

PSP0201

Week 5

Writeup

Group Name:HUSTLERS

Members

ID	Name	Role
1211100708	Muhammad Faiz BIn Mohd Fauzi	leader
1211101962	Barath A L Saravanan	member
1211101804	AKHILESHNAIDU A/L JAYA KUMAR	MEMBER

Day 16 - [Scripting] Help! Where is Santa?

Tools-Tryhackme & Google

Solution/walkthrough:

Question 1

Port number from web server by using nmap

```
Scanning 10.10.110.46:80 - Host: 10.10.110.46
Discovered open port 80/tcp on 10.10.110.46
```

Question 2

Templates are being used at the top left of the website

```
BULMA Home Examples View Source. Template not my own.
```

Question 3

Directory for the API

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

Question 4

Raw Data returned if no parameters are entered

```
TryHackMe | Learn Cy... TryHackMe Supp
{"detail":"Not Found"}
```

Question 5 & 6

Santa's location and correct API key

```
api_key 57
{"item_id":57,"q":"Winter Wonderland, Hyde Park, London."}
api_key 50
```

## DAY 17- [Reverse Engineering] ReverseELFneering

### TOOLS - Mozilla//Terminal

#### Question 1

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:

- $(Rb, Ri) = \text{MemoryLocation}[Rb + Ri]$
- $D(Rb, Ri) = \text{MemoryLocation}[Rb + Ri + D]$
- $(Rb, Ri, S) = \text{MemoryLocation}[Rb + S * Ri]$
- $D(Rb, Ri, S) = \text{MemoryLocation}[Rb + S * Ri + D]$

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

## Question 2

The screenshot shows a web browser window with the URL `https://tryhackme.com/room/learnCyberin25Days`. The page displays a task titled "aocmmre1" with an IP address of 10.10.144.22 and an expiration time of 46m 11s. The task description includes a terminal session where the user runs `./file1` and receives the output: "the value of a is 4, the value of b is 5 and the value of c is 9". The description also mentions that the program shows 3 variables (a, b, c) where c is the sum of a and b. The user is instructed to run the command `r2 -d ./file1` to analyze the program in debugging mode. The user then runs `aa` to analyze the program, and the output shows the main function at address 0x00400a30.

Task Title: aocmmre1  
IP Address: 10.10.144.22  
Expires: 46m 11s

Password: adventofcyber

Let's proceed to run through how Radare2 works exactly. Although you shouldn't do this if the program is unknown, it is safe for us to execute to see what should be happening like so:

```
ashu@ashu-Inspiron-5379 ~/B/T/C/Christmas-rc> ./file1
the value of a is 4, the value of b is 5 and the value of c is 9
```

The above program shows that there are 3 variables (a, b, c) where c is the sum of a and b.

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.

I.e. For general help, we can run `?` or if we wish to understand more about a specific feature, we could provide `as?`

3. Computer says...Done?!

Once the analysis is complete, you would want to know where to start analysing from - most programs have an entry point defined as main. To find a list of the functions run:

```
[0x00400a30]> afl | grep main
```

## Command to analyse the program in radare2

## Question 3

The screenshot shows a web browser window with the URL `https://tryhackme.com/room/learnCyberin25Days`. The page displays a task titled "aocmmre1" with an IP address of 10.10.144.22 and an expiration time of 45m 15s. The task description includes a terminal session where the user runs `r2 -d ./file1` and receives the output: "the value of a is 4, the value of b is 5 and the value of c is 9". The description also mentions that the program shows 3 variables (a, b, c) where c is the sum of a and b. The user is instructed to run the command `r2 -d ./file1` to analyze the program in debugging mode. The user then runs `aa` to analyze the program, and the output shows the main function at address 0x00400a30.

Task Title: aocmmre1  
IP Address: 10.10.144.22  
Expires: 45m 15s

Password: adventofcyber

Let's proceed to run through how Radare2 works exactly. Although you shouldn't do this if the program is unknown, it is safe for us to execute to see what should be happening like so:

```
ashu@ashu-Inspiron-5379 ~/B/T/C/Christmas-rc> ./file1
the value of a is 4, the value of b is 5 and the value of c is 9
```

The above program shows that there are 3 variables (a, b, c) where c is the sum of a and b.

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.

I.e. For general help, we can run `?` or if we wish to understand more about a specific feature, we could provide `as?`

3. Computer says...Done?!

Once the analysis is complete, you would want to know where to start analysing from - most programs have an entry point defined as main. To find a list of the functions run:

```
[0x00400a30]> afl | grep main
```

The line starting with sym.main indicates we're looking at the main function. The next 3 lines are used to represent the variables stored in the function. The second column indicates that they are integers(int), the 3rd column specifies the name that `r2` uses to reference them and the 4th column shows the actual memory location.

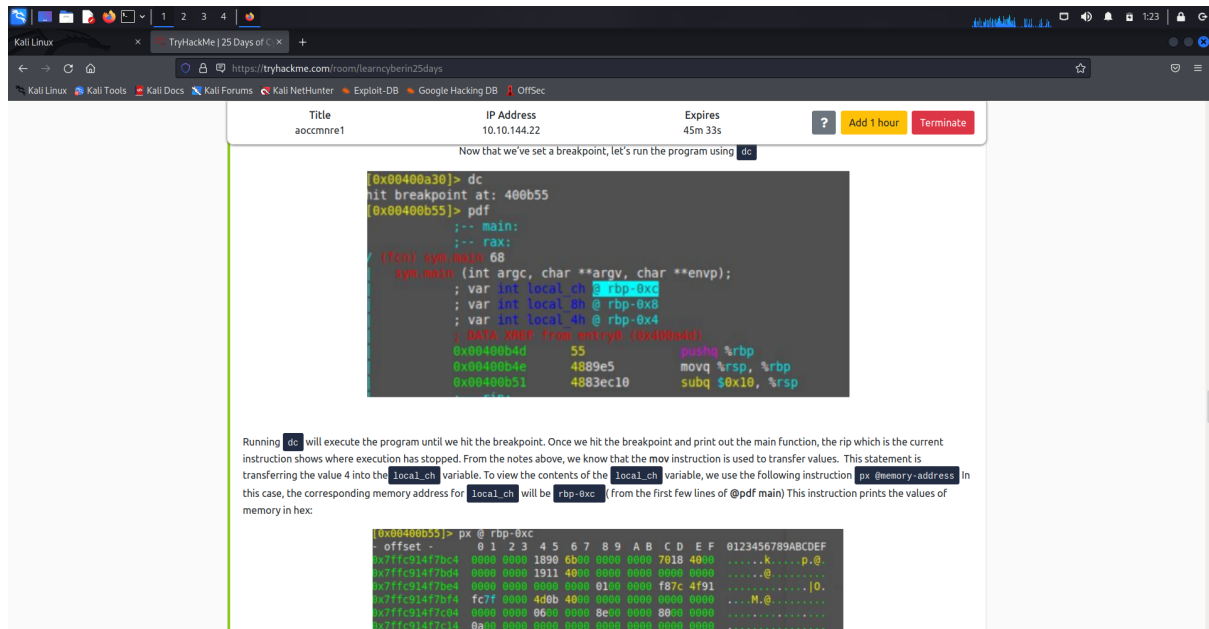
The first 3 instructions are used to allocate space on that stack (ensures that there's enough room for variables to be allocated and more). We'll start looking at the program from the 4th instruction (`movl $4, %eax`). We want to analyse the program while it runs and the best way to do this is by using breakpoints.

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.

```
[0x00400a30]> pdf @main
;-- main:
;[call] sym.main: 68
```

Command to set a breakpoint in radare2

#### Question 4



The screenshot shows a web browser window with the URL <https://tryhackme.com/room/learnCyberin25days>. The page displays a session titled "aocmmre1" with an IP address of 10.10.144.22 and an expiration time of 45m 33s. The session content shows a radare2 session where a breakpoint is set at address 400b55. The user then runs the program using the 'dc' command. The assembly instructions are displayed, showing a 'mov' instruction that transfers the value 4 into the 'local\_ch' variable. The user then uses the 'px' command to view the contents of the 'local\_ch' variable, which is located at memory address 0xc.

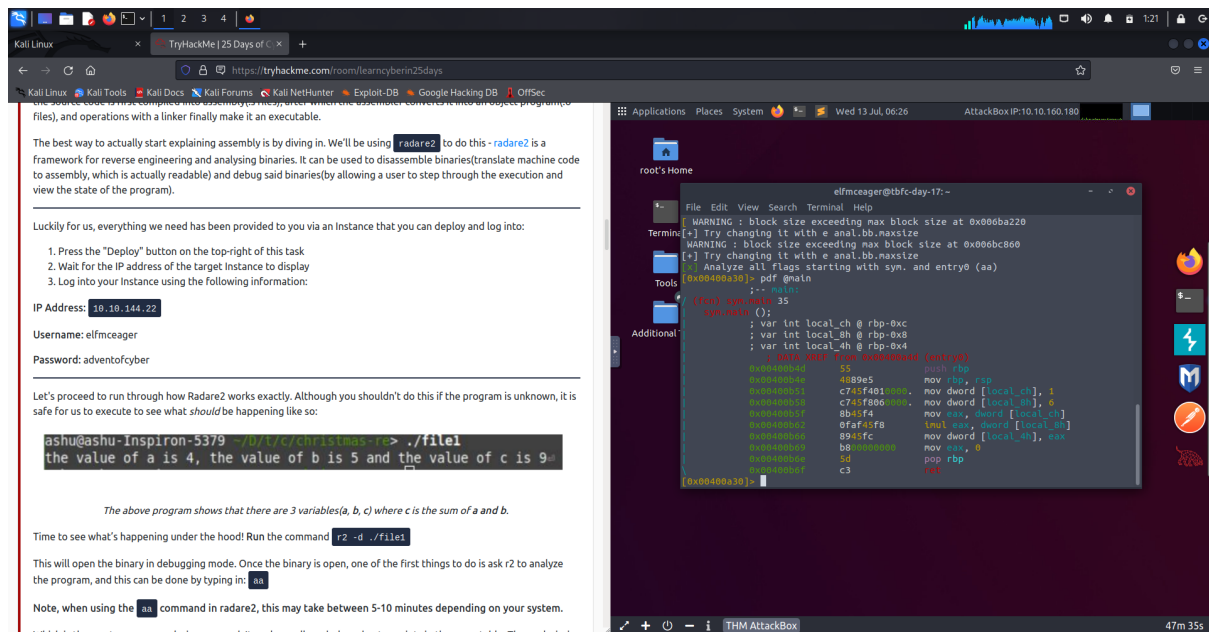
```
[0x00400a30]> dc
hit breakpoint at: 400b55
[0x00400b55]> pdf
;-- main:
;-- rax:
(intn) sym.main 68
sym.main (int argc, char **argv, char **envp);
; var int local_ch @ rbp-0xc
; var int local_8h @ rbp-0x8
; var int local_4h @ rbp-0x4
; DATA 00000000 from entry0 (0x400a4d)
0x00400b4d 55          pushq %rbp
0x00400b4e 4889e5      movq %rsp, %rbp
0x00400b51 4883ec10    subq $0x10, %rsp
```

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:

```
[0x00400b55]> px @ rbp-0xc
. offset - 0 1 2 3 4 5 6 7 8 9 A B C D E F 0123456789ABCDEF
0x77fc914f7b64 0000 0000 1890 60a0 0000 0000 7018 4000 .....k....p.0
0x77fc914f7b65 0000 0000 1911 40a0 0000 0000 0000 0000 .....@.....
0x77fc914f7b66 0000 0000 0000 0000 0000 0000 f87c 4f91 .....10.
0x77fc914f7b67 fc7f 0000 4d0b 4000 0000 0000 0000 0000 .....M.0
0x77fc914f7b68 0000 0000 0000 0000 8000 0000 8000 0000 .....
0x77fc914f7b69 0a00 0000 0000 0000 0000 0000 0000 0000 .....
0x77fc914f7b6a 0a00 0000 0000 0000 0000 0000 0000 0000 .....
```

Command to execute the program until we hit a breakpoint

#### Question 5



Answers are 1,6,6

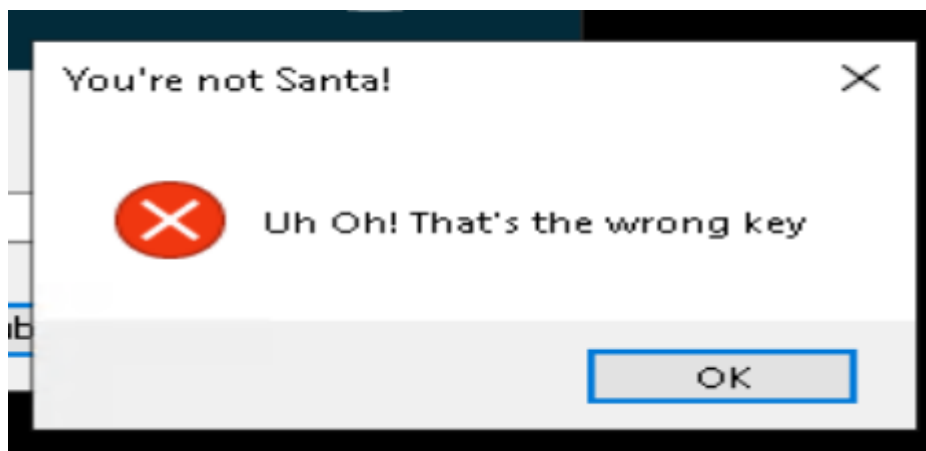
Day 18 - [Reverse Engineering] The Bits of Christmas

Tools used: TBFC, Attackbox, ILspy, Cyberchef

Solution/walkthrough:

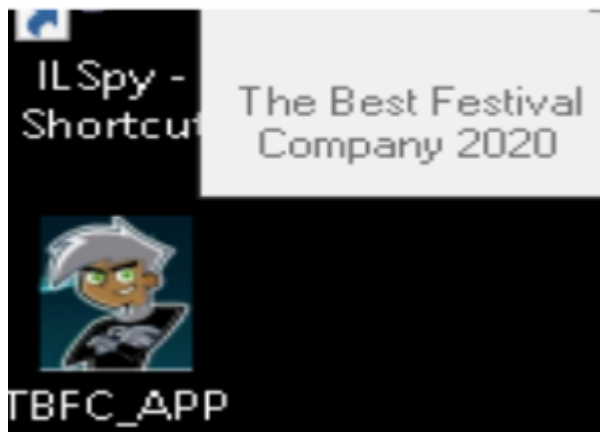
Question 1

This message showed up when i use wrong password



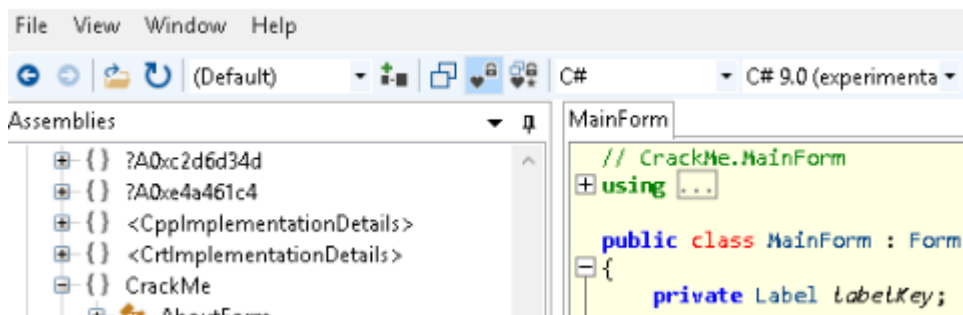
## Question 2

TBFC stands for The Best Festival Company



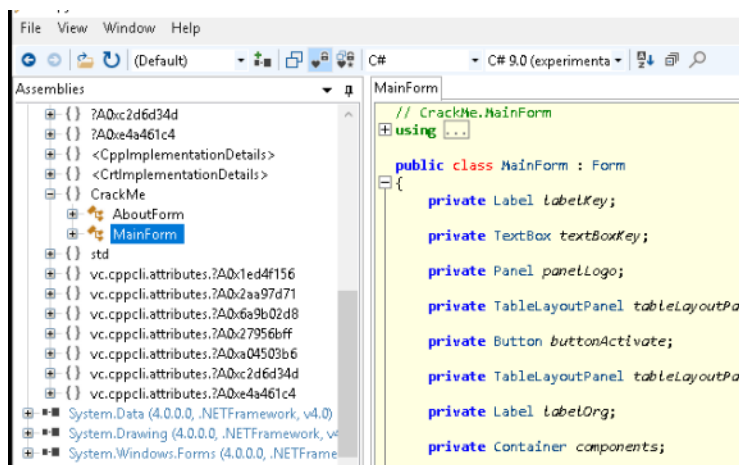
## Question 3

the module that catches your attention is CrackMe



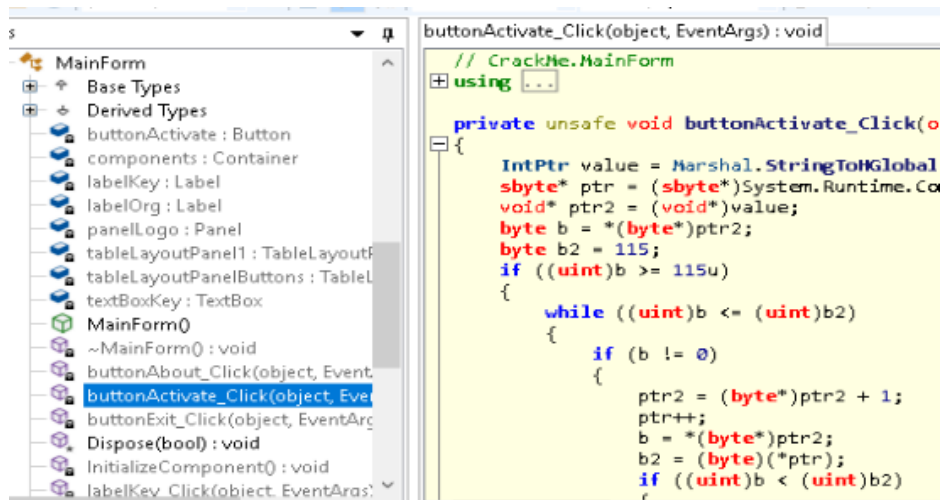
## Question 4

Mainform contains the information we are looking for



### Question 5

buttonActivate\_Click method within the form from Q4 will contain the information we are seeking



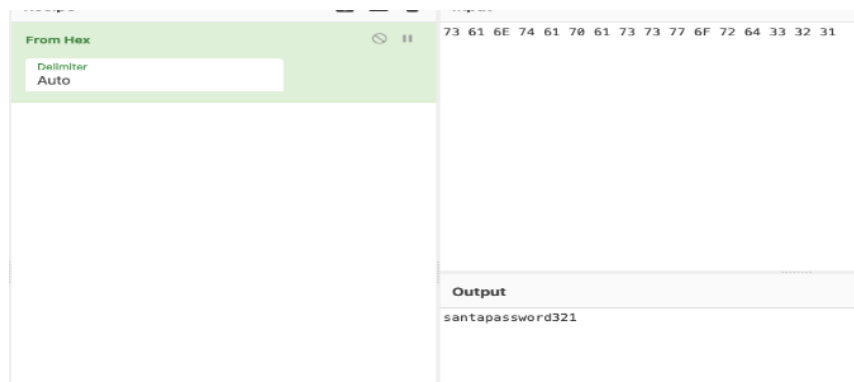
The screenshot shows the Visual Studio IDE with the 'MainForm' project selected in the Solution Explorer. The 'buttonActivate\_Click' method is highlighted in the 'Derived Types' list. The code in the editor is as follows:

```
buttonActivate_Click(object, EventArgs) : void
// CrackMe.MainForm
using ...

private unsafe void buttonActivate_Click(object o)
{
    IntPtr value = Marshal.StringToHGlobal(1024);
    sbyte* ptr = (sbyte*)System.Runtime.InteropServices.Marshal.GetObjectForHGlobal(value, 0);
    void* ptr2 = (void*)ptr;
    byte b = *(byte*)ptr2;
    byte b2 = 115;
    if ((uint)b >= 115u)
    {
        while ((uint)b <= (uint)b2)
        {
            if (b != 0)
            {
                ptr2 = (byte*)ptr2 + 1;
                ptr++;
                b = *(byte*)ptr2;
                b2 = (byte)(*ptr);
                if ((uint)b < (uint)b2)
                {
                    continue;
                }
            }
        }
    }
}
```

### Question 6

Decode the code we get by using Cyberchef

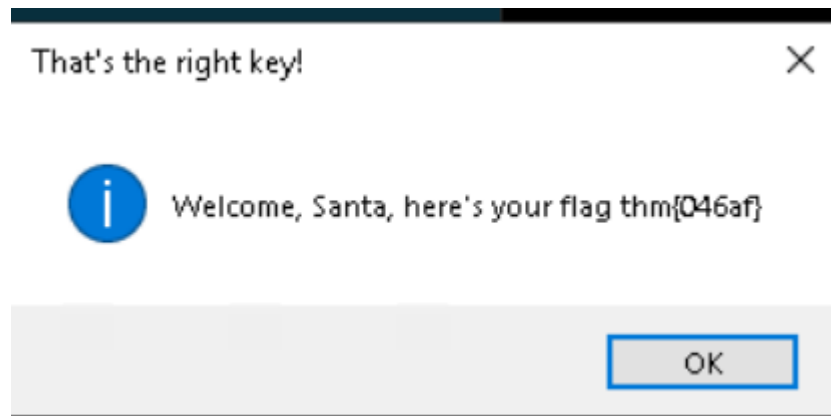


The screenshot shows the CyberChef web interface. The 'From Hex' tab is selected, and the input is a hex string: 73 61 6E 74 61 70 61 73 73 77 6F 72 64 33 32 31. The 'Output' section shows the decoded result: santapassword321.

### Question 7

When i try to login, the flag i get is





Day 19 - [Web Exploitation] The Naughty or Nice List

Tools used:mozilla

solution/walkthrough:

Question 1

Test out some name to know whether there are in naughty or nice list

## The 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:

## The 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:

John is on the Nice List.

## Question 2,3,4,5

### Alrd in thm

2. Enter a name in the form and click the "Search" button. When the page loads, it should tell you whether that name is on the Naughty List or the Nice List. Notice that the URL for the page looks something like this: [http://MACHINE\\_IP/?proxy=http%3A%2F%2Flist.hohoho%3A8080%2Fsearch.php%3Fname%3DTib3rius](http://MACHINE_IP/?proxy=http%3A%2F%2Flist.hohoho%3A8080%2Fsearch.php%3Fname%3DTib3rius)

If we use a URL decoder on the value of the "proxy" parameter, we get: <http://list.hohoho:8080/search.php?name=Tib3rius>

Since "list.hohoho" is not a valid hostname on the Internet (.hohoho is not a [top-level domain](#)), this hostname likely refers to some back-end machine. It seems that the web app works by taking this URL, making a request at the back-end, and then returning the result to the front-end web app. If the developer has not been careful, we may be able to exploit this functionality using Server-Side Request Forgery (SSRF).

3. The most obvious thing we can try to do first is to fetch the root of the same site. Browse to: [http://MACHINE\\_IP/?proxy=http%3A%2F%2Flist.hohoho%3A8080%2F](http://MACHINE_IP/?proxy=http%3A%2F%2Flist.hohoho%3A8080%2F)

This seems to have potential, as in place of the original "Tib3rius is on the Nice List." message, we instead see "Not Found. The requested URL was not found on this server." This seems like a generic 404 message, indicating that we were able to make the server request the modified URL and return the response.

There are many things we could do now, such as trying to find valid URLs for the "list.hohoho" site. We could also try changing the port number from 8080 to something else, to see if we can connect to any other services running on the host, even if these services are not web servers.

4. Try changing the port number from 8080 to just 80 (the default HTTP port): [http://MACHINE\\_IP/?proxy=http%3A%2F%2Flist.hohoho%3A80](http://MACHINE_IP/?proxy=http%3A%2F%2Flist.hohoho%3A80)

The message now changes to "Failed to connect to list.hohoho port 80: Connection refused" which suggests that port 80 is not open on list.hohoho.

5. Try changing the port number to 22 (the default SSH port): [http://MACHINE\\_IP/?proxy=http%3A%2F%2Flist.hohoho%3A22](http://MACHINE_IP/?proxy=http%3A%2F%2Flist.hohoho%3A22)

The message now changes to "Recv failure: Connection reset by peer" which suggests that port 22 is open but did not understand what was sent (this makes sense, as sending an HTTP request to an SSH server will not get you anywhere!)

Enumerating open ports via SSRF can be performed in this manner, by iterating over common ports and measuring the differences between responses. Even in cases where error messages aren't returned, it is often possible to detect which ports are open vs closed by measuring the time each request takes to complete.

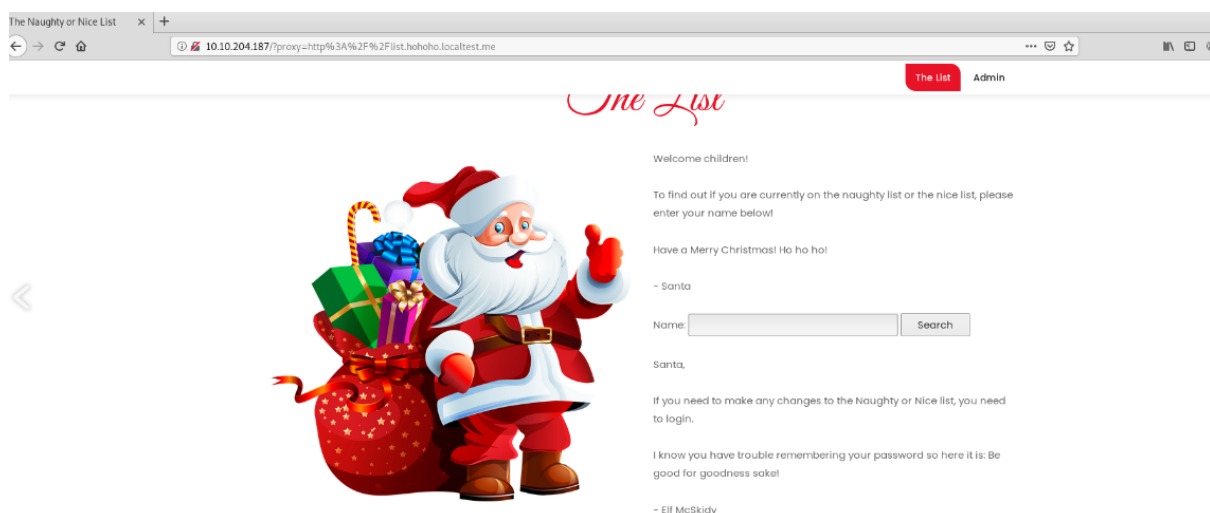
6. Another thing we can try to do with SSRF is access services running locally on the server. We can do this by replacing the list.hohoho hostname with "localhost" or "127.0.0.1" (among others). Try this now: [http://MACHINE\\_IP/?proxy=http%3A%2F%2Flocalhost](http://MACHINE_IP/?proxy=http%3A%2F%2Flocalhost)

Oops! It looks like the developer has a check in place for this, as the message returned says "Your search has been blocked by our security team."

Indeed, if you try other hostnames (e.g. 127.0.0.1, example.com, etc.) they will all be blocked. The developer has implemented a check to ensure that the hostname provided starts with "list.hohoho", and will block any hostnames that don't.

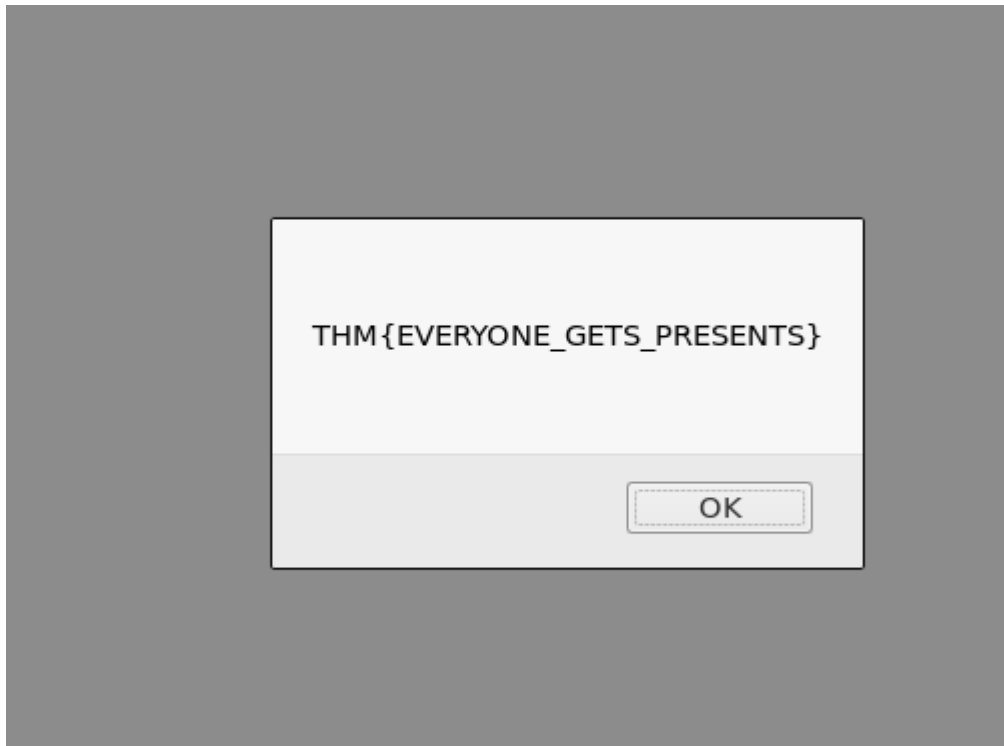
## Question 6

### Get password from message by Elf mcskidly with access the web server



### Question 7

Delete the naughty list and capture the flag



Day 20 - [Blue Teaming] Powershell to the rescue

Tools:ssh,powershell,linux,firefox

solution/walkthrough:

### Question 1

Check ssh manual

*-l login\_name*

Specifies the user to log in as on the remote machine. This also may be specified on a per-host basis in the configuration file.

## Question 2

Read content in hidden files with get-content

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

## Question 3

Search hidden directory and read the content file

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

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

## Question 4

Search hidden directory in entire windows

```
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
```

## Question 5

Use command measure-object-word

```
PS C:\Users\mceager\Desktop\elf2wo> Get-Content C:\Windows\System32\3lfthr3e\1.txt | Measure-Object -Word

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

## Question 6

Use get-content for specific file

```
PS C:\Users\mceager\Desktop\elf2wo> (Get-Content C:\Windows\System32\3lfthr3e\1.txt)[551]
Red
PS C:\Users\mceager\Desktop\elf2wo> (Get-Content C:\Windows\System32\3lfthr3e\1.txt)[6991]
Ryder
PS C:\Users\mceager\Desktop\elf2wo> █
```

## Question 7

Get second file

```
PS C:\Users\mceager\Desktop\elf2wo> Select-String -Path C:\Windows\System32\3lfthr3e\2.txt -Pattern 'redryder'
C:\Windows\System32\3lfthr3e\2.txt:558704:redryderbbgun
PS C:\Users\mceager\Desktop\elf2wo> █
```