Lab number 2

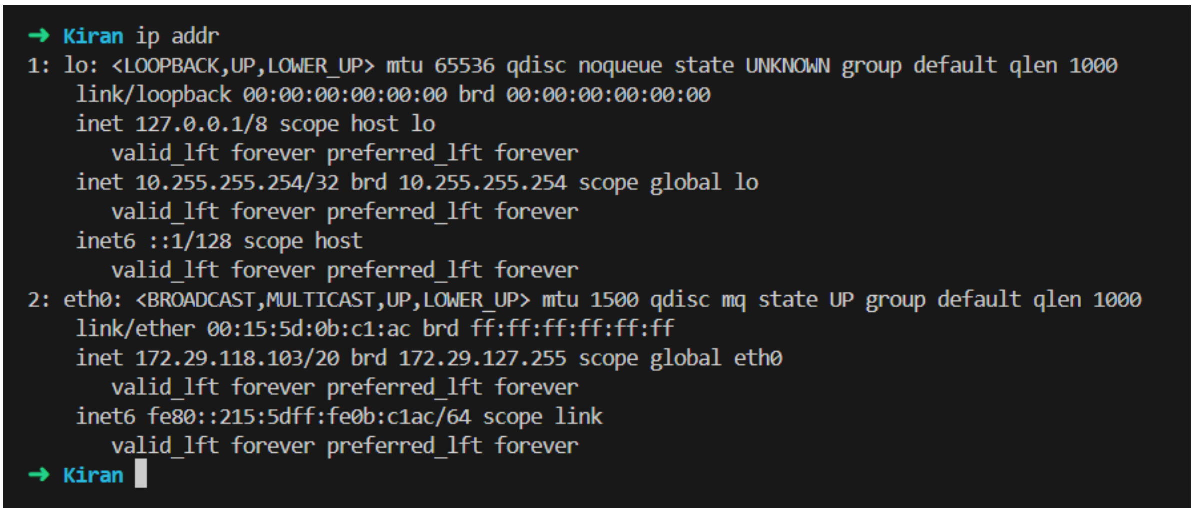
Aim: Basic Networking Commands in terminal for Linux

Overview of Basic Networking Commands

In modern computing, networks form the backbone of daily operations, enabling connectivity and data exchange between devices across local and global scales. Network administrators use various tools and commands to manage this complex infrastructure, ensuring efficiency, security, and reliability. Basic networking commands, whether on Windows, Linux, or other operating systems, are fundamental tools for diagnosing network issues, configuring network parameters, and gathering network information.

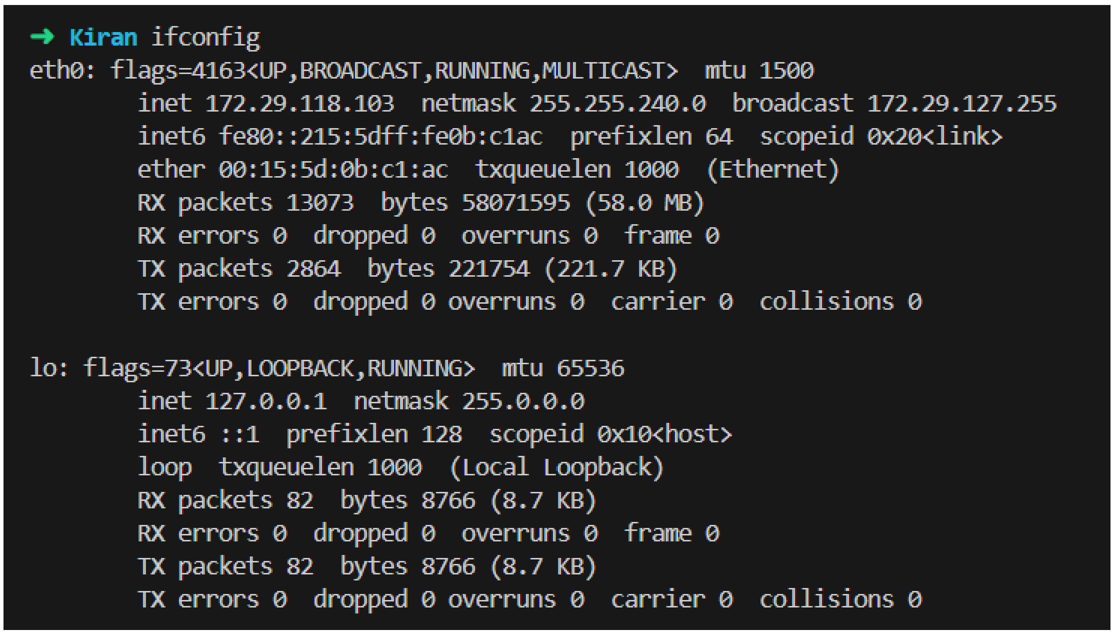
Networking Commands:

1. **ip addr**
   The "ip addr" command is used to display the IP addresses and network interfaces configured on the system. It provides detailed information about each network interface, including its IP address, MAC address, and status. Example Usage:
   $ ip addr



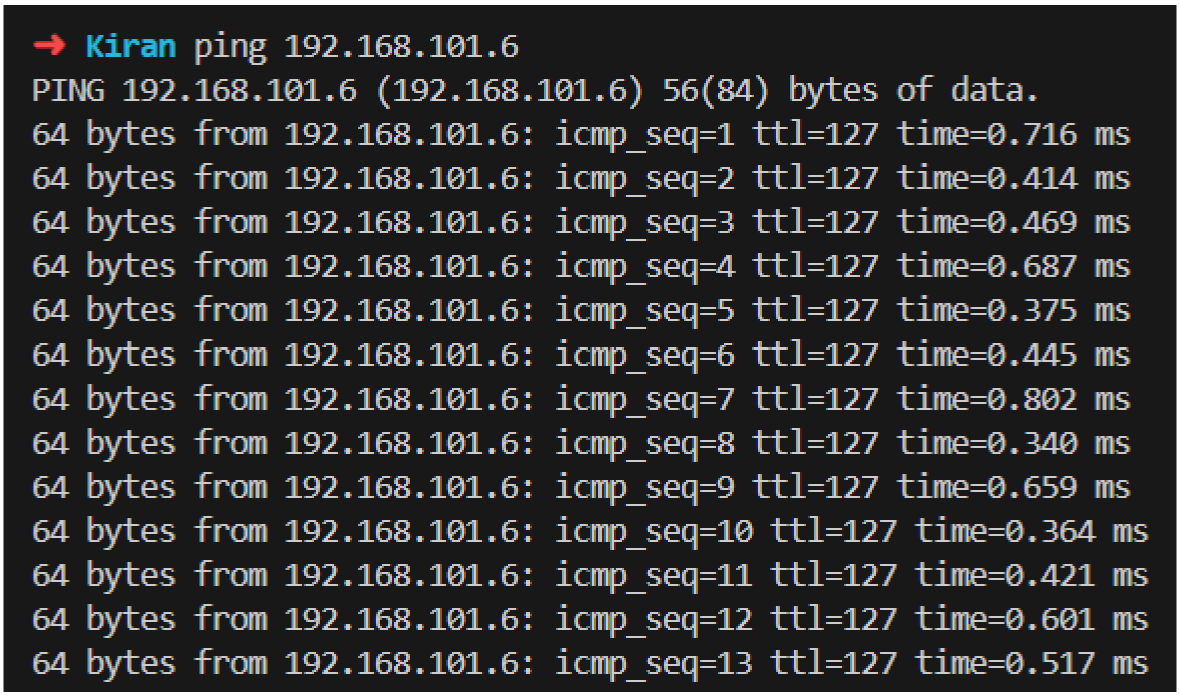
```
➜ Kiran ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet 10.255.255.254/32 brd 10.255.255.254 scope global lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:15:5d:0b:c1:ac brd ff:ff:ff:ff:ff:ff
    inet 172.29.118.103/20 brd 172.29.127.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::215:5dff:fe0b:c1ac/64 scope link
       valid_lft forever preferred_lft forever
➜ Kiran
```

2. **ifconfig**
   The "ifconfig" command is an older tool used to configure network interfaces and view their statuses. Although deprecated in some Linux distributions, it still provides useful information about IP addresses, network masks, and MAC addresses.
   Example Usage:
   $ ifconfig



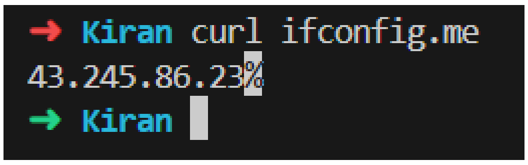
```
➜ Kiran ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.29.118.103  netmask 255.255.240.0  broadcast 172.29.127.255
        inet6 fe80::215:5dff:fe0b:c1ac  prefixlen 64  scopeid 0x20<link>
        ether 00:15:5d:0b:c1:ac  txqueuelen 1000  (Ethernet)
        RX packets 13073  bytes 58071595 (58.0 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2864  bytes 221754 (221.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 82  bytes 8766 (8.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 82  bytes 8766 (8.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

3. **ping**
   The "ping" command is used to test the connectivity between two devices. It sends ICMP echo requests to the specified IP address or hostname and waits for a reply. This command is useful for checking whether a network device is reachable.
   Example Usage:
   $ ping



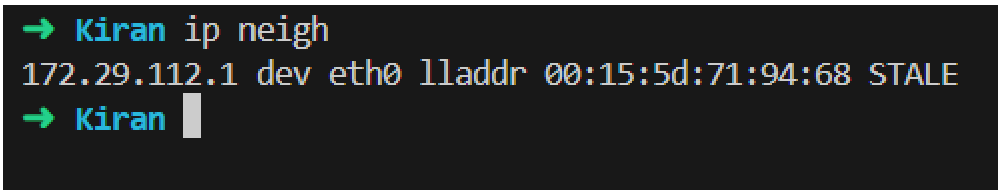
```
➜ Kiran ping 192.168.101.6
PING 192.168.101.6 (192.168.101.6) 56(84) bytes of data.
64 bytes from 192.168.101.6: icmp_seq=1 ttl=127 time=0.716 ms
64 bytes from 192.168.101.6: icmp_seq=2 ttl=127 time=0.414 ms
64 bytes from 192.168.101.6: icmp_seq=3 ttl=127 time=0.469 ms
64 bytes from 192.168.101.6: icmp_seq=4 ttl=127 time=0.687 ms
64 bytes from 192.168.101.6: icmp_seq=5 ttl=127 time=0.375 ms
64 bytes from 192.168.101.6: icmp_seq=6 ttl=127 time=0.445 ms
64 bytes from 192.168.101.6: icmp_seq=7 ttl=127 time=0.802 ms
64 bytes from 192.168.101.6: icmp_seq=8 ttl=127 time=0.340 ms
64 bytes from 192.168.101.6: icmp_seq=9 ttl=127 time=0.659 ms
64 bytes from 192.168.101.6: icmp_seq=10 ttl=127 time=0.364 ms
64 bytes from 192.168.101.6: icmp_seq=11 ttl=127 time=0.421 ms
64 bytes from 192.168.101.6: icmp_seq=12 ttl=127 time=0.601 ms
64 bytes from 192.168.101.6: icmp_seq=13 ttl=127 time=0.517 ms
```

4. **curl ifconfig.me**
The "curl ifconfig.me" command is a quick way to determine the public IP address of the system. It fetches the IP by making a request to an external web service (ifconfig.me).
Example Usage:
$ curl ifconfig.me



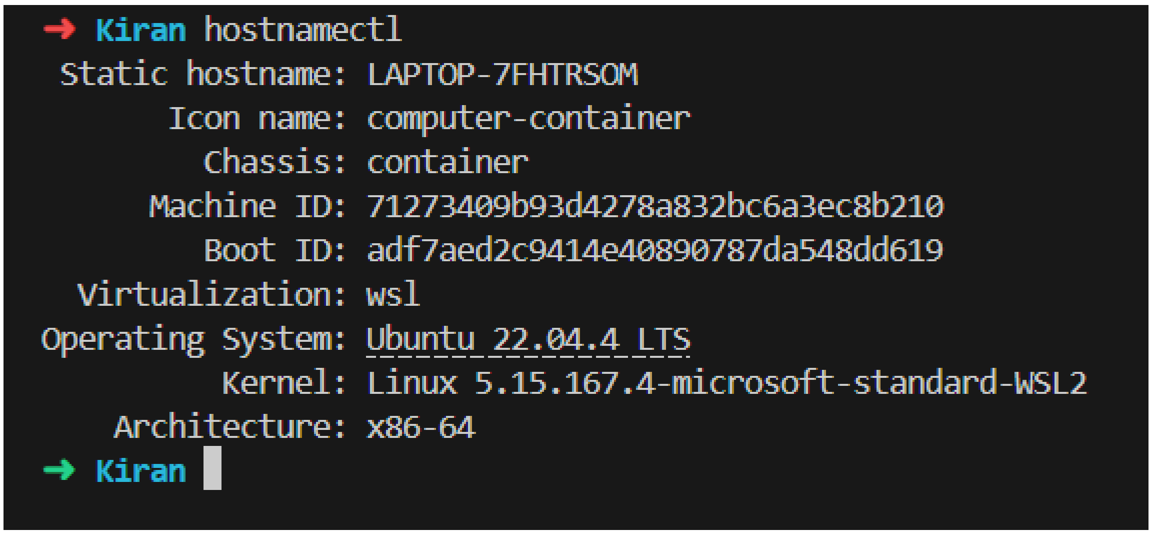
```
➜ Kiran curl ifconfig.me
43.245.86.23%
➜ Kiran
```

5. **ip neigh**
The "ip neigh" command is used to display the ARP (Address Resolution Protocol) table. It shows the IP-to-MAC address mappings for devices on the local network.
Example Usage:
$ ip neigh



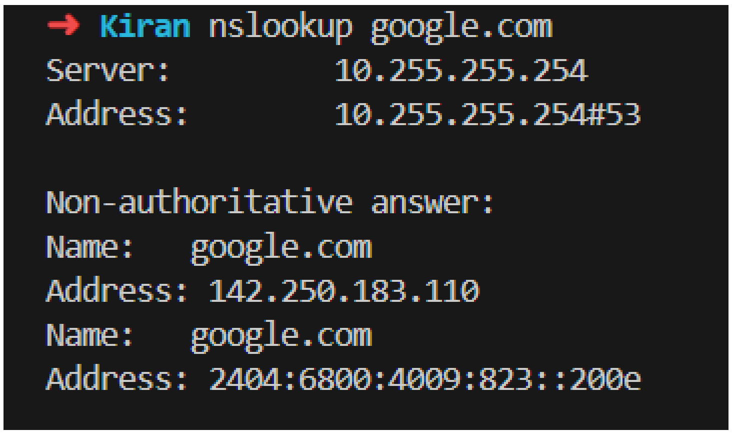
```
➜ Kiran ip neigh
172.29.112.1 dev eth0 lladdr 00:15:5d:71:94:68 STALE
➜ Kiran
```

6. **hostnamectl**
The "hostname" command prints the name of the machine along with other details. This name is used to identify the machine within a network.
Example Usage:
$ hostnamectl



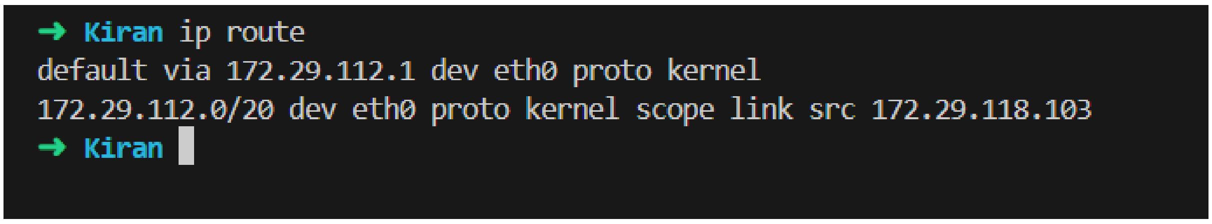
```
➜ Kiran hostnamectl
 Static hostname: LAPTOP-7FHTRSOM
       Icon name: computer-container
         Chassis: container
      Machine ID: 71273409b93d4278a832bc6a3ec8b210
         Boot ID: adf7aed2c9414e40890787da548dd619
  Virtualization: wsl
Operating System: Ubuntu 22.04.4 LTS
          Kernel: Linux 5.15.167.4-microsoft-standard-WSL2
    Architecture: x86-64
➜ Kiran
```

7. **nslookup**
The "nslookup" command is used to query DNS (Domain Name System) to obtain the IP address associated with a domain name. It is useful for troubleshooting DNS issues.
Example Usage:
$ nslookup



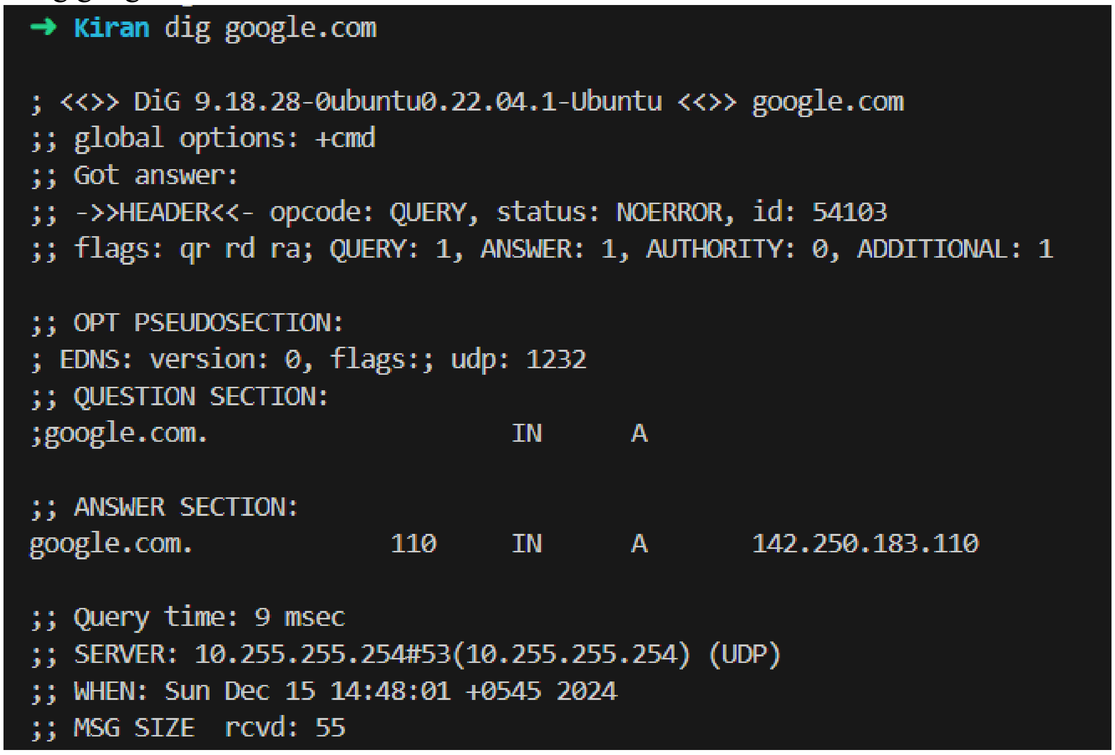
```
➜ Kiran nslookup google.com
Server:         10.255.255.254
Address:        10.255.255.254#53

Non-authoritative answer:
Name:   google.com
Address: 142.250.183.110
Name:   google.com
Address: 2404:6800:4009:823::200e
```

8. **ip route**
   The "ip route" command is used to display the routing table of the system. It shows the paths that data packets take to reach various network destinations.
   Example Usage:
   $ ip route



```
➜ Kiran ip route
default via 172.29.112.1 dev eth0 proto kernel
172.29.112.0/20 dev eth0 proto kernel scope link src 172.29.118.103
➜ Kiran
```

9. **dig**
   The dig command is used to query DNS servers for information about various Internet hosts. It's typically used to diagnose DNS problems and to get detailed DNS information.
   Example Usage:
   $ dig google.com



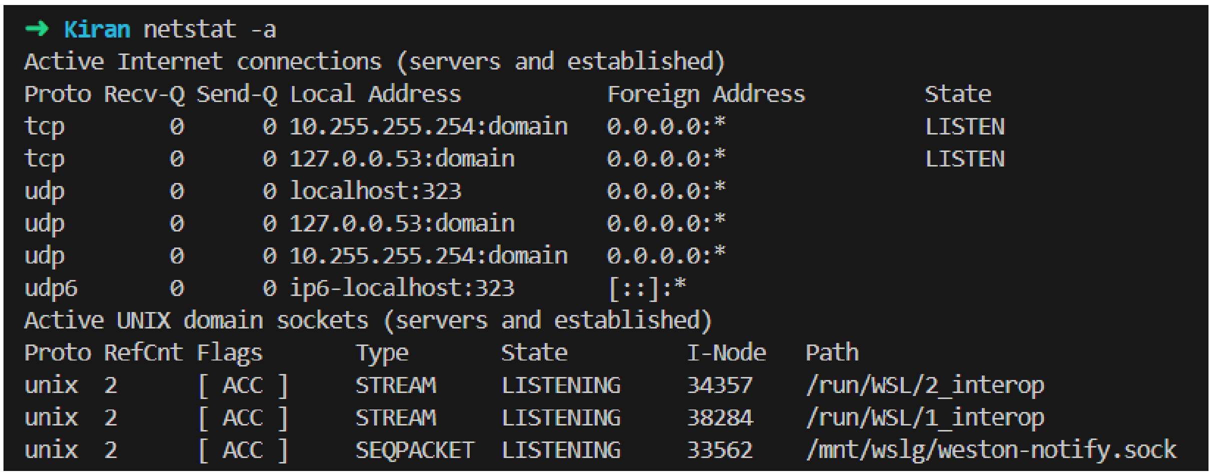
```
➜ Kiran dig google.com

; <<>> DiG 9.18.28-0ubuntu0.22.04.1-Ubuntu <<>> google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 54103
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
;; QUESTION SECTION:
;google.com.                    IN      A

;; ANSWER SECTION:
google.com.             110     IN      A       142.250.183.110

;; Query time: 9 msec
;; SERVER: 10.255.255.254#53(10.255.255.254) (UDP)
;; WHEN: Sun Dec 15 14:48:01 +0545 2024
;; MSG SIZE  rcvd: 55
```

10. netstat

The netstat command displays network connections (both incoming and outgoing), routing tables, and a number of network interface statistics. It is commonly used for checking which ports are open and which are being used by specific applications.

Example Usage:

$ netstat -a



```
➜ Kiran netstat -a
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 10.255.255.254:domain   0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.53:domain       0.0.0.0:*               LISTEN
udp        0      0 localhost:323           0.0.0.0:*
udp        0      0 127.0.0.53:domain       0.0.0.0:*
udp        0      0 10.255.255.254:domain   0.0.0.0:*
udp6       0      0 ip6-localhost:323       [::]:*
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ACC ]     STREAM     LISTENING     34357    /run/WSL/2_interop
unix  2      [ ACC ]     STREAM     LISTENING     38284    /run/WSL/1_interop
unix  2      [ ACC ]     SEQPACKET  LISTENING     33562    /mnt/wslg/weston-notify.sock
```

Conclusion

In this LAB, we delved into essential networking commands within Linux, learning how to effectively manage and troubleshoot networks. By practicing with commands like ip, ping, dig, and netstat, I gained hands-on skills crucial for understanding network configurations and addressing issues efficiently.

Discussion

The LAB was invaluable in demonstrating the practical use of these commands. I observed firsthand how tools like ping assess connectivity and netstat monitors active connections, which are essential for maintaining network health. The move from ifconfig to ip also highlighted the importance of keeping current with evolving network tools. This experience has equipped me with foundational skills to confidently handle real-world network challenges.