PSP0201

Week 5

Writeup

Group Name: Capybozos

Members

ID	Name	Role
1211201568	Muhammad Albukhari bin Norazmi	Leader
1211101392	Wong Yen Hong	Member
1211101399	Karthigeayah A/L Maniam	Member
1211100732	Ephraim Tee Yu Yang	Member

Day 16 - Help! Where is Santa?

Tools Used : Kali, Firefox, Nmap, Python (Request & BeautifulSoup4)

Q: What is the port number for the web server?

80

1. By doing simple port scanning with nmap, we can get the open port.

```
┌──(1211101392㉿kali)-[~/Desktop/25 Days of CyberSecurity/day16]
└─$ sudo nmap -sS 10.10.74.70 -Pn
[sudo] password for 1211101392:
Starting Nmap 7.92 ( https://nmap.org ) at 2022-07-06 03:23 EDT
Nmap scan report for 10.10.74.70
Host is up (0.21s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 13.03 seconds
```

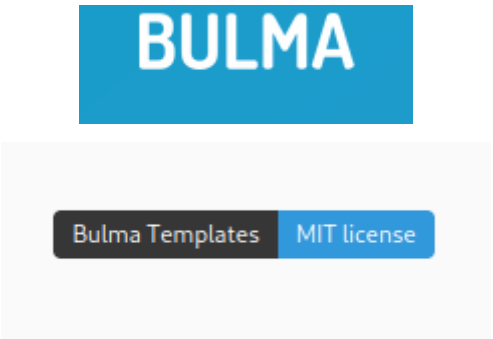
SSH is not for web server, so that leaves us with port 80, which is http.

Q: What templates are being used?

It's the five character name at the top left of the website. Answer in Uppercase.

BULMA

1. Open the site, and look for the template name.



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

/api/

1. FIrst, let's code a simple python script with request and beautifulsoup to get us all the <a> tag, which stands for hyperlink.

```
┌──(1211101392㉿kali)-[~/Desktop/25 Days of CyberSecurity/day16]
└─$ vim findlink.py
```

```
import requests
from bs4 import BeautifulSoup

def main():
    page = requests.get(url="http://10.10.74.70/")
    soup = BeautifulSoup(page.content, 'html.parser')
    for i in soup.find_all('a'):
        print(i)


main()
```

2. Execute the script.

```
┌──(1211101392㉿kali)-[~/Desktop/25 Days of CyberSecurity/day16]
└─$ python3 findlink.py
```

3. Look for a suspicious link that might be our interest.

```
<a href="https://tryhackme.com">Santa</a>
<a href="https://tryhackme.com">Santa</a>
<a href="https://tryhackme.com">humans</a>
<a href="https://tryhackme.com">click</a>
<a href="https://tryhackme.com">Python</a>
<a href="https://tryhackme.com">notice</a>
<a href="https://tryhackme.com">Skidy</a>
<a href="https://tryhackme.com">TryHackMe</a>
<a href="https://tryhackme.com">man</a>
<a href="https://tryhackme.com">613</a>
<a href="https://tryhackme.com">jumper</a>
<a href="#">Lorem ipsum dolor sit amet</a>
<a href="#">Vestibulum errato isse</a>
<a href="#">Lorem ipsum dolor sit amet</a>
<a href="#">Aisia caisia</a>
<a href="#">Murphy's law</a>
<a href="#">Flimsy Lavenrock</a>
<a href="#">Maven Mousie Lavender</a>
<a href="#">Labore et dolore magna aliqua</a>
<a href="#">Kanban airis sum eschelor</a>
<a href="http://machine_ip/api/api_key">Modular modern free</a>
<a href="#">The king of clubs</a>
<a href="#">The Discovery Dissipation</a>
<a href="#">Course Correction</a>
<a href="#">Better Angels</a>
<a href="#">Objects in space</a>
<a href="#">Playing cards with coyote</a>
<a href="#">Goodbye Yellow Brick Road</a>
<a href="#">The Garden of Forking Paths</a>
<a href="#">Future Shock</a>
<a class="icon" href="https://github.com/BulmaTemplates/bulma-templates">
```

```
<a href="http://machine_ip/api/api_key">Modular modern free</a>
```

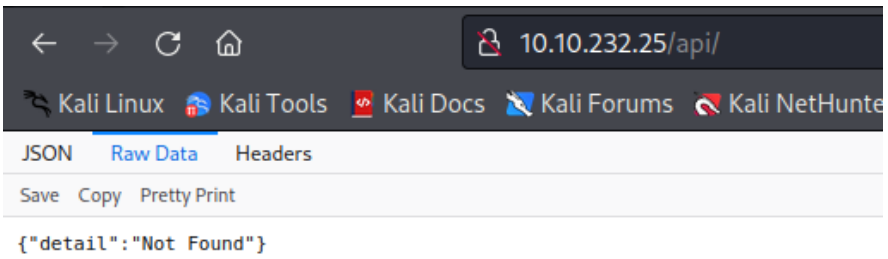
And we found it.

Q: Go the API endpoint. What is the Raw Data returned if no parameters are entered?

Copy and paste from THM's website response. (See the Raw Data tab in Firefox.) Include the curly brackets.

{"detail":"Not Found"}

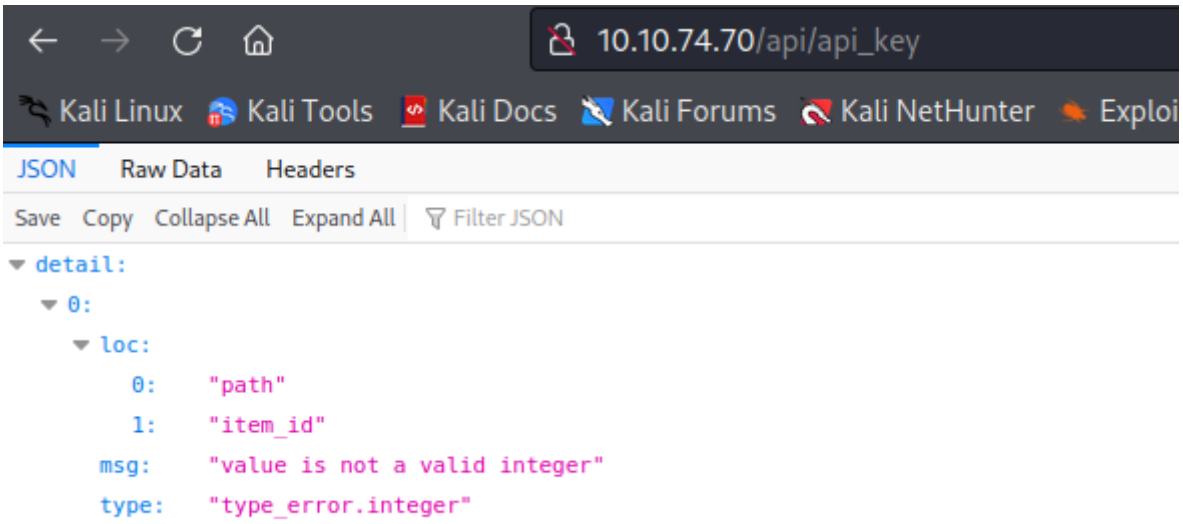
1. Go to the API endpoint without a parameter.



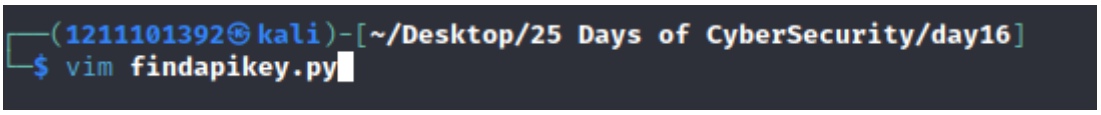
Q: `Where is Santa right now?`

Winter Wonderland, Hyde Park, London

1. First, try to access the link we just found, and it turned out that api returns a json, and we're using the wrong value. So, apparently, we have to use an integer after /api/. And elves told us that the api_key is a number between 0 to 100 and it's an odd number.



2. Next thing we do is to code another script to automate the task for us. Since the number is guaranteed to be an odd number so we just have to enumerate the odd numbers.



```python
import requests

def main():
    for i in range(1,100,2):
        page = requests.get(f"http://10.10.74.70/api/{i}")
        print(f"{i} - {page.json()}")

main()
```

3. Execute the script and look for the correct api key that returns the right json.

```
┌──(1211101392㉿kali)-[~/Desktop/25 Days of CyberSecurity/day16]
└─$ python3 findapikey.py
1 - {'item_id': 1, 'q': 'Error. Key not valid!'}
3 - {'item_id': 3, 'q': 'Error. Key not valid!'}
5 - {'item_id': 5, 'q': 'Error. Key not valid!'}
7 - {'item_id': 7, 'q': 'Error. Key not valid!'}
9 - {'item_id': 9, 'q': 'Error. Key not valid!'}
11 - {'item_id': 11, 'q': 'Error. Key not valid!'}
13 - {'item_id': 13, 'q': 'Error. Key not valid!'}
15 - {'item_id': 15, 'q': 'Error. Key not valid!'}
17 - {'item_id': 17, 'q': 'Error. Key not valid!'}
19 - {'item_id': 19, 'q': 'Error. Key not valid!'}
21 - {'item_id': 21, 'q': 'Error. Key not valid!'}
23 - {'item_id': 23, 'q': 'Error. Key not valid!'}
25 - {'item_id': 25, 'q': 'Error. Key not valid!'}
27 - {'item_id': 27, 'q': 'Error. Key not valid!'}
29 - {'item_id': 29, 'q': 'Error. Key not valid!'}
31 - {'item_id': 31, 'q': 'Error. Key not valid!'}
33 - {'item_id': 33, 'q': 'Error. Key not valid!'}
35 - {'item_id': 35, 'q': 'Error. Key not valid!'}
37 - {'item_id': 37, 'q': 'Error. Key not valid!'}
39 - {'item_id': 39, 'q': 'Error. Key not valid!'}
41 - {'item_id': 41, 'q': 'Error. Key not valid!'}
43 - {'item_id': 43, 'q': 'Error. Key not valid!'}
45 - {'item_id': 45, 'q': 'Error. Key not valid!'}
47 - {'item_id': 47, 'q': 'Error. Key not valid!'}
49 - {'item_id': 49, 'q': 'Error. Key not valid!'}
51 - {'item_id': 51, 'q': 'Error. Key not valid!'}
53 - {'item_id': 53, 'q': 'Error. Key not valid!'}
55 - {'item_id': 55, 'q': 'Error. Key not valid!'}
57 - {'item_id': 57, 'q': 'Winter Wonderland, Hyde Park, London.'}
59 - {'item_id': 59, 'q': 'Error. Key not valid!'}
61 - {'item_id': 61, 'q': 'Error. Key not valid!'}
63 - {'item_id': 63, 'q': 'Error. Key not valid!'}
65 - {'item_id': 65, 'q': 'Error. Key not valid!'}
67 - {'item_id': 67, 'q': 'Error. Key not valid!'}
69 - {'item_id': 69, 'q': 'Error. Key not valid!'}
71 - {'item_id': 71, 'q': 'Error. Key not valid!'}
73 - {'item_id': 73, 'q': 'Error. Key not valid!'}
```

57 - {'item_id': 57, 'q': 'Winter Wonderland, Hyde Park, London.'}

And we can see that key 57 returned santa's location, which is the answer to this question.

Q: 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.

To unblock yourself, simply terminate and re-deploy the target instance

57

```
57 - {'item_id': 57, 'q': 'Winter Wonderland, Hyde Park, London.'}
```

<u>Thought Process/ Methodology</u>

This is a good exercise for beginners like us, using requests and bs4 with python is something fresh for us! Okay let's get into our thought process throughout this exercise. So first, we're given a web server, and for the first question, we are to find the port number for the webserver, and that is relatively simple, we got that by doing a simple nmap port scanning with SYN scan. Next, we are to look for the hidden api link in the webpage, it would be tedious to look for it manually, as there are quite a number of hyperlinks, so as the website suggests, it's better if we do it with some script. So I decided to use the request module and bs4 to parse the html page, and look for all the hyperlink tags. Maybe there is a way to do it in a more efficient way which is with regular expressions, maybe we can filter out more things with it. After that, we're to look for the correct api key, I did not know where we have to enter the api key at first sight, but after accessing the site with the link given, i found out that we're supposed to enter the api key at the /api/{api_key} section. And with a little bit of research on google, I found out that we could use the request module to return the json file along with the link. And the rest is simply coding it out.

Day 17 - ReverseELFneering

Tools used: Kali Linux, radare2

Q: What is the command to analyse the program in radare2?
aa

1. Debug the file using radare2 and run `aa` to analyse the program.

```
elfmceager@tbfc-day-17:~$ r2 -d ./challenge1
Process with PID 1567 started...
= attach 1567 1567
bin.baddr 0x00400000
Using 0x400000
Warning: Cannot initialize dynamic strings
asm.bits 64
[0x00400a30]> aa
[ WARNING : block size exceeding max block size at 0x006ba220
[+] Try changing it with e anal.bb.maxsize
 WARNING : block size exceeding max block size at 0x006bc860
[+] Try changing it with e anal.bb.maxsize
[x] Analyze all flags starting with sym. and entry0 (aa)
[0x00400a30]>
```

Q: What is the command to set a breakpoint in radare2?
db

1. To set a breakpoint, use the following syntax `db <address>`. Let's say we want to set a breakpoint at the second line, with the mov instruction.

```
[0x00400a30]> pdf @main
            ;-- main:
/ (fcn) sym.main 35
|   sym.main ();
|           ; var int local_ch @ rbp-0xc
|           ; var int local_8h @ rbp-0x8
|           ; var int local_4h @ rbp-0x4
|              ; 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]> db 0x00400b4e
```

2. If we look at the main function again, we can observe a “b” on line 2. The breakpoint has been set on line 2.

```
[0x00400a30]> pdf @main
            ;-- main:
/ (fcn) sym.main 35
|   sym.main ();
|           ; var int local_ch @ rbp-0x c
|           ; var int local_8h @ rbp-0x8
|           ; var int local_4h @ rbp-0x4
|              ; DATA XREF from 0x00400a4d (entry0)
|           0x00400b4d      55              push rbp
|           0x00400b4e b    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
```

Q: What is the command to execute the program until we hit a breakpoint?

dc

1. Using the breakpoint from the previous question, run the dc command inside radare2.

```
[0x00400a30]> dc
hit breakpoint at: 400b4e
```

2. The program has been executed until it hit the breakpoint, as highlighted by radare2.

```
[0x00400b4e]> pdf @main
            ;-- main:
            ;-- rax:
/ (fcn) sym.main 35
|   sym.main ();
|           ; var int local_ch @ rbp-0x c
|           ; var int local_8h @ rbp-0x8
|           ; var int local_4h @ rbp-0x4
|              ; DATA XREF from 0x00400a4d (entry0)
|           0x00400b4d      55              push rbp
|           ;-- rip:
|           0x00400b4e b    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
```

Q: What is the value of **local_ch** when its corresponding movl instruction is called (first if multiple)?

1

1. Before doing anything else, the first thing to do is SSH into the target machine using the specified credentials.

```
┌──(1211201568㉿kali)-[~/Desktop/tryhackme/day17]
└─$ ssh elfmceager@10.10.146.68
The authenticity of host '10.10.146.68 (10.10.146.68)' can't be established.
ED25519 key fingerprint is SHA256:+Yl8Ef3BjQ7HNTMf6qew50LnmiqEXXSzLqgX82k/RSg.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:6: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.146.68' (ED25519) to the list of known hosts.
elfmceager@10.10.146.68's password:
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-128-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Tue Jul  5 04:03:25 UTC 2022

  System load:  0.63                Processes:           100
  Usage of /:   39.4% of 11.75GB    Users logged in:     0
  Memory usage: 8%                  IP address for ens5: 10.10.146.68
  Swap usage:   0%


0 packages can be updated.
0 updates are security updates.


Last login: Wed Dec 16 18:25:51 2020 from 192.168.190.1
elfmceager@tbfc-day-17:~$
```

2. By listing the contents of the current folder, we see the challenge1 file, use the following radare2 command to debug it.

```
elfmceager@tbfc-day-17:~$ ls
challenge1  file1
elfmceager@tbfc-day-17:~$ r2 -d ./challenge1
```

3. In debug mode, run the `aa` command to analyse it.

```
[0x00400a30]> aa
[ WARNING : block size exceeding max block size at 0x006ba220
[+] Try changing it with e anal.bb.maxsize
 WARNING : block size exceeding max block size at 0x006bc860
[+] Try changing it with e anal.bb.maxsize
[x] Analyze all flags starting with sym. and entry0 (aa)
```

4. Run `afl | grep main` to find a list of functions which contain the keyword 'main', then display the code of that function using `pdf @main`.

```
[0x00400a30]> afl | grep main
0x00400b4d    1 35           sym.main
0x00400de0   10 1007 → 219   sym.__libc_start_main
0x00403840   39 661  → 629   sym._nl_find_domain
0x00403ae0  308 5366 → 5301  sym._nl_load_domain
0x00415ef0    1 43           sym._IO_switch_to_main_get_area
0x0044ce10    1 8            sym._dl_get_dl_main_map
0x00470430    1 49           sym._IO_switch_to_main_wget_area
0x0048f9f0    7 73   → 69    sym._nl_finddomain_subfreeres
0x0048fa40   16 247  → 237   sym._nl_unload_domain
[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
```

5. As the first 3 lines represent the variables stored inside the function, we will pay attention to **local_ch** as that is the goal for this question. From here, set a breakpoint at line 7 using the following command at the highlighted address.

```
/ (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]> db 0x00400b51
```

6. Run the `dc` command until it hits the breakpoint, run `px @ rbp-0xc` to check the current value of the memory address containing the variable in hexadecimal form. Pay attention to the highlighted value which is currently 0.

```
[0x00400a30]> dc
hit breakpoint at: 400b51
[0x00400b51]> px @ rbp-0xc
- offset -       0 1  2 3  4 5  6 7  8 9  A B  C D  E F  0123456789ABCDEF
0x7ffc01e7b874  0000 0000 1890 6b00 0000 0000 4018 4000  .....k......@.@.
0x7ffc01e7b884  0000 0000 e910 4000 0000 0000 0000 0000  ......@.........
0x7ffc01e7b894  0000 0000 0000 0000 0100 0000 a8b9 e701  ................
0x7ffc01e7b8a4  fc7f 0000 4d0b 4000 0000 0000 0000 0000  ....M.@.........
0x7ffc01e7b8b4  0000 0000 1700 0000 0100 0000 0000 0000  ................
0x7ffc01e7b8c4  0000 0000 0000 0000 0200 0000 0000 0000  ................
0x7ffc01e7b8d4  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7ffc01e7b8e4  0000 0000 0000 0000 0000 0000 0004 4000  ..............@.
0x7ffc01e7b8f4  0000 0000 7d02 56fa dccb 33da e018 4000  ....}.V...3...@.
0x7ffc01e7b904  0000 0000 0000 0000 0000 0000 1890 6b00  ............k.
0x7ffc01e7b914  0000 0000 0000 0000 0000 0000 7d02 f6bb  ............}...
0x7ffc01e7b924  93c8 cb25 7d02 e2eb dccb 33da 0000 0000  ...%}.....3.....
0x7ffc01e7b934  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7ffc01e7b944  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7ffc01e7b954  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7ffc01e7b964  0000 0000 0000 0000 0000 0000 0000 0000  ................
```

7. Run the `ds` command, which is used to execute an instruction and go to the next one. Now, check the highlighted value again and notice that the value is now 1 which is the answer to this question.

```
[0x00400b51]> ds
[0x00400b51]> px @ rbp-0xc
- offset -       0 1  2 3  4 5  6 7  8 9  A B  C D  E F  0123456789ABCDEF
0x7ffc01e7b874  0100 0000 1890 6b00 0000 0000 4018 4000  .....k......@.@.
0x7ffc01e7b884  0000 0000 e910 4000 0000 0000 0000 0000  ......@.........
0x7ffc01e7b894  0000 0000 0000 0000 0100 0000 a8b9 e701  ................
0x7ffc01e7b8a4  fc7f 0000 4d0b 4000 0000 0000 0000 0000  ....M.@.........
0x7ffc01e7b8b4  0000 0000 1700 0000 0100 0000 0000 0000  ................
0x7ffc01e7b8c4  0000 0000 0000 0000 0200 0000 0000 0000  ................
0x7ffc01e7b8d4  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7ffc01e7b8e4  0000 0000 0000 0000 0000 0000 0004 4000  ..............@.
0x7ffc01e7b8f4  0000 0000 7d02 56fa dccb 33da e018 4000  ....}.V...3...@.
0x7ffc01e7b904  0000 0000 0000 0000 0000 0000 1890 6b00  ............k.
0x7ffc01e7b914  0000 0000 0000 0000 0000 0000 7d02 f6bb  ............}...
0x7ffc01e7b924  93c8 cb25 7d02 e2eb dccb 33da 0000 0000  ...%}.....3.....
0x7ffc01e7b934  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7ffc01e7b944  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7ffc01e7b954  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7ffc01e7b964  0000 0000 0000 0000 0000 0000 0000 0000  ................
```

Q: What is the value of **eax** when the imull instruction is called?

6

1. Set a breakpoint anywhere and slowly step to the instruction which moves the value of **local_ch** into the **eax** register.

```
/ (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 b    c745f4010000.   mov dword [local_ch], 1
|           0x00400b58      c745f8060000.   mov dword [local_8h], 6
|           ;-- rip:
|           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
```

2. To see the current value of the **eax** register, use the `dr` command. Pay attention to the highlighted value.

```
[0x00400b51]> dr
rax = 0x00400b4d
rbx = 0x00400400
rcx = 0x0044b9a0
rdx = 0x7ffe58b4e4c8
r8 = 0x01000000
r9 = 0x006bb8e0
r10 = 0x00000015
r11 = 0x00000000
r12 = 0x004018e0
r13 = 0x00000000
r14 = 0x006b9018
r15 = 0x00000000
rsi = 0x7ffe58b4e4b8
rdi = 0x00000001
rsp = 0x7ffe58b4e390
rbp = 0x7ffe58b4e390
rip = 0x00400b5f
rflags = 0x00000246
orax = 0xffffffffffffffff
```

3. Now execute the instruction which moves **local_ch** into the **eax** register by running `ds`. After that, run `dr` again to see the value in the register change.

```
[0x00400b51]> ds
[0x00400b51]> dr
rax = 0x00000001
rbx = 0x00400400
rcx = 0x0044b9a0
rdx = 0x7ffe58b4e4c8
r8 = 0x01000000
r9 = 0x006bb8e0
r10 = 0x00000015
r11 = 0x00000000
r12 = 0x004018e0
r13 = 0x00000000
r14 = 0x006b9018
r15 = 0x00000000
rsi = 0x7ffe58b4e4b8
rdi = 0x00000001
rsp = 0x7ffe58b4e390
rbp = 0x7ffe58b4e390
rip = 0x00400b62
rflags = 0x00000246
orax = 0xffffffffffffffff
```

4. Just to make sure where we currently are inside the code, run `pdf @main` again. We are now at the imul instruction.

```
[0x00400b51]> 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 b    c745f4010000.  mov dword [local_ch], 1
|           0x00400b58      c745f8060000.  mov dword [local_8h], 6
|           0x00400b5f      8b45f4         mov eax, dword [local_ch]
|           ;-- rip:
|           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
```

5. Run `ds` to execute the instruction, and check the value of **rax** by using `dr`.

```
[0x00400b51]> dr
rax = 0x00000006
rbx = 0x00400400
rcx = 0x0044b9a0
rdx = 0x7ffe58b4e4c8
r8 = 0x01000000
r9 = 0x006bb8e0
r10 = 0x00000015
r11 = 0x00000000
r12 = 0x004018e0
r13 = 0x00000000
r14 = 0x006b9018
r15 = 0x00000000
rsi = 0x7ffe58b4e4b8
rdi = 0x00000001
rsp = 0x7ffe58b4e390
rbp = 0x7ffe58b4e390
rip = 0x00400b66
rflags = 0x00000246
orax = 0xffffffffffffffff
```

6. The value of **rax** is now 6, which is the answer to this question.
7.

Q: What is the value of **local_4h** before **eax** is set to 0?

6

1. Run `pdf @main` and set a breakpoint at line 11 using the `db` command.

```
[0x00448a86]> 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
[0x00448a86]> db 0x00400b66
```

2. Run the `dc` command until it hits the breakpoint, then execute the instruction using `ds`.

```
[0x00400a30]> dc
hit breakpoint at: 400b66
[0x00400b66]> pdf @ main
            ;-- main:
/ (fcn) sym.main 35
|   sym.main ();
|           ; var int local_ch @ rbp-0x c
|           ; var int local_8h @ rbp-0x8
|           ; var int local_4h @ rbp-0x4
|               ; 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]
|           ;-- rip:
|           0x00400b66 b    8945fc          mov dword [local_4h], eax
|           0x00400b69      b800000000      mov eax, 0
|           0x00400b6e      5d              pop rbp
\           0x00400b6f      c3              ret
[0x00400b66]> ds
```

3. Check the value of **local_4h** using `px @ rbp-0x4`.

```
[0x00400b66]> px @ rbp-0x4
- offset -       0 1  2 3  4 5  6 7  8 9  A B  C D  E F  0123456789ABCDEF
0x7ffedff102dc  0600 0000 4018 4000 0000 0000 e910 4000  .....@.@........@.
0x7ffedff102ec  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7ffedff102fc  0100 0000 0804 f1df fe7f 0000 4d0b 4000  ..............M.@.
0x7ffedff1030c  0000 0000 0000 0000 0000 0000 1700 0000  ................
0x7ffedff1031c  0100 0000 0000 0000 0000 0000 0000 0000  ................
0x7ffedff1032c  0200 0000 0000 0000 0000 0000 0000 0000  ................
0x7ffedff1033c  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7ffedff1034c  0000 0000 0004 4000 0000 0000 ebf5 77c6  ......@.......w.
0x7ffedff1035c  3a04 340f e018 4000 0000 0000 0000 0000  :.4.....@.......
0x7ffedff1036c  0000 0000 1890 6b00 0000 0000 0000 0000  ......k.........
0x7ffedff1037c  0000 0000 ebf5 17f3 58bb c9f0 ebf5 c3d7  ........X.......
0x7ffedff1038c  3a04 340f 0000 0000 0000 0000 0000 0000  :.4.............
0x7ffedff1039c  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7ffedff103ac  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7ffedff103bc  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7ffedff103cc  0000 0000 0000 0000 0000 0000 0000 0000  ................
```

4. Notice that the value at the top left is 6, so the value of **local_4h** is 6 which is the answer to this question.

Thought Process/ Methodology

When debugging Assembly code, it's helpful to refer to cheat sheets to make the job easier such as the ones for radare2 and Assembly Language. Do pay attention to the syntax differences though, as the examples inside the tryhackme article use AT&T syntax while radare2 by default uses Intel syntax. The key difference is the order of the source and destination, Intel syntax will put the destination before the source for instructions whereas vice versa for AT&T syntax. With that out of the way, if you are familiar with debugging code in relatively higher-level programming languages compared to Assembly, such as C/C++ and Python, the process is similar for Assembly and you will be able to quickly adapt. First, we set breakpoints using the `db` command in our code and let the code run using the `dc` command until we hit the breakpoint, then we can analyse the code step-by-step using the `ds` command and check the values of the variables and registers as the code runs. To check values, we use the `dr` command for registers and `px` for variables. For the first question the source, 1 in this case, has been moved using the mov instruction to the destination, local_ch, resulting in the value of local_ch being 1. For the second question, the source, local_ch with the value of 1, has been moved using the mov instruction into the destination, the eax register. Next, the imul instruction multiplies the value of the eax register with that of local_8h, resulting in the value being 6. Finally, for the last question, the value of the source, eax, has been moved into the destination, local_4h, resulting in the value of local_4h being 6.

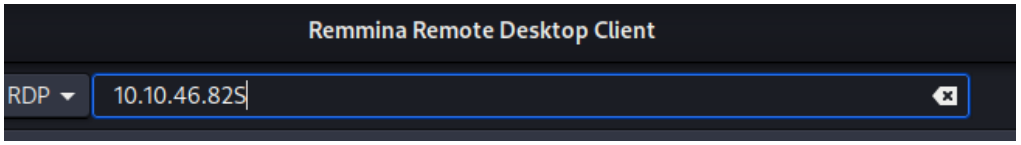
Day 18 - The Bits of Christmas

Tools Used : Kali, Remmina, ILSpy, cyberchef

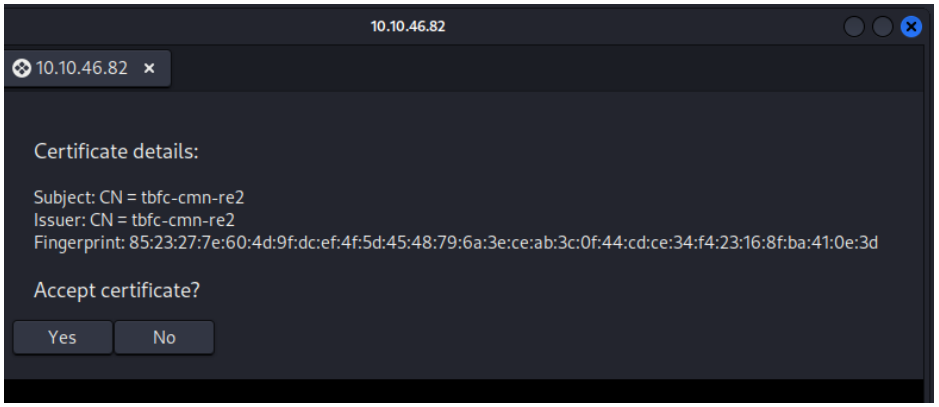
Q: What is the message that shows up if you enter the wrong password for TBFC_APP?

Uh Oh! That's the wrong key

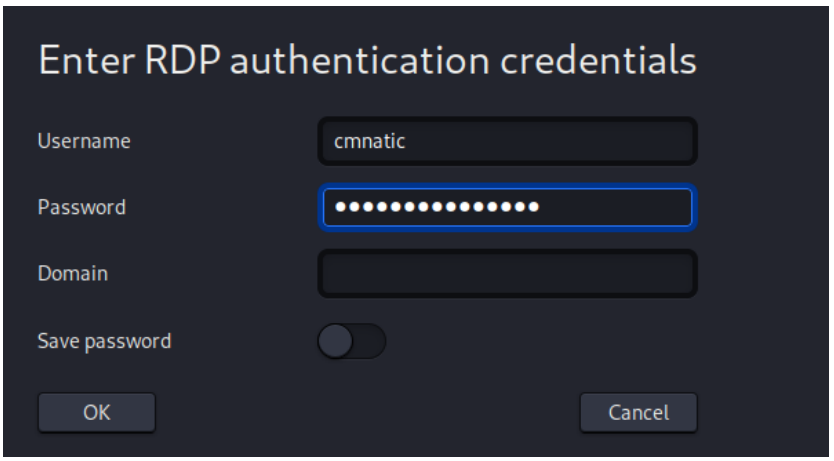
1. Run Remmina and enter the ip address of the target.



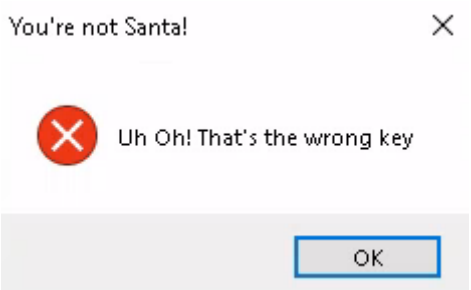
2. Accept the certificate.



3. Enter the username and password. For this instance, the username will be cmnatic and the password is Adventofcyber!



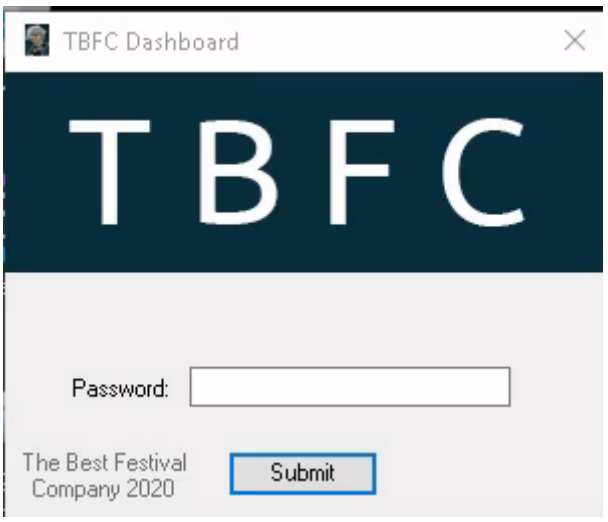
4. Open the TBFC app and enter a random password, chances are, it is the wrong password and you will receive this prompt.



Q: What does TBFC stand for?

The Best Festival Company

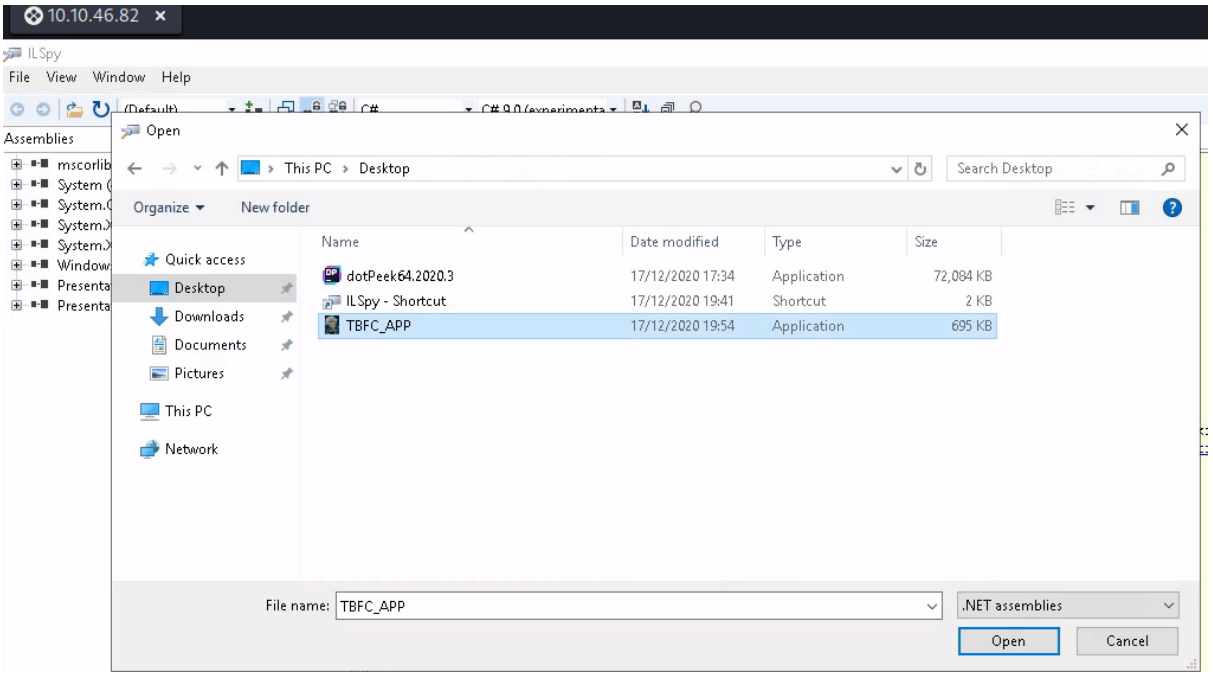
1. Open the TBFC app and you can see at the bottom left corner, it says “The Best Festival Company”.



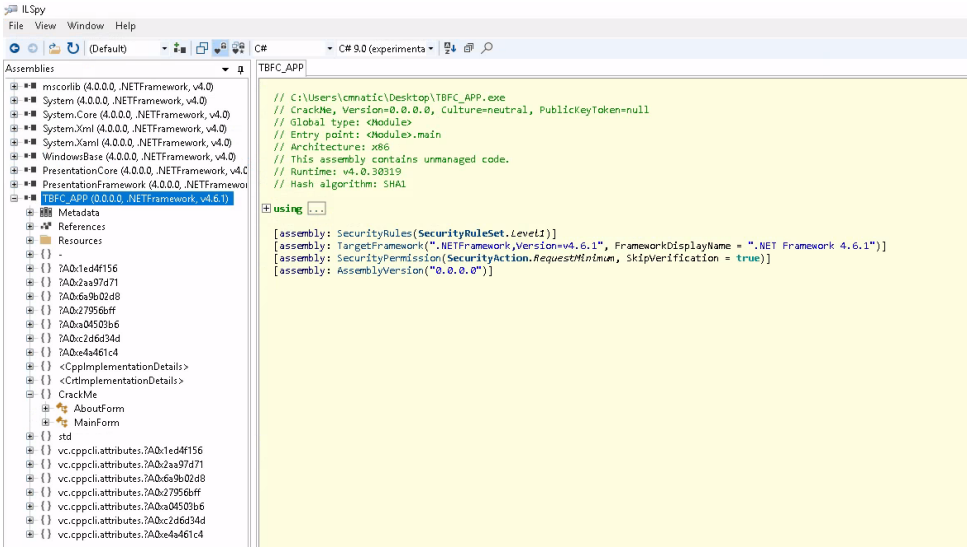
Q: Decompile the TBFC_APP with ILSpy. What is the module that catches your attention?

CrackMe

1. On the remote desktop, open up ILSpy. In ILSpy, go to File > Open and select the TBFC app.



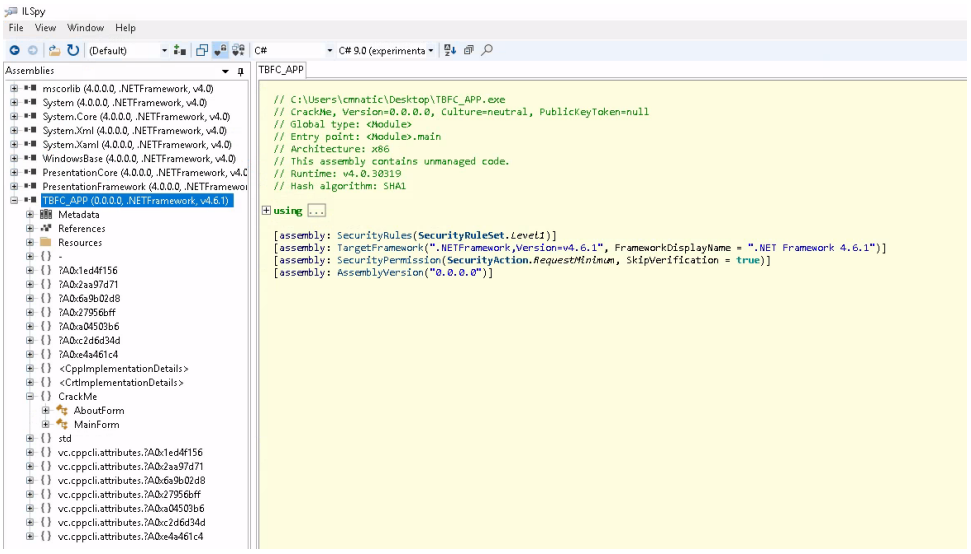
2. At the left side of the window, we can see all the modules used. The one that stands out the most is “CrackMe”



Q: Within the module, there are two forms. Which contains the information we are looking for?

MainForm

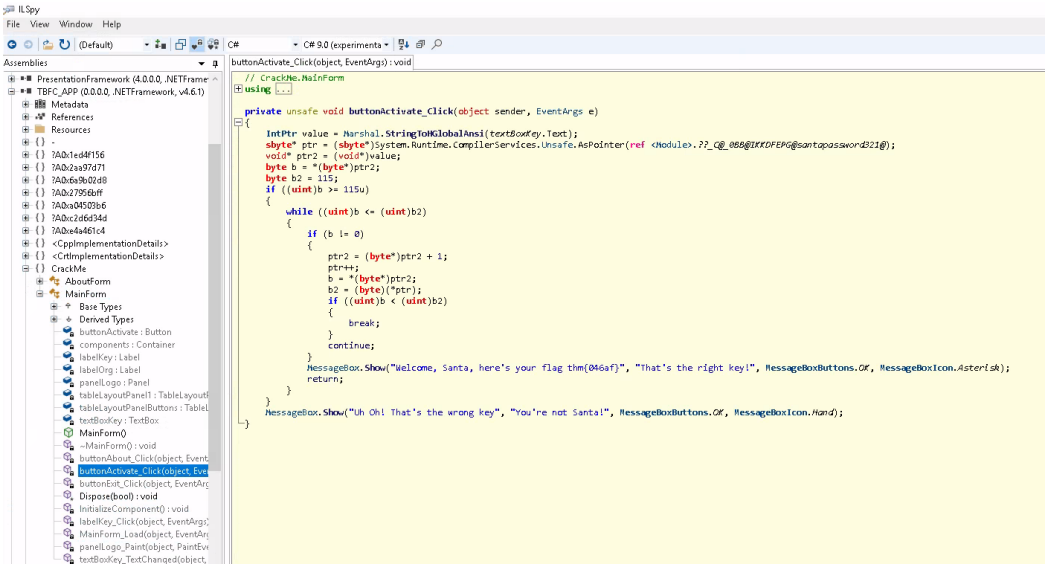
1. Under the TBFC app dropdown, select CrackMe and select MainForm.



Q: Which method within the form from Q4 will contain the information we are seeking?

buttonActivate_Click

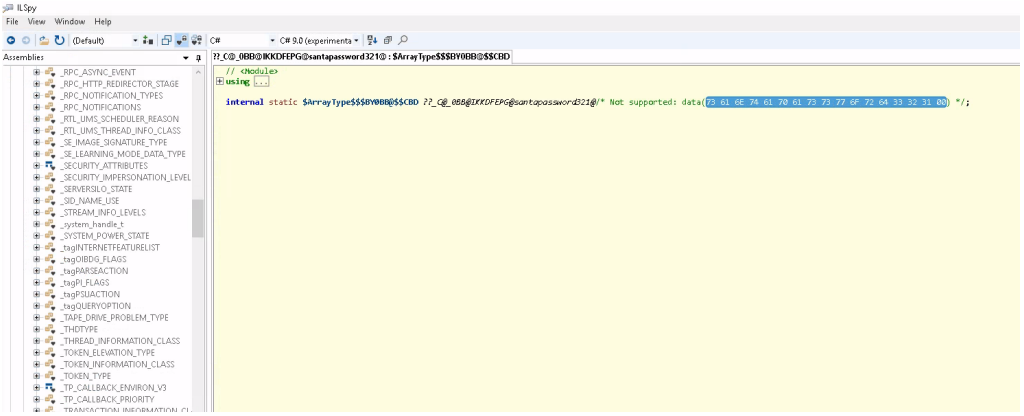
1. Go through the list of functions under MainForm until something related to the password is found. In this case, it would be in buttonActivate_Click.



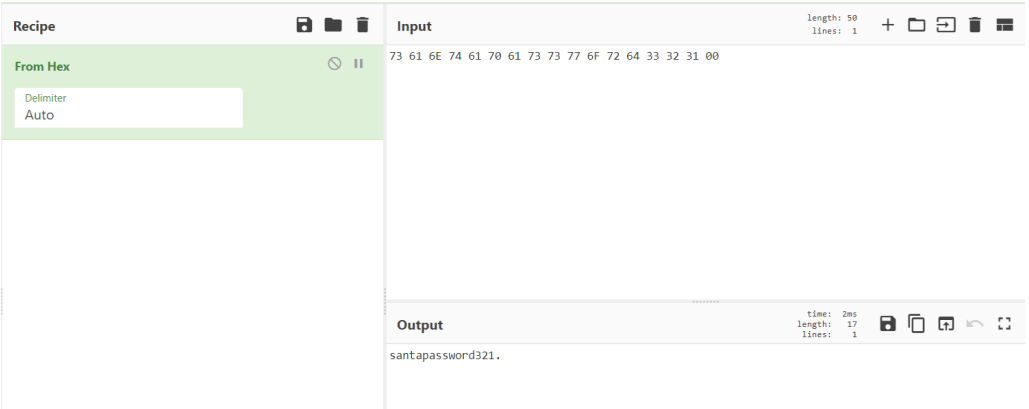
Q: What is Santa's password?

Santapassword321

1. Click on the line containing the string santapassword123 and we will be brought to this window.



2. Copy the hexadecimal code and decode it using a website of your choice. In this case, we will be using cyberchef.

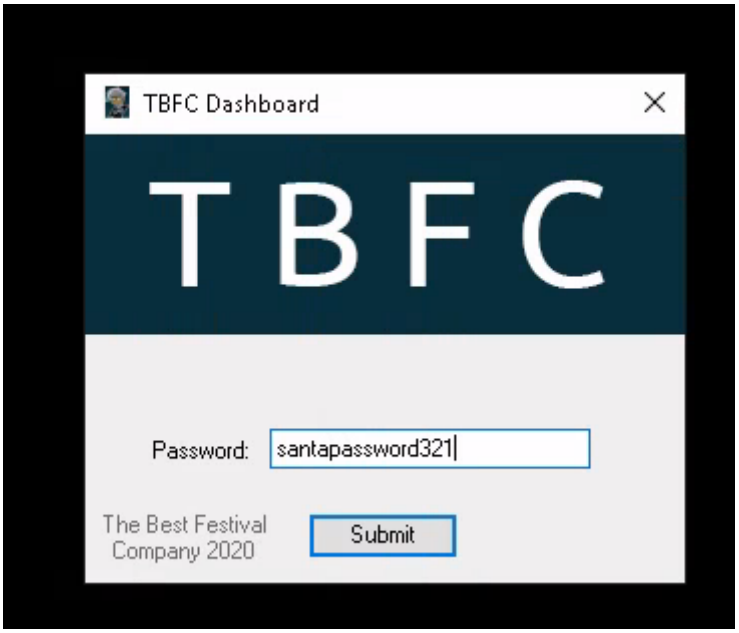


3. From that, we know that the password is santapassword321.

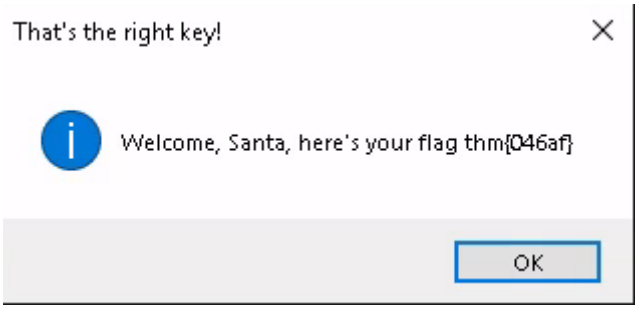
Q: Now that you've retrieved this password, try to login...What is the flag?

thm{046af}

1. While the flag can be seen directly from the code, we will try and login to retrieve it.
2. Open up the TBFC app and enter the password.



3. The flag will be shown to us.



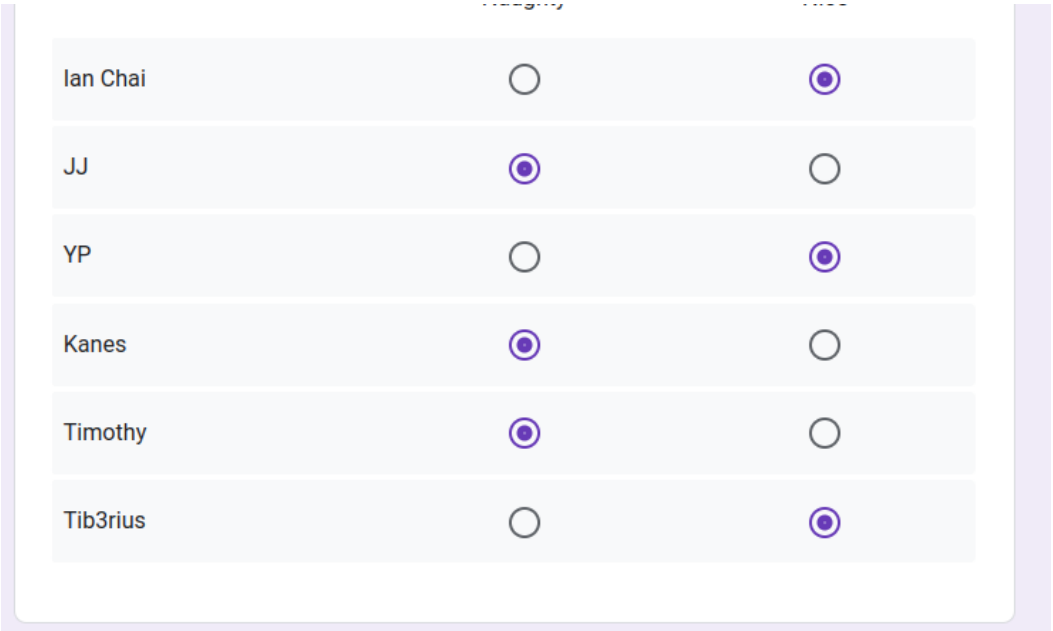
Thought Process/ Methodology

To complete this task, we needed a remote desktop client such as Remmina. Any remote desktop client of our preference can be used for this instance. After connecting to the remote desktop, things were fairly simple as ILSpy is already installed on the remote desktop. By opening up the TBFC app in ILSpy, we can get an idea on how the application was made. We looked for a function related to what happens when we submit a correct or wrong password in order to find out how the password authentication works for this application and if there are any loopholes. Sure enough, we find that the password to enter is hardcoded into the application and by using it we were able to login normally and retrieve the flag.

Day 19 - The Naughty or Nice List

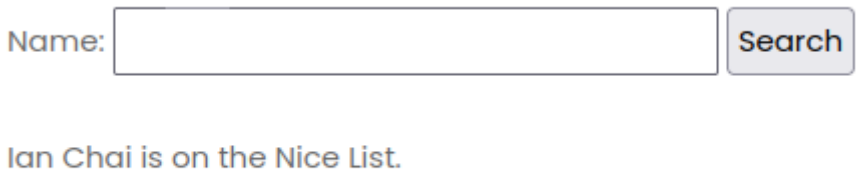
Q: `Which list is this person on?`

ANS:



Ian Chai	○	◉
JJ	◉	○
YP	○	◉
Kanes	◉	○
Timothy	◉	○
Tib3rius	○	◉

1. The first person on the list is Ian Chai.



2. The list told us that he is nice.

- Santa

Name: Search

JJ is on the Naughty List.

3. Repeat the steps above for the rest of the names.

Q: What is displayed on the page when you use "/?proxy=http%3A%2F%2Flist.hohoho%3A8080%2F"?

The requested URL was not found on this server.

- Santa

Name: Search

Not Found

The requested URL was not found on this server.

Q: What is displayed on the page when you use "/?proxy=http%3A%2F%2Flist.hohoho%3A80"?

Failed to connect to list.hohoho port 80: Connection refused

- Santa

Name: Search

Failed to connect to list.hohoho port 80: Connection refused

Q: What is displayed on the page when you use "/?proxy=http%3A%2F%2Flist.hohoho%3A22"?

Recv failure: Connection reset by peer

- Santa

Name: Search

Recv failure: Connection reset by peer

Q: What is displayed on the page when you use "/?proxy=http%3A%2F%2Flocalhost"?

Your search has been blocked by our security team.

Have a Merry Christmas! Ho ho ho!

- Santa

Name: Search

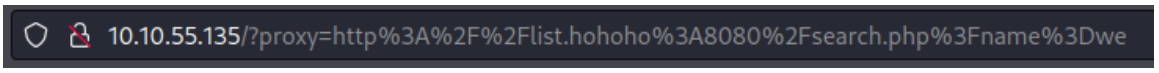
Your search has been blocked by our security team.

Q: What is Santa's password?

Be good for goodness sake!

1. First, we were presented with this website.

2. We can type a name on the search field.



3. We can use a url decoder to see the value of the urls

```
http://10.10.55.135/?proxy=http://list.hohoho:8080/search.php?name=fff
```

4. The developer has implemented a check to ensure that the hostname provided starts with "list.hohoho", and will block any hostnames that don't. So we can use a DNS subdomain which changes the url to

“http://10.10.55.135/?proxy=http%3A%2F%2Flist.hohoho.localtest.me” to localtestme and we will get this.

5.

If you need to make any changes to the Naughty or Nice list, you need to login.

I know you have trouble remembering your password so here it is: Be good for goodness sake!

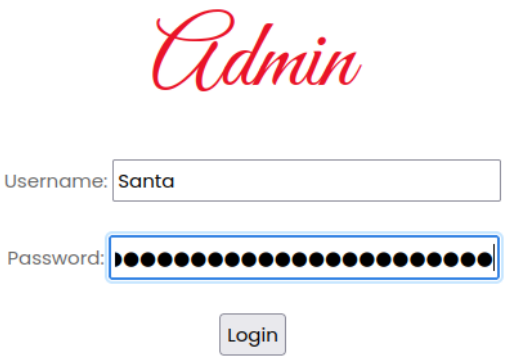
- Elf McSkidy

6. The password is stated in the website.

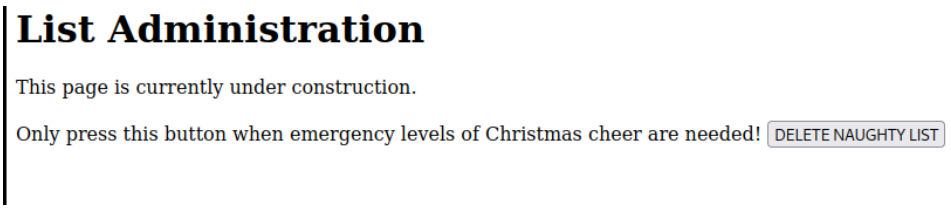
Q: What is the challenge flag?

THM{EVERYONE_GETS_PRESENTS}

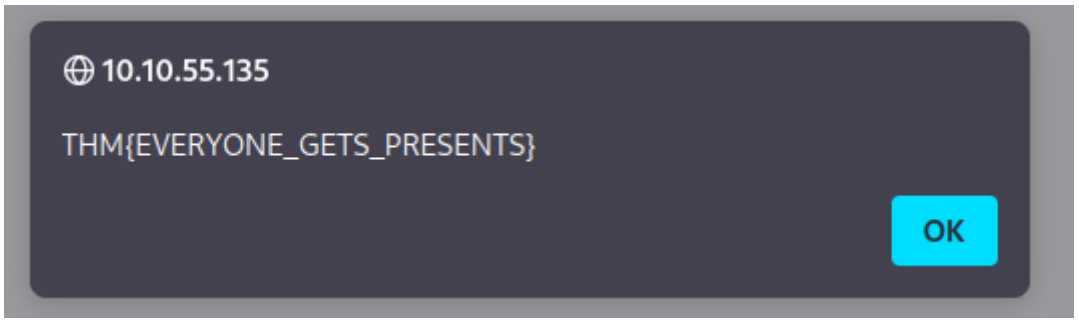
1. We can login by using the password above and the username as Santa



2. We will be presented with this.



3. Upon clicking the “DELETE NAUGHTY LIST” button, we are prompted with the flag.



<u>Thought Process/ Methodology</u>

To complete the first question we had to use brute force by using all attack methods we had to change the port request on the urls to 80 and 22 where we got an error. But when we use the localhost we get the message “your message has been blocked by our security team” which means the developer has blocked all hostnames.We will be presented with this except list.hohoho. To overcome this problem we can use DNS subdomains list.hohoho.localtest. Localtest is a domain that we have as a testing tool. We were able to get a note from Elf Mcskidy where the Password is stated. We can use the username and password provided to login and click on the “DELETE NAUGHTY LIST” button and the flag is prompted to us.

Day 20 - PowershELlf to the rescue

Tools Used : Kali, Terminal, Powershell

Q: Check the ssh manual. What does the parameter -l do?

login name

1. Use the command “ssh -help” to view all the options. The -l flag is for the login name parameter.

```
elfmceager@tbfc-day-17:~$ ssh -help
unknown option -- h
usage: ssh [-46AaCfGgKkMNnqsTtVvXxYy] [-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]]
           [user@]hostname [command]
```

Q: Search for the first hidden elf file within the Documents folder. Read the contents of this file. What does Elf 1 want?

2 front teeth

1. Use the command “ssh -l mceager 10.10.255.175” and enter the password “r0ckStar!” to log in, replacing the ip address with the ip address of your target.

```
File  Actions  Edit  View  Help
┌──(epsilon㉿1211100732)-[~]
└─$ ssh -l mceager 10.10.255.175
```

2. Next, launch powershell in the target instance.

```
c:\windows\system32\cmd.exe
File  Actions  Edit  View  Help
Microsoft Windows [Version 10.0.17763.737]
(c) 2018 Microsoft Corporation. All rights reserved.

mceager@ELFSTATION1 C:\Users\mceager>powershell
```

3. Change your directory to the documents folder.

```
PS C:\Users\mceager> Set-Location .\Documents\
PS C:\Users\mceager\Documents>
```

4. Use the command “Get-ChildItem” to see all files and directories under the current directory.

```
PS C:\Users\mceager\Documents> Get-ChildItem


    Directory: C:\Users\mceager\Documents


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----        11/23/2020   12:06 PM             22 elfone.txt
```

5. Use the “cat” command to see the contents of the file, the answer should be displayed here.

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

Q: Search on the desktop for a hidden folder that contains the file for Elf 2. Read the contents of this file. What is the name of that movie that Elf 2 wants?

Scrooged

1. Change the current directory to “Desktop”.

```
PS C:\Users\mceager\Documents> Set-Location ..\Desktop\
PS C:\Users\mceager\Desktop>
```

2. Use the command “Get-ChildItem -Hidden” to see all hidden files and directories.

```
PS C:\Users\mceager\Desktop> Get-ChildItem -Hidden


    Directory: C:\Users\mceager\Desktop


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
d--h--        12/7/2020  11:26 AM                elf2wo
-a-hs-        12/7/2020  10:29 AM            282 desktop.ini
```

3. Navigate to the “elf2wo” folder and list down all files inside.

```
PS C:\Users\mceager\Desktop> cd elf2wo
PS C:\Users\mceager\Desktop\elf2wo> ls


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


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

4. Use the “cat” command to view the contents of the file.

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

Q: Search the Windows directory for a hidden folder that contains files for Elf 3. What is the name of the hidden folder? (This command will take a while)

3lfthr3e

1. Navigate to the “Windows” directory.

```
PS C:\Users\mceager\Desktop\elf2wo> cd \Windows
PS C:\Windows>
```

2. After some trial and error, you’ll find that the file will be in the “System32” sub directory so navigate there.

```
PS C:\Windows> cd System32
```

3. Use the “Get-ChildItem -Directory -Hidden” to find the hidden directory.

```
PS C:\Windows\System32> Get-ChildItem -Directory -Hidden


    Directory: C:\Windows\System32


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
d--h--        11/23/2020   3:26 PM                3lfthr3e
d--h--        11/23/2020   2:26 PM                GroupPolicy
```

Q: How many words does the first file contain?

9999

1. Move to the “3lfthr3e” directory and list down all hidden files or directories.

```
PS C:\Windows\System32\3lfthr3e> Get-ChildItem -Hidden


    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
```

2. Use the command “Get-Content -Path 1.txt | Measure-Object -Word” to find the number of words in the file.

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

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

Q: What 2 words are at index 551 and 6991 in the first file?

Red Ryder

1. Use the command “” to see the word at the specified indexes, replacing “(Get-Content -Path 1.txt)[index]” with the specified index.

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

Q: This is only half the answer. Search in the 2nd file for the phrase from the previous question to get the full answer. What does Elf 3 want? (use spaces when submitting the answer)

Red Ryder BB Gun

1. Use the command “Select-String -Path 2.txt -Pattern ‘ryder’” to search for words in the file containing the string ‘ryder’.

```
PS C:\Windows\System32\3lfthr3e> Select-String -Path 2.txt -Pattern "ryder"

2.txt:558704:redryderbbgun
```

<u>Thought Process/ Methodology</u>

After logging in to the given account, we just navigated to the folder specified and listed down all the contents of that directory. With that, we can already find the first file “e1fone.txt” and see its contents. The same process is repeated for the next question, but we are searching for a hidden directory instead. To find the third hidden directory, we need to check through many folders in the windows directory until we eventually find it hidden in the “System32” directory. There, we repeat the same process as the previous step and find hidden directories. Sure enough, we eventually find the 3rd hidden directory. After that, we used various commands to

identify the word count inside a file, find words at a specified index, and search for words in a file containing specific keywords. With all of that, we managed to find what Elf 3 wanted.