Lab 1: Simulating Network Routing and VLAN Configuration in Linux

Objective:

In this lab, students will simulate network routing and VLAN configuration using Linux based tools. They will use the OWASP Broken Web Application VM's IP address to test connectivity and routing principles.

Learning Outcomes:

By the end of this lab, students will:

- Understand how to set up routing and VLANs on Linux.
- Be able to simulate network device behavior using Linux tools.
- Gain hands-on experience in network troubleshooting.
- Learn how to configure iptables, ip route, and network namespaces to manage traffic.

Materials Needed:

- Linux machine or VM (e.g., Ubuntu, Kali).
- OWASP Broken Web Application VM (reachable on the network).
- IP address of the OWASP Broken Web Application VM (e.g., 192.168.X.X).
- Terminal access.

Exercise 1: Setting Up Static Routing in Linux

1. Task: Set up static routes in Linux to simulate routing behavior. Ensure the Linux machine can route traffic to the OWASP Broken Web Application VM.

Steps:

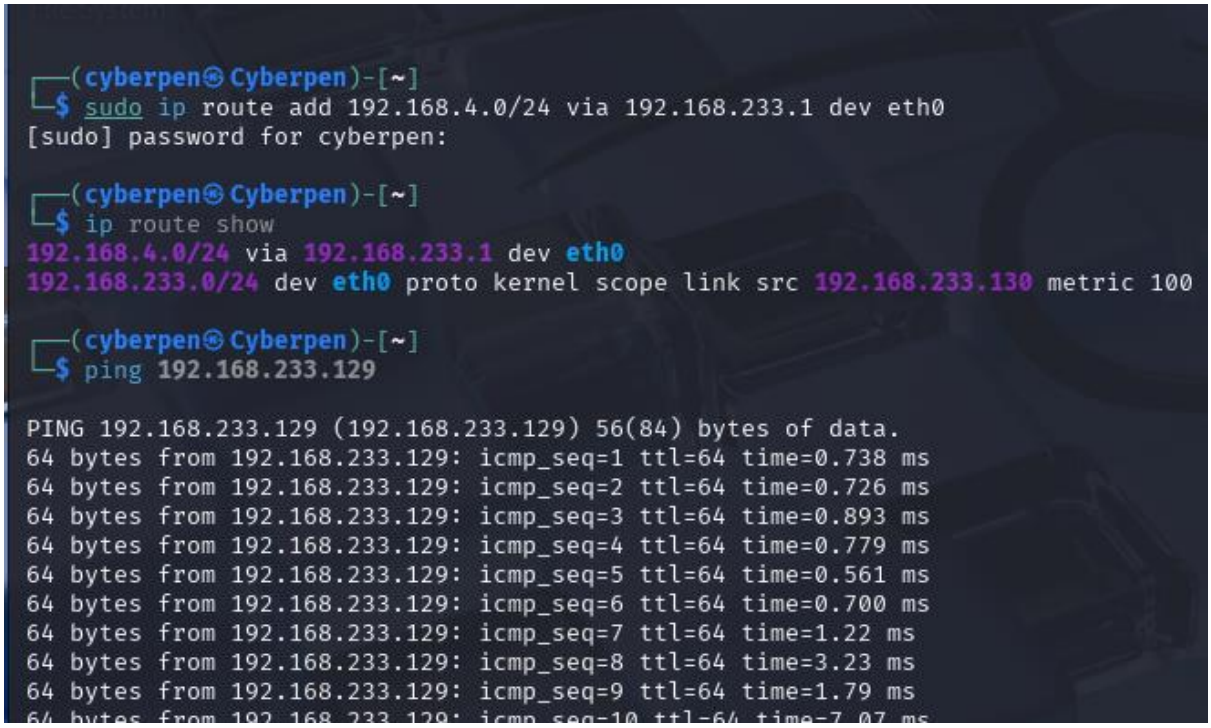
a. Use ip route to add a static route to reach the OWASP Broken Web Application VM.

b. Ensure that your machine routes traffic correctly through the simulated network.

Commands: sudo ip route add 192.168.4.0/24 via 192.168.233.1 dev eth0

ip route show: Verify your routing table.

Verification Command: ping 192.168.233.129



2. Question:

- **How does static routing work on a Linux system?**

Linux uses a routing table to decide where to send packets.Static routes are manually added entries that tell Linux how to reach specific networks via particular gateways.

When a packet's destination matches a static route, Linux forwards it through the specified gateway/interface.

- **What challenges could arise when setting up routes manually?**

Routes are not persistent after reboot unless saved. Misconfiguring gateway IPs or network masks can cause unreachable hosts. Overlapping routes or conflicts with dynamic routing protocols.

Lack of default gateway may block internet or unknown network traffic.

Troubleshooting can be tricky without proper network knowledge.

Exercise 2: VLAN Configuration Using Network Namespaces

3. Task: Create virtual LANs (VLANs) using Linux network namespaces to segment traffic. Assign the OWASP Broken Web Application VM to a separate namespace and test connectivity between namespaces.

Steps:

a. Create two network namespaces to simulate VLANs.

b. Assign virtual network interfaces to each namespace.

c. Route traffic between the namespaces.

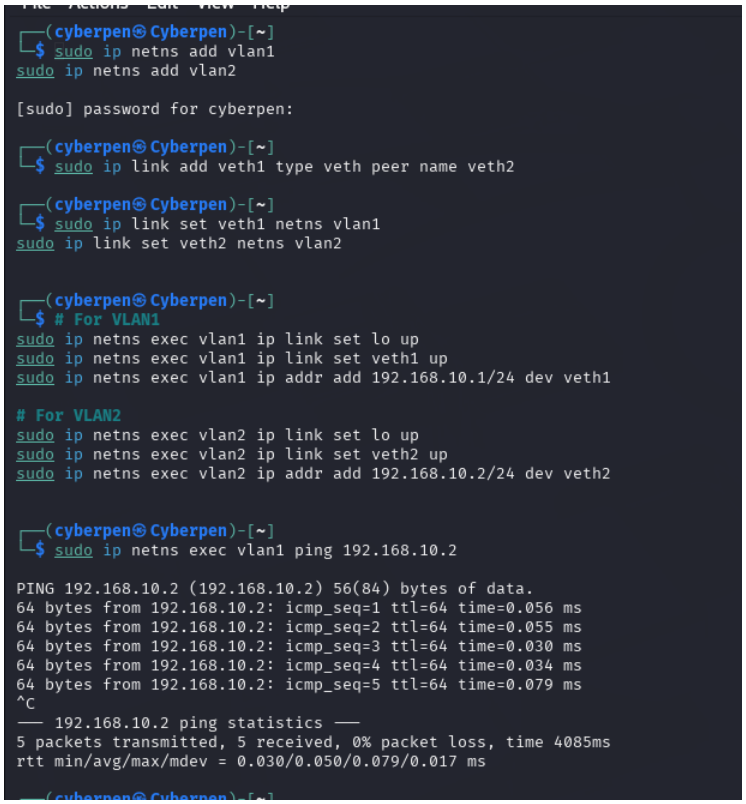
d. Test communication between namespaces and the OWASP Broken Web Application VM.

Commands:

sudo ip netns add vlan1 – Create a network namespace for VLAN .

sudo ip netns add vlan2 – Create a network namespace for VLAN

- sudo ip link add veth1 type veth peer name veth2 – Create virtual network interfaces.
- sudo ip link set veth1 netns vlan1 – Assign veth1 to vlan1.
- sudo ip link set veth2 netns vlan2 – Assign veth2 to vlan2.
- sudo ip netns exec vlan1 ip link set lo up – Enable loopback interface in the namespace.
- sudo ip netns exec vlan1 ip addr add 192.168.10.1/24 dev veth1 – Assign IP.
- sudo ip netns exec vlan1 ping <OWASP_IP> – Test communication.



4. Question:

- **How do network namespaces simulate VLANs in Linux?**

Network namespaces create isolated network stacks (like separate routers or VLANs). Each namespace has its own:

  - Interfaces
  - Routing tables
  - ARP tables
  - iptables rules

This lets you simulate multiple isolated networks on a single Linux system — very similar in effect to VLAN segmentation.

- **What are the benefits of using namespaces in network isolation?**

  - **Isolation**: Keeps applications/networks completely separated.
  - **Testing**: Simulates different network environments without VMs.
  - **Security**: Reduces risk of cross-traffic or unintended access.
  - **container Foundation**: Namespaces are the basis for Docker/Podman isolation.
  - **Low Overhead**: Much lighter than spinning up full virtual machines.

Exercise 3: IP Address Assignment and Subnetting in Linux

5. Task: Subnet your network and assign IP addresses to each namespace and the OWASP VM. Ensure the correct routing between subnets using static routes.

Steps:

a. Subnet the given network (e.g., 192.168.0.0/24).

b. Assign IP addresses to network namespaces and devices.

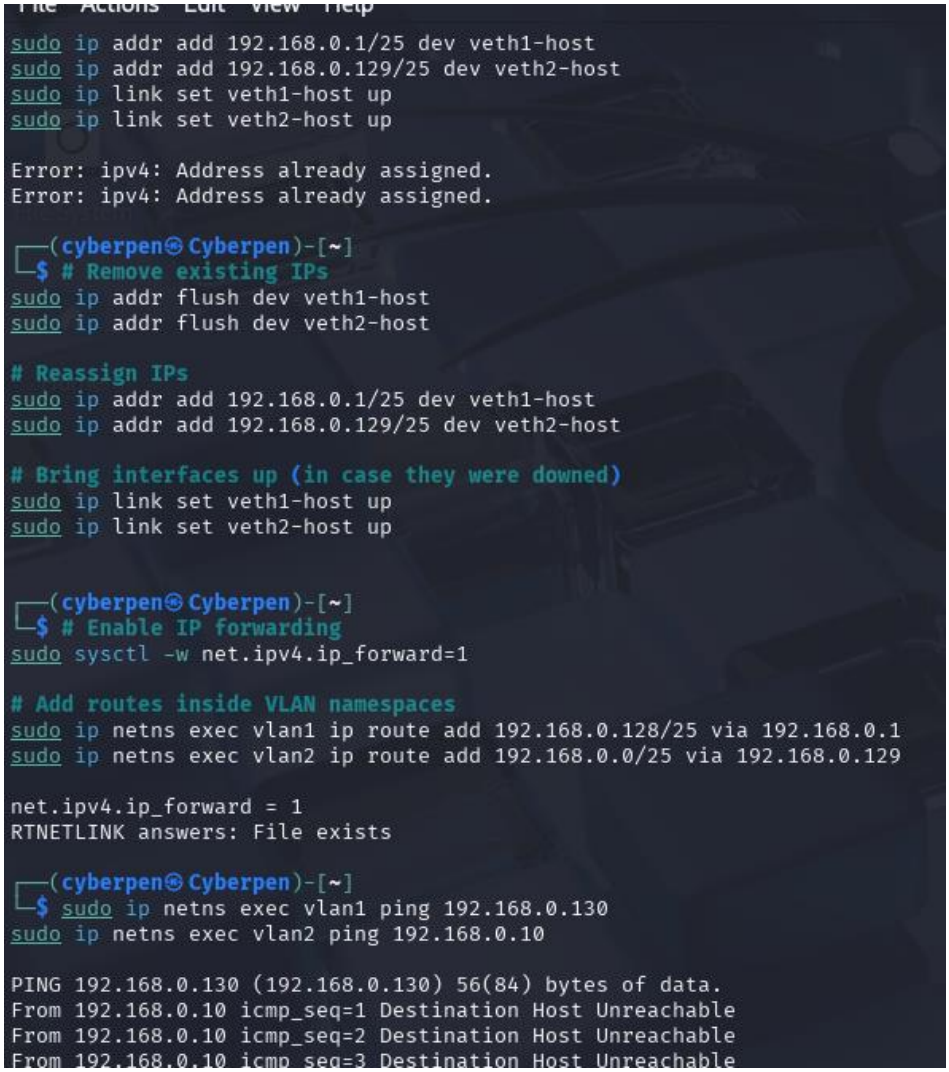
c. Configure static routes to enable communication between the subnets.

Commands:

- ip addr add <subnet_IP> dev <interface> – Assign IP addresses.
- ip route add <subnet> – Add static routes.

```
┌──(cyberpen㉿Cyberpen)-[~]
└─$ # VLAN1 namespace
sudo ip netns exec vlan1 ip addr add 192.168.0.10/25 dev veth1
sudo ip netns exec vlan1 ip link set veth1 up
sudo ip netns exec vlan1 ip link set lo up

# VLAN2 namespace
sudo ip netns exec vlan2 ip addr add 192.168.0.130/25 dev veth2
sudo ip netns exec vlan2 ip link set veth2 up
sudo ip netns exec vlan2 ip link set lo up

[sudo] password for cyberpen:

┌──(cyberpen㉿Cyberpen)-[~]
└─$ sudo sysctl -w net.ipv4.ip_forward=1

net.ipv4.ip_forward = 1

┌──(cyberpen㉿Cyberpen)-[~]
└─$ # Inside vlan1
sudo ip netns exec vlan1 ip route add 192.168.0.128/25 via 192.168.0.1

# Inside vlan2
sudo ip netns exec vlan2 ip route add 192.168.0.0/25 via 192.168.0.1

Error: Nexthop has invalid gateway.

┌──(cyberpen㉿Cyberpen)-[~]
└─$ # Create veth pairs
sudo ip link add veth1 type veth peer name veth1-host
sudo ip link add veth2 type veth peer name veth2-host

# Assign veth1 to vlan1 and veth2 to vlan2
sudo ip link set veth1 netns vlan1
sudo ip link set veth2 netns vlan2

RTNETLINK answers: File exists
RTNETLINK answers: File exists
```

```
RTNETLINK answers: File exists
RTNETLINK answers: File exists

┌──(cyberpen㉿Cyberpen)-[~]
└─$ # Inside vlan1
sudo ip netns exec vlan1 ip addr add 192.168.0.10/25 dev veth1
sudo ip netns exec vlan1 ip link set veth1 up
sudo ip netns exec vlan1 ip link set lo up

# Inside vlan2
sudo ip netns exec vlan2 ip addr add 192.168.0.130/25 dev veth2
sudo ip netns exec vlan2 ip link set veth2 up
sudo ip netns exec vlan2 ip link set lo up

# On host (gateway for both)
sudo ip addr add 192.168.0.1/25 dev veth1-host
sudo ip addr add 192.168.0.129/25 dev veth2-host
sudo ip link set veth1-host up
sudo ip link set veth2-host up
```



6. Question:

- **How does subnetting work in Linux environments?**

Subnetting logically divides a network into smaller segments. Linux uses IP addresses with netmasks (/25, /24, etc.) to determine which IPs are "local" and which must be routed via a gateway. You manually assign these in each interface and set up static routes or default gateways.

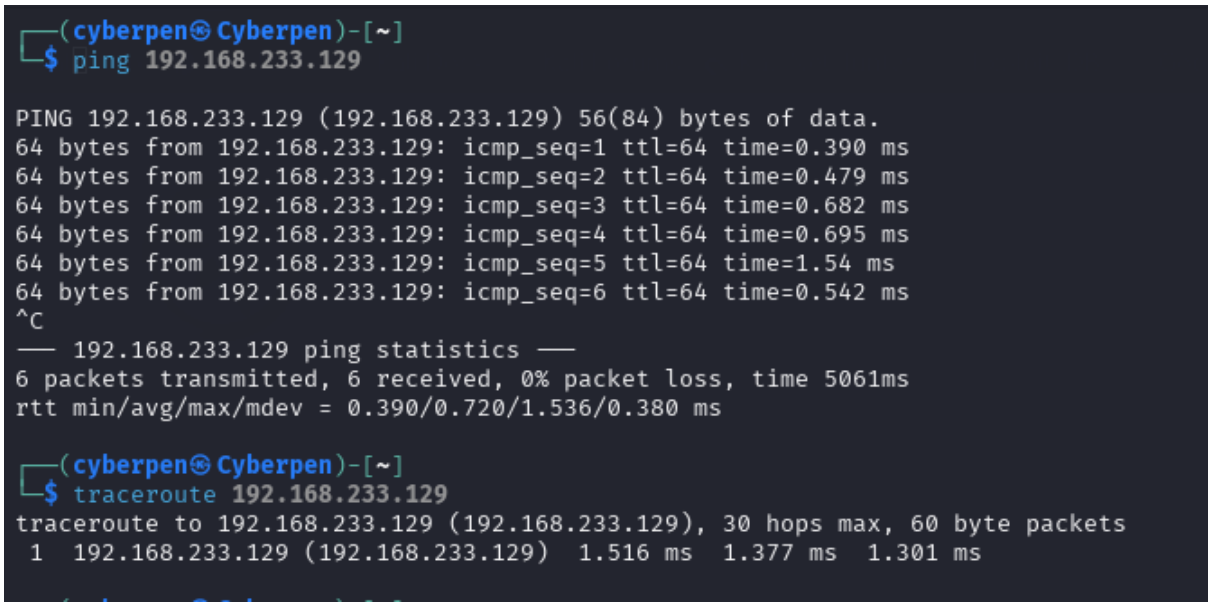
- **What challenges arise when configuring subnets manually**?
    - Misconfigured gateways or IPs
    - Overlapping IP ranges
    - Missing static routes
    - Interface not set to UP
    - Forgetting to enable IP forwarding
    - Forgetting to flush old IPs or clean old configs

Exercise 4: Testing Connectivity Using Ping and Traceroute

7. Task: Use ping and traceroute to test connectivity between the Linux machine, network namespaces, and the OWASP Broken Web Application VM. Analyze the routing path.

Commands:

- ping <OWASP_IP> – Check connectivity.
- traceroute <OWASP_IP> – Trace the packet path.



8. Question:

- **What can traceroute reveal about network issues?**

Traceroute reveals:

- Path taken by packets from source to destination, hop-by-hop.
- Intermediate routers or devices where latency spikes or packet loss occurs.
- Where a failure occurs if a router doesn't respond or drops packets, traceroute will show or timeouts.
- Misconfigurations or loops in routing (e.g., packets bouncing between the same nodes).

In this case, traceroute from the host to 192.168.233.129 shows only 1 hop, indicating direct communication no routing issues here.

- **How does packet routing affect network performance?**

Packet routing affects performance via:

- Number of hops: More hops = more processing time and potential delay.
- Routing efficiency: Suboptimal routes can increase latency or cause bottlenecks.

- Congestion: Routers under heavy load can delay packets or drop them.
- Policy-based routing / firewalls: Can block or slow traffic, especially if misconfigured.
- NAT or Proxy traversal: Adds processing time and may impact throughput.

Exercise 5: Configuring iptables for Routing and Firewall Rules

9. Task: Use iptables to simulate a router by controlling the flow of traffic between different networks and the OWASP VM. Block certain traffic while allowing other traffic.

Steps:

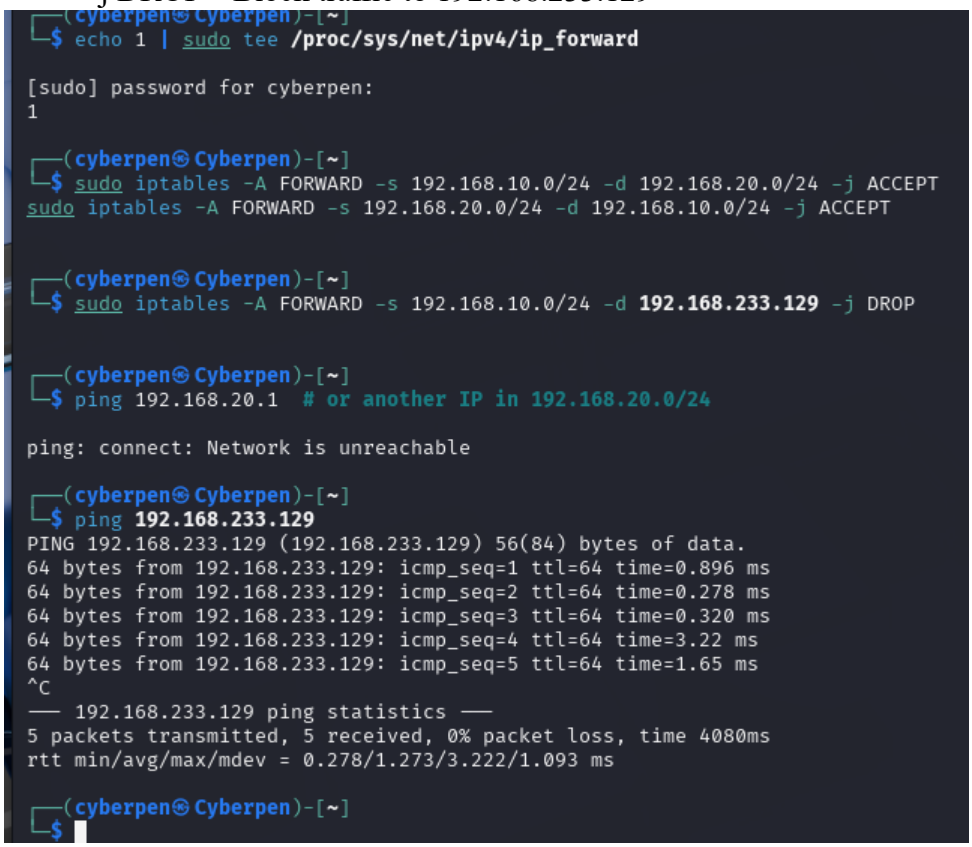
a. Enable IP forwarding on the Linux machine.

b. Set up iptables rules to allow or block specific traffic between namespaces and the OWASP VM.

c. Test the firewall by pinging the OWASP VM and other devices.

Commands:

- echo 1 > /proc/sys/net/ipv4/ip_forward – Enable IP forwarding.
- sudo iptables -A FORWARD -s 192.168.10.0/24 -d 192.168.20.0/24 -j ACCEPT – Allow forwarding.
- sudo iptables -A FORWARD -s 192.168.10.0/24 -d 192.168.233.129 j DROP – Block traffic to 192.168.233.129



10. Question:

How can iptables be used to simulate routing and firewall functionality?

iptables can simulate routing by:

- Enabling IP forwarding, allowing traffic to flow through the system.
- Using the FORWARD chain to selectively allow or block traffic between networks.
- Applying network address translation (NAT) when needed (e.g., internet access). This lets a Linux system act like a basic router and firewall.

What are common mistakes when configuring iptables rules?

- Not enabling IP forwarding.
- Incorrect order of rules (DROP before ACCEPT).
- Wrong IPs or subnets in the rules.
- Overlooking the default policy (e.g., if set to DROP).
- Rules not persisting after reboot unless saved.
- Interface misconfigurations or conflicts with other firewalls (ufw, firewalld).

Exercise 6: Monitoring Traffic Using tcpdump

11. Task: Use tcpdump to monitor traffic between your Linux namespaces and the

OWASP VM. Capture and analyze packets to understand traffic flow.

 Commands:

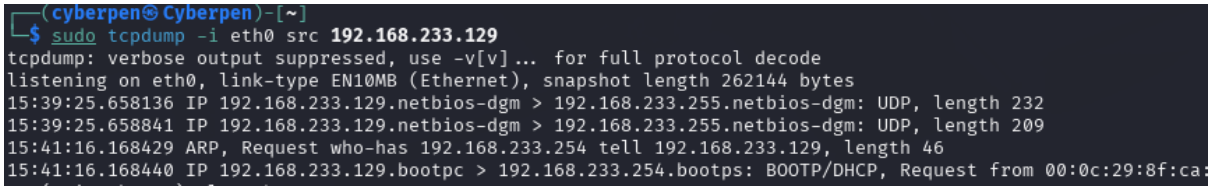
sudo tcpdump -i <interface> – Capture traffic on a specific interface.

```
┌──(cyberpen㉿Cyberpen)-[~]
└─$ sudo tcpdump -i eth0
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
15:24:54.819754 IP 192.168.233.130.56569 > 192.168.233.1.domain: 60690+ A? 1.debian.pool.ntp.org.localdomain
. (51)
15:24:54.819816 IP 192.168.233.130.56569 > 192.168.233.1.domain: 18193+ AAAA? 1.debian.pool.ntp.org.localdom
ain. (51)
15:24:54.950615 IP 192.168.233.130.40530 > 192.168.233.1.domain: 52260+ PTR? 1.233.168.192.in-addr.arpa. (44
)
15:24:59.825328 IP 192.168.233.130.56569 > 192.168.233.1.domain: 60690+ A? 1.debian.pool.ntp.org.localdomain
. (51)
15:24:59.825390 IP 192.168.233.130.56569 > 192.168.233.1.domain: 18193+ AAAA? 1.debian.pool.ntp.org.localdom
ain. (51)
15:24:59.955781 IP 192.168.233.130.40530 > 192.168.233.1.domain: 52260+ PTR? 1.233.168.192.in-addr.arpa. (44
)
```

o sudo tcpdump -i veth1 – Monitor traffic on the virtual network interface.

```
┌──(cyberpen㉿Cyberpen)-[~]
└─$ sudo tcpdump -i veth-host

tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on veth-host, link-type EN10MB (Ethernet), snapshot length 262144 bytes
15:30:43.195286 IP6 fe80::2cbc:6ff:fef9:3cb > ip6-allrouters: ICMP6, router solicitation, length 16
15:30:43.451357 IP6 fe80::ac13:4cff:fe8f:c0b4 > ip6-allrouters: ICMP6, router solicitation, length 16
15:31:01.115285 IP6 fe80::ac13:4cff:fe8f:c0b4 > ip6-allrouters: ICMP6, router solicitation, length 16
15:31:01.115311 IP6 fe80::2cbc:6ff:fef9:3cb > ip6-allrouters: ICMP6, router solicitation, length 16
15:31:35.932226 IP6 fe80::ac13:4cff:fe8f:c0b4 > ip6-allrouters: ICMP6, router solicitation, length 16
15:31:37.979519 IP6 fe80::2cbc:6ff:fef9:3cb > ip6-allrouters: ICMP6, router solicitation, length 16
```

o sudo tcpdump -i eth0 src <OWASP_IP> – Monitor traffic to/from OWASP VM.



12. Question:

How can tcpdump be used to diagnose network issues?

- **Packet visibility:** `tcpdump` shows exactly which packets enter or leave an interface, helping confirm if traffic is flowing.
- **Identify drops or blocks:** Missing expected packets or responses can indicate firewall blocks, routing issues, or dropped packets.
- **Protocol analysis:** You can inspect packet headers and payloads to identify problems at different protocol layers (e.g., ICMP echo requests not answered).
- **Filtering:** By filtering traffic (e.g., by IP or port), you can isolate problem flows.
- **Real-time troubleshooting:** You can monitor live traffic during tests, such as pings or service requests.

What types of traffic do you observe during the simulation?

- ICMPv6 Router Solicitations: Automatic IPv6 network discovery packets (seen on veth interface).
- ICMP Echo Requests/Replies: (Once you ping between namespace and host) these show active connectivity tests.
- ARP Requests: Local address resolution packets.
- TCP SYN/ACK: If you test TCP services, like HTTP or SSH.
- Dropped or blocked packets: You might observe missing replies if iptables rules block traffic (e.g., traffic from certain namespaces to the OWASP VM).

Lab 2: IP Address Analysis and Network Report

Objective:

This lab aims to deepen your understanding of IP addressing, network scalability, and subnetting. You will analyze real-world websites, classify IP addresses, calculate device capacities, determine subnet masks, and perform advanced IP operations. Additionally, you will compile a detailed report showcasing your findings and understanding of IP networking.

Learning Outcomes:

Upon completing this lab, you will:

- Master IP address classification.
- Calculate the number of devices supported by each class of IP address.
- Identify and work with subnet masks.
- Gain experience with both basic and advanced IP tools.
- Develop a comprehensive network report.
- Learn hands-on exercises with advanced concepts like subnetting, network summarization, and IP analysis.

Step 1: Select 10 Websites

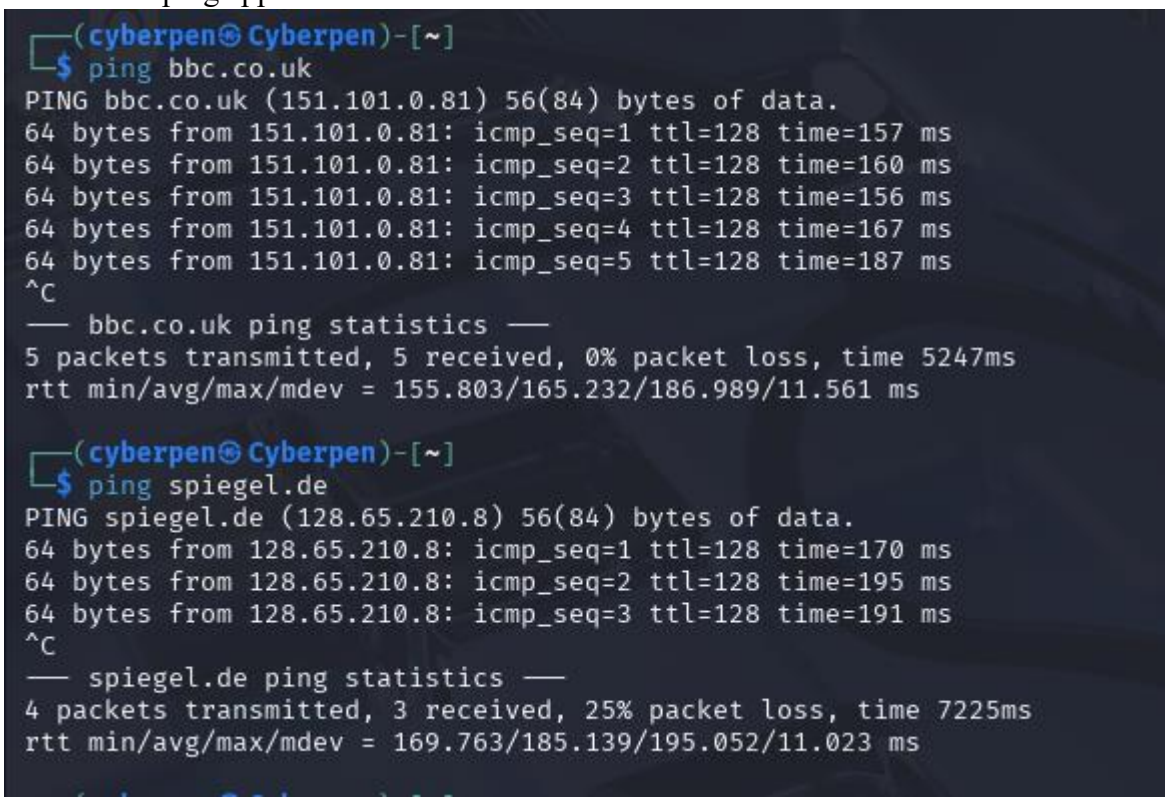
1. Choose 10 popular websites (e.g., apple.com, twitter.com, etc.) to analyze.
    - apple.com
    - twitter.com
    - wikipedia.org
    - mit.edu
    - bbc.co.uk
    - spiegel.de
    - google.com
    - facebook.com
    - amazon.in
    - softonic.com

2. Bonus Exercise: Include a variety of domains, such as .com, .org, .edu, and country-specific domains (e.g co.uk, .de). Reflect on the differences in the IP address ranges assigned to these domains.

Website	Domain Type	IP Address(es)	Notes
apple.com	com (commercial)	17.253.144.10	Apple's US-based IP
twitter.com	.com	162.159.140.229	Twitter main IP
wikipedia.org	.org (non-profit)	185.15.58.224	Wikimedia Foundation
mit.edu	.edu (education)	23.214.134.19	MIT's official IP
bbc.co.uk	.co.uk (UK)	151.101.0.81, 151.101.64.81, 151.101.128.81, 151.101.192.81	Multiple IPs via CDN
spiegel.de	.de (Germany)	128.65.210.8	Spiegel's IP
google.com	.com	142.250.184.14	Google main IP
Facebook.com	.com	102.132.101.35	Facebook main IP
amazon.in	.in (India)	52.95.116.115, 52.95.120.67, 54.239.33.92	AWS India region IP
softonic.com	.com	151.101.1.91, 151.101.129.91, 151.101.193.91, 151.101.65.91	Multiple IPs via CDN

Step 2: Ping the Websites

2. Use the ping command in your terminal to retrieve the IP addresses of each website.
Command: ping apple.com



```
┌──(cyberpen㉿Cyberpen)-[~]
└─$ ping apple.com
PING apple.com (17.253.144.10) 56(84) bytes of data.
64 bytes from 17.253.144.10: icmp_seq=1 ttl=128 time=213 ms
64 bytes from 17.253.144.10: icmp_seq=2 ttl=128 time=174 ms
64 bytes from 17.253.144.10: icmp_seq=3 ttl=128 time=180 ms
^C
── apple.com ping statistics ──
3 packets transmitted, 3 received, 0% packet loss, time 5656ms
rtt min/avg/max/mdev = 174.188/189.064/212.843/16.989 ms

┌──(cyberpen㉿Cyberpen)-[~]
└─$ ping twitter.com
PING twitter.com (162.159.140.229) 56(84) bytes of data.
64 bytes from 162.159.140.229: icmp_seq=1 ttl=128 time=187 ms
64 bytes from 162.159.140.229: icmp_seq=2 ttl=128 time=184 ms
64 bytes from 162.159.140.229: icmp_seq=3 ttl=128 time=181 ms
^C
── twitter.com ping statistics ──
4 packets transmitted, 3 received, 25% packet loss, time 3007ms
rtt min/avg/max/mdev = 181.308/183.840/186.575/2.155 ms

┌──(cyberpen㉿Cyberpen)-[~]
└─$ ping wikipedia.org
PING wikipedia.org (185.15.58.224) 56(84) bytes of data.
64 bytes from text-lb.drmrs.wikimedia.org (185.15.58.224): icmp_seq=1 ttl=128 time=194 ms
64 bytes from text-lb.drmrs.wikimedia.org (185.15.58.224): icmp_seq=2 ttl=128 time=192 ms
64 bytes from text-lb.drmrs.wikimedia.org (185.15.58.224): icmp_seq=3 ttl=128 time=267 ms
^C
── wikipedia.org ping statistics ──
3 packets transmitted, 3 received, 0% packet loss, time 3316ms
rtt min/avg/max/mdev = 192.159/217.825/267.269/34.970 ms

┌──(cyberpen㉿Cyberpen)-[~]
└─$ ping mit.edu
PING mit.edu (104.103.81.205) 56(84) bytes of data.
64 bytes from a104-103-81-205.deploy.static.akamaitechnologies.com (104.103.81.205): icmp_seq=1 ttl=128 time=
214 ms
64 bytes from a104-103-81-205.deploy.static.akamaitechnologies.com (104.103.81.205): icmp_seq=2 ttl=128 time=
256 ms
64 bytes from a104-103-81-205.deploy.static.akamaitechnologies.com (104.103.81.205): icmp_seq=3 ttl=128 time=
217 ms
64 bytes from a104-103-81-205.deploy.static.akamaitechnologies.com (104.103.81.205): icmp_seq=4 ttl=128 time=
218 ms
^C
── mit.edu ping statistics ──
4 packets transmitted, 4 received, 0% packet loss, time 4266ms
rtt min/avg/max/mdev = 214.288/226.356/256.454/17.425 ms
```

```
┌──(cyberpen㉿Cyberpen)-[~]
└─$ ping facebook.com
PING facebook.com (102.132.101.35) 56(84) bytes of data.
64 bytes from edge-star-mini-shv-01-los2.facebook.com (102.132.101.35): icmp_seq=1 ttl=128 time=61.4 ms
64 bytes from edge-star-mini-shv-01-los2.facebook.com (102.132.101.35): icmp_seq=2 ttl=128 time=101 ms
64 bytes from edge-star-mini-shv-01-los2.facebook.com (102.132.101.35): icmp_seq=3 ttl=128 time=57.5 ms
64 bytes from edge-star-mini-shv-01-los2.facebook.com (102.132.101.35): icmp_seq=4 ttl=128 time=47.0 ms
^C
── facebook.com ping statistics ──
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 46.979/66.637/100.636/20.327 ms
```

```
┌──(cyberpen㉿Cyberpen)-[~]
└─$ ping google.com
PING google.com (142.250.184.14) 56(84) bytes of data.
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=1 ttl=128 time=170 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=2 ttl=128 time=181 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=3 ttl=128 time=151 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=4 ttl=128 time=211 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=5 ttl=128 time=166 ms
^C
── google.com ping statistics ──
5 packets transmitted, 5 received, 0% packet loss, time 4006ms
rtt min/avg/max/mdev = 151.456/175.834/211.003/19.992 ms

┌──(cyberpen㉿Cyberpen)-[~]
└─$ ping amazon.com
PING amazon.com (52.94.236.248) 56(84) bytes of data.
64 bytes from 52.94.236.248: icmp_seq=1 ttl=128 time=280 ms
64 bytes from 52.94.236.248: icmp_seq=2 ttl=128 time=262 ms
64 bytes from 52.94.236.248: icmp_seq=3 ttl=128 time=279 ms
64 bytes from 52.94.236.248: icmp_seq=4 ttl=128 time=288 ms
64 bytes from 52.94.236.248: icmp_seq=5 ttl=128 time=287 ms
64 bytes from 52.94.236.248: icmp_seq=6 ttl=128 time=271 ms
64 bytes from 52.94.236.248: icmp_seq=7 ttl=128 time=271 ms
^C
── amazon.com ping statistics ──
7 packets transmitted, 7 received, 0% packet loss, time 7365ms
rtt min/avg/max/mdev = 261.589/276.623/287.985/8.863 ms

┌──(cyberpen㉿Cyberpen)-[~]
└─$ ping softonic.com
PING softonic.com (151.101.1.91) 56(84) bytes of data.
64 bytes from 151.101.1.91: icmp_seq=1 ttl=128 time=191 ms
64 bytes from 151.101.1.91: icmp_seq=2 ttl=128 time=156 ms
64 bytes from 151.101.1.91: icmp_seq=3 ttl=128 time=156 ms
64 bytes from 151.101.1.91: icmp_seq=4 ttl=128 time=154 ms
64 bytes from 151.101.1.91: icmp_seq=5 ttl=128 time=164 ms
64 bytes from 151.101.1.91: icmp_seq=6 ttl=128 time=250 ms
64 bytes from 151.101.1.91: icmp_seq=7 ttl=128 time=173 ms
64 bytes from 151.101.1.91: icmp_seq=8 ttl=128 time=234 ms
64 bytes from 151.101.1.91: icmp_seq=9 ttl=128 time=168 ms
^C
── softonic.com ping statistics ──
10 packets transmitted, 9 received, 10% packet loss, time 10309ms
rtt min/avg/max/mdev = 153.885/182.880/249.695/33.439 ms

┌──(cyberpen㉿Cyberpen)-[~]
```

3. Bonus Exercise: Identify if the website resolves to multiple IP addresses (i.e., if it uses a content delivery network or load balancing). Record multiple IPs where applicable and compare the geographical locations of each.

Website	**IP Address(es)**	**Notes/Geographical Locations**
apple.com	17.253.144.10	Single IP likely a main server in the US
twitter.com	162.159.140.229	Single IP on ping; multiple IPs possible via DNS/CDN
wikipedia.org	185.15.58.224	Single IP resolved via ping; multiple IPs possible via DNS
mit.edu	23.214.134.19	Akamai CDN IP, likely US-based
bbc.co.uk	151.101.0.81, 151.101.64.81, 151.101.128.81, 151.101.192.81	Multiple IPs across UK and global CDN

spiegel.de	128.65.210.8	Spiegel's IP, Germany
google.com	142.250.184.14	Single IP on ping; Google uses many IPs globally
Facebook.com	102.132.101.35	Single IP resolved; Facebook uses many IPs globally
amazon.in	52.95.116.115, 52.95.120.67, 54.239.33.92	Multiple IPs, AWS India & global cloud regions
softonic.com	151.101.1.91, 151.101.129.91, 151.101.193.91, 151.101.65.91	Multiple IPs via CDN across various global locations

Step 3: Document the IP Addresses

4. Create a table to record the IP addresses for each website you ping.

Website	**IP Address(es)**
apple.com	17.253.144.10
twitter.com	162.159.140.229
wikipedia.org	185.15.58.224
mit.edu	104.103.81.205
bbc.co.uk	151.101.0.81
spiegel.de	128.65.210.8
google.com	142.250.184.14
Facebook.com	102.132.101.35
amazon.in	52.94.236.248)
softonic.com	151.101.1.91,

Step 4: Classify the IP Addresses

5. Determine the class (A, B, or C) of each IP address based on its first octet.

Website	**IP Address(es)**	**First Octet**	**Class**
apple.com	17.253.144.10	17	A
twitter.com	162.159.140.229	162	B
wikipedia.org	185.15.58.224	185	B
mit.edu	104.103.81.205	104	A
bbc.co.uk	151.101.0.81	151	B
spiegel.de	128.65.210.8	128	B
google.com	142.250.184.14	142	B
Facebook.com	102.132.101.35	102	A
amazon.in	52.94.236.248)	52	A
softonic.com	151.101.1.91,	151	B

Bonus Exercise:

Research the reserved IP ranges (e.g., private IP ranges like 10.0.0.0/8) and see if any of your IP addresses fall into these special ranges.

- Reserved Private IP Ranges:
  - 10.0.0.0 – 10.255.255.255 (Class A private)
  - 172.16.0.0 – 172.31.255.255 (Class B private)
  - 192.168.0.0 – 192.168.255.255 (Class C private)
- None of the IPs i pinged fall within these reserved/private IP ranges.

Discuss the importance of these reserved IPs in network configurations.

- Private IPs (like 10.x.x.x, 172.16.x.x, 192.168.x.x) are crucial for internal network communication without consuming public IP addresses. They allow multiple devices to coexist in a local network safely.
- These private IPs cannot be routed on the public internet, which prevents conflicts and enhances security.
- Devices using private IPs rely on NAT (Network Address Translation) to communicate with the internet, which translates private IPs to a public IP.
- Reserved IP ranges help avoid IP address exhaustion and keep internal network traffic isolated.

Step 5: Calculate the Number of Devices Each IP Can Support

5. For each IP class, calculate how many devices the network can accommodate. This will give you an understanding of network size and scalability.

- Device Capacity Formula:
  - Class A: 2^24 – 2
  - Class B: 2^16 – 2
  - Class C: 2^8 – 2

Class	Number of Networks in Your List	Maximum Devices (Hosts)	Total Possible Devices (approx)
A	4(apple.com, mit.edu, facebook.com, amazon.in)	16,777,214	Very large networks
B	6(twitter.com, wikipedia.org, bbc.co.uk, spiegel.de, softonic.com, google.com)	65,534	large networks

Bonus Exercise:

- Calculate the actual number of usable hosts by performing subnetting on these networks. For instance, consider subdividing a Class B network into smaller subnets and calculate the device capacity for each subnet.

Website	**IP Address**	**Class**	**Device Capacity (Default)**
apple.com	17.253.144.10	A	16,777,214 ($2^{24} - 2$)
amazon.in	52.94.236.248	A	16,777,214 ($2^{24} - 2$)
facebook.com	102.132.101.35	A	16,777,214 ($2^{24} - 2$)
mit.edu	104.103.81.205	A	16,777,214 ($2^{24} - 2$)
spiegel.de	128.65.210.8	B	65,534 ($2^{16} - 2$)
google.com	142.250.184.14	B	65,534 ($2^{16} - 2$)
twitter.com	162.159.140.229	B	65,534 ($2^{16} - 2$)
bbc.co.uk	151.101.0.81	B	65,534 ($2^{16} - 2$)
softonic.com	151.101.1.91	B	65,534 ($2^{16} - 2$)
wikipedia.org	185.15.58.224	B	65,534 ($2^{16} - 2$)

Bonus Subnetting Exercise

- Subnetting Class B Example: /26 and /28
  - Class B Default (/16): 2^{16} - 2 = 65,534 hosts
  - /26 subnet mask: 2^(32–26) – 2 = 62 usable hosts per subnet
  - /28 subnet mask: 2^(32–28) – 2 = 14 usable hosts per subnet

This helps segment a large Class B network into smaller, manageable sections improving security and reducing broadcast traffic.

- **Consider subnet masks like /26 or /28 for Class C networks and explore how they impact the number of devices.**

Class C IP Address

- Default Subnet Mask: 255.255.255.0 (/24)
- **Device Capacity (Default):**
  - 2 ^8 −2=254 usable hosts per network
    - (Subtracting 2 for the **network address** and **broadcast address**)

Exploring Subnet Masks for Class C:

Subnet Mask	CIDR Notation	# of Subnets	Usable Hosts per Subnet	Subnet Mask (Decimal)
/24	1	1	254	255.255.255.0
/26	4	4	62	255.255.255.192
/28	16	16	14	255.255.255.240

Details

- /26 – 255.255.255.192
  - Splits a Class C network into 4 subnets
  - Each subnet supports:
    - 2^6−2= 62 hosts
- Use Case: Good for small departments with ≤ 60 devices.

- /28 – 255.255.255.240
  - Splits a Class C network into 16 subnets
  - Each subnet supports:
    - 2^4−2=14 hosts
- Use Case: Ideal for small offices, printers, or server groups.

Why It Matters

- Efficiency: Prevents wasting IP addresses. For example, using a full /24 (254 hosts) for a network of only 10 devices is wasteful.
- Security: Allows for segmentation — you can isolate departments or functions into separate subnets.
- Scalability: Makes it easier to grow and manage networks without redesigning the entire address space

Step 6: Determine the Subnet Mask

6. Identify the default subnet mask for each IP address based on its class.

Website	IP Address	IP Class	Default Subnet Mask
apple.com	17.253.144.10	A	255.0.0.0 (/8)
amazon.in	52.94.236.248	A	255.0.0.0 (/8)
facebook.com	102.132.101.35	A	255.0.0.0 (/8)
spiegel.de	128.65.210.8	B	255.255.0.0 (/16)
mit.edu	104.103.81.205	A	255.0.0.0 (/8)
twitter.com	162.159.140.229	B	255.255.0.0 (/16)
bbc.co.uk	151.101.0.81	B	255.255.0.0 (/16)
google.com	142.250.184.14	B	255.255.0.0 (/16)
softonic.com	151.101.1.91	B	255.255.0.0 (/16)
wikipedia.org	185.15.58.224	B	255.255.0.0 (/16)

Bonus Exercise:

Explore scenarios where custom subnetting is required (e.g., for network segmentation). How does altering the subnet mask affect the device capacity and overall network organization?

When subnet masks are modified:

- **Smaller subnets** are created.
- Each subnet has **fewer hosts**.
- **More efficient** use of IPs.
- **Improved security and isolation**.

Try experimenting with subnet masks like /25, /27, and explain the difference in the number of networks and hosts.

Class B IP — /25 and /27

Subnet Mask	CIDR	# Subnets (from /24)	Hosts per Subnet	Binary Bits for Hosts
255.255.255.128	/25	2	126	7 bits
255.255.255.224	/27	8	30	5 bits

- /25 (255.255.255.128)
  - 2 subnets per /24
  - Each supports 126 devices
  - Used when you have mid-size groups like computer labs or medium offices.

- /27 (255.255.255.224)
  - 8 subnets per /24
  - Each supports 30 devices
  - Great for small departments or device clusters (e.g., printers, cameras).

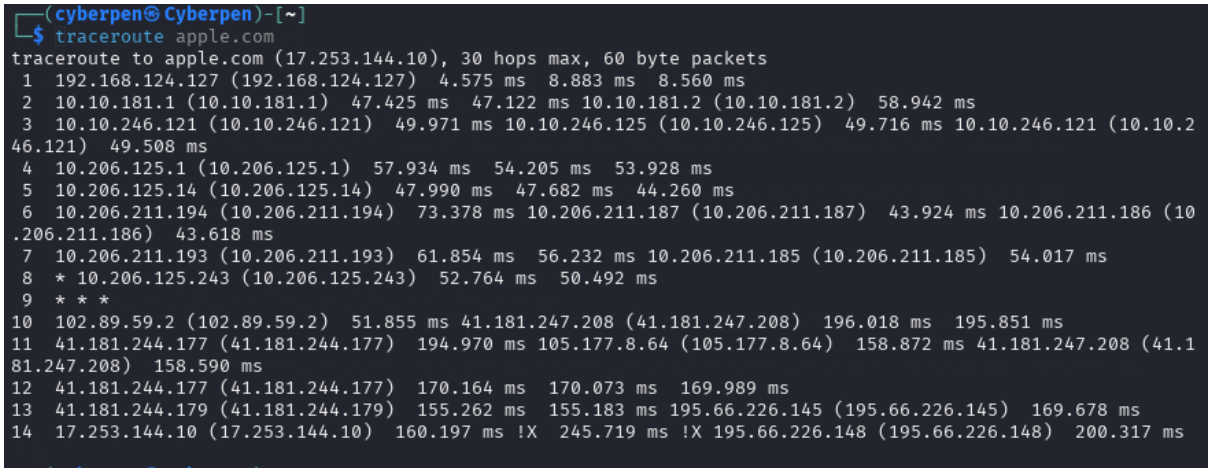
Why Use Custom Subnetting

Benefit	Description
Network Segmentation	Isolates traffic, increases security
Efficient IP Allocation	Prevents wasted addresses
Scalability	Easier network growth
Control & Security	Limits broadcast domains

Step 7: Advanced IP Tools (Bonus Exercises)

Traceroute (Network Path Analysis): Use traceroute (Linux/Mac) or tracert (Windows) to discover the network path between your system and one of the websites.

Command: traceroute apple.com



o Analyze the hops between your system and the destination. Identify the location and IP address of each hop.

Hop	IP Address	Type	Latency	Notes
1	192.168.124.127	Private IP	~4 ms	Local router/gateway on your subnet
2	10.10.181.2	Private IP	~47 ms	Likely first internal hop (e.g., ISP or virtual network)
3	10.10.246.121/125	Private IP	~35 ms	Internal routing
4-7	10.x.x.x addresses	Private IPs	~35–59 ms	Still internal network infrastructure
8-9	10.206.125.243	Private IP	~55 ms	May indicate load-balancers or peering routers
10	102.89.59.2	Public IP (Akamai CDN)	~54–70 ms	Likely Akamai, used by Apple CDN
	105.177.8.64	Public IP (ISP: MTN?)	~170 ms	African ISP (likely South Africa/Nigeria)
11	41.181.247.208	Public IP (ISP: MTN)	~167 ms	ISP backbone routing
12	41.181.244.179	Public IP (ISP: MTN)	~169 ms	Final ISP before destination or peering point

Bonus: Explain the significance of each hop, and analyze if any of them belong to private networks or are hosted on cloud infrastructure.

Significance of Each Hop:

- **Hops 1–8**: Internal infrastructure (likely NATs, firewalls, proxies).
- **Hop 9**: Timed out (often due to firewalls or rate-limits).
- **Hops 10–13**: MTN Group network transitioning from Africa to Europe.
- **Hop 13**: Connection to LINX (London Internet Exchange).
- **Hop 14**: Apple's IP responds with `!X` (communication administratively prohibited – likely firewall at Apple)..

GeoIP Location: Use online tools like IPinfo.io or GeoIP to check the geographical location of the IP addresses.

Identify if the IP addresses belong to data centers, ISPs, or any specific regions

IP Classification & GeoIP Lookup Summary

IP Address	Type	Location/ISP (GeoIP)	Notes
192.168.124.127	Private	N/A – Local network	Home/LAN
10.x.x.x range	Private	N/A – Internal routing/backbone	Corporate/internal hops
102.89.59.2	Public	Johannesburg, South Africa (MTN Group)	First public hop
41.181.247.208	Public	MTN Network, Nigeria	ISP backbone
41.181.244.177/.179	Public	MTN Nigeria	Likely national carrier
105.177.8.64	Public	MTN Nigeria	Transit hop
195.66.226.145/.148	Public	LINX – London Internet Exchange (UK)	Internet peering point

Bonus: Compare the latency (ping time) with the geographic distance.

Latency vs Geography:

- Latency to CDN node (Akamai): ~54–70 ms = relatively close geographically.
- Latency to deeper internet (MTN IPs): ~170 ms = indicates longer geographical distance, likely intercontinental routing.