PSP0201

WEEK 5

WRITEUP

Group name: Lion Force
Members:

ID	Name	Role
1211103128	Muhammad Ajwad bin Mohamad A'sim	Leader
1211102001	Muhammad Naqiuddin bin Edy Azhar	Member
1211103183	Nurul Nabilah binti Mohd Noor Hakim	Member
1211103165	Izzati Alia binti Incik Ghazali	Member

Table of Content

Day 16 - Scripting - Help! Where is Santa? 2

Day 17 - Reverse Engineering - ReverseELFneering 6

Day 18 - Reverse Engineering – The Bits of Christmas 9

Day 19 - Web Exploitation – The Naughty or Nice List 13

Day 20 - Blue Teaming - PowershELIF to the rescue 18

Day 16 - Scripting - Help! Where is Santa?

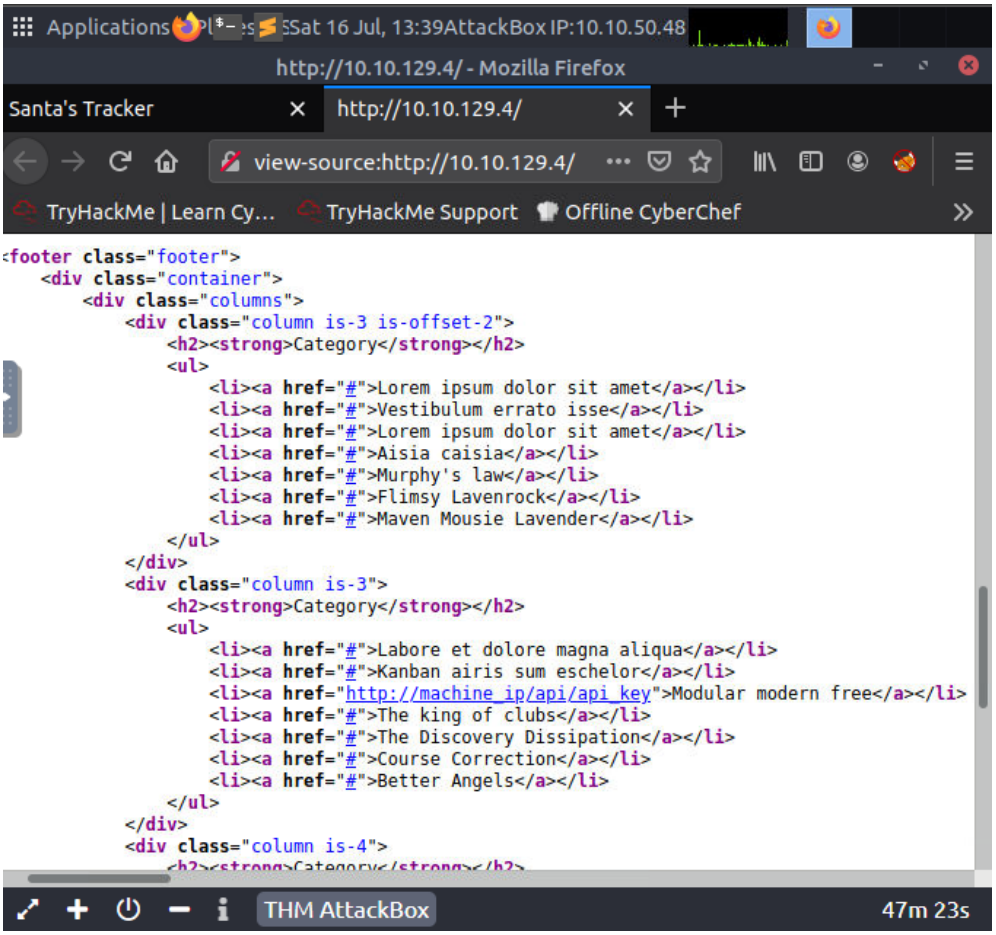
Tools Used: *Terminal Emulator, Text Editor(For Python), Python3*

Walkthrough

What is the port number for the webserver

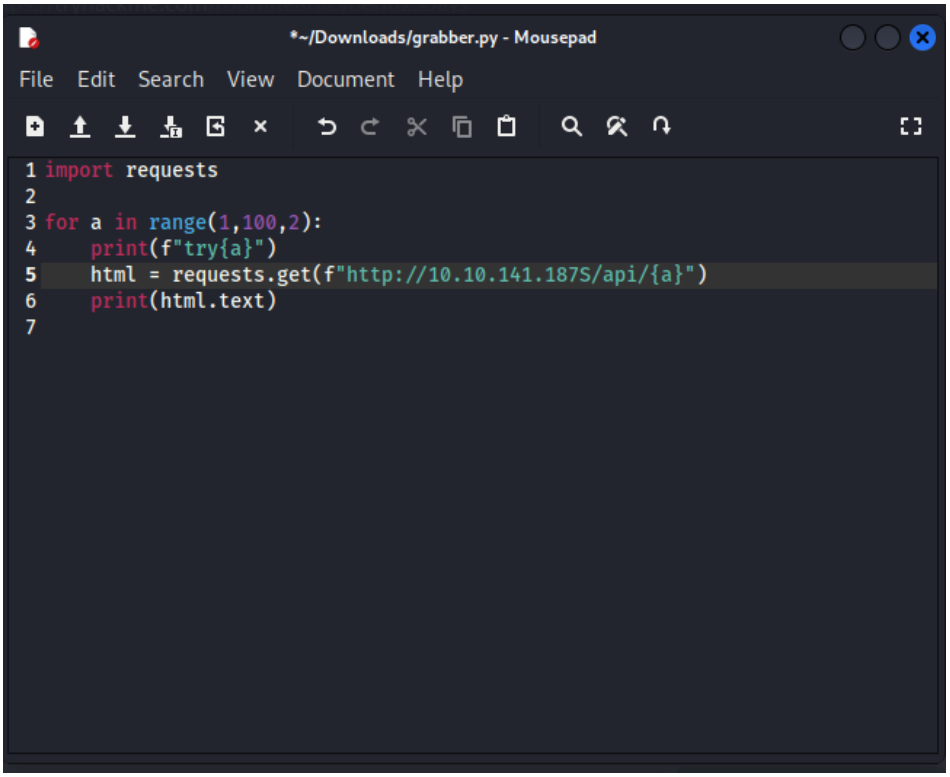
```
root@ip-10-10-50-48: ~
File  Edit  View  Search  Terminal  Help
root@ip-10-10-50-48:~# nmap 10.10.129.4

Starting Nmap 7.60 ( https://nmap.org ) at 2022-07-16 13:30 BST
Nmap scan report for ip-10-10-129-4.eu-west-1.compute.internal (10.10.129.
4)
Host is up (0.070s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
MAC Address: 02:7D:06:C0:5B:BD (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.99 seconds
root@ip-10-10-50-48:~#
```

Without using enumerations tools such as Dirbuster, what is the directory for the API? (without the API key)




```
<footer class="footer">
    <div class="container">
        <div class="columns">
            <div class="column is-3 is-offset-2">
                <h2><strong>Category</strong></h2>
                <ul>
                    <li><a href="#">Lorem ipsum dolor sit amet</a></li>
                    <li><a href="#">Vestibulum errato isse</a></li>
                    <li><a href="#">Lorem ipsum dolor sit amet</a></li>
                    <li><a href="#">Aisia caisia</a></li>
                    <li><a href="#">Murphy's law</a></li>
                    <li><a href="#">Flimsy Lavenrock</a></li>
                    <li><a href="#">Maven Mousie Lavender</a></li>
                </ul>
            </div>
            <div class="column is-3">
                <h2><strong>Category</strong></h2>
                <ul>
                    <li><a href="#">Labore et dolore magna aliqua</a></li>
                    <li><a href="#">Kanban airis sum eschelor</a></li>
                    <li><a href="http://machine_ip/api/api_key">Modular modern free</a></li>
                    <li><a href="#">The king of clubs</a></li>
                    <li><a href="#">The Discovery Dissipation</a></li>
                    <li><a href="#">Course Correction</a></li>
                    <li><a href="#">Better Angels</a></li>
                </ul>
            </div>
            <div class="column is-4">
                <h2><strong>Category</strong></h2>
```

Where is Santa right now?
&
Find out the correct API key. Remember, this is an odd number between 0-100. After too many attempts, Santa's Sled will block you.



```
1211102001@kali: ~/Downloads
1211102001@kali: ~ ×   1211102001@kali: ~/Downloads ×   1211102001@kali: ~ ×

┌──(1211102001㉿kali)-[~/Downloads]
└─$ python3 grabber.py
try1
{"item_id":1,"q":"Error. Key not valid!"}
try3
{"item_id":3,"q":"Error. Key not valid!"}
try5
{"item_id":5,"q":"Error. Key not valid!"}
try7
{"item_id":7,"q":"Error. Key not valid!"}
try9
{"item_id":9,"q":"Error. Key not valid!"}
try11
{"item_id":11,"q":"Error. Key not valid!"}
try13
{"item_id":13,"q":"Error. Key not valid!"}
try15
{"item_id":15,"q":"Error. Key not valid!"}
try17
{"item_id":17,"q":"Error. Key not valid!"}
try19
{"item_id":19,"q":"Error. Key not valid!"}
try21
{"item_id":21,"q":"Error. Key not valid!"}
try23
{"item_id":23,"q":"Error. Key not valid!"}
try25
{"item_id":25,"q":"Error. Key not valid!"}
try27
{"item_id":27,"q":"Error. Key not valid!"}
try29
{"item_id":29,"q":"Error. Key not valid!"}
try31
{"item_id":31,"q":"Error. Key not valid!"}
```

```
{"item_id":39,"q":"Error. Key not valid!"}
try41
{"item_id":41,"q":"Error. Key not valid!"}
try43
{"item_id":43,"q":"Error. Key not valid!"}
try45
{"item_id":45,"q":"Error. Key not valid!"}
try47
{"item_id":47,"q":"Error. Key not valid!"}
try49
{"item_id":49,"q":"Error. Key not valid!"}
try51
{"item_id":51,"q":"Error. Key not valid!"}
try53
{"item_id":53,"q":"Error. Key not valid!"}
try55
{"item_id":55,"q":"Error. Key not valid!"}
try57
{"item_id":57,"q":"Winter Wonderland, Hyde Park, London."}
try59
{"item_id":59,"q":"Error. Key not valid!"}
try61
```

Thought Process

Used nmap to scan available ports on the machine's ip, then proceed to go to machines ip with http port. There we right click and view the page's source code to find the api with api key. Then later we referred to day 15 to import the request module, then later begin to write a script to find the api key in the process of trial and error, then after the api key is found, we could also see the location where santa will be.

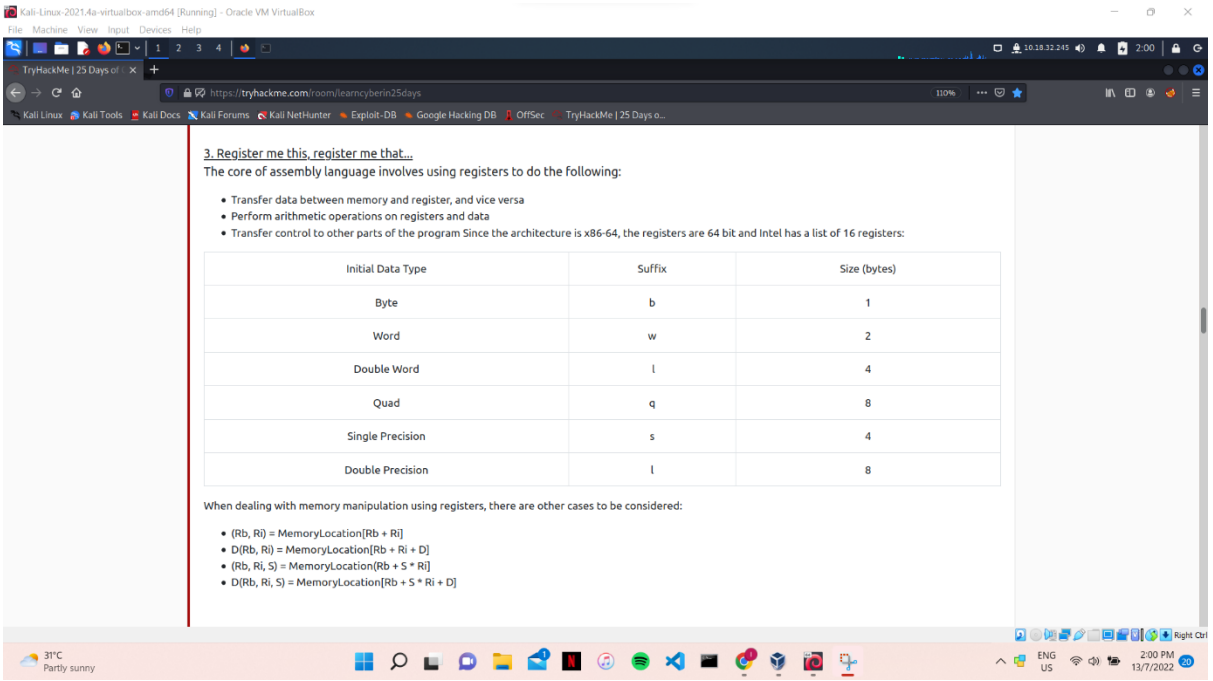
Day 17 - Reverse Engineering - ReverseELFneering

Tools used: Kali Linux, Firefox, Radare2

Solution:

Question 1

Read through the explanation under Day 17's task. Answer is provided in the table form.



3. Register me this, register me that...

The core of assembly language involves using registers to do the following:

- Transfer data between memory and register, and vice versa
- Perform arithmetic operations on registers and data
- Transfer control to other parts of the program Since the architecture is x86-64, the registers are 64 bit and Intel has a list of 16 registers:

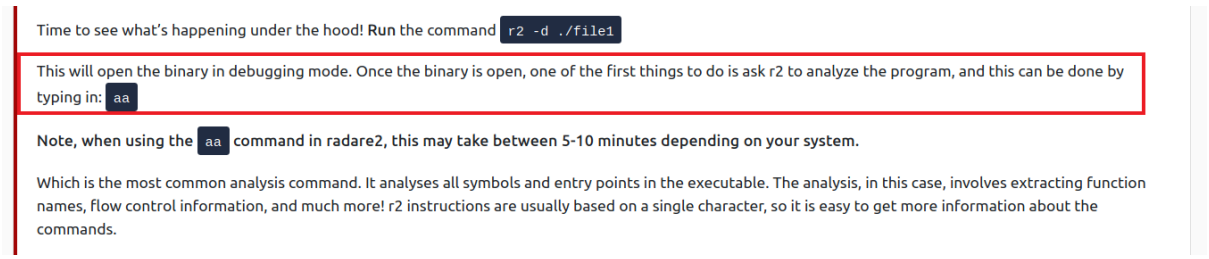
Initial Data Type	Suffix	Size (bytes)
Byte	b	1
Word	w	2
Double Word	l	4
Quad	q	8
Single Precision	s	4
Double Precision	l	8

When dealing with memory manipulation using registers, there are other cases to be considered:

- (Rb, Ri) = MemoryLocation[Rb + Ri]
- D(Rb, Ri) = MemoryLocation[Rb + Ri + D]
- (Rb, Ri, S) = MemoryLocation(Rb + S * Ri]
- D(Rb, Ri, S) = MemoryLocation[Rb + S * Ri + D]

Question 2

Answer has also been provided in the explanation.



Time to see what's happening under the hood! **Run** the command `r2 -d ./file1`

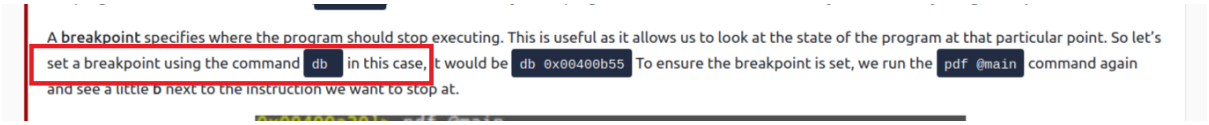
This will open the binary in debugging mode. Once the binary is open, one of the first things to do is ask r2 to analyze the program, and this can be done by typing in: `aa`

Note, when using the `aa` command in radare2, this may take between 5-10 minutes depending on your system.

Which is the most common analysis command. It analyses all symbols and entry points in the executable. The analysis, in this case, involves extracting function names, flow control information, and much more! r2 instructions are usually based on a single character, so it is easy to get more information about the commands.

Question 3

Answer can be obtained in the explanation.



A breakpoint specifies where the program should stop executing. This is useful as it allows us to look at the state of the program at that particular point. So let's set a breakpoint using the command `db` in this case, it would be `db 0x00400b55` To ensure the breakpoint is set, we run the `pdf @main` command again and see a little b next to the instruction we want to stop at.

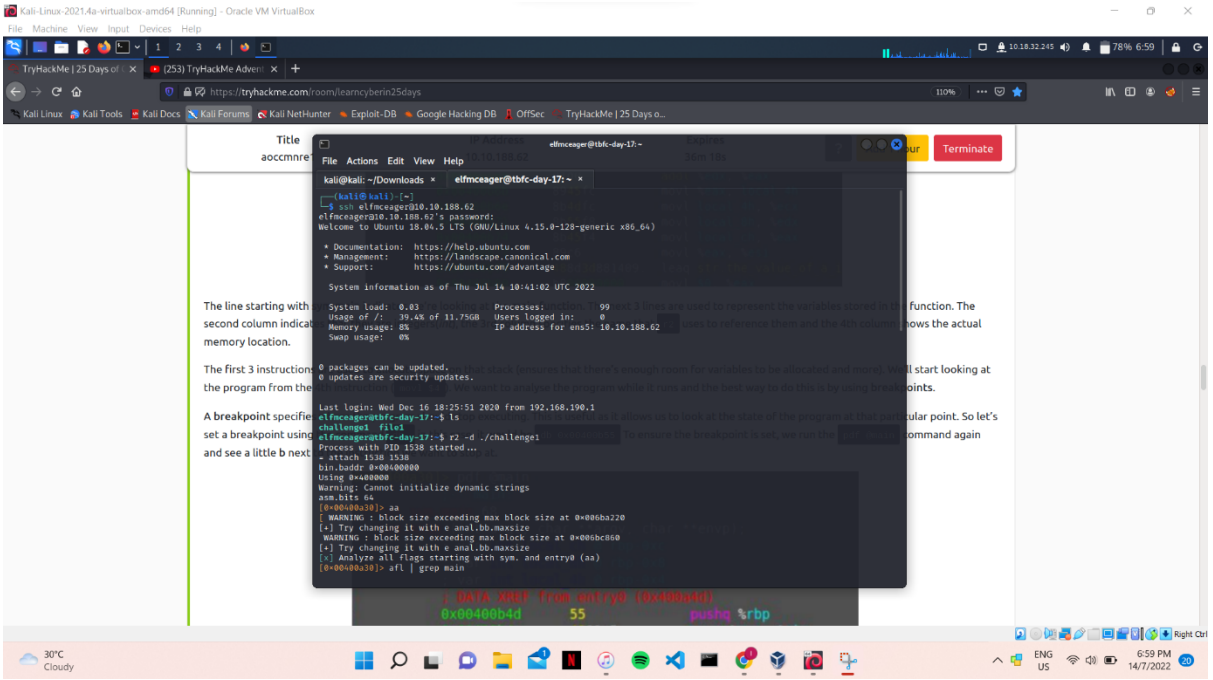
Question 4

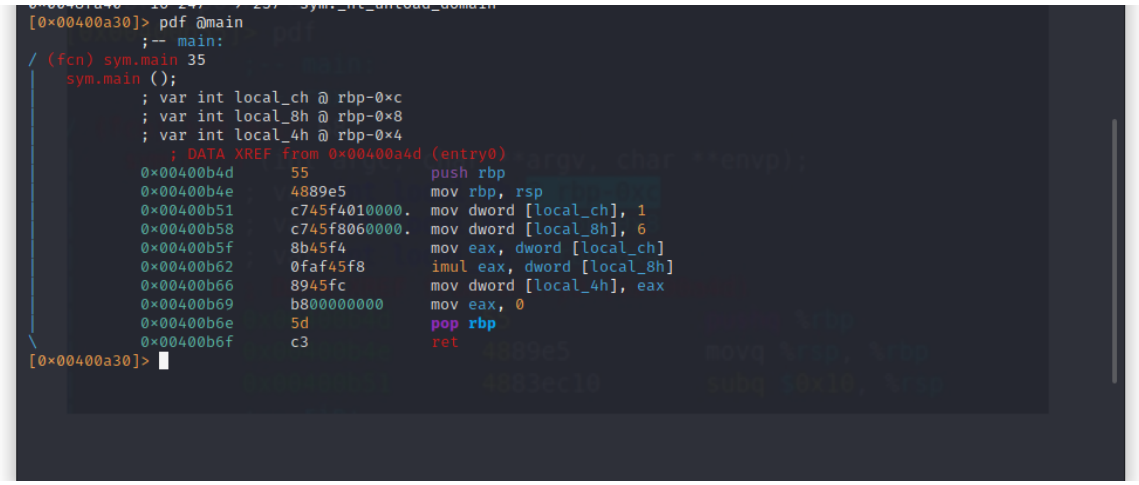
Obtain the answer from the explanation.

Running `dc` will execute the program until we hit the breakpoint. Once we hit the breakpoint and print out the main function, the rip which is the current instruction shows where execution has stopped. From the notes above, we know that the **mov** instruction is used to transfer values. This statement is transferring the value 4 into the `local_ch` variable. To view the contents of the `local_ch` variable, we use the following instruction `px @memory-address` In this case, the corresponding memory address for `local_ch` will be `rbp-0xc` (from the first few lines of **@pdf main**) This instruction prints the values of memory in hex:

Question 5

Deploy the machine and log into the instance by using IP address given. Then, run **radare2** to analyse the **challenge1** file. View the assembly code and obtain the answer.





Question 6

Answer can be obtained in the same assembly code as in question 5.

```
[0x00400a30]> pdf @main
            ;-- main:
/ (fcn) sym.main 35
|   sym.main ();
|           ; var int local_ch @ rbp-0x c
|           ; var int local_8h @ rbp-0x 8
|           ; var int local_4h @ rbp-0x 4
|             ; DATA XREF from 0x00400a4d (entry0)
|           0x00400b4d      55             push rbp
|           0x00400b4e      4889e5         mov rbp, rsp
|           0x00400b51      c745f4010000.  mov dword [local_ch], 1
|           0x00400b58      c745f8060000.  mov dword [local_8h], 6
|           0x00400b5f      8b45f4         mov eax, dword [local_ch]
|           0x00400b62      0faf45f8       imul eax, dword [local_8h]
|           0x00400b66      8945fc         mov dword [local_4h], eax
|           0x00400b69      b800000000     mov eax, 0
|           0x00400b6e      5d             pop rbp
\           0x00400b6f      c3             ret
[0x00400a30]>
```

Question 7

Answer can be obtained in the same assembly code as in question 5 and 6.

```
[0x00400a30]> pdf @main
            ;-- main:
/ (fcn) sym.main 35
|   sym.main ();
|           ; var int local_ch @ rbp-0x c
|           ; var int local_8h @ rbp-0x 8
|           ; var int local_4h @ rbp-0x 4
|             ; DATA XREF from 0x00400a4d (entry0)
|           0x00400b4d      55             push rbp
|           0x00400b4e      4889e5         mov rbp, rsp
|           0x00400b51      c745f4010000.  mov dword [local_ch], 1
|           0x00400b58      c745f8060000.  mov dword [local_8h], 6
|           0x00400b5f      8b45f4         mov eax, dword [local_ch]
|           0x00400b62      0faf45f8       imul eax, dword [local_8h]
|           0x00400b66      8945fc         mov dword [local_4h], eax
|           0x00400b69      b800000000     mov eax, 0
|           0x00400b6e      5d             pop rbp
\           0x00400b6f      c3             ret
[0x00400a30]>
```

Thought process/methodology:

Having the machine deployed and successfully obtained the IP address, we logged into the instance by running **ssh** command, with **elfmceager** as the username and **adventofcyber** as the password. Then, by using Radare2 we analysed **challenge1** file by running **r2 -d ./file1** command. Once it is done, we open the assembly code at main by running the command **pdf @main.** Within this code, we obtained all the values needed for question 5,6 and 7.

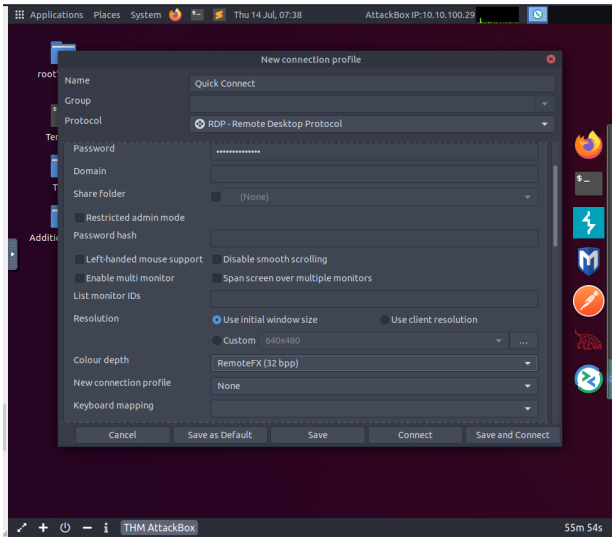
Day 18 - Reverse Engineering – The Bits of Christmas

Tools used: TryHackMe attackbox, mozilla firefox, Remmina, ILSpy

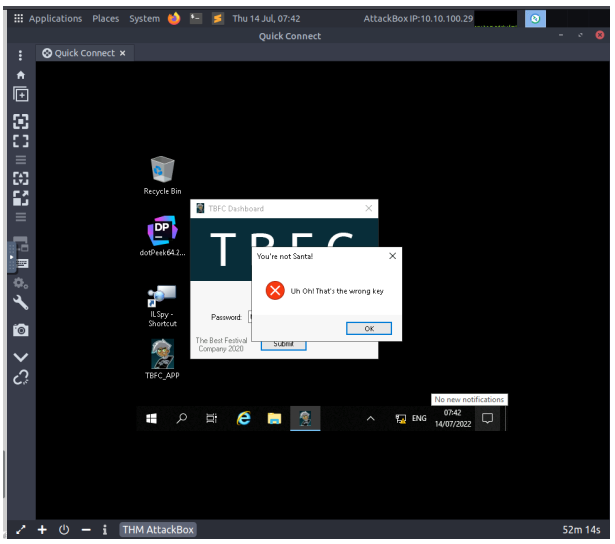
Solution/walkthrough

<u>Question 1</u>

Inserting the given credentials into Remmina

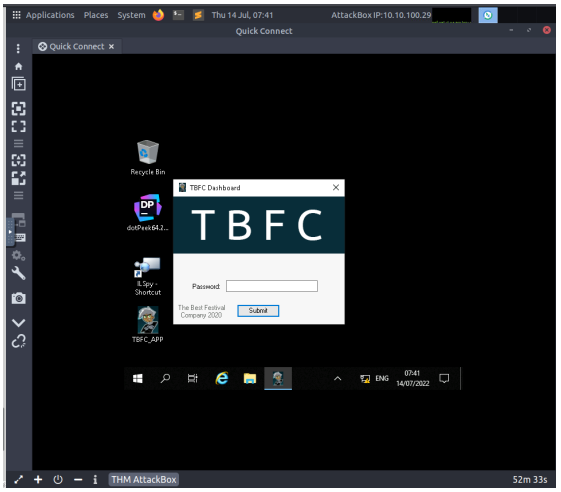


Trying to login into the TBFC_APP



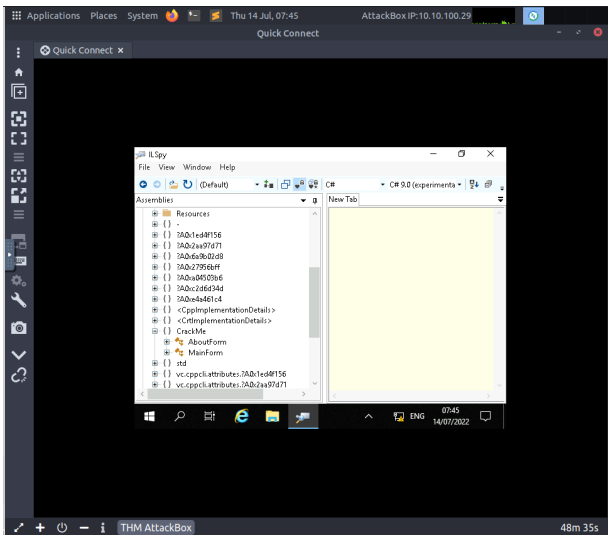
Question 2

Getting the full name of TBFC from TBC_APP



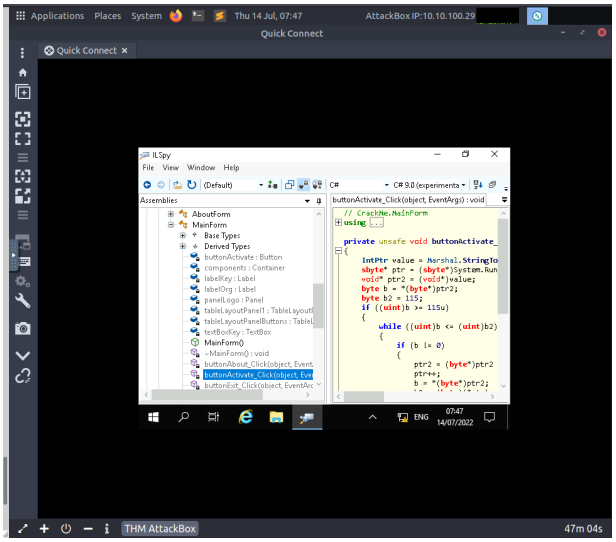
Question 3

Diving in deeper into “CrackMe” module



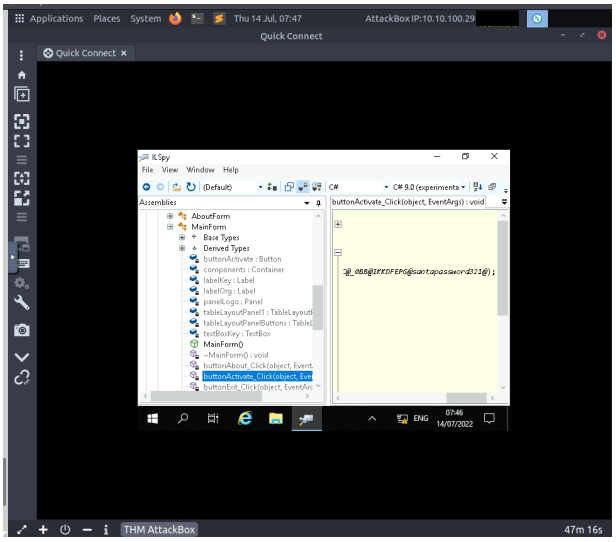
Question 4

Looking in the MainForm for the information that we are looking for.



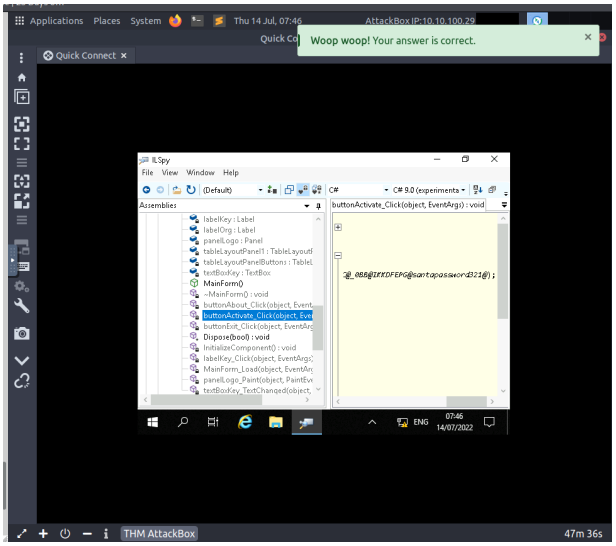
Question 5

Finding the module that contain Santa's password.



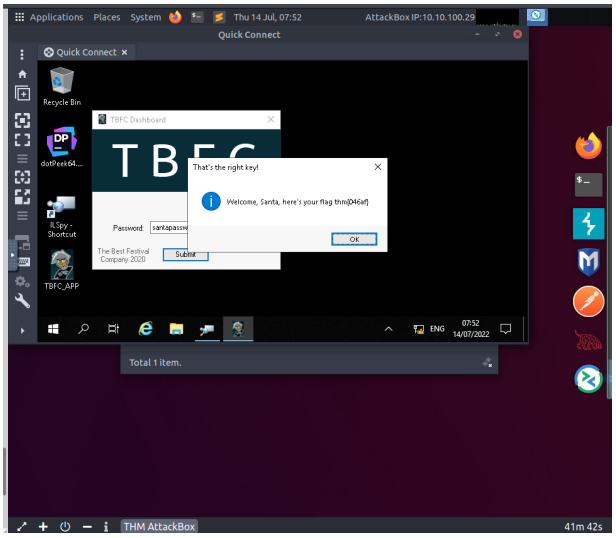
Question 6

Found the Santa's password



Question 7

Trying to login into TBFC_APP and get the flag.



Thought process/Methodology

Firstly, we launched the thm attackbox since Remmina and ILSpy are not in the kali linux virtual machine. After that, we tried to execute the TBFC_APP first to obtain the full name of TBFC. Then, we tried to enter a random password to retrieve the message that the app will show when entering the wrong password. We launched ILSpy and open TBFC_APP.exe to decompile it. We looked for the module with a unique name and we found module named “CrackMe”. We open that module and we looked into both of the forms in the CrackMe module to find the one that is relevant to the assignment. After that, we found the method containing Santa’s password which is buttonActivate_Click. Then, we execute TBFC_APP again to try the password obtained and capture the flag to end the question.

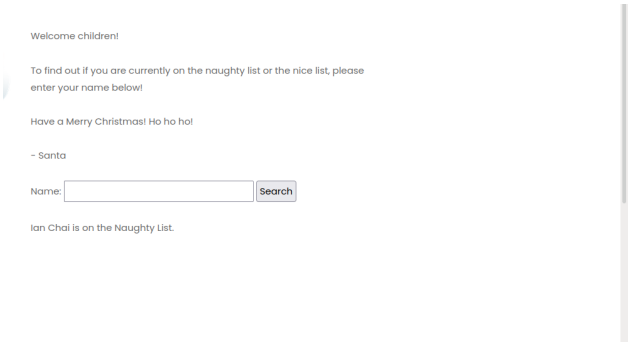
Day 19 - Web Exploitation – The Naughty or Nice List

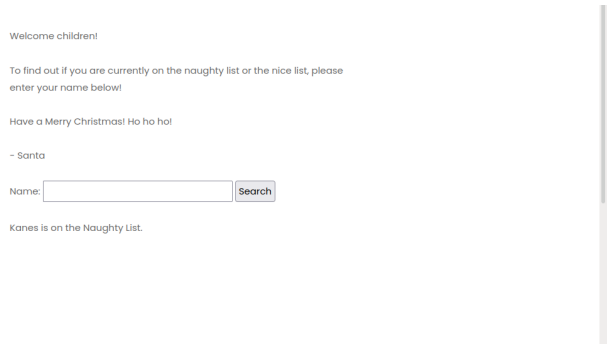
Tools used: kali linux, mozilla firefox

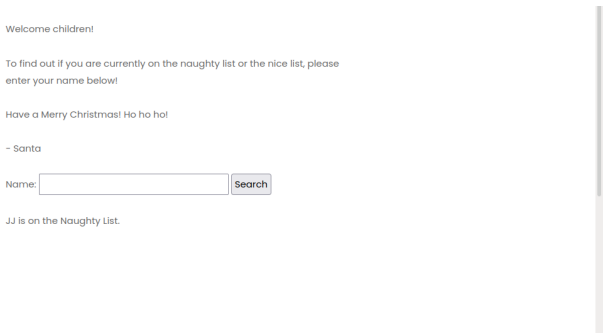
Solution/walkthrough

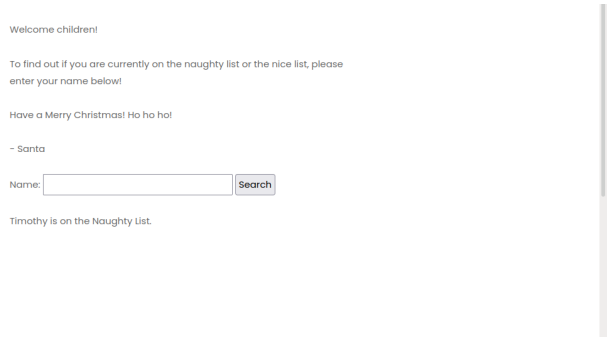
<u>Question 1</u>

Inserting the names into the naughty or nice search query









Welcome children!

To find out if you are currently on the naughty list or the nice list, please enter your name below!

Have a Merry Christmas! Ho ho ho!

- Santa

Name: Search

YP is on the Nice List.

Welcome children!

To find out if you are currently on the naughty list or the nice list, please enter your name below!

Have a Merry Christmas! Ho ho ho!

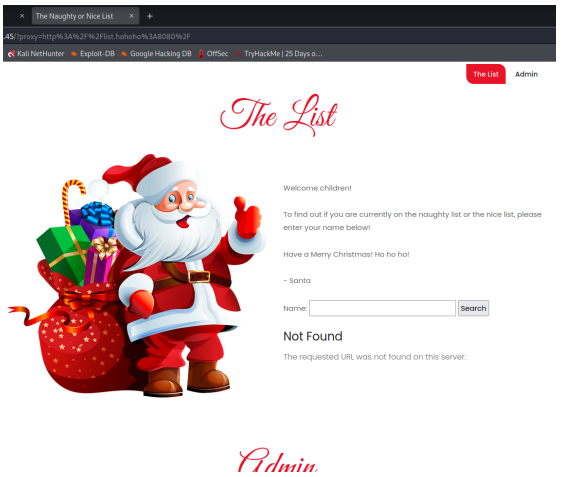
- Santa

Name: Search

Tib3rius is on the Nice List.

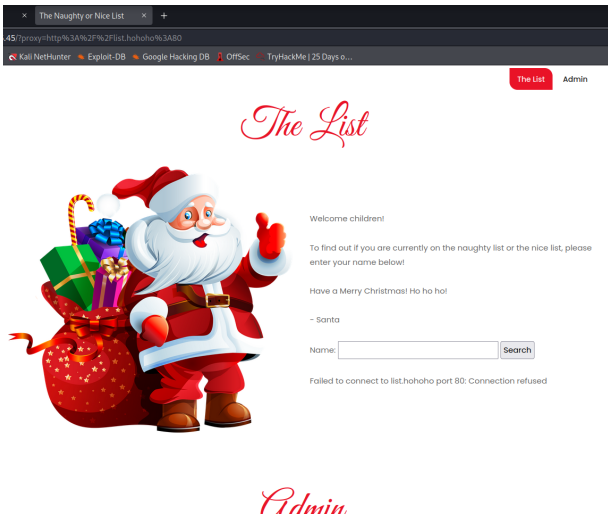
Question 2

Running the website on its root



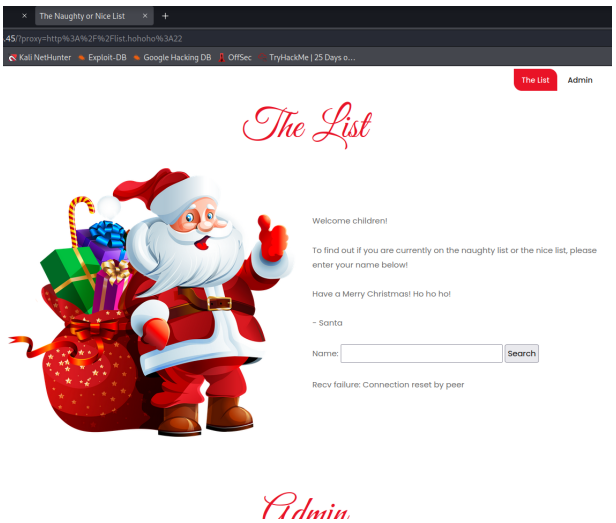
Question 3

Changing the port number to 80 (default http port)



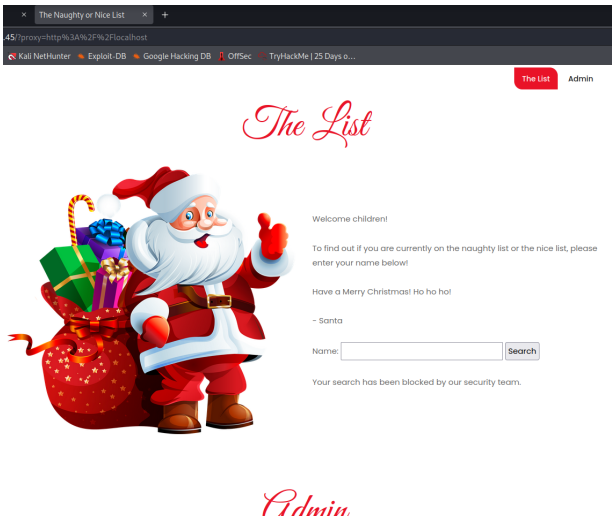
Question 4

Changing the port number to 22 (default ssh port)



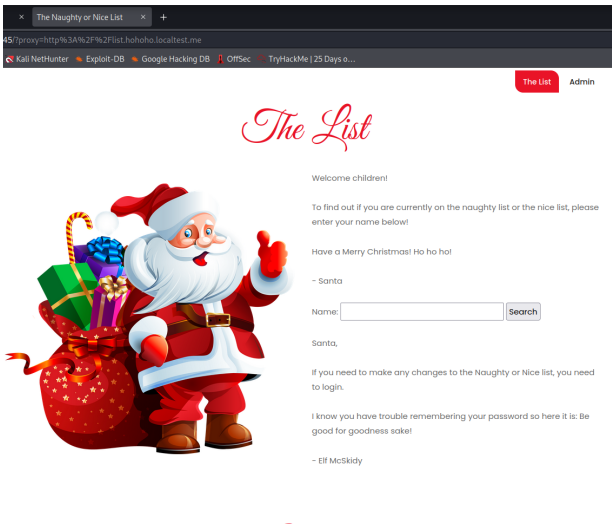
Question 5

Running the website on the localhost



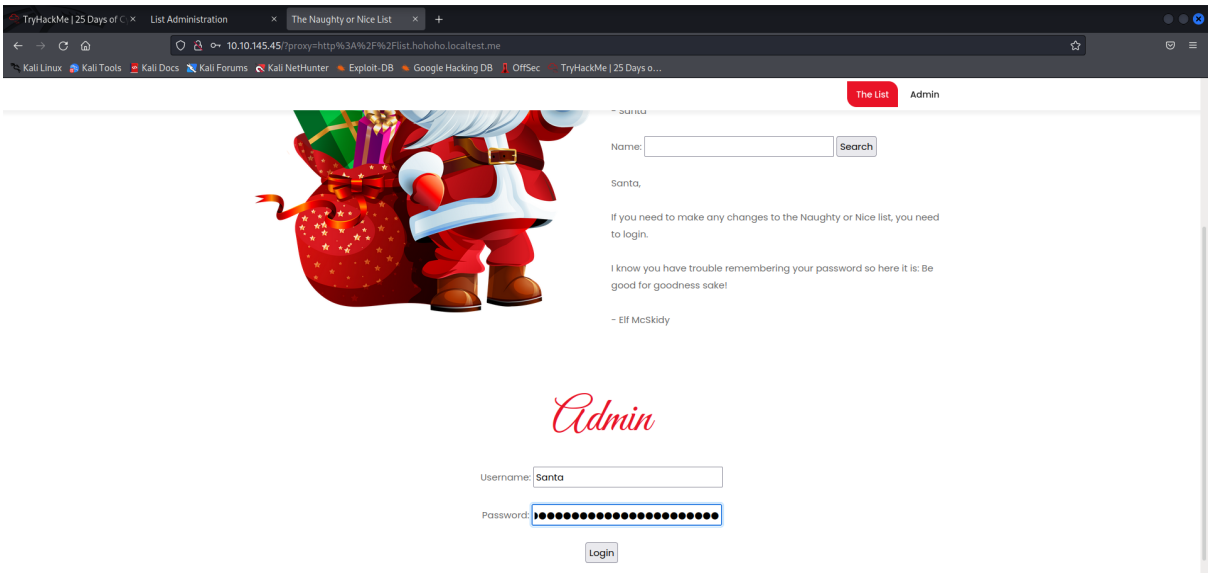
Question 6

Found the Santa's password

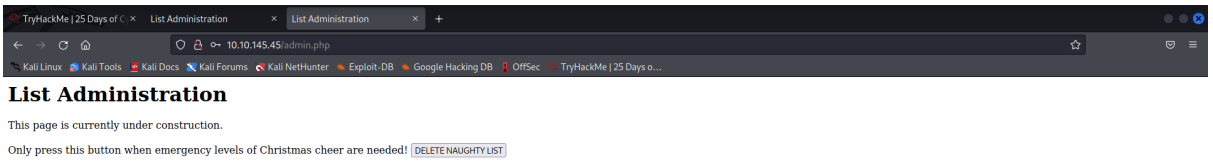


Question 7

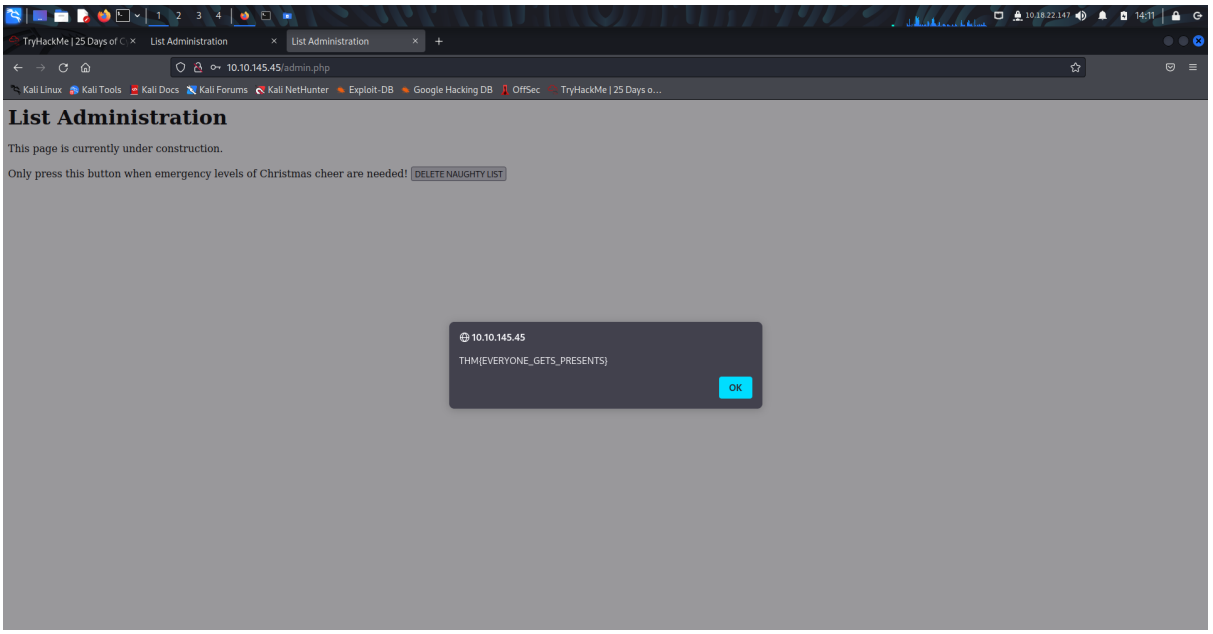
Logging into the website with the obtained credentials



Deleting the naughty list.



Captured the flag for the question



Thought process/Methodology

Firstly, we start the machine on thm website and go to the ip address of the machine. Then, we entered the name to know whether it is in naughty or nice list. After that, we tried running the website on its root, changing it to 80 and 22, and running it on the localhost. Then, we set the hostname in the url to list.hohoho.localtest.me to access the local services of the website. Through this method, we obtained Santa's password and we logged into the website to delete the naughty list. Then, we captured the flag for the question.

Day 20 - Blue Teaming - PowershELIF to the rescue

Tools used: Kali Linux, Terminal, PowerShell

Solution:

Question 1

Run the **ssh** command in the terminal and the answer will appear.

```
┌──(kali㉿kali)-[~]
└─$ ssh
usage: ssh [-46AaCfGgKkMNnqsTtVvXxYy] [-B bind_interface]
           [-b bind_address] [-c cipher_spec] [-D [bind_address:]port]
           [-E log_file] [-e escape_char] [-F configfile] [-I pkcs11]
           [-i identity_file] [-J [user@]host[:port]] [-L address]
           [-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option] [-p port]
           [-Q query_option] [-R address] [-S ctl_path] [-W host:port]
           [-w local_tun[:remote_tun]] destination [command [argument ... ]]
```

Question 2

Launch PowerShell and navigate to the Documents Folder. Use PowerShell command ***Set-Location .\Documents*** to access the 'Documents' directory. After we navigate into the directory, we can list the directory contents with **Get-ChildItem -Hidden -File** then we use command **Get-Content** to display the contents of the hidden file and the answer for Question 2.

```
PS C:\Users\mceager> Set-Location .\Documents\
PS C:\Users\mceager\Documents> Get-ChildItem -File -Hidden -ErrorAction Silently
Continue

    Directory: C:\Users\mceager\Documents


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a-hs-        12/7/2020  10:29 AM            402 desktop.ini
-arh--       11/18/2020   5:05 PM             35 e1fone.txt

PS C:\Users\mceager\Documents>
PS C:\Users\mceager\Documents> Get-Content -Path e1fone.txt
All I want is my '2 front teeth'!!!
```

Question 3

Same as Question 2, we navigate into Desktop directory and list all the contents inside it using **Get-ChildItem -Hidden -Directory** command. Then, we navigate into 'elf2wo' directory and list the directory's content inside and use **Get-Content** command to display the contents of a file and get the Question 3's answer.

```
PS C:\Users\mceager\Desktop> Set-Location .\elf2wo\
PS C:\Users\mceager\Desktop\elf2wo> Get-ChildItem

    Directory: C:\Users\mceager\Desktop\elf2wo

Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----       11/17/2020  10:26 AM             64 e70smsW10Y4k.txt


PS C:\Users\mceager\Desktop\elf2wo> Get-Content .\e70smsW10Y4k.txt
I want the movie Scrooged <3!
PS C:\Users\mceager\Desktop\elf2wo>
```

Question 4

Now we navigate to the windows directory and list the contents inside it using **Get-ChildItem -Hidden -Filter '*3*'.** The command will display us the hidden folder that contains files for Elf3.

```
PS C:\Users\mceager\Desktop\elf2wo> Set-Location C:\Windows\System32
PS C:\Windows\System32> Get-ChildItem -Hidden -Filter '*3*'


    Directory: C:\Windows\System32

Mode                LastWriteTime         Length Name
----                -------------         ------ ----
d--h--       11/23/2020   3:26 PM                3lfthr3e
```

Question 5

Navigate to the '3lfthr3e' directory, and list the file inside. There are two files by the name of '1.txt' and '2.txt'. To count the all words from file 1.txt, we have to run the command **Get-Content .\1.txt | Measure-Object -Word.**

```
PS C:\Windows\System32> Set-Location .\3lfthr3e\
PS C:\Windows\System32\3lfthr3e> Get-ChildItem -Hidden -File

    Directory: C:\Windows\System32\3lfthr3e


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-arh--        11/17/2020  10:58 AM          85887 1.txt
-arh--        11/23/2020   3:26 PM       12061168 2.txt


PS C:\Windows\System32\3lfthr3e> Get-Content .\1.txt | Measure-Object -Word

Lines Words Characters Property
-----  ----- ---------- --------
       9999
```

Question 6

We can use the command given in the task: **(Get-Content <path/file name>) [index]** to get the Question 6's answer.

```
PS C:\Windows\System32\3lfthr3e> (Get-Content .\1.txt)[551, 6991]
Red
Ryder
PS C:\Windows\System32\3lfthr3e>
```

Question 7

Same as Question 6, we use the command given: **Select-String <path/filename> -Pattern 'redryder'** to get the answer.

```
PS C:\Windows\System32\3lfthr3e> Select-String .\2.txt -Pattern 'redryder'

2.txt:558704:redryderbbgun
```

Thought process/methodology:

For Day 20's task we only using terminal and PowerShell to complete it. To get the answer for Question 1, we only have to run the **ssh** command in the terminal and it will list out all the ssh usage. Next we have to launch PowerShell then navigate to the directory which each question is asking by using the PowerShell command: **Set-Location**. Then, to list all the contents inside the directory, we use **Get-ChildItem** command, and to display the contents of the file, we use **Get-Content** command. These commands are used in the Question 2,3,4 and 5 meanwhile Question 6 and 7 we use the command given in the task.