

# SCANNING: **WIRESHARK & NMAP**

PRESENTED BY YASH PATEL



[www.linkedin.com/in/yash-c-patel](https://www.linkedin.com/in/yash-c-patel)





# Warning!

The content provided hereafter is intended solely for educational purposes.

# Agenda

**1** Introduction

**2** N & NSE

**3** Wireshark

**4** Project requirement & Brief

**5** Wireshark IDS/Firewall evade techniques

**6** NSE scanning

**7** Conclusion

**8** Reference



# Introduction

- This project is focused on mastering the techniques of scanning networks and websites through various tools and methods.
- It involves conducting network scans with Wireshark and Nmap, analyzing the outcomes, and pinpointing potential vulnerabilities or problems.



# Nmap & NSE



- **Nmap (Network Mapper)** is an open-source network discovery and security auditing tool. Its main features include: Scan ports, identify services, and detect operating systems.
- **NSE (Nmap Scripting Engine):** Nmap's built-in scripting engine that enables users to create and share advanced scanning programs

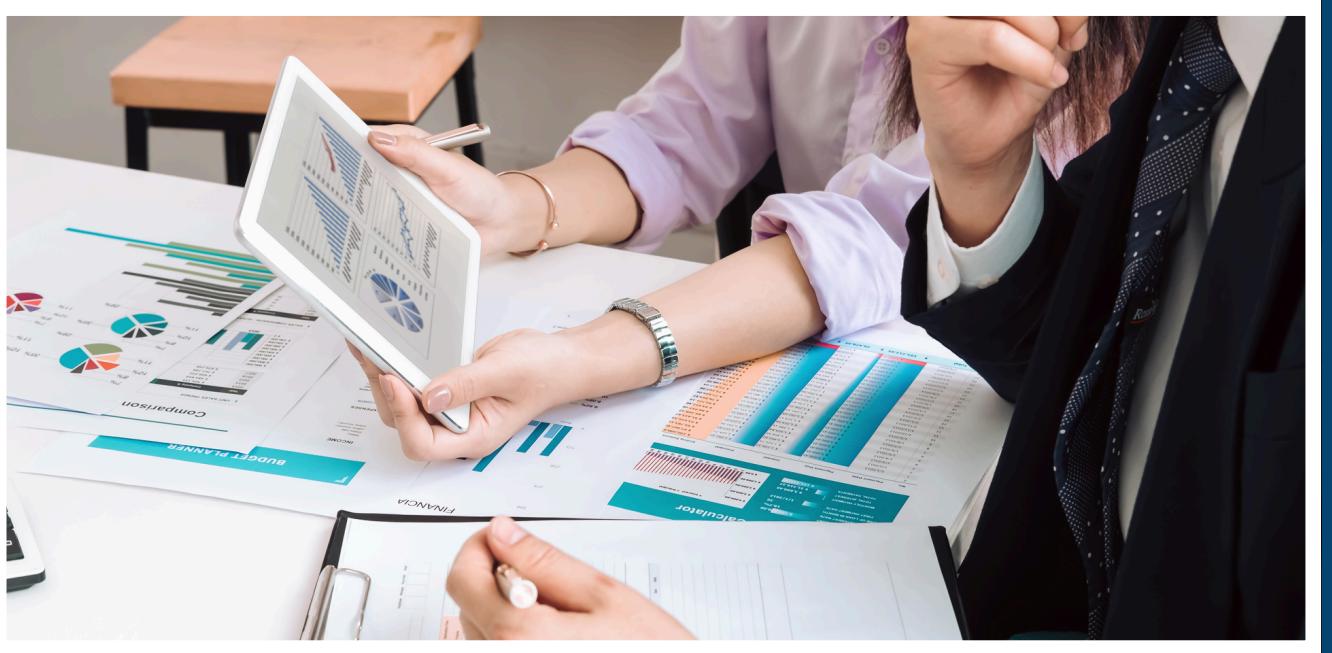
# Wireshark



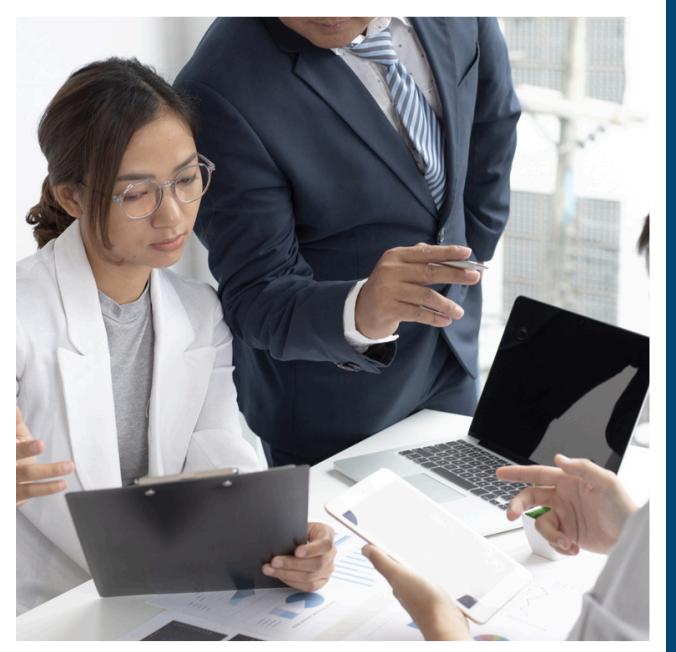
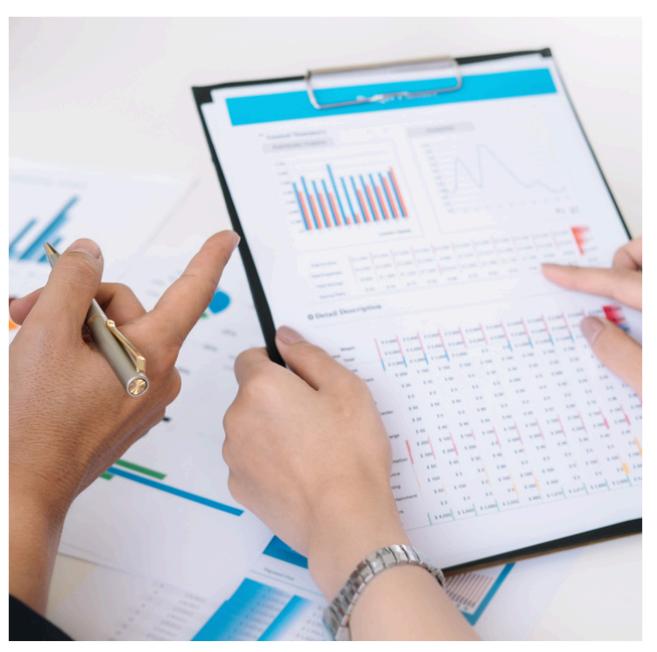
- Wireshark is a network protocol analyser that collects Monitor live traffic, diagnose network faults, and identify security flaws.
- Wireshark can record packets in real time, revealing precise information about their contents and allowing analysts to identify potential issues like as intrusions, strange traffic, or incorrectly configured networks.

# Project requirement & Brief

- Targets /Tools Used
  - Tools: Kali Linux, Nmap, Wireshark
  - Testing Websites: [www.tryhackme.org](http://www.tryhackme.org)
- Testing Scope
  - Project Outcome: Wireshark techniques for firewall/IDS evasion.
  - Nmap NSE scripts for vulnerability scanning.



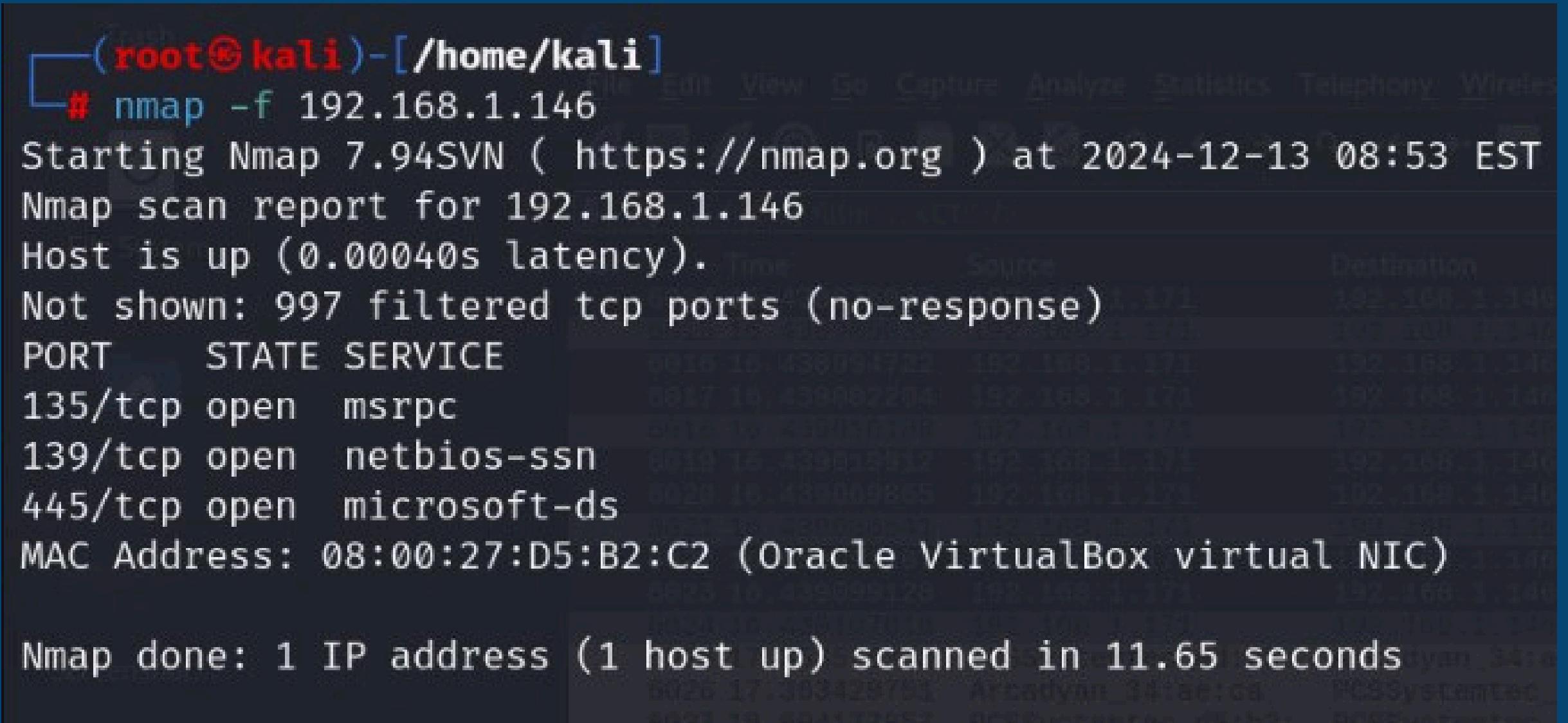
# Task 1: Firewall/ IDS Evasion



- IDS evasion refers to bypassing Intrusion Detection Systems (IDS) to avoid detection during network scanning or penetration testing.
- In this task, I will perform three key evasion techniques using Wireshark to analyze the network traffic:
  - 1. Packet Fragmentation
  - 2. Source Port Manipulation
  - 3. IP Address Decoy
- Wireshark will be used to capture and inspect traffic during these techniques, ensuring effectiveness in evading detection.

# 1. PACKET FRAGMENTATION

- Fragmentation is a necessary function of the TCP/IP suite, as different routers on the Internet (or internally) have different sized network links. Fragmentation dissects the packets into smaller pieces so that they can fit the smaller links as they travel the network.
- we see packets sent from a device with the MAC address SagemcomBroa\_53:4e. These are ARP (Address Resolution Protocol) packets that respond to queries for an IP address. Packet fragmentation is used to split packets into smaller parts, which can help evade detection by intrusion detection systems (IDS). In this case, the packets are 60 bytes in size, indicating they are relatively small, which could be the result of fragmentation.



(root㉿kali)-[~/home/kali]

```
# nmap -f 192.168.1.146
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-13 08:53 EST
Nmap scan report for 192.168.1.146
Host is up (0.00040s latency).

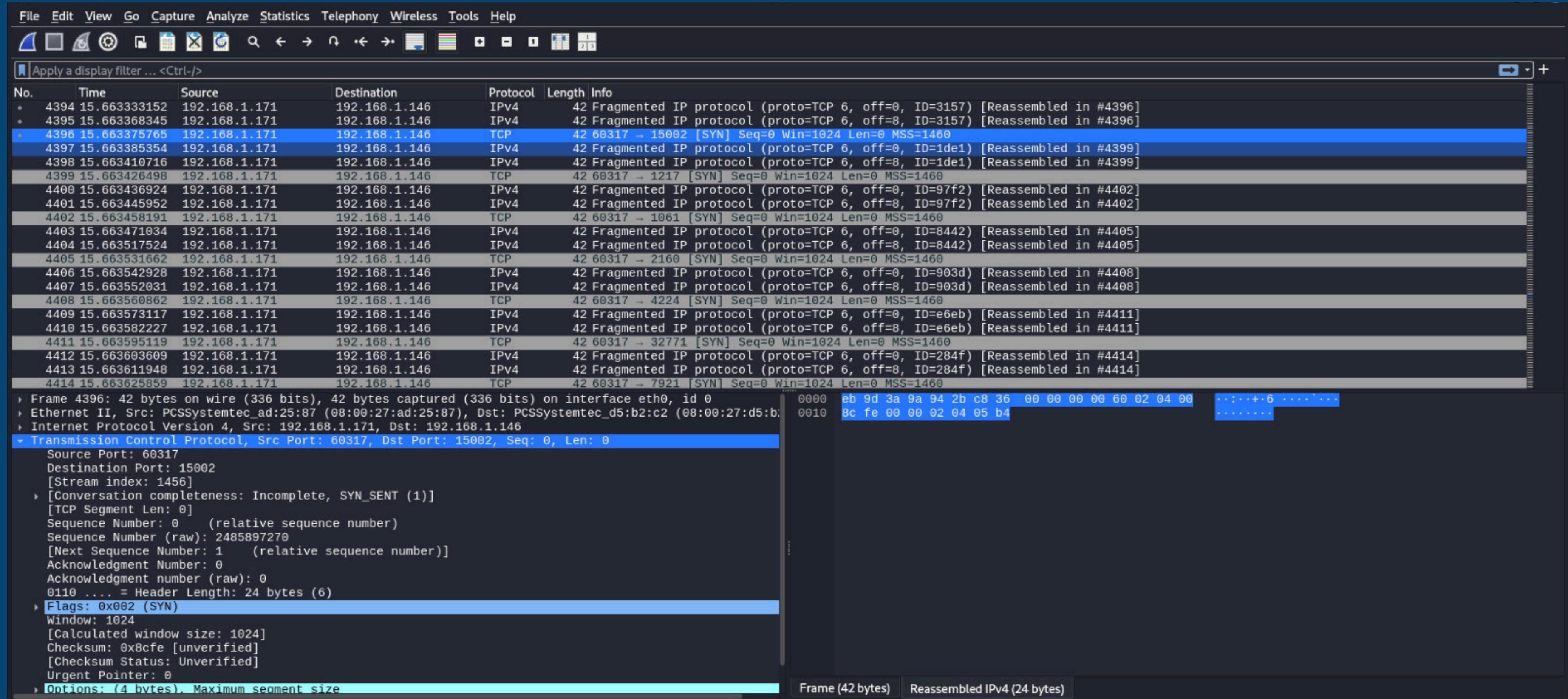
Not shown: 997 filtered tcp ports (no-response)

PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds

MAC Address: 08:00:27:D5:B2:C2 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 11.65 seconds
```

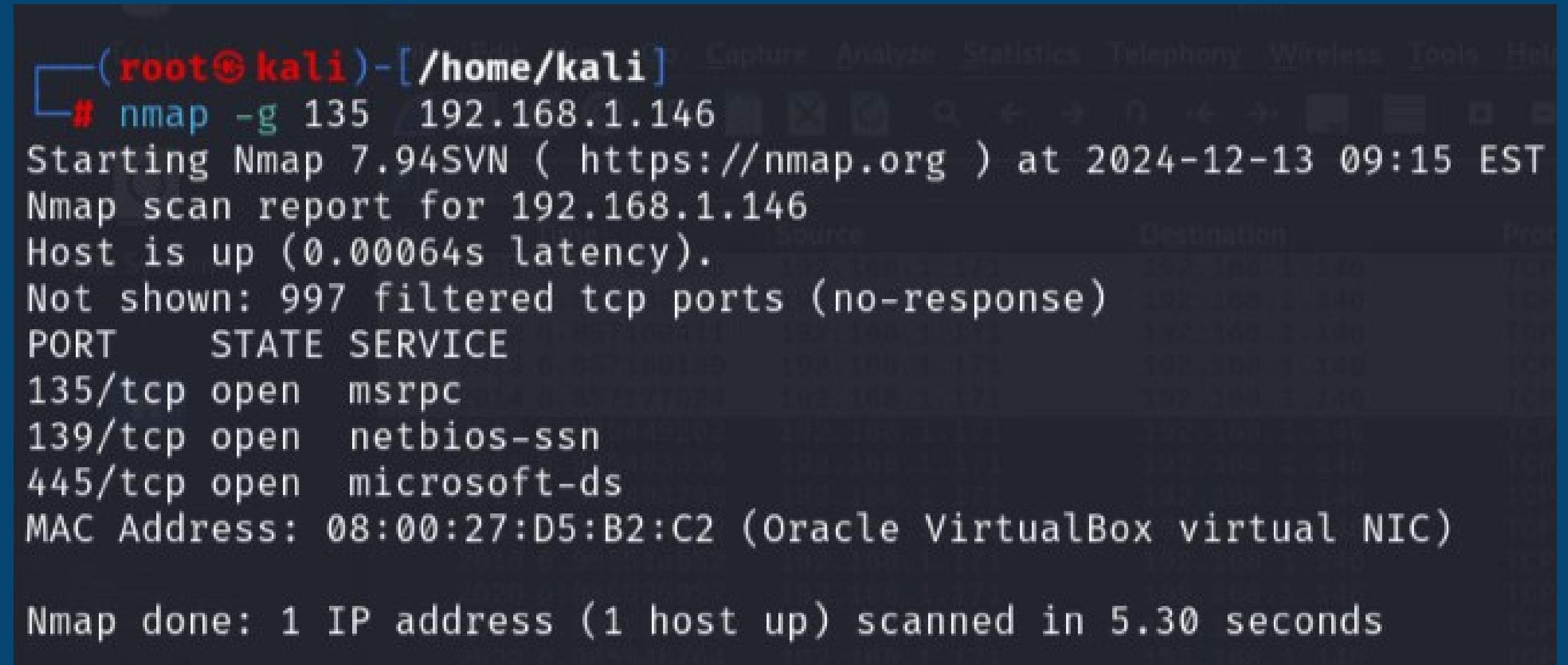
here I entered Nmap command 'nmap -f' and target IP.



The figure shows frame 4396 sent to 192.168.1.146. The highlighted part shows that this frame was reassembled in IP frame 4396. and fragmentation was done successfully

## 2. Source Port Manipulation

- A method used to evade detection and enhance security by altering the default port settings of services or applications.
- By changing the listening ports, attackers or defenders can obscure the presence of services, making it more difficult for unauthorized users to identify and exploit them. This technique is often employed to bypass firewalls and intrusion detection systems.



```
(root㉿kali)-[~/home/kali] # nmap -g 135 192.168.1.146
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-13 09:15 EST
Nmap scan report for 192.168.1.146
Host is up (0.00064s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
MAC Address: 08:00:27:D5:B2:C2 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 5.30 seconds
```

**here I entered Nmap command ‘nmap -g 135’ and target IP.**

\*eth0

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp.port == 135

No.	Time	Source	Destination	Protocol	Length	Info
1537	6.209007489	192.168.1.171	192.168.1.146	TCP	58	[TCP Port numbers reused] 135 → 1717 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1538	6.209067461	192.168.1.171	192.168.1.146	TCP	58	[TCP Port numbers reused] 135 → 50003 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1539	6.209085919	192.168.1.171	192.168.1.146	TCP	58	[TCP Port numbers reused] 135 → 12265 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1540	6.209100415	192.168.1.171	192.168.1.146	TCP	58	[TCP Port numbers reused] 135 → 563 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1541	6.209142747	192.168.1.171	192.168.1.146	TCP	58	[TCP Port numbers reused] 135 → 617 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1542	6.209166399	192.168.1.171	192.168.1.146	TCP	58	135 → 3269 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1543	6.209177862	192.168.1.171	192.168.1.146	TCP	58	135 → 500 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1544	6.209187450	192.168.1.171	192.168.1.146	TCP	58	135 → 1035 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1545	6.209197973	192.168.1.171	192.168.1.146	TCP	58	135 → 1417 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1546	6.209265467	192.168.1.171	192.168.1.146	TCP	58	135 → 3211 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1547	6.209279375	192.168.1.171	192.168.1.146	TCP	58	135 → 7778 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1548	6.209289218	192.168.1.171	192.168.1.146	TCP	58	135 → 1033 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1549	6.209298542	192.168.1.171	192.168.1.146	TCP	58	135 → 1092 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1550	6.209307959	192.168.1.171	192.168.1.146	TCP	58	135 → 9917 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1551	6.209317592	192.168.1.171	192.168.1.146	TCP	58	135 → 5960 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1552	6.209328018	192.168.1.171	192.168.1.146	TCP	58	135 → 9502 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1553	6.209370520	192.168.1.171	192.168.1.146	TCP	58	135 → 1244 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1554	6.209382181	192.168.1.171	192.168.1.146	TCP	58	135 → 4045 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1555	6.209391772	192.168.1.171	192.168.1.146	TCP	58	135 → 58080 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1556	6.209401170	192.168.1.171	192.168.1.146	TCP	58	135 → 1026 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1557	6.209410572	192.168.1.171	192.168.1.146	TCP	58	135 → 1166 [SYN] Seq=0 Win=1024 Len=0 MSS=1460

Frame 1543: 58 bytes on wire (464 bits), 58 bytes captured (464 bits) on interface eth0, id 0

Ethernet II, Src: PCSSystemtec\_ad:25:87 (08:00:27:ad:25:87), Dst: PCSSystemtec\_d5:b2:c2 (08:00:27:d5:b2:c2)

Internet Protocol Version 4, Src: 192.168.1.171, Dst: 192.168.1.146

Transmission Control Protocol, Src Port: 135, Dst Port: 500, Seq: 0, Len: 0

Source Port: 135  
 Destination Port: 500  
 [Stream index: 1516]  
 [Conversation completeness: Incomplete, SYN\_SENT (1)]  
 [TCP Segment Len: 0]  
 Sequence Number: 0 (relative sequence number)  
 Sequence Number (raw): 4202280902  
 [Next Sequence Number: 1 (relative sequence number)]  
 Acknowledgment Number: 0  
 Acknowledgment number (raw): 0  
 0110 .... = Header Length: 24 bytes (6)  
 Flags: 0x002 (SYN)  
 Window: 1024  
 [Calculated window size: 1024]  
 Checksum: 0x5add [unverified]  
 [Checksum Status: Unverified]  
 Urgent Pointer: 0  
 Options: (4 bytes), Maximum segment size

Transmission Control Protocol (tcp), 24 bytes

Packets: 2026 · Displayed: 2009 (99.2%) · Dropped: 0 (0.0%)

Profile: Default

The source port of packets can be specified using the -source-port or -g flag. The packets transmitted as a result of running the aforementioned command are seen in the image. It is evident that all SYN packets have port 500 as their source.

## 3. IP Address Decoy

- A technique used to conceal the true source of network traffic by sending packets from multiple decoy IP addresses.
- This approach confuses the target about the actual origin of the attack or request, making it difficult for security systems to track and mitigate malicious activities. It is commonly used in both offensive and defensive cybersecurity strategies to enhance anonymity.

```
(root㉿kali)-[~/home/kali]
# nmap -D RND:5 104.18.20.83
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-13 09:34 EST
Nmap scan report for 104.18.20.83
Host is up (0.023s latency).

Not shown: 996 filtered tcp ports (no-response)
PORT      STATE SERVICE
80/tcp    open  http
443/tcp   open  https
8080/tcp  open  http-proxy
8443/tcp  open  https-alt

Nmap done: 1 IP address (1 host up) scanned in 4.99 seconds
```

When performing this type of scan, there is two option.

nmap-D decoy1, decoy2, decoy3 etc.

This option allows you to manually specify the IP addresses of the decoys  
nmap-D RND:5 [Target IP or Target website] This option generates a random number of decoys.

\*eth0

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
11809	6.420124799	221.36.69.1	104.18.20.83	TCP	58	43047 → 4129 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11810	6.420129706	145.166.41.224	104.18.20.83	TCP	58	43047 → 4129 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11811	6.420134479	75.135.216.60	104.18.20.83	TCP	58	43047 → 4129 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11812	6.424298662	192.168.1.171	104.18.20.83	TCP	58	43047 → 6692 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11813	6.424355472	199.243.221.146	104.18.20.83	TCP	58	43047 → 6692 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11814	6.424379013	156.21.222.28	104.18.20.83	TCP	58	43047 → 6692 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11815	6.424386676	221.36.69.1	104.18.20.83	TCP	58	43047 → 6692 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11816	6.424393499	145.166.41.224	104.18.20.83	TCP	58	43047 → 6692 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11817	6.424400598	75.135.216.60	104.18.20.83	TCP	58	43047 → 6692 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11818	6.424409708	192.168.1.171	104.18.20.83	TCP	58	43047 → 3971 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11819	6.424417701	199.243.221.146	104.18.20.83	TCP	58	43047 → 3971 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11820	6.424424730	156.21.222.28	104.18.20.83	TCP	58	43047 → 3971 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11821	6.424431791	221.36.69.1	104.18.20.83	TCP	58	43047 → 3971 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11822	6.424438662	145.166.41.224	104.18.20.83	TCP	58	43047 → 3971 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11823	6.424447510	75.135.216.60	104.18.20.83	TCP	58	43047 → 3971 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11824	6.426673448	192.168.1.171	104.18.20.83	TCP	58	43047 → 9001 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11825	6.426793046	199.243.221.146	104.18.20.83	TCP	58	43047 → 9001 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11826	6.426823905	156.21.222.28	104.18.20.83	TCP	58	43047 → 9001 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11827	6.426833592	221.36.69.1	104.18.20.83	TCP	58	43047 → 9001 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11828	6.426845317	145.166.41.224	104.18.20.83	TCP	58	43047 → 9001 [SYN] Seq=0 Win=1024 Len=0 MSS=1460

```

> Frame 11814: 58 bytes on wire (464 bits), 58 bytes captured (464 bits) on interface eth0, id 0
> Ethernet II, Src: PCSSystemtec_ad:25:87 (08:00:27:ad:25:87), Dst: Arcadyan_34:ae:ca (74:90:bc:34:ae:ca)
> Internet Protocol Version 4, Src: 156.21.222.28, Dst: 104.18.20.83
-> Transmission Control Protocol, Src Port: 43047, Dst Port: 6692, Seq: 0, Len: 0
    Source Port: 43047
    Destination Port: 6692
    [Stream index: 11780]
    [Conversation completeness: Incomplete, SYN_SENT (1)]
    [TCP Segment Len: 0]
    Sequence Number: 0      (relative sequence number)
    Sequence Number (raw): 3856697708
    [Next Sequence Number: 1      (relative sequence number)]
    Acknowledgment Number: 0
    Acknowledgment number (raw): 0
    0110 .... = Header Length: 24 bytes (6)
    Flags: 0x002 (SYN)
    Window: 1024
    [Calculated window size: 1024]
    Checksum: 0x6bf6 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
    Options: (4 bytes), Maximum segment size

```

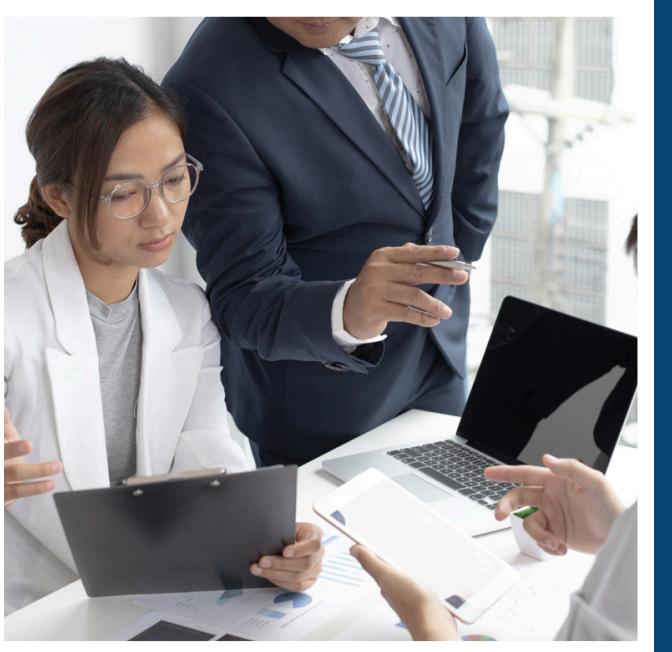
Transmission Control Protocol (tcp), 24 bytes

Packets: 12035 · Dropped: 0 (0.0%)

Profile: Default

traffic from multiple IP addresses is displayed. The device 192.168.1.171 is sending queries to various IP addresses (e.g., 221.36.69.1). This type of traffic may result from using the -D option in Nmap, where scanning is performed using multiple IP addresses (both real and decoy), making it harder to identify the true source of the scan

# Task 2: NSE Scanning



- In this task, I will perform Network Scanning using Nmap Scripting Engine (NSE). NSE allows for advanced network scanning by running scripts to automate various tasks, such as vulnerability detection, authentication bypass, and more. I will use the following 3 NSE scripts for testing:

- 1. http-headers.nse
- 2. http-auth.nse
- 3. daytime.nse

**Target Websites/System:**

**[www.tryhackme.org](http://www.tryhackme.org)**  
**192.168.1.146**

- By running these scripts, I aim to gather information about the target websites, including server headers, authentication mechanisms, WAF presence, and any potential internal IP disclosure.

# 1. http-headers.nse

## Description:

This Nmap NSE script performs HTTP header analysis by sending a HEAD request to the root directory ("/") of a web server. The script retrieves important server details such as server information, content type, and connection status. If the HEAD request fails, the script switches to a GET request automatically.

- Users can customize the path (e.g., /index.php) or force a GET request through script arguments.
- It is categorized as safe for discovery tasks during network scanning.

```
1 local http = require "http"
2 local shortport = require "shortport"
3 local stdnse = require "stdnse"
4 local table = require "table"
5
6 description = [[
7 Performs a HEAD request for the root folder ("/") of a web server and displays the HTTP headers returned.
8 ]]
9
10 --
11 -- @output
12 -- PORT STATE SERVICE
13 -- 80/tcp open http
14 -- | http-headers:
15 -- |   Date: Fri, 25 Jan 2013 17:39:08 GMT
16 -- |   Server: Apache/2.2.14 (Ubuntu)
17 -- |   Accept-Ranges: bytes
18 -- |   Vary: Accept-Encoding
19 -- |   Connection: close
20 -- |   Content-Type: text/html
21 -- |
22 -- |_ (Request type: HEAD)
23 --
24 --@args path The path to request, such as <code>/index.php</code>. Default <code>/</code>.
25 --@args useget Set to force GET requests instead of HEAD.
26 --
27 --@see http-security-headers.nse
28
```

## Outputs:

- Displays the server's HTTP headers.
- Indicates whether a HEAD or GET request was used.

# 1. http-headers.nse

```
(root㉿kali)-[~/home/kali]
└─# nmap --script http-headers.nse www.tryhackme.org
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-13 10:09 EST
Nmap scan report for www.tryhackme.org (70.32.1.32)
Host is up (0.093s latency).
rDNS record for 70.32.1.32: ip-70.32.1.32.hosted.by.gigenet.com
Not shown: 998 filtered tcp ports (no-response)
PORT      STATE SERVICE
53/tcp    closed domain
80/tcp    open   http
| http-headers:
|   date: Fri, 13 Dec 2024 15:09:23 GMT
|   server: Apache
|   set-cookie: __tad=1734102563.4913642; expires=Mon, 11-Dec-2034 15:09:23 GMT; Max-Age=315360000
|   vary: Accept-Encoding
|   content-length: 1049
|   content-type: text/html; charset=UTF-8
|   connection: close
|
|_ (Request type: GET)

Nmap done: 1 IP address (1 host up) scanned in 20.54 seconds
```

NSE Script executed of testing website ‘www.tryhackme.org’

## 2. http-auth.nse

### Description:

This Nmap NSE script is used to retrieve authentication schemes and realms from web services requiring authentication. When a web service responds with a 401 Unauthorized status, the script analyzes the WWW-Authenticate header to extract details such as:

- Authentication methods (e.g., Basic, NTLM, Digest)
- Relevant parameters like realm, nonce, and algorithm
- Users can customize the target path (e.g., /login) using script arguments.

```
1 local http = require "http"
2 local shortport = require "shortport"
3 local stdnse = require "stdnse"
4 local string = require "string"
5 local table = require "table"
6
7 description = [[
8 Retrieves the authentication scheme and realm of a web service that requires
9 authentication.
10 ]]
11
12 --
13 -- @usage
14 -- nmap --script http-auth [--script-args http-auth.path=/login] -p80 <host>
15 --
16 -- @output
17 -- PORT STATE SERVICE REASON
18 -- 80/tcp open http syn-ack
19 -- | http-auth:
20 -- | HTTP/1.1 401 Unauthorized
21 -- | Negotiate
22 -- | NTLM
23 -- | Digest charset=utf-8 nonce=+Upgraded+v1e4e256b4afb7f89be014e ... 968ccd60affb7c qop=auth algorithm=MD5-sess realm=example.com
24 -- |_ Basic realm=example.com
25 --
```

### Outputs:

- Lists the authentication schemes used by the web service.
- Displays associated parameters, such as the realm and algorithm.

## 2. http-auth.nse

```
(root㉿kali)-[~/home/kali]
# nmap --script http-auth.nse www.tryhackme.org
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-13 10:06 EST
Nmap scan report for www.tryhackme.org (70.32.1.32)
Host is up (0.094s latency).
rDNS record for 70.32.1.32: ip-70.32.1.32.hosted.by.gigenet.com
Not shown: 997 filtered tcp ports (no-response)
PORT      STATE SERVICE
53/tcp    closed domain
80/tcp    open   http
443/tcp   open   https

Nmap done: 1 IP address (1 host up) scanned in 27.45 seconds
```

NSE Script executed of testing website [www.tryhackme.org](http://www.tryhackme.org)

# 3. daytime.nse

## Descriptions:

This Nmap NSE script use to Retrieves the day and time from the Daytime service and it is categorized in discovery and safe route.

## Outputs:

- Here's a sample output from the daytime.nse script:

```
PORT STATE SERVICE 13/tcp open
daytime |_daytime:
Wed Mar 31 14:48:58 MDT 2010
```

```
1 local comm = require "comm"
2 local shortport = require "shortport"
3 local oops = require "oops"
4
5 description = [
6 Retrieves the day and time from the Daytime service.
7 ]
8
9 --
10 -- @output
11 -- PORT STATE SERVICE
12 -- 13/tcp open daytime
13 -- |_daytime: Wed Mar 31 14:48:58 MDT 2010
14
15 author = "Diman Todorov"
16
17 license = "Same as Nmap--See https://nmap.org/book/man-legal.html"
18
19 categories = {"discovery", "safe"}
20
21
22 portrule = shortport.port_or_service(13, "daytime", {"tcp", "udp"})
23
24 action = function(host, port)
25   return oops.output(comm.exchange(host, port, "dummy", {lines=1}))
26 end
```

## 3. daytime.nse

```
[root@kali]~[~/home/kali]
# nmap --script daytime.nse 192.168.1.146
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-13 10:13 EST
Nmap scan report for 192.168.1.146
Host is up (0.0010s latency).

Not shown: 997 filtered tcp ports (no-response)
PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
MAC Address: 08:00:27:D5:B2:C2 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 5.07 seconds
```

**NSE Script executed of testing System ip 192.168.1.146**

# Conclusion



- Nmap and Wireshark are both essential tools for network security, each serving distinct roles. Nmap is used for network scanning, identifying active hosts, open ports, and potential vulnerabilities, offering a broad overview of the network's security.
- Wireshark excels in real-time traffic capture and analysis, providing detailed insights into data exchanges and detecting anomalies. When used together, Nmap uncovers vulnerabilities, and Wireshark helps analyze traffic to identify suspicious activities or exploits. Together, these tools offer a comprehensive approach to network reconnaissance, vulnerability scanning, and cybersecurity assessments.

# Reference

- Nmap Guide:  
<https://nmap.org/book/man.html>
- Wireshark User Guide:  
[https://www.wireshark.org/docs/wsug\\_htm\\_l\\_chunked](https://www.wireshark.org/docs/wsug_htm_l_chunked)
- <https://www.infosecmatter.com/nmap-nse-library>
- Testing Website:  
<http://www.tryhackme.org>



# THANK YOU

By Yash Patel