

PSP0201

Week 4

Writeup

Group Name:HUSTLERS

Members

ID	Name	Role
1211100708	Muhammad Faiz BIn Mohd Fauzi	leader
1211101962	Barath A L Saravanan	member
1211101804	AKHILESHNAIDU A/L JAYA KUMAR	MEMBER

[Day 11] Networking The Rogue Gnome

Tools used: Attackbox, Terminal, linpeas.sh

Solution/walkthrough:

Question 1&2

11.4.2. Vertical Privilege Escalation:

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

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

Question 3

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

Linux Command to enumerate the key for SSH

```
File Edit View Search Terminal Help
-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
```

Question 5

Chmod +x find.sh

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

Question 6

python3 -m http.server 9999

```
python -m http.server 8080
```

Question 7

Contents of the file located at /root/flag.txt

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

solution/walkthrough:

Question 1

[illegible]

Version number = Apache Tomcat

9.0.17

Question 2

TryHackMe | Learn Cy... TryHackMe Support Offline CyberChef GitHub - swisskyrepo/...

EDB-ID: 49039	CVE: 2020-1938	Author: SUNCSR	Type: WEBAPPS
EDB Verified: ✓		Exploit: 📄 / {}	
Platform: MULTIPLE	Date: 2020-11-13		
Vulnerable App:			

Question 3

```
root@ip-10-10-123-16: ~
File Edit View Search Terminal Help
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
12/12/2020  03:23    <DIR>          .
12/12/2020  03:23    <DIR>          ..
19/11/2020  21:39                825 elfwhacker.bat
19/11/2020  22:06                 27 flag1.txt
12/12/2020  03:23            73,802 YqkFa.exe
               3 File(s)              74,654 bytes
               2 Dir(s)  13,490,581,504 bytes free

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}
C:\Program Files\Apache Software Foundation\Tomcat 9.0\webapps\ROOT\WEB-INF\cgi-
bin>
[0] 0:bash 1:ruby* 2:bash- "ip-10-10-123-16" 03:25 12-Dec-20
```

Contents of flag1.txt = thm{whacking_all_the_elves}

Question 4

Payload options (windows/meterpreter/reverse_tcp):

Name	Current Setting	Required	Description
-----	-----	-----	-----
EXITFUNC	process	yes	Exit technique (Accepted: '', seh, thread, process, none)
LHOST	10.10.123.16	yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

LHOST

LPORT

[Day 13] Networking Coal for Christmas

Tools used:nmap,kali

Solution:walkthrough:

Question 1

```
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-30 14:41 EDT
Nmap scan report for 10.10.174.202
Host is up (0.22s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT      STATE SERVICE
22/tcp    open  ssh
23/tcp    open  telnet
111/tcp   open  rpcbind
```

Question 2

```
Trying 10.10.174.202 ...
Connected to 10.10.174.202.
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
```

Question 3

Use cat to get version number that is running

```
$ cat /etc/*release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=12.04
DISTRIB_CODENAME=precise
DISTRIB_DESCRIPTION="Ubuntu 12.04 LTS"
```

Question 4

```
*****
// 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!
// W. L- Yours Truly, 2016-5195) is a privilege escalation vulnerability i
// The Grinch
*****
```

Question 5

Research for verbatim syntax use to compile

```
15 //
16 // Compile with:
17 // gcc -pthread dirty.c -o dirty -lcrypt
18 //
```

Question 6

Default new username

```
9 //
10 // To use this exploit modify the user values according to your needs.
11 // The default is "firefart".
12 //
```

Question 7

```
firefart@christmas:~# touch coal
firefart@christmas:~# ls -l
total 8
-rwxr-xr-x 1 firefart root 1422 Nov 21 20:37 christmas.sh
-rw-r--r-- 1 firefart root  0 Dec 15 11:59 coal
-rw-r--r-- 1 firefart root 611 Nov 21 20:37 message_from_the_grinch.txt
firefart@christmas:~# tree | md5sum
8b16f00dd3b51efadb02c1df7f8427cc -
firefart@christmas:~#
```

Question 8

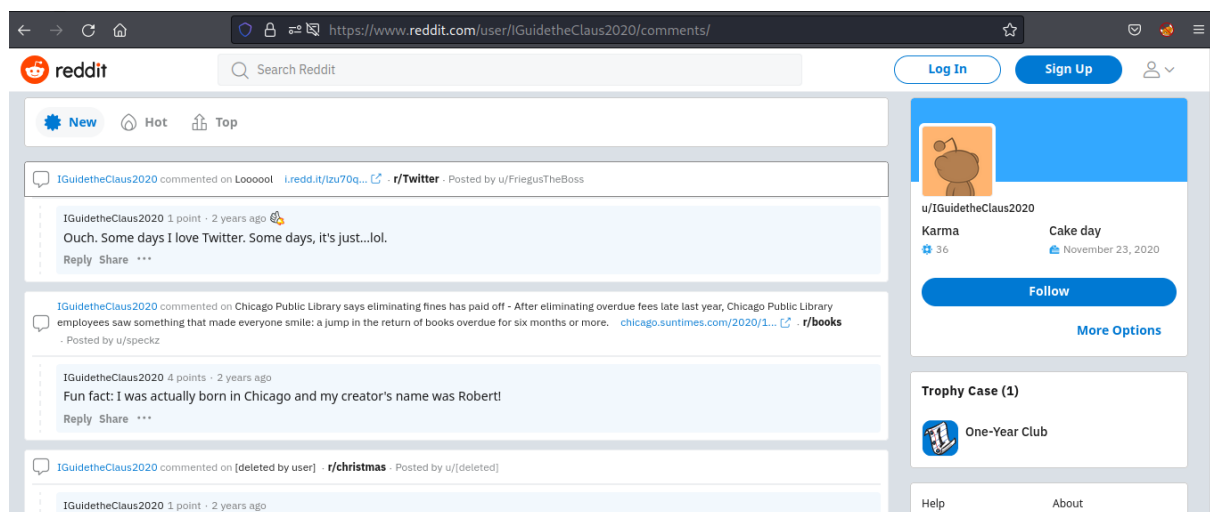
That C source code is a portion of a kernel exploit called **DirtyCow**. Dirty COW (CVE-2016-5195) is a privilege escalation vulnerability in the Linux Kernel, taking advantage of a race condition that was found in the way the Linux kernel's memory subsystem handled the copy-on-write (COW) breakage of private read-only memory mappings. An unprivileged local user could use this flaw to gain write access to otherwise read-only memory mappings and thus increase their privileges on the system.

Day 14 - [OSINT] Where's Rudolph?

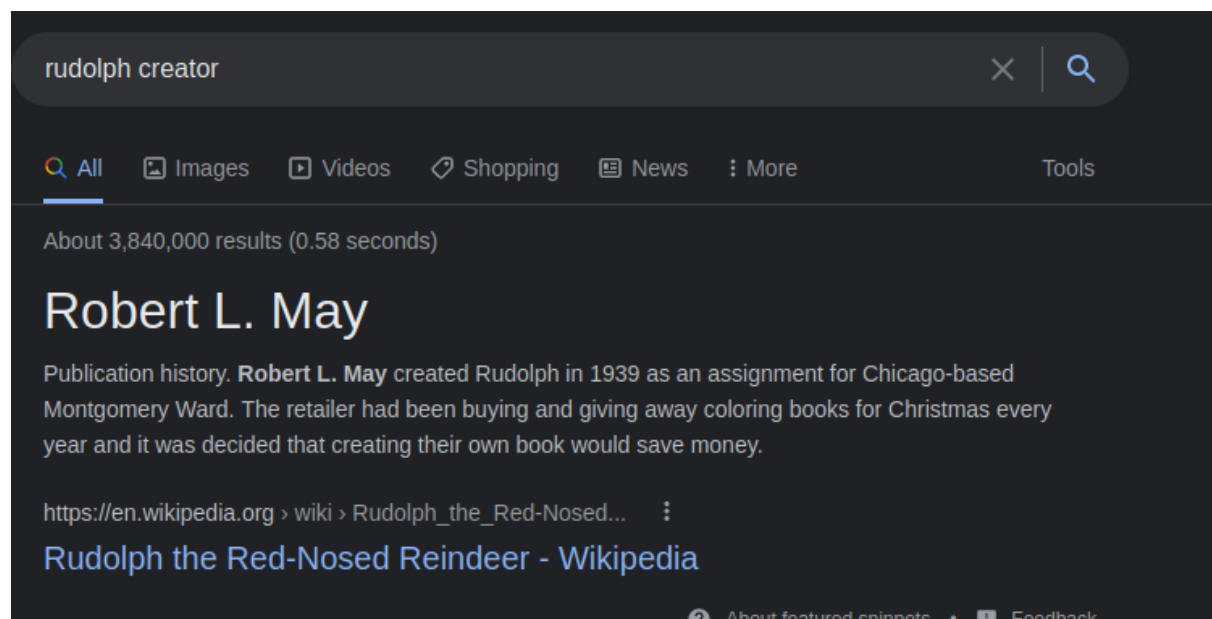
Tools used:google map,reddit,google image,twitter,name checkup site,

Solution/walkthrough

Question 1,2

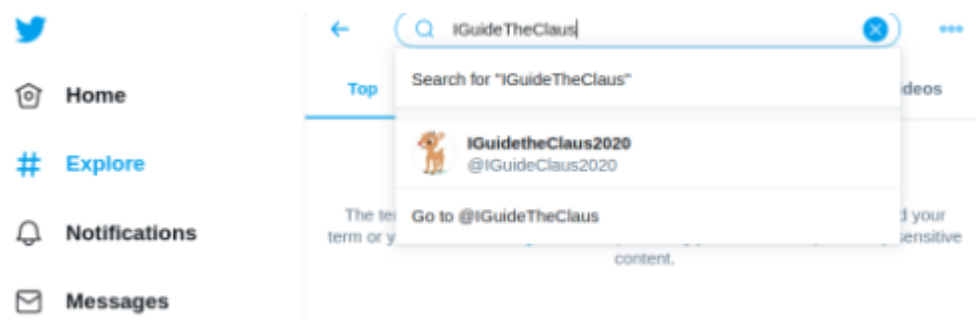


Question 3



Question 4

Question 5



Question 6



Question 7

Home > News & Events > Thompson Coburn 'floats' down Michigan Avenue in first Magnificent Mile Lights Festival appearance



Thompson Coburn 'floats' down Michigan Avenue in first Magnificent Mile Lights Festival appearance

December 9, 2019



On November 23, members of Thompson Coburn's Chicago office joined the annual BMO Harris Bank® Magnificent Mile Lights Festival® parade as both spectators and participants. As a 2019 Festival sponsor, Chicago attorneys and staff led a 30-foot-tall Rudolph the Red-Nosed Reindeer balloon down Michigan Avenue, followed closely behind by a Chicago trolley full of our attorneys and their families.

The Lights Festival parade, one of the largest holiday parades in the country, is part of a two-day holiday celebration that includes a tree-lighting ceremony and over one million holiday lights lining the northern stretch of Chicago's Michigan Avenue. A broadcast of the parade was shown the following evening on ABC7 Chicago and rebroadcast on several affiliate channels.

Question 8&9

Use exif.regex.info to get EXIF data and get flag

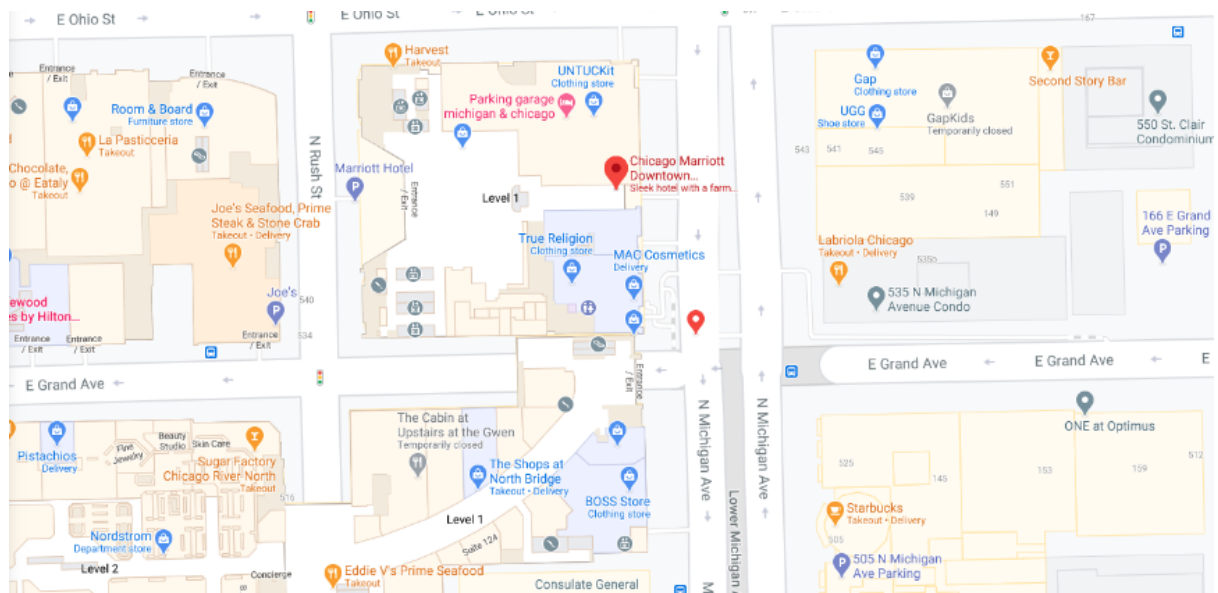
Basic Image Information

Target file: lights-festival-website.jpg

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

Q10

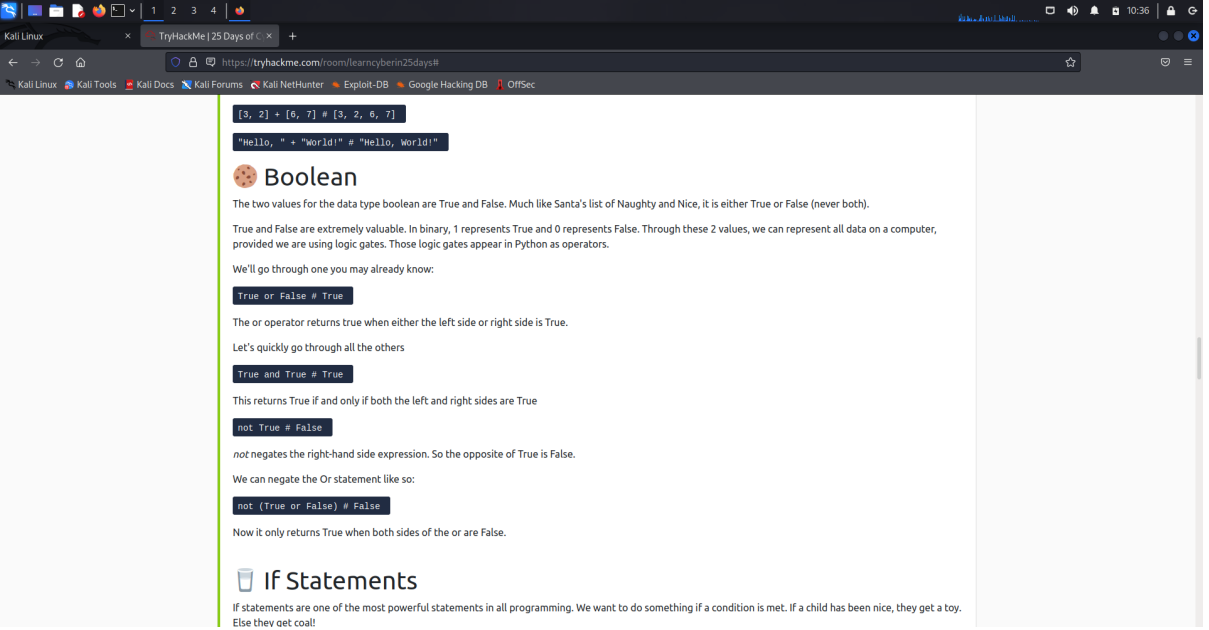
Question 11



Day15 - [Scripting] There's a Python in my stocking

TOOLS = MOZILA FIREFOX//PYTHON

Question 1



The screenshot shows a web browser window with the URL `https://tryhackme.com/room/learnpython25days#`. The page content is a Python tutorial section titled "Boolean". It explains that Boolean values are True and False, and introduces the `or`, `and`, and `not` operators. Code snippets are provided for each operator, showing how they evaluate expressions. The `or` operator returns True if either side is True. The `and` operator returns True only if both sides are True. The `not` operator negates the right-hand side expression. The section also mentions that Boolean values are used in If Statements.

```
[3, 2] + [6, 7] # [3, 2, 6, 7]
```

```
"Hello, " + "World!" # "Hello, World!"
```

Boolean

The two values for the data type boolean are True and False. Much like Santa's list of Naughty and Nice, it is either True or False (never both).

True and False are extremely valuable. In binary, 1 represents True and 0 represents False. Through these 2 values, we can represent all data on a computer, provided we are using logic gates. Those logic gates appear in Python as operators.

We'll go through one you may already know:

```
True or False # True
```

The `or` operator returns true when either the left side or right side is True.

Let's quickly go through all the others

```
True and True # True
```

This returns True if and only if both the left and right sides are True

```
not True # False
```

`not` negates the right-hand side expression. So the opposite of True is False.

We can negate the Or statement like so:

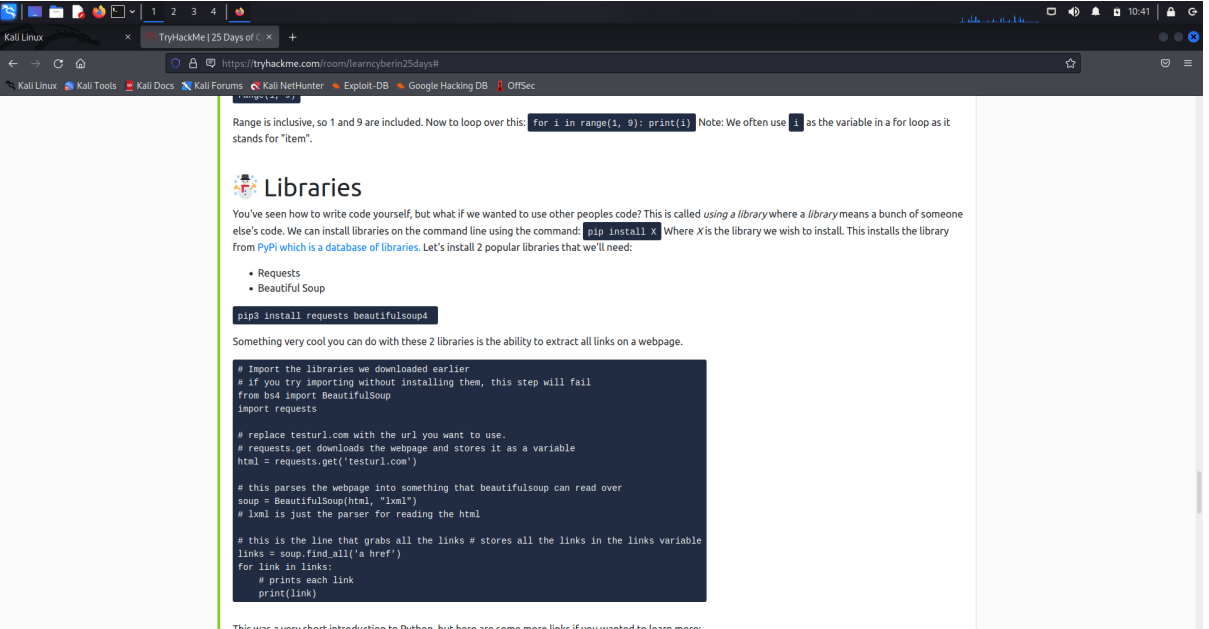
```
not (True or False) # False
```

Now it only returns True when both sides of the `or` are False.

If Statements

If statements are one of the most powerful statements in all programming. We want to do something if a condition is met. If a child has been nice, they get a toy. Else they get coal!

Question 2



The screenshot shows a web browser window with the URL `https://tryhackme.com/room/learnpython25days#`. The page content is a Python tutorial section titled "Libraries". It explains how to install libraries using `pip` and lists two popular libraries: Requests and BeautifulSoup. A code snippet shows how to install both libraries. Another code snippet shows how to use these libraries to extract all links from a webpage. The code imports `requests` and `BeautifulSoup`, then uses `requests.get` to fetch the webpage content and `BeautifulSoup` to parse it. Finally, it uses `soup.find_all` to extract all links from the webpage.

Range is inclusive, so 1 and 9 are included. Now to loop over this: `for i in range(1, 9): print(i)` Note: We often use `i` as the variable in a for loop as it stands for "item".

Libraries

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

- Requests
- BeautifulSoup

```
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

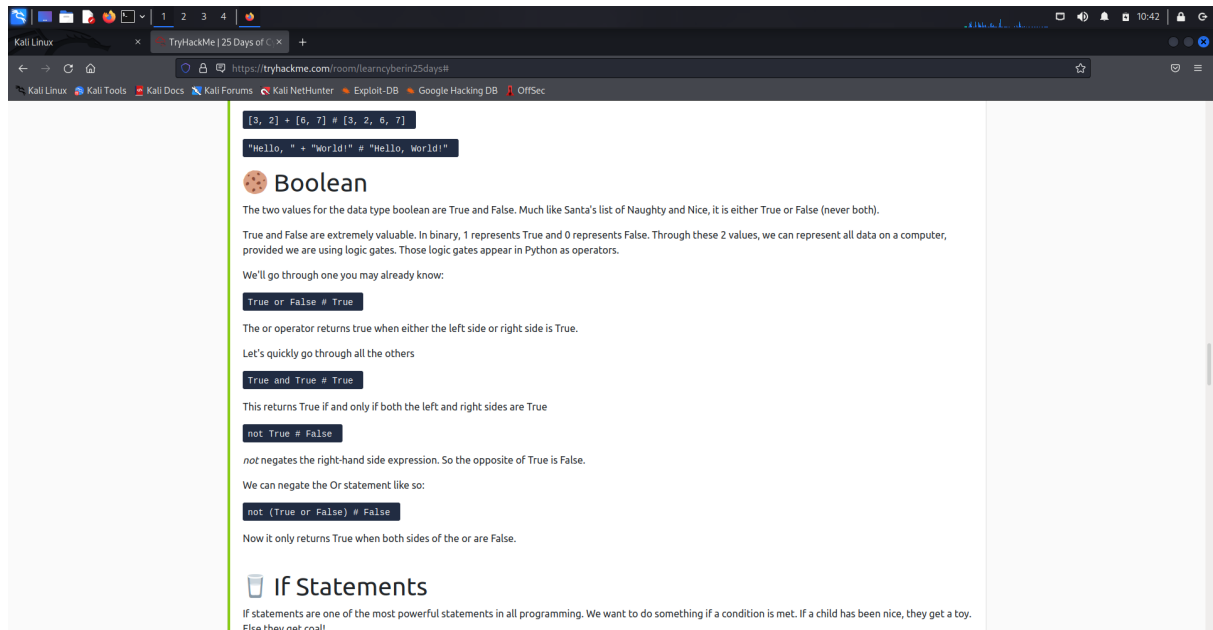
# replace testurl.com with the url you want to use.
# requests.get downloads the webpage and stores it as a variable
html = requests.get('testurl.com')

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

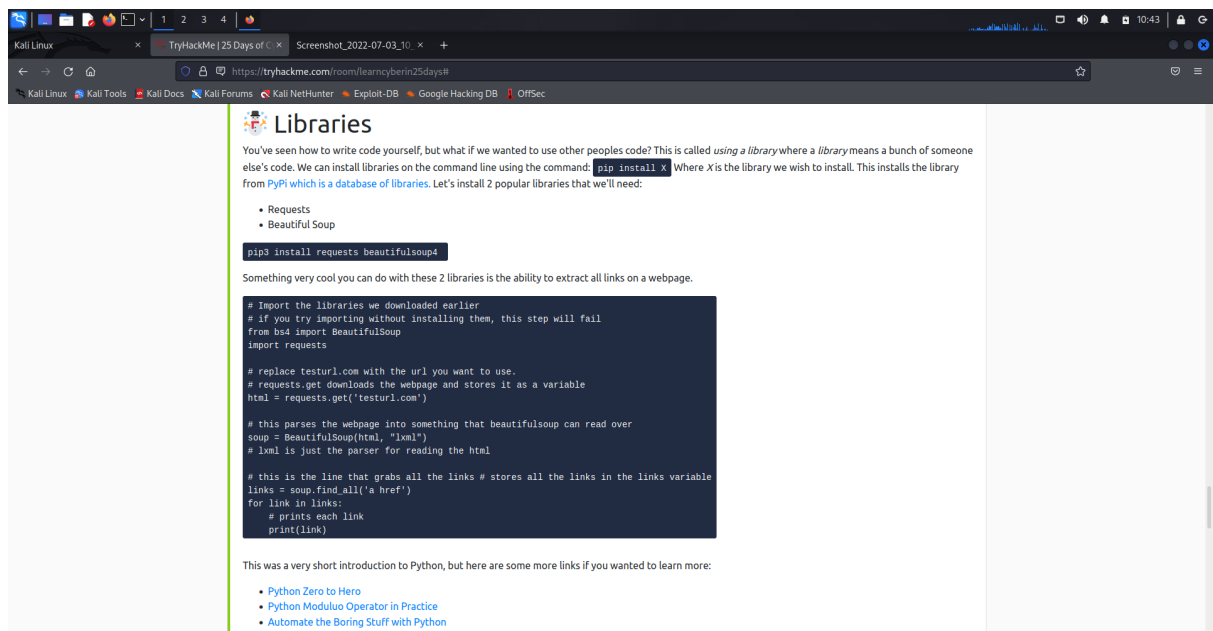
# this is the line that grabs all the links # stores all the links in the links variable
links = soup.find_all('a href')
for link in links:
    # prints each link
    print(link)
```

This was a short introduction to Libraries, but here are some more links if you wanted to learn more:

Question 3



Question 4



Question 5

```
home > jhdbw > Documentos > TryHackMe > Targets > learncyberin25days > hello.py > ...  
1  x = [1, 2, 3]  
2  
3  y = x  
4  
5  y.append(6)  
6  
7  print(x)  
8  
9  #import BeautifulSoup  
10 #import requests  
11 #html = requests.get('dragonsec.info')  
12 #soup = BeautifulSoup(html, "lxml")  
13 #links = soup.find_all('a href')  
14 #for link in links:  
15 #    print(link)  
16  
17 #names = ['Skidy', 'DorkStar', 'Ashu', 'Elf']  
18 #name = input("Cual es tu nombre? ")  
19 #print (names)  
20 #if name in names:  
21 #    print("Estas en la lista.")  
22 #else:  
23  
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: Python + [ ] [ ] [ ] [ ] [ ] [ ]  
(jhdbw@kali) - [~]  
$ python /home/jhdbw/Documentos/TryHackMe/Targets/learncyberin25days/hello.py  
[1, 2, 3, 6]  
(jhdbw@kali) - [~]  
$
```

Q6,7,8 have no proof because its already in it.