

## Basic Networking Commands in Linux operating systems


**Exp No:1B**

**Date: 30.7.2024**

**AIM: - Study of various Network commands used in Linux**

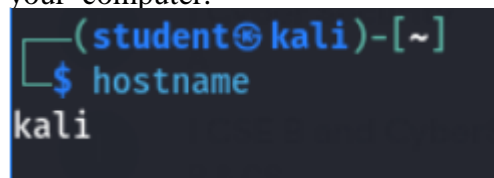
### BASIC NETWORKING COMMANDS:

**arp -a:** ARP is short form of address resolution protocol, It will show the IP address of your computer along with the IP address and MAC address of your router.



```
student@kali:~$ arp -a
? (172.16.52.166) at 88:ae:dd:15:ed:2b [ether] on eth0
? (172.16.52.180) at 88:ae:dd:15:ed:0e [ether] on eth0
? (172.16.52.142) at 1c:69:7a:ff:53:3d [ether] on eth0
? (172.16.52.175) at 88:ae:dd:15:ed:a8 [ether] on eth0
? (172.16.52.267) at 88:ae:dd:15:ed:97 [ether] on eth0
? (172.16.52.151) at 88:ae:dd:15:ed:1e [ether] on eth0
? (172.16.52.143) at 88:ae:dd:15:ed:c8 [ether] on eth0
? (172.16.52.168) at 88:ae:dd:15:ed:2a [ether] on eth0
? (172.16.53.195) at 88:ae:dd:14:72:38 [ether] on eth0
? (172.16.53.253) at 88:ae:dd:15:ed:56 [ether] on eth0
? (172.16.52.144) at 88:ae:dd:15:ed:49 [ether] on eth0
? (172.16.52.169) at 88:ae:dd:15:ed:ec [ether] on eth0
? (172.16.53.171) at 88:ae:dd:15:ed:b8 [ether] on eth0
? (172.16.52.183) at 88:ae:dd:15:ed:27 [ether] on eth0
? (172.16.53.188) at 88:ae:dd:15:ed:8d [ether] on eth0
? (172.16.52.181) at 88:ae:dd:12:ed:31 [ether] on eth0
? (172.16.52.153) at 88:ae:dd:15:ed:7c [ether] on eth0
? (172.16.53.168) at 88:ae:dd:15:ed:cd [ether] on eth0
? (172.16.52.145) at 88:ae:dd:15:ed:91 [ether] on eth0
? (172.16.52.178) at 88:ae:dd:15:ed:5c [ether] on eth0
? (172.16.53.173) at 88:ae:dd:15:ed:b9 [ether] on eth0
? (172.16.53.115) at 88:ae:dd:15:ed:1c [ether] on eth0
? (172.16.52.146) at 88:ae:dd:12:7e:13 [ether] on eth0
? (172.16.52.171) at 88:ae:dd:15:ed:38 [ether] on eth0
? (172.16.52.172) at c8:d3:ff:0d:1e:2d [ether] on eth0
? (172.16.52.196) at 2c:78:8a:84:a2:d7 [ether] on eth0
? (172.16.53.190) at 88:ae:dd:15:ed:8b [ether] on eth0
? (172.16.53.182) at 88:ae:dd:15:ed:9b [ether] on eth0
? (172.16.52.147) at 88:ae:dd:15:ed:17 [ether] on eth0
? (172.16.52.172) at 88:ae:dd:15:ed:c5 [ether] on eth0
? (172.16.53.207) at d4:3d:7e:c7:ab:c5 [ether] on eth0
? (172.16.52.280) at 88:ae:dd:15:ed:48 [ether] on eth0
? (172.16.52.164) at 88:ae:dd:15:ed:02 [ether] on eth0
? (172.16.53.249) at 88:ae:dd:15:ed:5b [ether] on eth0
? (172.16.52.248) at 88:ae:dd:15:ed:1a [ether] on eth0
? (172.16.52.173) at 88:ae:dd:15:ed:f3 [ether] on eth0
? (172.16.53.48) at 88:ae:dd:15:ed:5e [ether] on eth0
? (172.16.52.185) at 88:ae:dd:14:6d:f6 [ether] on eth0
? (172.16.52.149) at 88:ae:dd:15:ed:c7 [ether] on eth0
? (172.16.52.188) at 88:ae:dd:15:ed:1a [ether] on eth0
? (172.16.52.141) at 88:ae:dd:15:ed:ca [ether] on eth0
```

**hostname:** This is the simplest of all TCP/IP commands. It simply displays the name of your computer.



```
student@kali:~$ hostname
kali
```

**ipconfig /all:** This command displays detailed configuration information about your TCP/IP connection including Router, Gateway, DNS, DHCP, and type of Ethernet adapter in your system

**nbtstat -a:** This command helps solve problems with NetBIOS name resolution. (Nbt stands for NetBIOS over TCP/IP)

**netstat:** (network statistics) netstat displays a variety of statistics about a computer's active TCP/IP connections. It is a command line tool for monitoring network connections both incoming and outgoing as well as viewing routing tables, interface statistics etc. e.g.:- netstat -r

```

student@kali: ~
$ netstat -ttn
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 kali:55594             93.243.107.34.bc.:https ESTABLISHED
tcp        0      0 kali:52302             maa05s20-in-f5.1e:https ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.142:57226     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.171:46808     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.172:41444     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.148:53856     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.166:43218     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.164:51924     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.145:41456     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.168:44144     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.175:46438     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.149:57886     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.170:40720     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.150:35562     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.146:38238     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.151:39120     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.141:59982     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.169:37584     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.144:36528     ESTABLISHED
tcp6       0      0 172.16.52.152:35034     172.16.52.162:1716      ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.173:39308     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.147:59646     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.165:52082     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.153:39906     ESTABLISHED
tcp6       0      0 172.16.52.152:1716     172.16.52.143:53538     ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type       State      I-Node    Path
unix    3      [ ]         STREAM     CONNECTED  18883
unix    2      [ ]         DGRAM      CONNECTED  15065

```

**nslookup:** (name server lookup) is a tool used to perform DNS lookups in Linux. It is used to display DNS details, such as the IP address of a particular computer, the MX records for a domain or the NS servers of a domain. nslookup can operate in two modes: interactive and non-interactive.

e.g.:- nslookup [www.google.com](http://www.google.com)

```

(student@kali)-[~]
$ nslookup www.google.com
Server:      172.16.52.1
Address:     172.16.52.1#53

Non-authoritative answer:
Name:   www.google.com
Address: 142.250.182.4
Name:   www.google.com
Address: 2404:6800:4007:819::2004

```

**pathping:** Pathping is unique to Windows, and is basically a combination of the Ping and Tracert commands. Pathping traces the route to the destination address then launches a 25 second test of each router along the way, gathering statistics on the rate of data loss along each hop.

**ping:** (Packet INternet Groper) command is the best way to test connectivity between two nodes. Ping use ICMP (Internet Control Message Protocol) to communicate to other devices.

1. #ping hostname( ping localhost)
2. #ping ip address (ping 4.2.2.2)
3. #ping fully qualified domain name(ping [www.google.com](http://www.google.com))

```
(student@kali)-[~]
$ ping www.google.com
PING www.google.com (142.250.182.4) 56(84) bytes of data.
64 bytes from maa05s18-in-f4.1e100.net (142.250.182.4): icmp_seq=1 ttl=120 time=3.47 ms
64 bytes from maa05s18-in-f4.1e100.net (142.250.182.4): icmp_seq=2 ttl=120 time=3.93 ms
64 bytes from maa05s18-in-f4.1e100.net (142.250.182.4): icmp_seq=3 ttl=120 time=3.77 ms
64 bytes from maa05s18-in-f4.1e100.net (142.250.182.4): icmp_seq=4 ttl=120 time=3.57 ms
64 bytes from maa05s18-in-f4.1e100.net (142.250.182.4): icmp_seq=5 ttl=120 time=3.67 ms
64 bytes from maa05s18-in-f4.1e100.net (142.250.182.4): icmp_seq=6 ttl=120 time=3.55 ms
64 bytes from maa05s18-in-f4.1e100.net (142.250.182.4): icmp_seq=7 ttl=120 time=3.68 ms
64 bytes from maa05s18-in-f4.1e100.net (142.250.182.4): icmp_seq=8 ttl=120 time=3.94 ms
^C
--- www.google.com ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7013ms
rtt min/avg/max/mdev = 3.468/3.696/3.939/0.161 ms
(student@kali)-[~]
```

**Route:** route command is used to show/manipulate the IP routing table. It is primarily used to setup static routes to specific host or networks via an interface.

## Some important Linux networking commands

### 1. ip

---

The ip command is one of the basic commands every administrator will need in daily work, from setting up new systems and assigning IPs to troubleshooting existing systems. The ip command can show address information, manipulate routing, plus display network various devices, interfaces, and tunnels.

**ip <OPTIONS> <OBJECT> <COMMAND>**

Here are some common use cases for the ip command.

- a. To show the IP addresses assigned to an interface on your server:  
a. [root@server ~]# *ip address show*
- b. To assign an IP to an interface, for example, **enps03**:  
a. [root@server ~]# ip address add 192.168.1.254/24 dev enps03
- c. To delete an IP on an interface:  
a. [root@server ~]# ip address del 192.168.1.254/24 dev enps03
- d. Alter the status of the interface by bringing the interface **eth0** online:  
[root@server ~]# ip link set eth0 up
- e. Alter the status of the interface by bringing the interface **eth0** offline:  
[root@server ~]# ip link set eth0 down
- f. Alter the status of the interface by enabling promiscuous mode for **eth0**:  
[root@server ~]# ip link set eth0 promisc on
- g. Add a default route (for all addresses) via the local gateway 192.168.1.254 that can be reached on device **eth0**:  
[root@server ~]# ip route add default via 192.168.1.254 dev eth0
- h. Add a route to 192.168.1.0/24 via the gateway at 192.168.1.254:  
[root@server ~]# ip route add 192.168.1.0/24 via 192.168.1.254
- i. Add a route to 192.168.1.0/24 that can be reached on device **eth0**:  
[root@server ~]# ip route add 192.168.1.0/24 dev eth0
- j. Delete the route for 192.168.1.0/24 via the gateway at 192.168.1.254:  
[root@server ~]# ip route delete 192.168.1.0/24 via 192.168.1.254
- k. Display the route taken for IP 10.10.1.4:  
[root@server ~]# ip route get 10.10.1.4

## 2. ifconfig

The ifconfig command was/is a staple in many sysadmin's tool belt for configuring and troubleshooting networks. It has since been replaced by the ip command discussed above.

```
(student@kali)-[~]  
$ ifconfig  
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 172.16.52.152 netmask 255.255.252.0 broadcast 172.16.55.255  
    inet6 fe80::8aae:ddff:fe15:db24 prefixlen 64 scopeid 0x20<link>  
    ether 88:ae:dd:15:db:24 txqueuelen 1000 (Ethernet)  
    RX packets 100941 bytes 67340498 (64.2 MiB)  
    RX errors 0 dropped 2214 overruns 0 frame 0  
    TX packets 38235 bytes 10458587 (9.9 MiB)  
    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 25 bytes 1520 (1.4 KiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 25 bytes 1520 (1.4 KiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

### 3. mtr

---

MTR (Matt's traceroute) is a program with a command-line interface that serves as a network diagnostic and troubleshooting tool. This command combines the functionality of the ping and traceroute commands. Just like a traceroute, the mtr command will show the route from a computer to a specified host. mtr provides a lot of statistics about each hop, such as response time and percentage. With the mtr command, you will get more information about the route and be able to see problematic devices along the way. If you see a sudden increase in response time or packet loss, then obviously, there is a bad link somewhere.

The syntax of the command is as follows:

**mtr <options> hostname/IP**

Let's look at some common use cases.

- a. The basic mtr command shows you the statistics, including each hop (hostnames) with time and loss%:

```
[root@server ~]# mtr google.com
```

- b. Show numeric IP addresses (if you use -g, you will get IP addresses (numbers) instead of hostnames):

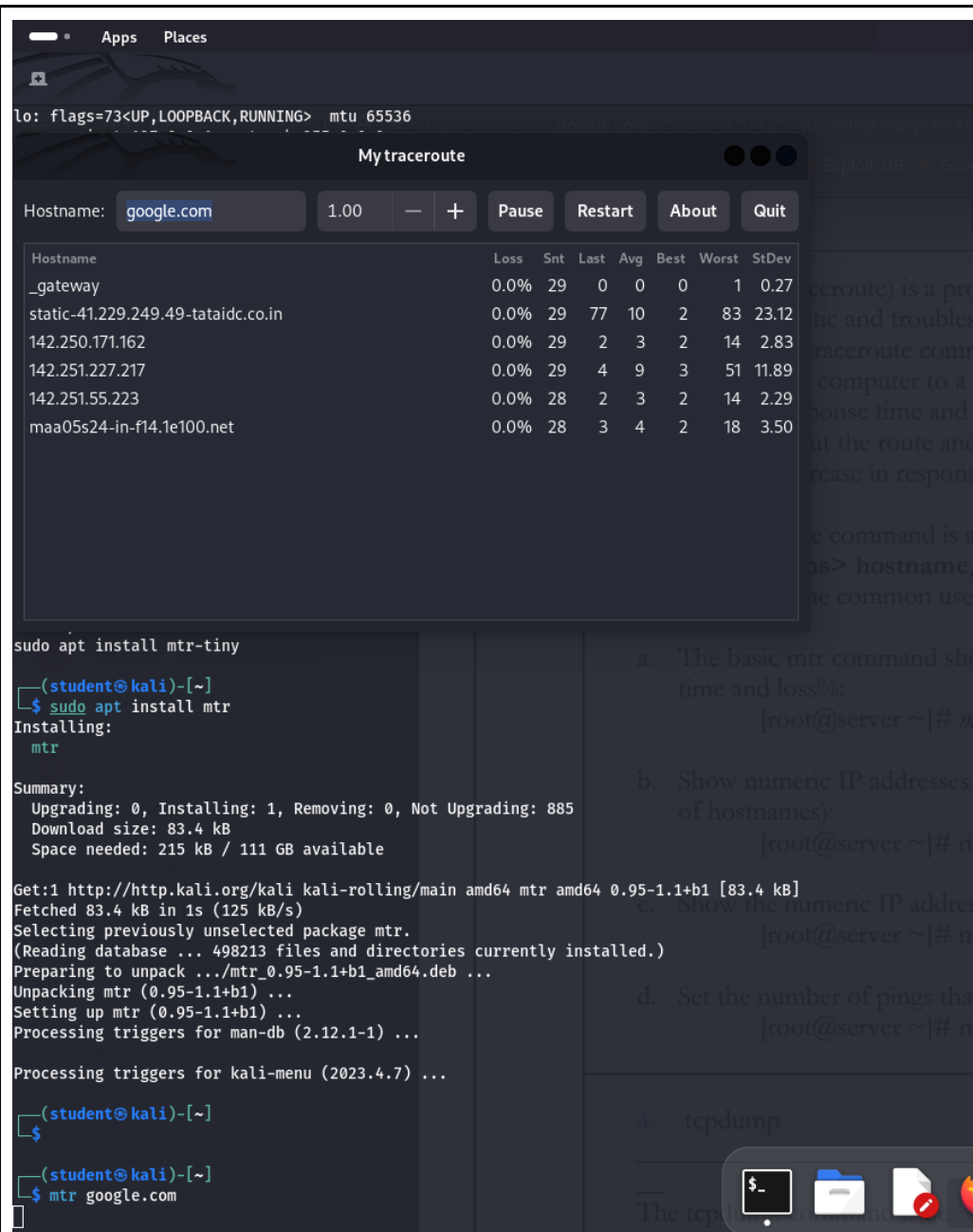
```
[root@server ~]# mtr -g google.com
```

- c. Show the numeric IP addresses and hostnames, too:

```
[root@server ~]# mtr -b google.com
```

- d. Set the number of pings that you want to send:

```
[root@server ~]# mtr -c 10 google.com
```



#### 4. tcpdump

The tcpdump command is designed for capturing and displaying packets.

You can install tcpdump with the command below:

```
[root@server ~]# dnf install -y tcpdump
```

Before starting any capture, you need to know which interfaces tcpdump can use.

You will need to use sudo or have root access in this case.

```
[root@server ~]# tcpdump -D
```

If you want to capture traffic on **eth0**, you can initiate that with `tcpdump -i eth0`

sample output:

```
[root@server ~]# tcpdump -i eth0
```

```
[root@server ~]# tcpdump -i eth0 -c 10
```

### **Capture traffic to and from one host**

You can filter out traffic coming from a specific host. For example, to find traffic coming from and going to 8.8.8.8, use the command:

```
[root@server ~]# tcpdump -i eth0 -c 10 host 8.8.8.8
```

For traffic coming from 8.8.8.8, use:

```
[root@server ~]# tcpdump -i eth0 src host 8.8.8.8
```

For outbound traffic going to 8.8.8.8, use:

```
[root@server ~]# tcpdump -i eth0 dst host 8.8.8.8
```

### **Capture traffic to and from a network**

You can also capture traffic to and from a specific network using the command below: [root@server ~]# tcpdump -i eth0 net 10.1.0.0 mask 255.255.255.0

or:

```
[root@server ~]# tcpdump -i eth0 net 10.1.0.0/24
```

### **Capture traffic to and from port numbers**

Capture only DNS port 53 traffic:

```
[root@server ~]# tcpdump -i eth0 port 53
```

For a specific host,

```
[root@server ~]# tcpdump -i eth0 host 8.8.8.8 and port 53
```

To capture only HTTPS traffic,

```
[root@server ~]# tcpdump -i eth0 -c 10 host www.google.com and port 443
```

To capture all port except port 80 and 25,

```
[root@server ~]# tcpdump -i eth0 port not 53 and not 25
```



## 5. ping

Ping is a tool that verifies IP-level connectivity to another TCP/IP computer by sending Internet Control Message Protocol (ICMP) Echo Request messages. The receipt of corresponding Echo Reply messages is displayed, along with round-trip times. Ping is the primary TCP/IP command used to troubleshoot connectivity, reachability, and name resolution.

```
[root@server ~]# ping google.com
PING google.com (216.58.206.174) 56(84) bytes of data:
64 bytes from sof02s27-in-f14.1e100.net (216.58.206.174): icmp_seq=1 ttl=56
time=10.7 ms
64 bytes from sof02s27-in-f14.1e100.net (216.58.206.174): icmp_seq=2 ttl=56
time=10.2 ms
64 bytes from sof02s27-in-f14.1e100.net (216.58.206.174): icmp_seq=3 ttl=56
time=10.4 ms
^C
```

You need to stop the ping command by pressing **CTRL+C**. Otherwise, it will ping until you stop it.

If you want to ping a host ten times, use the following command:

```
[root@server ~]# ping -c 10 google.com
```

While pinging a host, you'll find different output from the ping results, including the following three examples.

### **Destination Host Unreachable**

The possible best reason is there is no route from the local host system and to the destination desired destination host, or a remote router reports that it has no route to the destination host.

### **Request timed out**

This result means that no Echo Reply messages were received within the default time of one second or the time that you set while you are pinging that host. This can be due to many different causes; the most common include network congestion, failure of the ARP request, packet filtering/firewall, etc.

### **Unknown host/Ping Request Could Not Find Host**

Maybe you misspelled the hostname or the host does not exist at all in the network.

You must have 0% packet loss for every ping result with a good latency or lower response time. Depending on which transmission medium (UTP, fibre optics cable, Wi Fi) you're using, your latency will differ.

## Configuring an Ethernet connection by using nmcli

If you connect a host to the network over Ethernet, you can manage the connection's settings on the command line by using the **nmcli** utility.

### Procedure

1. List the NetworkManager connection profiles:

```
# nmcli connection show
```

```
NAME UUID TYPE DEVICE Wired connection 1 a5eb6490-cc20-3668-81f8-0314a27f3f75 ethernet enp1s0
```

2. 

```
# nmcli connection add con-name <connection-name> ifname <device-name> type ethernet
```

  
Skip this step to modify an existing profile.

3. Optional: Rename the connection profile:

```
# nmcli connection modify "Wired connection 1"
```

Here, "Wired connection 1" is the name of the connection

4. Display the current settings of the connection profile:

```
# nmcli connection show
```

```
connection.interface-name: enp1s0
```

```
connection.autoconnect: yes
```

```
ipv4.method: auto
```

```
ipv6.method: auto
```

```
...
```

5. Configure the IPv4 settings:

- To use DHCP, enter:

```
# nmcli connection modify "Wired connection 1" ipv4.method auto
```

Skip this step if `ipv4.method` is already set to `auto` (default).

- To set a static IPv4 address, network mask, default gateway, DNS servers, and search domain, enter:

```
# nmcli connection modify "Wired connection 1" ipv4.method manual
ipv4.addresses 192.0.2.1/24 ipv4.gateway 192.0.2.254 ipv4.dns
192.0.2.200 ipv4.dns-search example.com
```

6. Configure the IPv6 settings:

- To use stateless address autoconfiguration (SLAAC), enter:

```
# nmcli connection modify "Wired connection 1" ipv6.method auto
```

Skip this step if `ipv6.method` is already set to `auto` (default).

- To set a static IPv6 address, network mask, default gateway, DNS servers, and search domain, enter:

```
# nmcli connection modify "Wired connection 1" ipv6.method manual
ipv6.addresses 2001:db8:1::fffe/64 ipv6.gateway 2001:db8:1::fffe
ipv6.dns 2001:db8:1::ffbb ipv6.dns-search example.com
```

7. Activate the profile:

```
# nmcli connection up Internal-LAN
```

## Verification

1. Display the IP settings of the NIC:

```
# ip address show enp1s0
enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
fq_codel state UP group default qlen 1000
link/ether 52:54:00:17:b8:b6 brd ff:ff:ff:ff:ff:ff
inet 192.0.2.1/24 brd 192.0.2.255 scope global noprefixroute enp1s0
    valid_lft forever preferred_lft forever
inet6 2001:db8:1::fffe/64 scope global noprefixroute
    valid_lft forever preferred_lft forever
```

2. Display the IPv4 default gateway:

```
# ip route show default
```

```
default via 192.0.2.254 dev enp1s0 proto static metric 102
```

3. Display the IPv6 default gateway:

```
# ip -6 route show default
```

```
default via 2001:db8:1::fffe dev enp1s0 proto static metric 102 pref
```

medium 4. Display the DNS settings:

```
# cat /etc/resolv.conf
```

```
search example.com
```

```
nameserver 192.0.2.200
```

```
nameserver 2001:db8:1::ffbb
```

If multiple connection profiles are active at the same time, the order of nameserver entries depend on the DNS priority values in these profile and the connection types.

5. Use the ping utility to verify that this host can send packets to other

```
hosts: # ping <host-name-or-IP-address>
```

## Troubleshooting

- Verify that the network cable is plugged-in to the host and a switch.
- Check whether the link failure exists only on this host or also on other hosts connected to the same switch.
- Verify that the network cable and the network interface are working as expected. Perform hardware diagnosis steps and replace defect cables and network interface cards.
- If the configuration on the disk does not match the configuration on the device, starting or restarting NetworkManager creates an in-memory connection that reflects the configuration of the device.

Result :

Thus , the basic networking commands in Linux are studied.