

PSP0201

Week 4

Writeup

GROUP NAME: Cyborgs

MEMBERS

ID NUMBER	NAME	ROLE
1211102066	Hemma Ravindran	Leader
1211100614	Tivaasheny Ananthan	Member
1211102168	Nicholas Cheok Jia Jie	Member
1211100986	Sarvesh Munusamy	Member

Day 11 - Networking The Rogue Gnome

Tool used: google and kali

Solution/walkthrough:

Question 1 , 2 and 3 :

We can get answers from the try hack me page on day 11.

The screenshot shows a browser with multiple tabs open. The active tab displays the TryHackMe '11.4. The directions of privilege escalation' page. The page content discusses horizontal and vertical privilege escalation. Below the page, a Firefox window is visible, showing a dark-themed interface with a message about not starting the browser recently. The taskbar at the bottom shows various application icons and the date/time.

Question 4 :

The name of the file that contains a list of users who are a part of the sudo group is shown at the picture below.

The screenshot shows a browser with multiple tabs open. The active tab displays the TryHackMe 'examplefiles' page. The page content explains the structure of file permissions and the /etc/sudoers file. Below the page, a terminal window shows the command 'ls -l' outputting three files: examplefile2 and examplefile3. A table provides a key for understanding file permissions. The taskbar at the bottom shows various application icons and the date/time.

Question 5:

The Linux Command to enumerate the key for SSH is also shown in try hack me page on day 11 at (11.6).

11.6. You Thought Enumeration Stopped at Nmap?

Wrong! We were just getting started. After gaining initial access, it's essential to begin to build a picture of the internals of the machine. We can look for a plethora of information such as other services that are running, sensitive data including passwords, executable scripts or binaries to abuse and more!

For example, we can use the find command to search for common folders or files that we may suspect to be on the machine:

- backups
- password
- admin
- config

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 >> /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 `>> /dev/null` to only show matches to us.

Can you think of any other files or folders we may want to `find`?

11.7. The "Priv Esc Checklist"

As you progress through your pentesting journey, you will begin to pick up a certain workflow for how you approach certain stages of an engagement. Whilst this workflow is truly yours, it will revolve around some fundamental steps in looking for vulnerabilities for privilege escalation.

1 Determining the kernel of the machine (kernel exploitation such as Dirtycow)

Question 6:

If we have an executable file named `find.sh` that we just copied from another machine, we have to write it with given format to get the answer by refering the try hack me (Vulnerability: SUID 101)

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 -rwxrwxr-x):

```
-rwxrwxr-x 1 cmnatic cmnatic 0 Dec 8 18:43 backup.sh
```

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).

SUID is simply a permission added to an executable that does a similar thing as sudo. However, instead, allows users to run the executable as whoever owns it as demonstrated below:

Filename	File Owner	User who is executing the file	User that the file is executed as
ex1	root	cmnatic	root
ex2	cmnatic	cmnatic	cmnatic
ex3	service	danny	service

Suddenly with the introduction of SUID, users no longer have to be a sudoer to run an executable as root. This can be legitimately used to allow applications that

Question 7:

We used kali to get the command to host a http server using python3 on port 9999. We also refer to the (11.10.2.) in try hack me.

The screenshot shows a terminal window titled "Kali-Linux-2021.4a-virtualbox-amd64 (Snapshot 1) [Running] - Oracle VM Virt...". The terminal content is as follows:

```
kali@kali:~  
h  
-- 2022-06-27 22:52:44 -- http://raw.githubusercontent.com/rebootuser/LinEnum/  
master/LinEnum.sh  
Resolving raw.githubusercontent.com (raw.githubusercontent.com) ... 185.199.10.  
8.133, 185.199.10.133, 185.199.111.133, ...  
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.1.  
08.133|:80 ... connected.  
HTTP request sent, awaiting response ... 301 Moved Permanently  
Location: https://raw.githubusercontent.com/rebootuser/LinEnum/master/LinEnum.  
.sh [following]  
-- 2022-06-27 22:52:44 -- https://raw.githubusercontent.com/rebootuser/LinEnum/  
/master/LinEnum.sh  
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.1.  
08.133|: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 0.09s  
2022-06-27 22:52:45 (517 KB/s) - 'LinEnum.sh' saved [46631/46631]  
  
└──(kali㉿kali)-[~]  
    $ python3 -m http.server 9999  
    Serving HTTP on 0.0.0.0 port 9999 (http://0.0.0.0:9999/) ...
```

The desktop environment includes a dock with icons for various applications like a browser, terminal, file manager, and file explorer. The system tray shows battery level at 10%, time as 22:54, date as 28/6/2022, and a notification for 20 new messages.

Question 8:

This is the content of the file located at /root/flag.txt.

similarly to the subscriber Pathways.

Answer the questions below

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

Vertical **Correct Answer**

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

sudoers **Correct Answer**

Use SSH to log in to the vulnerable machine like so: `ssh cmnatic@10.10.154.1`

Input the following password when prompted: `aoc2020`

No answer needed **Question Done**

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.

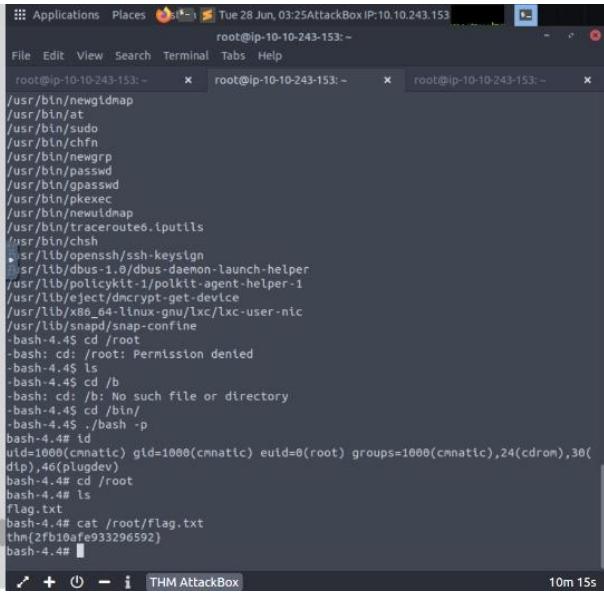
You may find uploading some of the enumeration scripts that were used during today's task to be useful.

No answer needed **Question Done** **Hint**

Use this executable to launch a system shell as root.

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

thm{2fb10afe933296592} **Correct Answer** **Hint**



Thought Process/Methodology:

Firstly, we need to start machine and also our attackbox. Next we need to type tmux. Then, we have to type `ssh cmnatic@10.10.52.17` and type yes also it's password `aoc2020`. After that, we need to type `sudo -il` enter `sudo -l` and followed by password. Then type `ls`, enter and all we need based on the youtube channel. We did in kali to get the command to host a http server using python3 on port 9999. Then in order to to get flag we need to type `find / -perm -4000 2>/dev/null`. Then type some command. After that at last we need to type `cat /root/flag.txt` then we will get answer for flag.

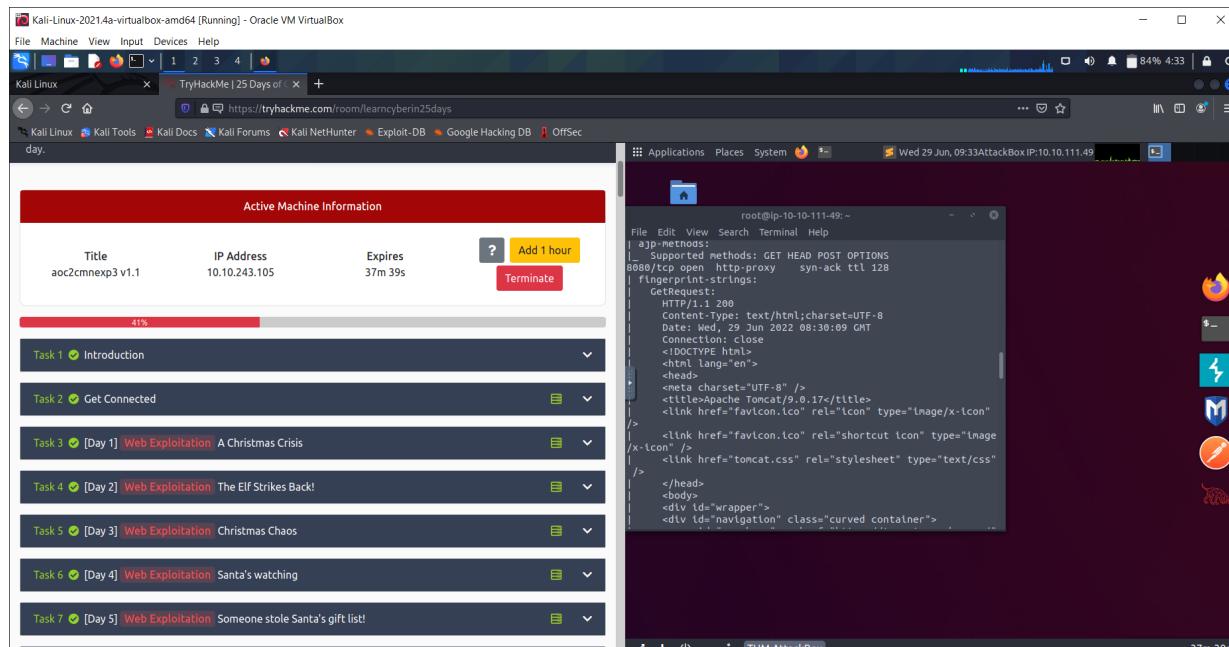
Day 12 - Networking Ready, set, elf

Tools used: google, attack box, Firefox,

Solution/walkthrough:

Question 1:

We found the version number of the web server .



Question 2:

If we google Apache Tomcat 9.0.17 we can find a page called CVE Details which talks about remote code execution.

Apache » Tomcat » 9.0.17 : Security Vulnerabilities Published in 2019

Cve Name: Apache tomcat 9.0.17
Jan February March April May June July August September October November December CVSS Scores Greater Than: 0 1 2 3 4 5 6 7 8 9
Sort Results By: CVE Number Ascending CVE Number Descending CVSS Score Ascending Number Of Exploits Descending

#	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Published Date	Updated Date	Score	Gained Access Level	Access	Complexity	Authentication	Conf.	Integ.	Avail.
1	CVE-2019-10072	400	0	None	2019-06-21	2019-06-25	5.6	None	Remote	Low	Not required	None	None	Partial
2	CVE-2019-0320	20	0	None	2019-04-15	2019-06-01	8.9	None	Remote	Medium	Not required	Complete	Complete	Complete
3	CVE-2019-0221	29	0	XSS	2019-05-28	2019-06-07	4.3	None	Remote	Medium	Not required	None	Partial	None

The first entry, CVE-2019-10072, is described as follows:
 Apache Tomcat versions 9.0.0.M1 to 9.0.19 and 8.5.0 to 8.5.40 were incomplete and did not address HTTP/2 connection window reduction on write in Apache Tomcat versions 9.0.0.M1 to 9.0.19 and 8.5.0 to 8.5.40. By not sending WINDOW_UPDATE messages for the connection window (stream 0) clients were able to cause server-side threads to block eventually leading to thread exhaustion and a DoS.

The second entry, CVE-2019-0320, is described as follows:
 When running on Windows with enableCmdLineArguments enabled, the CGI Server 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 Server is disabled by default. The CGI option enableCmdLineArguments is disabled by default in Tomcat 9.0. (and will be disabled by default in all versions in response to this vulnerability). For a detailed description of the bug, see Microsoft Knowledge Base Article 3122394. This issue was addressed by improved MSN log (https://technet.microsoft.com/en-us/library/bb732644.aspx).

The third entry, CVE-2019-0221, is described as follows:
 The SSI printenv command in Apache Tomcat 9.0.0.M1 to 9.0.0.17, 8.5.0 to 8.5.39 and 7.0.0 to 7.0.93 echoes user provided data without escaping and is, therefore, vulnerable to XSS. SSI is disabled by default. The printenv command is intended for debugging and is unlikely to be present in a production website.

Total number of vulnerabilities: 3 Page: 1 (This Page)

Question 3 :

We must run the shell to drop into a shell and find the flag, which is in the same directory you landed in C:\Program Files\Apache Software Foundation\Tomcat 9.0\webapps\ROOT\WEB-INF\cgi-bin\. We must run type flag.txt to get the flag.

```
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>
```

Question 4 :

We must set the Metasploit settings.

Thought Process/Methodology:

Firstly, we need to open up the terminal and put in the IP Address that we will be attacking. We have to do nmap to get the common scripts. After getting the scripts we can get the answer for the first question by scrolling through . We are supposed to find the port number and then type out exploit-db.com in the firefox to find the CVE number. After getting the CVE number we are supposed to put it in the terminal as msfconsole -q following the CVE number. we can go ahead and start configuring our settings. After that we have to type <http://10.0.0.1/cgi-bin/systeminfo.sh> with our ip address and get the information needed. We must run the shell to drop into a shell and find the flag, which is in the same directory you landed in C:\Program Files\Apache Software Foundation\Tomcat 9.0\webapps\ROOT\WEB-INF\cgi-bin\. We must run type flag.txt to get the flag.

Day 13 - Coal for Christmas

Tools used: google, attack box, Firefox,

Solution/walkthrough:

Question 1:

We google searched about the old, deprecated protocol and service is running which is Telnet.

A screenshot of a Google search results page for the query "telnet". The top result is the Wikipedia page for Telnet. Below it, there's a snippet of a Telnet session showing a command-line interface with a red arrow pointing to the "Enter host-name: >" prompt. To the right of the search results, there's a thumbnail image of the Telnet session and a summary box.

Question 2:

We typed in the command Telnet_MACHINE IP and we got the user name and also the password which is “clauschristmas”.

The image shows the AttackBox interface. On the left, there's a sidebar with sections like Port Scanning, Initial Access, and Enumeration. In the main area, there's a terminal window showing the output of an nmap scan for port 132 on 10.10.46.132. Below it, a terminal window shows a successful Telnet connection to the same host. The user has typed "HI SANTA!!!" and received a response about leaving cookies and milk. The Enumeration section notes that files can be viewed with cat.

Port Scanning

We will begin by scanning the machine. If you are working from the TryHackMe "Attackbox" or from a Kali Linux instance (or honestly, any Linux distribution where you have this installed), you can use nmap with syntax like so:

```
nmap 10.10.46.132
```

No answer needed

Question Done

What old, deprecated protocol and service is running?

telnet

Correct Answer

Initial Access

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 10.10.46.132 <PORT_FROM_NMAP_SCAN>
```

What credential was left for you?

clauschristmas

Correct Answer

Hint

Enumeration

Looks like you can slide right down the chimney! Log in and take a look around, enumerate a bit. You can view files and Folders in the current directory with ls, change directories with cd and view the contents of files with cat.

```
root@ip-10-10-56-18:~/aoc_day13
root@ip-10-10-56-18:~/aoc_day13# ssh santa@10.10.46.132
santa@10.10.46.132's password:
          _\ \
         /o \
        / \ \
       /_ \ \
      / \ \ \ \
     / \ \ \ \ \
    / \ \ \ \ \ \
   / \ \ \ \ \ \ \
  / \ \ \ \ \ \ \ \
 / \ \ \ \ \ \ \ \ \
/_\ \ \ \ \ \ \ \ \
[___]
```

Last login: Sat Nov 21 20:37:37 2020 from 10.0.2.2

\$

Question 3:

We typed in the command “cat /etc/*release” to get the distribution of Linux and version number is this server running which is “Ubuntu 12.04”.

What credential was left for you?

clauschristmas

Correct Answer

Hint

Enumeration

Looks like you can slide right down the chimney! Log in and take a look around, enumerate a bit. You can view files and folders in the current directory with ls, change directories with cd and view the contents of files with cat.

Often to enumerate you want to look at pertinent system information, like the version of the operating system or other release information. You can view some information with commands like this:

```
cat /etc/*release
```

```
uname -a
```

```
cat /etc/issue
```

There is a great list of commands you can run for enumeration here: <https://blog.gotmi1k.com/2011/08/basic-linux-privilege-escalation/>

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

Ubuntu 12.04

Correct Answer

This is a very old/version of Linux! This may be vulnerable to some kernel exploits, that we could use to escalate our privileges.

Take a look at the cookies and milk that the server owners left for you. You can do this with the cat command as mentioned earlier.

```
root@ip-10-10-56-18:~/aoc_day13
root@ip-10-10-56-18:~/aoc_day13# cat /etc/*release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=12.04
DISTRIB_CODENAME=precise
DISTRIB_DESCRIPTION="Ubuntu 12.04 LTS"
root@ip-10-10-56-18:~/aoc_day13# uname -a
Linux christmas 3.2.0-23-generic #36-Ubuntu SMP Tue Apr 10 20:39:51 UTC 2012 x86_64 x86_64 GNU/Linux
root@ip-10-10-56-18:~/aoc_day13# cat /etc/issue
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!

\$

Question 4:

We typed in the command “cat cookies_and_milk.txt” and ran the command and we received a message saying that The Grinch got there before Santa did.

cat /etc/issue

There is a great list of commands you can run for enumeration here: <https://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation/>

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

Ubuntu 12.04 Correct Answer ? Hint

This is a very old version of Linux! This may be vulnerable to some kernel exploits, that we could use to escalate our privileges.

Take a look at the cookies and milk that the server owners left for you. You can do this with the cat command as mentioned earlier.

cat cookies_and_milk.txt

Who got here first?

grinch Correct Answer ? Hint

The perpetrator took half of the cookies and milk! Weirdly enough, that file looks like C code...

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.

You can learn more about the DirtyCow exploit online here: <https://dirtycow.ninja/>

This cookies_and_milk.txt file looks like a modified rendition of a DirtyCow exploit, usually

```

root@ip-10-10-56-18:~/aoc_day13
bash < "cat /tmp/thmp.txt"

root@ip-10-10-56-18:~/aoc_day13
root@ip-10-10-56-18:~/aoc_day13
root@ip-10-10-56-18:~/aoc_day13

    exit(ret);

}

struct Userinfo user;
// set values, change as needed
user.username = "grinch";
user.user_id = 0;
user.group_id = 0;
user.info = "pwned";
user.home_dir = "/root";
user.shell = "/bin/bash";

}

*****+
// HAH! Too bad Santa I, the Grinch, got here
// before you did! I helped myself to some of
// the cookies 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
*****+
S

```

Question 5:

We pasted (char generate_password_hash(char plaintext_pw) {}) in google and we searched up for the real C source and got the verbatim syntax we can use to compile.

cat /etc/issue

There is a great list of commands you can run for enumeration here: <https://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation/>

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

Ubuntu 12.04 Correct Answer ? Hint

This is a very old version of Linux! This may be vulnerable to some kernel exploits, that we could use to escalate our privileges.

Take a look at the cookies and milk that the server owners left for you. You can do this with the cat command as mentioned earlier.

cat cookies_and_milk.txt

Who got here first?

grinch Correct Answer ? Hint

The perpetrator took half of the cookies and milk! Weirdly enough, that file looks like C code...

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.

You can learn more about the DirtyCow exploit online here: <https://dirtycow.ninja/>

This cookies_and_milk.txt file looks like a modified rendition of a DirtyCow exploit, usually

```

root@ip-10-10-56-18:~/aoc_day13
bash < "cat /tmp/thmp.txt"

root@ip-10-10-56-18:~/aoc_day13
root@ip-10-10-56-18:~/aoc_day13
root@ip-10-10-56-18:~/aoc_day13

pid_t pid;
pthread_t pth;
struct stat st;

struct Userinfo {
    char *username;
    char *hash;
    int user_id;
    int group_id;
    char *info;
    char *home_dir;
    char *shell;
};

char *generate_password_hash(char *plaintext_pw) {
    return crypt(plaintext_pw, salt);
}

char *generate_passwd_line(struct Userinfo u) {
    const char *format = "%s:%s:%d:%d:%s:%s\n";
    int size = snprintf(NULL, 0, format, u.username, u.hash,
        u.user_id, u.group_id, u.info, u.home_dir, u.shell);
    char *ret = malloc(size + 1);
    sprintf(ret, format, u.username, u.hash, u.user_id,

```

```

// This exploit uses the pokemon exploit of the dirtycow vulnerability
// as a base and automatically generates a new passwd line.
// The user will be prompted for the new password when the binary is run.
// The original /etc/passwd file is then backed up to /tmp/passwd.bak
// and overwrite the root account with the generated line.
// After running the exploit you should be able to login with the newly
// created user.
//
// To use this exploit modify the user values according to your needs.
// The default is "firefart".
//
// Original exploit (dirtycow's ptrace_pokedata "pokemon" method):
// https://github.com/dirtycow/dirtycow.github.io/blob/master/pokemon.c
//
// Compile with:
// gcc -pthread dirty.c -o dirty -lcrypt
//
// Then run the newly create binary by either doing:
// "./dirty" or "./dirty my-new-password"
//
// Afterwards, you can either "su firefart" or "ssh firefart@..."
//
// DON'T FORGET TO RESTORE YOUR /etc/passwd AFTER RUNNING THE EXPLOIT!
// mv /tmp/passwd.bak /etc/passwd
//
// Exploit adopted by Christian "FireFart" Mehlmauer
// https://firefart.at
//

```

Question 6:

We typed in “gcc -pthread dirty.c -o dirty -lcrypt” under the file name “nano dirty.c” and ran the command and we got the new username which is “firefart”.

parameters or arguments to include different updates, but importantly, the DirtyCow source code will explain what syntax to use.

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`

Correct Answer 💡 Hint

Privilege Escalation

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`

Correct Answer

Switch your user into that new user account, and hop over to the /root directory to own this server!

You can switch user accounts like so:

`su <user_to_change_to>`

No answer needed Question Done

Uh oh, looks like that perpetrator left a message! Follow his instructions to prove you really did leave Coal for Christmas!

After you leave behind the coal, you can run `tree | md5sum`

What is the MD5 hash output?

`8b16f00dd3b51efadb02c1df7f8427cc`

Correct Answer 💡 Hint

Question 7:

We typed in “tree | md5sum” and we got the MD5 hash output.

Switch your user into that new user account, and hop over to the /root directory to own this server!

You can switch user accounts like so:

`su <user_to_change_to>`

No answer needed Question Done

Uh oh, looks like that perpetrator left a message! Follow his instructions to prove you really did leave Coal for Christmas!

After you leave behind the coal, you can run `tree | md5sum`

What is the MD5 hash output?

`8b16f00dd3b51efadb02c1df7f8427cc`

Correct Answer 💡 Hint

Task 16 [Day 14] OSINT Where's Rudolph?

Task 17 [Day 15] Scripting There's a Python in my stocking!

Task 18 [Day 16] Scripting Help! Where is Santa?

Task 19 [Day 17] Reverse Engineering ReverseELFneering

Question 8:

We searched in the THM text and we found that (CVE-2016-5195) is the CVE for Dirty COW.

The perpetrator took half of the cookies and milk! Weirdly enough, that file looks like C code...

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.

You can learn more about the DirtyCow exploit online here: <https://dirtycow.ninja/>

This **cookies_and_milk.txt** file looks like a modified rendition of a DirtyCow exploit, usually written in C. Find a copy of that original file online, and get it on the target box. You can do this with some simple file transfer methods like netcat, or spinning up a quick Python HTTP server... or you can simply copy-and-paste it into a text editor on the box!

Thought Process/Methodology:

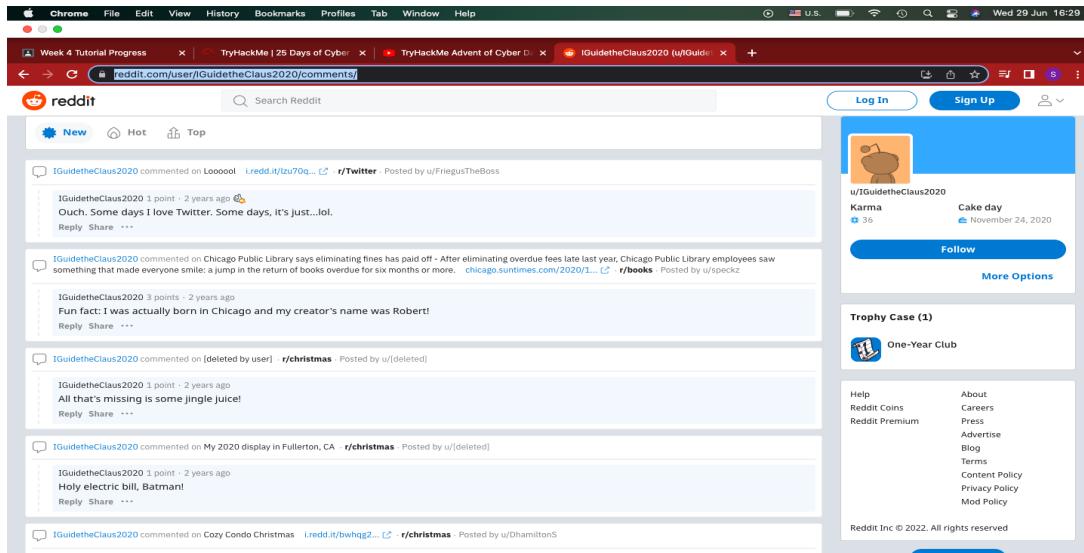
Firstly, we started our machine and attacked box and got our IP addresses. We google searched about SSH and Telnet about their details. We used the terminal to scan our machine by using nmap MACHINE_IP and ran it. After that we created a new tab, we entered the command Telnet MACHINE_IP so that we get the username and the password. After getting the username and password, we then opened a new tab and we typed in the command “ssh@santa MACHINE_IP” and we entered santa’s password. Then, we typed in the command “cat /etc/*release” to get the distribution of Linux and version number is this server running which was Ubuntu 12.04 . We also used the command “cat cookies_and_milk.txt” and received a message from The Grinch to Santa saying that he got there before Santa did. We explored the DirtyCow website. We scrolled up and we copied (char generate_password_hash(char plaintext_pw) {}) and pasted it in google to get the real C source and we copied the sources in the website and pasted it under “nano dirty.c” file and we saved the file in the terminal. Then, we typed in “gcc -pthread dirty.c -o dirty -lcrypt” under “nano dirty.c” file and got our new username and also the new password that we created. We also typed in “touch coal” and “tree” and got 3 files and piped them into “md5sum” and we got the MD5 hash output. Last but not least, We searched in the THM text and we found that (CVE-2016-5195) is the CVE for Dirty COW and terminated our machines.

Day 14 - Where's Rudolph?

Tools used: Google
Solution/walkthrough:

Question 1:

First, we should search reddit and find Rudolph's reddit page and then open the comment section while copying the url link above.



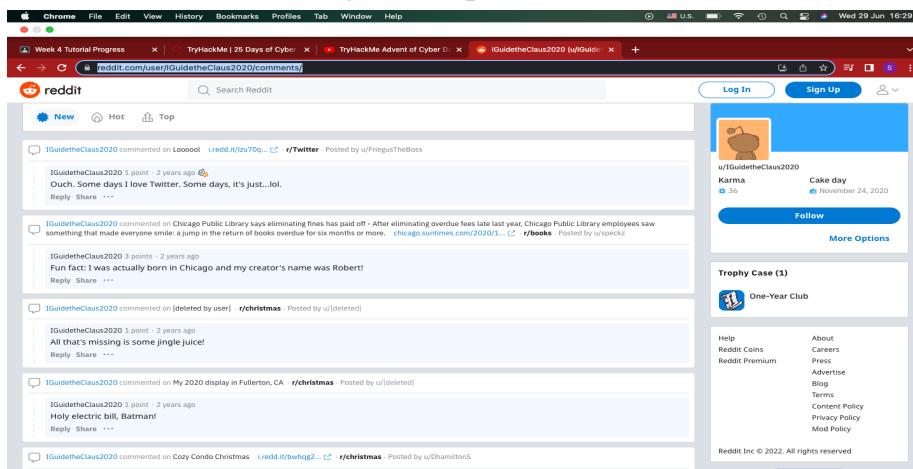
The screenshot shows a web browser window with multiple tabs open. The active tab is a reddit thread from user [u/IguideTheClaus2020](#). The comments section contains several posts:

- [u/IguideTheClaus2020](#) commented on [Loooloo](#): Ouch. Some days I love Twitter. Some days, it's just...lol.
- [u/IguideTheClaus2020](#) commented on [Chicago Public Library](#): says eliminating fines has paid off - After eliminating overdue fees late last year, Chicago Public Library employees saw something that made everyone smile: a jump in the return of books overdue for six months or more.
- [u/IguideTheClaus2020](#) commented on [My 2020 display in Fullerton, CA](#): Fun fact: I was actually born in Chicago and my creator's name was Robert!
- [u/IguideTheClaus2020](#) commented on [Cozy Condo Christmas](#): All that's missing is some jingle juice!
- [u/IguideTheClaus2020](#) commented on [Loooloo](#): Holy electric bill, Batman!
- [u/IguideTheClaus2020](#) commented on [Loooloo](#): Ouch. Some days I love Twitter. Some days, it's just...lol.
- [u/IguideTheClaus2020](#) commented on [Chicago Public Library](#): says eliminating fines has paid off - After eliminating overdue fees late last year, Chicago Public Library employees saw something that made everyone smile: a jump in the return of books overdue for six months or more.
- [u/IguideTheClaus2020](#) commented on [My 2020 display in Fullerton, CA](#): Fun fact: I was actually born in Chicago and my creator's name was Robert!
- [u/IguideTheClaus2020](#) commented on [Cozy Condo Christmas](#): All that's missing is some jingle juice!

The right sidebar shows the user profile for [u/IguideTheClaus2020](#), which includes a picture of a reindeer, a karma count of 36, and a "Cake day" on November 24, 2020. It also displays a trophy case with one achievement and links to Reddit's help center and legal policies.

Question 2:

We should scroll through rudolph's comment section and find out where he born.



This screenshot shows the same reddit thread as the previous one, but the comments section is empty. This likely indicates that the user has scrolled through the entire comment history for that post.

Question 3:

We should search for Rudolph the red reindeer last name robert in google in order to get Roberts's last name.

A screenshot of a web browser window showing search results for "rudolph the red nosed reindeer last name". The results page includes a snippet from the Chicago Tribune about Rudolph the Red-Nosed Reindeer's creation by Robert L. May, and a link to the Wikipedia page for Rudolph the Red-Nosed Reindeer.

A screenshot of a Google search results page for "rudolph the red nosed reindeer last name". It shows a snippet from the Chicago Tribune and a link to the Wikipedia page. Below the snippet, there is a "People also ask" section with several questions and their answers.

A screenshot of the Wikipedia page for Rudolph the Red-Nosed Reindeer. It includes a summary, publication history, and a section on the TV special.

Question 4:

We should scroll through Rudolph's comment section and find out what other social platforms Rudolph uses.

A screenshot of a Reddit comments section for a post by u/GuidetheClaus2020. The comments include various users discussing their favorite social media platforms, such as Twitter and Facebook.

Question 5:

We should search rudolph's twitter id using his reddit id and find his username.

Question 6:

We should scroll through rudolph's twitter feed and find out what is his favourite tv show.

Question 7,8, and 9:

We should use exif data retriever and find out the location and exact coordinate. Then we can receive the flag that was given.

YCbCr Positioning center of pixel array

Copyright [FLAG]ALWAYSCHECKTHEEXIFD4T4

Exif Version 0231

Components Configuration YCbCr

Flashpix Version 0100

GPS Latitude Ref N

GPS Latitude 41.89181510005266

GPS Longitude Ref W

GPS Longitude -87.62427730000896

Question 11:

Paste the coordinate in maps and find out which hotel he stayed. Then, we can find out the street number of the hotel.

Chicago Marriott Downtown Magnificent Mile
4.3 ★★★★☆ 2,863 reviews • 4-star hotel

RM 1,069 Aug 29 – 30

Directions Save Nearby Send to phone Share

CHECK AVAILABILITY

Compare prices Free cancellation only

Check in / Check out Mon, Aug 29 Tue, Aug 30 2

Ads • Featured options

- Expedia.com.my RM 1,625 Interactive Map • Instant Confirmation Free cancellation until 26 Aug
- Hotels.com RM 1,625 Free cancellation until 26 Aug
- Chirann Marriott Downtown RM 1,477

Thought Process/Methodology:

Firstly, we should read the instructions given and we should search reddit and find Rudolph's reddit page and then open the comment section while copying the url link above. Later on, after opening Rudolph's reddit, we should scroll through Rudolph's comment section and find out where he was born. Next, we should search for Rudolph the red reindeer last name robert in google in

order to get Roberts's last name which is May. Next, we should scroll through Rudolph's comment section and find out what other social platforms Rudolph uses, such as twitter. Later, we should search rudolph's twitter id using his reddit id and find his username which is @iguidetclaus2020. After scrolling through his twitter feed, we will be able to find his favourite show, which is bachorollate and also gain information about where he was paraded in. Next, we should use exif data retriever and find out the location and exact coordinate. Then we can receive the flag that was given. Lastly, we should paste the coordinates in maps and find out which hotel he stayed. Then, we can find out the street number of the hotel.

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

Tools used: tryhackme , Vscode

Solution/walkthrough:

Question 1:

The first question asks us to find the output of True + True. We got this answer by reading the information given.

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

Question 2:

The next question asks us the name of the database for installing other people's libraries. We found this by simply reading the challenge description. The name of this database is PyPi.

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
- 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")
# lxml is just the parser for reading the html

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

Question 3:

The next question asks us the output of `bool("False")`. The output is True. We are actually passing in a String containing the word False.

Question 4:

The next question asks us what library can be used to download the HTML of a webpage. We can find it by reading the description.

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
- 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")
# lxml is just the parser for reading the html

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

Question 5:

The next question asks us to analyse some code which uses the append function and some interesting variables. When we run through the code, we see that the output is [1, 2, 3, 6]. This is because the append functions as adding that input inside the y.

A screenshot of Visual Studio Code showing a Python script named 'Untitled-1.py'. The code contains three lines: 'y=[1, 2, 3]', 'y.append(6)', and 'print(y)'. The terminal below shows the output: '[1, 2, 3, 6]'. The status bar at the bottom indicates the file is 3.9.6.64-bit and was last modified on 30/6/2022 at 11:30 PM.

```
File Edit Selection View Go Run Terminal Help
Untitled-1.py - Visual Studio Code
D:\User's Files > Desktop > christmas > Untitled-1.py > (e)y
1 y=[1, 2, 3]
2 y.append(6)
3 print(y)

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6
PS D:\User's Files\Desktop\christmas> & 'c:\Users\Acer\AppData\Local\Programs\Python\Python39\python.exe' 'c:\Users\Acer\.vscode\extensions\ms-python.python-2022.8.1\pythonFiles\lib\python\debugpy\launcher' '5712' '--' 'd:\User's Files\Desktop\christmas\Untitled-1.py'
[1, 2, 3, 6]
PS D:\User's Files\Desktop\christmas>

Ln 1, Col 1 Spaces: 4 CRLF Python 3.9.6.64-bit 11:30 PM 30/6/2022
```

Question 6:

The next question asks us the reason why the output is such. This can happen because of pass by reference. With pass by reference, when we pass a variable into a function, we are passing a reference to the location in memory where the variable is stored.

A screenshot of a web page from tryhackme.com. It discusses the concept of pass by reference in Python. It explains that when a variable is passed to a function, a reference to the original variable is passed, not a copy. This can lead to unexpected results if the function modifies the variable. The page includes a code example and a section on operators.

tryhackme.com/room/learnpyin2days

- Integer - a whole number (50, 50, 91)
- Float - a floating-point number (21.3, -5.1921)
- List - a list of items ([1, 2, 3], ["hi", 6, 7.9])

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.

Operators

Let's talk about operators. An operator is something between 2 variables/values and does something to them. For example, the addition operator:

```
x = 3 + 1 # x = 4
```

Python supports many math operators:

```
3 + 1  
3 / 3 # divided by  
3 * 4 # times  
2 ** 2 # 2 to the power of 2  
2 % 3 # 2 % mod 3
```

Question 7:

This question asks us the output if the input is “Skiddy”. The output will be ``The Wise One has allowed you to come in.” because it's in the names list.

```
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 8:

This question asks us the output if the input is “elf”. The output will be ``The Wise One has not allowed you to come in.” because it's in the names list but with a capital ‘E’.

```
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.")
```

Thought Process/Methodology:

Firstly, we read through the instructions and the information given. It was easy to find the answers because all the answers were in the description. We just have to look through .