

PSP0201

WEEKLY WRITE UP

WEEK 5

GROUP 7

Group Name : Sang Haekeo (The Hackers)

Sang

- Taken from a Malay word , [sang](#) , meaning 'the'

Haekeo

- Taken from a Korean word , [해커](#) , meaning 'hacker'

| ID | NAME | ROLE |
|------------|-------------------------------------|--------|
| 1211102162 | AMILIA NADZEERA BINTI BAHARUDIN | Leader |
| 1211100930 | KU NAJWA SYAUQINA BINTI KU AZRIN | Member |
| 1211101693 | SAVITHA MURUGUMUNISEGARAN | Member |

Day 16: Scripting ; Help! Where is Santa?

Question 1: What is the port number for the web server?

The port number can be found using nmap and syntax , host number is shown.

Answer 1: 80

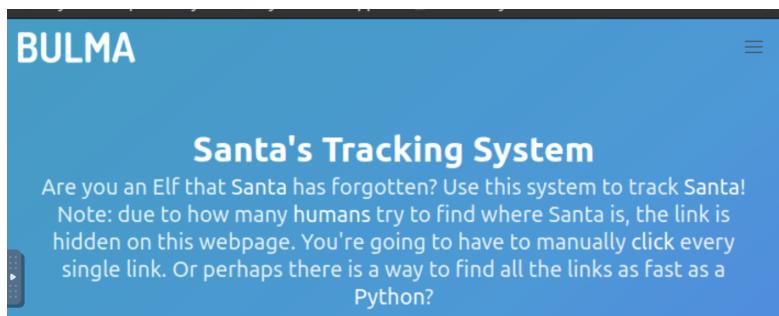
```
root@ip-10-10-96-38:~#
File Edit View Search Terminal Help
root@ip-10-10-96-38:~# nmap 10.10.223.177
Starting Nmap 7.60 ( https://nmap.org ) at 2022-07-13 06:36 BST
Nmap scan report for ip-10-10-223-177.eu-west-1.compute.internal (10.10.223.177)
Host is up (0.0079s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
MAC Address: 02:9E:7D:3D:0C:0B (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.73 seconds
```

Question 2: What templates are being used?

For this , we checked the top left of the website

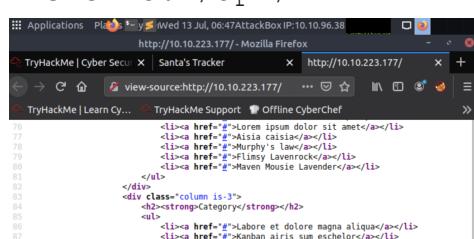
Answer 2: BULMA



Question 3: Without using enumeration tools such as Dirbuster, what is the directory for the API?

Open the developers tools in your browser , then open your elements tab. Scroll down and check the footer to find the HTML tag

Answer 3: /api/

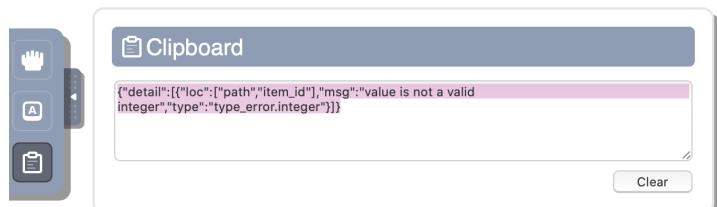


Question 4: Go to the API endpoint. What is the Raw Data returned if no parameters are entered?

the raw data returned was retrieved from the endpoint of the API.

Answer 4: {"detail": [{"loc": ["path", "item_id"], "msg": "value is not a valid integer", "type": "type_error.integer"}]}

```
{"detail": [{"loc": ["path", "item_id"], "msg": "value is not a valid integer", "type": "type_error.integer"}]}
```



Question 5: Where is Santa right now?

Answer 5: Winter Wonderland , Hyde Park , London

Question 6: Find out the correct API key. Remember, this is an odd number between 0-100.

Answer 6: 57

For both questions , we had to try our luck and key in an odd

```
$ cat apibrute.py
#!/usr/bin/env python3
# Import the libraries we downloaded earlier
# if you try importing without installing them, this step will fail
import requests

for api_key in range (1,100,2):
    print(f"api_key: {api_key}")
    html = requests.get(f"http://10.10.231.236/api/{api_key}")
    print(html.text)
```

```
{"item_id":49,"q":"Error. Key not valid!"}
api_key: 51
{"item_id":51,"q":"Error. Key not valid!"}
api_key: 53
{"item_id":53,"q":"Error. Key not valid!"}
api_key: 55
{"item_id":55,"q":"Error. Key not valid!"}
api_key: 57
{"item_id":57,"q":"Winter Wonderland, Hyde Park, London."}
api_key: 59
{"item_id":59,"q":"Error. Key not valid!"}
api_key: 61
{"item_id":61,"q":"Error. Key not valid!"}
api_key: 63
{"item_id":63,"q":"Error. Key not valid!"}
api_key: 65
{"item_id":65,"q":"Error. Key not valid!"}
api_key: 67
```

Day 17 : Reverse Engineering ; ReverseELFengineering

Question 1: Match the data type with the size in bytes

For this question , we followed the tale provided in TryHackMe .

Answer 1 :

- Byte - 1
- Word - 2
- Double Word - 4
- Quad - 8
- Single Precision - 4
- Double Precision - 8

| 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 |

Question 2: What is the command to analyse the program in radare2?

The answer was provided in the text shown below.

Answer 2: pdf@main

As seen here, there actually is a function at main. Let's examine the assembly code at main by running the command `pdf @main` Where `pdf` means print disassembly function. Doing so will give us the following view:



A screenshot of the radare2 debugger interface. The assembly code for the `main` function is displayed. The code starts with `main:`, followed by `pushq %rbp`, `movq %rsp, %rbp`, `subq $8, %rbp`, and `movl $4, %eax`. There are also some comments like `## stack frame setup (standard)` and `## stack frame cleanup (standard)`. The assembly code is color-coded with syntax highlighting.

Question 3: What is the command to set a breakpoint in radare2?

The answer was provided in the text shown below.

Answer 3: db

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(`i32`), 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`). 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.

Question 4: What is the command to execute the program until we hit a breakpoint?

The answer was provided in the text shown below.

Answer 4: dc

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:

```
[0x00400055]> px @ rbp-0xc
offset -      0 1 2 3 4 5 6 7 8 9 A B C D E F  0123456789ABCDEF
```

Question 5: What is the value of `local_ch` when its corresponding `movl` instruction is called (first if multiple)?

The value is 1 as the instruction stated `mov dwod [local_ch], 1`
Answer 5: 1

```
lfmceager@tbfc-day-17:~/s$ r2 -d ./challenge1
Process with PID 1615 started...
> attach 1615 1615
> bin.baddr 0x00400000
> using 0x400000
Warning: Cannot initialize dynamic strings
> sm.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]> 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]>
```

Question 6: What is the value of `eax` when the `imull` instruction is called?

Answer 6: 6

```

lfmceager@tbfc-day-17:~$ r2 -d ./challenge1
Process with PID 1615 started...
+ attach 1615 1615
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]> 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]>

```

Question 7: What is the value of local_4h before eax is set to 0?

Answer 7: 6

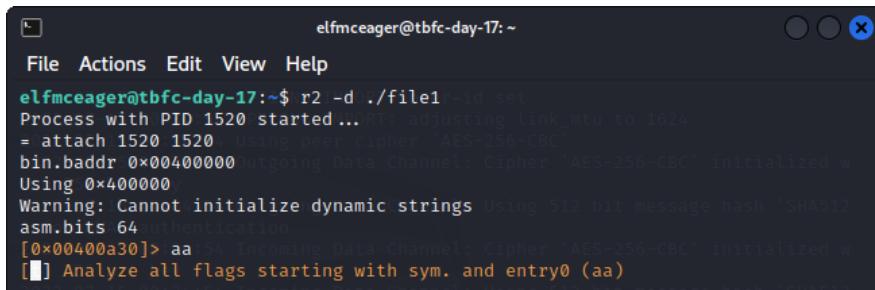
```

lfmceager@tbfc-day-17:~$ r2 -d ./challenge1
Process with PID 1615 started...
+ attach 1615 1615
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]> 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]>

```

[Day 17] – Reverse Engineering – Reverse ELFneering

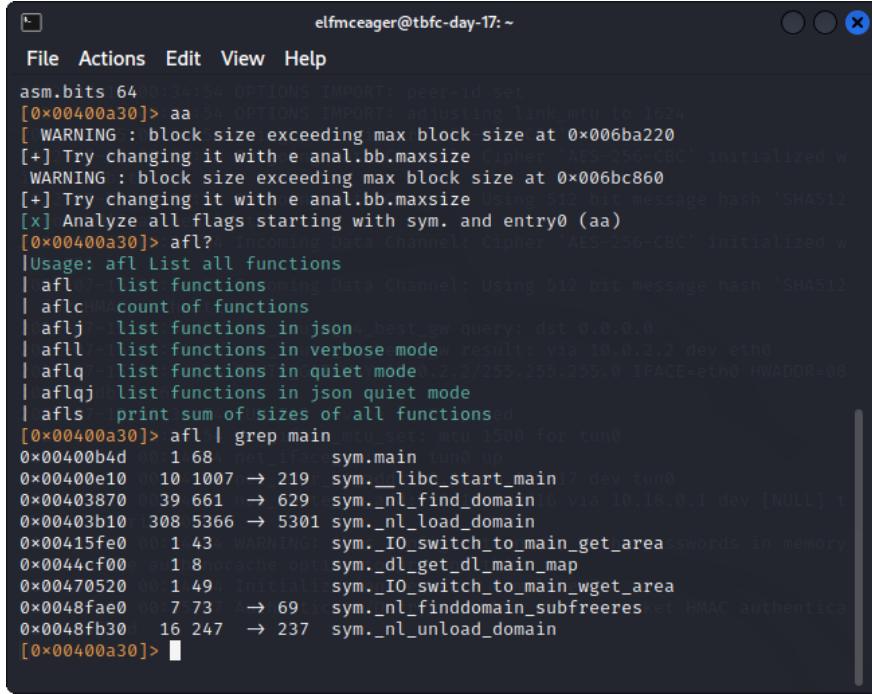
Log into the instance with the given credentials. Opens the binary files in debugging mode using command r2 -d ./file1



```
elfmceager@tbfc-day-17:~$ r2 -d ./file1 -id set
Process with PID 1520 started...RTT: adjusting link_mtu to 1624
= attach 1520 1520 Using peer cipher 'AES-256-CBC'
bin.baddr 0x00400000 Outgoing Data Channel: Cipher 'AES-256-CBC' initialized w
Using 0x400000
Warning: Cannot initialize dynamic strings Using 512 bit message hash 'SHA512
asm.bits 64 authentication
[0x00400a30]> aa 54 Incoming Data Channel: Cipher 'AES-256-CBC' initialized w
[!] Analyze all flags starting with sym. and entry0 (aa)
```

Once the analyzing completes we can seek out the entry point to the file.

```
afl | grep main
```



The terminal window shows the following AFL analysis session:

```
elfmceager@tbfc-day-17: ~
File Actions Edit View Help
asm.bits 64 0:34:54 OPTIONS IMPORT: peer-id set
[0x00400a30]> aa 54 OPTIONS IMPORT: adjusting link_mtu to 1624
[+] WARNING : block size exceeding max block size at 0x006ba220
[+] Try changing it with e anal.bb.maxsize Cipher 'AES-256-CBC' initialized w
WARNING : block size exceeding max block size at 0x006bc860
[+] Try changing it with e anal.bb.maxsize Using 512 bit message hash 'SHA512'
[x] Analyze all flags starting with sym. and entry0 (aa)
[0x00400a30]> afl? incoming Data Channel: Cipher 'AES-256-CBC' initialized w
Usage: afl List all functions
| afl list functions
| aflc count of functions
| aflj list functions in json
| afrm query: dst 0.0.0.0
| afrm list functions in verbose mode result: via 10.0.2.2 dev eth0
| aflq list functions in quiet mode .2.2/255.255.255.0 IFACE=eth0 HWADDR=08
| aflqj list functions in json quiet mode
| aqls print sum of sizes of all functions
[0x00400a30]> afl | grep main
main mtu_set tun0 mtu 1500 for tun0
0x00400b4d 0 1 68 net_ifaces sym.main tun0 up
0x00400e10 0 10 1007 → 219 sym._libc_start_main 7 dev tun0
0x00403870 0 39 661 → 629 sym._nl_find_domain 16 via 10.18.0.1 dev [NULL]
0x00403b10 0 308 5366 → 5301 sym._nl_load_domain
0x00415fe0 0 1 434 WARNING: sym._IO_switch_to_main_get_area swords in memory
0x00444cf00 0 1 8 vcache opt sym._dl_get_dl_main_map
0x00470520 0 0 1 494 Initialize sym._IO_switch_to_main_wget_area
0x0048fae0 0 0 7 737 → 691 sym._nl_fnddomain_subfreereset HMAC authentication
0x0048fb30 0 16 247 → 237 sym._nl_unload_domain
[0x00400a30]>
```

Now we can see where in memory the program starts from. Since we found a function here we can examine it with the pdf (Print Disassembly Function) command.

```
pdf @main
```

```
elfmceager@tbfc-day-17: ~
File Actions Edit View Help
2022-07-15 0:-- main: OPTIONS IMPORT: peer-id set
/(fcn) sym.main 68 OPTIONS IMPORT: adjusting link_mtu to 1634
0x0000000000400000 sym.main ()
0x0000000000400000 ; var int local_ch @ rbp-0xc
0x0000000000400000 ; var int local_8h @ rbp-0x8
0x0000000000400000 ; var int local_4h @ rbp-0x4
0x0000000000400000 ; DATA XREF from 0x00400a4d (entry0)
0x0000000000400000 0x00400b4d coming 55 Channel: push rbp AES-256-CBC' initialized w
0x0000000000400000 0x00400b4e 4889e5 mov rbp, rsp
0x0000000000400000 0x00400b51 coming 4883ec10h sub rsp, 0x10 message hash 'SHA512
0x0000000000400000 For HMAC auth: 0x00400b55 on c745f4040000. mov dword [local_ch], 4
0x0000000000400000 0x00400b5c _route 0x00400b55 on c745f8050000. mov dword [local_8h], 5
0x0000000000400000 0x00400b63 _route 8b55f4 stl_gw mov edx, dword [local_ch] id
0x0000000000400000 0x00400b66 _ROUTE_GAB 8bf458 0.0.2.2 mov eax, dword [local_8h] HWADDR=08
0x0000000000400000 0x00400b69 0100 add eax, edx
0x0000000000400000 0x00400b6b 8945fc tun0 open mov dword [local_4h], eax
0x0000000000400000 0x00400b6e _iface 8bf4dc sets mtu mov ecx, dword [local_4h]
0x0000000000400000 0x00400b71 _iface 8b55f8 ret tun0 mov edx, dword [local_8h]
0x0000000000400000 0x00400b74 _addr 8bf45f4 10.10.10.10 mov eax, dword [local_ch]
0x0000000000400000 0x00400b77 _route 89c6 mov esi, eax 10.10.0.1 dev [NULL] t
0x0000000000400000 0x00400b79 48bd3d881409. lea rdi, qword str.the_value_of_a_
is_d_the_value_of_b_is_d_and_the_value_of_c_is_d ; 0x492008; "the value
of a is %d, the value of b is %d and the value of c is %d"
0x0000000000400000 0x00400b80 initial b800000000 hence mov eax, 0
0x0000000000400000 0x00400b85 client e8f6ea0000 it pcall sym.__printf HMAC authenti
cation Failed 0x00400b8a b800000000 mov eax, 0
0x0000000000400000 0x00400b8f c9 leave
```

We can set a breakpoint with db.

db 0x00400b55

After setting our break point we, we can again run pdf @main and now we will see a lowercase b right after our break point location

Next, we use the dc command to run the program until it hits the break point. If we run pdf, the memory address at our breakpoint will be highlighted.

```

[0x00400b55]> 

```

Next, to view the contents of, in our case, the local_ch variable, we will run px @memory-address

px @rbp-0cx

```

[0x00400b55]> px @ rbp-0xc
- offset - 0 1 2 3 4 5 6 7 8 9 A B C D E F 0123456789ABCDEF512
0x7ffd291902c4 0000 0000 1890 6b00 0000 0000 7018 4000 .....k.....p.@.
0x7ffd291902d4 0000 0000 1911 4000 0000 0000 0000 0000 .....@.....
0x7ffd291902e4 0000 0000 0000 0000 0100 0000 f803 1929 .....(....)
0x7ffd291902f4 fd7f 0000 4d0b 4000 0000 0000 0000 0000 ....M.@.....R=08
0x7ffd29190304 0000 0000 1700 0000 0100 0000 0000 0000 .....(.....
0x7ffd29190314 0000 0000 0000 0000 0200 0000 0000 0000 .....(.....
0x7ffd29190324 0000 0000 0000 0000 0000 0000 0000 0000 .....(.....
0x7ffd29190334 0000 0000 0000 0000 0000 0000 0004 4000 .....@.
0x7ffd29190344 0000 0000 ac61 4748 1987 d477 1019 4000 .....aGH...w..@.
0x7ffd29190354 0000 0000 0000 0000 0000 0000 1890 6b00 .....(.....
0x7ffd29190364 0000 0000 0000 0000 0000 0000 ac61 677d .....ag}
0x7ffd29190374 abd5 2e88 ac61 3359 1987 d477 0000 0000 .....a3Y...w...memory
0x7ffd29190384 0000 0000 0000 0000 0000 0000 0000 0000 .....(.....
0x7ffd29190394 0000 0000 0000 0000 0000 0000 0000 0000 .....(.....
0x7ffd291903a4 0000 0000 0000 0000 0000 0000 0000 0000 .....(.....
0x7ffd291903b4 0000 0000 0000 0000 0000 0000 0000 0000 .....(.....
[0x00400b55]> 

```

But we do not see our variable 4 so we will step forward using ds. Then we will run px @rbp-0cx again.

```

[0x00400b55]> px @ rbp-0xc
- offset - 0 1 2 3 4 5 6 7 8 9 A B C D E F 0123456789ABCDEF512
0x7ffd291902c4 0400 0000 1890 6b00 0000 0000 7018 4000 .....k.....p.@.
0x7ffd291902d4 0000 0000 1911 4000 0000 0000 0000 0000 .....@.....
0x7ffd291902e4 0000 0000 0000 0000 0100 0000 f803 1929 .....(....)
0x7ffd291902f4 fd7f 0000 4d0b 4000 0000 0000 0000 0000 ....M.@.....R=08
0x7ffd29190304 0000 0000 1700 0000 0100 0000 0000 0000 .....(.....
0x7ffd29190314 0000 0000 0000 0000 0200 0000 0000 0000 .....(.....
0x7ffd29190324 0000 0000 0000 0000 0000 0000 0000 0000 .....(.....
0x7ffd29190334 0000 0000 0000 0000 0000 0000 0004 4000 .....@.
0x7ffd29190344 0000 0000 ac61 4748 1987 d477 1019 4000 .....aGH...w..@.
0x7ffd29190354 0000 0000 0000 0000 0000 0000 1890 6b00 .....(.....
0x7ffd29190364 0000 0000 0000 0000 0000 0000 ac61 677d .....ag}
0x7ffd29190374 abd5 2e88 ac61 3359 1987 d477 0000 0000 .....a3Y...w...memory
0x7ffd29190384 0000 0000 0000 0000 0000 0000 0000 0000 .....(.....
0x7ffd29190394 0000 0000 0000 0000 0000 0000 0000 0000 .....(.....
0x7ffd291903a4 0000 0000 0000 0000 0000 0000 0000 0000 .....(.....
0x7ffd291903b4 0000 0000 0000 0000 0000 0000 0000 0000 .....(.....
[0x00400b55]> 

```

And we now can see the variable 4 returned. Going ahead to the challenge file, we follow the same initial steps

```

elfmceager@tbfc-day-17:~$ r2 -d ./challenge1 -l69/17 dev.tun0
Process with PID 1555 started ... dev.tun0: 10.10.0.9/16 via 10.10.0.1 dev [NULL]
= attach 1555 1555
bin.baddr 0x00400000WARNING: this configuration may cache passwords in memory
Using 0x400000!-nocheck option to prevent this
Warning: Cannot initialize dynamic strings Completed
asm.bits 64
asm.sjnl 0x00400000Authentic/Decrypt packet error: packet HMAC authentication failed
[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]> 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_fnddomain_subfreeres
0x0048fa40   16 247 → 237  sym._nl_unload_domain
[0x00400a30]>

```

pdf @ main

```

[0x00400a30]> pdf @main
;-- main: Using Data Channel: Cipher 'AES-256-CBC' initialized w
/ (fcn) sym.main 35
|_ sym.main ();
|_ for HMAC ; var int local_ch @ rbp-0xc
|_ 22-07-19 ; var int local_8h @ rbp-0x8 : Cipher 'AES-256-CBC' initialized w
|_ h 256 bit ; var int local_4h @ rbp-0x4
|_ 22-07-19 ; DATA XREF from 0x00400a4d (entry0): bit message hash 'SHA512'
|_ for HMAC 0x00400bad10 55 push rbp
|_ 22-07-19 0x00400b4et rou 4889e5 mov rbp, rsp
|_ 22-07-19 0x00400b51f_rwu c745f4010000. mov dword [local_ch], 1
|_ 22-07-19 0x00400b58UTE_c745f8060000. mov dword [local_8h], 6 HWADDR=00
|_ 22-07-19 0x00400b5f 8b45f4 mov eax, dword [local_ch]
|_ 22-07-19 0x00400b62NTAP 0faf45f8 imul eax, dword [local_8h]
|_ 22-07-19 0x00400b661_f7 8945fc mov dword [local_4h], eax
|_ 22-07-19 0x00400b69 1f8b00000000 mov eax, 0
|_ 22-07-19 0x00400b6e7 add 5d addl 10, rbp pop rbp dev.tun0
\ 22-07-19 0x00400b6f 99 c3 ret

```

Q1 : Match the data type with the size in bytes:

Byte = 1

Word = 2

Double word = 4

Quad = 8

Single precision = 4

Double precision = 8

Q2: What is the command to analyse the program in radare2?

Ans : aa

Q3: What is the command to set a breakpoint in radare2 ?

Ans : db

Q4: What is the command to execute the program until we hit a breakpoint?

Ans : dc

Question 5

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

Ans : 1

Find the address of local_ch which is 0x00400b51 and put a breakpoint at this location using command db. Now use dc to run the program. We can analyse the contents of local_ch with the command px @ rbp-0xc. Run command ds to step forward and command px @ rbp-0xc again. We see the value 1.

```
elfmceager@tbfc-day-17:~
```

```
File Actions Edit View Help
File Actions Edit View Help
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
\ 22-02-15 0x00400b6f IONS c3 ret
[0x00400a30]> db 0x00400b51 IMPORTS: adjusting link.mn to 1024
[0x00400a30]> dc 0x00400b51 IMPORTS: adjusting link.mn to 1024
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-512
0x7ffdbe3c4744 0000 0000 1890 6b00 0000 0000 4018 4000 .....k.....@.a.
0x7ffdbe3c4754 0000 0000 e910 4000 0000 0000 0000 0000 .....@.....d.w
0x7ffdbe3c4764 0000 0000 0000 0000 0100 0000 7848 3cbe .....xH<.
0x7ffdbe3c4774 fd7f 0000 40db 4000 0000 0000 0000 0000 ..M.a.....512
0x7ffdbe3c4784 0000 0000 1700 0000 0100 0000 0000 0000
0x7ffdbe3c4794 0000 0000 0000 0000 0200 0000 0000 0000
0x7ffdbe3c47a4 0000 0000 0000 0000 0000 0000 0000 0000
0x7ffdbe3c47b4 0000 0000 0000 0000 0000 0000 0004 4000 .....@.a..+08
0x7ffdbe3c47c4 0000 0000 a30f 2f0f 0766 1c6d e018 4000 .../.f.m..@.
0x7ffdbe3c47d4 0000 0000 0000 0000 0000 0000 1890 6b00 .....k.
0x7ffdbe3c47e4 0000 0000 0000 0000 0000 0000 a30f 6fb1 .....o.
0x7ffdbe3c47f4 ff1a e792 a30f 9ble 0766 1c6d 0000 0000 .....f.m...
0x7ffdbe3c4804 0000 0000 0000 0000 0000 0000 0000 0000
0x7ffdbe3c4814 0000 0000 0000 0000 0000 0000 0000 0000
0x7ffdbe3c4824 0000 0000 0000 0000 0000 0000 0000 0000
0x7ffdbe3c4834 0000 0000 0000 0000 0000 0000 0000 0000 .....memory
[0x00400b51]> ds
[0x00400b51]> px @ rbp-0xc
- offset - 0 1 2 3 4 5 6 7 8 9 A B C D E F 0123456789ABCDEF-ica
0x7ffdbe3c4744 0100 0000 1890 6b00 0000 0000 4018 4000 .....k.....@.a.
0x7ffdbe3c4754 0000 0000 e910 4000 0000 0000 0000 0000 .....@.....d.
0x7ffdbe3c4764 0000 0000 0000 0000 0100 0000 7848 3cbe .....xH<.
0x7ffdbe3c4774 fd7f 0000 40db 4000 0000 0000 0000 0000 ..M.a.
0x7ffdbe3c4784 0000 0000 1700 0000 0100 0000 0000 0000
0x7ffdbe3c4794 0000 0000 0000 0000 0200 0000 0000 0000
0x7ffdbe3c47a4 0000 0000 0000 0000 0000 0000 0000 0000
0x7ffdbe3c47b4 0000 0000 a30f 2f0f 0766 1c6d e018 4000 .....@.
0x7ffdbe3c47c4 0000 0000 0000 0000 0000 0000 1890 6b00 .....k.
0x7ffdbe3c47e4 0000 0000 0000 0000 0000 0000 a30f 6fb1 .....o.
0x7ffdbe3c47f4 ff1a e792 a30f 9ble 0766 1c6d 0000 0000 .....f.m...
0x7ffdbe3c4804 0000 0000 0000 0000 0000 0000 0000 0000
0x7ffdbe3c4814 0000 0000 0000 0000 0000 0000 0000 0000
0x7ffdbe3c4824 0000 0000 0000 0000 0000 0000 0000 0000
0x7ffdbe3c4834 0000 0000 0000 0000 0000 0000 0000 0000 .....memory
[0x00400b51]> 
```

Question 6

What is the value of eax when the imull instruction is called?

Ans : 6

Next, we want to find the value of eax at the first imul instruction. We can simply use ds to move to the next line and then use dr to see the value of eax. Since we didn't see any number, Run command ds to step forward and dr to see the value of eax. We see that the value is 6.

```
[0x00400b51]> ds:s4 OPTIONS IMPORT: adjusting link_mtu to 1634
[0x00400b51]> pdf @main
-- main: outgoing Data Channel: Cipher 'AES-256-CBC' initialized w
/ (fcn) sym.main 35
| 22-07-15 ;-- main: outgoing Data Channel: Using 512 bit message hash 'SHA512'
| for HMAC ; var int local_ch @ rbp-0xc
| 22-07-15 ; var int local_8h @ rbp-0x8: Cipher 'AES-256-CBC' initialized w
| 22-07-15 ; var int local_4h @ rbp-0x4
| 22-07-15 ; DATA XREF from 0x00400a4d (entry6): 512 bit message hash 'SHA512'
| for HMAC 0x00400b4d on 55 push rbp
| 22-07-15 0x00400b4e lla 4889e5 stx_wl mov rbp, rsp 0
| 22-07-15 0x00400b51 breq c745f4010000. mov dword [local_ch], 1 tho
| 22-07-15 0x00400b58 lte G c745f8060000. mov dword [local_8h], 6 HWADDR=08
| 22-07-15 0x00400b5f 8b45f4 mov eax, dword [local_ch]
| 22-07-15 ;-- rip:
| 22-07-15 0x00400b62 lla 0faf45f8 imul eax, dword [local_8h]
| 22-07-15 0x00400b66 lla 8945fc mov dword [local_4h], eax
| 22-07-15 0x00400b69 add b800000000. mov eax, 0
| 22-07-15 0x00400b6e lla 5d4...0001 pop rbp
| 22-07-15 0x00400b6f c3 ret
\file 0 metr 0x00400b6f c3 ret
```

```
elfmceager@tbfc-day-17: ~
File Actions Edit View Help
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

[0x00400b51]> dr:s4 OPTIONS IMPORT: peer-id set
rax = 0x00000001:s4 OPTIONS IMPORT: adjusting link_mtu to 1634
rbx = 0x00400400:s4 Using peer cipher 'AES-256-CBC'
rcx = 0x0044b9a0:s4 Outgoing Data Channel: Cipher 'AES-256-CBC' initialized w
rdx = 0x7fdbe3c4888
r8 = 0x01000000:s4 Outgoing Data Channel: Using 512 bit message hash 'SHA512'
r9 = 0x006bb8e0:authentication
r10 = 0x000000015:s4 Incoming Data Channel: Cipher 'AES-256-CBC' initialized w
r11 = 0x00000000
r12 = 0x004018e0:s4 Incoming Data Channel: Using 512 bit message hash 'SHA512'
r13 = 0x00000000:authentication
r14 = 0x006b9018:s4 net_route_v4_best_gw query: dst 0.0.0.0
r15 = 0x00000000:s4 net_route_v4_best_gw result: via 10.0.0.2 dev eth0
rsi = 0x7fdbe3c4878 OUTE GATEWAY 10.0.2.2/255.255.255.0 IFACE=eth0 HWADDR=08
rdi = 0x00000001
rsp = 0x7fdbe3c4750 UN/TAP device tun0 opened
rbp = 0x7fdbe3c4750 net_iface_mtu_set: mtu 1500 for tun0
rip = 0x00400b62:s4 net_iface_up: set tun0 up
rflags = 0x000000246 net_iface_v4_addr: 10.18.40.169/17 dev tun0
orax = 0xffffffffffffffffff nete_v4_addr: 10.10.0.0/16 via 10.18.0.1 dev [NULL] t
[0x00400b51]> ds
[0x00400b51]> dr:s4 WARNING: this configuration may cache passwords in memory
rax = 0x00000006:nocache option to prevent this
rbx = 0x00400400:s4 Initialization Sequence Completed
rcx = 0x0044b9a0:7 Authenticate/Decrypt packet error: packet HMAC authentication
rdx = 0x7fdbe3c4888
r8 = 0x01000000
r9 = 0x006bb8e0
r10 = 0x00000015
r11 = 0x00000000
r12 = 0x004018e0
r13 = 0x00000000
r14 = 0x006b9018
r15 = 0x00000000
rsi = 0x7fdbe3c4878
rdi = 0x00000001
rsp = 0x7fdbe3c4750
rbp = 0x7fdbe3c4750
rip = 0x00400b66
rflags = 0x000000246
orax = 0xffffffffffffffffff
[0x00400b51]>
```

Question 7

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

Ans : 6

Lastly, we want to find the value of local_4h before the eax is set to 0. Use the ds command to get to that point and then check the value of local_4h with command px @ rbp-0x4. The value here is 6.

```
[0x00400b51]> pdf @main
;-- main:
/ (fcn) sym.main 35
| sym.main();
| local_15_ ; var int local_ch @ rbp-0xc
| local_15_ ; var int local_8h @ rbp-0x8
| local_4h_ ; var int local_4h @ rbp-0x4
| for HMAC
|   DATA XREF from 0x0400a4d (entry0)
|   0x00400b4d    55      push rbp
|   0x00400b4e    4889e5   mov rbp, rsp
|   0x00400b51    b       c745f4010000. mov dword [local_ch], 10
|   0x00400b58    .c745f8060000. mov dword [local_8h], 6
|   0x00400b5f    80454f   mov eax, dword [local_ch]
|   0x00400b62    0faf45ff imul eax, dword [local_8h]
|   ;-- rip:
|   0x00400b66    8945fc   mov dword [local_4h], eax
|   0x00400b69    b800000000. mov eax, 0
|   0x00400b6e    5d       pop rbp
|   0x00400b6f    c3       ret
[0x00400b51]> ds
[0x00400b51]> px @ rbp-0x4
0x00400b51<4: 0123456789ABCDEF ica
- offset - 0 1 2 3 4 5 6 7 8 9 A B C D E F
0>ffdbe3c474c 0600 0000 4018 4000 0000 0000 e910 4000 ...@.@.....@.
0>ffdbe3c475c 0000 0000 0000 0000 0000 0000 0000 0000 .....
0>ffdbe3c476c 0100 0000 7848 3cbe fd7f 0000 4d0b 4000 ...XH...M.@.
0>ffdbe3c477c 0000 0000 0000 0000 0000 0000 1700 0000 .....
0>ffdbe3c478c 0100 0000 0000 0000 0000 0000 0000 0000 .....
0>ffdbe3c479c 0200 0000 0000 0000 0000 0000 0000 0000 .....
0>ffdbe3c47ac 0000 0000 0000 0000 0000 0000 0000 0000 .....
0>ffdbe3c47bc 0000 0000 0004 4000 0000 0000 a30f 2f0f .....@.../.
0>ffdbe3c47cc 0766 1cd6 e018 4000 0000 0000 0000 0000 f.m..@.
0>ffdbe3c47dc 0000 0000 1890 6b00 0000 0000 0000 0000 ...k
0>ffdbe3c47ec 0000 0000 a30f 6fb1 ff1a e792 a30f 9b1e ...o..
0>ffdbe3c47fc 0766 1cd6 0000 0000 0000 0000 0000 0000 f.m
0>ffdbe3c480c 0000 0000 0000 0000 0000 0000 0000 0000 .....
0>ffdbe3c481c 0000 0000 0000 0000 0000 0000 0000 0000 .....
0>ffdbe3c482c 0000 0000 0000 0000 0000 0000 0000 0000 .....
0>ffdbe3c483c 0000 0000 0000 0000 0000 0000 0000 0000 .....
[0x00400b51]>
```

Thought Process/Methodology:

Log into the instance with the given credentials. Opens the binary files in debugging mode using command r2 -d ./file1 . Once the analyzing completes we can seek out the entry point to the file. Now we can see where in memory the program starts from. Since we found a function here we can examine it with the pdf (Print Disassembly Function) command. We can set a breakpoint with db. After setting our break point we can again run pdf @main and now we will see a lowercase b right after our break point location. Next, we use the dc command to run the program until it hits the break point. If we run pdf, the memory address at our breakpoint will be highlighted. Next, to view the contents of, in our case, the local_ch variable, we will run px @memory-address. But we do not see our variable 4 so we will step forward using ds. Then we will run px @rbp-0cx again. And we now can see the variable 4 returned. Going ahead to the challenge file, we follow the same initial steps. Find the address of local_ch which is 0x00400b51 and put a breakpoint at this location using command db. Now use dc to run the program. We can analyse the contents of local_ch with the command px @ rbp-0xc. Run command ds to step forward and command px @ rbp-0xc again. We see the value 1. Next, we want to find the value of eax at the first imul instruction. We can simply use ds to move to the next line and then use dr to see the value of eax. Since we didn't see any number, Run command ds to step forward and dr to see the value of eax. We see that the value is 6. Lastly, we want to find the value of local_4h before the eax is set to 0. Use the ds command to get to that point and then check the value of local_4h with command px @ rbp-0x4. The value here is 6.

Day 18: The Bits Of Christmas

Reverse Engineering the .Net app

Now we'll open the **TBFC APP** in **ILSpy**. This will allow us to conduct some reverse engineering and look at some of the app's source code.

When we open it, we are prompted to enter Santa's password. After some investigation, We discovered something intriguing in **CrackMe -> MainForm -> buttonActivate Click**.

```
private unsafe void buttonActivate_Click(object sender, EventArgs e)
{
    IntPtr value = Marshal.StringToGlobalAnsi(textBoxKey.Text);
    sbyte* ptr = (sbyte*)System.Runtime.CompilerServices.Unsafe.AsPointer(ref <Module>._C_0BB@IKKDFEPG@santapassword121);
    void* ptr2 = (void*)value;
    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)
                {
                    break;
                }
                continue;
            }
            MessageBox.Show("Welcome, Santa, here's your flag thm{046af}", "That's the right key!", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
            return;
        }
        MessageBox.Show("Uh Oh! That's the wrong key", "You're not Santa!", MessageBoxButtons.OK, MessageBoxIcon.Hand);
    }
}
```

It appears that the programmer committed the error of hard-coding several important data! The password we're looking for is santapassword123.

```
private unsafe void buttonActivate_Click(object sender, EventArgs e)
{
    IntPtr value = Marshal.StringToGlobalAnsi(textBoxKey.Text);
    sbyte* ptr = (sbyte*)System.Runtime.CompilerServices.Unsafe.AsPointer(ref <Module>._C_0BB@IKKDFEPG@santapassword121);
    void* ptr2 = (void*)value;
    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)
                {
                    break;
                }
                continue;
            }
            MessageBox.Show("Welcome, Santa, here's your flag thm{046af}", "That's the right key!", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
            return;
        }
        MessageBox.Show("Uh Oh! That's the wrong key", "You're not Santa!", MessageBoxButtons.OK, MessageBoxIcon.Hand);
    }
}
```

Question 1: santapassword123

We also see the flag that will be visible when we properly log in!

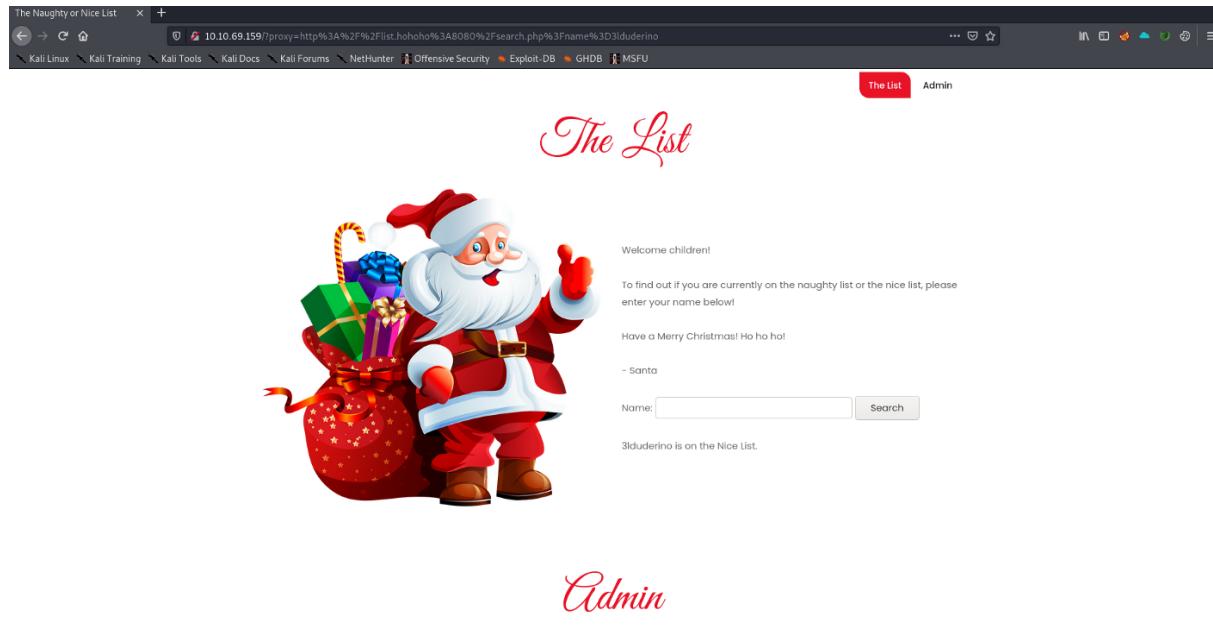
```
private unsafe void buttonActivate_Click(object sender, EventArgs e)
{
    IntPtr value = Marshal.StringToGlobalAnsi(textBoxKey.Text);
    sbyte* ptr = (sbyte*)System.Runtime.CompilerServices.Unsafe.AsPointer<ref <Module>.??_C@_0BB@IKKDFEPG@santa@password321@);
    void* ptr2 = (void*)value;
    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)
                {
                    break;
                }
                continue;
            }
            MessageBox.Show("Welcome, Santa, here's your flag thm{046af}", "That's the right key!", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
            return;
        }
        MessageBox.Show("Uh Oh! That's the wrong key", "You're not Santa!", MessageBoxButtons.OK, MessageBoxIcon.Hand);
    }
}
```

Question 2: thm{046af}

Day 19: The Naught Or Nice List!

Question 1: What is Santa's password?

When we first visit the website, we see this form, along with an admin login underneath it.



The URL is intriguing:

http://10.10.69.159/?proxy=http%3A%2F%2Flist.hohoho%3A8080%2Fsearch.php%3Fname%3D31 Duderino Also, we're not on the nice list, which is unfortunate. We conducted a short examination of the source code and found nothing of interest.

Burp did a short URL decoding and returned the following:

http://10.10.69.159/?proxy=http://list.hohoho:8080/search.php?name=31duderino

We search strip everything past the port number to determine if there is a root site and create a custom script to check for extra folders. In a failure site, we know what we're searching for (in this case, the root site).

Name:

Search

Not Found

The requested URL was not found on this server.

But for the time being, let's go on and see if there's another option.

So, it appears that we have a local server with port 8080 open; we may experiment with the port number to see what we can obtain. We can use this to perform basic port scanning.

Name:

Search

Failed to connect to list.hohoho port 21: Connection refused

However, it appears that we immediately receive a hit on port 22, which is open on the distant server.

Name:

Search

Recv failure: Connection reset by peer

We attempted to obtain the `passwd` file, but it appears that they are analysing the URL.

Name:

file://etc/passwd is on the Nice List.

When we try to determine whether there is a local only server operating, we are blocked (by their security team).

<http://10.10.69.159/?proxy=http%3A%2F%2Flocalhost>

Name:

Your search has been blocked by our security team.

Let's return to the port issue and see if we can find any local only services running on the server.

<http://10.10.69.159/?proxy=http%3A%2F%2Flist.hohoho.localtest.me:22>

With the connection reset by peer, we received the same problem as before. So far, so good. We may check to see if there are any ports that are exclusively accessible to the localhost. We now have a hit with port 80 containing some goodies, which addresses Question 1.

<http://10.10.69.159/?proxy=http%3A%2F%2Flist.hohoho.localtest.me:80>

Name:

Search

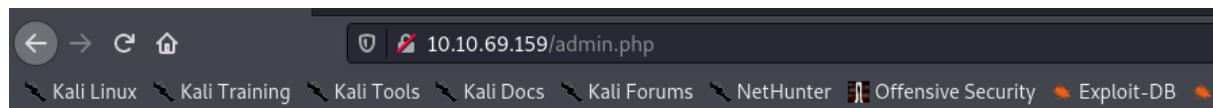
Santa,

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

Please keep in mind that the username is case sensitive.



List Administration

This page is currently under construction.

Only press this button when emergency levels of Christmas cheer are needed!

DELETE NAUGHTY LIST

We will obtain the flag for today's challenge after deleting the naughty list (or reading the source code).