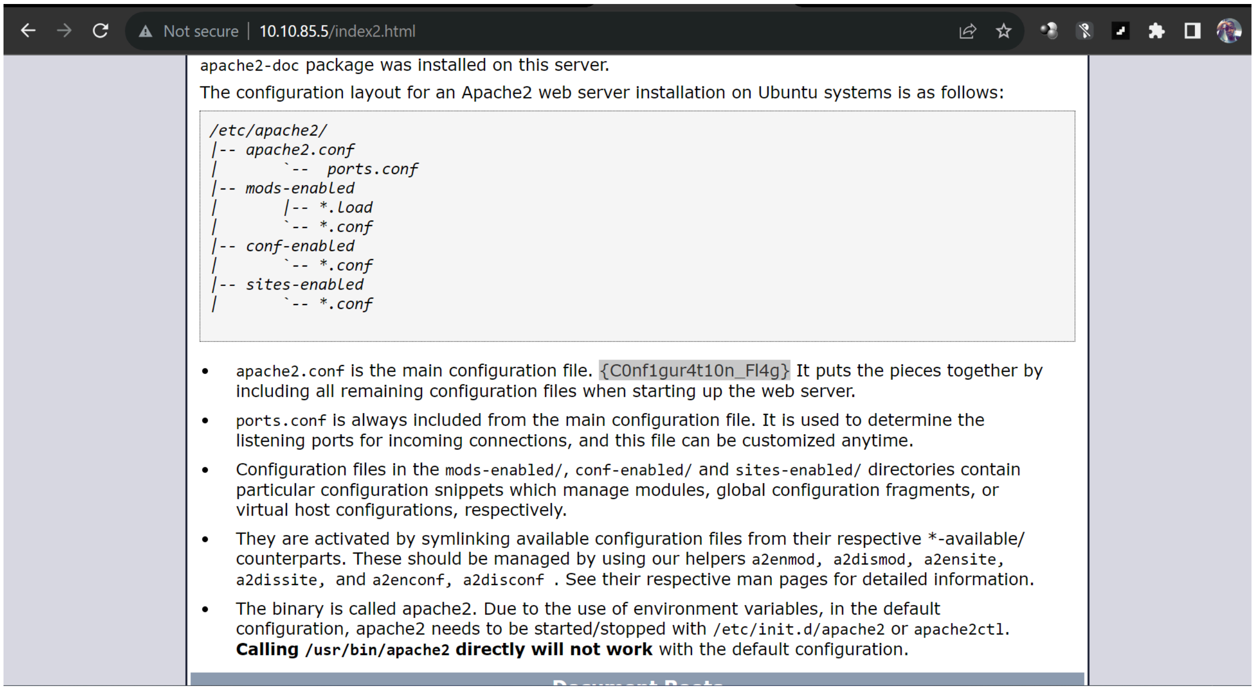
{H1d3_Fl4g}

from the above image the hide.html wasn't displayed in the results supplied by gobuster and i still have two flags left to find. this kind of depicts that there are still some hidden html pages and txt files not displayed by go buster. so in other to display this, i would use the command: `gobuster dir -u targetmachineipaddress -w /usr/share/wordlists/dirb/common.txt -x html,txt`



```
┌──(psalmmy㉿kali)-[~]
└─$ gobuster dir -u 10.10.85.5 -w /usr/share/wordlists/dirb/common.txt -x html,txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.85.5
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Extensions:              html,txt
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.html                (Status: 403) [Size: 275]
/.hta.txt             (Status: 403) [Size: 275]
/.hta                 (Status: 403) [Size: 275]
/.hta.html            (Status: 403) [Size: 275]
/.htpasswd.txt        (Status: 403) [Size: 275]
/.htpasswd            (Status: 403) [Size: 275]
/.htaccess.html       (Status: 403) [Size: 275]
/.htaccess.txt        (Status: 403) [Size: 275]
/.htaccess            (Status: 403) [Size: 275]
/.htpasswd.html       (Status: 403) [Size: 275]
/flag                 (Status: 301) [Size: 307] [--> http://10.10.85.5/flag/]
/hide.html            (Status: 200) [Size: 31963]
/index.html           (Status: 200) [Size: 35308]
/index.html           (Status: 200) [Size: 35308]
/index2.html          (Status: 200) [Size: 10946]
/robots.txt           (Status: 200) [Size: 50]
/robots.txt           (Status: 200) [Size: 50]
/secret.txt           (Status: 200) [Size: 14]
/server-status        (Status: 403) [Size: 275]
Progress: 13842 / 13845 (99.98%)
===============================================================
Finished
===============================================================

┌──(psalmmy㉿kali)-[~]
└─$ 
```

from the above, i found two new files which are index2.html and secret.txt. i opened the index2.html page and found my Seventh Flag **{C0nf1gur4t10n_Fl4g}**.



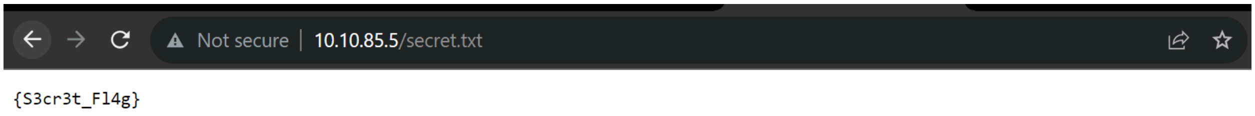
apache2-doc package was installed on this server.

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

```
/etc/apache2/
|-- apache2.conf
|        `--  ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-enabled
|       `-- *.conf
```

- `apache2.conf` is the main configuration file. {C0nf1gur4t10n_Fl4g} It puts the pieces together by including all remaining configuration files when starting up the web server.
- `ports.conf` is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
- Configuration files in the `mods-enabled/`, `conf-enabled/` and `sites-enabled/` directories contain particular configuration snippets which manage modules, global configuration fragments, or virtual host configurations, respectively.
- They are activated by symlinking available configuration files from their respective *-available/ counterparts. These should be managed by using our helpers `a2enmod`, `a2dismod`, `a2ensite`, `a2dissite`, and `a2enconf`, `a2disconf` . See their respective man pages for detailed information.
- The binary is called apache2. Due to the use of environment variables, in the default configuration, apache2 needs to be started/stopped with `/etc/init.d/apache2` or `apache2ctl`. **Calling `/usr/bin/apache2` directly will not work** with the default configuration.

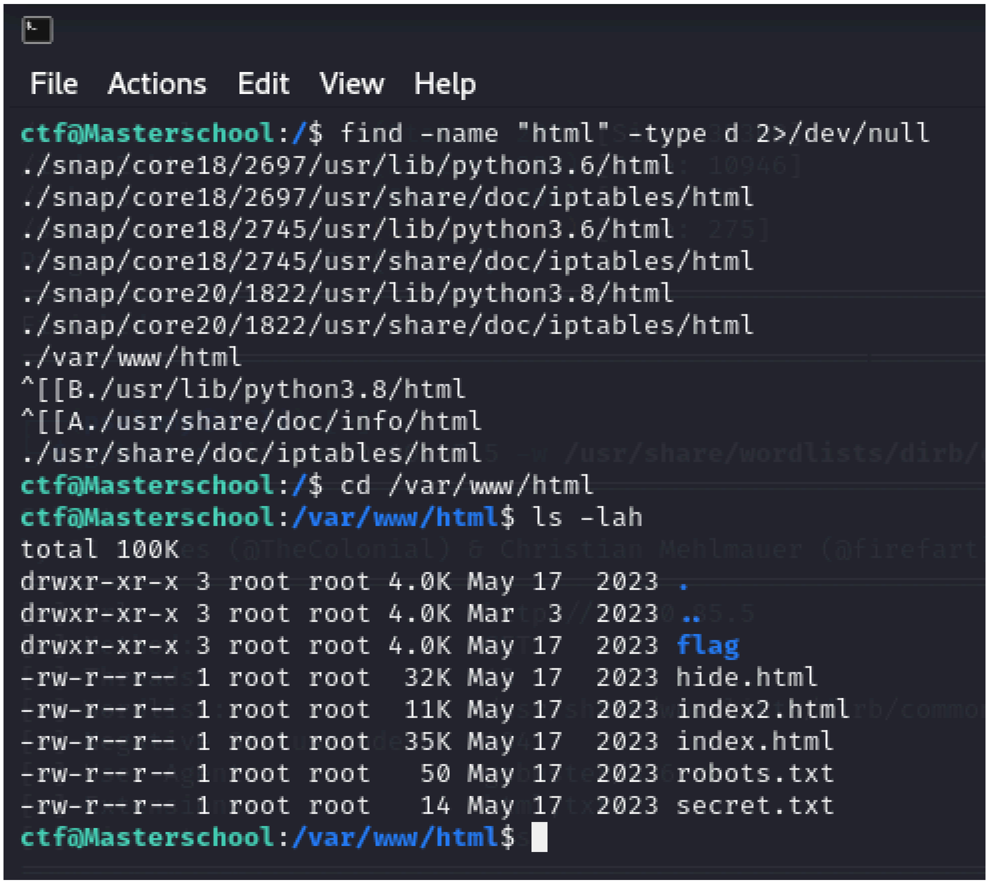
opened the secret.txt page and the Eight flag **{S3cr3t_Fl4g}** was there waiting for me



```
{S3cr3t_Fl4g}
```

The above exercise took a lot of trials and errors which if included in this documentation would make it cumbersome, but it was a wonderful experience and i love every aspect of it, there is another method of getting all the above flags using commands on my Linux machine, but i find the above more interesting as i had to learn outside the box, so i would show you the other method.

i used the find command to look up html directories, the i cd into the /var/www/html directory, this was where i used the command `ls -lah` which showed me all the folders & text i shared earlier

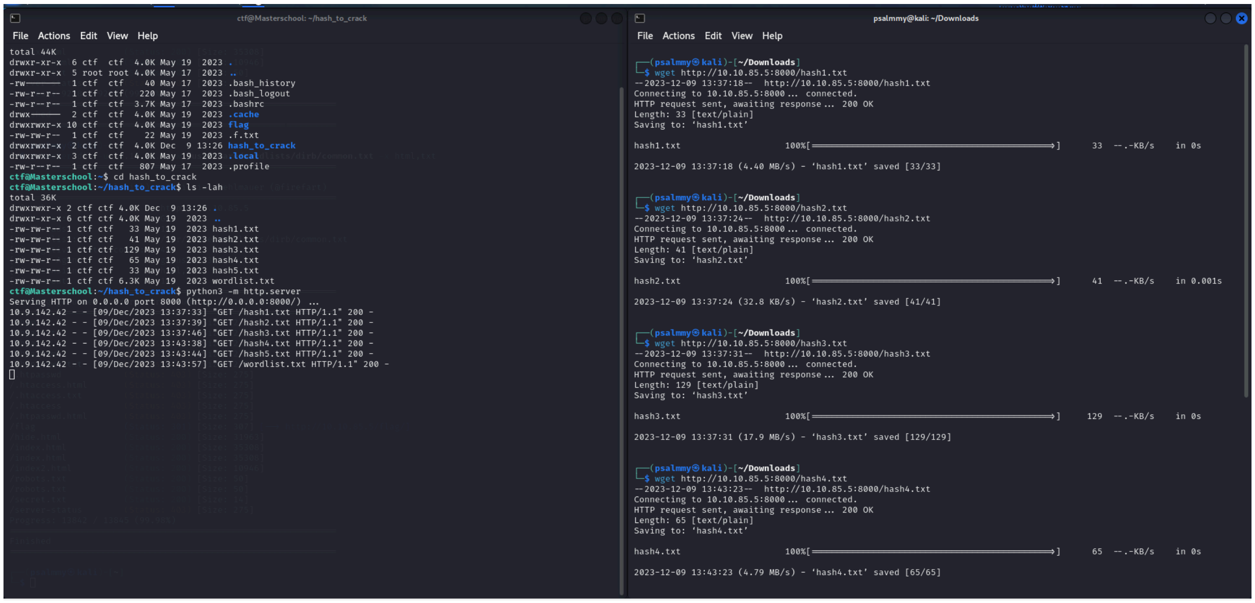


```
ctf@Masterschool:/$ find -name "html" -type d 2>/dev/null
./snap/core18/2697/usr/lib/python3.6/html
./snap/core18/2697/usr/share/doc/iptables/html
./snap/core18/2745/usr/lib/python3.6/html
./snap/core18/2745/usr/share/doc/iptables/html
./snap/core20/1822/usr/lib/python3.8/html
./snap/core20/1822/usr/share/doc/iptables/html
./var/www/html
^[[B./usr/lib/python3.8/html
^[[A./usr/share/doc/info/html
./usr/share/doc/iptables/html
ctf@Masterschool:/$ cd /var/www/html
ctf@Masterschool:/var/www/html$ ls -lah
total 100K
drwxr-xr-x 3 root root 4.0K May 17  2023 .
drwxr-xr-x 3 root root 4.0K Mar  3  2023 ..
drwxr-xr-x 3 root root 4.0K May 17  2023 flag
-rw-r--r-- 1 root root  32K May 17  2023 hide.html
-rw-r--r-- 1 root root  11K May 17  2023 index2.html
-rw-r--r-- 1 root root  35K May 17  2023 index.html
-rw-r--r-- 1 root root   50 May 17  2023 robots.txt
-rw-r--r-- 1 root root   14 May 17  2023 secret.txt
ctf@Masterschool:/var/www/html$ 
```

STEP 8: Hash Cracking

i navigated to the ctf directory and opened the hash_to_crack folder, my hash texts and wordlists are in this folder, but user ctf doesn't have adminsitrative access to install some of the tools which i need to crack the hash. this made me use the command `python3 -m http.server` to enable me download files from ctf to my machine. i used the `wget` `http://ctfipaddress:8000/file` `tobedownloaded` on my personal machine.

it can be noted that ctf machine recognized my download attempts and registered my ip address and the documents i was able to retrieve from it's machine.

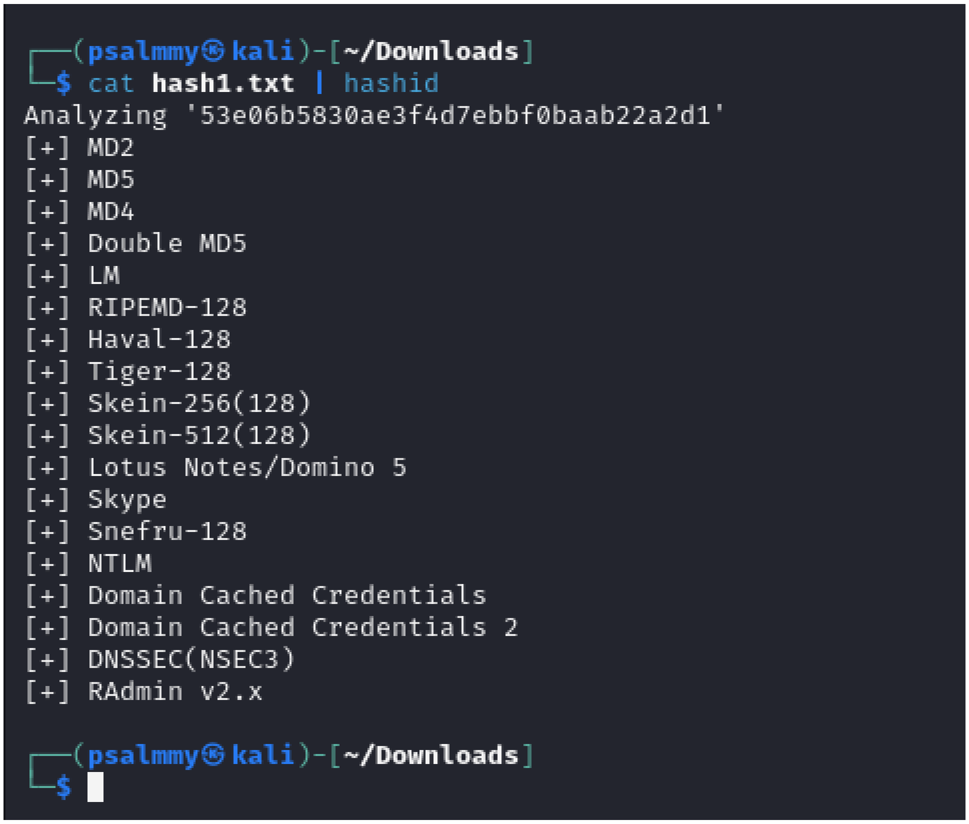

ctf@Masterschool: ~/hash_to_crack
File Actions Edit View Help
psalmmy@kali: ~/Downloads
File Actions Edit View Help

```
total 44K
drwxr-xr-x  6 ctf  ctf  4.0K May 19  2023 .
drwxr-xr-x  5 root root 4.0K May 17  2023 ..
-rw———————  1 ctf  ctf    40 May 17  2023 .bash_history
-rw-r--r--  1 ctf  ctf   220 May 17  2023 .bash_logout
-rw-r--r--  1 ctf  ctf  3.7K May 17  2023 .bashrc
drwx——————  2 ctf  ctf  4.0K May 19  2023 .cache
drwxrwxr-x 10 ctf  ctf  4.0K May 19  2023 flag
-rw-rw-r--  1 ctf  ctf    22 May 19  2023 .f.txt
drwxrwxr-x  2 ctf  ctf  4.0K Dec  9 13:26 hash_to_crack
drwxrwxr-x  3 ctf  ctf  4.0K May 19  2023 .local
-rw-r--r--  1 ctf  ctf   807 May 17  2023 .profile
ctf@Masterschool:~$ cd hash_to_crack
ctf@Masterschool:~/hash_to_crack$ ls -lah
total 36K
drwxrwxr-x 2 ctf ctf 4.0K Dec  9 13:26 .
drwxr-xr-x 6 ctf ctf 4.0K May 19  2023 ..
-rw-rw-r-- 1 ctf ctf   33 May 19  2023 hash1.txt
-rw-rw-r-- 1 ctf ctf   41 May 19  2023 hash2.txt
-rw-rw-r-- 1 ctf ctf  129 May 19  2023 hash3.txt
-rw-rw-r-- 1 ctf ctf   65 May 19  2023 hash4.txt
-rw-rw-r-- 1 ctf ctf   33 May 19  2023 hash5.txt
-rw-rw-r-- 1 ctf ctf 6.3K May 19  2023 wordlist.txt
ctf@Masterschool:~/hash_to_crack$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.9.142.42 - - [09/Dec/2023 13:37:33] "GET /hash1.txt HTTP/1.1" 200 -
10.9.142.42 - - [09/Dec/2023 13:37:39] "GET /hash2.txt HTTP/1.1" 200 -
10.9.142.42 - - [09/Dec/2023 13:37:46] "GET /hash3.txt HTTP/1.1" 200 -
10.9.142.42 - - [09/Dec/2023 13:43:38] "GET /hash4.txt HTTP/1.1" 200 -
10.9.142.42 - - [09/Dec/2023 13:43:44] "GET /hash5.txt HTTP/1.1" 200 -
10.9.142.42 - - [09/Dec/2023 13:43:57] "GET /wordlist.txt HTTP/1.1" 200 -
```

```
┌──(psalmmy㉿kali)-[~/Downloads]
└─$ wget http://10.10.85.5:8000/hash1.txt
--2023-12-09 13:37:18--  http://10.10.85.5:8000/hash1.txt
Connecting to 10.10.85.5:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 33 [text/plain]
Saving to: ‘hash1.txt’

hash1.txt                100%[===================================>]      33  --.-KB/s    in 0s

2023-12-09 13:37:18 (4.40 MB/s) - ‘hash1.txt’ saved [33/33]


┌──(psalmmy㉿kali)-[~/Downloads]
└─$ wget http://10.10.85.5:8000/hash2.txt
--2023-12-09 13:37:24--  http://10.10.85.5:8000/hash2.txt
Connecting to 10.10.85.5:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 41 [text/plain]
Saving to: ‘hash2.txt’

hash2.txt                100%[===================================>]      41  --.-KB/s    in 0.001s

2023-12-09 13:37:24 (32.8 KB/s) - ‘hash2.txt’ saved [41/41]


┌──(psalmmy㉿kali)-[~/Downloads]
└─$ wget http://10.10.85.5:8000/hash3.txt
--2023-12-09 13:37:31--  http://10.10.85.5:8000/hash3.txt
Connecting to 10.10.85.5:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 129 [text/plain]
Saving to: ‘hash3.txt’

hash3.txt                100%[===================================>]     129  --.-KB/s    in 0s

2023-12-09 13:37:31 (17.9 MB/s) - ‘hash3.txt’ saved [129/129]


┌──(psalmmy㉿kali)-[~/Downloads]
└─$ wget http://10.10.85.5:8000/hash4.txt
--2023-12-09 13:43:23--  http://10.10.85.5:8000/hash4.txt
Connecting to 10.10.85.5:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 65 [text/plain]
Saving to: ‘hash4.txt’

hash4.txt                100%[===================================>]      65  --.-KB/s    in 0s

2023-12-09 13:43:23 (4.79 MB/s) - ‘hash4.txt’ saved [65/65]
```

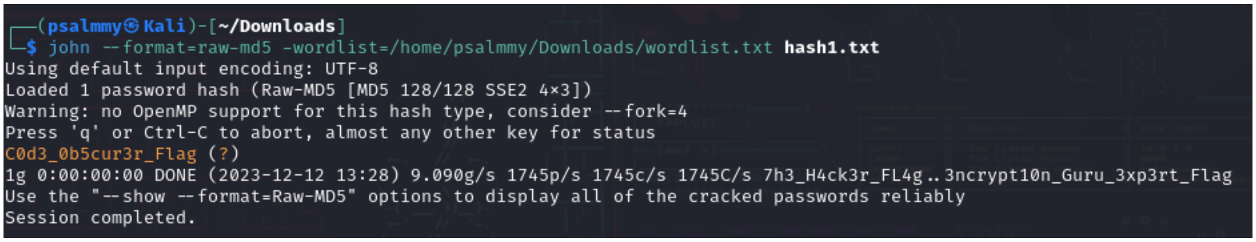
now that i have all the documents i need on my machine let’s go Hashing. first thing i do is get the hash type, i can use various commands like `cat hashfile | hashid` , `echo $(<hashfile) | hashid` , `hashid -m $(cat hashfile)`. learning how to do hash identification in three different ways was awesome. from the below i deduced that the hash type is MD5. now i can use john the ripper.

Hash 1:



```
┌──(psalmmy㉿kali)-[~/Downloads]
└─$ cat hash1.txt | hashid
Analyzing '53e06b5830ae3f4d7ebbf0baab22a2d1'
[+] MD2
[+] MD5
[+] MD4
[+] Double MD5
[+] LM
[+] RIPEMD-128
[+] Haval-128
[+] Tiger-128
[+] Skein-256(128)
[+] Skein-512(128)
[+] Lotus Notes/Domino 5
[+] Skype
[+] Snefru-128
[+] NTLM
[+] Domain Cached Credentials
[+] Domain Cached Credentials 2
[+] DNSSEC(NSEC3)
[+] RAdmin v2.x

┌──(psalmmy㉿kali)-[~/Downloads]
└─$ 
```

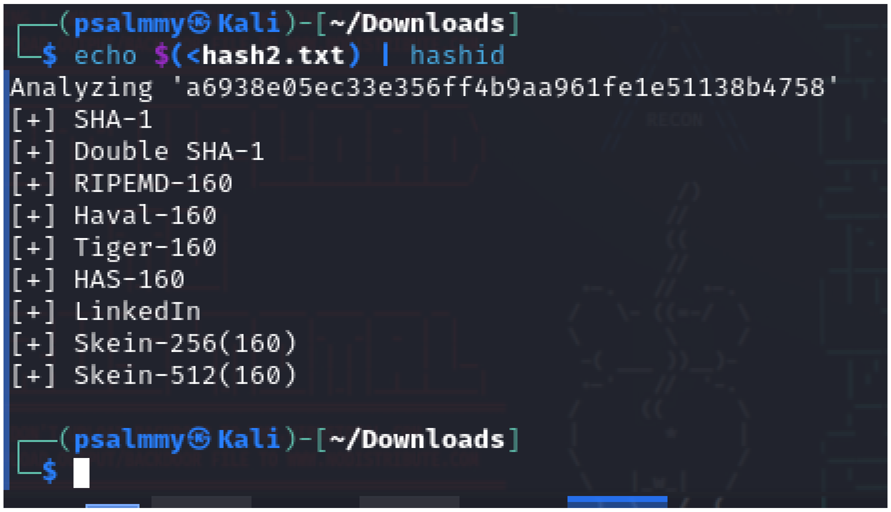
from the result below the password for for hash1.txt is {C0d3_0b5cur3r_Flag} and this is our first hash flag.



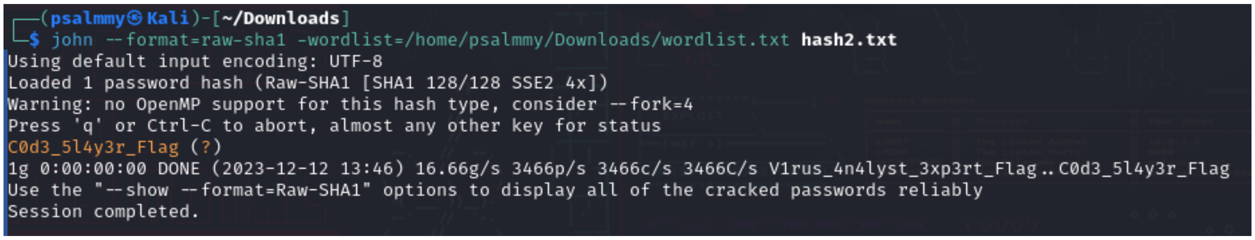
```
┌──(psalmmy㉿Kali)-[~/Downloads]
└─$ john --format=raw-md5 -wordlist=/home/psalmmy/Downloads/wordlist.txt hash1.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 128/128 SSE2 4×3])
Warning: no OpenMP support for this hash type, consider --fork=4
Press 'q' or Ctrl-C to abort, almost any other key for status
C0d3_0b5cur3r_Flag (?)
1g 0:00:00:00 DONE (2023-12-12 13:28) 9.090g/s 1745p/s 1745c/s 1745C/s 7h3_H4ck3r_FL4g..3ncrypt10n_Guru_3xp3rt_Flag
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.
```

Hash 2:

using the command `echo $(<hashfile) | hashid` to analyze the hash type



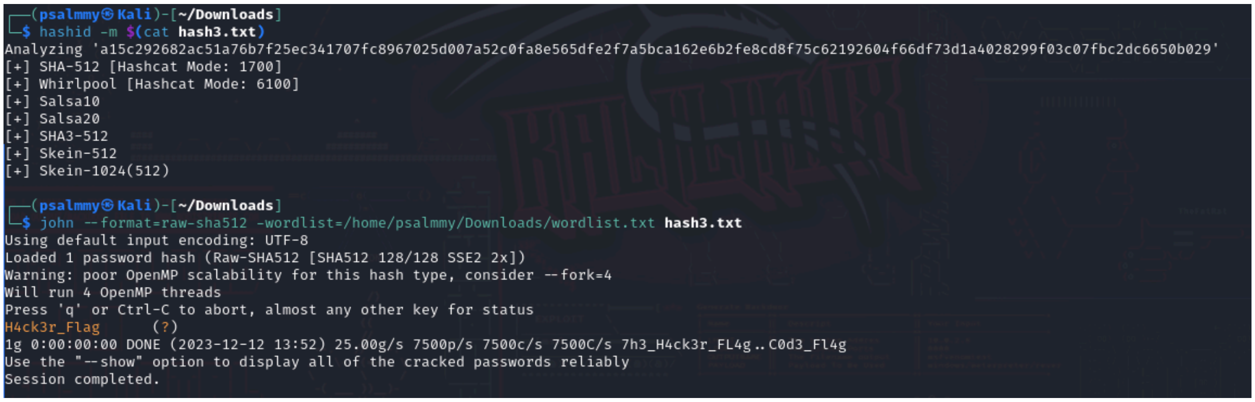
```
┌──(psalmmy㉿Kali)-[~/Downloads]
└─$ echo $(<hash2.txt) | hashid
Analyzing 'a6938e05ec33e356ff4b9aa961fe1e51138b4758'
[+] SHA-1
[+] Double SHA-1
[+] RIPEMD-160
[+] Haval-160
[+] Tiger-160
[+] HAS-160
[+] LinkedIn
[+] Skein-256(160)
[+] Skein-512(160)

┌──(psalmmy㉿Kali)-[~/Downloads]
└─$ 
```



```
┌──(psalmmy㉿Kali)-[~/Downloads]
└─$ john --format=raw-sha1 -wordlist=/home/psalmmy/Downloads/wordlist.txt hash2.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA1 [SHA1 128/128 SSE2 4x])
Warning: no OpenMP support for this hash type, consider --fork=4
Press 'q' or Ctrl-C to abort, almost any other key for status
C0d3_5l4y3r_Flag (?)
1g 0:00:00:00 DONE (2023-12-12 13:46) 16.66g/s 3466p/s 3466c/s 3466C/s V1rus_4n4lyst_3xp3rt_Flag..C0d3_5l4y3r_Flag
Use the "--show --format=Raw-SHA1" options to display all of the cracked passwords reliably
Session completed.
```

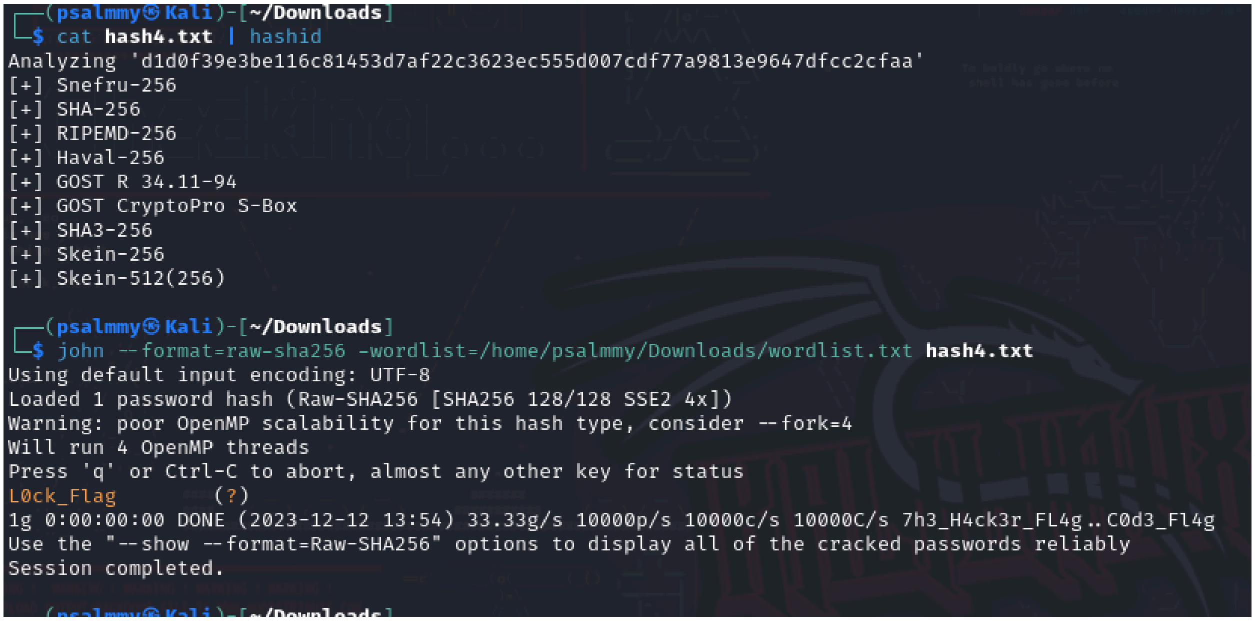
my Second hash flag is {C0d3_5l4y3r_Flag}

Hash 3:



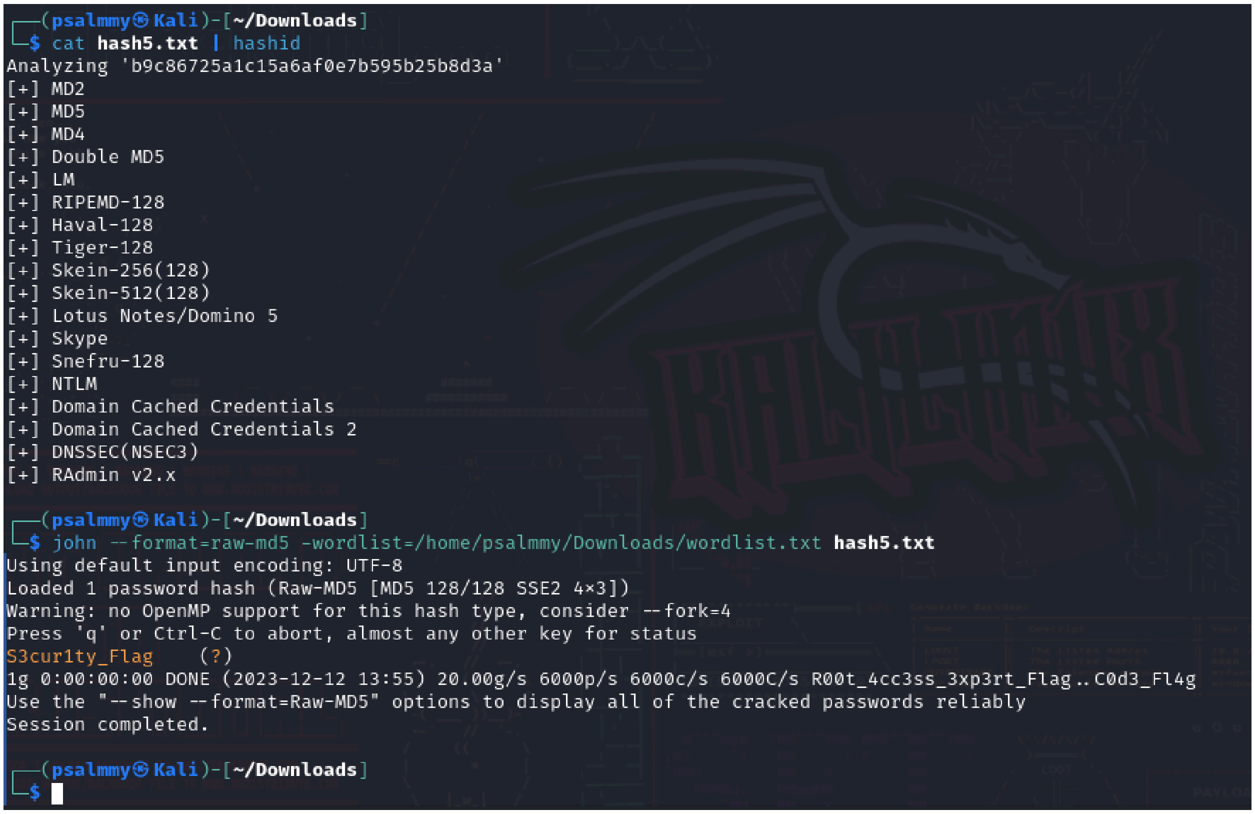
```
┌──(psalmmy㉿Kali)-[~/Downloads]
└─$ hashid -m $(cat hash3.txt)
Analyzing 'a15c292682ac51a76b7f25ec341707fc8967025d007a52c0fa8e565dfe2f7a5bca162e6b2fe8cd8f75c62192604f66df73d1a4028299f03c07fbc2dc6650b029'
[+] SHA-512 [Hashcat Mode: 1700]
[+] Whirlpool [Hashcat Mode: 6100]
[+] Salsa10
[+] Salsa20
[+] SHA3-512
[+] Skein-512
[+] Skein-1024(512)

┌──(psalmmy㉿Kali)-[~/Downloads]
└─$ john --format=raw-sha512 -wordlist=/home/psalmmy/Downloads/wordlist.txt hash3.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA512 [SHA512 128/128 SSE2 2x])
Warning: poor OpenMP scalability for this hash type, consider --fork=4
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
H4ck3r_Flag      (?)
1g 0:00:00:00 DONE (2023-12-12 13:52) 25.00g/s 7500p/s 7500c/s 7500C/s 7h3_H4ck3r_FL4g..C0d3_Fl4g
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Hash 4:



```
┌──(psalmmy㉿Kali)-[~/Downloads]
└─$ cat hash4.txt | hashid
Analyzing 'd1d0f39e3be116c81453d7af22c3623ec555d007cdf77a9813e9647dfcc2cfaa'
[+] Snefru-256
[+] SHA-256
[+] RIPEMD-256
[+] Haval-256
[+] GOST R 34.11-94
[+] GOST CryptoPro S-Box
[+] SHA3-256
[+] Skein-256
[+] Skein-512(256)

┌──(psalmmy㉿Kali)-[~/Downloads]
└─$ john --format=raw-sha256 -wordlist=/home/psalmmy/Downloads/wordlist.txt hash4.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA256 [SHA256 128/128 SSE2 4x])
Warning: poor OpenMP scalability for this hash type, consider --fork=4
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
L0ck_Flag        (?)
1g 0:00:00:00 DONE (2023-12-12 13:54) 33.33g/s 10000p/s 10000c/s 10000C/s 7h3_H4ck3r_FL4g..C0d3_Fl4g
Use the "--show --format=Raw-SHA256" options to display all of the cracked passwords reliably
Session completed.

┌──(psalmmy㉿Kali)-[~/Downloads]
```

Hash 5:



```
┌──(psalmmy㉿Kali)-[~/Downloads]
└─$ cat hash5.txt | hashid
Analyzing 'b9c86725a1c15a6af0e7b595b25b8d3a'
[+] MD2
[+] MD5
[+] MD4
[+] Double MD5
[+] LM
[+] RIPEMD-128
[+] Haval-128
[+] Tiger-128
[+] Skein-256(128)
[+] Skein-512(128)
[+] Lotus Notes/Domino 5
[+] Skype
[+] Snefru-128
[+] NTLM
[+] Domain Cached Credentials
[+] Domain Cached Credentials 2
[+] DNSSEC(NSEC3)
[+] RAdmin v2.x

┌──(psalmmy㉿Kali)-[~/Downloads]
└─$ john --format=raw-md5 -wordlist=/home/psalmmy/Downloads/wordlist.txt hash5.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 128/128 SSE2 4×3])
Warning: no OpenMP support for this hash type, consider --fork=4
Press 'q' or Ctrl-C to abort, almost any other key for status
S3cur1ty_Flag    (?)
1g 0:00:00:00 DONE (2023-12-12 13:55) 20.00g/s 6000p/s 6000c/s 6000C/s R00t_4cc3ss_3xp3rt_Flag..C0d3_Fl4g
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.

┌──(psalmmy㉿Kali)-[~/Downloads]
└─$ 
```